Oracle® Big Data Spatial and Graph

User's Guide and Reference

Release 3.0

F20636-02
December 2023

Oracle Big Data Spatial and Graph User’s Guide and Reference, Release 3.0

F20636-02

Primary Author: Lavanya Jayapalan

Contributors: Chuck Murray, Bill Beauregard, Hector Briseno, Hassan Chafi, Zazhil Herena, Sungpack Hong, Roberto Infante, Hugo Labra, Gabriela Montiel-Moreno, Siva Ravada, Carlos Reyes, Korbinian Schmid, Jane Tao, Zhe (Alan) Wu

Contents

2 Using Big Data Spatial and Graph with Spatial Data

4 Using Property Graphs in a Big Data Environment

A Third-Party Licenses for Bundled Software

B Hive and Spark Spatial SQL Functions

List of Figures

List of Tables

Preface

This document provides conceptual and usage information about Oracle Big Data Spatial and Graph, which enables you to create, store, and work with Spatial and Graph vector, raster, and property graph data in a Big Data environment.

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This document is intended for database and application developers in Big Data environments.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the titles in the Big Data Appliance library that contain *Oracle Big Data Spatial and Graph*, plus these other documents.

- *Oracle Big Data Connectors User's Guide*
- *Oracle Big Data Appliance Site Checklists*
- *Oracle Big Data Appliance Owner's Guide*
- *Oracle Big Data Appliance Safety and Compliance Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Changes in This Release for Oracle Big Data Spatial and Graph

Note:

Oracle Big Data Spatial and Graph is deprecated and is planned to be desupported in the future.

Big Data Spatial and Graph includes the following changes to the product:

- Changes for Release 3.0

Changes for Release 3.0

The following changes apply to Release 3.0 of Big Data Spatial and Graph.

New Features for Release 3.0

Significant new features for this release:

- GraphVisualization: Lightweight, single-page web application to visualize graphs.
- In-memory graph representation optimization for reduced memory usage and faster performance.
- Create custom graph algorithms and extend product graph algorithms with Java syntax.

Desupported for Release 3.0

- Apache Groovy-based shell is desupported.
- Property Graph support for data stored in Oracle NoSQL database is desupported.
- Deploying the single-machine, scale-up implementation of the in-memory analyst (PGX) via Hadoop Yarn is desupported in this release. (It was deprecated in Release 2.5.3.)

1 Big Data Spatial and Graph Overview

This chapter provides an overview of Oracle Big Data support for Oracle Spatial and Graph spatial and property graph features.

- About Big Data Spatial and Graph
  Oracle Big Data Spatial and Graph delivers advanced spatial and graph analytic capabilities on supported Apache Hadoop Big Data platforms.
- Spatial Features
  Spatial location information is a common element of Big Data.
- Property Graph Features
  Graphs manage networks of linked data as vertices, edges, and properties of the vertices and edges. Graphs are commonly used to model, store, and analyze relationships found in social networks, cyber security, utilities and telecommunications, life sciences and clinical data, and knowledge networks.
- Installing Oracle Big Data Spatial and Graph on an Oracle Big Data Appliance
  The Mammoth command-line utility for installing and configuring the Oracle Big Data Appliance software also installs the Oracle Big Data Spatial and Graph option, including the spatial and property graph capabilities.
- Installing and Configuring the Big Data Spatial Image Processing Framework
  Installing and configuring the Image Processing Framework depends upon the distribution being used.
- Installing the Oracle Big Data SpatialViewer Web Application
  To install the Oracle Big Data SpatialViewer web application (SpatialViewer), follow the instructions in this topic.
- Installing Big Data Spatial and Graph in Non-BDA Environments
  Some actions may be required if you install Big Data Spatial and Graph in an environment other than Oracle Big Data Appliance.
- Required Application Code Changes due to Upgrades
  Application code changes may be required due to upgrades, such as to more recent versions of Apache HBase and SolrCloud.

1.1 About Big Data Spatial and Graph

Oracle Big Data Spatial and Graph delivers advanced spatial and graph analytic capabilities on supported Apache Hadoop Big Data platforms.

The spatial features include support for data enrichment of location information, spatial filtering and categorization based on distance and location-based analysis, and spatial data processing for vector and raster processing of digital map, sensor, satellite and aerial imagery values, and APIs for map visualization.

1.2 Spatial Features

Spatial location information is a common element of Big Data.

Businesses can use spatial data as the basis for associating and linking disparate data sets. Location information can also be used to track and categorize entities based on proximity to another person, place, or object, or on their presence a particular area. Location information can facilitate location-specific offers to customers entering a particular geography, something known as *geo-fencing.* Georeferenced imagery and sensory data can be analyzed for a variety of business benefits.

The spatial features of Oracle Big Data Spatial and Graph support those use cases with the following kinds of services.

Vector Services:

- Ability to associate documents and data with names, such as cities or states, or longitude/latitude information in spatial object definitions for a default administrative hierarchy
- Support for text-based 2D and 3D geospatial formats, including GeoJSON files, Shapefiles, GML, and WKT, or you can use the Geospatial Data Abstraction Library (GDAL) to convert popular geospatial encodings such as Oracle SDO_Geometry, ST_Geometry, and other supported formats
- An HTML5-based map client API and a sample console to explore, categorize, and view data in a variety of formats and coordinate systems
- Topological and distance operations: Anyinteract, Inside, Contains, Within Distance, Nearest Neighbor, and others
- Spatial indexing for fast retrieval of data

Raster Services:

- Support for many image file formats supported by GDAL and image files stored in HDFS
- A sample console to view the set of images that are available
- Raster operations, including, subsetting, georeferencing, mosaics, and format conversion

1.3 Property Graph Features

Graphs manage networks of linked data as vertices, edges, and properties of the vertices and edges. Graphs are commonly used to model, store, and analyze relationships found in social networks, cyber security, utilities and telecommunications, life sciences and clinical data, and knowledge networks.

Typical graph analyses encompass graph traversal, recommendations, finding communities and influencers, and pattern matching. Industries including, telecommunications, life sciences and healthcare, security, media and publishing can benefit from graphs. These use cases are supported by the property graph features of Oracle Big Data Spatial and Graph.

Property Graph features in Big Data platforms are enabled by using Oracle Graph HDFS Connector which is part of Oracle Graph Server and Client. Relevant features of Oracle Graph Server and Client are supported by accessing data in Apache HDFS using this connector. See Oracle Database Graph Developer's Guide for Property Graph for more information.

1.4 Installing Oracle Big Data Spatial and Graph on an Oracle Big Data Appliance

The Mammoth command-line utility for installing and configuring the Oracle Big Data Appliance software also installs the Oracle Big Data Spatial and Graph option, including the spatial and property graph capabilities.

You can enable this option during an initial software installation, or afterward using the `bdacli` utility.

To use Oracle NoSQL Database as a graph repository, you must have an Oracle NoSQL Database cluster.

To use Apache HBase as a graph repository, you must have an Apache Hadoop cluster.

> **See Also:**
>
> *Oracle Big Data Appliance Owner's Guide* for software configuration instructions.

1.5 Installing and Configuring the Big Data Spatial Image Processing Framework

Installing and configuring the Image Processing Framework depends upon the distribution being used.

- The Oracle Big Data Appliance cluster distribution comes with a pre-installed setup, but you must follow few steps in Installing the Image Processing Framework for Oracle Big Data Appliance Distribution to get it working.
- For a commodity distribution, follow the instructions in Installing the Image Processing Framework for Other Distributions (Not Oracle Big Data Appliance).

For both distributions:

- You must download and compile PROJ libraries, as explained in Getting and Compiling the Cartographic Projections Library.
- After performing the installation, verify it (see Post-installation Verification of the Image Processing Framework).
- If the cluster has security enabled, make sure that the user executing the jobs is in the `princs` list and has an active Kerberos ticket.

- Getting and Compiling the Cartographic Projections Library
- Installing the Image Processing Framework for Oracle Big Data Appliance Distribution
  The Oracle Big Data Appliance distribution comes with a pre-installed configuration, though you must ensure that the image processing framework has been installed.

- Installing the Image Processing Framework for Other Distributions (Not Oracle Big Data Appliance)
  For Big Data Spatial and Graph in environments other than the Big Data Appliance, follow the instructions in this section.
- Post-installation Verification of the Image Processing Framework
  Several test scripts are provided to perform the following verification operations.

1.5.1 Getting and Compiling the Cartographic Projections Library

Before installing the Image Processing Framework, you must download the Cartographic Projections Library and perform several related operations.

1. Download the PROJ.4 source code and datum shifting files:

```
$ wget http://download.osgeo.org/proj/proj-4.9.1.tar.gz
$ wget http://download.osgeo.org/proj/proj-datumgrid-1.5.tar.gz
```

2. Untar the source code, and extract the datum shifting files in the `nad` subdirectory:

```
$ tar xzf proj-4.9.1.tar.gz
$ cd proj-4.9.1/nad
$ tar xzf ../../proj-datumgrid-1.5.tar.gz
$ cd ..
```

3. Configure, make, and install PROJ.4:

```
$ ./configure
$ make
$ sudo make install
$ cd ..
```

   `libproj.so` is now available at `/usr/local/lib/libproj.so`.

4. Copy the `libproj.so` file in the spatial installation directory:

```
cp /usr/local/lib/libproj.so /opt/oracle/oracle-spatial-graph/
spatial/raster/gdal/lib/libproj.so
```

5. Provide read and execute permissions for the `libproj.so` library for all users

```
sudo chmod 755 /opt/oracle/oracle-spatial-graph/spatial/raster/
gdal/lib/libproj.so
```

1.5.2 Installing the Image Processing Framework for Oracle Big Data Appliance Distribution

The Oracle Big Data Appliance distribution comes with a pre-installed configuration, though you must ensure that the image processing framework has been installed.

Be sure that the actions described in Getting and Compiling the Cartographic Projections Library have been performed, so that `libproj.so` (`PROJ.4`) is accessible to all users and is set up correctly.

For OBDA, ensure that the following directories exist:

- SHARED_DIR (shared directory for all nodes in the cluster): `/opt/shareddir`
- ALL_ACCESS_DIR (shared directory for all nodes in the cluster with Write access to the hadoop group): `/opt/shareddir/spatial`

1.5.3 Installing the Image Processing Framework for Other Distributions (Not Oracle Big Data Appliance)

For Big Data Spatial and Graph in environments other than the Big Data Appliance, follow the instructions in this section.

- Prerequisites for Installing the Image Processing Framework for Other Distributions
- Installing the Image Processing Framework for Other Distributions

1.5.3.1 Prerequisites for Installing the Image Processing Framework for Other Distributions

- Ensure that `HADOOP_LIB_PATH` is under `/usr/lib/hadoop`. If it is not there, find the path and use it as it your `HADOOP_LIB_PATH`.
- Install NFS.
- Have at least one folder, referred in this document as SHARED_FOLDER, in the Resource Manager node accessible to every Node Manager node through NFS.
- Provide write access to all the users involved in job execution and the yarn users to this SHARED_FOLDER
- Download `oracle-spatial-graph-<version>.x86_64.rpm` from the Oracle e-delivery web site.
- Execute `oracle-spatial-graph-<version>.x86_64.rpm` using the rpm command.
- After rpm executes, verify that a directory structure created at `/opt/oracle/oracle-spatial-graph/spatial/raster` contains these folders: `console`, `examples`, `jlib`, `gdal`, and `tests`. Additionally, `index.html` describes the content, and `javadoc.zip` contains the Javadoc for the API..

1.5.3.2 Installing the Image Processing Framework for Other Distributions

1. Make the `libproj.so (Proj.4)` Cartographic Projections Library accessible to the users, as explained in Getting and Compiling the Cartographic Projections Library.
2. In the Resource Manager Node, copy the `data` folder under `/opt/oracle/oracle-spatial-graph/spatial/raster/gdal` into the SHARED_FOLDER as follows:

   ```
   cp -R /opt/oracle/oracle-spatial-graph/spatial/raster/gdal/data
   SHARED_FOLDER
   ```
3. Create a directory `ALL_ACCESS_FOLDER` under `SHARED_FOLDER` with write access for all users involved in job execution. Also consider the `yarn` user in the write access because job results are written by this user. Group access may be used to configure this.

   Go to the shared folder.

   ```
   cd SHARED_FOLDER
   ```

Create a new directory.

```
mkdir ALL_ACCESS_FOLDER
```

Provide write access.

```
chmod 777 ALL_ACCESS_FOLDER
```

4. Copy the `data` folder under `/opt/oracle/oracle-spatial-graph/spatial/raster/examples` into `ALL_ACCESS_FOLDER`.

   ```
   cp -R /opt/oracle/oracle-spatial-graph/spatial/raster/examples/data
   ALL_ACCESS_FOLDER
   ```

5. Provide write access to the `data/xmls` folder as follows (or just ensure that users executing the jobs, including tests and examples, have write access):

   ```
   chmod 777 ALL_ACCESS_FOLDER/data/xmls/
   ```

1.5.4 Post-installation Verification of the Image Processing Framework

Several test scripts are provided to perform the following verification operations.

- Test the image loading functionality
- Test test the image processing functionality
- Test a processing class for slope calculation in a DEM and a map algebra operation
- Verify the image processing of a single raster with no mosaic process (it includes a user-provided function that calculates hill shade in the mapping phase).
- Test processing of two rasters using a mask operation

Execute these scripts to verify a successful installation of image processing framework.

If the cluster has security enabled, make sure the current user is in the `princs` list and has an active Kerberos ticket.

Make sure the user has write access to ALL_ACCESS_FOLDER and that it belongs to the owner group for this directory. It is recommended that jobs be executed in Resource Manager node for Big Data Appliance. If jobs are executed in a different node, then the default is the hadoop group.

For GDAL to work properly, the libraries must be available using $LD_LIBRARY_PATH. Make sure that the shared libraries path is set properly in your shell window before executing a job. For example:

```
export LD_LIBRARY_PATH=$ALLACCESSDIR/gdal/native
```

- Image Loading Test Script
- Image Processor Test Script (Mosaicking)
- Single-Image Processor Test Script
- Image Processor DEM Test Script
- Multiple Raster Operation Test Script

1.5.4.1 Image Loading Test Script

This script loads a set of six test rasters into the `ohiftest` folder in HDFS, 3 rasters of byte data type and 3 bands, 1 raster (DEM) of float32 data type and 1 band, and 2 rasters of int32 data type and 1 band. No parameters are required for OBDA environments and a single parameter with the ALL_ACCESS_FOLDER value is required for non-OBDA environments.

Internally, the job creates a split for every raster to load. Split size depends on the block size configuration; for example, if a block size >= 64MB is configured, 4 mappers will run; and as a result the rasters will be loaded in HDFS and a corresponding thumbnail will be created for visualization. An external image editor is required to visualize the thumbnails, and an output path of these thumbnails is provided to the users upon successful completion of the job.

The test script can be found here:

```
/opt/oracle/oracle-spatial-graph/spatial/raster/tests/runimageloader.sh
```

For ODBA environments, enter:

```
./runimageloader.sh
```

For non-ODBA environments, enter:

```
./runimageloader.sh ALL_ACCESS_FOLDER
```

Upon successful execution, the message `GENERATED OHIF FILES ARE LOCATED IN HDFS UNDER` is displayed, with the path in HDFS where the files are located (this path depends on the definition of ALL_ACCESS_FOLDER) and a list of the created images and thumbnails on HDFS. The output may include:

```
"THUMBNAILS CREATED ARE:
----------------------------------------------------------------------
total 13532
drwxr-xr-x 2 yarn yarn 4096 Sep 9 13:54 .
drwxr-xr-x 3 yarn yarn 4096 Aug 27 11:29 ..
-rw-r--r-- 1 yarn yarn 3214053 Sep  9 13:54 hawaii.tif.ohif.tif
-rw-r--r-- 1 yarn yarn 3214053 Sep  9 13:54 inputimageint32.tif.ohif.tif
-rw-r--r-- 1 yarn yarn 3214053 Sep  9 13:54 inputimageint32_1.tif.ohif.tif
-rw-r--r-- 1 yarn yarn 3214053 Sep  9 13:54 kahoolawe.tif.ohif.tif
-rw-r--r-- 1 yarn yarn 3214053 Sep  9 13:54 maui.tif.ohif.tif
-rw-r--r-- 1 yarn yarn 4182040 Sep  9 13:54 NapaDEM.tif.ohif.tif
YOU MAY VISUALIZE THUMBNAILS OF THE UPLOADED IMAGES FOR REVIEW FROM THE FOLLOWING PATH:
```

If the installation and configuration were not successful, then the output is not generated and a message like the following is displayed:

```
NOT ALL THE IMAGES WERE UPLOADED CORRECTLY, CHECK FOR HADOOP LOGS
```

The amount of memory required to execute mappers and reducers depends on the configured HDFS block size By default, 1 GB of memory is assigned for Java, but you can modify that and other properties in the `imagejob.prop` file that is included in this test directory.

1.5.4.2 Image Processor Test Script (Mosaicking)

This script executes the processor job by setting three source rasters of Hawaii islands and some coordinates that includes all three. The job will create a mosaic based on these coordinates and resulting raster should include the three rasters combined in a single one.

runimageloader.sh should be executed as a prerequisite, so that the source rasters exist in HDFS. These are 3 band rasters of byte data type.

No parameters are required for OBDA environments, and a single parameter "-s" with the ALL_ACCESS_FOLDER value is required for non-OBDA environments.

Additionally, if the output should be stored in HDFS, the "-o" parameters must be used to set the HDFS folder where the mosaic output will be stored.

Internally the job filters the tiles using the coordinates specified in the configuration input, xml, only the required tiles are processed in a mapper and finally in the reduce phase, all of them are put together into the resulting mosaic raster.

The test script can be found here:

```
/opt/oracle/oracle-spatial-graph/spatial/raster/tests/runimageprocessor.sh
```

For ODBA environments, enter:

```
./runimageprocessor.sh
```

For non-ODBA environments, enter:

```
./runimageprocessor.sh -s ALL_ACCESS_FOLDER
```

Upon successful execution, the message `EXPECTED OUTPUT FILE IS: ALL_ACCESS_FOLDER/processtest/hawaiimosaic.tif` is displayed, with the path to the output mosaic file. The output may include:

```
EXPECTED OUTPUT FILE IS: ALL_ACCESS_FOLDER/processtest/hawaiimosaic.tif
total 9452
drwxrwxrwx 2 hdfs    hdfs    4096 Sep 10 09:12 .
drwxrwxrwx 9 zherena dba     4096 Sep  9 13:50 ..
-rwxrwxrwx 1 yarn    yarn 4741101 Sep 10 09:12 hawaiimosaic.tif

MOSAIC IMAGE GENERATED
----------------------------------------------------------------------
YOU MAY VISUALIZE THE MOSAIC OUTPUT IMAGE FOR REVIEW IN THE FOLLOWING PATH:
ALL_ACCESS_FOLDER/processtest/hawaiimosaic.tif”
```

If the installation and configuration were not successful, then the output is not generated and a message like the following is displayed:

```
MOSAIC WAS NOT SUCCESSFULLY CREATED, CHECK HADOOP LOGS TO REVIEW THE PROBLEM
```

To test the output storage in HDFS, use the following command

For ODBA environments, enter:

```
./runimageprocessor.sh -o hdfstest
```

For non-ODBA environments, enter:

```
./runimageprocessor.sh -s ALL_ACCESS_FOLDER -o hdfstest
```

1.5.4.3 Single-Image Processor Test Script

This script executes the processor job for a single raster, in this case is a DEM source raster of North Napa Valley. The purpose of this job is process the complete input by using the user processing classes configured for the mapping phase. This class calculates the hillshade of the DEM, and this is set to the output file. No mosaic operation is performed here.

`runimageloader.sh` should be executed as a prerequisite, so that the source raster exists in HDFS. This is 1 band of float 32 data type DEM rasters.

No parameters are required for OBDA environments, and a single parameter "-s" with the ALL_ACCESS_FOLDER value is required for non-OBDA environments.

The test script can be found here:

```
/opt/oracle/oracle-spatial-graph/spatial/raster/tests/runsingleimageprocessor.sh
```

For ODBA environments, enter:

```
./runsingleimageprocessor.sh
```

For non-ODBA environments, enter:

```
./runsingleimageprocessor.sh -s ALL_ACCESS_FOLDER
```

Upon successful execution, the message `EXPECTED OUTPUT FILE: ALL_ACCESS_FOLDER/processtest/NapaSlope.tif` is displayed, with the path to the output DEM file. The output may include:

```
EXPECTED OUTPUT FILE: ALL_ACCESS_FOLDER/processtest/NapaDEM.tif
total 4808
drwxrwxrwx 2 hdfs    hdfs    4096 Sep 10 09:42 .
drwxrwxrwx 9 zherena dba     4096 Sep  9 13:50 ..
-rwxrwxrwx 1 yarn    yarn 4901232 Sep 10 09:42 NapaDEM.tif
IMAGE GENERATED
----------------------------------------------------------------------

YOU MAY VISUALIZE THE OUTPUT IMAGE FOR REVIEW IN THE FOLLOWING PATH: ALL_ACCESS_FOLDER/
processtest/NapaDEM.tif”
```

If the installation and configuration were not successful, then the output is not generated and a message like the following is displayed:

```
IMAGE WAS NOT SUCCESSFULLY CREATED, CHECK HADOOP LOGS TO REVIEW THE PROBLEM
```

1.5.4.4 Image Processor DEM Test Script

This script executes the processor job by setting a DEM source raster of North Napa Valley and some coordinates that surround it. The job will create a mosaic based on these coordinates and will also calculate the slope on it by setting a processing class in the mosaic configuration XML.

`runimageloader.sh` should be executed as a prerequisite, so that the source rasters exist in HDFS. This is 1 band of float 32 data type DEM raster.

No parameters are required for OBDA environments, and a single parameter "-s" with the ALL_ACCESS_FOLDER value is required for non-OBDA environments.

The test script can be found here:

```
/opt/oracle/oracle-spatial-graph/spatial/raster/tests/runimageprocessordem.sh
```

For ODBA environments, enter:

```
./runimageprocessordem.sh
```

For non-ODBA environments, enter:

```
./runimageprocessordem.sh -s ALL_ACCESS_FOLDER
```

Upon successful execution, the message `EXPECTED OUTPUT FILE: ALL_ACCESS_FOLDER/processtest/NapaSlope.tif` is displayed, with the path to the slope output file. The output may include:

```
EXPECTED OUTPUT FILE: ALL_ACCESS_FOLDER/processtest/NapaSlope.tif
total 4808
drwxrwxrwx 2 hdfs    hdfs    4096 Sep 10 09:42 .
drwxrwxrwx 9 zherena dba     4096 Sep  9 13:50 ..
-rwxrwxrwx 1 yarn    yarn 4901232 Sep 10 09:42 NapaSlope.tif
MOSAIC IMAGE GENERATED
----------------------------------------------------------------------

YOU MAY VISUALIZE THE MOSAIC OUTPUT IMAGE FOR REVIEW IN THE FOLLOWING PATH:
ALL_ACCESS_FOLDER/processtest/NapaSlope.tif"
```

If the installation and configuration were not successful, then the output is not generated and a message like the following is displayed:

```
MOSAIC WAS NOT SUCCESSFULLY CREATED, CHECK HADOOP LOGS TO REVIEW THE PROBLEM
```

You may also test the "if" algebra function, where every pixel in this raster with value greater than 2500 will be replaced by the value you set in the command line using the "–c" flag. For example:

For ODBA environments, enter:

```
./runimageprocessordem.sh –c 8000
```

For non-ODBA environments, enter:

```
./runimageprocessordem.sh -s ALL_ACCESS_FOLDER –c 8000
```

You can visualize the output file and notice the difference between simple slope calculation and this altered output, where the areas with pixel values greater than 2500 look more clear.

1.5.4.5 Multiple Raster Operation Test Script

This script executes the processor job for two rasters that cover a very small area of North Napa Valley in the US state of California.

These rasters have the same MBR, pixel size, SRID, and data type, all of which are required for complex multiple raster operation processing. The purpose of this job is process both rasters by using the *mask* operation, which checks every pixel in the second raster to validate if its value is contained in the mask list. If it is, the output raster will have the pixel value of the first raster for this output cell; otherwise, the zero (0) value is set. No mosaic operation is performed here.

`runimageloader.sh` should be executed as a prerequisite, so that the source rasters exist in HDFS. These are 1 band of int32 data type rasters.

No parameters are required for OBDA environments. For non-ODBA environments, a single parameter `-s` with the ALL_ACCESS_FOLDER value is required.

The test script can be found here:

```
/opt/oracle/oracle-spatial-graph/spatial/raster/tests/runimageprocessormultiple.sh
```

For ODBA environments, enter:

```
./runimageprocessormultiple.sh
```

For non-ODBA environments, enter:

```
./runimageprocessormultiple.sh -s ALL_ACCESS_FOLDER
```

Upon successful execution, the message `EXPECTED OUTPUT FILE: ALL_ACCESS_FOLDER/processtest/MaskInt32Rasters.tif` is displayed, with the path to the mask output file. The output may include:

```
EXPECTED OUTPUT FILE: ALL_ACCESS_FOLDER/processtest/MaskInt32Rasters.tif
total 4808
drwxrwxrwx 2 hdfs    hdfs    4096 Sep 10 09:42 .
drwxrwxrwx 9 zherena dba     4096 Sep  9 13:50 ..
-rwxrwxrwx 1 yarn    yarn 4901232 Sep 10 09:42 MaskInt32Rasters.tif
IMAGE GENERATED
----------------------------------------------------------------------

YOU MAY VISUALIZE THE OUTPUT IMAGE FOR REVIEW IN THE FOLLOWING PATH: ALL_ACCESS_FOLDER/
processtest/MaskInt32Rasters.tif”
```

If the installation and configuration were not successful, then the output is not generated and a message like the following is displayed:

```
IMAGE WAS NOT SUCCESSFULLY CREATED, CHECK HADOOP LOGS TO REVIEW THE PROBLEM
```

1.6 Installing the Oracle Big Data SpatialViewer Web Application

To install the Oracle Big Data SpatialViewer web application (SpatialViewer), follow the instructions in this topic.

- Assumptions for SpatialViewer
- Installing SpatialViewer on Oracle Big Data Appliance
- Installing SpatialViewer for Other Systems (Not Big Data Appliance)
- Configuring SpatialViewer on Oracle Big Data Appliance
- Configuring SpatialViewer for Other Systems (Not Big Data Appliance)

> **See Also:**
>
> Using the Oracle Big Data SpatialViewer Web Application

1.6.1 Assumptions for SpatialViewer

The following assumptions apply for installing and configuring SpatialViewer.

- The API and jobs described here run on a Cloudera CDH6 or similar Hadoop environment.
- Java 8 or a newer version is present in your environment.
- The image processing framework has been installed as described in Installing and Configuring the Big Data Spatial Image Processing Framework.

1.6.2 Installing SpatialViewer on Oracle Big Data Appliance

You can install SpatialViewer on Big Data Appliance as follows

1. Run the following script:

   ```
   sudo /opt/oracle/oracle-spatial-graph/spatial/configure-server/
   install-bdsg-consoles.sh
   ```

2. Start the web application by using **one** of the following commands (the second command enables you to view logs):

   ```
   sudo service bdsg start
   sudo /opt/oracle/oracle-spatial-graph/spatial/web-server/start-
   server.sh
   ```

   If any errors occur, see the README file located in `/opt/oracle/oracle-spatial-graph/spatial/configure-server`.

3. Open: `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/`
4. If the active nodes have changed after the installation or if Kerberos is enabled, then update the configuration file as described in Configuring SpatialViewer on Oracle Big Data Appliance.
5. Optionally, upload sample data (used with examples in other topics) to HDFS:

   ```
   sudo -u hdfs hadoop fs -mkdir /user/oracle/bdsg
   sudo -u hdfs hadoop fs -put /opt/oracle/oracle-spatial-graph/
   spatial/vector/examples/data/tweets.json /user/oracle/bdsg/
   ```

1.6.3 Installing SpatialViewer for Other Systems (Not Big Data Appliance)

Follow the steps for manual configuration described in Installing SpatialViewer on Oracle Big Data Appliance.

Then, change the configuration, as described in Configuring SpatialViewer for Other Systems (Not Big Data Appliance)

1.6.4 Configuring SpatialViewer on Oracle Big Data Appliance

To configure SpatialViewer on Oracle Big Data Appliance, follow these steps.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=swadmin`

2. Change the general configuration, as needed:
   - Local working directory: SpatialViewer local working directory. Absolute path. The default directory `/usr/oracle/spatialviewer` is created when installing SpatialViewer.
   - HDFS working directory: SpatialViewer HDFS working directory. The default directory `/user/oracle/spatialviewer` is created when installing SpatialViewer.
   - Hadoop configuration file: The Hadoop configuration directory. By default: `/etc/hadoop/conf`

     If you change this value, you must restart the server.
   - Spark configuration file: The Spark configuration directory. By default: `/etc/spark/conf`

     If you change this value, you must restart the server.
   - eLocation URL: URL used to get the eLocation background maps. By default: `http://elocation.oracle.com`
   - Kerberos keytab: If Kerberos is enabled, provide the full path to the file that contains the keytab file.
   - Display logs: If necessary, disable the display of the jobs in the Spatial Jobs screen. Disable this display if the logs are not in the default format. The default format is: `Date LogLevel LoggerName: LogMessage`

     The Date must have the default format: `yyyy-MM-dd HH:mm:ss,SSS`. For example: `2012-11-02 14:34:02,781`.

     If the logs are not displayed and the Display logs field is set to `Yes`, then ensure that `yarn.log-aggregation-enable` in `yarn-site.xml` is set to `true`. Also ensure that the Hadoop jobs configuration parameters `yarn.nodemanager.remote-app-log-dir` and `yarn.nodemanager.remote-app-log-dir-suffix` are set to the same value as in `yarn-site.xml`.
3. Change the raster configuration, as needed:
   - Shared directory: Directory used to read and write from different nodes, which requires that is be shared and have the greatest permissions or at least be in the Hadoop user group.
   - Network file system mount point: NFS mountpoint that allows the shared folders to be seen and accessed individually. Can be blank if you are using a non-distributed environment.
   - GDAL directory: Native GDAL installation directory. Must be accessible to all the cluster nodes.

     If you change this value, you must restart the server.
   - Shared GDAL data directory: GDAL shared data folder. Must be a shared directory. (See the instructions in Installing the Image Processing Framework for Other Distributions (Not Oracle Big Data Appliance).)
4. Change the Hadoop configuration, as needed.
5. Change the Spark configuration, as needed. The raster processor needs additional configuration details:
   - `spark.driver.extraClassPath, spark.executor.extraClassPath`: Specify your hive library installation using these keys. Example: `/usr/lib/hive/lib/*`

- `spark.kryoserializer.buffer.max`: Enter the memory for the data serialization. Example: `160m`

6. If Kerberos is enabled, then you may need to add the parameters:
   - `spark.yarn.keytab`: the full path to the file that contains the keytab for the principal.
   - `spark.yarn.principal`: the principal to be used to log in to Kerberos. The format of a typical Kerberos V5 principal is `primary/instance@REALM`.
7. On Linux systems, you may need to change the secure container executor to `LinuxContainerExecutor`. For that, set the following parameters:
   - Set `yarn.nodemanager.container-executor.class` to `org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor`.
   - Set `yarn.nodemanager.linux-container-executor.group` to `hadoop`.
8. Ensure that the user can read the keytab file.

1.6.5 Configuring SpatialViewer for Other Systems (Not Big Data Appliance)

Before installing the SpatialViewer on other systems, you must install the image processing framework as specified in Installing the Image Processing Framework for Other Distributions (Not Oracle Big Data Appliance).

Then follow the steps mentioned in Configuring SpatialViewer on Oracle Big Data Appliance.

Additionally, change the Hadoop and Spark configuration, replacing the Hadoop `conf.` directory and Spark `conf.` directory values according your Hadoop and Spark installations.

1.7 Installing Big Data Spatial and Graph in Non-BDA Environments

Some actions may be required if you install Big Data Spatial and Graph in an environment other than Oracle Big Data Appliance.

Starting with Big Data Spatial and Graph (BDSG) 2.5.3, third-party libraries provided by Cloudera required for interaction with Cloudera CDH are no longer distributed with the BDSG distribution. This topic describes the actions that may be needed to enable Cloudera CDH support with BDSG.

On Oracle Big Data Appliance (BDA), BDSG is preconfigured to work with Cloudera CDH "out of the box" as in previous BDSG releases. So any additional installation steps are not required for a BDA environment.

- Automatic Installation of BDSG
- Manual Installation of BDSG
- Configuring the BDSG Environment
- Managing BDSG Text Indexing Using Apache Lucene 7.0

- Managing BDSG Text Indexing Using SolrCloud 7.0

1.7.1 Automatic Installation of BDSG

After installing the .rpm, you can attempt an automatic installation by running the following script as root:

```
/opt/oracle/oracle-spatial-graph/property_graph/configure-hadoop.sh
```

This script makes many assumptions about your Hadoop distribution and version. If any commands in the script fails, perform a manual installation.

1.7.2 Manual Installation of BDSG

To perform a manual installation, use the subtopic relevant to your environment

- HDFS
- Yarn
- HBase

HDFS

Go into the BDSG property graph installation directory:

```
cd /opt/oracle/oracle-spatial-graph/property_graph
```

Set HADOOP_HOME to point to your Hadoop installation base path. For example:

```
HADOOP_HOME=/scratch/cloudera/parcels/CDH-6.0.1-1.cdh6.0.1.p0.590678
```

Copy the required HDFS libraries (and their dependencies) into the hadoop/hdfs directory. (The exact location and version of above JAR files may vary depending on Hadoop distribution and Hadoop version. So, you might have to change some of those input paths to match your cluster installation.)

```
cp $HADOOP_HOME/lib/hadoop/hadoop-auth-3.0.0-cdh6.0.1.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop/hadoop-common-3.0.0-cdh6.0.1.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/hadoop-hdfs-3.0.0-cdh6.0.1.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/hadoop-hdfs-client-3.0.0-cdh6.0.1.jar hadoop/
hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/commons-cli-1.2.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/commons-collections-3.2.2.jar hadoop/
hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/commons-lang-2.6.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/commons-logging-1.1.3.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/stax2-api-3.1.4.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/woodstox-core-5.0.3.jar hadoop/hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/htrace-core4-4.1.0-incubating.jar hadoop/
hdfs/
cp $HADOOP_HOME/lib/hadoop-hdfs/lib/protobuf-java-2.5.0.jar hadoop/hdfs/
```

To enable PGX server to access HDFS, you also need to copy the libraries into the .war file:

```
mkdir -p WEB-INF/lib
cp /opt/oracle/oracle-spatial-graph/property_graph/hadoop/hdfs/* WEB-
INF/lib/
jar -uvf /opt/oracle/oracle-spatial-graph/property_graph/pgx/webapp/
pgx-webapp-<version>.war WEB-INF/lib/
rm -r WEB-INF
```

Then start the server by either running the `./pgx/bin/start-server` script or by deploying the WAR file into an application server.

Yarn

Go into the BDSG property graph installation directory:

```
cd /opt/oracle/oracle-spatial-graph/property_graph
```

Locate the path to the Zookeeper JAR file of your Hadoop distribution, for example `$HADOOP_HOME/lib/zookeeper/zookeeper-3.4.5-cdh6.0.1.jar`. Then run the following script to configure your BDSG installation to work with Yarn:

```
TMP_DIR=$(mktemp -d)
cd "${TMP_DIR}"
jar xf "${HADOOP_HOME}/lib/zookeeper/zookeeper-3.4.5-cdh6.0.1.jar"
rm META-INF/MANIFEST.MF
jar -uf /opt/oracle/oracle-spatial-graph/property_graph/hadoop/yarn/
pgx-yarn-<version>.jar .
rm -rf "${TMP_DIR}"
```

HBase

Go into the BDSG property graph installation directory:

```
cd /opt/oracle/oracle-spatial-graph/property_graph
```

Create a hadoop/hbase directory to hold all the HBase libraries (and their dependencies) required for execution:

```
mkdir -p hadoop/hbase
```

Set HADOOP_HOME to point to your Hadoop installation base path. For example:

```
HADOOP_HOME=/scratch/cloudera/parcels/CDH-6.0.1-1.cdh6.0.1.p0.590678
```

Copy the required HDFS libraries (and their dependencies) into the hadoop/hdfs directory. (The exact location and version of above JAR files may vary depending on Hadoop distribution and Hadoop version. So, you might have to change some of those input paths to match your cluster installation.)

```
cp $HADOOP_HOME/lib/hbase/hbase-client-2.0.0-cdh6.0.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/hbase-common-2.0.0-cdh6.0.1.jar hadoop/hbase
```

```
cp $HADOOP_HOME/lib/hbase/hbase-protocol-2.0.0-cdh6.0.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/hbase-shaded-protobuf-2.1.0.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/shaded-clients/hbase-shaded-client-2.0.0-
cdh6.0.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/hadoop/hadoop-common-3.0.0-cdh6.0.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/hadoop-hdfs/hadoop-hdfs-3.0.0-cdh6.0.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/zookeeper/zookeeper-3.4.5-cdh6.0.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/protobuf-java-2.5.0.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/metrics-core-3.2.1.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/jettison-1.3.8.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/stax2-api-3.1.4.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/woodstox-core-5.0.3.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/client-facing-thirdparty/htrace-core4-4.2.0-
incubating.jar hadoop/hbase
cp $HADOOP_HOME/lib/hbase/lib/client-facing-thirdparty/audience-
annotations-0.5.0.jar hadoop/hbase
```

To enable the PGX server to access HBase, you also need to copy the HBase libraries into the .war file:

```
mkdir -p WEB-INF/lib
cp /opt/oracle/oracle-spatial-graph/property_graph/hadoop/hbase/* WEB-
INF/lib/
jar -uvf /opt/oracle/oracle-spatial-graph/property_graph/pgx/webapp/pgx-
webapp-<version>.war WEB-INF/lib/
rm -r WEB-INF
```

Then start the server by either running the `./pgx/bin/start-server` script or by deploying the WAR file into an application server.

1.7.3 Configuring the BDSG Environment

To configure the environment, use the subtopic relevant to your environment.

- HDFS
- Yarn
- HBase

HDFS

Set the HADOOP_CONF_DIR environment variable to point to the HDFS configuration directory of your cluster. For example:

```
export HADOOP_CONF_DIR=/etc/hadoop/conf
```

Set the BDSG_CLASSPATH environment variable to point to the libraries of the previous step before starting the shell. For example:

```
export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/property_graph/hadoop/
hdfs/*
```

Then start the shell as usual and access data from HDFS using the `hdfs` path prefix:

```
cd /opt/oracle/oracle-spatial-graph/property_graph
./pgx/bin/pgx
[WARNING] BDSG_CLASSPATH environment will be prepended to PGX
classpath. If this is not intended, do 'unset BDSG_CLASSPATH' and
restart.

PGX Shell 3.1.3
type :help for available commands
12:01:30,824 INFO Ctrl$1 - >>> PGX engine 3.1.3 running.
variables instance, session and analyst ready to use
pgx> g = session.readGraphWithProperties('hdfs:/tmp/data/
connections.edge_list.json')
==> PgxGraph[name=connections,N=78,E=164,created=1543176112779]
```

Yarn

Copy the required artifacts for Yarn deployments into a HDFS directory of your choice by running the following helper script:

```
./pgx/scripts/install-pgx-hdfs.sh <dest-dir>
```

where *<dest-dir>* could be `hdfs:/binaries/pgx`, for example.

Make sure the `hdfs` binary is on your PATH environment variable.

After the script finishes, make sure to update `pgx/conf/yarn.conf` to contain the paths to the installed binaries and the correct Zookeeper connection string of your cluster.

HBase

To access a property graph in Apache HBase using DAL **in a Java application**:

1. Set BDSG_HOME environment variable to the property graph installation directory. For example:

   ```
   export BDSG_HOME=/opt/oracle/oracle-spatial-graph/property_graph
   ```

2. Set BDSG_CLASSPATH environment variable to the hadoop/hbase directory. For example:

   ```
   export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/
   property_graph/hadoop/hbase/*:$BDSG_CLASSPATH
   ```

3. Set BDSG_CLASSPATH environment variable to the hadoop/hbase directory. For example:

   ```
   javac -classpath $BDSG_HOME/lib/'*':$BDSG_CLASSPATH filename.java
   ```

4. Run the Java application by executing the compiled code, as follows:

   ```
   java -classpath ./:$BDSG_HOME/lib/'*':$BDSG_CLASSPATH filename args
   ```

To access a property graph in Apache HBase using DAL **in a Groovy console**:

1. Set the BDSG_CLASSPATH environment variable to the hadoop/hbase directory. For example:

```
export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/property_graph/
hadoop/hbase/*:$BDSG_CLASSPATH
```

2. Start the shell as usual and access data from an Apache HBase storage using an `OraclePropertyGraph` instance.
   Note that from Apache HBase 2.0, the `HConnection` interface has been deprecated, so use a `Connection` object to connect to the database.

```
cd /opt/oracle/oracle-spatial-graph/property_graph/dal/groovy
sh gremlin-opg-hbase.sh

--------------------------------

opg-hbase> conf = HBaseConfiguration.create();
==>hbase.rs.cacheblocksonwrite=false
==>...
opg-hbase> conf.set("hbase.zookeeper.quorum", "localhost");
==>null
opg-hbase> conf.set("hbase.zookeeper.property.clientPort","2181");
==>null
opg-hbase> conn = ConnectionFactory.createConnection(conf);
==>hconnection-0x720653c2
opg-hbase> opg=OraclePropertyGraph.getInstance(conf, conn, "connections");
==>oraclepropertygraph with name connections
```

1.7.4 Managing BDSG Text Indexing Using Apache Lucene 7.0

To manage text indexing over property graph data using Apache Lucene:

1. Go into the BDSG property graph installation directory:

```
cd /opt/oracle/oracle-spatial-graph/property_graph
```

2. Create a `lucene` directory to hold all the Apache Lucene 7.0 libraries (and their dependencies) required for execution:

```
mkdir lucene
```

3. Set HADOOP_HOME to point to your Hadoop installation base path. For example:

```
HADOOP_HOME=/scratch/cloudera/parcels/CDH-6.0.1-1.c
```

4. Copy the required Apache Lucene libraries into the `lucene` directory:

```
cp $HADOOP_HOME/lib/search/lucene-core.jar lucene
cp $HADOOP_HOME/lib/search/lucene-queryparser.jar lucene
cp $HADOOP_HOME/lib/search/lucene-analyzers-common.jar lucene
```

Then, use the subtopic relevant to your environment:

- Managing Text Indexing in a Java Application
- Managing Text Indexing Using a Groovy Console

Managing Text Indexing in a Java Application

1. Set BDSG_HOME to the property graph installation directory. For example:

```
export BDSG_HOME=/opt/oracle/oracle-spatial-graph/property_graph
```

2. Set BDSG_CLASSPATH to the `lucene` directory. For example:

```
export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/
property_graph/lucene/*:$BDSG_CLASSPATH
```

3. Compile the Java code. For example:

```
javac -classpath $BDSG_HOME/lib/'*':$BDSG_CLASSPATH filename.java
```

4. Run the Java application by executing the compiled code. For example:

```
java -classpath ./:$BDSG_HOME/lib/'*':$BDSG_CLASSPATH filename args
```

Managing Text Indexing Using a Groovy Console

1. Set BDSG_CLASSPATH to the `lucene` directory. For example:

```
export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/
property_graph/lucene/*:$BDSG_CLASSPATH
```

2. Start the shell as usual to create a text index over a property graph stored in Apache HBase storage using an `OraclePropertyGraph` instance. For example:

```
cd /opt/oracle/oracle-spatial-graph/property_graph/dal/groovy
sh gremlin-opg-hbase.sh

--------------------------------

opg-hbase> conf = HBaseConfiguration.create();
==>hbase.rs.cacheblocksonwrite=false
==>...
opg-hbase> dop=2;
==>2
opg-hbase> conf.set("hbase.zookeeper.quorum", "localhost");
==>null
opg-hbase> conf.set("hbase.zookeeper.property.clientPort","2181");
==>null
opg-hbase> conn = ConnectionFactory.createConnection(conf);
==>hconnection-0x720653c2
opg-hbase> opg=OraclePropertyGraph.getInstance(conf, conn,
"connections");
==>oraclepropertygraph with name connections
opg-hbase> indexParams = OracleIndexParameters.buildFS(dop /*
```

```
number of directories */, dop /* number of connections used when indexing
*/, 10000 /* batch size before commit*/, 500000 /* commit size before
Lucene commit*/, true /* enable datatypes */, "./lucene-index" /* index
location */);
==>[parameter[search-engine,1], parameter[num-subdirectories,4],
parameter[directory-type,FS_DIRECTORY], parameter[reindex-numConns,4],
parameter[batch-size,10000], parameter[commit-batch-size,500000],
parameter[values-as-strings,true], parameter[directory-location,
[Ljava.lang.String;@5c1f6d57]]
opg-hbase> opg.setDefaultIndexParameters(indexParams);
==>null
opg-hbase> indexedKeys = new String[4]; indexedKeys[0] = "name";
indexedKeys[1] = "role"; indexedKeys[2] = "religion"; indexedKeys[3] =
"country";
==>name
==>role
==>religion
==>country
opg-hbase> opg.createKeyIndex(indexedKeys, Vertex.class);
==>null
```

1.7.5 Managing BDSG Text Indexing Using SolrCloud 7.0

To manage text indexing over property graph data using Apache Lucene:

1. Go into the BDSG property graph installation directory:

   ```
   cd /opt/oracle/oracle-spatial-graph/property_graph
   ```

2. Create a `solrcloud` directory to hold all the Apache Lucene 7.0 libraries (and their dependencies) required for execution:

   ```
   mkdir solrcloud
   ```

3. Set HADOOP_HOME to point to your Hadoop installation base path. For example:

   ```
   HADOOP_HOME=/scratch/cloudera/parcels/CDH-6.0.1-1.cdh6.0.1.p0.590678
   ```

4. Copy the required SolrCloud libraries into the `lucene` directory:

   ```
   cp $HADOOP_HOME/lib/solr/solr-solrj-7.0.0-cdh6.0.1.jar solrcloud

   cp $HADOOP_HOME/lib/solr/lib/noggit-0.8.jar solrcloud
   cp $HADOOP_HOME/lib/solr/lib/httpmime-4.5.3.jar solrcloud
   cp $HADOOP_HOME/lib/search/lucene-core.jar solrcloud
   cp $HADOOP_HOME/lib/search/lucene-queryparser.jar solrcloud
   cp $HADOOP_HOME/lib/search/lucene-analyzers-common.jar solrcloud
   ```

5. Set BDSG_CLASSPATH to the `solrcloud` directory. For example:

   ```
   export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/property_graph/
   solrcloud/*:$BDSG_CLASSPATH
   ```

Then, use the subtopic relevant to your environment:

- Managing Text Indexing in a Java Application
- Managing Text Indexing Using a Groovy Console

Managing Text Indexing in a Java Application

1. Set BDSG_HOME to the property graph installation directory. For example:

```
export BDSG_HOME=/opt/oracle/oracle-spatial-graph/property_graph
```

2. Set BDSG_CLASSPATH to the `solrcloud` directory. For example:

```
export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/
property_graph/solrcloud/*:$BDSG_CLASSPATH
```

3. Compile the Java code. For example:

```
javac -classpath $BDSG_HOME/lib/'*':$BDSG_CLASSPATH filename.java
```

4. Run the Java application by executing the compiled code. For example::

```
java -classpath ./:$BDSG_HOME/lib/'*':$BDSG_CLASSPATH filename args
```

Managing Text Indexing Using a Groovy Console

1. Set BDSG_CLASSPATH to the `solrcloud` directory. For example:

```
export BDSG_CLASSPATH=/opt/oracle/oracle-spatial-graph/
property_graph/solrcloud/*:$BDSG_CLASSPATH
```

2. Start the shell as usual to create a text index over a property graph stored in Apache HBase storage using an `OraclePropertyGraph` instance. For example:

```
cd /opt/oracle/oracle-spatial-graph/property_graph/dal/groovy
sh gremlin-opg-hbase.sh

--------------------------------

opg-hbase> conf = HBaseConfiguration.create();
==>hbase.rs.cacheblocksonwrite=false
==>...
opg-hbase> dop=2;
==>2
opg-hbase> conf.set("hbase.zookeeper.quorum", "localhost");
==>null
opg-hbase> conf.set("hbase.zookeeper.property.clientPort","2181");
==>null
opg-hbase> conn = ConnectionFactory.createConnection(conf);
==>hconnection-0x720653c2
opg-hbase> opg=OraclePropertyGraph.getInstance(conf, conn,
"connections");
==>oraclepropertygraph with name connections
opg-hbase> indexParams =
```

```
OracleIndexParameters.buildSolr("opgconfig" /* solr config */,
"localhost:2181/solr" /* solr server url */, "localhost:8983_solr" /*
solr node set */, 15 /* zookeeper timeout in seconds */, 1 /* total
number of shards */, 1 /* Replication factor */, 1 /* maximum number of
shardsper node */, 4 /* dop used for scan */, 10000 /* batch size before
commit */, 500000 /* commit size before SolrCloud commit */, 15 /* write
timeout in seconds */);
==>[parameter[search-engine,0], parameter[config-name,opgconfig],
parameter[solr-server-url,localhost:2181/solr], parameter[solr-admin-
url,localhost:8983_solr], parameter[zk-timeout,15], parameter[replication-
factor,1], parameter[num-shards,1], parameter[max-shards-per-node,1],
parameter[reindex-numConns,4], parameter[batch-size,10000],
parameter[commit-batch-size,500000], parameter[write-timeout,15]]
opg-hbase>
opg-hbase> opg.setDefaultIndexParameters(indexParams);
==>null


opg-hbase> indexedKeys = new String[4]; indexedKeys[0] = "name";
indexedKeys[1] = "role"; indexedKeys[2] = "religion"; indexedKeys[3] =
"country";
==>name
==>role
==>religion
==>country
opg-hbase> opg.createKeyIndex(indexedKeys, Vertex.class);
==>null
```

1.8 Required Application Code Changes due to Upgrades

Application code changes may be required due to upgrades, such as to more recent versions of Apache HBase and SolrCloud.

- Changes Due to Upgrade from Apache HBase 1.x to Apache HBase 2.x
- Changes Due to Upgrade from SolrCloud 4.10.3 to SolrCloud 7.0.0

1.8.1 Changes Due to Upgrade from Apache HBase 1.x to Apache HBase 2.x

Big Data Spatial and Graph 2.5.3 supports Cloudera CDH6, which upgraded Apache HBase to a newer version.

- Creating a Property Graph Instance
- Parallel Retrieval of Vertices/Edges
- Dropping an Existing Graph

Creating a Property Graph Instance

Effective with Apache HBase 2.0, the `HConnection` interface has been deprecated, so the data acesss layer requires using a Connection object to connect to the database. The following code snippet illustrates how to create an OraclePropertyGraph instance from an Apache HBase 2.0 Connection object.

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.client.*;

...

Configuration conf = HBaseConfiguration.create();
conf.set("hbase.zookeeper.quorum", szQuorum);
conf.set("hbase.zookeeper.property.clientPort","2181");
Connection conn = ConnectionFactory.createConnection(conf);
OraclePropertyGraph opg = OraclePropertyGraph.getInstance(conf, hconn,
szGraphName);
```

Parallel Retrieval of Vertices/Edges

The following code snippet opens an array of connections to HBase (using the `Connection`/`ConnectionFactory` APIs from Apache HBase 2.x), and executes a parallel query to retrieve all vertices and edges using the opened connections. The number of calls to the `getVerticesPartitioned`/`getEdgesPartitioned` method is controlled by the total number of splits and the number of connections used.

```
int dop = 4;
Configuration conf = HBaseConfiguration.create();
conf.set("hbase.zookeeper.quorum", szQuorum);
conf.set("hbase.zookeeper.property.clientPort","2181");
Connection conn = ConnectionFactory.createConnection(conf);
OraclePropertyGraph opg = OraclePropertyGraph.getInstance(conn,
"connections");

// Create connections used in parallel query
Connection[] conns= new Connection[dop];
for (int i = 0; i < dop; i++) {
  Configuration conf_new =
HBaseConfiguration.create(opg.getConfiguration());
  conns[i] = ConnectionFactory.createConnection(conf_new);
}

long lCountV = 0;
// Iterate over all the vertices¿ splits to count all the vertices
for (int split = 0; split < opg.getVertexTableSplits(); split += dop)
{
  Iterable<Vertex>[] iterables = opg.getVerticesPartitioned(conns /*
Connection array */,
                                                            true /*
skip store to cache */,
                                                            split /*
starting split */);


  for (Iterable<Vertex> iterable : iterables) {
    lCountV += OraclePropertyGraphUtils.size(iterable); /* consume
iterables */
```

```
  }
}

// Count all vertices
System.out.println("Vertices found using parallel query: " + lCountV);

long lCountE = 0;
// Iterate over all the edges¿ splits to count all the edges
for (int split = 0; split < opg.getEdgeTableSplits();  split += dop) {
  Iterable<Edge>[] iterables = opg.getEdgesPartitioned(conns /* Connection 
array */,
                                                      true /* skip store to 
cache */,
                                                      split /* starting 
split */);

  for (Iterable<Vertex> iterable : iterables) {
    lCountE += consumeIterables(iterables); /* consume iterables */
  }
}

// Count all edges
System.out.println("Edges found using parallel query: " + lCountE);

// Close the connections to the database after completed
for (int idx = 0; idx < conns.length; idx++) {
  conns[idx].close();
}
```

Dropping an Existing Graph

For Apache HBase 2.x, the `OraclePropertyGraphUtils.dropPropertyGraph` method uses the Hadoop nodes and the Apache HBase port number for the connection. The following code fragment deletes a graph named `my_graph` from Apache HBase 2.x.

```
int dop = 4;
Configuration conf = HBaseConfiguration.create();
conf.set("hbase.zookeeper.quorum", szQuorum);
conf.set("hbase.zookeeper.property.clientPort","2181");
Connection conn = ConnectionFactory.createConnection(conf);
OraclePropertyGraphUtils.dropPropertyGraph(conn, "my_graph");
```

1.8.2 Changes Due to Upgrade from SolrCloud 4.10.3 to SolrCloud 7.0.0

The upgrade from SolrCloud 4.10.3 to SolrCloud 7.0.0 may require some application code changes.

- Parallel Query on Text Indexes for Property Graph Data
- Using Native Query Results with SolrCloud

Parallel Query on Text Indexes for Property Graph Data

With SolrCloud 7.0, the `SolrCloudServer` interface has been deprecated, so the data acess layer requires using a `CloudSolrClient` object to connect to SolrCloud text search engine. In order to execute parallel queries over a SolrCloud-based text index, you must specify a set of `CloudSolrClient` instances. To create a `CloudSolrClient` instance, you can rely on the `SolrIndexUtils.getCloudSolrClient` API, because the operation `SolrIndexUtils.getCloudSolrServer` is now deprecated

The following code snippet generates an automatic text index using the SolrCloud Search engine and executes a parallel text query. The number of calls to the `getPartitioned` method in the `SolrIndex` class is controlled by the total number of shards in the index and the number of connections used.

```
OraclePropertyGraph opg = OraclePropertyGraph.getInstance(args,
szGraphName);


String configName = "opgconfig";
String solrServerUrl = args[4];//"localhost:2181/solr"
String solrNodeSet = args[5]; //"localhost:8983_solr";

int zkTimeout = 15; // zookeeper timeout in seconds
int numShards = Integer.parseInt(args[6]); // number of shards in the
index
int replicationFactor = 1; // replication factor
int maxShardsPerNode = 1; // maximum number of shards per node

// Create an automatic index using SolrCloud
OracleIndexParameters indexParams =
OracleIndexParameters.buildSolr(configName, solrServerUrl,
solrNodeSet, zkTimeout /* zookeeper timeout in seconds */,
numShards /* total number of shards */, replicationFactor /*
Replication factor */, maxShardsPerNode /* maximum number of shardsper
node*/, 4 /* dop used for scan */, 10000 /* batch size before
commit*/, 500000 /* commit size before SolrCloud commit*/, 15 /* write
timeout in seconds */);

opg.setDefaultIndexParameters(indexParams);

// Create auto indexing on name property for all vertices
System.out.println("Create automatic index on name for vertices");
opg.createKeyIndex("name", Vertex.class);

// Get the SolrIndex object
SolrIndex<Vertex> index = (SolrIndex<Vertex>)
opg.getAutoIndex(Vertex.class);

// Open an array of connections to handle connections to SolrCloud
needed for parallel text search
CloudSolrClient[] conns = new CloudSolrClient[dop];


for (int idx = 0; idx < conns.length; idx++) {
  conns[idx] = index.getCloudSolrClient(15 /* write timeout in secs*/);
}

// Iterate to cover all the shards in the index
long lCount = 0;
```

```
for (int split = 0; split < index.getTotalShards(); split += conns.length) {
  // Gets elements from split to split + conns.length
  Iterable<Vertex>[] iterAr = index.getPartitioned(conns /* connections */,
"name"/* key */, "*" /* value */, true /* wildcards */, split /* start split
ID */);
  for (Iterable<Vertex> iterable : iterables) {
    lCount += OraclePropertyGraphUtils.size(iterable); /* consume iterables
*/
  }
}

// Close the connections to SolrCloud after completed
for (int idx = 0; idx < conns.length; idx++) {
  conns[idx].close();
}
```

Using Native Query Results with SolrCloud

You can use native query results using SolrCloud by calling the method `get(QueryResponse)` in `SolrIndex`. A `QueryResponse` object provides a set of Documents matching a text search query over a specific SolrCloud collection. `SolrIndex` will produce an Iterable object holding all the vertices (or edges) from the documents found in the `QueryResponse` object.

With SolrCloud 7.0, the `SolrCloudServer` interface has been deprecated, so the data access layer requires use of a `CloudSolrClient` object to process native query results over a text index in Oracle Property Graph. The following code fragment generates an automatic text index using the Apache SolrCloud Search engine, creates a `SolrQuery` object, and executes it against a `CloudSolrClient` object to get a QueryResponse object. Later, an Iterable object of vertices is created from the given result object.

```
OraclePropertyGraph opg = OraclePropertyGraph.getInstance(args, szGraphName);

 String configName = "opgconfig";
String solrServerUrl = args[4];//"localhost:2181/solr"
String solrNodeSet = args[5]; //"localhost:8983_solr";

int zkTimeout = 15; // zookeeper timeout in seconds
int numShards = Integer.parseInt(args[6]); // number of shards in the index
int replicationFactor = 1; // replication factor
int maxShardsPerNode = 1; // maximum number of shards per node

// Create an automatic index using SolrCloud
OracleIndexParameters indexParams =
OracleIndexParameters.buildSolr(configName, solrServerUrl, solrNodeSet,
zkTimeout /* zookeeper timeout in seconds */, numShards /* total number of
shards */, replicationFactor /* Replication factor */, maxShardsPerNode /*
maximum number of shardsper node*/, 4 /* dop used for scan */, 10000 /*
batch size before commit*/, 500000 /* commit size before SolrCloud commit*/,
15 /* write timeout in seconds */);

opg.setDefaultIndexParameters(indexParams);

 // Create auto indexing on name property for all vertices
System.out.println("Create automatic index on name and country for
```

```
vertices"); String[] indexedKeys = new String[2];
indexedKeys[0]="name"; indexedKeys[1]="country";
opg.createKeyIndex(indexedKeys, Vertex.class);

 // Get the SolrIndex object
SolrIndex<Vertex> index = (SolrIndex<Vertex>)
opg.getAutoIndex(Vertex.class);

 // Search first for Key name with property value Beyon* using only
string data types
String szQueryStrBey = index.buildSearchTerm("name", "Beyo*",
String.class);
String key = index.appendDatatypesSuffixToKey("country", String.class);
String value = index.appendDatatypesSuffixToValue("United States",
String.class);
String szQueryStrCountry = key + ":" + value;
Solrquery query = new SolrQuery(szQueryStrBey + " AND " +
szQueryStrCountry);

 CloudSolrClient conn = index.getCloudSolrClient(15 /* write timeout
in secs*/);

//Query using get operation
QueryResponse qr = conn.query(query, SolrRequest.METHOD.POST);

Iterable<Vertex> it = index.get(qr);
long lCount = 0;
while (it.hasNext()) {
  System.out.println(it.next());
  lCount++;
}

System.out.println("Vertices found: "+ lCount);
```

2 Using Big Data Spatial and Graph with Spatial Data

This chapter provides conceptual and usage information about loading, storing, accessing, and working with spatial data in a Big Data environment.

- About Big Data Spatial and Graph Support for Spatial Data
  Oracle Big Data Spatial and Graph features enable spatial data to be stored, accessed, and analyzed quickly and efficiently for location-based decision making.
- Oracle Big Data Vector and Raster Data Processing
  Oracle Big Data Spatial and Graph supports the storage and processing of both vector and raster spatial data.
- Oracle Big Data Spatial Hadoop Image Processing Framework for Raster Data Processing
  Oracle Spatial Hadoop Image Processing Framework allows the creation of new combined images resulting from a series of processing phases in parallel.
- Loading an Image to Hadoop Using the Image Loader
  The first step to process images using the Oracle Spatial and Graph Hadoop Image Processing Framework is to actually have the images in HDFS, followed by having the images separated into smart tiles.
- Processing an Image Using the Oracle Spatial Hadoop Image Processor
  Once the images are loaded into HDFS, they can be processed in parallel using Oracle Spatial Hadoop Image Processing Framework.
- Loading and Processing an Image Using the Oracle Spatial Hadoop Raster Processing API
  The framework provides a raster processing API that lets you load and process rasters without creating XML but instead using a Java application. The application can be executed inside the cluster or on a remote node.
- Using the Oracle Spatial Hadoop Raster Simulator Framework to Test Raster Processing
  When you create custom processing classes. you can use the Oracle Spatial Hadoop Raster Simulator Framework to do the following by "pretending" to plug them into the Oracle Raster Processing Framework.
- Oracle Big Data Spatial Raster Processing for Spark
  Oracle Big Data Spatial Raster Processing for Apache Spark is a spatial raster processing API for Java.
- Spatial Raster Processing Support in Big Data Cloud Service
  Oracle Big Data Spatial Raster Processing is supported in Big Data Cloud Service (BDCS) by making use of the Oracle Object Storage platform.
- Oracle Big Data Spatial Vector Analysis
  Oracle Big Data Spatial Vector Analysis is a Spatial Vector Analysis API, which runs as a Hadoop job and provides MapReduce components for spatial processing of data stored in HDFS.

- Oracle Big Data Spatial Vector Analysis for Spark
  Oracle Big Data Spatial Vector Analysis for Apache Spark is a spatial vector analysis API for Java and Scala that provides spatially-enabled RDDs (Resilient Distributed Datasets) that support spatial transformations and actions, spatial partitioning, and indexing.
- Oracle Big Data Spatial Vector Hive Analysis
  Oracle Big Data Spatial Vector Hive Analysis provides spatial functions to analyze the data using Hive.
- Using the Oracle Big Data SpatialViewer Web Application
  You can use the Oracle Big Data SpatialViewer Web Application (SpatialViewer) to perform a variety of tasks.

2.1 About Big Data Spatial and Graph Support for Spatial Data

Oracle Big Data Spatial and Graph features enable spatial data to be stored, accessed, and analyzed quickly and efficiently for location-based decision making.

Spatial data represents the location characteristics of real or conceptual objects in relation to the real or conceptual space on a Geographic Information System (GIS) or other location-based application.

The spatial features are used to geotag, enrich, visualize, transform, load, and process the location-specific two and three dimensional geographical images, and manipulate geometrical shapes for GIS functions.

- What is Big Data Spatial and Graph on Apache Hadoop?
- Advantages of Oracle Big Data Spatial and Graph
- Oracle Big Data Spatial Features and Functions
- Oracle Big Data Spatial Files, Formats, and Software Requirements

2.1.1 What is Big Data Spatial and Graph on Apache Hadoop?

Oracle Big Data Spatial and Graph on Apache Hadoop is a framework that uses the MapReduce programs and analytic capabilities in a Hadoop cluster to store, access, and analyze the spatial data. The spatial features provide a schema and functions that facilitate the storage, retrieval, update, and query of collections of spatial data. Big Data Spatial and Graph on Hadoop supports storing and processing spatial images, which could be geometric shapes, raster, or vector images and stored in one of the several hundred supported formats.

> **Note:**
>
> *Oracle Spatial and Graph Developer's Guide* for an introduction to spatial concepts, data, and operations

2.1.2 Advantages of Oracle Big Data Spatial and Graph

The advantages of using Oracle Big Data Spatial and Graph include the following:

- Unlike some of the GIS-centric spatial processing systems and engines, Oracle Big Data Spatial and Graph is capable of processing both structured and unstructured spatial information.
- Customers are not forced or restricted to store only one particular form of data in their environment. They can have their data stored both as a spatial or nonspatial business data and still can use Oracle Big Data to do their spatial processing.
- This is a framework, and therefore customers can use the available APIs to custom-build their applications or operations.
- Oracle Big Data Spatial can process both vector and raster types of information and images.

2.1.3 Oracle Big Data Spatial Features and Functions

The spatial data is loaded for query and analysis by the Spatial Server and the images are stored and processed by an Image Processing Framework. You can use the Oracle Big Data Spatial and Graph server on Hadoop for:

- Cataloguing the geospatial information, such as geographical map-based footprints, availability of resources in a geography, and so on.
- Topological processing to calculate distance operations, such as nearest neighbor in a map location.
- Categorization to build hierarchical maps of geographies and enrich the map by creating demographic associations within the map elements.

The following functions are built into Oracle Big Data Spatial and Graph:

- Indexing function for faster retrieval of the spatial data.
- Map function to display map-based footprints.
- Zoom function to zoom-in and zoom-out specific geographical regions.
- Mosaic and Group function to group a set of image files for processing to create a mosaic or subset operations.
- Cartesian and geodetic coordinate functions to represent the spatial data in one of these coordinate systems.
- Hierarchical function that builds and relates geometric hierarchy, such as country, state, city, postal code, and so on. This function can process the input data in the form of documents or latitude/longitude coordinates.

2.1.4 Oracle Big Data Spatial Files, Formats, and Software Requirements

The stored spatial data or images can be in one of these supported formats:

- GeoJSON files
- Shapefiles
- Both Geodetic and Cartesian data

- Other GDAL supported formats

You must have the following software, to store and process the spatial data:

- Java runtime
- GCC Compiler - Only when the GDAL-supported formats are used

2.2 Oracle Big Data Vector and Raster Data Processing

Oracle Big Data Spatial and Graph supports the storage and processing of both vector and raster spatial data.

- Oracle Big Data Spatial Raster Data Processing
- Oracle Big Data Spatial Vector Data Processing

2.2.1 Oracle Big Data Spatial Raster Data Processing

For processing the raster data, the GDAL loader loads the raster spatial data or images onto a HDFS environment. The following basic operations can be performed on a raster spatial data:

- Mosaic: Combine multiple raster images to create a single mosaic image.
- Subset: Perform subset operations on individual images.
- Raster algebra operations: Perform algebra operations on every pixel in the rasters (for example, add, divide, multiply, log, pow, sine, sinh, and acos).
- User-specified processing: Raster processing is based on the classes that user sets to be executed in mapping and reducing phases.

This feature supports a MapReduce framework for raster analysis operations. The users have the ability to custom-build their own raster operations, such as performing an algebraic function on a raster data and so on. For example, calculate the slope at each base of a digital elevation model or a 3D representation of a spatial surface, such as a terrain. For details, see Oracle Big Data Spatial Hadoop Image Processing Framework for Raster Data Processing.

2.2.2 Oracle Big Data Spatial Vector Data Processing

This feature supports the processing of spatial vector data:

- Loaded and stored on to a Hadoop HDFS environment
- Stored either as Cartesian or geodetic data

The stored spatial vector data can be used for performing the following query operations and more:

- Point-in-polygon
- Distance calculation
- Anyinteract
- Buffer creation

Sevetal data service operations are supported for the spatial vector data:

- Data enrichment

- Data categorization
- Spatial join

In addition, there is a limited Map Visualization API support for only the HTML5 format. You can access these APIs to create custom operations. For details, see "Oracle Big Data Spatial Vector Analysis."

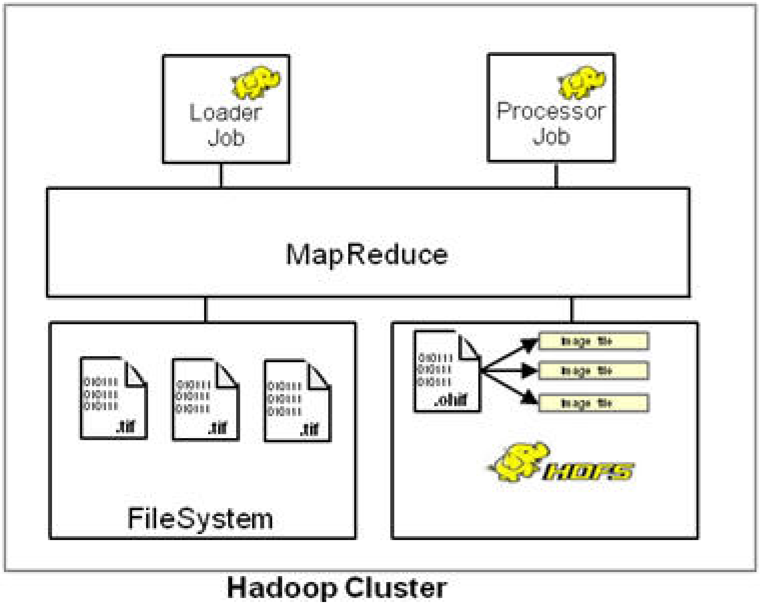
2.3 Oracle Big Data Spatial Hadoop Image Processing Framework for Raster Data Processing

Oracle Spatial Hadoop Image Processing Framework allows the creation of new combined images resulting from a series of processing phases in parallel.

It includes the following features:

- HDFS Images storage, where every block size split is stored as a separate tile, ready for future independent processing
- Subset, user-defined, and map algebra operations processed in parallel using the MapReduce framework
- Ability to add custom processing classes to be executed in the mapping or reducing phases in parallel in a transparent way
- Fast processing of georeferenced images
- Support for GDAL formats, multiple bands images, DEMs (digital elevation models), multiple pixel depths, and SRIDs
- Java API providing access to framework operations; useful for web services or standalone Java applications
- Framework for testing and debugging user processing classes in the local environment

The Oracle Spatial Hadoop Image Processing Framework consists of two modules, a Loader and Processor, each one represented by a Hadoop job running on different stages in a Hadoop cluster, as represented in the following diagram. Also, you can load and process the images using the SpatialViewer web application, and you can use the Java API to expose the framework’s capabilities.



For installation and configuration information, see:

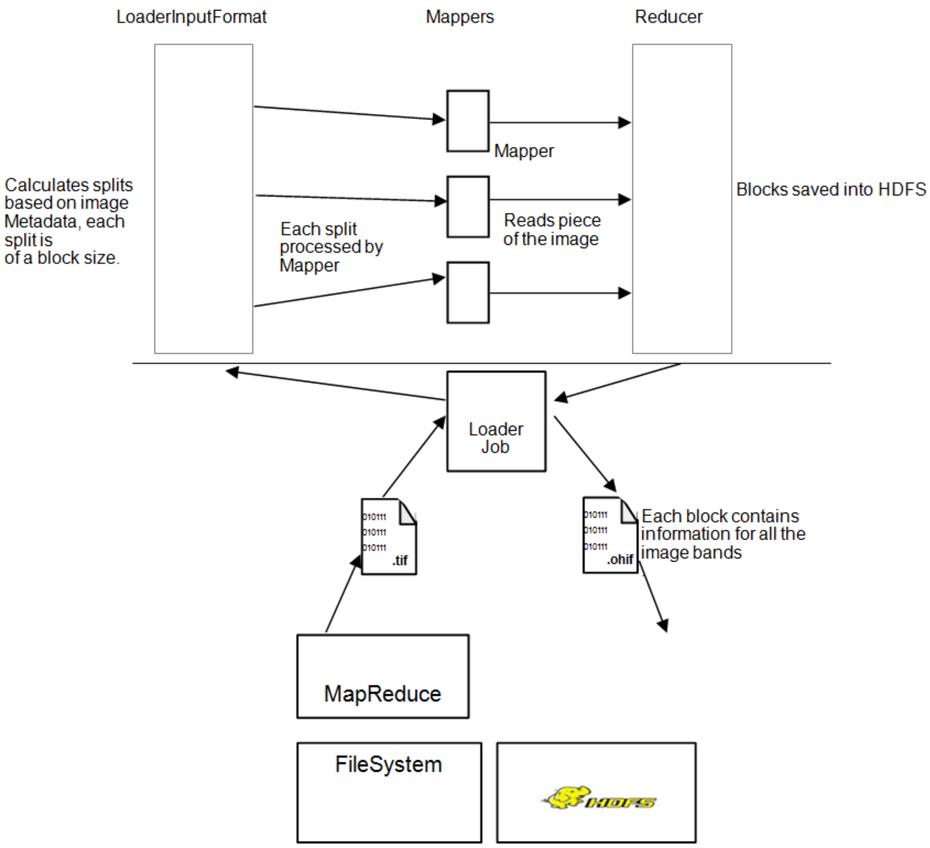
- Installing Oracle Big Data Spatial and Graph on an Oracle Big Data Appliance
- Installing and Configuring the Big Data Spatial Image Processing Framework
- Image Loader
- Image Processor

2.3.1 Image Loader

The Image Loader is a Hadoop job that loads a specific image or a group of images into HDFS.

- While importing, the image is tiled and stored as an HDFS block.
- GDAL is used to tile the image.
- Each tile is loaded by a different mapper, so reading is parallel and faster.
- Each tile includes a certain number of overlapping bytes (user input), so that the tiles cover area from the adjacent tiles.
- A MapReduce job uses a mapper to load the information for each tile. There are 'n' number of mappers, depending on the number of tiles, image resolution and block size.
- A single reduce phase per image puts together all the information loaded by the mappers and stores the images into a special `.ohif` format, which contains the resolution, bands, offsets, and image data. This way the file offset containing each tile and the node location is known.
- Each tile contains information for every band. This is helpful when there is a need to process only a few tiles; then, only the corresponding blocks are loaded.

The following diagram represents an Image Loader process:

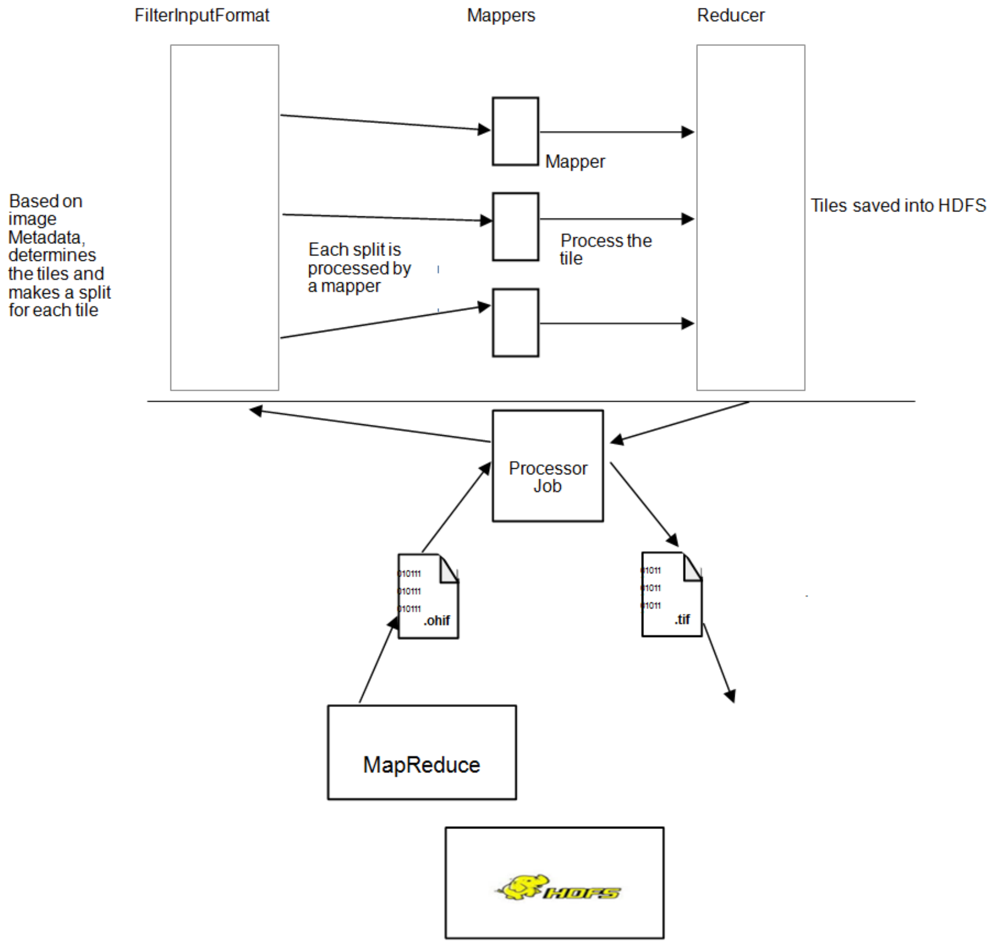


2.3.2 Image Processor

The Image Processor is a Hadoop job that filters tiles to be processed based on the user input and performs processing in parallel to create a new image.

- Processes specific tiles of the image identified by the user. You can identify one, zero, or multiple processing classes. These classes are executed in the mapping or reducing phase, depending on your configuration. For the mapping phase, after the execution of processing classes, a mosaic operation is performed to adapt the pixels to the final output format requested by the user. If no mosaic operation was requested, the input raster is sent to reduce phase as is. For reducer phase, all the tiles are put together into a GDAL data set that is input for user reduce processing class, where final output may be changed or analyzed according to user needs.
- A mapper loads the data corresponding to one tile, conserving data locality.
- Once the data is loaded, the mapper filters the bands requested by the user.
- Filtered information is processed and sent to each mapper in the reduce phase, where bytes are put together and a final processed image is stored into HDFS or regular File System depending on the user request.

The following diagram represents an Image Processor job:



2.4 Loading an Image to Hadoop Using the Image Loader

The first step to process images using the Oracle Spatial and Graph Hadoop Image Processing Framework is to actually have the images in HDFS, followed by having the images separated into smart tiles.

This allows the processing job to work separately on each tile independently. The Image Loader lets you import a single image or a collection of them into HDFS in parallel, which decreases the load time.

The Image Loader imports images from a file system into HDFS, where each block contains data for all the bands of the image, so that if further processing is required on specific positions, the information can be processed on a single node.

- Image Loading Job
- Input Parameters
- Output Parameters

2.4.1 Image Loading Job

The image loading job has its custom input format that splits the image into related image splits. The splits are calculated based on an algorithm that reads square blocks of the image covering a defined area, which is determined by

area = ((blockSize - metadata bytes) / number of bands) / bytes per pixel.

For those pieces that do not use the complete block size, the remaining bytes are refilled with zeros.

Splits are assigned to different mappers where every assigned tile is read using GDAL based on the `ImageSplit` information. As a result an `ImageDataWritable` instance is created and saved in the context.

The metadata set in the `ImageDataWritable` instance is used by the processing classes to set up the tiled image in order to manipulate and process it. Since the source images are read from multiple mappers, the load is performed in parallel and faster.

After the mappers finish reading, the reducer picks up the tiles from the context and puts them together to save the file into HDFS. A special reading process is required to read the image back.

2.4.2 Input Parameters

The following input parameters are supplied to the Hadoop command:

```
hadoop jar /opt/oracle/oracle-spatial-graph/spatial/raster/jlib/hadoop-imageloader.jar
  -files <SOURCE_IMGS_PATH>
  -out <HDFS_OUTPUT_FOLDER>
  -gdal <GDAL_LIB_PATH>
  -gdalData <GDAL_DATA_PATH>
  [-overlap <OVERLAPPING_PIXELS>]
  [-thumbnail <THUMBNAIL_PATH>]
  [-expand <false|true>]
  [-extractLogs <false|true>]
  [-logFilter <LINES_TO_INCLUDE_IN_LOG>]
  [-pyramid <OUTPUT_DIRECTORY, LEVEL, [RESAMPLING]>]
```

Where:

`SOURCE_IMGS_PATH` is a path to the source image(s) or folder(s). For multiple inputs use a comma separator. This path must be accessible via NFS to all nodes in the cluster.

`HDFS_OUTPUT_FOLDER` is the HDFS output folder where the loaded images are stored.

`OVERLAPPING_PIXELS` is an optional number of overlapping pixels on the borders of each tile, if this parameter is not specified a default of two overlapping pixels is considered.

`GDAL_LIB_PATH` is the path where GDAL libraries are located.

`GDAL_DATA_PATH` is the path where GDAL data folder is located. This path must be accessible through NFS to all nodes in the cluster.

`THUMBNAIL_PATH` is an optional path to store a thumbnail of the loaded image(s). This path must be accessible through NFS to all nodes in the cluster and must have write access permission for yarn users.

`-expand` controls whether the HDFS path of the loaded raster expands the source path, including all directories. If you set this to `false`, the `.ohif` file is stored directly in the output directory (specified using the `-o` option) without including that directory's path in the raster.

`-extractLogs` controls whether the logs of the executed application should be extracted to the system temporary directory. By default, it is not enabled. The extraction does not include logs that are not part of Oracle Framework classes.

`-logFilter <LINES_TO_INCLUDE_IN_LOG>` is a comma-separated String that lists all the patterns to include in the extracted logs, for example, to include custom processing classes packages.

`-pyramid <OUTPUT_DIRECTORY, LEVEL, [RESAMPLING]>` allows the creation of pyramids while making the initial raster load. An OUPUT_DIRECTORY must be provided to store the local pyramids before uploading to HDFS; pyramids are loaded in the same HDFSA directory requested for load. A pyramid LEVEL must be provided to indicate how many pyramids are required for each raster. A RESAMPLING algorithm is optional to specify the method used to execute the resampling; if none is set, then `BILINEAR` is used.

For example, the following command loads all the georeferenced images under the `images` folder and adds an overlapping of 10 pixels on every border possible. The HDFS output folder is `ohiftest` and thumbnail of the loaded image are stored in the `processtest` folder.

```
hadoop jar /opt/oracle/oracle-spatial-graph/spatial/raster/jlib/hadoop-
imageloader.jar   -files /opt/shareddir/spatial/demo/imageserver/images/
hawaii.tif -out ohiftest -overlap 10 -thumbnail /opt/shareddir/spatial/
processtest -gdal /opt/oracle/oracle-spatial-graph/spatial/raster/gdal/lib -
gdalData /opt/shareddir/data
```

By default, the Mappers and Reducers are configured to get 2 GB of JVM, but users can override this settings or any other job configuration properties by adding an `imagejob.prop` properties file in the same folder location from where the command is being executed. This properties file may list all the configuration properties that you want to override. For example,

```
mapreduce.map.memory.mb=2560
mapreduce.reduce.memory.mb=2560
mapreduce.reduce.java.opts=-Xmx2684354560
mapreduce.map.java.opts=-Xmx2684354560
```

Java heap memory (`java.opts` properties) must be equal to or less than the total memory assigned to mappers and reducers (`mapreduce.map.memory` and `mapreduce.reduce.memory`). Thus, if you increase Java heap memory, you might also need to increase the memory for mappers and reducers.

For GDAL to work properly, the libraries must be available using $LD_LIBRARY_PATH. Make sure that the shared libraries path is set properly in your shell window before executing a job. For example:

```
export LD_LIBRARY_PATH=$ALLACCESSDIR/gdal/native
```

2.4.3 Output Parameters

The reducer generates two output files per input image. The first one is the `.ohif` file that concentrates all the tiles for the source image, each tile may be processed as a separated instance by a processing mapper. Internally each tile is stored as a HDFS block, blocks are located in several nodes, one node may contain one or more blocks of a specific `.ohif` file. The `.ohif` file is stored in user specified folder with -out flag, under the `/user/<USER_EXECUTING_JOB>/OUT_FOLDER/<PARENT_DIRECTORIES_OF_SOURCE_RASTER>` if the flag `-expand` was not used. Otherwise, the `.ohif` file will be located at `/user/<USER_EXECUTING_JOB>/OUT_FOLDER/`, and the file can be identified as `original_filename.ohif`.

The second output is a related metadata file that lists all the pieces of the image and the coordinates that each one covers. The file is located in HDFS under the metadata location, and its name is hash generated using the name of the `ohif` file. This file is for Oracle internal use only, and lists important metadata of the source raster. Some example lines from a metadata file:

```
srid:26904
datatype:1
resolution:27.90809458890406,-27.90809458890406
file:/user/hdfs/ohiftest/opt/shareddir/spatial/data/rasters/hawaii.tif.ohif
bands:3
mbr:532488.7648166901,4303164.583549625,582723.3350767174,4269619.053853762
0,532488.7648166901,4303164.583549625,582723.3350767174,4269619.053853762
thumbnailpath:/opt/shareddir/spatial/thumb/
```

If the `-thumbnail` flag was specified, a thumbnail of the source image is stored in the related folder. This is a way to visualize a translation of the `.ohif` file. Job execution logs can be accessed using the command `yarn logs -applicationId <applicationId>`.

2.5 Processing an Image Using the Oracle Spatial Hadoop Image Processor

Once the images are loaded into HDFS, they can be processed in parallel using Oracle Spatial Hadoop Image Processing Framework.

You specify an output, and the framework filters the tiles to fit into that output, processes them, and puts them all together to store them into a single file. Map algebra operations are also available and, if set, will be the first part of the processing phase. You can specify additional processing classes to be executed before the final output is created by the framework.

The image processor loads specific blocks of data, based on the input (mosaic description or a single raster), and selects only the bands and pixels that fit into the final output. All the specified processing classes are executed and the final output is stored into HDFS or the file system depending on the user request.

- Image Processing Job
- Input Parameters
- Job Execution
- Processing Classes and ImageBandWritable
- Map Algebra Operations
- Multiple Raster Algebra Operations
- Pyramids
- Output

2.5.1 Image Processing Job

The image processing job has different flows depending on the type of processing requested by the user.

- Default Image Processing Job Flow: executed for processing that includes a mosaic operation, single raster operation, or basic multiple raster operation.
- Multiple Raster Image Processing Job Flow: executed for processing that includes complex multiple raster algebra operations.
- Default Image Processing Job Flow
- Multiple Raster Image Processing Job Flow

2.5.1.1 Default Image Processing Job Flow

The default image processing job flow is executed when any of the following processing is requested:

- Mosaic operation
- Single raster operation
- Basic multiple raster algebra operation

The flow has its own custom `FilterInputFormat`, which determines the tiles to be processed, based on the SRID and coordinates. Only images with same data type (pixel depth) as the mosaic input data type (pixel depth) are considered. Only the tiles that intersect with coordinates specified by the user for the mosaic output are included. For processing of a single raster or basic multiple raster algebra operation (excluding mosaic), the filter includes all the tiles of the input rasters, because the processing will be executed on the complete images. Once the tiles are selected, a custom `ImageProcessSplit` is created for each image.

When a mapper receives the `ImageProcessSplit`, it reads the information based on what the `ImageSplit` specifies, performs a filter to select only the bands indicated by the user, and executes the list of map operations and of processing classes defined in the request, if any.

Each mapper process runs in the node where the data is located. After the map algebra operations and processing classes are executed, a validation verifies if the user is requesting mosaic operation or if analysis includes the complete image; and if a mosaic operation is requested, the final process executes the operation. The mosaic operation selects from every tile only the pixels that fit into the output and makes the necessary resolution changes to add them in the mosaic output. The single process operation just copies the previous raster tile bytes as they are. The resulting bytes are stored in NFS to be recovered by the reducer.

A single reducer picks the tiles and puts them together. If you specified any basic multiple raster algebra operation, then it is executed at the same time the tiles are merged into the final output. This operation affects only the intersecting pixels in the mosaic output, or in every pixel if no mosaic operation was requested. If you specified a reducer processing class, the GDAL data set with the output raster is sent to this class for analysis and processing. If you selected HDFS output, the `ImageLoader` is called to store the result into HDFS. Otherwise, by default the image is prepared using GDAL and is stored in the file system (NFS).

2.5.1.2 Multiple Raster Image Processing Job Flow

The multiple raster image processing job flow is executed when a complex multiple raster algebra operation is requested. It applies to rasters that have the same MBR, pixel type, pixel size, and SRID, since these operations are applied pixel by pixel in the corresponding cell, where every pixel represents the same coordinates.

The flow has its own custom `MultipleRasterInputFormat`, which determines the tiles to be processed, based on the SRID and coordinates. Only images with same MBR, pixel type, pixel size and SRID are considered. Only the rasters that match with coordinates specified by the first raster in the catalog are included. All the tiles of the input rasters are considered, because the processing will be executed on the complete images.

Once the tiles are selected, a custom `MultipleRasterSplit` is created. This split contains a small area of every original tile, depending on the block size, because now all the rasters must be included in a split, even if it is only a small area. Each of these is called an `IndividualRasterSplit`, and they are contained in a parent `MultipleRasterSplit`.

When a mapper receives the `MultipleRasterSplit`, it reads the information of all the raster´s tiles that are included in the parent split, performs a filter to select only the bands indicated by the user and only the small corresponding area to process in this specific mapper, and then executes the complex multiple raster algebra operation.

Data locality may be lost in this part of the process, because multiple rasters are included for a single mapper that may not be in the same node. The resulting bytes for every pixel are put in the context to be recovered by the reducer.

A single reducer picks pixel values and puts them together. If you specified a reducer processing class, the GDAL data set with the output raster is sent to this class for analysis and processing. The list of tiles that this class receives is null for this scenario, and the class can only work with the output data set. If you selected HDFS output, the `ImageLoader` is called to store the result into HDFS. Otherwise, by default the image is prepared using GDAL and is stored in the file system (NFS).

2.5.2 Input Parameters

The following input parameters can be supplied to the hadoop command:

```
hadoop jar /opt/oracle/oracle-spatial-graph/spatial/raster/jlib/hadoop-
imageprocessor.jar
  -config  <MOSAIC_CONFIG_PATH>
  -gdal  <GDAL_LIBRARIES_PATH>
  -gdalData  <GDAL_DATA_PATH>
  [-catalog  <IMAGE_CATALOG_PATH>]
  [-usrlib  <USER_PROCESS_JAR_PATH>]
  [-thumbnail  <THUMBNAIL_PATH>]
  [-nativepath  <USER_NATIVE_LIBRARIES_PATH>]
  [-params  <USER_PARAMETERS>]
  [-file  <SINGLE_RASTER_PATH>]
```

Where:

`MOSAIC_CONFIG_PATH` is the path to the mosaic configuration xml, that defines the features of the output.

`GDAL_LIBRARIES_PATH` is the path where GDAL libraries are located.

`GDAL_DATA_PATH` is the path where the GDAL data folder is located. This path must be accessible via NFS to all nodes in the cluster.

`IMAGE_CATALOG_PATH` is the path to the catalog xml that lists the HDFS image(s) to be processed. This is optional because you can also specify a single raster to process using `-file` flag.

`USER_PROCESS_JAR_PATH` is an optional user-defined jar file or comma-separated list of jar files, each of which contains additional processing classes to be applied to the source images.

`THUMBNAIL_PATH` is an optional flag to activate the thumbnail creation of the loaded image(s). This path must be accessible via NFS to all nodes in the cluster and is valid only for an HDFS output.

`USER_NATIVE_LIBRARIES_PATH` is an optional comma-separated list of additional native libraries to use in the analysis. It can also be a directory containing all the native libraries to load in the application.

`USER_PARAMETERS` is an optional key/value list used to define input data for user processing classes. Use a semicolon to separate parameters. For example: `azimuth=315;altitude=45`

`SINGLE_RASTER_PATH` is an optional path to the `.ohif` file that will be processed by the job. If this is set, you do not need to set a catalog.

For example, the following command will process all the files listed in the catalog file `input.xml` file using the mosaic output definition set in `testFS.xml` file.

```
hadoop jar /opt/oracle/oracle-spatial-graph/spatial/raster/jlib/hadoop-
imageprocessor.jar -catalog /opt/shareddir/spatial/demo/imageserver/images/
input.xml -config /opt/shareddir/spatial/demo/imageserver/images/testFS.xml -
thumbnail /opt/shareddir/spatial/processtest -gdal /opt/oracle/oracle-spatial-
graph/spatial/raster/gdal/lib -gdalData /opt/shareddir/data
```

By default, the Mappers and Reducers are configured to get 2 GB of JVM, but users can override this settings or any other job configuration properties by adding an `imagejob.prop` properties file in the same folder location from where the command is being executed.

For GDAL to work properly, the libraries must be available using $LD_LIBRARY_PATH. Make sure that the shared libraries path is set properly in your shell window before executing a job. For example:

```
export LD_LIBRARY_PATH=$ALLACCESSDIR/gdal/native
```

- Catalog XML Structure
- Mosaic Definition XML Structure

2.5.2.1 Catalog XML Structure

The following is an example of input catalog XML used to list every source image considered for mosaic operation generated by the image processing job.

```
-<catalog>
  -<image>
<raster>/user/hdfs/ohiftest/opt/shareddir/spatial/data/rasters/maui.tif.ohif</
raster>
<bands datatype='1' config='1,2,3'>3</bands>
   </image>
</catalog>
```

A <catalog> element contains the list of <image> elements to process.

Each <image> element defines a source image or a source folder within the <raster> element. All the images within the folder are processed.

The <bands> element specifies the number of bands of the image, The datatype attribute has the raster data type and the config attribute specifies which band should appear in the mosaic output band order. For example: 3,1,2 specifies that mosaic output band number 1 will have band number 3 of this raster, mosaic band number 2 will have source band 1, and mosaic band number 3 will have source band 2. This order may change from raster to raster.

2.5.2.2 Mosaic Definition XML Structure

The following is an example of a mosaic configuration XML used to define the features of the output generated by the image processing job.

```
-<mosaic exec="false">
  -<output>
   <SRID>26904</SRID>
   <directory type="FS">/opt/shareddir/spatial/processOutput</directory>
   <!--directory type="HDFS">newData</directory-->
   <tempFSFolder>/opt/shareddir/spatial/tempOutput</tempFSFolder>
   <filename>littlemap</filename>
   <format>GTIFF</format>
  <width>1600</width>
  <height>986</height>
  <algorithm order="0">2</algorithm>
  <bands layers="3" config="3,1,2"/>
  <nodata>#000000</nodata>
  <pixelType>1</pixelType>
  </output>
  -<crop>
   -<transform>
    356958.985610072,280.38843650364862,0,2458324.0825054757,0,-280.38843650364862 </
transform>
  </crop>
<process><classMapper
params="threshold=454,2954">oracle.spatial.hadoop.twc.FarmTransformer</
classMapper><classReducer
params="plot_size=100400">oracle.spatial.hadoop.twc.FarmAlignment</classReducer></
process>
   <operations>
        <localif operator="<" operand="3" newvalue="6"/>
                   <localadd arg="5"/>
                   <localsqrt/>
                   <localround/>
   </operations>
</mosaic>
```

The <mosaic> element defines the specifications of the processing output. The exec attribute specifies if the processing will include mosaic operation or not. If set to "false", a mosaic operation is not executed and a single raster is processed; if set to "true" or not set, a mosaic operation is performed. Some of the following elements are required only for mosaic operations and ignored for single raster processing.

The <output> element defines the features such as <SRID> considered for the output. All the images in different SRID are converted to the mosaic SRID in order to decide if any of its tiles fit into the mosaic or not. This element is not required for single raster processing, because the output rster has the same SRID as the input.

The <directory> element defines where the output is located. It can be in an HDFS or in regular FileSystem (FS), which is specified in the tag type.

The <tempFsFolder> element sets the path to store the mosaic output temporarily. The attribute delete=”false” can be specified to keep the output of the process even if the loader was executed to store it in HDFS.

The <filename> and <format> elements specify the output filename. <filename> is not required for single raster process; and if it is not specified, the name of the input file (determined by the -file attribute during the job call) is used for the output file. <format> is not required for single raster processing, because the output raster has the same format as the input.

The <width> and <height> elements set the mosaic output resolution. They are not required for single raster processing, because the output raster has the same resolution as the input.

The <algorithm> element sets the order algorithm for the images. A 1 order means, by source last modified date, and a 2 order means, by image size. The order tag represents ascendant or descendant modes. (These properties are for mosaic operations where multiple rasters may overlap.)

The <bands> element specifies the number of bands in the output mosaic. Images with fewer bands than this number are discarded. The config attribute can be used for single raster processing to set the band configuration for output, because there is no catalog.

The <nodata> element specifies the color in the first three bands for all the pixels in the mosaic output that have no value.

The <pixelType> element sets the pixel type of the mosaic output. Source images that do not have the same pixel size are discarded for processing. This element is not required for single raster processing: if not specified, the pixel type will be the same as for the input.

The <crop> element defines the coordinates included in the mosaic output in the following order: startcoordinateX, pixelXWidth, RotationX, startcoordinateY, RotationY, and pixelheightY. This element is not required for single raster processing: if not specified, the complete image is considered for analysis.

The <process> element lists all the classes to execute before the mosaic operation.

The <classMapper> element is used for classes that will be executed during mapping phase, and the <classReducer> element is used for classes that will be executed during reduce phase. Both elements have the params attribute, where you can send input parameters to processing classes according to your needs.

The <operations> element lists all the map algebra operations that will be processed for this request. This element can also include a request for pyramid operations; for example:

```
<operations>
    <pyramid resampling="NEAREST_NEIGHBOR" redLevel="6"/>
</operations>
```

2.5.3 Job Execution

The first step of the job is to filter the tiles that would fit into the output. As a start, the location files that hold tile metadata are sent to the`InputFormat`.

By extracting the `pixelType`, the filter decides whether the related source image is valid for processing or not. Based on the user definition made in the catalog xml, one of the following happens:

- If the image is valid for processing, then the SRID is evaluated next
- If it is different from the user definition, then the MBR coordinates of every tile are converted into the user SRID and evaluated.

This way, every tile is evaluated for intersection with the output definition.

- For a mosaic processing request, only the intersecting tiles are selected, and a split is created for each one of them.
- For a single raster processing request, all the tiles are selected, and a split is created for each one of them.
- For a complex multiple raster algebra processing request, all the tiles are selected if the MBR and pixel size is the same. Depending on the number of rasters selected and the blocksize, a specific area of every tile´s raster (which does not always include the complete original raster tile) is included in a single parent split.

A mapper processes each split in the node where it is stored. (For complex multiple raster algebra operations, data locality may be lost, because a split contains data from several rasters.) The mapper executes the sequence of map algebra operations and processing classes defined by the user, and then the mosaic process is executed if requested. A single reducer puts together the result of the mappers and, for user-specified reducing processing classes, sets the output data set to these classes for analysis or process. Finally, the process stores the image into FS or HDFS upon user request. If the user requested to store the output into HDFS, then the `ImageLoader` job is invoked to store the image as an `.ohif` file.

By default, the mappers and reducers are configured to get 1 GB of JVM, but you can override this settings or any other job configuration properties by adding an `imagejob.prop` properties file in the same folder location from where the command is being executed.

2.5.4 Processing Classes and ImageBandWritable

The processing classes specified in the catalog XML must follow a set of rules to be correctly processed by the job. All the processing classes in the mapping phase must implement the `ImageProcessorInterface` interface. For the reducer phase, they must implement the `ImageProcessorReduceInterface` interface.

When implementing a processing class, you may manipulate the raster using its object representation `ImageBandWritable`. An example of an processing class is provided with the framework to calculate the slope on DEMs. You can create mapping operations, for example, to transforms the pixel values to another value by a function. The `ImageBandWritable` instance defines the content of a tile, such as resolution, size, and pixels. These values must be reflected in the properties that create the definition of the tile. The integrity of the mosaic output depends on the correct manipulation of these properties.

The `ImageBandWritable` instance defines the content of a tile, such as resolution, size, and pixels. These values must be reflected in the properties that create the definition of the tile. The integrity of the output depends on the correct manipulation of these properties.

Table 2-1 ImageBandWritable Properties

Type - Property	Description
IntWritable dstWidthSize	Width size of the tile
IntWritable dstHeightSize	Height size of the tile
IntWritable bands	Number of bands in the tile
IntWritable dType	Data type of the tile
IntWritable offX	Starting X pixel, in relation to the source image
IntWritable offY	Starting Y pixel, in relation to the source image
IntWritable totalWidth	Width size of the source image
IntWritable totalHeight	Height size of the source image
IntWritable bytesNumber	Number of bytes containing the pixels of the tile and stored into baseArray
BytesWritable[] baseArray	Array containing the bytes representing the tile pixels, each cell represents a band
IntWritable[][] basePaletteArray	Array containing the int values representing the tile palette, each array represents a band. Each integer represents an entry for each color in the color table, there are four entries per color
IntWritable[] baseColorArray	Array containing the int values representing the color interpretation, each cell represents a band
DoubleWritable[] noDataArray	Array containing the NODATA values for the image, each cell contains the value for the related band
ByteWritable isProjection	Specifies if the tile has projection information with Byte.MAX_VALUE
ByteWritable isTransform	Specifies if the tile has the geo transform array information with Byte.MAX_VALUE
ByteWritable isMetadata	Specifies if the tile has metadata information with Byte.MAX_VALUE
IntWritable projectionLength	Specifies the projection information length
BytesWritable projectionRef	Specifies the projection information in bytes
DoubleWritable[] geoTransform	Contains the geo transform array
IntWritable metadataSize	Number of metadata values in the tile
IntWritable[] metadataLength	Array specifying the length of each metadataValue
BytesWritable[] metadata	Array of metadata of the tile
GeneralInfoWritable mosaicInfo	The user-defined information in the mosaic xml. Do not modify the mosaic output features. Modify the original xml file in a new name and run the process using the new xml
MapWritable extraFields	Map that lists key/value pairs of parameters specific to every tile to be passed to the reducer phase for analysis

Processing Classes and Methods

When modifying the pixels of the tile, first get the band information into an array using the following method:

```
byte [] bandData1 =(byte []) img.getBand(0);
```

The bytes representing the tile pixels of band 1 are now in the bandData1 array. The base index is zero.

The `getBand(int bandId)` method will get the band of the raster in the specified `bandId` position. You can cast the object retrieved to the type of array of the raster; it could be byte, short (unsigned int 16 bits, int 16 bits), int (unsigned int 32 bits, int 32 bits), float (float 32 bits), or double (float 64 bits).

With the array of pixels available, it is possible now to transform them upon a user request.

After processing the pixels, if the same instance of ImageBandWritable must be used, then execute the following method:

```
img.removeBands;
```

This removes the content of previous bands, and you can start adding the new bands. To add a new band use the following method:

```
img.addBand(Object band);
```

Otherwise, you may want to replace a specific band by using trhe following method:

```
img.replaceBand(Object band, int bandId)
```

In the preceding methods, `band` is an array containing the pixel information, and `bandID` is the identifier of the band to be replaced.. Do not forget to update the instance size, data type, bytesNumber and any other property that might be affected as a result of the processing operation. Setters are available for each property.

- Location of the Classes and Jar Files

2.5.4.1 Location of the Classes and Jar Files

All the processing classes must be contained in a single jar file if you are using the Oracle SpatialViewer web application. The processing classes might be placed in different jar files if you are using the command line option.

When new classes are visible in the classpath, they must be added to the mosaic XML in the `<process><classMapper>` or `<process><classReducer>` section. Every `<class>` element added is executed in order of appearance: for mappers, just before the final mosaic operation is performed; and for reducers, just after all the processed tiles are put together in a single data set.

2.5.5 Map Algebra Operations

You can process local map algebra operations on the input rasters, where pixels are altered depending on the operation. The order of operations in the configuration XML determines the order in which the operations are processed. After all the map algebra operations are processed, the processing classes are run, and finally the mosaic operation is performed.

The following map algebra operations can be added in the `<operations>` element in the mosaic configuration XML, with the operation name serving as an element name. (The data types for which each operation is supported are listed in parentheses.)

- `localnot`: Gets the negation of every pixel, inverts the bit pattern. If the result is a negative value and the data type is unsigned, then the NODATA value is set. If the raster does not have a specified NODATA value, then the original pixel is set. (Byte, Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits)
- `locallog`: Returns the natural logarithm (base *e*) of a pixel. If the result is NaN, then original pixel value is set; if the result is Infinite, then the NODATA value is set. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `locallog10`: Returns the base 10 logarithm of a pixel. If the result is NaN, then the original pixel value is set; if the result is Infinite, then the NODATA value is set. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localadd`: Adds the specified value as argument to the pixel .Example: `<localadd arg="5"/>`. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localdivide`: Divides the value of each pixel by the specified value set as argument. Example: `<localdivide arg="5"/>`. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localif`: Modifies the value of each pixel based on the condition and value specified as argument. Valid operators: = , <, >, >=, < !=. Example: `<localif operator="<" operand="3" newvalue="6"/>`, which modifies all the pixels whose value is less than 3, setting the new value to 6. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localmultiply`: Multiplies the value of each pixel times the value specified as argument. Example: `<localmultiply arg="5"/>`. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localpow`: Raises the value of each pixel to the power of the value specified as argument. Example: `<localpow arg="5"/>`. If the result is infinite, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localsqrt`: Returns the correctly rounded positive square root of every pixel. If the result is infinite or NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localsubstract`: Subtracts the value specified as argument to every pixel value. Example: `<localsubstract arg="5"/>`. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localacos`: Calculates the arc cosine of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)

- `localasin`: Calculates the arc sine of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localatan`: Calculates the arc tangent of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localcos`: Calculates the cosine of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localcosh`: Calculates the hyperbolic cosine of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localsin`: Calculates the sine of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localtan`: Calculates the tangent of a pixel. The pixel is not modified if the cosine of this pixel is 0. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localsinh`: Calculates the arc hyperbolic sine of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localtanh`: Calculates the hyperbolic tangent of a pixel. If the result is NaN, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localdefined`: Maps an integer typed pixel to 1 if the cell value is not NODATA; otherwise, 0. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits, Float 32 bits)
- `localundefined`: Maps an integer typed Raster to 0 if the cell value is not NODATA; otherwise, 1. (Unsigned int 16 bits, Unsigned int 32 bits, Int 16 bits, Int 32 bits)
- `localabs`: Returns the absolute value of signed pixel. If the result is Infinite, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localnegate`: Multiplies by -1 the value of each pixel. (Int 16 bits, Int 32 bits, Float 32 bits, Float 64 bits)
- `localceil`: Returns the smallest value that is greater than or equal to the pixel value and is equal to a mathematical integer. If the result is Infinite, the NODATA value is set to this pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Float 32 bits, Float 64 bits)
- `localfloor`: Returns the smallest value that is less than or equal to the pixel value and is equal to a mathematical integer. If the result is Infinite, the NODATA value is set to this

pixel. If the raster does not have a specified NODATA value, then the original pixel is set. (Float 32 bits, Float 64 bits)

- `localround`: Returns the closest integer value to every pixel. (Float 32 bits, Float 64 bits)

2.5.6 Multiple Raster Algebra Operations

You can process raster algebra operations that involve more than one raster, where pixels are altered depending on the operation and taking in consideration the pixels from all the involved rasters in the same cell.

Only one operation can be processed at a time and it is defined in the configuration XML using the `<multipleops>` element. Its value is the operation to process.

There are two types of operations:

- Basic Multiple Raster Algebra Operations are executed in the reduce phase right before the Reduce User Processing classes.
- Complex Multiple Raster Algebra Operations are processed in the mapping phase.
- Basic Multiple Raster Algebra Operations
- Complex Multiple Raster Algebra Operations

2.5.6.1 Basic Multiple Raster Algebra Operations

Basic multiple raster algebra operations are executed in the reducing phase of the job.

They can be requested along with a mosaic operation or just a process request. If requested along with a mosaic operation, the input rasters must have the same MBR, pixel size, SRID and data type.

When a mosaic operation is performed, only the intersecting pixels (pixels that are identical in both rasters) are affected.

The operation is processed at the time that mapping tiles are put together in the output dataset, the pixel values that intersect (if a mosaic operation was requested) or all the pixels (when mosaic is not requested) are altered according to the requested operation.

The order in which rasters are added to the data set is the mosaic operation order if it was requested; otherwise, it is the order of appearance in the catalog.

The following basic multiple raster algebra operations are available:

- `add`: Adds every pixel in the same cell for the raster sequence.
- `substract`: Subtracts every pixel in the same cell for the raster sequence.
- `divide`: Divides every pixel in the same cell for the raster sequence.
- `multiply`: Multiplies every pixel in the same cell for the raster sequence.
- `min`: Assigns the minimum value of the pixels in the same cell for the raster sequence.
- `max`: Assigns the maximum value of the pixels in the same cell for the raster sequence.

- `mean`: Calculates the mean value for every pixel in the same cell for the raster sequence.
- `and`: Processes binary “and” operation on every pixel in the same cell for raster sequence, “and“ operation copies a bit to the result if it exists in both operands.
- `or`: Processes binary “or” operation on every pixel in the same cell for raster sequence, “or” operation copies a bit if it exists in either operand.
- `xor`: Processes binary “xor” operation on every pixel in the same cell for raster sequence, “xor” operation copies the bit if it is set in one operand but not both.

2.5.6.2 Complex Multiple Raster Algebra Operations

Complex multiple raster algebra operations are executed in the mapping phase of the job, and a job can only process this operation; any request for resizing, changing the SRID, or custom mapping must have been previously executed. The input for this job is a series of rasters with the same MBR, SRID, data type, and pixel size.

The tiles for this job include a piece of all the rasters in the catalog. Thus, every mapper has access to an area of cells in all the rasters, and the operation is processed there. The resulting pixel for every cell is written in the context, so that reducer can put results in the output data set before processing the reducer processing classes.

The order in which rasters are considered to evaluate the operation is the order of appearance in the catalog.

The following complex multiple raster algebra operations are available:

- `combine`: Assigns a unique output value to each unique combination of input values in the same cell for the raster sequence.
- `majority`: Assigns the value within the same cells of the rasters sequence that is the most numerous. If there is a values tie, the one on the right is selected.
- `minority`: Assigns the value within the same cells of the raster sequence that is the least numerous. If there is a values tie, the one on the right is selected.
- `variety`: Assigns the count of unique values at each same cell in the sequence of rasters.
- `mask`: Generates a raster with the values from the first raster, but only includes pixels in which the corresponding pixel in the rest of rasters of the sequence is set to the specified mask values. Otherwise, 0 is set.
- `inversemask`: Generates a raster with the values from the first raster, but only includes pixels in which the corresponding pixel in the rest of rasters of the sequence is *not* set to the specified mask values. Otherwise, 0 is set.
- `equals`: Creates a raster with data type byte, where cell values equal 1 if the corresponding cells for all input rasters have the same value. Otherwise, 0 is set.
- `unequal`: Creates a raster with data type byte, where cell values equal 1 if the corresponding cells for all input rasters have a different value. Otherwise, 0 is set.
- `greater`: Creates a raster with data type byte, where cell values equal 1 if the cell value in the first raster is greater than the rest of corresponding cells for all input. Otherwise, 0 is set.
- `greaterorequal`: Creates a raster with data type byte, where cell values equal 1 if the cell value in the first raster is greater or equal than the rest of corresponding cells for all input. Otherwise, 0 is set.

- `less`: Creates a raster with data type byte, where cell values equal 1 if the cell value in the first raster is less than the rest of corresponding cells for all input. Otherwise, 0 is set.
- `lessorequal`: Creates a raster with data type byte, where cell values equal 1 if the cell value in the first raster is less or equal than the rest of corresponding cells for all input. Otherwise, 0 is set.

2.5.7 Pyramids

Pyramids are subobjects of a raster object that represent the raster image or raster data at differing sizes and degrees of resolution.

The size is usually related to the amount of time that an application needs to retrieve and display an image, particularly over the web. That is, the smaller the image size, the faster it can be displayed; and as long as detailed resolution is not needed (for example, if the user has "zoomed out" considerably), the display quality for the smaller image is adequate.

Pyramid levels represent reduced or increased resolution images that require less or more storage space, respectively. (Big Data Spatial and Graph supports only reduced resolution pyramids.) A pyramid level of 0 indicates the original raster data; that is, there is no reduction in the image resolution and no change in the storage space required. Values greater than 0 (zero) indicate increasingly reduced levels of image resolution and reduced storage space requirements.

A single raster is processed for each pyramid request, and the following parameters apply:

- Pyramid level (required): the maximum reduction level; that is, the number of pyramid levels to create at a reduced size than the original object. For example, `redLevel=”6”` causes pyramid levels to be created for levels 0 through 5.

  The dimension sizes at each lower level are: `r(n) = r(n - 1)/2` and `c(n) = c(n - 1)/2` where:

  `r(n)` and `c(n)` are the row and column sizes for a pyramid at level `n`

  The smaller of the row and column dimension sizes of the top-level overview is between 64 and 128 (maximum reduced-resolution level): `(int)(log2(a/64))` where `a` is the smaller of the original row or column dimension size.

  If an `rLevel` value greater than the maximum reduced-resolution level is specified, the `rLevel` value is set to the maximum reduced-resolution level.

- Resampling algorithm: the resampling method to use.

  Must be one of the following: `NEAREST_NEIGHBOR`, `BILINEAR`, `AVERAGE4`, `AVERAGE16`. (`BILINEAR` and `AVERAGE4` have the same effect.) If no resampling algorithm is specified, `BILINEAR` is used by default.

Pyramids can be created while loading multiple rasters or processing a single raster:

- While loading the rasters in HDFS, by adding the `-pyramid` parameter to the loader command line call or by using the API `loader.addPyramid()`
- For processing a single raster, by adding the operation in the user request XML or by using the API `processor.addPyramid()`

2.5.8 Output

When you specify an HDFS directory in the configuration XML, the output generated is an `.ohif` file as in the case of an `ImageLoader` job,

When the user specifies a FS directory in the configuration XML, the output generated is an image with the filename and type specified and is stored into regular FileSystem.

In both the scenarios, the output must comply with the specifications set in the configuration XML. The job execution logs can be accessed using the command `yarn logs -applicationId <applicationId>`.

2.6 Loading and Processing an Image Using the Oracle Spatial Hadoop Raster Processing API

The framework provides a raster processing API that lets you load and process rasters without creating XML but instead using a Java application. The application can be executed inside the cluster or on a remote node.

The API provides access to the framework operations, and is useful for web service or standalone Java applications.

To execute any of the jobs, a `HadoopConfiguration` object must be created. This object is used to set the necessary configuration information (such as the jar file name and the GDAL paths) to create the job, manipulate rasters, and execute the job. The basic logic is as follows:

```
     //Creates Hadoop Configuration
     HadoopConfiguration hadoopConf = new HadoopConfiguration();
     //Assigns GDAL_DATA location based on specified SHAREDDIR, this data folder is
required by gdal to look for data tables that allow SRID conversions
     String gdalData = sharedDir + ProcessConstants.DIRECTORY_SEPARATOR + "data";
     hadoopConf.setGdalDataPath(gdalData);
     //Sets jar name for processor
     hadoopConf.setMapreduceJobJar("hadoop-imageprocessor.jar");
     //Creates the job
     RasterProcessorJob processor = (RasterProcessorJob)
hadoopConf.createRasterProcessorJob();
```

If the API is used on a remote node, you can set properties in the Hadoop Configuration object to connect to the cluster. For example:

```
        //Following config settings are required for standalone execution. (REMOTE
ACCESS)
        hadoopConf.setUser("hdfs");
        hadoopConf.setHdfsPathPrefix("hdfs://sys3.example.com:8020");
        hadoopConf.setResourceManagerScheduler("sys3.example.com:8030");
        hadoopConf.setResourceManagerAddress("sys3.example.com:8032");
        hadoopConf.setYarnApplicationClasspath("/etc/hadoop/conf/,/usr/lib/
hadoop/*,/usr/lib/hadoop/lib/*," +
                          "/usr/lib/hadoop-hdfs/*,/usr/lib/hadoop-hdfs/lib/*,/usr/lib/
hadoop-yarn/*," +
                          "/usr/lib/hadoop-yarn/lib/*,/usr/lib/hadoop-
mapreduce/*,/usr/lib/hadoop-mapreduce/lib/* ");
```

After the job is created, the properties for its execution must be set depending on the job type. There are two job classes: `RasterLoaderJob` to load the rasters into HDFS, and `RasterProcessorJob` to process them.

The following example loads a Hawaii raster into the APICALL_HDFS directory. It creates a thumbnail in a shared folder, and specifies 10 pixels overlapping on each edge of the tiles.

```
    private static void executeLoader(HadoopConfiguration hadoopConf){
        hadoopConf.setMapreduceJobJar("hadoop-imageloader.jar");
        RasterLoaderJob loader = (RasterLoaderJob)
hadoopConf.createRasterLoaderJob();
        loader.setFilesToLoad("/net/den00btb/scratch/zherena/hawaii/hawaii.tif");
        loader.setTileOverlap("10");
        loader.setOutputFolder("APICALL");
        loader.setRasterThumbnailFolder("/net/den00btb/scratch/zherena/
processOutput");
        try{
        loader.setGdalPath("/net/den00btb/scratch/zherena/gdal/lib");

        boolean loaderSuccess = loader.execute();
            if(loaderSuccess){
                System.out.println("Successfully executed loader job");
            }
            else{
                System.out.println("Failed to execute loader job");
            }
        }catch(Exception e ){
        System.out.println("Problem when trying to execute raster loader " +
e.getMessage());
        }
    }
}
```

The following example processes the loaded raster.

```
private static void executeProcessor(HadoopConfiguration hadoopConf){
    hadoopConf.setMapreduceJobJar("hadoop-imageprocessor.jar");
    RasterProcessorJob processor = (RasterProcessorJob)
hadoopConf.createRasterProcessorJob();

    try{
    processor.setGdalPath("/net/den00btb/scratch/zherena/gdal/lib");
    MosaicConfiguration mosaic = new MosaicConfiguration();
        mosaic.setBands(3);
        mosaic.setDirectory("/net/den00btb/scratch/zherena/processOutput");
        mosaic.setFileName("APIMosaic");
        mosaic.setFileSystem(RasterProcessorJob.FS);
        mosaic.setFormat("GTIFF");
        mosaic.setHeight(3192);
        mosaic.setNoData("#FFFFFF");
        mosaic.setOrderAlgorithm(ProcessConstants.ALGORITMH_FILE_LENGTH);
        mosaic.setOrder("1");
        mosaic.setPixelType("1");
        mosaic.setPixelXWidth(67.457513);
        mosaic.setPixelYWidth(-67.457513);
        mosaic.setSrid("26904");
        mosaic.setUpperLeftX(830763.281336);
        mosaic.setUpperLeftY(2259894.481403);
        mosaic.setWidth(1300);
    processor.setMosaicConfigurationObject(mosaic.getCompactMosaic());
```

```
        RasterCatalog catalog = new RasterCatalog();
        Raster raster = new Raster();
        raster.setBands(3);
        raster.setBandsOrder("1,2,3");
        raster.setDataType(1);
        raster.setRasterLocation("/user/hdfs/APICALL/net/den00btb/scratch/zherena/
hawaii/hawaii.tif.ohif");
        catalog.addRasterToCatalog(raster);

        processor.setCatalogObject(catalog.getCompactCatalog());
    boolean processorSuccess = processor.execute();
        if(processorSuccess){
            System.out.println("Successfully executed processor job");
        }
        else{
            System.out.println("Failed to execute processor job");
        }
    }catch(Exception e ){
    System.out.println("Problem when trying to execute raster processor " +
e.getMessage());
    }
}
```

In the preceding example, the thumbnail is optional if the mosaic results will be stored in HDFS. If a processing jar file is specified (used when the additional user processing classes are specified), the location of the jar file containing these lasses must be specified. The other parameters are required for the mosaic to be generated successfully.

Several examples of using the processing API are provided `/opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/src`. Review the Java classes to understand their purpose. You may execute them using the scripts provided for each example located under `/opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/cmd`.

After you have executed the scripts and validated the results, you can modify the Java source files to experiment on them and compile them using the provided script `/opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/build.xml`. Ensure that you have write access on the `/opt/oracle/oracle-spatial-graph/spatial/raster/jlib directory`.

2.7 Using the Oracle Spatial Hadoop Raster Simulator Framework to Test Raster Processing

When you create custom processing classes. you can use the Oracle Spatial Hadoop Raster Simulator Framework to do the following by "pretending" to plug them into the Oracle Raster Processing Framework.

- Develop user processing classes on a local computer
- Avoid the need to deploy user processing classes in a cluster or in Big Data Lite to verify their correct functioning
- Debug user processing classes
- Use small local data sets
- Create local debug outputs
- Automate unit tests

The Simulator framework will emulate the loading and processing processes in your local environment, as if they were being executed in a cluster. You only need to create a Junit test case that loads one or more rasters and processes them according to your specification in XML or a configuration object.

Tiles are generated according to specified block size, so you must set a block size. The number of mappers and reducers to execute depends on the number of tiles, just as in regular cluster execution. OHIF files generated during the loading process are stored in local directory, because no HDFS is required.

- Simulator (“Mock”) Objects
- User Local Environment Requirements
- Sample Test Cases to Load and Process Rasters

Simulator (“Mock”) Objects

To load rasters and convert them into .OHIF files that can be processed, a `RasterLoaderJobMock` must be executed. This class constructor receives the `HadoopConfiguration` that must include the block size, the directory or rasters to load, the output directory to store the OHIF files, and the gdal directory. The parameters representing the input files and the user configuration vary in terms of how you specify them:

- Location Strings for catalog and user configuration XML file
- Catalog object (`CatalogMock`)
- Configuration objects (`MosaicProcessConfigurationMock` or `SingleProcessConfigurationMock`)
- Location for a single raster processing and a user configuration (`MosaicProcessConfigurationMock` or `SingleProcessConfigurationMock`)

User Local Environment Requirements

Before you create test cases, you need to configure your local environment.

1. 1. Ensure that a directory has the native gdal libraries, `gdal-data` and `libproj`.

   For Linux:

   a. Follow the steps in Getting and Compiling the Cartographic Projections Library to obtain `libproj.so`.
   b. Get the gdal distribution from the Spatial installation on your cluster or BigDataLite VM at `/opt/oracle/oracle-spatial-graph/spatial/raster/gdal`.
   c. Move `libproj.so` to your local gdal directory under `gdal/lib` with the rest of the native gdal libraries.

   For Windows:

   a. Get the gdal distribution from your Spatial install on your cluster or BigDataLite VM at `/opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/mock/lib/gdal_windows.x64.zip`.
   b. Be sure that Visual Studio installed. When you install it, make sure you select the *Common Tools for Visual C++*.

   c. Download the PROJ 4 source code, version branch 4.9 from https://trac.osgeo.org/proj4j.
   d. Open the Visual Studio Development Command Prompt and type:

```
cd PROJ4/src_dir
nmake /f makefile.vc
```

   e. Move `proj.dll` to your local gdal directory under `gdal/bin` with the rest of the native gdal libraries.
2. Add GDAL native libraries to system path.

   For Linux: Export **LD_LIBRARY_PATH** with corresponding native gdal libraries directory

   For Windows: Add to the **Path** environment variable the native gdal libraries directory.
3. Ensure that the Java project has Junit libraries.
4. Ensure that the Java project has the following Hadoop jar and Oracle Image Processing Framework files in the classpath You may get them from the Oracle BigDataLite VM or from your cluster; these are all jars included in the Hadoop distribution, and for specific framework jars, go to `/opt/oracle/oracle-spatial-graph/spatial/raster/jlib`:

   (In the following list, `VERSION_INCLUDED` refers to the version number from the Hadoop installation containing the files; it can be a BDA cluster or a BigDataLite VM.)

```
commons-collections-VERSION_INCLUDED.jar
commons-configuration-VERSION_INCLUDED.jar
commons-lang-VERSION_INCLUDED.jar
commons-logging-VERSION_INCLUDED.jar
commons-math3-VERSION_INCLUDED.jar
gdal.jar
guava-VERSION_INCLUDED.jar
hadoop-auth-VERSION_INCLUDED-cdhVERSION_INCLUDED.jar
hadoop-common-VERSION_INCLUDED-cdhVERSION_INCLUDED.jar
hadoop-imageloader.jar
hadoop-imagemocking-fwk.jar
hadoop-imageprocessor.jar
hadoop-mapreduce-client-core-VERSION_INCLUDED-cdhVERSION_INCLUDED.jar
hadoop-raster-fwk-api.jar
jackson-core-asl-VERSION_INCLUDED.jar
jackson-mapper-asl-VERSION_INCLUDED.jar
log4j-VERSION_INCLUDED.jar
slf4j-api-VERSION_INCLUDED.jar
slf4j-log4j12-VERSION_INCLUDED.jar
```

Sample Test Cases to Load and Process Rasters

After your Java project is prepared for your test cases, you can test the loading and processing of rasters.

The following example creates a class with a `setUp` method to configure the directories for gdal, the rasters to load, your configuration XML files, the output thumbnails, ohif files, and process results. It also configures the block size (8 MB). (A small block size is recommended for single computers.)

```
/**
 * Set the basic directories before starting the test execution
 */
@Before
public void setUp(){
```

```
        String sharedDir = "C:\\Users\\zherena\\Oracle Stuff\\Hadoop\\Release 4\
\MockTest";
        String allAccessDir = sharedDir +  "/out/";
        gdalDir = sharedDir + "/gdal";
        directoryToLoad = allAccessDir + "rasters";
        xmlDir = sharedDir + "/xmls/";
        outputDir = allAccessDir;
        blockSize = 8;
    }
```

The following example creates a RasterLoaderJobMock object, and sets the rasters to load and the output path for OHIF files:

```
/**
     * Loads a directory of rasters, and generate ohif files and thumbnails
       * for all of them
     * @throws Exception if there is a problem during load process
     */
    @Test
    public void basicLoad() throws Exception {
        System.out.println("***LOAD OF DIRECTORY WITHOUT EXPANSION***");
        HadoopConfiguration conf = new HadoopConfiguration();
        conf.setBlockSize(blockSize);
        System.out.println("Set block size of: " +
                                conf.getProperty("dfs.blocksize"));
        RasterLoaderJobMock loader = new RasterLoaderJobMock(conf,
                                    outputDir, directoryToLoad, gdalDir);
        //Puts the ohif file directly in the specified output directory
        loader.dontExpandOutputDir();
        System.out.println("Starting execution");

System.out.println("-----------------------------------------------------------------
-----------------------------------------------");
        loader.waitForCompletion();
        System.out.println("Finished loader");
        System.out.println("LOAD OF DIRECTORY WITHOUT EXPANSION ENDED");
        System.out.println();
        System.out.println();
    }
```

The following example specifies catalog and user configuration XML files to the `RasterProcessorJobMock` object. Make sure your `catalog xml` points to the correct location of your local OHIF files.

```
    /**
     * Creates a mosaic raster by using configuration and catalog xmls.
         * Only two bands are selected per raster.
     * @throws Exception    if there is a problem during mosaic process.
     */
    @Test
    public void mosaicUsingXmls() throws Exception {
        System.out.println("***MOSAIC PROCESS USING XMLS***");
        HadoopConfiguration conf = new HadoopConfiguration();
        conf.setBlockSize(blockSize);
        System.out.println("Set block size of: " +
               conf.getProperty("dfs.blocksize"));
               String catalogXml = xmlDir + "catalog.xml";
           String configXml = xmlDir + "config.xml";
    RasterProcessorJobMock processor = new  RasterProcessorJobMock(conf,
configXml, catalogXml, gdalDir);
        System.out.println("Starting execution");
```

```
System.out.println("----------------------------------------------------------------------
----------------------------------------");
            processor.waitForCompletion();
        System.out.println("Finished processor");
        System.out.println("***********************************************MOSAIC
PROCESS USING XMLS ENDED***********************************************");
        System.out.println();
        System.out.println();
```

Additional examples using the different supported configurations for `RasterProcessorJobMock` are provided in `/opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/mock/src`.They include an example using an external processing class, which is also included and can be debugged.

2.8 Oracle Big Data Spatial Raster Processing for Spark

Oracle Big Data Spatial Raster Processing for Apache Spark is a spatial raster processing API for Java.

This API allows the creation of new combined images resulting from a series of user-defined processing phases, with the following features:

- HDFS images storage, where every block size split is stored as a separate tile, ready for future independent processing
- Subset, mosaic, and raster algebra operations processed in parallel using Spark to divide the processing.
- Support for GDAL formats, multiple bands images, DEMs (digital elevation models), multiple pixel depths, and SRIDs

Currently the API supports Spark 1.6 and Spark 2.2. The only visible change in the API is the substitution of `Dataframe` with `Dataset<Row>` in the results of Spark 2.2 SQL queries.

- Spark Raster Loader
- Spark SQL Raster Processor
- Using the Spark Raster Processing API

2.8.1 Spark Raster Loader

The first step in using the raster processing for Spark Java API is to have the images in HDFS, followed by having the images separated into smart tiles. This allows the processor to work on each tile independently. The Spark raster loader lets you import a single image or a collection of them into HDFS in parallel, which decreases the load time. Each block contains data for all the raster bands, so that if further processing is required on specific pixels, the information can be processed on a single node.

The basic workflow for the Spark raster loader is as follows.

1. GDAL is used to import the rasters, tiling them according to block size and then storing each tile as an HDFS block.
2. The set of rasters to be loaded is read into a `SpatialRasterJavaRDD`, which is an extension of `JavaRDD`. This RDD is a collection of `ImagePieceWritable` objects that represent the information of the tiles to create per raster, based on the number of bands,

pixel size, HDFS block size, and raster resolution. This is accomplished by using the custom input format used in the spatial Hadoop loader.

3. The raster information for each tile is loaded. This load is performed by an executor for each tile, so reading is performed parallel. Each tile includes a certain number of overlapping bytes (user input), so that the tiles cover area from the adjacent tiles. There are “n” number of Spark executors, depending on the number of tiles, image resolution, and block size.
4. The RDD is grouped by key, so that all the tiles that correspond to the same raster are part of the same record. This RDD is saved as OHIF using the `OhifOutputFormat`, which puts together all the information loaded by the executors and stores the images into a special `.ohif` format, which contains the resolution, bands, offsets, and image data. In this way, the file offset containing each tile and the node location is known. A special reading process is required to read the image back and is included in the Spark SQL raster processor.

Each tile contains information for every band. This is helpful when there is a need to process only a few tiles; then, only the corresponding blocks are loaded.

The loader can be configured by setting parameters on the command line or by using the Spark API.

- Input Parameters to the Spark Raster Loader
- Expected Output of the Spark Raster Loader

2.8.1.1 Input Parameters to the Spark Raster Loader

The following example shows input parameters supplied using the spark-submit command:

```
spark-submit
    --class <DRIVER_CLASS>
    --driver-memory <DRIVER_JVM>
    --driver-class-path <DRIVER_CLASSPATH>
    --jars <EXECUTORS_JARS>
    <DRIVER_JAR>
    -files <SOURCE_IMGS_PATH>
    -out <HDFS_OUTPUT_FOLDER>
    -gdal <GDAL_LIB_PATH>
    -gdalData <GDAL_DATA_PATH>
    [-overlap <OVERLAPPING_PIXELS>]
    [-thumbnail <THUMBNAIL_PATH>]
    [-expand <false|true>]
       [-rasterSource objectStore]
       [-containerService containername.service]
       [-workDir <OS_WORK_DIR>]
       [-credential spatialraster]
```

Where:

- `DRIVER_CLASS` is the class that has the driver code and that Spark will execute.
- `DRIVER_JVM` is the memory to assign to driver´s JVM.
- `DRIVER_CLASSPATH` is the classpath for driver class, jars are separated by colon.

- `EXECUTOR_JARS` is the classpath to be distributed to executors, jars are separated by comma.
- `DRIVER_JAR` is the jar that contains the <DRIVER_CLASS> to execute by Spark.
- `SOURCE_IMGS_PATH` is a path to the source raster(s) or folder(s). For multiple inputs use a comma separator. This path must be accessible via NFS to all nodes in the cluster.
- `HDFS_OUTPUT_FOLDER` is the HDFS output folder where the loaded images are stored.
- `OVERLAPPING_PIXELS` is an optional number of overlapping pixels on the borders of each tile, if this parameter is not specified a default of two overlapping pixels is considered.
- `GDAL_LIB_PATH` is the path where GDAL libraries are located.
- `GDAL_DATA_PATH` is the path where GDAL data folder is located. This path must be accessible through NFS to all nodes in the cluster.
- `THUMBNAIL_PATH` is an optional path to store a thumbnail of the loaded image(s). This path must be accessible through NFS to all nodes in the cluster and must have write access permission for yarn users.
- `-expand` controls whether the HDFS path of the loaded raster expands the source path, including all directories. If you set this to `false`, the `.ohif` file is stored directly in the output directory (specified using the `-o` option) without including that directory's path in the raster.
- `-rasterSource` should be set to *objectStore* if the rasters are in Oracle Object Storage; if not set, the default flow is followed.
- `-containerService` specifies the Object Storage container and service where the rasters are stored. This is a required field.
- `-workDir` specifies the local temporary directory to download Object Storage rasters. Required for Object Storage loads.
- `-credential` specifies the ID of the credential stored in Credential Store to the secure Object Storage password.

Each tile contains information for every band. This is helpful when there is a need to process only a few tiles; then, only the corresponding blocks are loaded.

The loader can be configured by setting parameters on the command line or by using the Spark API.

2.8.1.2 Expected Output of the Spark Raster Loader

For each input image to the Spark raster loader, there are two output files per input image.

- The `.ohif` file that concentrates all the tiles for the source image. Each tile (stored as a HDFS block) may be processed as a separated instance by a processing executor. The `.ohif` file is stored in a user-specified folder with `-out` flag, under `/user/<USER_EXECUTING_JOB>/OUT_FOLDER/<PARENT_DIRECTORIES_OF_SOURCE_RASTER>` if the flag `-expand` was not used. Otherwise, the `.ohif` file will be located at `/user/<USER_EXECUTING_JOB>/OUT_FOLDER/`, and the file can be identified as `original_filename.ohif`.
- A related metadata file that lists all the pieces of the image and the coordinates that each one covers. This file is located in HDFS under the `spatial_raster/metadata` location, and its name is hash-generated using the name of the `.ohif` file. This file is for Oracle

internal use only, and lists important metadata of the source raster. Some example lines from a metadata file:

```
size:3200,2112
srid:26904
datatype:1
resolution:27.90809458890406,-27.90809458890406
file:/user/hdfs/ohiftest/opt/shareddir/spatial/data/rasters/
hawaii.tif.ohif
bands:3
mbr:532488.7648166901,4303164.583549625,582723.3350767174,4269619.05
3853762
0,532488.7648166901,4303164.583549625,582723.3350767174,4269619.0538
53762
thumbnailpath:/opt/shareddir/spatial/thumb/
```

If the `-thumbnail` flag was specified, a thumbnail of the source image is stored in the related folder. This is a way to visualize a translation of the `.ohif` file. Execution logs can be accessed using the command `yarn logs -applicationId <applicationId>`.

2.8.2 Spark SQL Raster Processor

Once the images are loaded into HDFS, they can be processed using Spark SQL Raster Processor. You specify the expected raster output features using the Mosaic Definition XML Structure or the Spark API, and the mosaic UDF filters the tiles to fit into that output and processes them. Raster algebra operations are also available in UDF.

A custom `InputFormat`, which is also used in the Hadoop raster processing framework, loads specific blocks of data, based on the input (mosaic description or a single raster) using raster SRID and coordinates, and selects only the bands and pixels that fit into the final output before accepting processing operations:

- For a mosaic processing request, only the intersecting tiles are selected, and a split is created for each one of them.
- For a single raster processing request, all the tiles are selected, and a split is created for each one of them.

The Spark SQL Raster Processor allows you to filter the OHIF tiles based on input catalog or raster into a Dataframe, with every row representing a tile, and to use Spatial UDF Spark functions to process them.

A simplified pseudocode representation of Spark SQL raster processing is:

```
sqlContext.udf().register("localop", new
LocalOperationsFunction(),DataTypes.createStructType(SpatialRasterJavaR
DD.createSimpleTileStructField(dataTypeOfTileToProcess)));
tileRows.registerTempTable("tiles");
String query = "SELECT localop(tileInfo, userRequest,  \"localnot\"),
userRequest FROM tiles";
DataFrame processedTiles = sqlContext.sql(query);
```

The basic workflow of the Spark SQL raster processor is as follows.

1. The rasters to process are first loaded in tiles metadata as RDD. These tiles may be filtered if the user set a configuration for mosaic operation. The RDD is later converted to a Spark DataFrame (or Dataset<Row> for Spark 2.2) of two complex rows: the first row is `tileInfo`, which has all the metadata for the tiles, and the second row is the `userRequest`, which has the user input configuration listing the expected features of the raster output.
2. Once the DataFrame or Dataset<Row> is created, the driver must register the “localop” UDF, and also register the DataFrame or Dataset<Row> as a table before executing a query to process. The mosaic UDF can only be executed if the user configured all the required parameters correctly. If no XML is used and the configuration is set using the API, then by default a mosaic operation configuration is expected unless the `setExecuteMosaic(false)` method is set.
3. The mosaic operation selects from every tile only the pixels that fit into the output, and makes the necessary resolution changes to add them in the mosaic output.
4. Once the query is executed, an executor loads the data corresponding tile, conserving data locality, and the specified local raster algebra operation is executed.
5. The row in the DataFrame or Dataset<Row> is updated with the new pixel data and returned to the driver for further processing if required.
6. Once the processing is done, the DataFrame or Dataset<Row> is converted to a list of `ImageBandWritable` objects, which are the MapReduce representation of processed tiles. These are input to the `ProcessedRasterCreator`, where resulting bytes of local raster algebra and/or mosaic operations are put together, and a final raster is stored into HDFS or the regular file system depending on the user request.

Only images with same data type (pixel depth) as the user configuration input data type (pixel depth) are considered. Only the tiles that intersect with coordinates specified by the user for the mosaic output are included. For processing of a single raster, the filter includes all the tiles of the input rasters, because the processing will be executed on the complete images.

- Input Parameters to the Spark SQL Raster Processor
- Expected Output of the Spark SQL Raster Processor

2.8.2.1 Input Parameters to the Spark SQL Raster Processor

The following example shows input parameters supplied using the spark-submit command:

```
spark-submit
    --class <DRIVER_CLASS>
    --driver-memory <DRIVER_JVM>
    --driver-class-path <DRIVER_CLASSPATH>
    --jars <EXECUTORS_JARS>
    <DRIVER_JAR>
    -config  <MOSAIC_CONFIG_PATH>
    -gdal  <GDAL_LIBRARIES_PATH>
    -gdalData  <GDAL_DATA_PATH>
    [-catalog  <IMAGE_CATALOG_PATH>]
    [-file  <SINGLE_RASTER_PATH>]
```

Where:

- `DRIVER_CLASS` is the class that has the driver code and that Spark will execute.

- `DRIVER_JVM` is the memory to assign to driver´s JVM.
- `DRIVER_CLASSPATH` is the classpath for driver class, jars are separated by colon.
- `EXECUTOR_JARS` is the classpath to be distributed to executors, jars are separated by comma.
- `DRIVER_JAR` is the jar that contains the <DRIVER_CLASS> to execute by Spark.
- `MOSAIC_CONFIG_PATH` is the path to the mosaic configuration XML, which defines the features of the output.
- `GDAL_LIBRARIES_PATH` is the path where GDAL libraries are located.
- `GDAL_DATA_PATH` is the path where the GDAL data folder is located. This path must be accessible via NFS to all nodes in the cluster.
- `IMAGE_CATALOG_PATH` is the path to the catalog xml that lists the HDFS image(s) to be processed. This is optional because you can also specify a single raster to process using –file flag.
- `SINGLE_RASTER_PATH` is an optional path to the .ohif file that will be processed by the job. If this is set, you do not need to set a catalog.

The following example command will process all the files listed in the catalog file `inputSPARK.xml` using the mosaic output definition set in the `testFS.xml` file.

```
spark-submit --class
oracle.spatial.spark.raster.test.SpatialRasterTest --driver-memory
2048m  --driver-class-path /opt/oracle/oracle-spatial-graph/spatial/
raster/jlib/hadoop-raster-fwk-api.jar:/opt/oracle/oracle-spatial-graph/
spatial/raster/jlib/gdal.jar:/opt/oracle/oracle-spatial-graph/spatial/
raster/jlib/hadoop-imageloader.jar:/opt/oracle/oracle-spatial-graph/
spatial/raster/jlib/hadoop-imageprocessor.jar --jars /opt/oracle/
oracle-spatial-graph/spatial/raster/jlib/hadoop-imageloader.jar,/opt/
oracle/oracle-spatial-graph/spatial/raster/jlib/hadoop-
imageprocessor.jar,/opt/oracle/oracle-spatial-graph/spatial/raster/
jlib/gdal.jar /opt/oracle/oracle-spatial-graph/spatial/raster/jlib/
spark-raster-fwk-api.jar -taskType algebra -catalog /opt/shareddir/
spatial/data/xmls/inputSPARK.xml -config /opt/shareddir/spatial/data/
xmls/testFS.xml -gdal /opt/oracle/oracle-spatial-graph/spatial/raster/
gdal/lib -gdalData /opt/shareddir/data
```

> **Note:**
>
> For Spark 2.2, use the `spark2-submit` command if there is a Spark 1.6 installation on the same node.

2.8.2.2 Expected Output of the Spark SQL Raster Processor

For Spark processing, only file system output is supported, which means that the output generated is an image with the file name and type specified and is stored in a regular FileSystem.

The job execution logs can be accessed using the command yarn logs -applicationId <applicationId>.

2.8.3 Using the Spark Raster Processing API

You can use the Spark raster API to load and process rasters by creating the driver class.

Some example classes are provided under /opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/src. The /opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/cmd directory also contains scripts to execute these examples from command line.

After executing the scripts and validated the results, you can modify the Java source files to experiment on them and compile them using the provided script /opt/oracle/oracle-spatial-graph/spatial/raster/examples/java/build.xml. Ensure that there is write access on the /opt/oracle/oracle-spatial-graph/spatial/raster/jlib directory.

For GDAL to work properly, the libraries must be available using $LD_LIBRARY_PATH. Make sure that the shared libraries path is set properly in your shell window before executing a job. For example:

```
export LD_LIBRARY_PATH=$ALLACCESSDIR/gdal/native
```

- Using the Spark Raster Loader API
- Configuring for Using the Spark SQL Processor API
- Creating the DataFrame or Dataset<Row>
- Using the Spark SQL UDF for Raster Algebra Operations

2.8.3.1 Using the Spark Raster Loader API

To perform image loading, you must create a SpatialRasterLoader object. This object is used to set the necessary configuration information for the execution. There are two ways of creating an instance:

- Send as a parameter the array of arguments received from the command line. For example:

```
//args is the String[] received from command line
SpatialRasterLoader core = new SpatialRasterLoaderCore(args);
```

- Configure directly in the driver class using the API, which is the subject of this topic

Using the Loader API, set the GDAL library path, since it will internally initialize the SparkContext and its corresponding Hadoop configuration. For example:

```
SpatialRasterLoader core = new SpatialRasterLoader();
core.setGdalLibrary("/opt/sharedddir/spatial/gdal");
core.setFilesToLoad("/opt/shareddir/spatial/rasters");
core.setHDFSOutputDirectory("ohifsparktest");
core.setGdalData("/opt/shareddir/data");
core.setOverlap("20");
core.setThumbnailDirectory("/opt/shareddir/spatial/");
```

You can optionally change the block size, depending on the most common size of rasters involved. For example, if the cluster HDFS block size is by default too big (such as 256 MB) and the average size of the user rasters is 64 MB in average, you should avoid using HDFS space that contains no real data, because every tile occupies a block in HDFS even if the pixels do not fill it. In this scenario, you can change the block side to 64 MB, as in this example:

```
JavaSparkContext sc = core.getRasterSparkContext();
core.getHadoopConfiguration().set("dfs.blocksize", "67108864");
```

To execute the loader, use the `loadRasters` method, which returns `true` if rasters were loaded with success and `false` otherwise. For example:

```
    if (core.loadRasters(sc, StorageLevel.DISK_ONLY())) {
        LOG.info("Successfully loaded raster files");
    }
```

If the processing finished successfully, the OHIF files are in HDFS and the corresponding thumbnails are in the specified directory for user validation.

2.8.3.2 Configuring for Using the Spark SQL Processor API

To execute a processor, you must create a `SpatialRasterProcessor` object to set the necessary configuration information for the execution. There are two ways to create an instance:

- Send as a parameter the array of arguments received from the command line. For example:

  ```
  //args is the String[] received from command line
  SpatialRasterProcessor processor = new SpatialRasterProcessor(args);
  ```

- Configure directly in the driver class using the API, which is the subject of this topic.

Using the Loader API, set the GDAL library path, because it will internally initialize the `SparkContext` and its corresponding Hadoop configuration. For example:

```
SpatialRasterProcessor processor = new SpatialRasterProcessor();
processor.setGdalLibrary("/opt/sharedddir/spatial/gdal");
processor.setGdalData("/opt/sharedddir/spatial/data");
```

Specify the rasters that will be processed.

- For adding a catalog of rasters to process, especially if a mosaic operation will be performed, consider the following example:

  ```
  String ohifPath = "ohifsparktest/opt/shareddir/spatial/data/
  rasters");
  //Creates a catalog to list the rasters to process
  RasterCatalog catalog = new RasterCatalog();

  //Creates a raster object for the catalog
  ```

```
Raster raster = new Raster();
//raster of 3 bands
raster.setBands(3);
//the tree bands will appear in order 1,2,3. You may list less bands here.
raster.setBandsOrder("1,2,3");
//raster data type is byte
raster.setDataType(1);

raster.setRasterLocation(ohifPath + "hawaii.tif.ohif");
//Add raster to catalog
//catalog.addRasterToCatalog(raster);

Raster rasterKahoolawe = new Raster();
rasterKahoolawe.setBands(3);
rasterKahoolawe.setBandsOrder("1,2,3");
rasterKahoolawe.setDataType(1);
rasterKahoolawe.setRasterLocation(ohifPath + "kahoolawe.tif.ohif");
catalog.addRasterToCatalog(rasterKahoolawe);

//Sets the catalog to the job
processor.setCatalogObject(catalog.getCompactCatalog());
```

- For processing a single raster, consider the following example:

```
String ohifPath = "ohifsparktest/opt/shareddir/spatial/data/rasters");
//Set the file to process to the job
processor.setFileToProcess(ohifPath + "NapaDEM.tif.ohif");*/
```

Set the user configuration request, which defines details for the output raster.

- If a mosaic operation will be performed, then all the features of the expected output must be set in a MosaicConfiguration object, including the coordinates. the following example creates a raster that includes both Hawaii rasters added to the catalog previously:

```
MosaicConfiguration mosaic = new MosaicConfiguration();
mosaic.setFormat("GTIFF");
mosaic.setBands(3);
mosaic.setFileSystem(RasterProcessorJob.FS);
mosaic.setDirectory("/opt/shareddir/spatial/processtest");
mosaic.setFileName("HawaiiIslands");
mosaic.setHeight(986);
//value for pixels where there is no data, starts with #, followed by
//two characters per band
mosaic.setNoData("#FFFFFF");
//byte datatype
mosaic.setPixelType("1");
//width for pixels in X and Y
mosaic.setPixelXWidth(280.388143);
mosaic.setPixelYWidth(-280.388143);
mosaic.setSrid("26904");
//upper left coordinates
mosaic.setUpperLeftX(556958.985610);
mosaic.setUpperLeftY(2350324.082505);
mosaic.setWidth(1600);
mosaic.setOrderAlgorithm(ProcessConstants.ALGORITHM_FILE_LENGTH);
```

```
mosaic.setOrder(RasterProcessorJob.DESC);
//mosaic configuration must be set to the job
processor.setUserRequestConfigurationObject(mosaic.getCompactMosaic(
));
```

- If a mosaic operation will not be performed, then a much simpler configuration is required. For example:

```
MosaicConfiguration mosaic = new MosaicConfiguration();
mosaic.setExecuteMosaic(false);
mosaic.setBands(1);
mosaic.setLayers("1");
mosaic.setDirectory("/opt/shareddir/spatial/processtest");
mosaic.setFileSystem(RasterProcessorJob.FS);
mosaic.setNoData("#00");
```

At this point, all required configuration is done. You can now start processing.

2.8.3.3 Creating the DataFrame or Dataset<Row>

Before running queries against the rasters, you must load them into a distributed collection of structured data, organized into named columns where every row represents a split. This structure is known as Dataframe, and Dataset<Row> for Spark 2.2.

The splits are created into a `SpatialJavaRDD` of tiles, which are then converted to a DataFrame or Dataset<Row>. Depending on your available JVM runtime memory, it is recommended that you cache the DataFrame in memory or on disk. For disk caching, your Spark installation must have Kryo.

The DataFrame or Dataset<Row> consists of two complex columns: `tileInfo` and `userRequest`.

- `tileInfo`: Data for every tile, including not only pixel information but also metadata details.

Table 2-2 tileInfo Column Data

Column	DataType	Nullable	Description
dstWidthSize	Integer	False	Width
dstHeightSize	Integer	False	Height
bands	Integer	False	Number of bands
dType	Integer	False	Data type
piece	Integer	False	Piece number of total pieces in source raster
offX	Integer	False	Offset in X
offY	Integer	False	Offset in Y
sourceWidth	Integer	False	Source raster width
sourceHeight	Integer	False	Source raster height
bytesNumber	Integer	False	Number of bytes
baseArray	[[Pixel DataType]]	False	Array of pixels, one per band

Table 2-2 (Cont.) tileInfo Column Data

Column	DataType	Nullable	Description
basePaletteArray	[[Integer]]	True	Array of palette interpretation, if the raster has it, one per band
baseColorArray	[Integer]	False	Array of colors, one per band
noDataArray	[Double]	False	Array of NODATA value, one per band
Overlap	Integer	False	Number of overlapping pixels
leftOv	Byte	False	Flag to indicate if there are any overlapping pixels on the left
rightOv	Byte	False	Flag to indicate if there are any overlapping pixels on the right
upOv	Byte	False	Flag to indicate if there are any overlapping pixels on the top
downOv	Byte	False	Flag to indicate if there are any overlapping pixels on the bottom
projectionRef	String	False	Projection reference
geoTransform	[Double]	False	Geo Transformation array
Metadata	[String]	False	Location metadata
lastModified	Long	False	Source raster last modification date
imageLength	Double	False	Source raster length
dataLength	Integer	True	Number of bytes after mosaic
xCropInit	Integer	True	Pixel start in X after mosaic
yCropInit	Integer	True	Pixel start in Y after mosaic
xCropLast	Integer	True	Pixel end in X after mosaic
yCropLast	Integer	True	Pixel end in Y after mosaic
catalogOrder	Integer	False	Order in the catalog
baseMountPoint	String	False	Source raster path
sourceResolution	String	False	Source raster resolution
extraFields	[String]	True	Extra fields map, NA

- `userRequest`: User request configuration, where expected output raster features are defined.

Table 2-3 userRequest Column Data

Column	DataType	Nullable	Description
offset	Long	False	Offset
piece	Integer	False	Piece number
splitSize	Long	False	Split size
bandsToAdd	String	False	Bands to include in output i.e. “1,2,3”

Table 2-3 (Cont.) userRequest Column Data

Column	DataType	Nullable	Description
upperLeftX	Double	True	Coordinate of output in X upper left, used when mosaic is requested
upperLeftY	Double	True	Coordinate of output in Y upper left, used when mosaic is requested
lowerRightX	Double	True	Coordinate of output in X lower right, used when mosaic is requested
lowerRightY	Double	True	Coordinate of output in Y lower right, used when mosaic is requested
width	Integer	True	Output width, used when mosaic is requested
height	Integer	True	Output height, used when mosaic is requested
srid	String	True	Output SRID, used when mosaic is requested
order	String	True	Output order , Ascendant or Descendant, used when mosaic is requested
format	String	True	Output GDALformat, used when mosaic is requested
noData	String	False	Output NODATA value, a # followed by two digits per band, i.e. for 3 band output "#000000"
pixelType	String	True	Output GDAL Data type, used when mosaic is requested
Directory	String	False	Output directory
pixelXWidth	Double	True	Output pixel width, used when mosaic is requested
pixelYWidth	Double	True	Output pixel height, used when mosaic is requested
wkt	String	False	Source projection reference
mosaicWkt	String	True	Output projection reference, used when mosaic is requested
processingClasses	String	True	User processing classes to execute, still not supported in Spark
reducingClasses	String	True	User reducing classes to execute, still not supported in Spark
tempOut	String	True	Temporary output folder when HDFS output is requested, still not supported in Spark
filename	String	False	Output filename

Table 2-3 (Cont.) userRequest Column Data

Column	DataType	Nullable	Description
contextId	String	False	Execution context Id
sourceResolution	String	False	Source raster resolution
catalogOrder	Integer	False	Source raster order in catalog
executeMosaic	Boolean	False	Flag to indicate if mosaic operation is requested or not
osContainer	[String]	True	Object Storage Container and Service, used to connect using swift
swiftConfig	[String]	True	List of swift configuration properties, including url , user and tenant
hdfsConfig	[String]	False	HDFS URL for BDCS

The following example creates a DataFrame and displays information about it:

```
JavaSparkContext sc = processor.getRasterSparkContext();
SpatialRasterJavaRDD<GeneralInfoWritable> spatialRDD =
processor.getProcessSplits();
HiveContext sqlContext = new HiveContext(sc.sc());
DataFrame tileRows = RDDUtils.createDataFrameFromRDD(sqlContext,
spatialRDD.createSpatialTileRows(StorageLevel.DISK_ONLY()));

Row[] rows = tileRows.collect();
System.out.println("First Tile info: ");
System.out.println("Width " + rows[0].getStruct(0).getInt(0));
System.out.println("Height " + rows[0].getStruct(0).getInt(1));
System.out.println("Total width " + rows[0].getStruct(0).getInt(7));
System.out.println("Total height " + rows[0].getStruct(0).getInt(8));
System.out.println("File " + rows[0].getStruct(0).getString(30));

System.out.println("First Tile User request data: ");

System.out.println("Bands to add " + rows[0].getStruct(1).getString(3));
```

2.8.3.4 Using the Spark SQL UDF for Raster Algebra Operations

A Spark UDF `localop` allows the execution of the raster algebra operations described in Map Algebra Operations for processing images using the Hadoop image processor. The operation names and supported data types for the Spark SQL UDF are the same as for Hadoop

Before any query is executed, the driver class must register the UDF and must register the tiles' DataFrame or Dataset<Row> as a temporary table. For example:

```
sqlContext.udf().register("localop", new LocalOperationsFunction(),
DataTypes.createStructType(RDDUtils.createSimpleTileStructField(dataType)));
```

Now that localop UDF is registered, it is ready to be used. This function accepts two parameters:

- A `tileInfo` row
- A string with the raster algebra operations to execute. Multiple operations may be executed in the same query, and they must be separated by a semicolon. For operations that receive parameters, they must be separated by commas.

The function returns the `tileInfo` that was sent to query, but with the pixel data updated based on the executed operations.

Following are some examples for the execution of different operations.

```
String query = "SELECT localop(tileInfo, \"localnot\"),
                       userRequest FROM tiles";

String query = "SELECT localop(tileInfo,\"localadd,456;localdivide,2;
                                     localif,>,0,12;localmultiply,20;
                                     localpow,2;localsubstract,4;
                                     localsqrt;localacos\"),
                                     userRequest FROM tiles";
String query = "SELECT localop(tileInfo,\"localnot;localatan;localcos;
                                     localasin;localtan;localcosh;
                                     localtanh\"), userRequest FROM
tiles";
```

To execute the query (Spark 1.6), enter the following:

```
DataFrame cachedTiles = RDDUtils.queryAndCache(query, sqlContext);
```

This new DataFrame has the updated pixels. You can optionally save the content of a specific tile as a TIF file, in which it will be stored in the configured output directory. For example:

```
Row[] pRows = cachedTiles.collect();
processor.debugTileBySavingTif(pRows[0],
                               processor.getHadoopConfiguration());
```

To execute the mosaic operation (Spark 1.6), first perform any raster algebra processing, and then perform the mosaic operation. A Spark UDF is used for the mosaic operation; it receives the `tileInfo` and `userRequest` columns, and returns the updated `tileInfo` that fits in the mosaic. For example:

```
sqlContext.udf().register("mosaic", new MosaicFunction(),
DataTypes.createStructType(RDDUtils.createSimpleTileStructField(dataTyp
e)));
DataFrame mosaicTiles = RDDUtils.queryAndCache(queryMosaic,
sqlContext);
```

After the processing is done, you can put together the tiles into the output raster by using `ProcessedRasterCreator`, which receives a temporary HDFS directory for internal work, the DataFrame to merge, and the Spark Context from the Hadoop configuration. This will create the expected output raster in the specified output directory. For example:

```
try {
        ProcessedRasterCreator creator = new ProcessedRasterCreator();
     creator.create(new Text("createOutput"), mosaicTiles,
                    sc.hadoopConfiguration());
     LOG.info("Finished");
} catch (Exception e) {
            LOG.error("Failed processor job due to " + e.getMessage());
            throw e;
     }
```

To execute the query in Spark 2.2, enter the following:

```
SparkSession spark = SparkSession
                      .builder()
                      .appName("Java Spark SQL raster processor")
                      .getOrCreate();

Dataset<Row> tileRows =
DatasetSupport.createDatasetFromRDD(spark.sqlContext(),
spatialRDD.createSpatialTileRows(StorageLevel.DISK_ONLY()));

spark.sqlContext().udf().register("localop", new LocalOperationsFunction(),
DataTypes.createStructType(RDDUtils.createSimpleTileStructField(dataType)));
tileRows.createTempView("tiles");

String query = "SELECT localop(tileInfo, \"localnot\"), userRequest FROM
tiles";
Dataset<Row> cachedTiles = DatasetSupport.queryAndCache(query,
spark.sqlContext());
```

This new Dataset has the updated pixels. You can optionally save the content of a specific tile as a TIF file, which will be stored in the configured output directory. For example:

```
List<Row> pRows = cachedTiles.collectAsList();
processor.debugTileBySavingTif(pRows.get(0),
processor.getHadoopConfiguration());
```

To execute the mosaic operation (Spark 2.2), first perform any raster algebra processing, and then perform the mosaic operation. A new Spark UDF is used for the mosaic operation; it receives the tileInfo and userRequest columns, and returns the updated tileInfo that fits in the mosaic. For example:

```
spark.sqlContext().udf().register("mosaic", new MosaicFunction(),
DataTypes.createStructType(RDDUtils.createSimpleTileStructField(dataType)));
cachedTiles.createTempView("processedTiles");
String queryMosaic = "SELECT mosaic(tileInfo, userRequest), userRequest FROM
processedTiles";
```

```
Dataset<Row> mosaicTiles = DatasetSupport.queryAndCache(queryMosaic,
spark.sqlContext());
```

After the processing is done, you can put together the tiles into the output raster by using `ProcessedRasterCreator`, which receives a temporary HDFS directory for internal work, the DataFrame to merge, and the Spark Context from the Hadoop configuration. This will create the expected output raster in the specified output directory. For example:

```
 try {
                ProcessedRasterCreator creator = new
ProcessedRasterCreator();
                creator.create(new Text("createOutput"),  (Row[])
mRows.toArray(), sc.hadoopConfiguration());
                LOG.info("Finished");
            } catch (Exception e) {
                LOG.error("Failed processor job");
                throw e;
            }
```

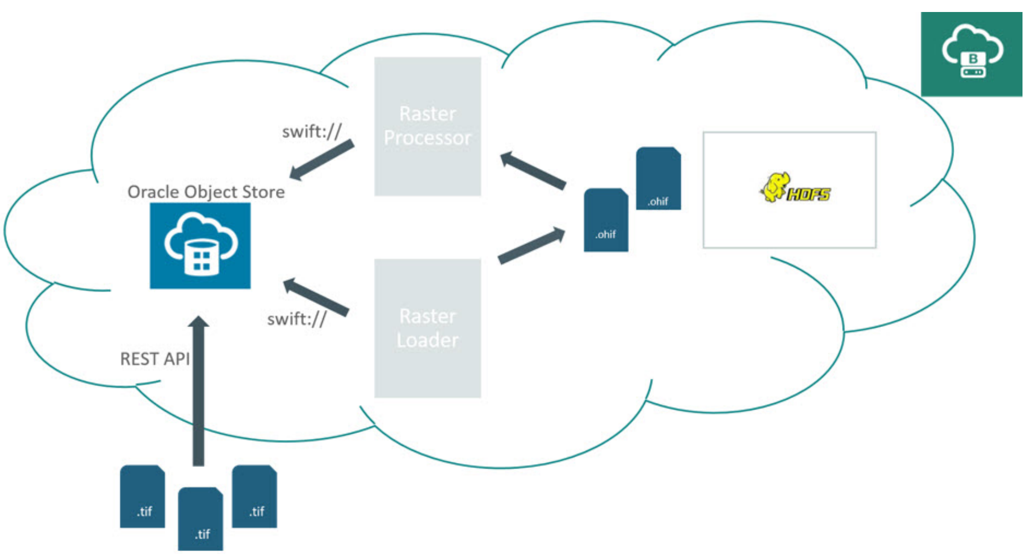
2.9 Spatial Raster Processing Support in Big Data Cloud Service

Oracle Big Data Spatial Raster Processing is supported in Big Data Cloud Service (BDCS) by making use of the Oracle Object Storage platform.

NFS shared directories SHARED_DIR and ALL_ACCESS_DIR are not required in this configuration, instead a connection and valid account for Oracle Object Storage is required.

The general flow of raster processing in Big Data Cloud Service is shown in the following figure.

Figure 2-1 Spatial Raster Processing in Big Data Cloud Service



As shown in the preceding figure:

1. Rasters (.tif) are loaded in Oracle Object Storage.
2. Big Data Spatial Raster Loader imports them and converts them to OHIF (.ohif) into HDFS.
3. Big Data Spatial Raster Processor stores the output in Oracle Object Storage.

- Oracle Object Storage Container Configuration
- BDCS Loader Configuration
- Raster Processing in BDCS
- Oracle Object Storage Password Configuration

Oracle Object Storage Container Configuration

You must have an Oracle Object Storage connection and a valid user. You must create a container and store your input rasters there. For information about how to create the container and add rasters to bject Storage, see REST API for Standard Storage in Oracle Cloud Infrastructure Object Storage Classic.

You must configure the Object Storage endpoint and login details for your jobs by using the following properties.

- fs.swift.service.*YOURSERVICEID*.auth.url: Endpoint authentication RL.
- fs.swift.service.*YOURSERVICEID*.public: Indicates if all URLs are public.
- fs.swift.service.*YOURSERVICEID*.tenant: Tenant to connect (required).
- fs.swift.service.*YOURSERVICEID*.username: Username to authenticate.

These properties must be set using Hadoop Configuration.

BDCS Loader Configuration

When loading rasters from Object Storage, you must provide a directory with enough space to download from Object Storage all the rasters that will be processed using the Spatial Raster Processor. This directory must be located in the node where the job is being called.

Use a command such as in the following example:

```
hadoop jar hadoop-imageloader.jar
-files <SOURCE_IMGS_PATH>
-out <HDFS_OUTPUT_FOLDER>
-gdal <GDAL_LIB_PATH>
-gdalData <GDAL_DATA_PATH>
[-overlap <OVERLAPPING_PIXELS>]
[-thumbnail <THUMBNAIL_PATH>]
[-rasterSource objectStore]
[-containerService containername.service]
[-workDir /system123/scratch/user3]
[-credential spatialraster]
```

Where:

- `rasterSource`: Set to objectStore if the rasters are in Oracle Object Storage, if not set it follows default flow.

- `containerService`: Specifies the Object Storage container and service where the rasters are stored. This is required field.
- `workDir`: Local temporary directory to download Object Storage rasters, this is required for Object Storage loads. This is required field.
- `credential`: Specifies the ID of the credential stored in Credential Store to secure Object Storage password.

The same properties must be set when using Spatial Raster Processing API. For example:

```
loader.setObjectRasterSource();
loader.setObjectStorageContainer("spatial.melli");
loader.setObjectStorageTemporaryDir("/opt/test");
loader.setObjectStorageCredential("spatialraster");
```

Raster Processing in BDCS

To request a raster process in BDCS, you must specify the following elements in the User Request Configuration XML:

- `directory type`: Set to *OS* for Oracle Object Storage Processing.
- `directory container`: Specifies the container and service where the output will be stored *<containername.service>*, which is required for Object Storage Processing.
- `directory credential`: Set the ID of the credential stored in Credential Store to the secure Object Storage password.
- `directory`: Output directory in Object Storage, which is required for Object Storage Processing.
- `tempFsFolder`: HDFS temporary directory to store mapper processing results, which is required for Object Storage Processing.

The XML should look like this:

```
<directory type="OS" container="spatial.user3"
credential="spatialraster">/user/oracle/output/<directory>
<tempFsFolder>tempOS/<tempFsFolder>
```

The same properties must be set when using Spatial Raster Processing API. For example:

```
mosaic.setFileSystem(RasterProcessorJob.OBJECT_STORAGE);
mosaic.setOSContainer("spatial.user3");
//Output directory in Object Storage
mosaic.setDirectory("/user/oracle/RasterAPIOutput");
//HDFS directory for temporary local processing results
mosaic.setTemporaryFolder("processtmpOS");
//Credential ID in Credential Store to connect to Object Storage
mosaic.setCredential("spatialraster");
```

Oracle Object Storage Password Configuration

To connect to Oracle Object Storage to download the input rasters or to store the processing results, you must provide a password. To secure this password, you must store it in Cluster Credential Store so that it is not passed in clear text in command line parameters or job code.

To store credentials in the credential store for a cluster:

1. Open the cluster console for the cluster.
2. Click **Settings**, then click the **Credentials** tab.
3. In the User Credentials section, click **New Credential** to create a new credential.
4. In the **Key** field, enter the desired name or identifier for the credential. For example: `database_password`.
5. In the **Value** field, enter the value for the credential, then click **Save**.

The identifier used to store the credential (key field) will be provided to the Raster Loader and Processor as the *credential*. The framework will retrieve the password using this identifier and use it to connect to Object Storage.

Use the `oracle` user to execute the jobs in order to correctly retrieve the password, because Credential Store will currently grant access to this password only to `oracle`.

2.10 Oracle Big Data Spatial Vector Analysis

Oracle Big Data Spatial Vector Analysis is a Spatial Vector Analysis API, which runs as a Hadoop job and provides MapReduce components for spatial processing of data stored in HDFS.

These components make use of the Spatial Java API to perform spatial analysis tasks. There is a web console provided along with the API.

- Multiple Hadoop API Support
- Spatial Indexing
- Using MVSuggest
- Spatial Filtering
- Classifying Data Hierarchically
- Generating Buffers
- Spatial Binning
- Spatial Clustering
- Spatial Join
- Spatial Partitioning
- RecordInfoProvider
- HierarchyInfo
- Using JGeometry in MapReduce Jobs
- Support for Different Data Sources
- Job Registry

- Tuning Performance Data of Job Running Times Using the Vector Analysis API

> **See Also:**
>
> See the following topics for understanding the implementation details:
>
> - RecordInfoProvider
> - HierarchyInfo
> - Using JGeometry in MapReduce Jobs
> - Tuning Performance Data of Job Running Times Using the Vector Analysis API

2.10.1 Multiple Hadoop API Support

Oracle Big Data Spatial Vector Analysis provides classes for both the old and new (context objects) Hadoop APIs. In general, classes in the `mapred` package are used with the old API, while classes in the `mapreduce` package are used with the new API

The examples in this guide use the old Hadoop API; however, all the old Hadoop Vector API classes have equivalent classes in the new API. For example, the old class `oracle.spatial.hadoop.vector.`**`mapred`**`.job.SpatialIndexing` has the equivalent new class named `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.SpatialIndexing`. In general, and unless stated otherwise, only the change from `mapred` to `mapreduce` is needed to use the new Hadoop API Vector classes.

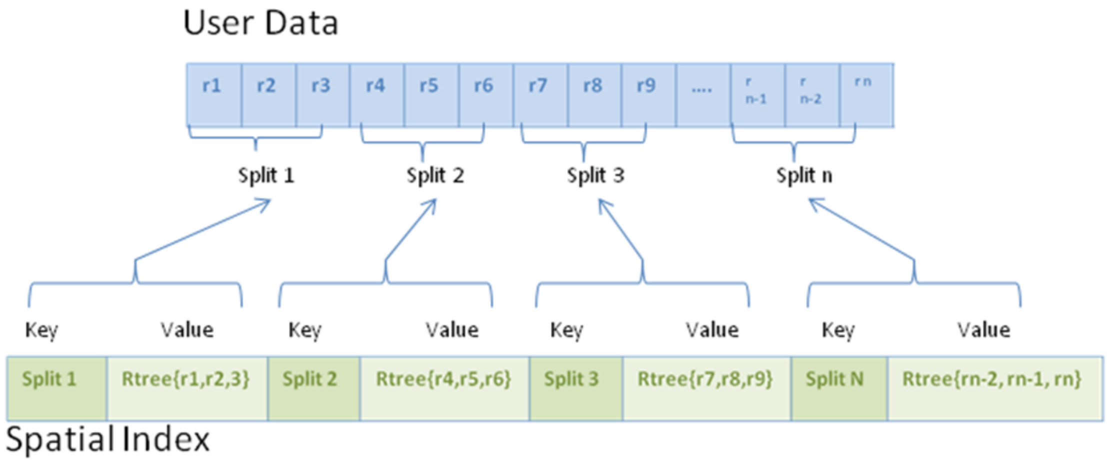
Classes such as `oracle.spatial.hadoop.vector.RecordInfo`, which are not in the mapred or mapreduce package, are compatible with both Hadoop APIs.

2.10.2 Spatial Indexing

A spatial index is in the form of a key/value pair and generated as a Hadoop MapFile. Each MapFile entry contains a spatial index for one split of the original data. The key and value pair contains the following information:

- Key: a split identifier in the form: path + start offset + length.
- Value: a spatial index structure containing the actual indexed records.

The following figure depicts a spatial index in relation to the user data. The records are represented as r1, r2, and so on. The records are grouped into splits (Split 1, Split 2, Split 3, Split n). Each split has a Key-Value pair where the key identifies the split and the value identifies an Rtree index on the records in that split.



- Spatial Indexing Class Structure
- Configuration for Creating a Spatial Index
- Spatial Index Metadata
- Input Formats for a Spatial Index
- Support for GeoJSON and Shapefile Formats
- Removing a Spatial Index

2.10.2.1 Spatial Indexing Class Structure

Records in a spatial index are represented using the class `oracle.spatial.hadoop.vector.RecordInfo`. A `RecordInfo` typically contains a subset of the original record data and a way to locate the record in the file where it is stored. The specific `RecordInfo` data depends on two things:

- `InputFormat` used to read the data
- `RecordInfoProvider` implementation, which provides the record's data

The fields contained within a `RecordInfo`:

- Id: Text field with the record Id.
- Geometry: `JGeometry` field with the record geometry.
- Extra fields: Additional optional fields of the record can be added as name-value pairs. The values are always represented as text.
- Start offset: The position of the record in a file as a byte offset. This value depends on the `InputFormat` used to read the original data.
- Length: The original record length in bytes.
- Path: The file path can be added optionally. This is optional because the file path can be known using the spatial index entry key. However, to add the path to the `RecordInfo` instances when a spatial index is created, the value of the configuration property `oracle.spatial.recordInfo.includePathField` key is set to `true`.

2.10.2.2 Configuration for Creating a Spatial Index

A spatial index is created using a combination of `FileSplitInputFormat`, `SpatialIndexingMapper`, `InputFormat`, and `RecordInfoProvider`, where the last two are provided by the user. The following code example shows part of the configuration needed to run a job that creates a spatial index for the data located in the HDFS folder `/user/data`.

```
//input

conf.setInputFormat(FileSplitInputFormat.class);
FileSplitInputFormat.setInputPaths(conf, new Path("/user/data"));
FileSplitInputFormat.setInternalInputFormatClass(conf, GeoJsonInputFormat.class);
FileSplitInputFormat.setRecordInfoProviderClass(conf,
GeoJsonRecordInfoProvider.class);

//output

conf.setOutputFormat(MapFileOutputFormat.class);
FileOutputFormat.setOutputPath(conf, new Path("/user/data_spatial_index"));

//mapper

conf.setMapperClass(SpatialIndexingMapper.class);
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(RTreeWritable.class);
```

In this example,

- The `FileSplitInputFormat` is set as the job `InputFormat`. `FileSplitInputFormat` is a subclass of `CompositeInputFormat` (`WrapperInputFormat` in the new Hadoop API version), an abstract class that uses another `InputFormat` implementation (`internalInputFormat`) to read the data. The internal `InputFormat` and the `RecordInfoProvider` implementations are specified by the user and they are set to `GeoJsonInputFormat` and `GeoJsonRecordInfoProvider`, respectively.
- The `MapFileOutputFormat` is set as the `OutputFormat` in order to generate a `MapFile`
- The mapper is set to `SpatialIndexingMappper`. The mapper output key and value types are `Text` (splits identifiers) and `RTreeWritable` (the actual spatial indexes).
- No reducer class is specified so it runs with the default reducer. The reduce phase is needed to sort the output MapFile keys.

Alternatively, this configuration can be set easier by using the `oracle.spatial.hadoop.vector.mapred.job.SpatialIndexing` class. `SpatialIndexing` is a job driver that creates a spatial index. In the following example, a `SpatialIndexing` instance is created, set up, and used to add the settings to the job configuration by calling the `configure()` method. Once the configuration has been set, the job is launched.

```
SpatialIndexing<LongWritable, Text> spatialIndexing = new
SpatialIndexing<LongWritable, Text>();

//path to input data

spatialIndexing.setInput("/user/data");
```

```
//path of the spatial index to be generated
spatialIndexing.setOutput("/user/data_spatial_index");
//input format used to read the data
spatialIndexing.setInputFormatClass(TextInputFormat.class);
//record info provider used to extract records information
spatialIndexing.setRecordInfoProviderClass(TwitterLogRecordInfoProvider.class);
//add the spatial indexing configuration to the job configuration
spatialIndexing.configure(jobConf);
//run the job
JobClient.runJob(jobConf);
```

2.10.2.3 Spatial Index Metadata

A metadata file is generated for every spatial index that is created. The spatial index metadata can be used to quickly find information related to a spatial index, such as the number of indexed records, the minimum bounding rectangle (MBR) of the indexed data, and the paths of both the spatial index and the indexed source data. The spatial index metadata can be retrieved using the spatial index name.

A spatial index metadata file contains the following information:

- Spatial index name
- Path to the spatial index
- Number of indexed records
- Number of local indexes
- Extra fields contained in the indexed records
- Geometry layer information such as th SRID, dimensions, tolerance, dimension boundaries, and whether the geometries are geodetic or not
- The following information for each of the local spatial index files: path to the indexed data, path to the local index, and MBR of the indexed data

The following metadata proeprties can be set when creating a spatial index using the `SpatialIndexing` class:

- `indexName`: Name of the spatial index. If not set, the output folder name is used.
- `metadataDir`: Path to the directory where the metadata file will be stored.
  - By default, it will be stored in the following path relative to the user directory: `oracle_spatial/index_metadata`. If the user is `hdfs`, it will be `/user/hdfs/oracle_spatial/index_metadata`.
- `overwriteMetadata`: If set to `true`, then when a spatial index metadata file already exists for a spatial index with the same `indexName` in the current `metadataDir`, the spatial index metadata will be overwritten. If set to `false` and if a spatial index metadata file already

exists for a spatial index with the same `indexName` in the current `metadataDir`, then an error is raised.

The following example sets the metadata directory and spatial index name, and specifies to overwrite any existing metadata if the index already exists:

```
spatialIndexing.setMetadataDir("/user/hdfs/myIndexMetadataDir");
spatialIndexing.setIndexName("testIndex");
spatialIndexing.setOverwriteMetadata(true);
```

An existing spatial index can be passed to other jobs by specifying only the `indexName` and optionally the `indexMetadataDir` where the index metadata can be found. When the index name is provided, there is no need to specify the spatial index path and the input format.

The following job drivers accept the `indexName` as a parameter:

- `oracle.spatial.hadoop.vector.mapred.job.Categorization`
- `oracle.spatial.hadoop.vector.mapred.job.SpatialFilter`
- `oracle.spatial.hadoop.vector.mapred.job.Binning`
- Any driver that accepts `oracle.spatial.hadoop.vector.InputDataSet`, such as `SpatialJoin` and `Partitioning`

If the index name is not found in the `indexMetadataDir` path, an error is thrown indicating that the spatial index could not be found.

The following example shows a spatial index being set as the input data set for a binning job:

```
Binning binning = new Binning();
binning.setIndexName("indexExample");
binning.setIndexMetadataDir("indexMetadataDir");
```

2.10.2.4 Input Formats for a Spatial Index

An `InputFormat` must meet the following requisites to be supported:

- It must be a subclass of `FileInputFormat`.
- The `getSplits()` method must return either `FileSplit` or `CombineFileSplit` split types.
- For the old Hadoop API, the `RecordReader`’s `getPos()` method must return the current position to track back a record in the spatial index to its original record in the user file. If the current position is not returned, then the original record cannot be found using the spatial index.

  However, the spatial index still can be created and used in operations that do not require the original record to be read. For example, additional fields can be added as extra fields to avoid having to read the whole original record.

> **Note:**
>
> The spatial indexes are created for each split as returned by the `getSplits()` method. When the spatial index is used for filtering (see Spatial Filtering), it is recommended to use the same `InputFormat` implementation than the one used to create the spatial index to ensure the splits indexes can be found.

The `getPos()` method has been removed from the Hadoop new API; however, `org.apache.hadoop.mapreduce.lib.input.TextInputFormat` and `CombineTextInputFormat` are supported, and it is still possible to get the record start offsets.

Other input formats from the new API are supported, but the record start offsets will not be contained in the spatial index. Therefore, it is not possible to find the original records. The requirements for a new API input format are the same as for the old API. However, they must be translated to the new APIs `FileInputFormat`, `FileSplit`, and `CombineFileSplit`.

2.10.2.5 Support for GeoJSON and Shapefile Formats

The Vector API comes with `InputFormat` and `RecordInfoProvider` implementations for GeoJSON and Shapefile file formats.

The following `InputFormat/RecordInfoProvider` pairs can be used to read and interpret GeoJSON and ShapeFiles, respectively:

```
oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat /
oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProvider

oracle.spatial.hadoop.vector.shapefile.mapred.ShapeFileInputFormat /
oracle.spatial.hadoop.vector.shapefile.ShapeFileRecordInfoProvider
```

More information about the usage and properties is available in the Javadoc.

2.10.2.6 Removing a Spatial Index

A previously generated spatial index can be removed by executing the following.

```
oracle.spatial.hadoop.vector.util.Tools removeSpatialIndex indexName=<INDEX_NAME>
[indexMetadataDir=<PATH>] [removeIndexFiles=<true|false*>]
```

Where:

- `indexName`: Name of a previously generated index.
- `indexMetadataDir` (optional): Path to the index metadata directory. If not specified, the following path relative to the user directory will be used: oracle_spatial/index_metadata
- `removeIndexFiles` (optional): `true` if generated index map files need to be removed in addition to the index metadata file. By default, it is `false`.

2.10.3 Using MVSuggest

`MVSuggest` can be used at the time of spatial indexing to get an approximate location for records that do not have geometry but have some text field. This text field can be used to determine the record location. The geometry returned by `MVSuggest` is used to include the record in the spatial index.

Because it is important to know the field containing the search text for every record, the `RecordInfoProvider` implementation must also implement `LocalizableRecordInfoProvider`. Alternatively, the configuration parameter `oracle.spatial.recordInfo.locationField` can be set with the name of the field containing the search text. For more information, see the Javadoc for `LocalizableRecordInfoProvider`.

A standalone version of `MVSuggest` is shipped with the Vector API and it can be used in some jobs that accept the `MVSConfig` as an input parameter.

The following job drivers can work with `MVSuggest` and all of them have the `setMVSConfig()` method which accepts an instance of `MVSConfig`:

- oracle.spatial.hadoop.vector.mapred.job.SpatialIndexing: has the option of using `MVSuggest` to get approximate spatial location for records which do not contain geometry.
- oracle.spatial.hadoop.vector.mapred.job.Categorization: `MVSuggest` can be used to assign a record to a specific feature in a layer, for example, the feature California in the USA states layer.
- oracle.spatial.hadoop.vector.mapred.job.SuggestService: A simple job that generates a file containing a search text and its match per input record.

The MVSuggest configuration is passed to a job using the MVSConfig or the LocalMVSConfig classes. The basic MVSuggest properties are:

- `serviceLocation`: It is the minimum property required in order to use `MVSuggest`. It contains the path or URL where the `MVSuggest` directory is located or in the case of a URL, where the `MVSuggest` service is deployed.
- `serviceInterfaceType`: the type of `MVSuggest` implementation used. It can be LOCAL(default) for a standalone version and WEB for the web service version.
- `matchLayers`: an array of layer names used to perform the searches.

When using the standalone version of `MVSuggest`, you must specify an `MVSuggest` directory or repository as the `serviceLocation`. An `MVSuggest` directory must have the following structure:

```
mvsuggest_config.json
repository folder
   one or more layer template files in .json format
   optionally, a _config_ directory
   optionally, a _geonames_ directory
```

The `examples` folder comes with many layer template files and a `_config_` directory with the configuration for each template.

It is possible to set the repository folder (the one that contains the templates) as the mvsLocation instead of the whole `MVSuggest` directory. In order to do that, the class `LocalMVSConfig` can be used instead of `MVSConfig` and the `repositoryLocation` property must be set to `true` as shown in the following example:

```
LocalMVSConfig lmvsConf = new LocalMVSConfig();
lmvsConf.setServiceLocation("file:///home/user/mvs_dir/repository/");
lmvsConf.setRepositoryLocation(true);
lmvsConf.setPersistentServiceLocation("/user/hdfs/hdfs_mvs_dir");
spatialIndexingJob.setMvsConfig(lmvsConf);
```

The preceding example sets a repository folder as the MVS service location. `setRepositoryLocation` is set to `true` to indicate that the service location is a repository instead of the whole `MVSuggest` directory. When the job runs, a whole `MVSuggest` directory will be created using the given repository location; the repository will be indexed and will be placed in a temporary folder while the job finishes. The previously indexed `MVSuggest` directory can be persisted so it can be used later. The preceding example saves the generated `MVSuggest` directory in the HDFS path `/user/hdfs/hdfs_mvs_dir`. Use the MVSDirectory if the `MVSuggest` directory already exists.

2.10.4 Spatial Filtering

Once the spatial index has been generated, it can be used to spatially filter the data. The filtering is performed before the data reaches the mapper and while it is being read. The following sample code example demonstrates how the `SpatialFilterInputFormat` is used to spatially filter the data.

```
//set input path and format

FileInputFormat.setInputPaths(conf, new Path("/user/data/"));
conf.setInputFormat(SpatialFilterInputFormat.class);

//set internal input format

SpatialFilterInputFormat.setInternalInputFormatClass(conf, TextInputFormat.class);
if( spatialIndexPath != null )
{

     //set the path to the spatial index and put it in the distributed cache

     boolean useDistributedCache = true;
     SpatialFilterInputFormat.setSpatialIndexPath(conf, spatialIndexPath,
useDistributedCache);
}
else
{
     //as no spatial index is used a RecordInfoProvider is needed

     SpatialFilterInputFormat.setRecordInfoProviderClass(conf,
TwitterLogRecordInfoProvider.class);
}

//set spatial operation used to filter the records

SpatialOperationConfig spatialOpConf = new SpatialOperationConfig();
spatialOpConf.setOperation(SpatialOperation.IsInside);
spatialOpConf.setJsonQueryWindow("{\"type\":\"Polygon\", \"coordinates\":[[-106.64595,
25.83997, -106.64595, 36.50061, -93.51001, 36.50061, -93.51001, 25.83997 , -106.64595,
25.83997]]}");
spatialOpConf.setSrid(8307);
spatialOpConf.setTolerance(0.5);
spatialOpConf.setGeodetic(true);
```

`SpatialFilterInputFormat` has to be set as the job's `InputFormat`. The `InputFormat` that actually reads the data must be set as the internal `InputFormat`. In this example, the internal `InputFormat` is `TextInputFormat`.

If a spatial index is specified, it is used for filtering. Otherwise, a `RecordInfoProvider` must be specified in order to get the records geometries, in which case the filtering is performed record by record.

As a final step, the spatial operation and query window to perform the spatial filter are set. It is recommended to use the same internal `InputFormat` implementation used when the spatial index was created or, at least, an implementation that uses the same criteria to generate the splits. For details see "Input Formats for a Spatial Index."

If a simple spatial filtering needs to be performed (that is, only retrieving records that interact with a query window), the built-in job driver `oracle.spatial.hadoop.vector.mapred.job.SpatialFilter` can be used instead. This job driver accepts indexed or non-indexed input and a `SpatialOperationConfig` to perform the filtering.

- Filtering Records
- Filtering Using the Input Format

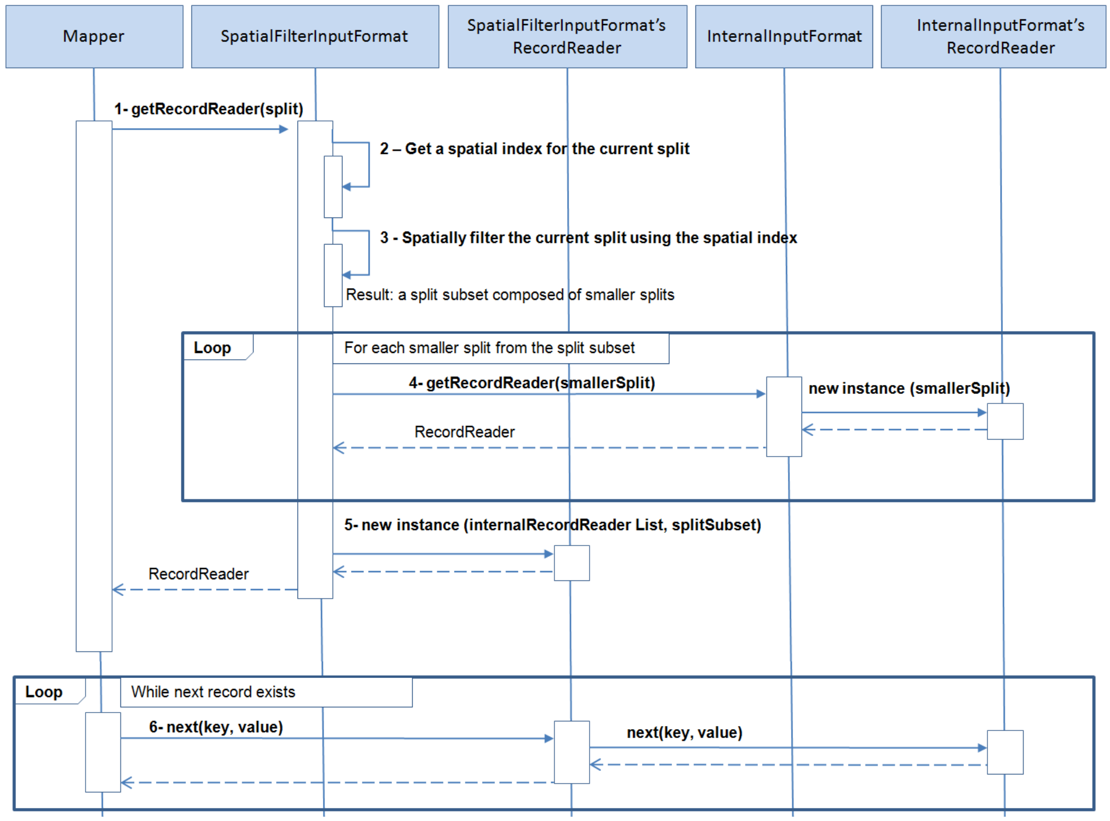
2.10.4.1 Filtering Records

The following steps are executed when records are filtered using the `SpatialFilterInputFormat` and a spatial index.

1. `SpatialFilterInputFormat getRecordReader()` method is called when the mapper requests a `RecordReader` for the current split.
2. The spatial index for the current split is retrieved.
3. A spatial query is performed over the records contained in it using the spatial index.

   As a result, the ranges in the split that contains records meeting the spatial filter are known. For example, if a split goes from the file position 1000 to 2000, upon executing the spatial filter it can be determined that records that fulfill the spatial condition are in the ranges 1100-1200, 1500-1600 and 1800-1950. So the result of performing the spatial filtering at this stage is a subset of the original filter containing smaller splits.
4. An Internal`InputFormat RecordReader` is requested for every small split from the resulting split subset.
5. A RecordReader is returned to the caller mapper. The returned RecordReader is actually a wrapper RecordReader with one or more RecordReaders returned by the internal `InputFormat`.
6. Every time the mapper calls the RecordReader, the call to next method to read a record is delegated to the internal RecordReader.

These steps are shown in the following spatial filter interaction diagram.



2.10.4.2 Filtering Using the Input Format

A previously generated Spatial Index can be read using the input format implementation `oracle.spatial.hadoop.vector.mapred.input.SpatialIndexInputFormat` (or its new Hadoop API equivalent with the `mapreduce` package instead of `mapred`). `SpatialIndexInputFormat` is used just like any other `FileInputFormat` subclass in that it takes an input path and it is set as the job's input format. The key and values returned are the id `(Text)` and record information `(RecordInfo)` of the records stored in the spatial index.

Aditionally, a spatial filter opertion can be performed by specifying a spatial operation configuration to the input format, so that only the records matching some spatial interaction will be returned to a mapper. The following example shows how to configure a job to read a spatial index to retrieve all the records that are inside a specific area.

```
JobConf conf = new JobConf();
conf.setMapperClass(MyMapper.class);
conf.setInputFormat(SpatialIndexInputFormat.class);
SpatialOperationConfig spatialOpConf = new SpatialOperationConfig();
spatialOpConf.setOperation(SpatialOperation.IsInside);
spatialOpConf.setQueryWindow(JGeometry.createLinearPolygon(new double[]
{47.70, -124.28, 47.70,  -95.12, 35.45, -95.12, 35.45, -124.28, 47.70,
-124.28}, 2, 8307));
SpatialIndexInputFormat.setFilterSpatialOperationConfig(spatialOpConf, conf);
```

The mapper in the preceding example can add a nonspatial filter by using the `RecordInfo` extra fields, as shown in the following example.

```
public class MyMapper extends MapReduceBase implements Mapper<Text,
RecordInfo, Text, RecordInfo>{
    @Override
    public void map(Text key, RecordInfo value, OutputCollector<Text,
RecordInfo> output, Reporter reporter)
            throws IOException {
        if( Integer.valueOf(value.getField("followers_count")) > 0){
            output.collect(key, value);
        }
    }
}
```

2.10.5 Classifying Data Hierarchically

The Vector Analysis API provides a way to classify the data into hierarchical entities. For example, in a given set of catalogs with a defined level of administrative boundaries such as continents, countries and states, it is possible to join a record of the user data to a record of each level of the hierarchy data set. The following example generates a summary count for each hierarchy level, containing the number of user records per continent, country, and state or province:

```
Categorization catJob = new Categorization();
//set a spatial index as the input

catJob.setIndexName("indexExample");

//set the job's output

catJob.setOutput("hierarchy_count");

//set HierarchyInfo implementation which describes the world administrative
boundaries hierarchy

catJob.setHierarchyInfoClass( WorldDynaAdminHierarchyInfo.class );

//specify the paths of the hierarchy data

Path[] hierarchyDataPaths = {
                new Path("file:///home/user/catalogs/world_continents.json"),
                new Path("file:///home/user/catalogs/world_countries.json"),
                new Path("file:///home/user/catalogs/
world_states_provinces.json")};
catJob.setHierarchyDataPaths(hierarchyDataPaths);

//set the path where the index for the previous hierarchy data will be generated

catJob.setHierarchyIndexPath(new Path("/user/hierarchy_data_index/"));

//setup the spatial operation which will be used to join records from the two
datasets (spatial index and hierarchy data).
SpatialOperationConfig spatialOpConf = new SpatialOperationConfig();
spatialOpConf.setOperation(SpatialOperation.IsInside);
spatialOpConf.setSrid(8307);
spatialOpConf.setTolerance(0.5);
spatialOpConf.setGeodetic(true);
```

```
catJob.setSpatialOperationConfig(spatialOpConf);

//add the previous setup to the job configuration

catJob.configure(conf);

//run the job
RunningJob rj = JobClient.runJob(conf);
```

The preceding example uses the `Categorization` job driver. The configuration can be divided into the following categories:

- Input data: A previously generated spatial index (received as the job input).
- Output data: A folder that contains the summary counts for each hierarchy level.
- Hierarchy data configuration: This contains the following:
  - `HierarchyInfo` class: This is an implementation of `HierarchyInfo` class in charge of describing the current hierarchy data. It provides the number of hierarchy levels, level names, and the data contained at each level.
  - Hierarchy data paths: This is the path to each one of the hierarchy catalogs. These catalogs are read by the `HierarchyInfo` class.
  - Hierarchy index path: This is the path where the hierarchy data index is stored. Hierarchy data needs to be preprocessed to know the parent-child relationships between hierarchy levels. This information is processed once and saved at the hierarchy index, so it can be used later by the current job or even by any other jobs.
- Spatial operation configuration: This is the spatial operation to be performed between records of the user data and the hierarchy data in order to join both datasets. The parameters to set here are the Spatial Operation type (IsInside), SRID (8307), Tolerance (0.5 meters), and whether the geometries are Geodetic (true).

Internally, the `Categorization.configure()` method sets the mapper and reducer to be `SpatialHierarchicalCountMapper` and `SpatialHierarchicalCountReducer`, respectively. `SpatialHierarchicalCountMapper`'s output key is a hierarchy entry identifier in the form `hierarchy_level` + `hierarchy_entry_id`. The mapper output value is a single count for each output key. The reducer sums up all the counts for each key.

> **Note:**
>
> The entire hierarchy data may be read into memory and hence the total size of all the catalogs is expected to be significantly less than the user data. The hierarchy data size should not be larger than a couple of gigabytes.

If you want another type of output instead of counts, for example, a list of user records according to the hierarchy entry. In this case, the `SpatialHierarchicalJoinMapper` can be used. The `SpatialHierarchicalJoinMapper` output value is a `RecordInfo` instance, which can be gathered in a user-defined reducer to produce a different output. The following user-defined reducer generates a MapFile for each hierarchy level using the `MultipleOutputs` class. Each MapFile has the hierarchy entry ids as keys and `ArrayWritable` instances containing the matching records for each hierarchy entry as values. The following is an user-defined reducer that returns a list of records by hierarchy entry:

```
public class HierarchyJoinReducer extends MapReduceBase implements Reducer<Text,
RecordInfo, Text, ArrayWritable> {

       private MultipleOutputs mos = null;
       private Text outKey = new Text();
       private ArrayWritable outValue = new ArrayWritable( RecordInfo.class );

       @Override
       public void configure(JobConf conf)
       {
         super.configure(conf);

         //use MultipleOutputs to generate different outputs for each hierarchy
level

         mos = new MultipleOutputs(conf);
        }
        @Override
        public void reduce(Text key, Iterator<RecordInfo> values,
                          OutputCollector<Text, RecordInfoArrayWritable> output,
Reporter reporter)
                          throws IOException
        {

          //Get the hierarchy level name and the hierarchy entry id from the key

          String[] keyComponents =
HierarchyHelper.getMapRedOutputKeyComponents(key.toString());
          String hierarchyLevelName = keyComponents[0];
          String entryId = keyComponents[1];
          List<Writable> records = new LinkedList<Writable>();

          //load the values to memory to fill output ArrayWritable

          while(values.hasNext())
          {
            RecordInfo recordInfo = new RecordInfo( values.next() );
            records.add( recordInfo );
          }
          if(!records.isEmpty())
          {

            //set the hierarchy entry id as key

            outKey.set(entryId);

            //list of records matching the hierarchy entry id

            outValue.set( records.toArray(new Writable[]{} ) );

            //get the named output for the given hierarchy level

            hierarchyLevelName =
FileUtils.toValidMONamedOutput(hierarchyLevelName);
            OutputCollector<Text, ArrayWritable> mout =
mos.getCollector(hierarchyLevelName, reporter);

           //Emit key and value

           mout.collect(outKey, outValue);
          }
```

```
}

        @Override
        public void close() throws IOException
        {
          mos.close();
        }
}
```

The same reducer can be used in a job with the following configuration to generate a list of records according to the hierarchy levels:

```
JobConf conf = new JobConf(getConf());

//input path

FileInputFormat.setInputPaths(conf, new Path("/user/data_spatial_index/") );

//output path

FileOutputFormat.setOutputPath(conf, new Path("/user/records_per_hier_level/") );

//input format used to read the spatial index

conf.setInputFormat( SequenceFileInputFormat.class);

//output format: the real output format will be configured for each multiple output
later

conf.setOutputFormat(NullOutputFormat.class);

//mapper

conf.setMapperClass( SpatialHierarchicalJoinMapper.class );
conf.setMapOutputKeyClass(Text.class);
conf.setMapOutputValueClass(RecordInfo.class);

//reducer

conf.setReducerClass( HierarchyJoinReducer.class );
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(ArrayWritable.class);

////////////////////////////////////////////////////////////////////

//hierarchy data setup

//set HierarchyInfo class implementation

conf.setClass(ConfigParams.HIERARCHY_INFO_CLASS, WorldAdminHierarchyInfo.class,
HierarchyInfo.class);

//paths to hierarchical catalogs

Path[] hierarchyDataPaths = {
new Path("file:///home/user/catalogs/world_continents.json"),
new Path("file:///home/user/catalogs/world_countries.json"),
new Path("file:///home/user/catalogs/world_states_provinces.json")};

//path to hierarchy index
```

```
Path hierarchyDataIndexPath = new Path("/user/hierarchy_data_index/");

//instantiate the HierarchyInfo class to index the data if needed.

HierarchyInfo hierarchyInfo = new WorldAdminHierarchyInfo();
hierarchyInfo.initialize(conf);

//Create the hierarchy index if needed. If it already exists, it will only load
the hierarchy index to the distributed cache

HierarchyHelper.setupHierarchyDataIndex(hierarchyDataPaths,
hierarchyDataIndexPath, hierarchyInfo, conf);

///////////////////////////////////////////////////////////////////

//setup the multiple named outputs:

int levels = hierarchyInfo.getNumberOfLevels();
for(int i=1; i<=levels; i++)
{
    String levelName = hierarchyInfo.getLevelName(i);

    //the hierarchy level name is used as the named output

    String namedOutput = FileUtils.toValidMONamedOutput(levelName);
    MultipleOutputs.addNamedOutput(conf, namedOutput, MapFileOutputFormat.class,
Text.class, ArrayWritable.class);
}

//finally, setup the spatial operation

SpatialOperationConfig spatialOpConf = new SpatialOperationConfig();
spatialOpConf.setOperation(SpatialOperation.IsInside);
spatialOpConf.setSrid(8307);
spatialOpConf.setTolerance(0.5);
spatialOpConf.setGeodetic(true);
spatialOpConf.store(conf);

//run job

JobClient.runJob(conf);
```

Supposing the output value should be an array of record ids instead of an array of `RecordInfo` instances, it would be enough to perform a couple of changes in the previously defined reducer.

The line where `outValue` is declared, in the previous example, changes to:

```
private ArrayWritable outValue = new ArrayWritable(Text.class);
```

The loop where the input values are retrieved, in the previous example, is changed. Therefore, the record ids are got instead of the whole records:

```
while(values.hasNext())
{
  records.add( new Text(values.next().getId()) );
}
```

While only the record id is needed the mapper emits the whole `RecordInfo` instance. Therefore, a better approach is to change the mappers output value. The mappers output value can be changed by extending `AbstractSpatialJoinMapper`. In the following example, the mapper emits only the record ids instead of the whole `RecorInfo` instance every time a record matches some of the hierarchy entries:

```
public class IdSpatialHierarchicalMapper extends AbstractSpatialHierarchicalMapper<
Text >
{
       Text outValue = new Text();

       @Override
       protected Text getOutValue(RecordInfo matchingRecordInfo)
       {

         //the out value is the record's id

         outValue.set(matchingRecordInfo.getId());
         return outValue;
       }
}
```

- Changing the Hierarchy Level Range
- Controlling the Search Hierarchy
- Using MVSuggest to Classify the Data

2.10.5.1 Changing the Hierarchy Level Range

By default, all the hierarchy levels defined in the `HierarchyInfo` implementation are loaded when performing the hierarchy search. The range of hierarchy levels loaded is from level 1 (parent level) to the level returned by `HierarchyInfo.getNumberOfLevels()` method. The following example shows how to setup a job to only load the levels 2 and 3.

```
conf.setInt( ConfigParams.HIERARCHY_LOAD_MIN_LEVEL, 2);
conf.setInt( ConfigParams.HIERARCHY_LOAD_MAX_LEVEL, 3);
```

Note:

These parameters are useful when only a subset of the hierarchy levels is required and when you do not want to modify the `HierarchyInfo` implementation.

2.10.5.2 Controlling the Search Hierarchy

The search is always performed only at the bottom hierarchy level (the higher level number). If a user record matches some hierarchy entry at this level, then the match is propagated to the parent entry in upper levels. For example, if a user record matches Los Angeles, then it also matches California, USA, and North America. If there are no matches for a user record at the bottom level, then the search does not continue into the upper levels.

This behavior can be modified by setting the configuration parameter `ConfigParams.HIERARCHY_SEARCH_MULTIPLE_LEVELS` to `true`. Therefore, if a search at the bottom hierarchy level resulted in some unmatched user records, then search continues into the upper levels until the top hierarchy level is reached or there are no more user records to join. This behavior can be used when the geometries of parent levels do not perfectly enclose the geometries of their child entries

2.10.5.3 Using MVSuggest to Classify the Data

`MVSuggest` can be used instead of the spatial index to classify data. For this case, an implementation of `LocalizableRecordInfoProvider` must be known and sent to MVSuggest to perform the search. See the information about `LocalizableRecordInfoProvider`.

In the following example, the program option is changed from spatial to MVS. The input is the path to the user data instead of the spatial index. The `InputFormat` used to read the user record and an implementation of `LocalizableRecordInfoProvider` are specified. The `MVSuggest` service configuration is set. Notice that there is no spatial operation configuration needed in this case.

```
Categorization<LongWritable, Text> hierCount = new Categorization<LongWritable, Text>();

// the input path is the user's data

hierCount.setInput("/user/data/");

// set the job's output

hierCount.setOutput("/user/mvs_hierarchy_count");

// set HierarchyInfo implementation which describes the world
// administrative boundaries hierarchy

hierCount.setHierarchyInfoClass(WorldDynaAdminHierarchyInfo.class);

// specify the paths of the hierarchy data

Path[] hierarchyDataPaths = { new Path("file:///home/user/catalogs/world_continents.json"),
        new Path("file:///home/user/catalogs/world_countries.json"),
        new Path("file:///home/user/catalogs/world_states_provinces.json") };
hierCount.setHierarchyDataPaths(hierarchyDataPaths);

// set the path where the index for the previous hierarchy data will be
// generated

hierCount.setHierarchyIndexPath(new Path("/user/hierarchy_data_index/"));

// No spatial operation configuration is needed, Instead, specify the
// InputFormat used to read the user's data and the
// LocalizableRecordInfoProvider class.

hierCount.setInputFormatClass(TextInputFormat.class);
hierCount.setRecordInfoProviderClass(MyLocalizableRecordInfoProvider.class);

// finally, set the MVSuggest configuration

LocalMVSConfig lmvsConf = new LocalMVSConfig();
lmvsConf.setServiceLocation("file:///home/user/mvs_dir/oraclemaps_pub");
lmvsConf.setRepositoryLocation(true);
hierCount.setMvsConfig(lmvsConf);

// add the previous setup to the job configuration
hierCount.configure(conf);
```

```
// run the job

JobClient.runJob(conf);
```

> **Note:**
>
> When using `MVSuggest`, the hierarchy data files must be the same as the layer template files used by `MVSuggest`. The hierarchy level names returned by the `HierarchyInfo.getLevelNames()` method are used as the matching layers by `MVSuggest`.

2.10.6 Generating Buffers

The API provides a mapper to generate a buffer around each record's geometry. The following code sample shows how to run a job to generate a buffer for each record geometry by using the `BufferMapper` class.

```
//configure input
conf.setInputFormat(FileSplitInputFormat.class);
FileSplitInputFormat.setInputPaths(conf, "/user/waterlines/");
FileSplitInputFormat.setRecordInfoProviderClass(conf, GeoJsonRecordInfoProvider.class);

//configure output
conf.setOutputFormat(SequenceFileOutputFormat.class);
SequenceFileOutputFormat.setOutputPath(conf, new Path("/user/data_buffer/"));

//set the BufferMapper as the job mapper
conf.setMapperClass(BufferMapper.class);
conf.setMapOutputKeyClass(Text.class);
conf.setMapOutputValueClass(RecordInfo.class);
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(RecordInfo.class);

//set the width of the buffers to be generated
conf.setDouble(ConfigParams.BUFFER_WIDTH, 0.2);

//run the job
JobClient.runJob(conf);
```

`BufferMapper` generates a buffer for each input record containing a geometry. The output key and values are the record id and a `RecordInfo` instance containing the generated buffer. The resulting file is a Hadoop `MapFile` containing the mapper output key and values. If necessary, the output format can be modified by implementing a reducer that takes the mapper's output keys and values, and outputs keys and values of a different type.

`BufferMapper` accepts the following parameters:

Parameter	ConfigParam constant	Type	Description
oracle.spatial.buffer.width	BUFFER_WIDTH	double	The buffer width

Parameter	ConfigParam constant	Type	Description
oracle.spatial.buffer.sma	BUFFER_SMA	double	The semi major axis for the datum used in the coordinate system of the input
oracle.spatial.buffer.iFlat	BUFFER_IFLAT	double	The flattening value
oracle.spatial.buffer.arcT	BUFFER_ARCT	double	The arc tolerance used for geodetic densification

2.10.7 Spatial Binning

The Vector API provides the class `oracle.spatial.hadoop.vector.mapred.job.Binning` to perform spatial binning over a spatial data set. The `Binning` class is a MapReduce job driver that takes an input data set (which can be spatially indexed or not), assigns each record to a bin, and generates a file containing all the bins (which contain one or more records and optionally aggregated values).

A binning job can be configured as follows:

1. Specify the data set to be binned and the way it will be read and interpreted (`InputFormat` and `RecordInfoProvider`), or, specify the name of an existing spatial index.
2. Set the output path.
3. Set the grid MBR, that is, the rectangular area to be binned.
4. Set the shape of the bins: `RECTANGLE` or `HEXAGON`.
5. Specify the bin (cell) size. For rectangles, specify the width and height. For hexagon-shaped cells, specify the hexagon width. Each hexagon is always drawn with only one of its vertices as the base.
6. Optionally, pass a list of numeric field names to be aggregated per bin.

The resulting output is a text file where each record is a bin (cell) in JSON format and contains the following information:

- id: the bin id
- geom: the bin geometry; always a polygon that is a rectangle or a hexagon
- count: the number of points contained in the bin
- aggregated fields: zero or more aggregated fields

The following example configures and runs a binning job:

```
//create job driver
Binning<LongWritable, Text> binJob = new Binning<LongWritable, Text>();
//setup input
binJob.setInput("/user/hdfs/input/part*");
binJob.setInputFormatClass(GeoJsonInputFormat.class);
binJob.setRecordInfoProviderClass(GeoJsonRecordInfoProvider.class);
//set binning output
binJob.setOutput("/user/hdfs/output/binning");
//create a binning configuration to produce rectangular cells
```

```
BinningConfig binConf = new BinningConfig();
binConf.setShape(BinShape.RECTANGLE);
//set the bin size
binConf.setCellHeight(0.2);
binConf.setCellWidth(0.2);
//specify the area to be binned
binConf.setGridMbr(new double[]{-50,10,50,40});
binJob.setBinConf(binConf);
//save configuration
binJob.configure(conf);
//run job
JobClient.runJob(conf);
```

2.10.8 Spatial Clustering

The job driver class `oracle.spatial.hadoop.mapred.KMeansClustering` can be used to find spatial clusters in a data set. This class uses a distributed version of the K-means algorithm.

Required parameters:

- Path to the input data set, the `InputFormat` class used to read the input data set and the `RecordInfoProvider` used to extract the spatial information from records.
- Path where the results will be stored.
- Number of clusters to be found.

Optional parameters:

- Maximum number of iterations before the algorithm finishes.
- Criterion function used to determine when the clusters converge. It is given as an implementation of `oracle.spatial.hadoop.vector.cluster.kmeans.CriterionFunction`. The Vector API contains the following criterion function implementations: `SquaredErrorCriterionFunction` and `EuclideanDistanceCriterionFunction`.
- An implementation of `oracle.spatial.hadoop.vector.cluster.kmeans.ClusterShapeGenerator`, which is used to generate a geometry for each cluster. The default implementation is `ConvexHullClusterShapeGenerator` and generates a convex hull for each cluster. If no cluster geometry is needed, the `DummyClusterShapeGenerator` class can be used.
- The initial k cluster points as a sequence of x,y ordinates. For example: x1,y1,x2,y2,…xk,yk

The result is a file named `clusters.json`, which contains an array of clusters called features. Each cluster contains the following information:

- id: Cluster id
- memberCount: Number of elements in the cluster
- geom: Cluster geometry

The following example runs the `KMeansClustering` algorithm to find 5 clusters. By default, the `SquredErrorCriterionFunction` and `ConvexHullClusterShapeGenerator` are used , so you do not need yo set these classes explicitly. Also note that `runIterations()` is called to run the algorithm; internally, it launches one MapReduce per iteration. In this example, the number 20 is passed to `runIterations()` as the maximum number of iterations allowed.

```
//create the cluster job driver
KMeansClustering<LongWritable, Text> clusterJob = new
KMeansClustering<LongWritable, Text>();
//set input properties:
//input dataset path
clusterJob.setInput("/user/hdfs/input/part*");
//InputFormat class
clusterJob.setInputFormatClass(GeoJsonInputFormat.class);
//RecordInfoProvider implementation
clusterJob.setRecordInfoProviderClass(GeoJsonRecordInfoProvider.class);
//specify where the results will be saved
clusterJob.setOutput("/user/hdfs/output/clusters");
//5 cluster will be found
clusterJob.setK(5);
//run the algorithm
success = clusterJob.runIterations(20, conf);
```

2.10.9 Spatial Join

The spatial join feature allows detecting spatial interactions between records of two different large data sets.

The driver class `oracle.spatial.hadoop.vector.mapred.job.SpatialJoin` can be used to execute or configure a job to perform a spatial join between two data sets. The job driver takes the following inputs:

- Input data sets: Two input data sets are expected. Each input data set is represented using the class `oracle.spatial.hadoop.vector.InputDataSet`, which holds information about where to find and how to read a data set, such as path(s), spatial index, input format, and record info provider used to interpret records from the data set. It also accepts a spatial configuration for the data set.
- Spatial operation configuration: The spatial operation configuration defines the spatial interaction used to determine if two records are related to each other. It also defines the area to cover (MBR), that is, only records within or intersecting the MBR will be considered in the search.
- Partitioning result file path: An optional parameter that points to a previously generated partitioning result for both data sets. Data need to be partitioned in order to distribute the work; if this parameter is not provided, a partitioning process will be executed over the input data sets. (See Spatial Partitioning for more information.)
- Output path: The path where the result file will be written.

The spatial join result is a text file where each line is a pair of records that meet the spatial interaction defined in the spatial operation configuration.

The following table shows the currently supported spatial interactions for the spatial join.

Spatial Operation	Extra Parameters	Type
AnyInteract	None	(NA)
IsInside	None	(N/A)
WithinDistance	oracle.spatial.hadoop.vector.util.SpatialOperationConfig.PARAM_WD_DISTANCE	double

For a `WithinDistance` operation, the distance parameter can be specified in the `SpatialOperationConfig`, as shown in the following example:

```
spatialOpConf.setOperation(SpatialOperation.WithinDistance);
spatialOpConf.addParam(SpatialOperationConfig.PARAM_WD_DISTANCE, 5.0);
```

The following example runs a Spatial Join job for two input data sets. The first data set, postal boundaries, is specified providing the name of its spatial index. For the second data set, tweets, the path to the file, input format, and record info provider are specified. The spatial interaction to detect is `IsInside`, so only tweets (points) that are inside a postal boundary (polygon) will appear in the result along with their containing postal boundary.

```
SpatialJoin spatialJoin = new SpatialJoin();
List<InputDataSet> inputDataSets = new ArrayList<InputDataSet>(2);

// set the spatial index of the 3-digit postal boundaries of the USA as the
first input data set
InputDataSet pbInputDataSet = new InputDataSet();
pbInputDataSet.setIndexName("usa_pcb3_index");

//no input format or record info provider are required here as a spatial
index is provided
inputDataSets.add(pbInputDataSet);

// set the tweets data set in GeoJSON format as the second data set
InputDataSet tweetsDataSet = new InputDataSet();
tweetsDataSet.setPaths(new Path[]{new Path("/user/example/tweets.json")});
tweetsDataSet.setInputFormatClass(GeoJsonInputFormat.class);
tweetsDataSet.setRecordInfoProviderClass(GeoJsonRecordInfoProvider.class);
inputDataSets.add(tweetsDataSet);

//set input data sets
spatialJoin.setInputDataSets(inputDataSets);

//spatial operation configuration
SpatialOperationConfig spatialOpConf = new SpatialOperationConfig();
spatialOpConf.setOperation(SpatialOperation.IsInside);
spatialOpConf.setBoundaries(new double[]{47.70, -124.28, 35.45, -95.12});
spatialOpConf.setSrid(8307);
spatialOpConf.setTolerance(0.5);
spatialOpConf.setGeodetic(true);
spatialJoin.setSpatialOperationConfig(spatialOpConf);

//set output path
spatialJoin.setOutput("/user/example/spatialjoin");

// prepare job
JobConf jobConf = new JobConf(getConf());

//preprocess will partition both data sets as no partitioning result file
was specified
spatialJoin.preprocess(jobConf);
```

```
spatialJoin.configure(jobConf);
JobClient.runJob(jobConf);
```

2.10.10 Spatial Partitioning

The partitioning feature is used to spatially partition one or more data sets.

Spatial partitioning consists of dividing the space into multiple rectangles, where each rectangle is intended to contain approximately the same number of points. Eventually these partitions can be used to distribute the work among reducers in other jobs, such as Spatial Join.

The spatial partitioning process is run or configured using the `oracle.spatial.hadoop.mapred.job.Partitioning` driver class, which accepts the following input parameters:

- Input data sets: One or more input data sets can be specified. Each input data set is represented using the class `oracle.spatial.hadoop.vector.InputDataSet`, which holds information about where to find and how to read a data set, such as path(s), spatial index, input format, and record info provider used to interpret records from the data set. It also accepts a spatial configuration for the data set.
- Sampling ratio: Only a fraction of the entire data set or sets is used to perform the partitioning. The sample ratio is the ratio of the sample size to the whole input data set size. If it is not specified, 10 percent (0.1) of the input data set size is used.
- Spatial configuration: Defines the spatial properties of the input data sets, such as the SRID. You must specify at least the dimensional boundaries.
- Output path: The path where the result file will be written.

The generated partitioning result file is in GeoJSON format and contains information for each generated partition, including the partition's geometry and the number of points contained (from the sample).

The following example partitions a tweets data set. Because the sampling ratio is not provided, 0.1 is used by default.

```
Partitioning partitioning = new Partitioning();
List<InputDataSet> inputDataSets = new ArrayList<InputDataSet>(1);

//define the input data set
InputDataSet dataSet = new InputDataSet();
dataSet.setPaths(new Path[]{new Path("/user/example/tweets.json")});
dataSet.setInputFormatClass(GeoJsonInputFormat.class);
dataSet.setRecordInfoProviderClass(GeoJsonRecordInfoProvider.class);
inputDataSets.add(dataSet);
partitioning.setInputDataSets(inputDataSets);

//spatial configuration
SpatialConfig spatialConf = new SpatialConfig();
spatialConf.setSrid(8307);
spatialConf.setBoundaries(new double[]{-180,-90,180,90});
partitioning.setSpatialConfig(spatialConf);

//set output
partitioning.setOutput("/user/example/tweets_partitions.json");
```

```
//run the partitioning process
partitioning.runFullPartitioningProcess(new JobConf());
```

2.10.11 RecordInfoProvider

A record read by a MapReduce job from HDFS is represented in memory as a key-value pair using a Java type (typically) Writable subclass, such as LongWritable, Text, ArrayWritable or some user-defined type. For example, records read using TextInputFormat are represented in memory as LongWritable, Text key-value pairs.

`RecordInfoProvider` is the component that interprets these memory record representations and returns the data needed by the Vector Analysis API. Thus, the API is not tied to any specific format and memory representations.

The `RecordInfoProvider` interface has the following methods:

- void setCurrentRecord(K key, V value)
- String getId()
- JGeometry getGeometry()
- boolean getExtraFields(Map<String, String> extraFields)

There is always a `RecordInfoProvider` instance per `InputFormat`. The method `setCurrentRecord()` is called passing the current key-value pair retrieved from the `RecordReader`. The `RecordInfoProvider` is then used to get the current record id, geometry, and extra fields. None of these fields are required fields. Only those records with a geometry participates in the spatial operations. The Id is useful for differentiating records in operations such as categorization. The extra fields can be used to store any record information that can be represented as text and which is desired to be quickly accessed without reading the original record, or for operations where `MVSuggest` is used.

Typically, the information returned by RecordInfoProvider is used to populate `RecordInfo` instances. A RecordInfo can be thought as a light version of a record and contains the information returned by the RecordInfoProvider plus information to locate the original record in a file.

- Sample RecordInfoProvider Implementation
- LocalizableRecordInfoProvider

2.10.11.1 Sample RecordInfoProvider Implementation

This sample implementation, called `JsonRecordInfoProvider`, takes text records in JSON format, which are read using `TextInputFormat`. A sample record is shown here:

```
{ "_id":"ABCD1234", "location":" 119.31669, -31.21615", "locationText":"Boston, Ma",
"date":"03-18-2015", "time":"18:05", "device-type":"cellphone", "device-name":"iPhone"}
```

When a JsonRecordInfoProvider is instantiated, a JSON ObjectMapper is created. The ObjectMapper is used to parse records values later when `setCurrentRecord()` is called. The record key is ignored. The record id, geometry, and one extra field are retrieved from the _id, location and locationText JSON properties. The geometry is represented as latitude-longitude pair and is used to create a point geometry using `JGeometry.createPoint()` method. The extra field (locationText) is added to the extraFields map, which serves as an out parameter and true is returned indicating that an extra field was added.

```
public class JsonRecordInfoProvider implements RecordInfoProvider<LongWritable,
Text> {
private Text value = null;
private ObjectMapper jsonMapper = null;
private JsonNode recordNode = null;

public JsonRecordInfoProvider(){

//json mapper used to parse all the records

jsonMapper = new ObjectMapper();

}

@Override
public void setCurrentRecord(LongWritable key, Text value) throws Exception {
        try{

           //parse the current value

           recordNode = jsonMapper.readTree(value.toString());
           }catch(Exception ex){
              recordNode = null;
              throw ex;
        }
}

@Override
public String getId() {
        String id = null;
        if(recordNode != null ){
                id = recordNode.get("_id").getTextValue();
        }
        return id;
}
@Override
public JGeometry getGeometry() {
        JGeometry geom = null;
        if(recordNode!= null){
                //location is represented as a lat,lon pair
                String location = recordNode.get("location").getTextValue();
                String[] locTokens = location.split(",");
                double lat = Double.parseDouble(locTokens[0]);
                double lon = Double.parseDouble(locTokens[1]);
                geom =  JGeometry.createPoint( new double[]{lon, lat},  2, 8307);
        }
        return geom;
}

@Override
public boolean getExtraFields(Map<String, String> extraFields) {
        boolean extraFieldsExist = false;
        if(recordNode != null) {
                extraFields.put("locationText",
recordNode.get("locationText").getTextValue() );
                extraFieldsExist = true;
        }
        return extraFieldsExist;
}
}
```

2.10.11.2 LocalizableRecordInfoProvider

This interface extends `RecordInfoProvider` and is used to know the extra fields that can be used as the search text, when `MVSuggest` is used.

The only method added by this interface is `getLocationServiceField()`, which returns the name of the extra field that will be sent to MVSuggest.

In addition, the following is an implementation based on "Sample RecordInfoProvider Implementation." The name returned in this example is `locationText`, which is the name of the extra field included in the parent class.

```
public class LocalizableJsonRecordInfoProvider extends JsonRecordInfoProvider
implements LocalizableRecordInfoProvider<LongWritable, Text> {

@Override
public String getLocationServiceField() {
    return  "locationText";
}
}
```

An alternative to `LocalizableRecordInfoProvider` is to set the configuration property `oracle.spatial.recordInfo.locationField` with the name of the search field, which value should be sent to `MVSuggest`. Example:
`configuration.set(LocatizableRecordInfoProvider.CONF_RECORD_INFO_LOCATION_FIELD, “locationField”)`

2.10.12 HierarchyInfo

The `HierarchyInfo` interface is used to describe a hierarchical dataset. This implementation of HierarchyInfo is expected to provide the number, names, and the entries of the hierarchy levels of the hierarchy it describes.

The root hierarchy level is always the hierarchy level 1. The entries in this level do not have parent entries and this level is referred as the top hierarchy level. Children hierarchy levels will have higher level values. For example: the levels for the hierarchy conformed by continents, countries, and states are 1, 2 and 3 respectively. Entries in the continent layer do not have a parent, but have children entries in the countries layer. Entries at the bottom level, the states layer, do not have children.

A HierarchyInfo implementation is provided out of the box with the Vector Analysis API. The `DynaAdminHierarchyInfo` implementation can be used to read and describe the known hierarchy layers in GeoJSON format. A DynaAdminHierarchyInfo can be instantiated and configured or can be subclassed. The hierarchy layers to be contained are specified by calling the `addLevel()` method, which takes the following parameters:

- The hierarchy level number
- The hierarchy level name, which must match the file name (without extension) of the GeoJSON file that contains the data. For example, the hierarchy level name for the file `world_continents.json` must be `world_continents`, for `world_countries.json` it is `world_countries`, and so on.
- Children join field: This is a JSON property that is used to join entries of the current level with child entries in the lower level. If a null is passed, then the entry id is used.

- Parent join field: This is a JSON property used to join entries of the current level with parent entries in the upper level. This value is not used for the top most level without an upper level to join. If the value is set null for any other level greater than 1, an `IsInside` spatial operation is performed to join parent and child entries. In this scenario, it is supposed that an upper level geometry entry can contain lower level entries.

For example, let us assume a hierarchy containing the following levels from the specified layers: 1- world_continents, 2 - world_countries and 3 - world_states_provinces. A sample entry from each layer would look like the following:

```
world_continents:
 {"type":"Feature","_id":"NA","geometry": {"type":"MultiPolygon", "coordinates":
[ x,y,x,y,x,y] }"properties":{"NAME":"NORTH AMERICA",
"CONTINENT_LONG_LABEL":"North America"},"label_box":
[-118.07998,32.21006,-86.58515,44.71352]}

world_countries: {"type":"Feature","_id":"iso_CAN","geometry":
{"type":"MultiPolygon","coordinates":[x,y,x,y,x,y]},"properties":
{"NAME":"CANADA","CONTINENT":"NA","ALT_REGION":"NA","COUNTRY
CODE":"CAN"},"label_box":[-124.28092,49.90408,-94.44878,66.89287]}

world_states_provinces:
{"type":"Feature","_id":"6093943","geometry": {"type":"Polygon", "coordinates":
[ x,y,x,y,x,y]},"properties":{"COUNTRY":"Canada", "ISO":"CAN",
"STATE_NAME":"Ontario"},"label_box":[-91.84903,49.39557,-82.32462,54.98426]}
```

A DynaAdminHierarchyInfo can be configured to create a hierarchy with the above layers in the following way:

```
DynaAdminHierarchyInfo dahi = new DynaAdminHierarchyInfo();

dahi.addLevel(1, "world_continents", null /*_id is used by default to join with
child entries*/, null /*not needed as there are not upper hierarchy levels*/);

dahi.addLevel(2, "world_countries", "properties.COUNTRY CODE"/*field used to
join with child entries*/, "properties.CONTINENT" /*the value "NA" will be used
to find Canada's parent which is North America and which _id field value is also
"NA" */);

dahi.addLevel(3, "world_states_provinces", null /*not needed as not child
entries are expected*/, "properties.ISO"/*field used to join with parent
entries. For Ontario, it is the same value than the field properties.COUNTRY
CODE specified for Canada*/);

//save the previous configuration to the job configuration

dahi.initialize(conf);
```

A similar configuration can be used to create hierarchies from different layers, such as countries, states and counties, or any other layers with a similar JSON format.

Alternatively, to avoid configuring a hierarchy every time a job is executed, the hierarchy configuration can be enclosed in a DynaAdminHierarchyInfo subclass as in the following example:

```
public class WorldDynaAdminHierarchyInfo extends DynaAdminHierarchyInfo \

{
       public WorldDynaAdminHierarchyInfo()
```

```
        {
                super();
                addLevel(1, "world_continents", null, null);
                addLevel(2, "world_countries", "properties.COUNTRY CODE",
"properties.CONTINENT");
                addLevel(3, "world_states_provinces", null, "properties.ISO");
        }

}
```

- Sample HierarchyInfo Implementation

2.10.12.1 Sample HierarchyInfo Implementation

The `HierarchyInfo` interface contains the following methods, which must be implemented to describe a hierarchy. The methods can be divided in to the following three categories:

- Methods to describe the hierarchy
- Methods to load data
- Methods to supply data

Additionally there is an `initialize()` method, which can be used to perform any initialization and to save and read data both to and from the job configuration

```
void initialize(JobConf conf);

//methods to describe the hierarchy

String getLevelName(int level);
int getLevelNumber(String levelName);
int getNumberOfLevels();

//methods to load data

void load(Path[] hierDataPaths, int fromLevel, JobConf conf) throws Exception;
void loadFromIndex(HierarchyDataIndexReader[] readers, int fromLevel, JobConf conf)
throws Exception;

//methods to supply data

Collection<String> getEntriesIds(int level);
JGeometry getEntryGeometry(int level, String entryId);
String getParentId(int childLevel, String childId);
```

The following is a sample HierarchyInfo implementation, which takes the previously mentioned world layers as the hierarchy levels. The first section contains the initialize method and the methods used to describe the hierarchy. In this case, the initialize method does nothing. The methods mentioned in the following example use the `hierarchyLevelNames` array to provide the hierarchy description. The instance variables `entriesGeoms` and `entriesParent` are arrays of `java.util.Map`, which contains the entries geometries and entries parents respectively. The entries ids are used as keys in both cases. Since the arrays indices are zero-based and the hierarchy levels are one-based, the array indices correlate to the hierarchy levels as *array index + 1 = hierarchy level*.

```
public class WorldHierarchyInfo implements HierarchyInfo
{

        private String[] hierarchyLevelNames = {"world_continents", "world_countries",
```

```
"world_states_provinces"};
       private Map<String, JGeometry>[] entriesGeoms = new Map[3];
       private Map<String, String>[] entriesParents = new Map[3];

       @Override
       public void initialize(JobConf conf)
      {

         //do nothing for this implementation
}

        @Override
        public int getNumberOfLevels()
        {
          return hierarchyLevelNames.length;
}

        @Override
        public String getLevelName(int level)
        {
           String levelName = null;
           if(level >=1 && level <= hierarchyLevelNames.length)
           {
             levelName = hierarchyLevelNames[ level - 1];
           }
         return levelName;
         }

        @Override
        public int getLevelNumber(String levelName)
        {
           for(int i=0; i< hierarchyLevelNames.length; i++ )
           {
             if(hierarchyLevelNames.equals( levelName) ) return i+1;
   }
   return -1;
}
```

The following example contains the methods that load the different hierarchy levels data. The `load()` method reads the data from the source files `world_continents.json`, `world_countries.json`, and `world_states_provinces.json`. For the sake of simplicity, the internally called `loadLevel()` method is not specified, but it is supposed to parse and read the JSON files.

The `loadFromIndex()` method only takes the information provided by the `HierarchyIndexReader` instances passed as parameters. The `load()` method is supposed to be executed only once and only if a hierarchy index has not been created, in a job. Once the data is loaded, it is automatically indexed and `loadFromIndex()` method is called every time the hierarchy data is loaded into the memory.

```
      @Override
      public void load(Path[] hierDataPaths, int fromLevel, JobConf conf) throws
Exception {
      int toLevel = fromLevel + hierDataPaths.length - 1;
      int levels = getNumberOfLevels();

      for(int i=0, level=fromLevel; i<hierDataPaths.length && level<=levels; i+
+, level++)
```

```
        {

          //load current level from the current path

          loadLevel(level, hierDataPaths[i]);
         }
     }

     @Override
     public void loadFromIndex(HierarchyDataIndexReader[] readers, int fromLevel, 
JobConf conf)
                  throws Exception
     {
      Text parentId = new Text();
      RecordInfoArrayWritable records = new RecordInfoArrayWritable();
      int levels = getNumberOfLevels();

      //iterate through each reader to load each level's entries

      for(int i=0, level=fromLevel; i<readers.length && level<=levels; i++, level++)
      {
        entriesGeoms[ level - 1 ] = new Hashtable<String, JGeometry>();
        entriesParents[ level - 1 ] = new Hashtable<String, String>();

        //each entry is a parent record id (key) and a list of entries as RecordInfo 
(value)

        while(readers[i].nextParentRecords(parentId, records))
        {
           String pId = null;

           //entries with no parent will have the parent id UNDEFINED_PARENT_ID. Such 
is the case of the first level entries

            if( ! UNDEFINED_PARENT_ID.equals( parentId.toString() ) )
            {
            pId = parentId.toString();
            }

          //add the current level's entries

            for(Object obj : records.get())
            {
               RecordInfo entry = (RecordInfo) obj;
               entriesGeoms[ level - 1 ].put(entry.getId(), entry.getGeometry());
               if(pId != null)
               {
               entriesParents[ level -1 ].put(entry.getId(), pId);
               }
            }//finishin loading current parent entries
         }//finish reading single hierarchy level index
      }//finish iterating index readers
}
```

Finally, the following code listing contains the methods used to provide information of individual entries in each hierarchy level. The information provided is the ids of all the entries contained in a hierarchy level, the geometry of each entry, and the parent of each entry.

```
@Override
public Collection<String> getEntriesIds(int level)
{
```

```
   Collection<String> ids = null;

   if(level >= 1 && level <= getNumberOfLevels() && entriesGeoms[ level - 1 ] !=
null)
   {

     //returns the ids of all the entries from the given level

     ids = entriesGeoms[ level - 1 ].keySet();
   }
   return ids;
}

@Override
public JGeometry getEntryGeometry(int level, String entryId)
{
   JGeometry geom = null;
   if(level >= 1 && level <= getNumberOfLevels() && entriesGeoms[ level - 1 ] !=
null)
   {

      //returns the geometry of the entry with the given id and level

      geom = entriesGeoms[ level - 1 ].get(entryId);
    }
    return geom;
}

@Override
public String getParentId(int childLevel, String childId)
{
   String parentId = null;
   if(childLevel >= 1 && childLevel <= getNumberOfLevels() &&
entriesGeoms[ childLevel - 1 ] != null)
   {

      //returns the parent id of the entry with the given id and level

      parentId = entriesParents[ childLevel - 1 ].get(childId);
   }
   return parentId;
   }
}//end of class
```

2.10.13 Using JGeometry in MapReduce Jobs

The Spatial Hadoop Vector Analysis only contains a small subset of the functionality provided by the Spatial Java API, which can also be used in the MapReduce jobs. This section provides some simple examples of how JGeometry can be used in Hadoop for spatial processing. The following example contains a simple mapper that performs the `IsInside` test between a dataset and a query geometry using the JGeometry class.

In this example, the query geometry ordinates, srid, geodetic value and tolerance used in the spatial operation are retrieved from the job configuration in the configure method. The query geometry, which is a polygon, is preprocessed to quickly perform the `IsInside` operation.

The map method is where the spatial operation is executed. Each input record value is tested against the query geometry and the id is returned, when the test succeeds.

```
public class IsInsideMapper extends MapReduceBase implements Mapper<LongWritable,
Text, NullWritable, Text>
{
       private JGeometry queryGeom = null;
       private int srid = 0;
       private double tolerance = 0.0;
       private boolean geodetic = false;
       private Text outputValue = new Text();
       private double[] locationPoint = new double[2];


       @Override
       public void configure(JobConf conf)
       {
           super.configure(conf);
           srid = conf.getInt("srid", 8307);
           tolerance = conf.getDouble("tolerance", 0.0);
           geodetic = conf.getBoolean("geodetic", true);

    //The ordinates are represented as a string of comma separated double values

            String[] ordsStr = conf.get("ordinates").split(",");
            double[] ordinates = new double[ordsStr.length];
            for(int i=0; i<ordsStr.length; i++)
            {
              ordinates[i] = Double.parseDouble(ordsStr[i]);
             }

    //create the query geometry as two-dimensional polygon and the given srid

             queryGeom = JGeometry.createLinearPolygon(ordinates, 2, srid);

    //preprocess the query geometry to make the IsInside operation run faster

              try
              {
                queryGeom.preprocess(tolerance, geodetic, EnumSet.of(FastOp.ISINSIDE));
               }
               catch (Exception e)
               {
                 e.printStackTrace();
                }

          }

          @Override
          public void map(LongWritable key, Text value,
                      OutputCollector<NullWritable, Text> output, Reporter reporter)
                      throws IOException
          {

     //the input value is a comma separated values text with the following columns:
id, x-ordinate, y-ordinate

           String[] tokens = value.toString().split(",");

     //create a geometry representation of the record's location

            locationPoint[0] = Double.parseDouble(tokens[1]);//x ordinate
            locationPoint[1] = Double.parseDouble(tokens[2]);//y ordinate
            JGeometry location = JGeometry.createPoint(locationPoint, 2, srid);
```

```
      //perform spatial test

             try
             {
               if( location.isInside(queryGeom, tolerance, geodetic)){

                //emit the record's id

                outputValue.set( tokens[0] );
                output.collect(NullWritable.get(), outputValue);
               }
      }
              catch (Exception e)
              {
                 e.printStackTrace();
               }
}
}
```

A similar approach can be used to perform a spatial operation on the geometry itself. For example, by creating a buffer. The following example uses the same text value format and creates a buffer around each record location. The mapper output key and value are the record id and the generated buffer, which is represented as a `JGeometryWritable`. The `JGeometryWritable` is a Writable implementation contained in the Vector Analysis API that holds a JGeometry instance.

```
public class BufferMapper extends MapReduceBase implements Mapper<LongWritable,
Text, Text, JGeometryWritable>
{
       private int srid = 0;
       private double bufferWidth = 0.0;
       private Text outputKey = new Text();
       private JGeometryWritable outputValue = new JGeometryWritable();
       private double[] locationPoint = new double[2];

       @Override
       public void configure(JobConf conf)
       {
              super.configure(conf);
              srid = conf.getInt("srid", 8307);

              //get the buffer width

              bufferWidth = conf.getDouble("bufferWidth", 0.0);
        }

        @Override
        public void map(LongWritable key, Text value,
               OutputCollector<Text, JGeometryWritable> output, Reporter
reporter)
               throws IOException
        {

               //the input value is a comma separated record with the following
columns: id, longitude, latitude

               String[] tokens = value.toString().split(",");

               //create a geometry representation of the record's location
```

```
                locationPoint[0] = Double.parseDouble(tokens[1]);
                locationPoint[1] = Double.parseDouble(tokens[2]);
                JGeometry location = JGeometry.createPoint(locationPoint, 2, srid);

                try
                {

                   //create the location's buffer

                   JGeometry buffer = location.buffer(bufferWidth);

                   //emit the record's id and the generated buffer

                   outputKey.set( tokens[0] );
                   outputValue.setGeometry( buffer );
                   output.collect(outputKey, outputValue);
                 }

                 catch (Exception e)
                 {
                    e.printStackTrace();
                  }
         }
}
```

2.10.14 Support for Different Data Sources

In addition to file-based data sources (that is, a file or a set of files from a local or a distributed file system), other types of data sources can be used as the input data for a Vector API job.

Data sources are referenced as input data sets in the Vector API. All the input data sets implement the interface `oracle.spatial.hadoop.vector.data.AbstractInputDataSet`. Input data set properties can be set directly for a Vector job using the methods `setInputFormatClass()`, `setRecordInfoProviderClass()`, and `setSpatialConfig()`. More information can be set, depending the type of input data set. For example, `setInput()` can specify the input string for a file data source, or `setIndexName()` can be used for a spatial index. The job determines the input data type source based on the properties that are set.

Input data set information can also be set directly for a Vector API job using the job's method `setInputDataSet()`. With this method, the input data source information is encapsulated, you have more control, and it is easier to identify the type of data source that is being used.

The Vector API provides the following implementations of `AsbtractInputDataSet`:

- `SimpleInputDataSet`: Contains the minimum information required by the Vector API for an input data set. Typically, this type of input data set should be used for non-file based input data sets, such as Apache Hbase, an Oracle database, or any other non-file-based data source.
- `FileInputDataSet`: Encapsulates file-based input data sets from local or distributed file systems. It provides properties for setting the input path as an array of Path instances or as a string that can be a regular expression for selecting paths.
- `SpatialIndexInputDataSet`: A subclass of `FileInputDataSet` optimized for working with spatial indexes generated by the Vector API. It is sufficient to specify the index name for this type of input data set.

- `NoSQLInputDataSet`: Specifies Oracle NoSQL data sources. It should be used in conjunction with Vector NoSQL API. If the NoSQL `KVInputFormat` or `TableInputFormat` classes need to be used, use `SimpleInputFormat` instead.
- `MultiInputDataSet`: Input data set that encapsulates two or more input data sets.

Multiple Input Data Sets

Most of the Hadoop jobs provided by the Vector API (except Categorization) are able to manage more than one input data set by using the class `oracle.spatial.hadoop.vector.data.MultiInputDataSet`.

To add more than one input data set to a job, follow these steps.

1. Create and configure two or more instances of `AbstractInputDataSet` subclasses.
2. Create an instance of `oracle.spatial.hadoop.vector.data.MultiInputDataSet`.
3. Add the input data sets created in step 1 to the `MultiInputDataSet` instance.
4. Set `MultiInputDataSet` instance as the job's input data set.

The following code snippet shows how to set multiple input data sets to a Vector API.

```
//file input data set
FileInputDataSet fileDataSet = new FileInputDataSet();
fileDataSet.setInputFormatClass(GeoJsonInputFormat.class);
fileDataSet.setRecordInfoProviderClass(GeoJsonRecordInfoProvider.class)
;
fileDataSet.setInputString("/user/myUser/geojson/*.json");

//spatial index input data set
SpatialIndexInputDataSet indexDataSet = new SpatialIndexInputDataSet();
indexDataSet.setIndexName("myIndex");

//create multi input data set
MultiInputDataSet multiDataSet = new MultiInputDataSet();

//add the previously defined input data sets
multiDataSet.addInputDataSet(fileDataSet);
multiDataSet.addInputDataSet(indexDataSet);

Binning binningJob = new Binning();
//set multiple input data sets to the job
binningJob.setInputDataSet(multiDataSet);
```

NoSQL Input Data Set

The Vector API provides classes to read data from Oracle NoSQL Database. The Vector NoSQL components let you group multiple key-value pairs into single records, which are passed to Hadoop mappers as `RecordInfo` instances. They also let you map NoSQL entries (key and value) to Hadoop records fields (`RecordInfo`'s `id`, `geometry`, and extra fields).

The NoSQL parameters are passed to a Vector job using the `NoSQLInputDataSet` class. You only need to fill and set a `NoSQLConfiguration` instance that contains the KV store, hosts, parent key, and additional information for the NoSQL data source.

InputFormat and RecordInfoProvider classes do not need to be set because the default ones are used.

The following example shows how to configure a job to use NoSQL as data source, using the Vector NoSQL classes.

```
//create NoSQL configuration
NoSQLConfiguration nsqlConf = new NoSQLConfiguration();
// set connection data
nsqlConf.setKvStoreName("mystore");
nsqlConf.setKvStoreHosts(new String[] { "myserver:5000" });
nsqlConf.setParentKey(Key.createKey("tweets"));
// set NoSQL entries to be included in the Hadoop records
// the entries with the following minor keys will be set as the
// RecordInfo's extra fields
nsqlConf.addTargetEntries(new String[] { "friendsCount", "followersCount" });
// add an entry processor to map the spatial entry to a RecordInfo's
// geometry
nsqlConf.addTargetEntry("geometry", NoSQLJGeometryEntryProcessor.class);
//create and set the NoSQL input data set
NoSQLInputDataSet nsqlDataSet = new NoSQLInputDataSet();
//set noSQL configuration
nsqlDataSet.setNoSQLConfig(nsqlConf);
//set spatial configuration
SpatialConfig spatialConf = new SpatialConfig();
spatialConf.setSrid(8307);
nsqlDataSet.setSpatialConfig(spatialConf);
```

Target entries refer to the NoSQL entries that will be part of the Hadoop records and are specified by the NoSQL minor keys. In the preceding example, the entries with the minor keys friendsCount and followersCount will be part of a Hadoop record. These NoSQL entries will be parsed as text values and assigned to the Hadoop RecordInfo as the extra fields called friendsCount and followersCount. By default, the major key is used as record id. The entries that contain “geometry” as minor key are used to set the RecordInfo’s geometry field.

In the preceding example, the value type of the geometry NoSQL entries is JGeometry, so it is necessary to specify a class to parse the value and assign it to the RecordInfo’s geometry field. This requires setting an implementation of the NoSQLEntryProcessor interface. In this case, the NoSQLJGeometryEntryProcessor class is used, and it reads the value from the NoSQL entry and sets that value to the current RecordInfo’s geometry field. You can provide your own implementation of NoSQLEntryProcessor for parsing specific entry formats.

By default, NoSQL entries sharing the same major key are grouped into the same Hadoop record. This behavior can be changed by implementing the interface oracle.spatial.hadoop.nosql.NoSQLGrouper and setting the NoSQLConfiguration property entryGrouperClass with the new grouper class.

The Oracle NoSQL library kvstore.jar is required when running Vector API jobs that use NoSQL as the input data source.

Other Non-File-Based Data Sources

Other non-file-based data sources can be used with the Vector API, such as NoSQL (using the Oracle NoSQL classes) and Apache HBase. Although the Vector API does not provide specific classes to manage every type of data source, you can associate the specific data source with the job configuration and specify the following information to the Vector job:

- `InputFormat`: The `InputFormat` implementation used to read data from the data source.
- `RecordInfoProvider`: An implementation of `RecordInfoProvider` to extract required information such as `id`, spatial information, and extra fields from the key-value pairs returned by the current InputFormat.
- Spatial configuration: Describes the spatial properties of the input data, such as the SRID and the dimension boundaries.

The following example shows how to use Apache HBase data in a Vector job.

```
//create job
Job job = Job.getInstance(getConf());
job.setJobName(getClass().getName());
job.setJarByClass(getClass());

//Setup hbase parameters
Scan scan = new Scan();
scan.setCaching(500);
scan.setCacheBlocks(false);
scan.addColumn(Bytes.toBytes("location_data"),
Bytes.toBytes("geometry"));
scan.addColumn(Bytes.toBytes("other_data"),
Bytes.toBytes("followers_count"));
scan.addColumn(Bytes.toBytes("other_data"), Bytes.toBytes("user_id"));

//initialize job configuration with hbase parameters
TableMapReduceUtil.initTableMapperJob(
        "tweets_table",
        scan,
        null,
        null,
        null,
        job);
//create binning job
Binning<ImmutableBytesWritable, Result> binningJob = new
Binning<ImmutableBytesWritable, Result>();
//setup the input data set
SimpleInputDataSet inputDataSet = new SimpleInputDataSet();
//use HBase's TableInputFormat
inputDataSet.setInputFormatClass(TableInputFormat.class);
//Set a RecordInfoProvider which can extract information from HBase
TableInputFormat's returned key and values
inputDataSet.setRecordInfoProviderClass(HBaseRecordInfoProvider.class);
//set spatial configuration
SpatialConfig spatialConf = new SpatialConfig();
spatialConf.setSrid(8307);
inputDataSet.setSpatialConfig(spatialConf);
binningJob.setInputDataSet(inputDataSet);

//job output
```

```
binningJob.setOutput("hbase_example_output");

//binning configuration
BinningConfig binConf = new BinningConfig();
binConf.setGridMbr(new double[]{-180, -90, 180, 90});
binConf.setCellHeight(5);
binConf.setCellWidth(5);
binningJob.setBinConf(binConf);

//configure the job
binningJob.configure(job);

//run
boolean success = job.waitForCompletion(true);
```

The `RecordInfoProvider` class set in the preceding example is a custom implementation called `HBaseRecordInfoProvider`, the definition of which is as follows.

```
public class HBaseRecordInfoProvider implements
RecordInfoProvider<ImmutableBytesWritable, Result>, Configurable{

    private Result value = null;
    private Configuration conf = null;
    private int srid = 0;

    @Override
    public void setCurrentRecord(ImmutableBytesWritable key, Result value) throws
Exception {
        this.value = value;
    }

    @Override
    public String getId() {
        byte[] idb = value.getValue(Bytes.toBytes("other_data"),
Bytes.toBytes("user_id"));
        String id = idb != null ? Bytes.toString(idb) : null;
        return id;
    }

    @Override
    public JGeometry getGeometry() {
        byte[] geomb = value.getValue(Bytes.toBytes("location_data"),
Bytes.toBytes("geometry"));
        String geomStr = geomb!=null ? Bytes.toString(geomb) : null;
        JGeometry geom = null;
        if(geomStr != null){
            String[] pointsStr = geomStr.split(",");
            geom = JGeometry.createPoint(new double[]{Double.valueOf(pointsStr[0]),
Double.valueOf(pointsStr[1])}, 2, srid);
        }
        return geom;
    }

    @Override
    public boolean getExtraFields(Map<String, String> extraFields) {
        byte[] fcb =  value.getValue(Bytes.toBytes("other_data"),
Bytes.toBytes("followers_count"));
        if(fcb!=null){
            extraFields.put("followers_count", Bytes.toString(fcb));
```

```
        }
        return fcb!=null;
    }

    @Override
    public Configuration getConf() {
        return conf;
    }

    @Override
    public void setConf(Configuration conf) {
        srid = conf.getInt(ConfigParams.SRID, 0);
    }

}
```

2.10.15 Job Registry

Every time a Vector API job is launched using the command line interface or the web console, a registry file is created for that job. A job registry file contains the following information about the job:

- Job name
- Job ID
- User that executed the job
- Start and finish time
- Parameters used to run the job
- Jobs launched by the first job (called *child jobs*). Child jobs contain the same fields as the parent job.

A job registry file preserves the parameters used to run the job, which can be used as an aid for running an identical job even when it was not initially run using the command line interface.

By default, job registry files are created under the HDFS path relative to the user folder `oracle_spatial/job_registry` (for example, `/user/hdfs/oracle_spatial/job_registry` for the hdfs user).

Job registry files can be removed directly using HDFS commands or using the following utility methods from class `oracle.spatial.hadoop.commons.logging.registry.RegistryManager`:

- `public static int removeJobRegistry(long beforeDate, Configuration conf)`: Removes all the job registry files that were created before the specified time stamp from *the default* job registry folder.
- `public static int removeJobRegistry(Path jobRegDirPath, long beforeDate, Configuration conf)`: Removes all the job registry files that were created before the specified time stamp from *a specified* job registry folder.

2.10.16 Tuning Performance Data of Job Running Times Using the Vector Analysis API

The table lists some running times for jobs built using the Vector Analysis API. The jobs were executed using a 4-node cluster. The times may vary depending on the characteristics of the cluster. The test dataset contains over One billion records and the size is above 1 terabyte.

Table 2-4 Performance time for running jobs using Vector Analysis API

Job Type	Time taken (approximate value)
Spatial Indexing	2 hours
Spatial Filter with Spatial Index	1 hour
Spatial Filter without Spatial Index	3 hours
Hierarchy count with Spatial Index	5 minutes
Hierarchy count without Spatial Index	3 hours

The time taken for the jobs can be decreased by increasing the maximum split size using any of the following configuration parameters.

```
mapred.max.split.size
mapreduce.input.fileinputformat.split.maxsize
```

This results in more splits are being processed by each single mapper and improves the execution time. This is done by using the `SpatialFilterInputFormat` (spatial indexing) or `FileSplitInputFormat` (spatial hierarchical join, buffer). Also, the same results can be achieved by using the implementation of `CombineFileInputFormat` as internal `InputFormat`.

2.11 Oracle Big Data Spatial Vector Analysis for Spark

Oracle Big Data Spatial Vector Analysis for Apache Spark is a spatial vector analysis API for Java and Scala that provides spatially-enabled RDDs (Resilient Distributed Datasets) that support spatial transformations and actions, spatial partitioning, and indexing.

These components make use of the Spatial Java API to perform spatial analysis tasks. The supported features include the following.

- Spatial RDD (Resilient Distributed Dataset)
- Spatial Transformations
- Spatial Actions (MBR and NearestNeighbors)
- Spatially Indexing a Spatial RDD
- Spatial DStream Transformations
- Support for Common Spatial Formats
- Spatial Spark SQL API
- Rendering Spatial Indexes on Maps
- JDBC Data Sources for Spatial RDDs

2.11.1 Spatial RDD (Resilient Distributed Dataset)

A spatial RDD (Resilient Distributed Dataset) is a Spark RDD that allows you to perform spatial transformations and actions.

The current spatial RDD implementation is the class `oracle.spatial.spark.vector.rdd.SpatialJavaRDD` for Java and `oracle.spatial.spark.vector.scala.rdd.SpatialRDD` for Scala. A spatial RDD implementation can be created from an existing instance of RDD or JavaRDD, as shown in the following examples:

Java:

```
//create a regular RDD
JavaRDD<String> rdd = sc.textFile("someFile.txt");
//create a SparkRecordInfoProvider to extract spatial information from
the source RDD's records
SparkRecordInfoProvider<String> recordInfoProvider = new
MySparkRecordInfoProvider();
//create a spatial RDD
SpatialJavaRDD<String> spatialRDD = SpatialJavaRDD.fromJavaRDD(rdd,
recordInfoProvider, String.class));
```

Scala:

```
//create a regular RDD
val rdd: RDD[String] = sc.textFile("someFile.txt")
//create a SparkRecordInfoProvider to extract spatial information from
the source RDD's records
val recordInfoProvider: SparkRecordInfoProvider[String] = new
MySparkRecordInfoProvider()
//create a spatial RDD
val spatialRDD: SpatialRDD[String] = SpatialRDD(rdd,
recordInfoProvider)
```

A spatial RDD takes an implementation of the interface `oracle.spatial.spark.vector.SparkRecordInfoProvider`, which is used for extracting spatial information from each RDD element.

A regular RDD can be transformed into a spatial RDD of the same generic type, that is, if the source RDD contains records of type String. The spatial RDD will also contain String records.

You can also create a Spatial RDD with records of type `oracle.spatial.spark.vector.SparkRecordInfo`. A `SparkRecordInfo` is an abstraction of a record from the source RDD; it holds the source record's spatial information and may contain a subset of the source record's data.

The following examples show how to create an RDD of `SparkRecordInfo` records.

Java:

```
//create a regular RDD
JavaRDD<String> rdd = sc.textFile("someFile.txt");
```

```
//create a SparkRecordInfoProvider to extract spatial information from the
source RDD’s records
SparkRecordInfoProvider<String> recordInfoProvider = new
MySparkRecordInfoProvider();
//create a spatial RDD
SpatialJavaRDD<SparkRecordInfo> spatialRDD = SpatialJavaRDD.fromJavaRDD(rdd,
recordInfoProvider));
```

Scala:

```
//create a regular RDD
val rdd: RDD[String] = sc.textFile("someFile.txt")
//create a SparkRecordInfoProvider to extract spatial information from the
source RDD’s records
val recordInfoProvider: SparkRecordInfoProvider[String] = new
MySparkRecordInfoProvider()
//create a spatial RDD
val spatialRDD: SpatialRDD[SparkRecordInfo] = SpatialRDD.fromRDD(rdd,
recordInfoProvider))
```

A spatial RDD of `SparkRecordInfo` records has the advantage that spatial information does not need to be extracted from each record every time it is needed for a spatial operation.

You can accelerate spatial searches by spatially indexing a spatial RDD. Spatial indexing is described in section Spatially Indexing a Spatial RDD.

The spatial RDD provides the following spatial transformations and actions, which are described in the sections Spatial Transformations and Spatial Actions (MBR and NearestNeighbors).

Spatial transformations:

- filter
- flatMap
- join (available when creating a spatial index)

Spatial Actions:

- MBR
- nearestNeighbors

Spatial Pair RDD

A pair version of the Java class`SpatialJavaRDD` is provided and is implemented as the class `oracle.spatial.spark.vector.rdd.SpatialJavaPairRDD`. A spatial pair RDD is created from an existing pair RDD and contains the same spatial transformations and actions as the single spatial RDD. A `SparkRecordInfoProvider` used for a spatial pair RDD should receive records of type `scala.Tuple2<K,V>`, where `K` and `V` correspond to the pair RDD key and value types, respectively.

Example 2-1 SparkRecordInfoProvider to Read Information from a CSV File

The following example shows how to implement a simple `SparkRecordInfoProvider` to read information from a CSV file.

```
public class CSVRecordInfoProvider implements
SparkRecordInfoProvider<String>{
    private int srid = 8307;

    //receives an RDD record and fills the given recordInfo
    public boolean getRecordInfo(String record, SparkRecordInfo
recordInfo) {
        try {
            String[] tokens = record.split(",");
            //expected records have the format: id,name,last_name,x,y
where x and y are optional
            //output recordInfo will contain the fields id, last name
and geometry
            recordInfo.addField("id", tokens[0]);
            recordInfo.addField("last_name", tokens[2]);
            if (tokens.length == 5) {

recordInfo.setGeometry(JGeometry.createPoint(tokens[3], tokens[4], 2,
srid));
            }
        } catch (Exception ex) {
            //return false when there is an error extracting data from
the input value
            return false;
        }
        return true;
    }

    public void setSrid(int srid) {this.srid = srid;}
    public int getSrid() {return srid;}
}
```

In this example, the record's ID and last-name fields are extracted along with the spatial information to be set to the `SparkRecordInfo` instance used as an out parameter. Extracting additional information is only needed when the goal is to create a spatial RDD containing `SparkRecordInfo` elements and is necessary to preserve a subset of the original records information. Otherwise, it is only necessary to extract the spatial information.

The call to `SparkRecordInfoProvider.getRecordInfo()` should return `true` whenever the record should be included in a transformation or considered in a search. If `SparkRecordInfoProvider.getRecordInfo()` returns `false`, the record is ignored.

2.11.2 Spatial Transformations

The transformations described in the following subtopics are available for spatial RDD, spatial pair RDD, and the distributed spatial index unless stated otherwise (for example, a join transformation is only available for a distributed spatial index).

- Filter Transformation

2.11.2.1 Filter Transformation

A filter transformation is a spatial version of the regular RDD's filter() transformation. In addition to a user-provided filtering function, it takes an instance of `oracle.spatial.hadoop.vector.util.SpatialOperationConfig`, which is used to describe the spatial operation used to filter spatial records. A `SpatialOperationConfig` contains a query window which is the geometry used as reference and a spatial operation. The spatial operation is executed in the form: `(RDD record's geometry) (spatial operation) (query window)`. For example: `(RDD record) IsInside (queryWindow)`

Spatial operations available are `AnyInteract`, `IsInside`, `Contains`, and `WithinDistance`.

The following examples return an RDD containing only records that are inside the given query window and with not null ID.

Java:

```
SpatialOperationConfig soc = new SpatialOperationConfig();
soc.setOperation(SpatialOperation.IsInside);
soc.setQueryWindow(JGeometry.createLinearPolygon(new double[] { 2.0, 1.0,
2.0, 3.0, 6.0, 3.0, 6.0, 1.0, 2.0, 1.0 }, 2, srid));
SpatialJavaRDD<SparkRecordInfo> filteredSpatialRDD = spatialRDD.filter(
(record) -> {
return record.getField("id") != null;
}, soc);
```

Scala:

```
val soc = new SpatialOperationConfig()
soc.setOperation(SpatialOperation.IsInside)
soc.setQueryWindow(JGeometry.createLinearPolygon(Array(2.0, 1.0, 2.0, 3.0,
6.0, 3.0, 6.0, 1.0, 2.0, 1.0 ), 2, srid))
val filteredSpatialRDD: SpatialRDD[SparkRecordInfo] = spatialRDD.filter(
record => { record.getField("id") != null }, soc)
```

2.11.2.2 FlatMap Transformation

A FlatMap transformation is a spatial version of the regular RDD's `flatMap()` transformation. In addition to the user-provided function, it takes a SpatialOperationConfig to perform a spatial filtering. It works like the Filter Transformation, except that spatially filtered results are passed to the map function and flattened.

The following examples create an RDD that contains only elements that interact with the given query window and geometries that have been buffered.

Java:

```
SpatialOperationConfig soc = new SpatialOperationConfig();
soc.setOperation(SpatialOperation.AnyInteract);
soc.setQueryWindow(JGeometry.createLinearPolygon(new double[] { 2.0,
1.0, 2.0, 3.0, 6.0, 3.0, 6.0, 1.0, 2.0, 1.0 }, 2, srid));
JavaRDD<SparkRecordInfo> mappedRDD = spatialRDD.flatMap(
(record) -> {
    JGeometry buffer = record.getGeometry().buffer(2.5);
    record.setGeometry(buffer);
return Collections.singletonList(record);
}, soc);
```

Scala:

```
val soc = new SpatialOperationConfig()
soc.setOperation(SpatialOperation.AnyInteract)
soc.setQueryWindow(JGeometry.createLinearPolygon(Array( 2.0, 1.0, 2.0,
3.0, 6.0, 3.0, 6.0, 1.0, 2.0, 1.0 ), 2, srid))
val mappedRDD: RDD[SparkRecordInfo] = spatialRDD.flatMap(
record => {
    val buffer: JGeometry = record.getGeometry().buffer(2.5)
    record.setGeometry(buffer)
record
}, soc)
```

> **Note:**
>
> As of Spark 2, the Java class `org.apache.spark.api.java.function.FlatMapFunction` received by the `flatMap` transformation returns an instance of `java.util.Iterator` instead of `Iterable`, so the return line of the preceding `flatMap` transformation Java example changes for Spark 2 to: `return Collections.singletonList(record).iterator();`

2.11.2.3 Join Transformation

A join transformation joins two spatial RDDs based on a spatial relationship between their records. In order to perform this transformation, one of the two RDDs must be spatially indexed. (See Spatial Indexing for more information about indexing a spatial RDD.)

The result type of a spatial join transformation is defined by a user-provided lambda function that is called for each pair of joined records.

The following examples join all the records from both data sets that interact in any way.

Java:

```
DistributedSpatialIndex index =
DistributedSpatialIndex.createIndex(sparkContext, spatialRDD1, new
```

```
QuadTreeConfiguration());
SpatialJavaRDD<SparkRecordInfo> spatialRDD2 =
SpatialJavaRDD.fromJavaRDD(rdd2, new RegionsRecordInfoProvider(srid));
SatialOperationConfig soc = new SpatialOperationConfig();
soc.setOperation(SpatialOperation.AnyInteract);
JavaRDD<Tuple2<SparkRecordInfo, SparkRecordInfo> joinedRDD =
index.spatialJoin( spatialRDD2,
(recordRDD1, recordRDD2) -> {
return Collections.singletonList( new Tuple2<>(recordRDD1,
recordRDD2)).iterator());
}, soc);
```

Scala:

```
val index: DistributedSpatialIndex[SparkRecordInfo] =
DistributedSpatialIndex.createIndex(spatialRDD1, new QuadTreeConfiguration())
val spatialRDD2: SpatialRDD[SparkRecordInfo] = SpatialRDD.fromRDD(rdd2, new
RegionsRecordInfoProvider(srid))
val soc = new SpatialOperationConfig()
soc.setOperation(SpatialOperation.AnyInteract)
val joinedRDD: RDD[(SparkRecordInfo, SparkRecordInfo)] =
index.join( spatialRDD2,
(recordRDD1, recordRDD2) => {Seq((recordRDD1, recordRDD2))}, soc)
```

2.11.2.4 Controlling Spatial Evaluation

When executing a filtering transformation or nearest neighbors action, by default the spatial operation is executed before calling the user-defined filtering function; however, you can change this behavior. Executing a user-defined filtering function before the spatial operation can improve performance in scenarios where the spatial operation is costly in comparison to the user-defined filtering function.

To set the user-defined function to be executed *before* the spatial operation, set the following parameter to the `SpatialOperationConfig` passed to either a filter transformation or nearest neighbors action.

```
SpatialOperationConfig spatialOpConf = new
SpatialOperationConfig(SpatialOperation.AnyInteract, qryWindow, 0.05);
//set the spatial operation to be executed after the user-defined filtering
function
spatialOpConf.addParam(SpatialOperationConfig.PARAM_SPATIAL_EVAL_STAGE,
SpatialOperationConfig.VAL_SPATIAL_EVAL_STAGE_POST);
spatialRDD.filter((r)->{ return r.getFollowersCount()>1000;}, spatialOpConf);
```

The preceding example applies to both spatial RDDs and a distributed spatial index.

2.11.2.5 Spatially Enabled Transformations

Spatial operations can be performed in regular transformations by creating a `SpatialTransformationContext` before executing any transformation.

After the `SpatialTransformationContext` instance is in the transformation function, that instance can be used to get the record’s geometry and apply spatial operations, as shown in the following example, which transforms an RDD of String records into a pair RDD where the key and value corresponds to the source record ID and a buffered geometry.

Java:

```
SpatialJavaRDD<String> spatialRDD = SpatialJavaRDD.fromJavaRDD(rdd,
new CSVRecordInfoProvider(srid), String.class);
SpatialTransformationContext stCtx =
spatialRDD.createSpatialTransformationContext();
JavaPairRDD<String, JGeometry> bufferedRDD = spatialRDD.mapToPair(
(record) -> {
    SparkRecordInfo recordInfo = stCtx.getRecordInfo(record);
    String id = (String) recordInfo.getField("id")
    JGeometry geom. = recordInfo.getGeometry(record);
    JGeometry buffer = geom.buffer(0.5);
return new Tuple2(id, buffer);
});
```

Scala:

```
val spatialRDD: SpatialRDD[String]= SpatialRDD.fromRDD(rdd, new
CSVRecordInfoProvider(srid))
val stCtx: SpatialTransformationContext[String] =
spatialRDD.createSpatialTransformationContext()
val bufferedRDD: RDD[(String, JGeometry)] = spatialRDD.map(
record => {
    val recordInfo: SparkRecordInfo = stCtx.getRecordInfo(record)
    val id: String = recordInfo.getField("id").asInstanceOf[String]
    val geom: JGeometry = recordInfo.getGeometry(record)
    val buffer: JGeometry = geom.buffer(0.5)
(id, buffer)
})
```

When working on a per-partition basis, you should use a stateful version of `SpatialTransformationContext`, which avoids creating multiple instances of `SparkRecordInfo`. The following pattern can be followed when working on a per-partition basis:

```
val stCtx: SpatialTransformationContext[String] =
spatialRDD.createSpatialTransformationContext()
val bufferedRDD: RDD[(String, JGeometry)] = spatialRDD.mapPartitions(
(records) => {
    val sSTCtx = new StatefulSpatialTransformationContext(stCtx)
    records.map(record=>{
        val recordInfo: SparkRecordInfo = sSTCtx.getRecordInfo(record)
        val id: String = recordInfo.getField("id").asInstanceOf[String]
        val geom: JGeometry = recordInfo.getGeometry(record)
        val buffer: JGeometry = geom.buffer(0.5)
             (id, buffer)
    })
}, true)
```

2.11.3 Spatial Actions (MBR and NearestNeighbors)

Spatial RDDs,spatial pair RDDs, and the distributed spatial index provide the following spatial actions.

- **MBR**: Calculates the RDD's minimum bounding rectangle (MBR). The MBR is only calculated once and cached so the second time it is called, it will not be recalculated. The following examples show how to get the MBR from a spatial RDD. (This transformation is not available for `DistributedSpatialIndex`.)

  Java:

  ```
  doubl[] mbr = spatialRDD.getMBR();
  ```

  Scala:

  ```
  val mbr: Array[Double] = spatialRDD.getMBR()
  ```

- **NearestNeighbors**: Returns a list containing the K nearest elements from an RDD or distributed spatial index to a given geometry. Additionally, a user-defined filter lambda function can be passed, so that only the records that pass the filter will be candidates to be part of the K nearest neighbors list. The following examples show how to get the 5 records closest to the given point.

  Java:

  ```
  JGeometry qryWindow = JGeometry.createPoint(new double[] { 2.0, 1.0 }, 2,
  srid));
  SpatialOperationConfig soc = new
  SpatialOperationConfig(SpatialOperation.None, qryWindow, 0.05);
  List<SparkRecordInfo> nearestNeighbors = spatialRDD.nearestNeighbors(
  (record)->{
      return ((Integer)record.getField("followers_count"))>1000;
  }, 5, soc);
  ```

  Scala:

  ```
  val qryWindow: JGeometry = JGeometry.createPoint(Array(2.0, 1.0 ), 2,
  srid))
  val soc: SpatialOperationConfig = new
  SpatialOperationConfig(SpatialOperation.None, qryWindow, 0.05)
  val nearestNeighbors: Seq[SparkRecordInfo] = spatialRDD.nearestNeighbors(
  record=>{ record.getField("followers_count").asInstanceOf[Int]>1000 }, 5,
  soc);
  ```

2.11.4 Spatially Indexing a Spatial RDD

A spatial RDD can be spatially indexed to speed up spatial searches when performing spatial transformations.

A spatial index repartitions the spatial RDD so that each partition only contains records on some specific area. This allows partitions that do not contain results in a spatial search to be quickly discarded, making the search faster.

A spatial index is created through the Java abstract class `oracle.spatial.spark.vector.index.DistributedSpatialIndex` or its Scala equivalent `oracle.spatial.spark.vector.scala.index.DistributedSpatialIndex`, both of which use a specific implementation to create the actual spatial index. The following examples show how to create a spatial index using a QuadTree-based spatial index implementation.

Java:

```
DistributedSpatialIndex<String> index =
DistributedSpatialIndex.createIndex(sparkContext, spatialRDD1, new
QuadTreeConfiguration());
```

Scala:

```
val index: DistributedSpatialIndex[String] =
DistributedSpatialIndex.createIndex(spatialRDD1, new
QuadTreeConfiguration())(sparkContext)
```

The type of spatial index implementation is determined by the last parameter, which is a subtype of `oracle.spatial.spark.vector.index.SpatialPartitioningConfiguration`. Depending on the index implementation, the configuration parameter may accept different settings for performing partitioning and indexing. Currently, the only implementation of a spatial index is the class `oracle.spatial.spark.vector.index.quadtree.QuadTreeDistIndex`, and it receives a configuration of type `oracle.spatial.spark.vector.index.quadtree.QuadTreeConfiguration`.

The `DistributedSpatialIndex` class currently supports the filter, flatMap, join, and nearestNeighbors transformations, which are described in Spatial Transformations.

A spatial index can be persisted using the method `DistributedSpatialIndex.save()`, which takes an existing `SparkContext` and a path where the index will be stored. The path may be in a local or a distributed (HDFS) file system. Similarly, a persisted spatial index can be loaded by calling the method `DistributedSpatialIndex.load()`, which also takes an existing `SparkContext` and the path where the index is stored.

- Spatial Partitioning of a Spatial RDD
- Local Spatial Indexing of a Spatial RDD

2.11.4.1 Spatial Partitioning of a Spatial RDD

A spatial RDD can be partitioned through an implementation of the class `oracle.spatial.spark.vector.index.SpatialPartitioning`. The `SpatialPartitioning` class represents a spatial partitioning algorithm that transforms a spatial RDD into a spatially partitioned spatial pair RDD whose keys point to a spatial partition.

A SpatialPartitioning algorithm is used internally by a spatial index, or it can be used directly by creating a concrete class. Currently, there is a QuadTree-based implementation called oracle.spatial.spark.vector.index.quadtree.QuadTreePartitioning. The following example shows how to spatially partition a spatial RDD.

```
QuadTreePartitioning<T> partitioning = new
QuadTreePartitioning<>(sparkContext, spatialRDD, new
QuadTreeConfiguration());
SpatialJavaPairRDD<PartitionKey, T> partRDD =
partitioning.getPartitionedRDD();
```

2.11.4.2 Local Spatial Indexing of a Spatial RDD

A local spatial index can be created for each partition of a spatial RDD. Locally partitioning the content of each partition helps to improve spatial searches when working on a partition basis.

A local index can be created for each partition by setting the parameter `useLocalIndex` to `true` when creating a distributed spatial index. A spatially partitioned RDD can also be transformed so each partition is locally indexed by calling the utility method `oracle.spatial.spark.vector.index.local.LocalIndex.createLocallyIndexedRDD(SpatialJavaPairRDD<PartitionKey, T> rdd).`

2.11.5 Spatial DStream Transformations

A Spatial DStream is a Spark DStream that allows spatial transformations to be performed.

The current Spatial DStream implementations are the class `oracle.spatial.spark.vector.streaming.dstream.SpatialJavaDStream` and `oracle.spatial.spark.vector.streaming.dstream.SpatialJavaPairDStream` for Java, and `oracle.spatial.spark.vector.scala.streaming.dstream.SpatialDStream` for Scala. A spatial DStream can be created from an existing instance of DStream or JavaDStream, as shown in the following examples.

Java:

```
//create a regular DStram
JavaDStream<String> stream = ssc.socketTextStream(host, port);
//create a SparkRecordInfoProvider to extract spatial information from the
stream
SparkRecordInfoProvider<String> recordInfoProvider = new
TextRecordInfoProvider();
//create a Spatial DStream
SpatialJavaDStream<SparkRecordInfo> spatialStream =
SpatialJavaDStream.fromJavaDStream(stream, recordInfoProvider);
```

Scala:

```
//create a regular RDD
val stream: DStream[String] = ssc.socketTextStream(host, port)
//create a SparkRecordInfoProvider to extract spatial information from the
stream
val recordInfoProvider: SparkRecordInfoProvider[String] = new
TextRecordInfoProvider()
//create a Spatial DStream
```

```
val spatialStream: SpatialDStream[SparkRecordInfo] =
SpatialDStream.fromDStream(stream, recordInfoProvider)
```

A Spatial DStream takes an implementation of the interface `oracle.spatial.spark.vector.SparkRecordInfoProvider`, which is used for extracting spatial information from each element contained in the stream.

A regular DStream can be transformed into a Spatial DStream of the same generic type; that is, if the source DStream contains records of type String, the Spatial DStream will also contain String records. You can also create a Spatial DStream with records of type `oracle.spatial.spark.vector.SparkRecordInfo`. A SparkRecordInfo is an abstraction of a record from the source DStream; it holds the source record's spatial information and may contain a subset of the source record's data. The following examples show how to create a Spatial DStream of SparkRecordInfo records.

Java:

```
//create a regular DStram
JavaDStream<String> stream = ssc.socketTextStream(host, port);
//create a SparkRecordInfoProvider to extract spatial information from
the stream
SparkRecordInfoProvider<String> recordInfoProvider = new
TextRecordInfoProvider();
//create a Spatial DStream
SpatialJavaDStream<SparkRecordInfo> spatialStream =
SpatialJavaDStream.fromJavaDStream(stream, recordInfoProvider);
```

Scala:

```
//create a regular RDD
val stream: DStream[String] = ssc.socketTextStream(host, port)
//create a SparkRecordInfoProvider to extract spatial information from
the stream
val recordInfoProvider: SparkRecordInfoProvider[String] = new
TextRecordInfoProvider()
//create a Spatial DStream
val spatialStream: SpatialDStream[SparkRecordInfo] =
SpatialDStream.fromDStream(stream, recordInfoProvider)
```

A Spatial DStream of SparkRecordInfo records has the advantage that spatial information does not need to be extracted from each record every time it is needed for a spatial operation.

The Spatial DStream provides the following spatial transformations, which are available for both the Java and Scala Spatial DStream implementations.

- Filter Transformation (Spatial DStream)
- FlatMap Transformation (Spatial DStream)
- NearestNeighbors Transformation (Spatial DStream)
- Enrich Transformation (Spatial DStream)

2.11.5.1 Filter Transformation (Spatial DStream)

A filter transformation is a spatial version of the regular DStream's filter() transformation. In addition to a user-provided filtering function, it takes an instance of `oracle.spatial.hadoop.vector.util.SpatialOperationConfig`, which is used to describe the spatial operation used to filter spatial records. A `SpatialOperationConfig` contains a query window, which is the geometry used as reference, and a spatial operation. The spatial operation is executed in the form: `(DStream record's geometry) (spatial operation) (query window)`. For example: `(RDD record) IsInside (queryWindow)`

Spatial operations available are `AnyInteract`, `IsInside`, `Contains`, and `WithinDistance`.

The following examples return a Spatial DStream containing only records that are inside the given query window and with not null ID.

Java:

```
SpatialOperationConfig soc = new SpatialOperationConfig();
soc.setOperation(SpatialOperation.IsInside);
soc.setQueryWindow(JGeometry.createLinearPolygon(new double[] { 2.0, 1.0,
2.0, 3.0, 6.0, 3.0, 6.0, 1.0, 2.0, 1.0 }, 2, srid));
SpatialJavaDStream<SparkRecordInfo> filteredSpatialStream =
spatialStream.filter(
(record) -> {
return record.getField("id") != null;
}, soc);
```

Scala:

```
val soc = new SpatialOperationConfig()
soc.setOperation(SpatialOperation.IsInside)
soc.setQueryWindow(JGeometry.createLinearPolygon(Array(2.0, 1.0, 2.0, 3.0,
6.0, 3.0, 6.0, 1.0, 2.0, 1.0 ), 2, srid))
val filteredSpatialStream: SpatialDStream[SparkRecordInfo] =
spatialStream.filter(
record => { record.getField("id") != null }, soc)
```

2.11.5.2 FlatMap Transformation (Spatial DStream)

A FlatMap transformation is a spatial version of the regular RDD's `flatMap()` transformation. In addition to a user-provided function, it takes a SpatialOperationConfig to perform a spatial filtering. It works like the Filter Transformation (Spatial DStream), except that spatially filtered results are passed to the map function and flattened.

The following examples create an DStream that contains only elements that interact with the given query window and geometries that have been buffered.

Java:

```
SpatialOperationConfig soc = new SpatialOperationConfig();
soc.setOperation(SpatialOperation.AnyInteract);
soc.setQueryWindow(JGeometry.createLinearPolygon(new double[] { 2.0, 1.0,
2.0, 3.0, 6.0, 3.0, 6.0, 1.0, 2.0, 1.0 }, 2, srid));
```

```
JavaDStream<SparkRecordInfo> mappedStream = spatialStream.flatMap(
(record) -> {
    JGeometry buffer = record.getGeometry().buffer(2.5);
    record.setGeometry(buffer);
    return Collections.singletonList(record);
}, soc);
```

Scala:

```
val soc = new SpatialOperationConfig()
soc.setOperation(SpatialOperation.AnyInteract)
soc.setQueryWindow(JGeometry.createLinearPolygon(Array( 2.0, 1.0, 2.0,
3.0, 6.0, 3.0, 6.0, 1.0, 2.0, 1.0 ), 2, srid))
val mappedStream: DStream[SparkRecordInfo] = spatialStream.flatMap(
record => {
    val buffer: JGeometry = record.getGeometry().buffer(2.5)
    record.setGeometry(buffer)
    Seq(record)
}, soc)
```

2.11.5.3 NearestNeighbors Transformation (Spatial DStream)

A NearestNeighbors transformation returns a stream containing a single list of the K nearest elements from a Spatial DStream to a given geometry. The elements in the list are tuples of the form: `(distance, Spatial DStream’s record)`, and are sorted by distance in ascending order. Additionally, a user-defined filter lambda function can be passed. Only the records that pass the filter will be candidates to be part of the K nearest neighbors list.

The following example gets the five closest records to the given point that have a `followers_count` value greater than 1000.

Java:

```
JGeometry qryWindow = JGeometry.createPoint(new double[] { 2.0, 1.0 },
2, srid));
SpatialOperationConfig soc = new
SpatialOperationConfig(SpatialOperation.None, qryWindow, 0.05);
DStream<List<SparkRecordInfo>> nearestNeighborsStream =
spatialStream.nearestNeighbors(
(record)->{
    return ((Integer)record.getField(“followers_count”))>1000;
}, 5, soc);
```

Scala:

```
val qryWindow: JGeometry = JGeometry.createPoint(Array(2.0, 1.0 ), 2,
srid))
val soc: SpatialOperationConfig = new
SpatialOperationConfig(SpatialOperation.None, qryWindow, 0.05)
val nearestNeighborsStream: DStream[Seq[SparkRecordInfo]] =
spatialStream.nearestNeighbors(
```

```
record=>{ record.getField("followers_count").asInstanceOf[Int]>1000 }, 5,
soc)
```

2.11.5.4 Enrich Transformation (Spatial DStream)

An Enrich transformation uses a GeoEnricher Component to associate features from different data layers to spatial records from a Spatial DStream. The spatial records and the layer features are matched by their spatial relationship.

The transformation has the following input and output.

- Input:
  - Lambda function: A user-provided function that is called for each record from the stream (having associated features or not). The lambda function takes two parameters: a record from the stream and an iterator of SpatialFeature instances associated to the stream's record. The return type is an Iterator (Java) or a TransversableOnce (Scala) of a type specified by the user.
  - enricher: An instance of `GeoEnricher` used to perform the matching between geometries from the stream´s records and features from data layers.
- Output: A DStream whose size and elements' type is defined by the return type of the user-provided lambda function.

The following example executes the enrich transformation from an existing spatial stream. The transformation associates each stream's record to a list of features from data layers describing the world political boundaries for continents, countries, states/provinces, and cities. The order of the returned features is from the more specific layer (in this case, from cities), to the most general layer (continents).

In this example, the resulting stream will contain only those records from the spatial stream which were associated to a feature from the world political boundaries, so records with no features are discarded.

Java:

```
// Create a GeoEnricher
GeoEnricher enricher = new GeoJSONGeoEnricher(
    //path to the folder containing the world political boundaries
    spatialDataLayersDir,
    /*a predefined configuration which includes the world political
boundaries for: continents, countries, states/provinces, and cities.*/
    GeoJSONGeoEnricher.WORLD_POLITICAL_BOUNDARIES_CONF,
    //a Hadoop configuration
    hadoopConfiguration);

/*Perform the enrich transformation to create a stream a of pairs where each
pair is a record with its associated features*/
JavaPairDStream<SparkRecordInfo, List<SpatialFeature>> enrichedStream =
spatialStream.enrich(
    (record, features) -> {
        //return only records with features
        List<SpatialFeature> featureList = null;
        while (features.hasNext()) {
            if(featureList == null){
                featureList = new LinkedList<>();
```

```
                }
                featureList.add(features.next());
            }
            if(featureList != null){
                return Collections.singletonList(new Tuple2<>(record,
featureList)).iterator();
            }else{
                return Collections.emptyIterator();
            }
        }
        enricher
).mapToPair(tuple->{return tuple;});
```

Scala:

```
// Create a GeoEnricher
val enricher: GeoEnricher = new GeoJSONGeoEnricher(
    /*path to the folder containing files with the world political
boundaries*/
    spatialDataLayersDir,
    /*a predefined configuration which includes the world political
boundaries for: continents, countries, states/provinces, and cities.*/
    GEOJSONGeoEnricher.WORLD_POLITICAL_BOUNDARIES_CONF,
    // a Hadoop configuration
    hadoopConfiguration)

/*Perform the enrich transformation to create a stream a of pairs
where each pair is a record and its associated features*/
val enrichedStream: DStream[(SparkRecordInfo, Seq[SpatialFeature])] =
spatialStream.enrich(
    (record, features) => {
        //return only records with features
        Seq((record, features.toSeq)).filter(!_._2.isEmpty)
    }
    enricher
)
```

- GeoEnricher Component

2.11.5.4.1 GeoEnricher Component

The interface `oracle.spatial.spark.vector.geoenrichment.GeoEnricher` is used to perform enrichment of geometries. It provides a method that takes a geometry and returns an iterator of SpatialFeature instances that spatially interact with the given geometry.

The current implementation of GeoEnricher is the class `oracle.spatial.spark.vector.geoenrichment.GeoJSONGeoEnricher`, which uses a hierarchy of data layers defined as GeoJSON files. The SpatialFeature instances returned by the enrich method are features from each level in the hierarchy where the first element will be the feature at the last hierarchy level, followed by its parent, and so on.

A GeoJSONGeoEnricher can be created with the following parameters:

- Path to the folder containing the GeoJSON files
- An instance of GeoJSONSpatialLayerHierarchyConfiguration. This class defines a set of spatial layers to be loaded as a hierarchy.
- Optionally, a Hadoop configuration instance if the spatial data layers are stored in HDFS

A GeoJSONSpatialLayerHierarchyConfiguration receives an array of SpatialLayerDescriptor instances. Each SpatialLayerDescriptor describes a spatial layer defined in a GeoJSON file. The hierarchy level of each spatial layer corresponds to the index in the array, so the top parent layer will be the one with index 0, the layer with index 1 will be the child layer of the layer with index 0, and so on.

A SpatialLayerDescriptor has the following information:

- `filename`: The name of the GeoJSON file where the current layer is defined, for example: world_countries.json
- `name`: The name of the spatial layer. This is a text provided by the user used to identify the spatial layer. If no name is provided, the name defined at the `collectionName` field from the GeoJSON file is used.
- `parentRefField`: The name of a field contained at every feature of the current layer, used to associate a feature from the current layer to a feature from the parent layer. If this field is null, the parents will be associated by finding the features from the parent layer which spatially contain or interact with features from the child layers. Any property from the properties list of a GeoJSON feature can be used as a `parentRefField`.
- `parentRefFieldMapping`: The name of a field contained at every feature of the parent layer which is expected to have the same value than parentRefField in the child layer. That is, when looking for a parent feature is expected that `childFeature.parentRefField = parentFeature.parentRefFieldMapping` if `parentFeature` is the parent of `childFeature`. If this value is set to null, a field with the name defined by `parentRefField` will be considered as the parent layer.

The following example defines a GeoJSONSpatialLayerHierarchyConfiguration for the world political boundaries from continents to cities. The files can be found at the installation folder at `spatial/vector/examples/templates`.

```
GeoJSONSpatialLayerDescriptor[] descriptors = new
GeoJSONSpatialLayerDescriptor[4];//4 hierarchy levels
//First level: continents. Top parent layer
descriptors[0]=new GeoJSONSpatialLayerDescriptor("world_continents.json",
"continents");
/*Second level: countries. Features from this layer have a property called
Continent which can be used to find a parent at the continents layer. The
Continent property from the current layer points to the _id field at the
parent layer*/
descriptors[1]=new GeoJSONSpatialLayerDescriptor("world_countries.json",
"countries", "Continent", "_id");
/*Third level: states/provinces. The current layer's ISO property points to
the parent layer's Country Code property.*/
descriptors[2]=new
GeoJSONSpatialLayerDescriptor("world_states_provinces.json", "states/
provinces", "ISO", "Country Code");
/*Last level: cities. Parents will be associated by finding states/provinces
containing cities from this layer as no fields are provided.*/
descriptors[3]=new GeoJSONSpatialLayerDescriptor("world_cities.json",
```

```
"cities");
//Finally a configuration is created.
GeoJSONSpatialLayerHierarchyConfiguration worldPoliticalBoundsConf =
new GeoJSONSpatialLayerHierarchyConfiguration(
descriptors,
,8307//SRID
,0.05//tolerance
);
```

2.11.6 Support for Common Spatial Formats

The Spark Vector API provides utilities to easily read data from common spatial formats such as GeoJSON and ESRI ShapeFile.

The Java class `oracle.spatial.spark.vector.io.SpatialSources` and the Scala class `oracle.spatial.spark.vector.scala.io.SpatialSources` contain static methods to read data from GeoJSON and ShapeFile formats by specifying the data path, the data Spatial Reference System ID (SRID), and the list of non-spatial fields to be loaded.

The following examples show how to load data from a GeoJSON file. The records are automatically transformed to instances of `SparkRecordInfo`, which contain the spatial information plus the `_id` and `followers_count` fields. If all the fields need to be retrieved, null can be passed instead of the whole list of fields. Both GeoJSON and Shapefile read methods contain an overload that returns the original records as String and MapWritable representations, respectively.

Java:

```
//list of GeoJSON field names to be loaded for each feature
List<String> fieldNames = new ArrayList<String>();
fieldNames.add("_id");
fieldNames.add("followers_count");

//create a spatial RDD from a GeoJSON file
SpatialJavaRDD<SparkRecordInfo> spatialRDD =
SpatialSources.readGeoJSONRecordInfo(geoJSONInputPath, 8307,
fieldNames, sparkContext);
```

Scala:

```
//create a spatial RDD from a GeoJSON file
val spatialRDD =
SpatialSources.readGeoJSONRecordInfo(geoJSONInputPath, 8307,
Seq("_id","followers_count"))(sparkContext)
```

Or, using implicit classes:

```
//create a spatial RDD from a GeoJSON file
import
oracle.spatial.spark.vector.scala.io.SpatialSources.ImplicitSpatialSour
ces
```

```
val spatialRDD = sparkContext.readGeoJSONRecordInfo(geoJSONInputPath, 8307,
Seq("_id","followers_count"))
```

2.11.7 Spatial Spark SQL API

The Spatial Spark SQL API supports Spark SQL DataFrame objects containing spatial information in any format.

Oracle Big Data Spatial Vector Hive Analysis can be used with Spark SQL.

Example 2-2 Creating a Spatial DataFrame for Querying Tweets

The following example uses the Spark 1.x API to create a spatial DataFrame for querying tweets. Ithe data is loaded using a spatial RDD, then a DataFrame can be created using the function `SpatialJavaRDD.createSpatialDataFrame`.

```
//create HiveContext
HiveContext sqlContext = new HiveContext(sparkContext.sc());
//get the spatial DataFrame from the SpatialRDD
//the geometries are in GeoJSON format
DataFrame spatialDataFrame = spatialRDD.createSpatialDataFrame(sqlContext,
properties);
// Register the DataFrame as a table.
spatialDataFrame.registerTempTable("tweets");
//register UDFs
sqlContext.sql("create temporary function ST_Polygon as
'oracle.spatial.hadoop.vector.hive.ST_Polygon'");
sqlContext.sql("create temporary function ST_Point as
'oracle.spatial.hadoop.vector.hive.ST_Point'");
sqlContext.sql("create temporary function ST_Contains as
'oracle.spatial.hadoop.vector.hive.function.ST_Contains'");
// SQL can be run over RDDs that have been registered as tables.
StringBuffer query = new StringBuffer();
query.append("SELECT geometry, friends_count, location, followers_count FROM
tweets ");
query.append("WHERE ST_Contains( ");
query.append("    ST_Polygon('{\"type\": \"Polygon\",\"coordinates\":
[[[-106, 25], [-106, 30], [-104, 30], [-104, 25], [-106, 25]]]}', 8307) ");
query.append("    , ST_Point(geometry, 8307) ");
query.append("    , 0.05)");
query.append("    and followers_count > 50");
DataFrame results = sqlContext.sql(query.toString());
//Filter the tweets in a query window (somewhere in the north of Mexico)
//and with more than 50 followers.
//Note that since the geometries are in GeoJSON format it is possible to
create the ST_Point like
//ST_Point(geometry, 8307)
//instead of
//ST_Point(geometry,
'oracle.spatial.hadoop.vector.hive.json.GeoJsonHiveRecordInfoProvider')
List<String> filteredTweets = results.javaRDD().map(new Function<Row,
String>() {
  public String call(Row row) {
      StringBuffer sb = new StringBuffer();
      sb.append("Geometry: ");
```

```
        sb.append(row.getString(0));

        sb.append("\nFriends count: ");
        sb.append(row.getString(1));
        sb.append("\nLocation: ");
        sb.append(row.getString(2));
        sb.append("\nFollowers count: ");
        sb.append(row.getString(3));
        return sb.toString();
  }
}).collect();
//print the filtered tweets
filteredTweets.forEach(tweet -> System.out.println("Tweet: "+tweet));
```

- Spark 2 API Enhancements
- Spatial Analysis Spark SQL UDFs

2.11.7.1 Spark 2 API Enhancements

New Spark SQL capabilities have been added to the Spark 2 Vector API.

- Spatial DataSet/DataFrame
- Spatial UDFs
- Spatial Index
- Spatial Join
- Nearest Neighbors
- Performance Considerations with a Spatial Index Over Spark 2 SQL

Spatial DataSet/DataFrame

Spatial RDDs can be transformed to DataSets/DataFrames using the functions provided by the class `oracle.spatial.spark.vector.sql.SpatialJavaRDDConversions` (Java) and `oracle.spatial.spark.vector.scala.sql. SpatialRDDConversions` (Scala). The latter provides an implicit class in order to make it possible to call the transformation from the Spatial RDD instance. The following examples show how to transform a Spatial RDD to a DataFrame.

Java:

```
List<String> fields = Arrays.asList(new String[]
{("friends_count","location", "followers_count"});
Dataset<Row> spatialDataFrame =
SpatialJavaRDDConversions.toDataFrame(spatialRDD, fields,
sparkSession);
```

Scala:

```
//using implicit classes
import
oracle.spatial.spark.vector.scala.sql.SpatialRDDConversions.ImplicitSpa
```

```
tialRDDConversions
val spatialDataFrame =
spatialRDD.toDataFrame(Seq("friends_count","location", "followers_count"))
(sparkSession)
```

A spatial DataFrame can also be created from a GeoJSON file. The following examples show how a GeoJSON file can be loaded into a spatial DataFrame. The sample GeoJSON content is also shown.

GeoJSON:

```
{ "type":"FeatureCollection",
  "attr_names":["id","category"],
  "features":[
    {"type":"Feature","_id":"1","geometry":{"type":"Point","coordinates":
[-122.40849,37.7972]},"properties":{"category":"6"}}
    {"type":"Feature","_id":"2","geometry":{"type":"Point","coordinates":
[-122.40816,37.79769]},"properties":{"category":"4"}}
}
```

Java:

```
String[] nonSpatialCols = { "_id", "category" };
sparkSession.read().format(GeoJSONRelation.Format()).option("srid",
8307).schema(SchemaUtils.createStringFieldsSchema(Arrays.asList(nonSpatialCol
s))).load("/someptah/geojson.json").createOrReplaceTempView("spatialTable");
```

Scala:

```
val nonSpatialCols = Seq("_id", "category")
sparkSession.read.format(GeoJSONRelation.Format).option("srid",
8307).schema(SchemaUtils.createStringFieldsSchema(nonSpatialCols)).load("/
someptah/geojson.json").createOrReplaceTempView("spatialTable")
```

Spatial UDFs

The same set of Hive UDFs is available as Spark UDFs for the Spark 2 Vector API. For details, see Spatial Analysis Spark SQL UDFs.

The following line registers the spatial UDFs and should be executed before using any spatial UDF.

```
SpatialEnvironment.setup(sparkSession)
```

Spatial Index

An existing Spark Vector API's spatial index can be used from Spark 2 SQL to perform faster spatial queries.

The following examples show how to transform an instance of a spatial index to a DataFrame:

Java:

```
// Create a spatial RDD from a GeoJSON file
List<String> fieldNames = Arrays.asList(new String[] {"id",
"followers_count"});
SpatialJavaRDD<SparkRecordInfo> spatialRDD =
SpatialSources.readGeoJSONRecordInfo(path, srid, fieldNames,
sparkContext);

//Create a spatial index
DistributedSpatialIndex<SparkRecordInfo> index =
DistributedSpatialIndex.createIndex(sparkContext, spatialRDD, new
QuadTreeConfiguration());

//Specify the columns as StructFields. The geometry column is always
included by default
StructField[] fields = SchemaUtils.toStringStructFields(fieldNames);

//options can be null if there are no options to be passed
Map<String, Object> options = new HashMap<>();
//include the CRS to all the geometries to avoid using SDO_<TYPE>
wrappers in spatial UDF's
options.put(QuadTreeIndexRelation.OptIncludeCRS(), true);

//transform the existing spatial index to DataFrame and register as a
temporal table
QuadTreeIndexRelation.toDataFrame(index, SparkRecordInfo.class,
fields, options, sparkSession).createOrReplaceTempView("tweets_index");
```

Scala:

```
import
oracle.spatial.spark.vector.scala.io.SpatialSources.ImplicitSpatialSour
ces
import
oracle.spatial.spark.vector.scala.sql.index.quadtree.QuadTreeIndexRelat
ion._
import
oracle.spatial.spark.vector.scala.sql.SpatialRDDConversions.ImplicitSpa
tialRDDConversions

//List of field names to be loaded from the GeoJSON file
val fieldNames = Seq("id", "followers_count")

//create a spatial RDD
val spatialRDD = sparkContext.readGeoJSON(path, srid, fieldNames)

//spatially index the spatial RDD
val index = DistributedSpatialIndex.createIndex(spatialRDD, new
QuadTreeConfiguration())(implicitly, sparkContext)

//transform the existing spatial index to DataFrame and register as a
temporal table
//fieldNames are automatically transformed to an array of string
```

```
StructFields thanks to the //import of QuadTreeIndexRelation._
//toDataFrame can be called from the index thanks to the import of //
ImplicitSpatialRDDConversions
index.toDataFrame(fieldNames, Map(QuadTreeIndexRelation.OptIncludeCRS->true))
(sparkSession).createOrReplaceTempView("tweets_index")
```

It is also possible to load directly a persisted spatial index into a DataFrame, as the following examples show.

Java:

```
// list of GeoJSON field names to be loaded for each feature
List<String> fieldNames = Arrays.asList(new String[] { "id",
"followers_count"});

// Create the required schema for the index. In this case, the schema
// contains only fields of type StringType. A schema with other data
// types can be passed if needed.
StructType schema = SchemaUtils.createStringFieldsSchema(fieldNames);

// read an existing spatial index and register it as table
sparkSession.read().format(QuadTreeIndexRelation.Format()).schema(schema).loa
d(indexPath).createOrReplaceTempView("tweets_index");
```

Scala:

```
//List of field names from the spatial index to be included as columns.
val fieldNames = Seq("id", "followers_count")

//Create the required schema for the index.
//In this case, the schema contains only fields of type StringType.
//A schema with other data types can be passed if needed.
val schema = SchemaUtils.createStringFieldsSchema(fieldNames)

//read an existing spatial index and register it as a table
sparkSession.read.format(QuadTreeIndexRelation.Format).schema(schema).load(in
dexPath).createOrReplaceTempView("tweets_index")
```

After a spatial index is transformed to a DataFrame, it can be used as any other spatial DataFrame.

A spatial index can be also created from an existing spatial DataFrame, as shown in the following examples.

Java:

```
//load a spatial DataFrame from a GeoJSON file
Dataset<Row> spatialDataFrame =
sparkSession.read().format(GeoJSONRelation.Format()).option("srid",
srid).schema(mySchema).load("/user/x/customers.json");
//index the spatial DataFrame and register as table index
QuadTreeConfiguration qtConf = new QuadTreeConfiguration();
Map<String, Object> options = new HashMap<>();
options.put(QuadTreeIndexRelation.OptIncludeCRS(), true);
```

```
QuadTreeIndexRelation.indexSpatialDataFrame(spatialDataFrame, srid,
qtConf, options, sparkSession).createOrReplaceTempView("index");
```

Scala:

```
//load a spatial DataFrame from a GeoJSON file
val spatialDataFrame =
sparkSession.read.format(GeoJSONRelation.Format).option("srid",
srid).schema(mySchema).load("/user/x/customers.json")
//index the spatial DataFrame and register as table index
val qtConf = new QuadTreeConfiguration()
val options = Map(QuadTreeIndexRelation.OptIncludeCRS->true)
QuadTreeIndexRelation.indexSpatialDataFrame(spatialDataFrame, srid,
qtConf, options, sparkSession).createOrReplaceTempView("index")
```

A spatially indexed DataFrame can be persisted. The following examples show how to save a spatial index as a persistent table.

Java:

```
Dataset<Row> index =
QuadTreeIndexRelation.indexSpatialDataFrame(spatialDataFrame, srid,
qtConf, options, sparkSession);
index.write().format(QuadTreeIndexRelation.Format()).saveAsTable("index
");
```

Scala:

```
val index =
QuadTreeIndexRelation.indexSpatialDataFrame(spatialDataFrame, srid,
qtConf, options, sparkSession)
index.write.format(QuadTreeIndexRelation.Format).saveAsTable("index")
```

Spatial Join

Two spatial DataFrames can be joined using a spatial condition specified by a two-operands UDF, such as `st_anyinteract` or `st_contains`. Records from both sides that meet the specified spatial condition will be joined.

A spatial index can be used to perform an optimized spatial join, which generally should perform faster.

The following examples show how to perform spatial join between two DataFrames where one is a spatial index.

Java:

```
//load a spatial index
sparkSession.read().format(QuadTreeIndexRelation.Format()).schema(custS
chema).load(indexPath).createOrReplaceTempView("customers");
//load a spatial dataframe
sparkSession.read().format(GeoJSONRelation.Format()).option("srid",
8307).schema(storesSchema).load(storesPath).createOrReplaceTempView("st
ores");
```

```
//retrieve all the customers within 2 kilometers from each store
String query = "select * from stores s, customers c where
st_withindistance(st_point(s.geometry, 8307), st_point(c.geometry, 8307),
2000.0, 0.05)";
sparkSession.sql(query).show();
```

Scala:

```
//load a spatial index
sparkSession.read.format(QuadTreeIndexRelation.Format).schema(custSchema).loa
d(indexPath).createOrReplaceTempView("customers")
//load a spatial dataframe
sparkSession.read.format(GeoJSONRelation.Format).option("srid",
8307).schema(storesSchema).load(storesPath).createOrReplaceTempView("stores")
//retrieve all the customers within 2 kilometers from each store
val query = "select * from stores s, customers c where
st_withindistance(st_point(s.geometry, 8307), st_point(c.geometry, 8307),
2000.0, 0.05)"
sparkSession.sql(query).show()
```

Nearest Neighbors

The UDF ST_NN can be used to find the nearest neighbors for a specified location. This location can be a single geometry or the locations from an existing table.

The following examples show how to get the nearest neighbors for a specified geometry. Note that the DataFrame must be spatially indexed.

Java:

```
//read an existing spatial index
spark.read().format(QuadTreeIndexRelation.Format()).schema(schema).load(index
Path).createOrReplaceTempView("index");
//define a polygon geometry
String polygonJSON = "{\"type\": \"Polygon\", \"coordinates\": [[[-106, 25],
[-106, 30], [-104, 30], [-104, 25], [-106, 25]]]}";
//look for the 5 nearest neighbors for the previously defined polygon
String query = "SELECT location FROM index WHERE ST_NN( ST_POLYGON(geometry,
8307), ST_POLYGON('"+polygonJSON+"', 8307), 5, 0.05)";
sparkSession.sql(query).show();
```

Scala:

```
//read an existing spatial index
spark.read.format(QuadTreeIndexRelation.Format).schema(schema).load(indexPath
).createOrReplaceTempView("index")
//define a polygon geometry
val polygonJSON = """{"type": "Polygon", "coordinates": [[[-106, 25], [-106,
30], [-104, 30], [-104, 25], [-106, 25]]]}"""
//look for the 5 nearest neighbors for the previously defined polygon
val query = s"SELECT location FROM index WHERE ST_NN( ST_POLYGON(geometry,
8307), ST_POLYGON('$polygonJSON', 8307), 5, 0.05)"
sparkSession.sql(query).show()
```

The following examples show how to get the five nearest neighbors for all the rows from the table named STORES. When this version of the `nearestneighbors` operation is executed, one of the two DataFrames must be spatially indexed.

Java:

```
//load a spatial index
sparkSession.read().format(QuadTreeIndexRelation.Format()).schema(custS
chema).load(indexPath).createOrReplaceTempView("customers");
//load a spatial dataframe
sparkSession.read().format(GeoJSONRelation.Format()).option("srid",
8307).schema(storesSchema).load(storesPath).createOrReplaceTempView("st
ores");
//retrieve the five nearest customers for each store
String query = "select * from stores s, customers c where
st_nn(st_point(c.geometry, 8307), st_point(s.geometry, 8307), 5,
0.05)";
sparkSession.sql(query).show();
```

Scala:

```
//load a spatial index
sparkSession.read.format(QuadTreeIndexRelation.Format).schema(custSchem
a).load(indexPath).createOrReplaceTempView("customers")
//load a spatial dataframe
sparkSession.read.format(GeoJSONRelation.Format).option("srid",
8307).schema(storesSchema).load(storesPath).createOrReplaceTempView("st
ores")
//retrieve the five nearest customers for each store
String query = "select * from stores s, customers c where
st_nn(st_point(c.geometry, 8307), st_point(s.geometry, 8307), 5,
0.05)";
sparkSession.sql(query).show()
```

Performance Considerations with a Spatial Index Over Spark 2 SQL

A Spatial index performs faster when using only a spatial filter or a spatial filter and AND conditions in the WHERE clause. The following queries take full advantage of a spatial index as the spatial data is pre filtered before executing the SQL query:

```
SELECT * FROM tweets_index WHERE
ST_ANYINTERACT( ST_POLYGON('$polygonJSON',8307),
ST_POINT(geometry,8307), 0.05 )

SELECT * FROM tweets_index WHERE
ST_CONTAINS( ST_POLYGON('$polygonJSON',8307), ST_POINT(geometry,8307),
0.05 ) AND followers_count > 50

SELECT * FROM tweets_index WHERE ST_INSIDE( ST_POINT(geometry,8307),
ST_POLYGON('$polygonJSON',8307), 0.05 ) AND followers_count > 50 AND
id != null
```

Using OR conditions avoids the spatial data to be pre filtered, however, some spatial index optimizations are applied. The following query is an example of this case:

```
SELECT * FROM tweets_index WHERE
ST_CONTAINS( ST_POLYGON('$polygonJSON',8307), ST_POINT(geometry,8307),
0.05 ) OR followers_count > 50
```

When using more than one spatial filter in a WHERE clause, no spatial index optimizations are used and the query is performed as if there were no spatial index. For example:

```
SELECT * FROM tweets_index
    WHERE
     ST_ANYINTERACT( ST_POLYGON('$polygonJSON1',8307),
ST_POINT(geometry,8307), 0.05 )
     AND
     ST_CONTAINS( ST_POLYGON('$polygonJSON2',8307), ST_POINT(geometry,8307),
0.05 )
```

2.11.7.2 Spatial Analysis Spark SQL UDFs

Spatial analysis functions are available as Spark 2 SQL UDFs (user-defined functions).

The same set of Hive UDFs is available as Spark UDFs for the Spark 2 Vector API. In order to start using the Spatial UDFs, the following method from class `oracle.spatial.spark.vector.scala.sql.SpatialEnvironment` needs to be executed before calling any query containing a spatial UDF:

```
SpatialEnvironment.setup(sparkSession)
```

The input spatial data can be in GeoJSON, WKT, or WKB format. You can also use a spatial index for faster processing.

In the queries, spatial geometry type constructors, such as ST_Polygon or ST_Point, can be used to create a GeoJSON representation of the input geometry and to add a SRID (coordinate system) for the geometry. Such constructors must be used if a geometry is specified in the query, even if the geometry is already in GeoJSON format – ***unless*** you use the spatial index **option** to set the SRID in the geometry, in which case a spatial geometry type constructor is not needed; for example:

```
spark.read().format(QuadTreeIndexRelation.Format()).schema(schema)
         .option(QuadTreeIndexRelation.OptIncludeCRS(), true) //avoid using
Type Functions
         .load(indexPath).createOrReplaceTempView("tweets_index");
```

- Prerequisite Libraries for Spatial Analysis Spark SQL UDFs
- Using Spark SQL UDFs
- Using Spatial Indexes with Spark UDFs

Prerequisite Libraries for Spatial Analysis Spark SQL UDFs

The required libraries for Spatial Analysis Spark SQL UDFs are:

- `sdohadoop-vector.jar`
- `sdospark2-vector.jar`
- `sdoutl.jar`
- `sdoapi.jar`
- `ojdbc8.jar`

Using Spark SQL UDFs

Spatial analysis Spark SQL UDFs are a series of Spark SQL user-defined functions used to create geometries and perform spatial operations using one or two geometries in creating a Spark SQL query.

Hive and Spark Spatial SQL Functions provides reference information for the available spatial functions.

The following example returns the tweet records within a specific geographical polygon and where there are more than 50 followers. The general steps for the example are:

1. Set up the spatial SQL environment.
2. Create a spatial RDD from geographical input.
3. Create a DataSet from the SpatialRDD. A spatial DataSet contains a column called *geometry* whose values are in GeoJSON format.
4. Register the DataSet so it can be used within SQL statements as a table.
5. Create the query to filter the records.
6. Execute the filter.

Java Example:

```
import java.util.Arrays;
import java.util.List;

import org.apache.spark.api.java.JavaSparkContext;
import org.apache.spark.sql.Dataset;
import org.apache.spark.sql.Row;
import org.apache.spark.sql.SparkSession;

import oracle.spatial.spark.vector.SparkRecordInfo;
import oracle.spatial.spark.vector.io.SpatialSources;
import oracle.spatial.spark.vector.rdd.SpatialJavaRDD;
import oracle.spatial.spark.vector.scala.sql.SpatialEnvironment;
import oracle.spatial.spark.vector.sql.SpatialJavaRDDConversions;

public class SpatialQueryExample {
  public static void main(String[] args) {
    SparkSession spark = 
SparkSession.builder().appName("SpatialEx").getOrCreate();
         //Setup spatial SQL environment
         SpatialEnvironment.setup(spark);
         String geoJSONInput = args[0];
         //The coordinate system the spatial data is expected to be
         int srid = 8307;
         // list of GeoJSON field names to be loaded for each feature
```

```
          List<String> fieldNames = Arrays.asList(new String[] {
                      "id", "followers_count", "friends_count", "location"} );
          // Create a spatial RDD from a GeoJSON file
     SpatialJavaRDD<SparkRecordInfo> spatialRDD =
         SpatialSources.readGeoJSONRecordInfo(geoJSONInput, srid, fieldNames,
             JavaSparkContext.fromSparkContext(spark.sparkContext()));
          // Create a DataSet from the SpatialRDD.
          Dataset<Row> spatialDF = SpatialJavaRDDConversions.toDataFrame(
                 spatialRDD, fieldNames, spark);
          // Register the dataset so it can be used within SQL statements
          spatialDF.createOrReplaceTempView("sample_tweets");
     //polygon used to spatially filter data
          String qryWindow = "{\"type\": \"Polygon\",\"coordinates\":
[[[-106, 25], [-106,
                30], [-104, 30], [-104, 25], [-106, 25]]]}";

     // Filter the tweets within the query window (somewhere in the north of
Mexico)
          StringBuilder query =new StringBuilder()
                 .append(" SELECT geometry, friends_count, location,
followers_count")
                 .append("       FROM sample_tweets ")
                 .append("       WHERE ")
                 .append("
ST_CONTAINS(ST_POLYGON('").append(qryWindow).append("', 8307),
                                        ST_POINT(geometry, 8307), 0.05)")
                 .append("   AND followers_count > 50 ");
     //Execute the query
          spark.sql(query.toString()).show();
   }
}
```

Scala Example:

```
import org.apache.spark.sql.SparkSession
import oracle.spatial.spark.vector.sql.udf.function.FunctionExecutor
import
oracle.spatial.spark.vector.scala.io.SpatialSources.ImplicitSpatialSources
import
oracle.spatial.spark.vector.scala.sql.SpatialRDDConversions.ImplicitSpatialRD
DConversions
import scala.collection.mutable.StringBuilder
import oracle.spatial.spark.vector.scala.sql.SpatialEnvironment

object SpatialQueryExample {
  def main(args: Array[String]): Unit = {
    val spark =
SparkSession.builder().appName("SpatialQueryExample").getOrCreate()
    //Setup spatial SQL environment
         SpatialEnvironment.setup(spark)
         val geoJSONInput = args(0)
         //The coordinate system the spatial data is expected to be
         val srid = 8307
         // list of GeoJSON field names to be loaded for each feature
```

```
        val fieldNames = Seq("id", "followers_count",
"friends_count", "location")
        // Create a spatial RDD from a GeoJSON file
        val spatialRDD =
spark.sparkContext.readGeoJSONRecordInfo(geoJSONInput, srid,

fieldNames)
        // Create a DataSet from the SpatialRDD.
        val spatialDF = spatialRDD.toDataFrame(fieldNames)(spark)
        // Register the dataset so it can be used within SQL
statements
        spatialDF.createOrReplaceTempView("sample_tweets")
        //polygon used to spatially filter data
        val qryWindow = """{"type": "Polygon","coordinates":
              [[[-106, 25], [-106, 30], [-104, 30], [-104, 25],
[-106, 25]]]}"""

    // Filter the tweets within the query window (somewhere in the
north of Mexico)
        val query =s""" SELECT geometry, friends_count, location,
followers_count
                      | FROM sample_tweets
                      | WHERE
                      |   ST_CONTAINS(ST_POLYGON('$qryWindow', $srid),
                ST_POINT(geometry, $srid), 0.05)
                      |   AND followers_count > 50 """.stripMargin
    //Execute the query
        val results = spark.sql(query)
        results.show()
  }
}
```

Using Spatial Indexes with Spark UDFs

Spatial Spark SQL UDFs can process indexed data sets. You can create an index on the fly or you can use a persisted spatial index. For more information, see Spatially Indexing a Spatial RDD.

The following example filters the tweet records that spatially interact with a specified polygon or with fewer than 2 followers, and it uses the spatial index option to include the SRID in the geometry column. In this scenario there is no need to wrap the geometry in a Type function.

The general steps are:

1. Set up the spatial SQL environment.
2. Read a persisted index into a DataSet and register it as a table.
3. Create the query to filter the records.
4. Execute the filter.

Java Example:

```
import org.apache.spark.SparkConf;
import org.apache.spark.sql.SparkSession;
import org.apache.spark.sql.types.DataTypes;
```

```
import org.apache.spark.sql.types.Metadata;
import org.apache.spark.sql.types.StructField;
import org.apache.spark.sql.types.StructType;

import oracle.spatial.spark.vector.scala.sql.SpatialEnvironment;
import
oracle.spatial.spark.vector.scala.sql.index.quadtree.QuadTreeIndexRelation;
import
oracle.spatial.spark.vector.serialization.SpatialVectorKryoRegistrator;

public class IndexOptionsAndSchemaTypesExample {
  public static void main(String[] args) {
    SparkConf conf = new SparkConf();
    // the index is expected to have its partitions indexed with an R-Tree
         // so the following line is required if Kryo is used
         SpatialVectorKryoRegistrator.register(conf);
    SparkSession
spark=SparkSession.builder().config(conf).appName("I").getOrCreate();
    //Setup spatial SQL environment
         SpatialEnvironment.setup(spark);
         String indexPath = args[0];
         //Create the required schema for the index.
         StructType schema = new StructType(new StructField[]{
    new StructField("followers_count", DataTypes.IntegerType, true,
Metadata.empty()),
         new StructField("friends_count", DataTypes.IntegerType, true,
Metadata.empty()),
         new StructField("location", DataTypes.StringType, true,
Metadata.empty())
         });
    //read an existing spatial index and register it as table called
"tweets_index"
         spark.read().format(QuadTreeIndexRelation.Format()).schema(schema)
         .option(QuadTreeIndexRelation.OptIncludeCRS(), true)//avoid using
Type Functions
         .load(indexPath).createOrReplaceTempView("tweets_index");

    //polygon used to spatially filter data
         String qryWindow = "{\"type\": \"Polygon\",\"coordinates\":
[[[-106, 25],
                [-106, 30], [-104, 30], [-104, 25], [-106, 25]]]}";

    // Retrieve all the tweets which spatially interact with the given
polygon
  // Note that geometry column is not surrounded by the ST_POINT function
         StringBuilder query =new StringBuilder()
                .append(" SELECT geometry, friends_count, location,
followers_count")
                .append("       FROM tweets_index ")
                .append("       WHERE ")
                .append("   ST_ANYINTERACT(

ST_POLYGON('").append(qryWindow).append("', 8307),
                                                   geometry, 0.05)")
                .append("   OR followers_count = 2 ");
```

```
                System.out.println(query);
                spark.sql(query.toString()).show();
  }
}
```

Scala Example:

```
import org.apache.spark.sql.SparkSession
import oracle.spatial.spark.vector.sql.udf.function.FunctionExecutor
import
oracle.spatial.spark.vector.scala.io.SpatialSources.ImplicitSpatialSour
ces
import
oracle.spatial.spark.vector.scala.sql.SpatialRDDConversions.ImplicitSpa
tialRDDConversions
import scala.collection.mutable.StringBuilder
import org.apache.spark.SparkConf
import
oracle.spatial.spark.vector.serialization.SpatialVectorKryoRegistrator
import oracle.spatial.spark.vector.scala.sql.SpatialEnvironment
import
oracle.spatial.spark.vector.scala.sql.index.quadtree.QuadTreeIndexRelat
ion
import oracle.spatial.spark.vector.scala.sql.util.SchemaUtils
import org.apache.spark.sql.types.StructField
import oracle.spatial.spark.vector.scala.sql.util.SchemaUtils
import org.apache.spark.sql.types.StructType
import org.apache.spark.sql.types.IntegerType
import org.apache.spark.sql.types.Metadata
import org.apache.spark.sql.types.StringType

object IndexOptionsAndSchemaTypesExample {
  def main(args: Array[String]): Unit = {
    val conf = new SparkConf
    //the index is expected to have its partitions indexed with an R-
Tree
    //so the following line is required if Kryo is used
    SpatialVectorKryoRegistrator.register(conf)
    val spark =
SparkSession.builder().config(conf).appName("IndexEx").getOrCreate()
    //Setup spatial SQL environment
    SpatialEnvironment.setup(spark)
    val indexPath = args(0)

    //Create the required schema for the index
    val schema = StructType(Array(
                    StructField("followers_count",IntegerType, true,
Metadata.empty),
            StructField("friends_count",IntegerType, true,
Metadata.empty),
            StructField("location",StringType, true, Metadata.empty)))

    //read an existing spatial index and register it as table called
"tweets_index"
```

```
    spark.read.format(QuadTreeIndexRelation.Format).schema(schema)
    .option(QuadTreeIndexRelation.OptIncludeCRS, true)//set to avoid using
Type Functs
         .load(indexPath).createOrReplaceTempView("tweets_index")

    //polygon used to spatially filter the data
    val polygonJSON = """{"type": "Polygon", "coordinates": [[[-106, 25],
[-106, 30],
            [-104, 30], [-104, 25], [-106, 25]]]}"""

    //Spatial reference system ID of the data
    val srid = 8307
    //Retrieve tweets which spatially interact with the given polygon
    //Note that geometry column is not surrounded by the ST_POINT function
    val query = s"""SELECT geometry, location, friends_count, followers_count
                | FROM tweets_index
                | WHERE
                |  ST_ANYINTERACT( ST_POLYGON('$polygonJSON',$srid),
geometry, 0.05 )
                |  OR followers_count = 2 """.stripMargin
    println(s"Executing: \n$query")
    val results = spark.sql(query)
    results.show()
  }
}
```

2.11.8 Rendering Spatial Indexes on Maps

The Java API provides two ways to generate results based on a spatial index that can be rendered on maps using the Oracle Map API.

- Render all the records or filtered records as an image. Some examples of the possibilities are:
  - Render an image using a numerical attribute and setting the minimum and maximum values.
  - Render an image using a numerical attribute and setting limits with defined colors.
  - Render an image using a textual attribute and setting possible values with defined colors.
- Render the records as a categorization result (such as by countries or regions). Some examples of the possibilities are:
  - Categorize by country using a numerical attribute and setting minimum and maximum values, using default colors and the average or sum as categorization operation.
  - Categorize using the count operation and setting limits and colors.
  - Categorize using the count operation, the default colors, and the automatic limits detection.
  - Categorize using textual attribute and setting possible values and colors.

Detailed code examples can be found under the folder `/opt/oracle/oracle-spatial-graph/spatial/vector/examples/`.

2.11.9 JDBC Data Sources for Spatial RDDs

Oracle Database data can be used as the data source of a Spatial RDD by using the Spark Vector Analysis API.

The class `oracle.spatial.spark.vector.util.JDBCUtils` (or `oracle.spatial.spark.vector.scala.util.JDBCUtils` for Scala) provides convenience methods for creating a Spatial RDD from an Oracle database table or from a SQL query to an Oracle database. The table or SQL query should contain one column of type SDO_GEOMETRY in order to create a Spatial RDD.

Both the from-table and from-query method versions require a connection to the Oracle database, which is supplied by a lambda function defined by the template `oracle.spatial.spark.vector.util.ConnectionSupplier` (or `oracle.spatial.spark.vector.scala.util.ConnectionSupler` for Scala).

The resulting Spatial RDD type parameter will always be `SparkRecordInfo`, that is, the resulting RDD will contain records of the type `SparkRecordInfo`, which will contain the fields specified when querying the table or the columns in the SELECT section of the SQL query. By default, the name and type of the columns retrieved are inferred using the `ResultSet` metadata; however, you can control the naming and type of the retrieved fields by supplying an implementation of `SparkRecordInfoProvider`

The following examples show how to create a Spatial RDD from a table and from a SQL query respectively.

Example 2-3 Creating a Spatial RDD from a Database Table

```
SpatialJavaRDD<SparkRecordInfoProvider> jdbcSpatialRDD =
JDBCUtils.createSpatialRDDFromTable(
    sparkContext, //spark context
    ()->{
        Class.forName("oracle.jdbc.driver.OracleDriver");
        return new DriverManager.getConnection(connURL, usr, pwd);
    }, //DB connection supplier lambda
    "VEHICLES", //DB table
    Arrays.asList(new String[]{"ID","DESC","LOCATION"}), //list of
fields to retrieve
    null //SparkRecordInfoProvider<ResultSet, SparkRecordIngo>
(optional)
);
```

Example 2-4 Creating a Spatial RDD from a SQL Query to the Database

```
SpatialJavaRDD<SparkRecordInfoProvider> jdbcSpatialRDD =
JDBCUtils.createSpatialRDDFromQuery(
    sparkContext, //spark context
    ()->{
        Class.forName("oracle.jdbc.driver.OracleDriver");
        return new DriverManager.getConnection(connURL, usr, pwd);
    }, //DB connection supplier lambda
    "SELECT * FROM VEHICLES WHERE category > 5", //SQL query
    null //SparkRecordInfoProvider<ResultSet, SparkRecordIngo>
```

```
(optional)
);
```

In the preceding examples, data from the Oracle database is queried and partitioned to create a Spark RDD. The number and size of the partitions is determined automatically by the Spark Vector Analysis API.

You can also specify the desired number of database rows to be contained in a Spark partition by calling a method overload that takes this number as a parameter. Manually specifying the number of rows per partition can improve the performance of the Spatial RDD creation.

2.12 Oracle Big Data Spatial Vector Hive Analysis

Oracle Big Data Spatial Vector Hive Analysis provides spatial functions to analyze the data using Hive.

The spatial data can be in any Hive supported format. You can also use a spatial index created with the Java analysis API (see Spatial Indexing) for fast processing.

The supported features include:

- Using the Hive Spatial API
- Using Spatial Indexes in Hive

See also HiveRecordInfoProvider for details about the implementation of these features.

Hive and Spark Spatial SQL Functions provides reference information about the available functions.

Prerequisite Libraries

The following libraries are required by the Spatial Vector Hive Analysis API.

- `sdohadoop-vector-hive.jar`
- `sdohadoop-vector.jar`
- `sdoutil.jar`
- `sdoapi.jar`
- `ojdbc.jar`

- HiveRecordInfoProvider
- Using the Hive Spatial API
- Using Spatial Indexes in Hive

2.12.1 HiveRecordInfoProvider

A record in a Hive table may contain a geometry field in any format like JSON, WKT, or a user-specifiedformat. Geometry constructors like ST_Geometry can create a geometry receiving the GeoJSON, WKT, or WKB representation of the geometry. If the geometry is stored in another format, a HiveRecordInfoProvider can be used.

`HiveRecordInfoProvider` is a component that interprets the geometry field representation and returns the geometry in a GeoJSON format.

The returned geometry must contain the geometry SRID, as in the following example format:

```
{"type":<geometry-type", "crs": {"type": "name", "properties": {"name":
"EPSG:4326"}}"coordinates":[c1,c2,....cn]}
```

The `HiveRecordInfoProvider` interface has the following methods:

- `void setCurrentRecord(Object record)`
- `String getGeometry()`

The method `setCurrentRecord()` is called by passing the current geometry field provided when creating a geometry in Hive. The `HiveRecordInfoProvider` is used then to get the geometry or to return null if the record has no spatial information.

The information returned by the `HiveRecordInfoProvider` is used by the Hive Spatial functions to create geometries (see Hive and Spark Spatial SQL Functions).

Sample HiveRecordInfoProvider Implementation

This sample implementation, named `SimpleHiveRecordInfoProvider`, takes text records in JSON format. The following is a sample input record:

```
{"longitude":-71.46, "latitude":42.35}
```

When `SimpleHiveRecordInfoProvider` is instantiated, a JSON `ObjectMapper` is created. The `ObjectMapper` is used to parse records values later when `setCurrentRecord()` is called. The geometry is represented as latitude-longitude pair, and is used to create a point geometry using the `JsonUtils.readGeometry()` method. Then the GeoJSON format to be returned is created using `GeoJsonGen.asGeometry()`, and the SRID is added to the GeoJSON using `JsonUtils.addSRIDToGeoJSON()`.

```
public class SimpleHiveRecordInfoProvider implements
HiveRecordInfoProvider{
  private static final Log LOG =
    LogFactory.getLog(SimpleHiveRecordInfoProvider.class.getName());

  private JsonNode recordNode = null;
  private ObjectMapper jsonMapper = null;

  public SimpleHiveRecordInfoProvider(){
    jsonMapper = new ObjectMapper();
  }

  @Override
  public void setCurrentRecord(Object record) throws Exception {
    try{
      if(record != null){
        //parse the current value
        recordNode = jsonMapper.readTree(record.toString());
      }
    }catch(Exception ex){
      recordNode = null;
      LOG.warn("Problem reading JSON record
```

```
        value:"+record.toString(), ex);
      }
    }

    @Override
    public String getGeometry() {
      if(recordNode == null){
        return null;
      }

      JGeometry geom = null;

      try{
        geom = JsonUtils.readGeometry(recordNode,
            2, //dimensions
            8307 //SRID
            );
      }catch(Exception ex){
        recordNode = null;
        LOG.warn("Problem reading JSON record
          geometry:"+recordNode.toString(), ex);
      }

      if(geom != null){
        StringBuilder res = new StringBuilder();
        //Get a GeoJSON representation of the JGeometry
        GeoJsonGen.asGeometry(geom, res);
        String result = res.toString();
        //add SRID to GeoJSON and return the result
        return JsonUtils.addSRIDToGeoJSON(result, 8307);
      }

       return null;
    }
  }
```

2.12.2 Using the Hive Spatial API

The Hive Spatial API consists of Oracle-supplied Hive User Defined Functions that can be used to create geometries and perform operations using one or two geometries.

The functions can be grouped into logical categories: types, single-geometry, and two-geometries. (Hive and Spark Spatial SQL Functions lists the functions in each category and provides reference information about each function.)

Example 2-5 Hive Script

The following example script returns information about Twitter users in a data set who are within a specified geographical polygon and who have more than 50 followers. It does the following:

1. Adds the necessary jar files:

```
add jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/ojdbc8.jar
```

```
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoutl.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoapi.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector-hive.jar;
```

2. Creates the Hive user-defined functions that will be used:

```
create temporary function ST_Point as
'oracle.spatial.hadoop.vector.hive.ST_Point';
create temporary function ST_Polygon as
'oracle.spatial.hadoop.vector.hive.ST_Polygon';
create temporary function ST_Contains as
'oracle.spatial.hadoop.vector.hive.function.ST_Contains';
```

3. Creates a Hive table based on the files under the HDFS directory `/user/oracle/twitter`. The `InputFormat` used in this case is `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat` and the Hive SerDe is a user-provided SerDe `oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe`.

```
CREATE EXTERNAL TABLE IF NOT EXISTS sample_tweets (id STRING,
geometry STRING, followers_count STRING, friends_count STRING,
location STRING)
ROW FORMAT SERDE
'oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe'
STORED AS INPUTFORMAT
'oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat'
OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION '/user/oracle/twitter';
```

4. Runs a spatial query receiving an ST_Polygon query area and the ST_Point tweets geometry, and using 0.5 as the tolerance value for the spatial operation. The output will be information about Twitter users in the query area who have more than 50 followers.

```
SELECT id, followers_count, friends_count, location FROM
sample_tweets
WHERE ST_Contains(
  ST_Polygon(
    '{"type": "Polygon",
    "coordinates":
      [[[-106, 25],[-106, 30], [-104, 30], [-104, 25], [-106,
25]]]}',
    8307
  ),
  ST_Point(geometry, 8307),
  0.5
)
and followers_count > 50;
```

The complete script is as follows:

```
add jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/ojdbc8.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoutl.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoapi.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-vector.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-vector-
hive.jar;

create temporary function ST_Point as
'oracle.spatial.hadoop.vector.hive.ST_Point';
create temporary function ST_Polygon as
'oracle.spatial.hadoop.vector.hive.ST_Polygon';
create temporary function ST_Contains as
'oracle.spatial.hadoop.vector.hive.function.ST_Contains';

CREATE EXTERNAL TABLE IF NOT EXISTS sample_tweets (id STRING, geometry
STRING, followers_count STRING, friends_count STRING, location
STRING)
ROW FORMAT SERDE
'oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe'
STORED AS INPUTFORMAT
'oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION '/user/oracle/twitter';


SELECT id, followers_count, friends_count, location FROM sample_tweets
WHERE
ST_Contains(
  ST_Polygon(
    '{"type": "Polygon",
    "coordinates":
      [[[-106, 25],[-106, 30], [-104, 30], [-104, 25], [-106, 25]]]}',
    8307
  ),
  ST_Point(geometry, 8307),
  0.5
)
and followers_count > 50;
```

2.12.3 Using Spatial Indexes in Hive

Hive spatial queries can use a previously created spatial index, which you can create using the Java API (see Spatial Indexing).

If you do not need to use the index in API functions that will access the original data, you can specify `isMapFileIndex=false` when you call `oracle.spatial.hadoop.vector.mapred.job.SpatialIndexing`, or you can use the function `setMapFileIndex(false)`. In these cases, the index will have the following structure:

```
HDFSIndexDirectory/part-xxxxx
```

And in these cases, when creating a Hive table, just provide the folder where you created the index.

If you need to access the original data and you do not set the parameter `isMapFileIndex=false`, the index structure is as follows:

```
part-xxxxx
  data
  index
```

In such cases, to create a Hive table, the data files of the index are needed. Copy the `data` files into a new HDFS folder, with each data file having a different name, like data1, data2,, and so on. The new folder will be used to create the Hive table.

The index contains the geometry records and extra fields. That data can be used when creating the Hive table.

(Note that Spatial Indexing Class Structure describes the index structure, and RecordInfoProvider provides an example of a `RecordInfoProvider` adding extra fields.)

`InputFormat oracle.spatial.hadoop.vector.mapred.input.SpatialIndexTextInputFormat` will be used to read the index. The output of this `InputFormat` is GeoJSON.

Before running any query, you can specify a minimum bounding rectangle (MBR) that will perform a first data filtering using `SpatialIndexTextInputFormat`..

Example 2-6 Hive Script Using a Spatial Index

The following example script returns information about Twitter users in a data set who are within a specified geographical polygon and who have more than 50 followers. It does the following:

1. Adds the necessary jar files:

   ```
   add jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/ojdbc8.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoutl.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoapi.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
   vector.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
   vector-hive.jar;
   ```

2. Creates the Hive user-defined functions that will be used:

   ```
   create temporary function ST_Point as
   'oracle.spatial.hadoop.vector.hive.ST_Point';
   create temporary function ST_Polygon as
   'oracle.spatial.hadoop.vector.hive.ST_Polygon';
   create temporary function ST_Contains as
   'oracle.spatial.hadoop.vector.hive.function.ST_Contains';
   ```

3. Sets the data maximum and minimum boundaries (dim1Min,dim2Min,dim1Max,dim2Max):

```
set oracle.spatial.boundaries=-180,-90,180,90;
```

4. Sets the extra fields contained in the spatial index that will be included in the table creation:

```
set
oracle.spatial.index.includedExtraFields=followers_count,friends_count,loc
ation;
```

5. Creates a Hive table based on the files under the HDFS directory /user/oracle/twitter. The `InputFormat` used in this case is `oracle.spatial.hadoop.vector.mapred.input.SpatialIndexTextInputFormat` and the Hive SerDe is a user-provided SerDe `oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe`. (The code for `oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe` is included with the Hive examples.) The geometry of the tweets will be saved in the geometry column with the format {"longitude":*n*, "latitude":*n*} :

```
CREATE EXTERNAL TABLE IF NOT EXISTS sample_tweets_index (id STRING,
geometry STRING, followers_count STRING, friends_count STRING, location
STRING)
ROW FORMAT SERDE
'oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe'
STORED AS INPUTFORMAT
'oracle.spatial.hadoop.vector.mapred.input.SpatialIndexTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION '/user/oracle/twitter/index';
```

6. Defines the minimum bounding rectangle (MBR) to filter in the `SpatialIndexTextInputFormat`. Any spatial query will only have access to the data in this MBR. If no MBR is specified, then the data boundaries will be used. This setting is recommended to improve the performance.

```
set oracle.spatial.spatialQueryWindow={"type": "Polygon","coordinates":
[[[-107, 24], [-107, 31], [-103, 31], [-103, 24], [-107, 24]]]};
```

7. Runs a spatial query receiving an ST_Polygon query area and the ST_Point tweets geometry, and using 0.5 as the tolerance value for the spatial operation. The tweet geometries are in GeoJSON format, and the ST_Point function is usedspecifying the SRID as 8307.. The output will be information about Twitter users in the query area who have more than 50 followers.

```
SELECT id, followers_count, friends_count, location FROM sample_tweets
WHERE ST_Contains(
  ST_Polygon('{"type": "Polygon","coordinates": [[[-106, 25], [-106, 30],
[-104, 30], [-104, 25], [-106, 25]]]}', 8307)
  , ST_Point(geometry, 8307)
  , 0.5)
  and followers_count > 50;
```

The complete script is as follows. (Differences between this script and the one in Using the Hive Spatial API are marked in bold; however, all of the steps are described in the preceding list.)

```
add jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/ojdbc8.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoutl.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoapi.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector.jar
  /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector-hive.jar;

create temporary function ST_Polygon as
'oracle.spatial.hadoop.vector.hive.ST_Polygon';
create temporary function ST_Point as
'oracle.spatial.hadoop.vector.hive.ST_Point';
create temporary function ST_Contains as
'oracle.spatial.hadoop.vector.hive.function.ST_Contains';

set oracle.spatial.boundaries=-180,-90,180,90;
set
oracle.spatial.index.includedExtraFields=followers_count,friends_count,
location;

CREATE EXTERNAL TABLE IF NOT EXISTS sample_tweets_index (id STRING,
geometry STRING, followers_count STRING, friends_count STRING,
location STRING)
ROW FORMAT SERDE
'oracle.spatial.hadoop.vector.hive.json.GeoJsonSerDe'
STORED AS INPUTFORMAT
'oracle.spatial.hadoop.vector.mapred.input.SpatialIndexTextInputFormat'
OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION '/user/oracle/twitter/index';

set oracle.spatial.spatialQueryWindow={"type":
"Polygon","coordinates": [[[-107, 24], [-107, 31], [-103, 31], [-103,
24], [-107, 24]]]};

SELECT id, followers_count, friends_count, location FROM sample_tweets
WHERE ST_Contains(
  ST_Polygon('{"type": "Polygon","coordinates": [[[-106, 25], [-106,
30], [-104, 30], [-104, 25], [-106, 25]]]}', 8307)
  , ST_Point(geometry, 8307)
  , 0.5)
  and followers_count > 50;
```

2.13 Using the Oracle Big Data SpatialViewer Web Application

You can use the Oracle Big Data SpatialViewer Web Application (SpatialViewer) to perform a variety of tasks.

These include tasks related to spatial indexing, creating and showing thematic maps, loading rasters into HDFS, visualizing uploaded rasters in the globe, selecting individual or multiple footprints, performing raster algebra operations, dealing with gaps and overlaps, combining selected footprints, generating a new image with the specified file format from the selected footprints, and applying user-specific processing.

- Creating a Hadoop Spatial Index Using SpatialViewer
- Exploring the Hadoop Indexed Spatial Data
- Creating a Spark Spatial Index Using SpatialViewer
- Performing Spatial Analysis
- Running a Categorization Job Using SpatialViewer
- Viewing the Categorization Results
- Saving Categorization Results to a File
- Creating and Deleting Templates
- Configuring Templates
- Running a Clustering Job Using SpatialViewer
- Viewing the Clustering Results
- Saving Clustering Results to a File
- Running a Binning Job Using SpatialViewer
- Viewing the Binning Results
- Saving Binning Results to a File
- Running a Job to Create an Index Using the Command Line
- Running a Job to Create a Categorization Result
- Running a Job to Create a Clustering Result
- Running a Job to Create a Binning Result
- Running a Job to Perform Spatial Filtering
- Running a Job to Get Location Suggestions
- Running a Job to Perform a Spatial Join
- Running a Job to Perform Partitioning
- Using Multiple Inputs
- Loading Images from the Local Server to the HDFS Hadoop Cluster
- Visualizing Rasters in the Globe
- Processing a Raster or Multiple Rasters with the Same MBR
- Creating a Mosaic Directly from the Globe
- Adding Operations for Raster Processing

- Creating a Slope Image from the Globe
- Changing the Image File Format from the Globe

2.13.1 Creating a Hadoop Spatial Index Using SpatialViewer

To create a Hadoop spatial index using SpatialViewer, follow these steps.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`
2. Click **Spatial Index**.
3. Specify all the required details:
   a. Index name.
   b. Path of the file or files to index in HDFS. For example, `/user/oracle/bdsg/tweets.json`.
   c. SRID of the geometries to be indexed. Example: 8307
   d. Tolerance of the geometries to be indexed. Example: 0.05
   e. Input Format class: The input format class. For example: `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat`
   f. Record Info Provider class: The class that provides the spatial information. For example: `oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProvider`.

      > **Note:**
      >
      > If the `InputFormat` class or the `RecordInfoProvider` class is not in the API, or in the hadoop API classes, then a jar with the user-defined classes must be provided. To be able to use this jar, you must add it in the `/opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/WEB-INF/lib` directory and restart the server.

   g. Whether the enrichment service (`MVSuggest`) must be used or not. If the geometry has to be found from a location string, then use the `MVSuggest` service. In this case the provided `RecordInfoProvider` must implement the interface `oracle.spatial.hadoop.vector.LocalizableRecordInfoProvider`.
   h. MVSuggest Templates (Optional): When using the `MVSuggest` service, you can define the templates used to create the index.
4. Click **Create**.

   A URL will be displayed to track the job.

2.13.2 Exploring the Hadoop Indexed Spatial Data

To explore Hadoop indexed spatial data, follow these steps.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`

2. Click **Explore Data**.

   For example, you can:

   - Select the desired indexed data and use the rectangle tool to display the data in the desired area.
   - Change the background map style.
   - Show data using a heat map.

2.13.3 Creating a Spark Spatial Index Using SpatialViewer

To create a Spark spatial index using SpatialViewer, follow these steps.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vectorspark`
2. Click **Spatial Index**.
3. Specify all the required details:
   a. Index name.
   b. Path of the file or files to index in HDFS. For example, `/user/oracle/bdsg/tweets.json`.
   c. SRID of the geometries to be indexed. Example: 8307
   d. Input Format class (optional): The input format class. For example: `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat`
   e. Key class (required if an input format class is defined): Class of the input format keys. For example: `org.apache.hadoop.io.LongWritable`
   f. Value class (required if an input format class is defined): Class of the input format values. For example: `org.apache.hadoop.io.Text`
   g. Record Info Provider class: The class that provides the spatial information. For example: `oracle.spatial.spark.vector.recordinfoprovider.GeoJsonRecordInfoProvider`

   > **Note:**
   >
   > If the `InputFormat` class or the `RecordInfoProvider` class is not in the API, or in the hadoop API classes, then a jar with the user-defined classes must be provided. To be able to use this jar the user must add it in the `/opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/WEB-INF/lib directory` and restart the server.

4. Click **Create**.

   A URL will be displayed to track the job.

2.13.4 Performing Spatial Analysis

To start spatial analysis:

1. Open the console: http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vectorspark
2. Click **Spatial Analysis**.

Then perform spatial analysis actions as desired. For example, you can:

- Select the desired indexed data and use the rectangle tool to display the data in the desired area.
- Change the background map style.
- Perform the following spatial analysis operations on indexed data or on previous analysis results:
  - Buffer: Expand a shape by a specified distance.
  - Convex hull: Envelop a shape disallowing inward curves.
  - Distance: Compute the shortest distance between shapes.
  - Envelope: Return the smallest rectangle fitting the shape.
  - Simplify: Reduce vertices maintaining approximate shape using the Douglas-Peucker algorithm.
  - SimplifyVW: Reduce vertices maintaining approximate shape using the Visvalingham-Whyatt algorithm.
  - Any interaction: Filter shapes interacting in any way.
  - Contains: Filter shapes containing a specified geometry.
  - Inside: Filter shapes that are inside a specified geometry.
  - Within distance: Filter shapes that are within a specified distance apart from other shapes.
  - Nearest neighbors: Find the specified number of shapes closest to a shape.

> **Note:**
>
> You can also set non-spatial conditions on the indexed data to have them considered in the analysis.
>
> When two geometries are specified, the second one can be indexed data, previous analysis results, or a specified geometry picked from a list of predefined geometries (for example a city, region or country), picked in a map, or defined using GeoJSON. (For predefined geometries, see Creating and Deleting Templates to add or delete templates for selecting.)

- Create spatial views on indexed data or on analysis results. For example, create views to:
  - Render the records as an image. Possibilities include:
    * Render an image using a numerical attribute and setting the minimum and maximum values.
    * Render an image using a numerical attribute and setting limits with defined colors.

        * Render an image using a textual attribute and setting possible values with defined colors.
    – Render the records as a categorization result (by countries or regions for example). Possibilities include:
        * Categorize by country using a numerical attribute and setting minimum and maximum values. The average or sum can be used as the categorization operation.
        * Categorize using the count operation and setting limits and colors.
        * Categorize using the count operation, default colors, and automatic limits detection.
        * Categorize using a textual attribute and setting possible values and colors.

2.13.5 Running a Categorization Job Using SpatialViewer

You can run a categorization job with or without the spatial index. Follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Categorization**, then **Categorization Job**.
3. Select either With Index or Without Index and provide the following details, as required:
    - With Index
        a. Index name
    - Without Index
        a. Path of the data: Provide the HDFS data path. For example, `/user/oracle/bdsg/tweets.json`.
        b. JAR with user classes (Optional): If the `InputFormat` class or the `RecordInfoProvider` class is not in the API, or in the hadoop API classes, then a jar with the user-defined classes must be provided. To be able to use this jar the user must add it in the `/opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/WEB-INF/lib` directory and restart the server.
        c. Input Format class: The input format class. For example: `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat`
        d. Record Info Provider class: The class that will provide the spatial information. For example: `oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProvider`.
        e. Whether the enrichment service `MVSuggest` service must be used or not. If the geometry must be found from a location string, then use the `MVSuggest` service. In this case the provided `RecordInfoProvider` has to implement the interface `oracle.spatial.hadoop.vector.LocalizableRecordInfoProvider`.
        f. Templates: The templates to create the thematic maps.

> **Note:**
>
> If a template refers to point geometries (for example, cities), the result returned is empty for that template, if `MVSuggest` is not used. This is because the spatial operations return results only for polygons.

> **Tip:**
>
> When using the `MVSuggest` service the results will be more accurate if all the templates that could match the results are provided. For example, if the data can refer to any city, state, country, or continent in the world, then the better choice of templates to build results are World Continents, World Countries, World State Provinces, and World Cities. On the other hand, if the data is from the USA states and counties, then the suitable templates are USA States and USA Counties. If an index that was created using the `MVSuggest` service is selected, then select the top hierarchy for an optimal result. For example, if it was created using World Countries, World State Provinces, and World Cities, then use World Countries as the template.

g. Result name: The result name. If a result exists for a template with the same name, it is overwritten. For example, `Tweets test`.

Click **Create**. A URL will be displayed to track the job.

2.13.6 Viewing the Categorization Results

To view the categorization results, follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Categorization**, then **Results**.
3. Click any one of the Templates. For example, World Continents.

   The World Continents template is displayed.
4. Click any one of the Results displayed.

   Different continents appear with different patches of colors.
5. Click any continent from the map. For example, North America.

   The template changes to World Countries and the focus changes to North America with the results by country.

2.13.7 Saving Categorization Results to a File

You can save categorization results to a file (for example, the result file created with a job executed from the command line) on the local system for possible future uploading and use. The templates are located in the folder `/opt/oracle/oracle-spatial-`

graph/spatial/web-server/spatialviewer/templates. The templates are GeoJSON files with features and all the features have ids. For example, the first feature in the template *USA States* starts with: {"type":"Feature","_id":"WYOMING",...

The results must be JSON files with the following format: {"id":"JSONFeatureId","result":result}.

For example, if the template *USA States* is selected, then a valid result is a file containing: {"id":"WYOMING","result":3232} {"id":"SOUTH DAKOTA","result":74968}

1. Click **Categorization**, then **Results**.
2. Select a Template .
3. Click the icon for saving the results.
4. Specify a Name.
5. Click Choose File to select the File location.
6. Click Save.

   The results can be located in the folder clustering_results contained in the SpatialViewer local working directory (see Configuring SpatialViewer on Oracle Big Data Appliance).

2.13.8 Creating and Deleting Templates

To create new templates do the following:

1. Add the template JSON file in the folder /opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/templates/.
2. Add the template configuration file in the folder /opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/templates/_config_.

To delete the template, delete the JSON and configuration files added in steps 1 and 2.

2.13.9 Configuring Templates

Each template has a configuration file. The template configuration files are located in the folder /opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/templates/_config_. The name of the configuration file is the same as the template files suffixed with config.json instead of .json.For example, the configuration file name of the template file usa_states.json is usa_states.config.json. The configuration parameters are:

- name: Name of the template to be shown on the console. For example, name: USA States.
- display_attribute: When displaying a categorization result, a cursor move on the top of a feature displays this property and result of the feature. For example, display_attribute: STATE NAME.
- point_geometry: True, if the template contains point geometries and false, in case of polygons. For example, point_geometry: false.
- child_templates (optional): The templates that can have several possible child templates separated by a coma. For example, child_templates: ["world_states_provinces, usa_states(properties.COUNTRY CODE:properties.PARENT_REGION)"].

If the child templates do not specify a linked field, it means that all the features inside the parent features are considered as child features. In this case, the `world_states_provinces` doesn't specify any fields. If the link between parent and child is specified, then the spatial relationship doesn't apply and the feature properties link are checked. In the above example, the relationship with the `usa_states` is found with the property `COUNTRY CODE` in the current template, and the property P`ARENT_REGION` in the template file `usa_states.json`.

- srid: The SRID of the template's geometries. For example, `srid: 8307`.
- back_polygon_template_file_name (optional): A template with polygon geometries to set as background when showing the defined template. For example, `back_polygon_template_file_name: usa_states`.
- vectorLayers: Configuration specific to the `MVSuggest` service. For example:

```
{
"vectorLayers": [
{
        "gnidColumns":["_GNID"],
        "boostValues":[2.0,1.0,1.0,2.0]
     }
     ]
 }
```

Where:

  – gnidColumns is the name of the column(s) within the Json file that represents the Geoname ID. This value is used to support multiple languages with `MVSuggest`. (See references of that value in the file `templates/_geonames_/alternateNames.json`.) There is nodefault value for this property.

  – boostValues is an array of float numbers that represent how important a column is within the "properties" values for a given row. The higher the number, the more important that field is. A value of zero means the field will be ignored. When boostValues is not present, all fields receive a default value of 1.0, meaning they all are equally important properties. The `MVSuggest` service may return different results depending on those values. For a Json file with the following properties, the boost values might be as follows:

```
"properties":{"Name":"New York City","State":"NY","Country":"United
States","Country Code":"US","Population":8491079,"Time Zone":"UTC-5"}
"boostValues":[3.0,2.0,1.0,1.0,0.0,0.0]
```

2.13.10 Running a Clustering Job Using SpatialViewer

To run a clustering job using SpatialViewer, follow these steps.

1. Open: `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`
2. Click **Clustering**, then **Clustering Job**.
3. Provide the following details, as required:
   a. Path of the data: Provide the HDFS data path. For example, `/user/oracle/bdsg/tweets.json`.
   b. The SRID of the geometries. For example: 8307
   c. The tolerance of the geometries. For example: 0.05

   d. JAR with user classes (Optional): If the `InputFormat` class or the `RecordInfoProvider` class is not in the API, or in the hadoop API classes, then a jar with the user-defined classes must be provided. To be able to use this jar the user must add it in the `/opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/WEB-INF/lib` directory and restart the server.
   e. Input Format class: The input format class. For example: `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat`
   f. Record Info Provider class: The class that will provide the spatial information. For example: `oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProvider`.
   g. Number of clusters: The number of clusters to be found.
   h. Result name: The result name. If a result exists for a template with the same name, it is overwritten. For example, Tweets test.
4. Click **Create**.

   A URL will be displayed to track the job.

2.13.11 Viewing the Clustering Results

To view the clustering results, follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Clustering**, then **Results**.
3. Click any one of the Results displayed.

2.13.12 Saving Clustering Results to a File

You can save clustering results to a file on your local system, for later uploading and use. To save the clustering results to a file, follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Clustering**, then **Results**.
3. Click the icon for saving the results.
4. Specify a name.
5. Specify the SRID of the geometries. For example: 8307
6. Click **Choose File** and select the file location.
7. Click **Save**.

2.13.13 Running a Binning Job Using SpatialViewer

You can run a binning job with or without the spatial index. Follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Binning**, then **Binning Job**.
3. Select either With Index or Without Index and provide the following details, as required:

- With Index
  a. Index name
- Without Index
  a. Path of the data: Provide the HDFS data path. For example, `/user/oracle/bdsg/tweets.json`
  b. The SRID of the geometries. For example: 8307
  c. The tolerance of the geometries. For example: 0.05
  d. JAR with user classes (Optional): If the `InputFormat` class or the `RecordInfoProvider` class is not in the API, or in the hadoop API classes, then a jar with the user-defined classes must be provided. To be able to use this jar the user must add it in the `/opt/oracle/oracle-spatial-graph/spatial/web-server/spatialviewer/WEB-INF/lib` directory and restart the server.
  e. Input Format class: The input format class. For example: `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat`
  f. Record Info Provider class: The class that will provide the spatial information. For example: `oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProvider`.

4. Binning grid minimum bounding rectangle (MBR). You can click the icon for seeing the MBR on the map.
5. Binning shape: hexagon (specify the hexagon width) or rectangle (specify the width and height).
6. Thematic attribute: If the job uses an index, double-click to see the possible values, which are those returned by the function `getExtraFields` of the `RecordInfoProvider` used when creating the index. If the job does not use an index, then the field can be one of the fields returned by the function `getExtraFields` of the specified `RecordInfoProvider` class. In any case, the `count` attribute is always available and specifies the number of records in the bin.
7. Result name: The result name. If a result exists for a template with the same name, it is overwritten. For example, Tweets test.

Click **Create**. A URL will be displayed to track the job.

2.13.14 Viewing the Binning Results

To view the binning results, follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Binning**, then **Results**.
3. Click any of the Results displayed.

2.13.15 Saving Binning Results to a File

You can save binning results to a file on your local system, for later uploading and use. To save the binning results to a file, follow these steps.

1. Open `http://<oracle_big_data_spatial_vector_console>:8045/spatialviewer/?root=vector`.
2. Click **Binning**, then **View Results**.
3. Click the icon for saving the results.
4. Specify the SRID of the geometries. For example: 8307
5. Specify the thematic attribute, which must be a property of the features in the result. For example, the count attribute can be used to create results depending on the number of results per bin.
6. Click **Choose File** and select the file location.
7. Click **Save**.

2.13.16 Running a Job to Create an Index Using the Command Line

To create a spatial index, use a command in the following format:

```
hadoop jar <HADOOP_LIB_PATH>/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.SpatialIndexing [generic options] input=<path|
comma_separated_paths|path_pattern> output=<path> inputFormat=<InputFormat_subclass>
recordInfoProvider=<RecordInfoProvider_subclass> [srid=<integer_value>]
[geodetic=<true|false>] [tolerance=<double_value>] [boundaries=<minX,minY,maxX,maxY>]
[indexName=<index_name>] [indexMetadataDir=<path>] [overwriteIndexMetadata=<true|
false>] [  mvsLocation=<path|URL> [mvsMatchLayers=<comma_separated_layers>]
[mvsMatchCountry=<country_name>][mvsSpatialResponse=<[NONE, FEATURE_GEOMETRY,
FEATURE_CENTROID]>][mvsInterfaceType=<LOCAL, WEB>][mvsIsRepository=<true|false>]
[rebuildMVSIndex=<true|false>][mvsPersistentLocation=<hdfs_path>]
[mvsOverwritePersistentLocation=<true|false>] ]
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.SpatialIndexing` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.SpatialIndexing`.

Input/output arguments:

- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression.
- `inputFormat`: the `inputFormat` class implementation used to read the input data.
- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class.
- `output`: the path where the spatial index will be stored

Spatial arguments:

- `srid` (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- `geodetic` (optional, default depends on the srid): boolean value that indicates whether the geometries are geodetic or not.
- `tolerance` (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.
- `boundaries` (optional, default=unbounded): the minimum and maximum values for each dimension, expressed as comma separated values in the form: minX,minY,maxX,maxY

Spatial index metadata arguments:

- `indexName` (optional, default=output folder name):The name of the index to be generated.
- `indexMetadataDir` (optional, default=hdfs://server:port/user/<current_user>/oracle_spatial/index_metadata/): the directory where the spatial index metadata will be stored.
- `overwriteIndexMetadata` (optional, default=false) boolean argument that indicates whether the index metadata can be overwritten if an index with the same name already exists.

`MVSuggest` arguments:

- `mvsLocation`: The path to the MVSuggest directory or repository for local standalone instances of MVSuggest or the service URL when working with a remote instance. This argument is required when working with MVSuggest.
- `mvsMatchLayers` (optional, default=all): comma separated list of layers. When provided, MVSuggest will only use these layers to perform the search.
- `mvsMatchCountry` (optional, default=none): a country name which MVSuggest will give higher priority when performing matches.
- `mvsSpatialResponse` (optional, default=CENTROID): the type of the spatial results contained in each returned match. It can be one of the following values: NONE, FEATURE_GEOMETRY, FEATURE_CENTROID.
- `mvsInterfaceType` (optional: default=LOCAL): the type of MVSuggest service used, it can be LOCAL or WEB.
- `mvsIsRepository` (optional: default=false) (LOCAL only): boolean value which specifies whether mvsLocation points to a whole MVS directory(false) or only to a repository(true). An MVS repository contains only JSON templates; it may or not contain a _config_ and _geonames_ folder.
- `mvsRebuildIndex` (optional, default=false)(LOCAL only):boolean value specifying whether the repository index should be rebuilt or not.
- `mvsPersistentLocation` (optional, default=none)(LOCAL only): an HDFS path where the MVSuggest directory will be saved.
- `mvsIsOverwritePersistentLocation` (optional, default=false): boolean argument that indicates whether an existing mvsPersistentLocation must be overwritten in case it already exists.

Example: Create a spatial index called `indexExample`. The index metadata will be stored in the HDFS directory `spatialMetadata`.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-
vector.jar oracle.spatial.hadoop.vector.mapred.job.SpatialIndexing
input="/user/hdfs/demo_vector/tweets/part*" output=/user/hdfs/
demo_vector/tweets/spatial_index
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFor
mat
recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordIn
foProvider srid=8307 geodetic=true tolerance=0.5
indexName=indexExample indexMetadataDir=indexMetadataDir
overwriteIndexMetadata=true
```

Example: Create a spatial index using `MVSuggest` to assign a spatial location to records that do not contain geometries.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.SpatialIndexing input="/user/hdfs/
demo_vector/tweets/part*" output=/user/hdfs/demo_vector/tweets/spatial_index
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat
recordInfoProvider=mypackage.Simple LocationRecordInfoProvider srid=8307
geodetic=true tolerance=0.5 indexName=indexExample
indexMetadataDir=indexMetadataDir overwriteIndexMetadata=true
mvsLocation=file:///local_folder/mvs_dir/oraclemaps_pub/ mvsRepository=true
```

2.13.17 Running a Job to Create a Categorization Result

To create a categorization result, use a command in one of the following formats.

With a Spatial Index

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.Categorization [generic options]
( indexName=<indexName> [indexMetadataDir=<path>] )  | ( input=<path|
comma_separated_paths|path_pattern> isInputIndex=true [srid=<integer_value>]
[geodetic=<true|false>] [tolerance=<double_value>]
[boundaries=<min_x,min_y,max_x,max_y>]  ) output=<path>
hierarchyIndex=<hdfs_hierarchy_index_path> hierarchyInfo=<HierarchyInfo_subclass>
[hierarchyDataPaths=<level1_path,level2_path,,levelN_path>] spatialOperation=<[None,
IsInside, AnyInteract]>
```

Without a Spatial Index

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.Categorization [generic options] input=<path|
comma_separated_paths|path_pattern> inputFormat=<InputFormat_subclass>
recordInfoProvider=<RecordInfoProvider_subclass> [srid=<integer_value>]
[geodetic=<true|false>] [tolerance=<double_value>]
[boundaries=<min_x,min_y,max_x,max_y>]  output=<path>
hierarchyIndex=<hdfs_hierarchy_index_path> hierarchyInfo=<HierarchyInfo_subclass>
hierarchyDataPaths=<level1_path,level2_path,,levelN_path>] spatialOperation=<[None,
IsInside, AnyInteract]>
```

Using MVSuggest

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.Categorization [generic options]
(indexName=<indexName> [indexMetadataDir=<path>])  |
(
(input=<path|comma_separated_paths|path_pattern> isInputIndex=true)  | (input=<path|
comma_separated_paths|path_pattern> inputFormat=<InputFormat_subclass>
recordInfoProvider=<LocalizableRecordInfoProvider_subclass>)
[srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>]
[boundaries=<min_x,min_y,max_x,max_y>]
) output=<path>
mvsLocation=<path|URL> [mvsMatchLayers=<comma_separated_layers>]
[mvsMatchCountry=<country_name>] [mvsSpatialResponse=<[NONE, FEATURE_GEOMETRY,
FEATURE_CENTROID]>] [mvsInterfaceType=<[UNDEFINED, LOCAL, WEB]>]
[mvsIsRepository=<true|false>] [mvsRebuildIndex=<true|false>]
[mvsPersistentLocation=<hdfs_path>] [mvsOverwritePersistentLocation=<true|false>]
[mvsMaxRequestRecords=<integer_number>]  hierarchyIndex=<hdfs_hierarchy_index_path>
hierarchyInfo=<HierarchyInfo_subclass>
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.Categorization` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.Categorization`.

Input/output arguments:

- `indexName`: the name of an existing spatial index. The index information will be looked at the path given by indexMetadataDir. When used, the argument `input` is ignored.
- `indexMetadataDir` (optional, default=hdfs://server:port/user/<current_user>/oracle_spatial/index_metadata/): the directory where the spatial index metadata is located
- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression. (Ignored if `indexName` is specified.)
- `inputFormat`: the `inputFormat` class implementation used to read the input data. (Ignored if `indexName` is specified.)
- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class. (Ignored if `indexName` is specified.)
- `output`: the path where the spatial index will be stored

Spatial arguments:

- `srid` (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- `geodetic` (optional, default depends on the srid): boolean value that indicates whether the geometries are geodetic or not.
- `tolerance` (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.
- `boundaries` (optional, default=unbounded): the minimum and maximum values for each dimension, expressed as comma separated values in the form: minX,minY,maxX,maxY
- `spatialOperation`: the spatial operation to perform between the input data set and the hierarchical data set. Allowed values are `IsInside` and `AnyInteract`.

Hierarchical data set arguments:

- `hierarchyIndex`: the HDFS path of an existing hierarchical index or where it can be stored if it needs to be generated.
- `hierarchyInfo`: the fully qualified name of a `HierarchyInfo` subclass which is used to describe the hierarchical data.
- `hierarchyDataPaths` (optional, default=none): a comma separated list of paths of the hierarchy data. The paths should be sorted in ascending way by hierarchy level. If a hierarchy index path does not exist for the given hierarchy data, this argument is required.

`MVSuggest` arguments:

- `mvsLocation`: The path to the MVSuggest directory or repository for local standalone instances of MVSuggest or the service URL when working with a remote instance. This argument is required when working with MVSuggest.

- mvsMatchLayers (optional, default=all): comma separated list of layers. When provided, MVSuggest will only use these layers to perform the search.
- mvsMatchCountry (optional, default=none): a country name which MVSuggest will give higher priority when performing matches.
- mvsSpatialResponse (optional, default=CENTROID): the type of the spatial results contained in each returned match. It can be one of the following values: NONE, FEATURE_GEOMETRY, FEATURE_CENTROID.
- mvsInterfaceType (optional: default=LOCAL): the type of MVSuggest service used, it can be LOCAL or WEB.
- mvsIsRepository (optional: default=false) (LOCAL only): Boolean value that specifies whether mvsLocation points to a whole MVS directory(false) or only to a repository(true). An MVS repository contains only JSON templates; it may or not contain a _config_ and _geonames_ folder.
- mvsRebuildIndex (optional, default=false)(LOCAL only):boolean value specifying whether the repository index should be rebuilt or not.
- mvsPersistentLocation (optional, default=none)(LOCAL only): an HDFS path where the MVSuggest directory will be saved.
- mvsIsOverwritePersistentLocation (optional, default=false): boolean argument that indicates whether an existing mvsPersistentLocation must be overwritten in case it already exists.

Example: Run a Categorization job to create a summary containing the records counts by continent, country, and state/provinces. The input is an existing spatial index called indexExample. The hierarchical data will be indexed and stored in HDFS at the path hierarchyIndex.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.Categorization indexName=
indexExample  output=/user/hdfs/demo_vector/tweets/hier_count_spatial
hierarchyInfo=vectoranalysis.categorization.WorldAdminHierarchyInfo
hierarchyIndex=hierarchyIndex  hierarchyDataPaths=file:///templates/
world_continents.json,file:///templates/world_countries.json,file:///
templates/world_states_provinces.json spatialOperation=IsInside
```

Example: Run a Categorization job to create a summary of tweet counts per continent, country, states/provinces, and cities using MVSuggest.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.Categorization input="/user/hdfs/
demo_vector/tweets/part*" inputFormat=<InputFormat_subclass>
recordInfoProvider=<LocalizableRecordInfoProvider_subclass>  output=/user/
hdfs/demo_vector/tweets/hier_count_mvs
hierarchyInfo=vectoranalysis.categorization.WorldAdminHierarchyInfo
hierarchyIndex=hierarchyIndex mvsLocation=file:///mvs_dir
mvsMatchLayers=world_continents,world_countries,world_states_provinces
spatialOperation=IsInside
```

2.13.18 Running a Job to Create a Clustering Result

To create a clustering result, use a command in the following format:

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.KMeansClustering [generic options]
input=<path|comma_separated_paths|path_pattern>
inputFormat=<InputFormat_subclass>
recordInfoProvider=<RecordInfoProvider_subclass> output=<path>
[srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>]
[boundaries=<min_x,min_y,max_x,max_y>]  k=<number_of_clusters>
[clustersPoints=<comma_separated_points_ordinates>] [deleteClusterFiles=<true|
false>] [maxIterations=<integer_value>]
[critFunClass=<CriterionFunction_subclass>]
[shapeGenClass=<ClusterShapeGenerator_subclass>]
[maxMemberDistance=<double_value>]
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.KMeansClustering` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.KMeansClustering`.

Input/output arguments:

- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression.
- `inputFormat`: the `inputFormat` class implementation used to read the input data.
- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class.
- `output`: the path where the spatial index will be stored

Spatial arguments:

- `srid` (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- `geodetic` (optional, default depends on the srid): Boolean value that indicates whether the geometries are geodetic or not.
- `tolerance` (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.
- `boundaries` (optional, default=unbounded): the minimum and maximum values for each dimension, expressed as comma separated values in the form: minX,minY,maxX,maxY
- `spatialOperation`: the spatial operation to perform between the input data set and the hierarchical data set. Allowed values are `IsInside` and `AnyInteract`.

Clustering arguments:

- `k`: the number of clusters to be found.
- `clusterPoints` (optional, default=none): the initial cluster centers as a comma-separated list of point ordinates in the form: p1_x,p1_y,p2_x,p2_y,…,pk_x,pk_y
- `deleteClusterFiles` (optional, default=true): Boolean arguments that specifies whether the intermediate cluster files generated between iterations should be deleted or not

- `maxIterations` (optional, default=calculated based on the number k): the maximum number of iterations allowed before the job completes.
- `critFunClass` (optional, default=oracle.spatial.hadoop.vector.cluster.kmeans. SquaredErrorCriterionFunction) a fully qualified name of a `CriterionFunction` subclass.
- `shapeGenClass` (optional, default= oracle.spatial.hadoop.vector.cluster.kmeans. ConvexHullClusterShapeGenerator) a fully qualified name of a `ClusterShapeGenerator` subclass used to generate the geometry of the clusters.
- `maxMemberDistance` (optional, default=undefined): a double value that specifies the maximum distance between a cluster center and a cluster member.

Example: Run a Clustering job to generate 5 clusters. The generated clusters geometries will be the convex hull of all .

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.KMeansClustering input="/user/hdfs/
demo_vector/tweets/part*" output=/user/hdfs/demo_vector/tweets/result
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat
recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProv
ider srid=8307 geodetic=true tolerance=0.5 k=5
shapeGenClass=oracle.spatial.hadoop.vector.cluster.kmeans.ConvexHullClusterSh
apeGenerator
```

2.13.19 Running a Job to Create a Binning Result

To create a binning result, use a command in the following format:

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.Binning [generic options]
(indexName=<INDEX_NAME> [indexMetadataDir=<INDEX_METADATA_DIRECTORY>]) |
(input=<DATA_PATH> inputFormat=<INPUT_FORMAT_CLASS>
recordInfoProvider=<RECORD_INFO_PROVIDER_CLASS> [srid=<SRID>] [geodetic=<GEODETIC>]
[tolerance=<TOLERANCE>]) output=<RESULT_PATH> cellSize=<CELL_SIZE> gridMbr=<GRID_MBR>
[cellShape=<CELL_SHAPE>] [aggrFields=<EXTRA_FIELDS>]
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.Binning` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.Binning`.

Input/output arguments:

- `indexName`: the name of an existing spatial index. The index information will be looked at the path given by indexMetadataDir. When used, the argument `input` is ignored.
- `indexMetadataDir` (optional, default=hdfs://server:port/user/<current_user>/ oracle_spatial/index_metadata/): the directory where the spatial index metadata is located
- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression.
- `inputFormat`: the `inputFormat` class implementation used to read the input data.
- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class.
- `output`: the path where the spatial index will be stored

Spatial arguments:

- `srid` (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- `geodetic` (optional, default depends on the srid): Boolean value that indicates whether the geometries are geodetic or not.
- `tolerance` (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.

Binning arguments:

- `cellSize`: the size of the cells in the format: width,height
- `gridMbr` : the minimum and maximum dimension values for the grid in the form: minX,minY,maxX,maxY
- `cellShape` (optional, default=RECTANGLE): the shape of the cells. It can be RECTANGLE or HEXAGON
- `aggrFields` (optional, default=none): a comma-separated list of field names that will be aggregated.

Example: Run a spatial binning job to generate a grid of hexagonal cells and aggregate the value of the field SALES..

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-
vector.jar oracle.spatial.hadoop.vector.mapred.job.Binning
indexName=indexExample indexMetadataDir=indexMetadataDir output=/user/
hdfs/demo_vector/result cellShape=HEXAGON cellSize=5
gridMbr=-175,-85,175,85 aggrFields=SALES
```

2.13.20 Running a Job to Perform Spatial Filtering

To perform spatial filtering, use a command in the following format:

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.SpatialFilter [generic options]
( indexName=<indexName> [indexMetadataDir=<path>] )  |
(
(input=<path|comma_separated_paths|path_pattern> isInputIndex=true)  |
(input=<path|comma_separated_paths|path_pattern>
inputFormat=<InputFormat_subclass>
recordInfoProvider=<RecordInfoProvider_subclass>)
[srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>]
[boundaries=<min_x,min_y,max_x,max_y>]
) output=<path>  spatialOperation=<[IsInside, AnyInteract]> queryWindow=<json-
geometry>
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.SpatialFilter` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.SpatialFilter`.

Input/output arguments:

- `indexName`: the name of an existing spatial index. The index information will be looked at the path given by indexMetadataDir. When used, the argument `input` is ignored.

- `indexMetadataDir` (optional, default=hdfs://server:port/user/<current_user>/oracle_spatial/index_metadata/): the directory where the spatial index metadata is located
- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression.
- `inputFormat`: the `inputFormat` class implementation used to read the input data.
- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class.
- `output`: the path where the spatial index will be stored

Spatial arguments:

- `srid` (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- `geodetic` (optional, default depends on the srid): Boolean value that indicates whether the geometries are geodetic or not.
- `tolerance` (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.

Binning arguments:

- `cellSize`: the size of the cells in the format: width,height
- `gridMbr` : the minimum and maximum dimension values for the grid in the form: minX,minY,maxX,maxY
- `cellShape` (optional, default=RECTANGLE): the shape of the cells. It can be RECTANGLE or HEXAGON
- `aggrFields` (optional, default=none): a comma-separated list of field names that will be aggregated.
- `boundaries` (optional, default=unbounded): the minimum and maximum values for each dimension, expressed as comma separated values in the form: minx,minY,maxX,maxY
- `spatialOperation`: the operation to be applied between the queryWindow and the geometries from the input data set
- `queryWindow`: the geometry used to filter the input dataset.

Example: Perform a spatial filtering operation.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.SpatialFilter indexName=indexExample
indexMetadataDir=indexMetadataDir output=/user/hdfs/demo_vector/result
spatialOperation=IsInside queryWindow='{"type":"Polygon", "coordinates":
[[-106, 25, -106, 30, -104, 30, -104, 25, -106, 25]]}'
```

2.13.21 Running a Job to Get Location Suggestions

To create a job to get location suggestions, use a command in the following format.

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.SuggestService [generic options] input=<path|
comma_separated_paths|path_pattern> inputFormat=<InputFormat_subclass>
recordInfoProvider=<RecordInfoProvider_subclass> output=<path> mvsLocation=<path|URL>
```

```
[mvsMatchLayers=<comma_separated_layers>] [mvsMatchCountry=<country_name>]
[mvsSpatialResponse=<[NONE, FEATURE_GEOMETRY, FEATURE_CENTROID]>]
[mvsInterfaceType=<[UNDEFINED, LOCAL, WEB]>] [mvsIsRepository=<true|false>]
[mvsRebuildIndex=<true|false>] [mvsPersistentLocation=<hdfs_path>]
[mvsOverwritePersistentLocation=<true|false>]
[mvsMaxRequestRecords=<integer_number>]
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.SuggestService` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.SuggestService`.

Input/output arguments:

- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression. (Ignored if `indexName` is specified.)
- `inputFormat`: the `inputFormat` class implementation used to read the input data. (Ignored if `indexName` is specified.)
- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class. (Ignored if `indexName` is specified.)
- `output`: the path where the spatial index will be stored

`MVSuggest` arguments:

- `mvsLocation`: The path to the MVSuggest directory or repository for local standalone instances of MVSuggest or the service URL when working with a remote instance. This argument is required when working with MVSuggest.
- `mvsMatchLayers` (optional, default=all): comma separated list of layers. When provided, MVSuggest will only use these layers to perform the search.
- `mvsMatchCountry` (optional, default=none): a country name which MVSuggest will give higher priority when performing matches.
- `mvsSpatialResponse` (optional, default=CENTROID): the type of the spatial results contained in each returned match. It can be one of the following values: NONE, FEATURE_GEOMETRY, FEATURE_CENTROID.
- `mvsInterfaceType` (optional: default=LOCAL): the type of MVSuggest service used, it can be LOCAL or WEB.
- `mvsIsRepository` (optional: default=false) (LOCAL only): Boolean value that specifies whether `mvsLocation` points to a whole MVS directory(false) or only to a repository(true). An MVS repository contains only JSON templates; it may or not contain a `_config_` and `_geonames_` folder.
- `mvsRebuildIndex` (optional, default=false)(LOCAL only):boolean value specifying whether the repository index should be rebuilt or not.
- `mvsPersistentLocation` (optional, default=none)(LOCAL only): an HDFS path where the MVSuggest directory will be saved.
- `mvsIsOverwritePersistentLocation` (optional, default=false): boolean argument that indicates whether an existing `mvsPersistentLocation` must be overwritten in case it already exists.

Example: Get suggestions based on location texts from the input data set.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job.SuggestService input="/user/hdfs/
demo_vector/tweets/part*" inputFormat=<InputFormat_subclass>
recordInfoProvider=<LocalizableRecordInfoProvider_subclass> output=/user/
hdfs/demo_vector/tweets/suggest_res mvsLocation=file:///mvs_dir
mvsMatchLayers=world_continents,world_countries,world_states_provinces
```

2.13.22 Running a Job to Perform a Spatial Join

To perform a spatial join operation on two data sets, use a command in the following format.

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job. SpatialJoin [generic options]
inputList={
 {
  ( indexName=<dataset1_spatial_index_name>
indexMetadataDir=<dataset1_spatial_index_metadata_dir_path> )
  |
  ( input=<dataset1_path|comma_separated_paths|path_pattern>
inputFormat=<dataset1_InputFormat_subclass>
recordInfoProvider=<dataset1_RecordInfoProvider_subclass> )
  [boundaries=<min_x,min_y,max_x,max_y>]
 }
 {
  (indexName=<dataset2_spatial_index_name>
indexMetadataDir=<dataset2_spatial_index_metadata_dir_path>
  )
  |
  ( input=<dataset2_path|comma_separated_paths|path_pattern>
inputFormat=<dataset2_InputFormat_subclass>
recordInfoProvider=<dataset2_RecordInfoProvider_subclass>
  )
  [boundaries=<min_x,min_y,max_x,max_y>]
 }
} output=<path>[srid=<integer_value>] [geodetic=<true|false>]
[tolerance=<double_value>] boundaries=<min_x,min_y,max_x,max_y>
spatialOperation=<AnyInteract|IsInside|WithinDistance> [distance=<double_value>]
[samplingRatio=<decimal_value_between_0_and_1> | partitioningResult=<path>]
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.SpatialJoin` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.SpatialJoin`.

`InputList`: A list of two input data sets. The list is enclosed by curly braces ({}). Each list element is an input data set, which is enclosed by curly braces. An input data set can contain the following information, depending on whether the data set is specified as a spatial index.

If specified as a spatial index:

- `indexName`: the name of an existing spatial index.
- `indexMetadataDir` : the directory where the spatial index metadata is located

If not specified as a spatial index:

- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression. (Ignored if `indexName` is specified.)

- inputFormat: the inputFormat class implementation used to read the input data. (Ignored if indexName is specified.)
- recordInfoProvider: the recordInfoProvider implementation used to extract information from the records read by the InputFormat class. (Ignored if indexName is specified.)

output: the path where the results will be stored

Spatial arguments:

- srid (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- geodetic (optional, default depends on the srid): boolean value that indicates whether the geometries are geodetic or not.
- tolerance (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.
- boundaries (optional, default=unbounded): the minimum and maximum values for each dimension, expressed as comma separated values in the form: minX,minY,maxX,maxY
- spatialOperation: the spatial operation to perform between the input data set and the hierarchical data set. Allowed values are IsInside and AnyInteract.
- distance: distance used for WithinDistance operations.

Partitioning arguments:

- samplingRatio (optional, default=0.1): ratio used to sample the data sets when partitioning needs to be performed
- partitioningResult (optional, default=none): Path to a previously generated partitioning result file

Example: Perform a spatial join on two data sets.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-
vector.jar oracle.spatial.hadoop.vector.mapred.job.SpatialJoin
inputList="{{input=/user/hdfs/demo_vector/world_countries.json
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFor
mat
recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordIn
foProvider} {input=file="/user/hdfs/demo_vector/tweets/part*”
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFor
mat
recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordIn
foProvider}}" output=/user/hdfs/demo_vector/spatial_join srid=8307
spatialOperation=AnyInteract boundaries=-180,-90,180,90
```

2.13.23 Running a Job to Perform Partitioning

To perform a spatial partitioning, use a command in the following format.

```
hadoop jar <HADOOP_LIB_PATH >/sdohadoop-vector.jar
oracle.spatial.hadoop.vector.mapred.job. SpatialJoin [generic options]
inputList={
 {
```

```
  ( indexName=<dataset1_spatial_index_name>
indexMetadataDir=<dataset1_spatial_index_metadata_dir_path> )
  |
  ( input=<dataset1_path|comma_separated_paths|path_pattern>
inputFormat=<dataset1_InputFormat_subclass>
recordInfoProvider=<dataset1_RecordInfoProvider_subclass> )
  [boundaries=<min_x,min_y,max_x,max_y>]
 }
[
 {
  (indexName=<dataset2_spatial_index_name>
indexMetadataDir=<dataset2_spatial_index_metadata_dir_path>
  )
  |
  ( input=<dataset2_path|comma_separated_paths|path_pattern>
inputFormat=<dataset2_InputFormat_subclass>
recordInfoProvider=<dataset2_RecordInfoProvider_subclass>
  )
  [boundaries=<min_x,min_y,max_x,max_y>]
 }
  ……
 {
  (indexName=<datasetN_spatial_index_name>
indexMetadataDir=<datasetN_spatial_index_metadata_dir_path>
  )
  |
  ( input=<datasetN_path|comma_separated_paths|path_pattern>
inputFormat=<datasetN_InputFormat_subclass>
recordInfoProvider=<datasetN_RecordInfoProvider_subclass>
  )
  [boundaries=<min_x,min_y,max_x,max_y>]
 }

}
] output=<path>[srid=<integer_value>] [geodetic=<true|false>]
[tolerance=<double_value>] boundaries=<min_x,min_y,max_x,max_y>
[samplingRatio=<decimal_value_between_0_and_1>]
```

To use the new Hadoop API format, replace `oracle.spatial.hadoop.vector.`*`mapred`*`.job.Partitioning` with `oracle.spatial.hadoop.vector.`**`mapreduce`**`.job.Partitioning`.

`InputList`: A list of two input data sets. The list is enclosed by curly braces ({}). Each list element is an input data set, which is enclosed by curly braces. An input data set can contain the following information, depending on whether the data set is specified as a spatial index.

If specified as a spatial index:

- `indexName`: the name of an existing spatial index.
- `indexMetadataDir` : the directory where the spatial index metadata is located

If not specified as a spatial index:

- `input` : the location of the input data. It can be expressed as a path, a comma separated list of paths, or a regular expression. (Ignored if `indexName` is specified.)
- `inputFormat`: the `inputFormat` class implementation used to read the input data. (Ignored if `indexName` is specified.)

- `recordInfoProvider`: the `recordInfoProvider` implementation used to extract information from the records read by the `InputFormat` class. (Ignored if `indexName` is specified.)

`output`: the path where the results will be stored

Spatial arguments:

- `srid` (optional, default=0): the spatial reference system (coordinate system) ID of the spatial data.
- `geodetic` (optional, default depends on the srid): boolean value that indicates whether the geometries are geodetic or not.
- `tolerance` (optional, default=0.0): double value that represents the tolerance used when performing spatial operations.
- `boundaries` (optional, default=unbounded): the minimum and maximum values for each dimension, expressed as comma separated values in the form: minX,minY,maxX,maxY

Partitioning arguments:

- `samplingRatio` (optional, default=0.1): ratio used to sample the data sets when partitioning needs to be performed

Example: Partition two data sets.

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop/lib/sdohadoop-
vector.jar oracle.spatial.hadoop.vector.mapred.job.Partitioning
inputList="{{input=/user/hdfs/demo_vector/world_countries.json
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFor
mat
recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordIn
foProvider} {input=file="/user/hdfs/demo_vector/tweets/part*”
inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFor
mat
recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordIn
foProvider}}" output=/user/hdfs/demo_vector/partitioning srid=8307
boundaries=-180,-90,180,90
```

2.13.24 Using Multiple Inputs

Multiple input data sets can be specified to a Vector job through the command line interface using the `inputList` parameter. The `inputList` parameter value is a group of input data sets. The `inputList` parameter format is as follows:

```
inputList={ {input_data_set_1_params} {input_data_set_2_params} …
{input_data_set_N_params} }
```

Each individual input data set can be one of the following input data sets:

- Non-file input data set: `inputFormat=<InputFormat_subclass> recordInfoProvider=<RecordInfoProvider_subclass> [srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>] [boundaries=<min_x,min_y,max_x,max_y>]`
- File input data set: `input=<path|comma_separated_paths|path_pattern> inputFormat=<FileInputFormat_subclass>`

recordInfoProvider=<RecordInfoProvider_subclass> [srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>] [boundaries=<min_x,min_y,max_x,max_y>]

- Spatial index input data set: ((indexName=<<indexName>> [indexMetadataDir=<<path>>]) | (isInputIndex=<true> input=<path|comma_separated_paths|path_pattern>)) [srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>] [boundaries=<min_x,min_y,max_x,max_y>]
- NoSQL input data set: kvStore=<kv store name> kvStoreHosts=<comma separated list of hosts> kvParentKey=<parent key> [kvConsistency=<Absolute|NoneRequired|NoneRequiredNoMaster>] [kvBatchSize=<integer value>] [kvDepth=<CHILDREN_ONLY|DESCENDANTS_ONLY|PARENT_AND_CHILDREN|PARENT_AND_DESCENDANTS>] [kvFormatterClass=<fully qualified class name>] [kvSecurity=<properties file path>] [kvTimeOut=<long value>] [kvDefaultEntryProcessor=<fully qualified class name>] [kvEntryGrouper=<fully qualified class name>] [kvResultEntries={ { minor key 1: a minor key name relative to the major key [fully qualified class name: a subclass of NoSQLEntryProcessor class used to process the entry with the given key] } * }] [srid=<integer_value>] [geodetic=<true|false>] [tolerance=<double_value>] [boundaries=<min_x,min_y,max_x,max_y>]

Notes:

- A Categorization job does not support multiple input data sets.
- A SpatialJoin job only supports two input data sets.
- A SpatialIndexing job does not accept input data sets of type spatial index.
- NoSQL input data sets can only be used when kvstore.jar is present in the classpath.

2.13.25 Loading Images from the Local Server to the HDFS Hadoop Cluster

1. Open the console: http://<oracle_big_data_spatial_vector_console>:8045.
2. Click the **Raster** tab.
3. Click **Select File or Path** and browse to the demo folder that contains a set of Hawaii images (/opt/shareddir/spatial/data/rasters).
4. By default, Spark is selected. If you want to use Hadoop, click the **Use Spar**k button to change it to **Use Hadoop**.
5. Select the rasters folder and click **Load images**.

   You will receive a message about the job being accepted, with a tracking URL. You can track the job status using that URL.

   After the job finishes, you can see the uploaded images in the globe in the Viewer tab.

Note:

If you cannot find the raster files, you can copy them to the shared directory folder created during the installation: check the Admin tab for the directory location, then copy the raster files into it.

If you receive an error, check the Raster Configuration details. If GDAL native library is not set-up correctly, much of the raster functionality of the web application will not work.

2.13.26 Visualizing Rasters in the Globe

Before you can visualize the rasters in the globe, you must upload the raster files to HDFS, as explained in Loading Images from the Local Server to the HDFS Hadoop Cluster.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045`.
2. Click the **Raster** tab.
3. Click the **Hadoop Viewer** tab.
4. Click **Refresh Footprints** to update the footprints in the globe, and wait until all footprints are displayed on the globe.

   Identical rasters are displayed with a yellow edge

2.13.27 Processing a Raster or Multiple Rasters with the Same MBR

Before you can visualize the rasters in the globe, you must upload the raster files to HDFS, as explained in Loading Images from the Local Server to the HDFS Hadoop Cluster.

Before processing rasters with the same MBR (minimum bounding rectangle), you must upload the raster files to HDFS, as explained in Loading Images from the Local Server to the HDFS Hadoop Cluster, and visualize the rasters, as explained in Visualizing Rasters in the Globe.

1. Right click over a raster. If you select a raster with a red or yellow edge, a tooltip with a list of rasters may appear.
2. Click **Process Rasters with Same MBR**. You can exclude rasters from the process by clicking the X button on the left side of every row. If single raster was select, click Process Image (No Mosaic).

   The Raster Process dialog box is displayed.
3. By default, Spark is selected to process the job. To use Hadoop instead, click **Use Spark** to toggle the button to **Use Hadoop**.
4. In the Raster Process dialog, scroll down and click **Create Mosaic**.

   Wait until the raster processing is finished. The result will displayed in the Result tab.
5. Optionally, download the result by clicking **Download Full Size Image** below the result image.

2.13.28 Creating a Mosaic Directly from the Globe

Before you can create the mosaic image, you must upload the raster files to HDFS, as explained in Loading Images from the Local Server to the HDFS Hadoop Cluster.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045`.
2. Click the **Raster** tab.
3. Click the **Hadoop Viewer** tab.
4. Click **Refresh Footprints** to update the footprints in the globe, and wait until all footprints are displayed on the globe.

   Identical rasters are displayed with a yellow edge
5. Click **Select and crop coordinates of Footprints**.
6. Draw a rectangle that wraps the rasters (at least one) and desired area, zooming in or out as necessary.
7. Right-click on the map and select **Generate Mosaic**.

   The raster process dialog is displayed.
8. By default, Spark is select to process the job. To use Hadoop instead, click **Use Spark** to toggle the button to **Use Hadoop**.
9. In the raster process dialog, scroll down and click **Create Mosaic**.

   Wait until the raster processing is finished. The result will displayed in the Result tab.
10. Optionally, download the result by clicking **Download Full Size Image**.

> **Note:**
>
> Spark raster processing does not yet support all the options provided for Hadoop raster processing. For Spark raster processing, you must specify additional configuration parameters in the Spark Configuration section of the Admin tab:
>
> - `spark.driver.extraClassPath, spark.executor.extraClassPath`: Specify your hive library installation using these keys. Example: `/usr/lib/hive/lib/*`
> - `spark.kryoserializer.buffer.max`: Enter a value to support the kryo serialization. Example: `160m`

2.13.29 Adding Operations for Raster Processing

Before you add algebra operations for raster processing or image mosaic creation, follow the instructions in Processing a Raster or Multiple Rasters with the Same MBR until you have the raster processing dialog displayed. Before clicking Create Mosaic, perform these steps:

1. Click **Advanced options**.

   A group of new elements is displayed for adding add the advanced options.
2. Scroll down until you see the raster operations.

3. Choose a raster operation from the list. If you want to add a complex operation, toggle the **Hide Complex Operations** checkbox.

   Only one complex operation is allowed per raster processing action.

4. After you select an operation from the list on the left, add it to the process by clicking the right arrow.

   Some operations also require parameters.

5. Add more operations if you want.

   To remove an operation, select it in the list on the right and click the left arrow. You can also remove all operations in the list.

6. By default, Spark is selected to process the job. To use Hadoop instead, click **Use Spark** to toggle the button to **Use Hadoop**.

7. Click **Create Mosaic**.

   Wait until the raster processing is finished. The result will displayed in the Result tab.

8. Optionally, download the result by clicking **Download Full Size Image**.

> **Note:**
>
> For some raster process operations using spark, you need to supply memory details to the spark drivers and executors, with the details depending of the size and details of the rasters in the process. For Spark raster processing, you must specify additional configuration parameters in the Spark Configuration section of the Admin tab:
>
> - `spark.driver.extraClassPath, spark.executor.extraClassPath`: Specify your hive library installation using these keys. Example: `/usr/lib/hive/lib/*`
> - `spark.kryoserializer.buffer.max`: Enter a value to support the kryo serialization. Example: `160m`

2.13.30 Creating a Slope Image from the Globe

Before you can create the mosaic image, you must upload the raster files to HDFS, as explained in Loading Images from the Local Server to the HDFS Hadoop Cluster.

1. Open the console: `http://<oracle_big_data_spatial_vector_console>:8045`.
2. Click the **Raster** tab.
3. Click the **Hadoop Viewer** tab.
4. Click **Refresh Footprints** to update the footprints in the globe, and wait until all footprints are displayed on the globe.

   Identical rasters are displayed with a yellow edge

5. Click **Select and crop coordinates of Footprints**.
6. Draw a rectangle that wraps the rasters (at least one) and desired area, zooming in or out as necessary.

7. Right-click on the map and select **Generate Mosaic**.

   The raster process dialog is displayed.

8. By default, Spark is select to process the job. To use Hadoop instead, click **Use Spark** to toggle the button to **Use Hadoop**.

9. Select the appropriate **Pixel Type**

   Usually these images are Float 32 Bits.

10. Click **Advanced Options**.

    You will see a group of new elements to add as advanced options.

11. Scroll down until you see the Process Classes controls.

12. Specify the **Fully Qualified Class Name**, then click **Add**.

    The framework provides a default process class for slope: `oracle.spatial.hadoop.imageprocessor.process.ImageSlope`

13. Click **Create Mosaic**

    Wait until the raster processing is finished.

    The result will be displayed in the Result tab.

> **Note:**
>
> Spark raster processing does not yet support custom process classes.

2.13.31 Changing the Image File Format from the Globe

Before you can change the image file format, follow the instructions in Processing a Raster or Multiple Rasters with the Same MBR until you have the raster processing dialog displayed. Before clicking Create Mosaic, perform these steps:

1. Select the desired image **Output Format**.

2. By default, Spark is select to process the job. To use Hadoop instead, click **Use Spark** to toggle the button to **Use Hadoop**.

3. Scroll down and click **Create Mosaic**.

   Wait until the raster processing is finished. The result will be displayed in the Result tab.

4. Optionally, download the result by clicking **Download Full Size Image**.

3 Integrating Big Data Spatial and Graph with Oracle Database

You can use Oracle Big Data Connectors to facilitate spatial data access between Big Data Spatial and Graph and Oracle Database.

This chapter assumes that you have a working knowledge of the following:

- Oracle SQL Connector for HDFS

  For information, see Oracle SQL Connector for Hadoop Distributed File System.
- Oracle Loader for Hadoop

  For information, see Oracle Loader for Hadoop
- Apache Hive

  For information, see the Apache Hive documentation at https://cwiki.apache.org/confluence/display/Hive/Home#Home-UserDocumentation.

- Using Oracle SQL Connector for HDFS with Delimited Text Files
  This topic is applicable when the files in HDFS are delimited text files (fields must be delimited using single-character markers, such as commas or tabs) **and** the spatial data is stored as GeoJSON or WKT format.
- Using Oracle SQL Connector for HDFS with Hive Tables
  Oracle SQL Connector for HDFS (OSCH) directly supports HIVE tables defined on HDFS.
- Using Oracle SQL Connector for HDFS with Files Generated by Oracle Loader for Hadoop
  To use Oracle SQL Connector for HDFS (OSCH) with files generated by Oracle Loader for Hadoop (OLH), you must understand how OLH is used to move data from HDFS to Oracle Database.
- Integrating HDFS Spatial Data with Oracle Database Using Oracle Big Data SQL
  You can use Oracle Big Data SQL to facilitate spatial data access between HDFS and Oracle Database.

3.1 Using Oracle SQL Connector for HDFS with Delimited Text Files

This topic is applicable when the files in HDFS are delimited text files (fields must be delimited using single-character markers, such as commas or tabs) **and** the spatial data is stored as GeoJSON or WKT format.

If such data is to be used by Big Data Spatial and Graph and is to be accessed from an Oracle database using the Oracle SQL connection for HDFS, certain configuration steps are needed.

For this example, assume that the files in HDFS contain records separated by new lines, and the fields within each record are separated by tabs, such as in the following:

```
"6703"    1    62    "Hong Kong"    3479846    POINT (114.18306 22.30693)
"6702"    57    166    "Singapore"    1765655    POINT (103.85387 1.29498)
```

1. Log in to a node of the Hadoop cluster.
2. Create the configuration file required by OSCH (Oracle SQL Connector for HDFS), such as the following example:

```
<?xml version="1.0"?>
 <configuration>
    <property>
      <name>oracle.hadoop.exttab.tableName</name>
      <value>TWEETS_EXT_TAB_FILE</value>
    </property>
   <property>
      <name>oracle.hadoop.exttab.sourceType</name>
      <value>text</value>
    </property>
   <property>
      <name>oracle.hadoop.exttab.dataPaths</name>
      <value>/user/scott/simple_tweets_data/*.log</value>
    </property>
    <property>
      <name>oracle.hadoop.connection.url</name>
      <value>jdbc:oracle:thin:@//myhost:1521/myservicename</value>
    </property>
    <property>
      <name>oracle.hadoop.connection.user</name>
      <value>scott</value>
    </property>
    <property>
      <name>oracle.hadoop.exttab.fieldTerminator</name>
      <value>\u0009</value>
    </property>
    <property>
      <name>oracle.hadoop.exttab.columnNames</name>
      <value>ID,FOLLOWERS_COUNT,FRIENDS_COUNT,LOCATION,USER_ID,GEOMETRY</
value>
    </property>
    <property>
      <name>oracle.hadoop.exttab.defaultDirectory</name>
      <value>TWEETS_DT_DIR</value>
    </property>
</configuration>
```

3. Name the configuration file `tweets_text.xml`.
4. On a node of the Hadoop cluster, execute the following command:

```
hadoop jar $OSCH_HOME/jlib/orahdfs.jar \
       oracle.hadoop.exttab.ExternalTable \
       -conf /home/oracle/tweets_text.xml \
       -createTable
```

   The command prompts for the database password .

   You can either create the OSCH_HOME environment variable or replace OSCH_HOME in the command syntax with the full path to the installation directory for Oracle SQL Connector for HDFS. On Oracle Big Data Appliance, this directory is: /opt/oracle/orahdfs-version

The table TWEETS_EXT_TAB_FILE is now ready to query. It can be queried like any other table from the database. The database is the target database specified in the configuration file in a previous step.. The following query selects the count of rows in the table:

```
select count(*) from TWEETS_EXT_TAB_FILE;
```

You can perform spatial operations on that table just like any other spatial table in the database. The following example retrieves information about users that are tweeting within in a quarter-mile (0.25 mile) radius of a specific movie theater:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
0.05, 'UNIT=MILE'), ci.name, tw.user_id
from CINEMA ci, TWEETS_EXT_TAB_FILE tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
'DISTANCE=0.25 UNIT=MILE') = 'TRUE'
```

Here the table CINEMA is a spatial table in the Oracle database, and the HDFS table TWEETS_EXT_TAB_FILE can be used to query against this table. The data from the tweets table is read in as WKT (well known text), and the WKT constructor of SDO_GEOMETRY is used to materialize this data as a geometry in the database.

Note that the SRID of the geometries is 8307. Also ,if the spatial data is in GeoJSON format, then the query should be as follows:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry,
'', 8307), 0.05, 'UNIT=MILE'), ci.name, tw.user_id
from CINEMA ci, TWEETS_EXT_TAB_FILE tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry, '',
8307), 'DISTANCE=0.25 UNIT=MILE') = 'TRUE'
```

3.2 Using Oracle SQL Connector for HDFS with Hive Tables

Oracle SQL Connector for HDFS (OSCH) directly supports HIVE tables defined on HDFS.

The Hive tables must be nonpartitioned, and defined using ROW FORMAT DELIMITED and FILE FORMAT TEXTFILE clauses. The spatial data must be in GeoJSON or WKT format.

Both Hive-managed tables and Hive external tables are supported.

For example, the Hive command to create a table on the file described in Using Oracle SQL Connector for HDFS with Delimited Text Files is as follows. It assumes that the user already has a Hive table defined on HDFS data. The data in HDFS must be in the supported format, and the spatial data must be in GeoJSON or WKT format.

```
CREATE EXTERNAL TABLE IF NOT EXISTS TWEETS_HIVE_TAB(
  ID string,
  FOLLOWERS_COUNT int,
  FRIENDS_COUNT int,
  LOCATION string,
  USER_ID int,
```

```
  GEOMETRY string)
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY '\t'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  '/user/scott/simple_tweets_data';
```

The following example queries the table.

```
select ID, FOLLOWERS_COUNT, FRIENDS_COUNT, LOCATION, USER_ID, GEOMETRY
from TWEETS_HIVE_TAB limit 10;
```

The output looks as follow:

```
"6703"    1    62    "Hong Kong"    3479846    POINT (114.18306 22.30693)
"6702"    57    166    "Singapore"    1765655    POINT (103.85387 1.29498)
```

1. Log in to a node of the Hadoop cluster.
2. Create the configuration file required by OSCH (Oracle SQL Connector for HDFS), such as the following example:

```
<?xml version="1.0"?>
 <configuration>
    <property>
      <name>oracle.hadoop.exttab.tableName</name>
      <value>TWEETS_EXT_TAB_HIVE</value>
    </property>
   <property>
      <name>oracle.hadoop.exttab.sourceType</name>
      <value>hive</value>
    </property>
   <property>
      <name>oracle.hadoop.exttab.hive.tableName</name>
      <value>TWEETS_HIVE_TAB</value>
    </property>
   <property>
      <name>oracle.hadoop.exttab.hive.databaseName</name>
      <value>default</value>
    </property>
    <property>
      <name>oracle.hadoop.connection.url</name>
      <value>jdbc:oracle:thin:@//myhost:1521/myservicename</value>
    </property>
    <property>
      <name>oracle.hadoop.connection.user</name>
      <value>scott</value>
    </property>
    <property>
      <name>oracle.hadoop.exttab.defaultDirectory</name>
      <value>TWEETS_DT_DIR</value>
    </property>
</configuration>
```

3. Name the configuration file `tweets_text.xml`.

4. On a node of the Hadoop cluster, execute the following command:

```
# Add HIVE_HOME/lib* to HADOOP_CLASSPATH.
export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$HIVE_HOME/lib/*
hadoop jar $OSCH_HOME/jlib/orahdfs.jar \
       oracle.hadoop.exttab.ExternalTable \
       -conf /home/oracle/tweets_hive.xml \
       -createTable
```

   The command prompts for the database password. You can either create the OSCH_HOME environment variable or replace OSCH_HOME in the command syntax with the full path to the installation directory for Oracle SQL Connector for HDFS. On Oracle Big Data Appliance, this directory is: `/opt/oracle/orahdfs-version`

   Set the environment variable HIVE_HOME to point to the Hive installation directory (for example, `/usr/lib/hive`).

The table TWEETS_EXT_TAB_FILE is now ready to query. It can be queried like any other table from the database. The following query selects the count of rows in the table:

```
select count(*) from TWEETS_EXT_TAB_HIVE;;
```

You can perform spatial operations on that table just like any other spatial table in the database. The following example retrieves information about users that are tweeting within in a quarter-mile (0.25 mile) radius of a specific movie theater:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
0.05, 'UNIT=MILE'), ci.name, tw.user_id
from CINEMA ci, TWEETS_EXT_TAB_HIVE tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
'DISTANCE=0.25 UNIT=MILE') = 'TRUE'
```

Here the table CINEMA is a spatial table in the Oracle database, and the HDFS table TWEETS_EXT_TAB_FILE can be used to query against this table. The data from the tweets table is read in as WKT (well known text), and the WKT constructor of SDO_GEOMETRY is used to materialize this data as a geometry in the database.

Note that the SRID of the geometries is 8307. Also ,if the spatial data is in GeoJSON format, then the query should be as follows:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry,
'', 8307), 0.05, 'UNIT=MILE'), ci.name, tw.user_id
from CINEMA ci, TWEETS_EXT_TAB_HIVE tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry, '',
8307), 'DISTANCE=0.25 UNIT=MILE') = 'TRUE'
```

3.3 Using Oracle SQL Connector for HDFS with Files Generated by Oracle Loader for Hadoop

To use Oracle SQL Connector for HDFS (OSCH) with files generated by Oracle Loader for Hadoop (OLH), you must understand how OLH is used to move data from HDFS to Oracle Database.

Modifications are required for moving Big Data Spatial and Graph spatial data into the database. This solution generally applies for any kind of files in HDFS or any kind of Hive data. The spatial information can be in a well known format or a custom format.

First, an example of how to create external tables from files in HDFS containing spatial information in a user defined format. Assume that the files in HDFS have records the following format:

```
{
    "type":"Feature",
    "id":"6703",
    "followers_count":1,
    "friends_count":62,
    "location":"Hong Kong",
    "user_id":3479846,
    "longitude":114.18306,
    "latitude":22.30693
}

{
    "type":"Feature",
    "id":"6702",
    "followers_count":57,
    "friends_count":166,
    "location":"Singapore",
    "user_id":1765655,
    "longitude":103.85387,
    "latitude":1.29498
}
```

The Hive command to create a table for those records is as follows:

```
add jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/ojdbc8.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoutl.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoapi.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector-hive.jar
     … (add here jars containing custom SerDe and/or InputFormats);
CREATE EXTERNAL TABLE IF NOT EXISTS CUST_TWEETS_HIVE_TAB (id STRING,
geometry STRING, followers_count STRING, friends_count STRING,
location STRING, user_id
STRING)
ROW FORMAT SERDE 'mypackage.TweetsSerDe'
STORED AS INPUTFORMAT
'oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat'
OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION '/user/scott/simple_tweets_data';
```

The `InputFormat` object `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat` can read those records even if they are not strict GeoJSON. Thus, the preceding example does not need a custom `InputFormat` specification. However, it does require a custom Hive Serializer and Deserializer (SerDe) to transform the latitude and longitude into a WKT or GeoJSON geometry. For that, the Spatial Java API can be used in the deserialize function of the SerDe, as the following example

```
    @Override
    public Object deserialize(Writable w) throws SerDeException {
        Text rowText = (Text) w;
        List<Text> row = new ArrayList<Text>(columnNames.size());

        //default all values to null
        for(int i=0;i<columnNames.size();i++){
            row.add(null);
        }

        // Try parsing row into JSON object
        JsonNode recordNode = null;

        try {
            String txt = rowText.toString().trim();
            recordNode = jsonMapper.readTree(txt);
            row.set(columnNames.indexOf("id"), new 
Text(recordNode.get("id").getTextValue()));
            row.set(columnNames.indexOf("followers_count"), new 
Text(recordNode.get("followers_count").toString()));
            row.set(columnNames.indexOf("friends_count"), new 
Text(recordNode.get("friends_count").toString()));
            row.set(columnNames.indexOf("location"), new 
Text(recordNode.get("location").getTextValue()));
            row.set(columnNames.indexOf("user_id"), new 
Text(recordNode.get("user_id").toString()));

            Double longitude = recordNode.get("longitude").getDoubleValue();
            Double latitude = recordNode.get("latitude").getDoubleValue();

            //use the Spatial API to create the geometry
            JGeometry geom = JGeometry.createPoint(new double[]{
                    longitude,
                    latitude},
                    2, //dimensions
                    8307 //SRID
                    );
            //Transform the JGeometry to WKT
             String geoWKT = new String(wkt.fromJGeometry(geom));
            row.set(columnNames.indexOf("geometry"), new Text(geoWKT));
        } catch (Exception e) {
            throw new SerDeException("Exception parsing JSON: " 
+e.getMessage(), e);
        }

        return row;
    }
```

In the preceding example, to return the geometries in GeoJSON format, replace the following:

```
String geoWKT = new String(wkt.fromJGeometry(geom));
row.set(columnNames.indexOf("geometry"), new Text(geoWKT));
```

with this:

```
row.set(columnNames.indexOf("geometry"), new Text(geom.toGeoJson()));
```

More SerDe examples to transform data in GeoJSON, WKT, or ESRI Shapefiles with the Spatial Java API are available in the folder: `/opt/oracle/oracle-spatial-graph/spatial/vector/examples/hive/java/src/oracle/spatial/hadoop/vector/hive/java/src/serde`

The following example queries the Hive table:

```
select ID, FOLLOWERS_COUNT, FRIENDS_COUNT, LOCATION, USER_ID, GEOMETRY
from CUST_TWEETS_HIVE_TAB limit 10;
```

The output looks like the following:

```
6703    1    62    Hong Kong    3479846    POINT (114.18306 22.30693)
6702    57    166    Singapore    1765655    POINT (103.85387 1.29498)
```

- Creating HDFS Data Pump Files or Delimited Text Files
- Creating the SQL Connector for HDFS

3.3.1 Creating HDFS Data Pump Files or Delimited Text Files

You can use the Hive table from Using Oracle SQL Connector for HDFS with Files Generated by Oracle Loader for Hadoop to create HDFS Data Pump files or delimited text files.

1. Create a table in the Oracle database as follows:

   ```
   CREATE TABLE tweets_t(id INTEGER
     PRIMARY KEY, geometry VARCHAR2(4000), followers_count NUMBER,
     friends_count NUMBER, location VARCHAR2(4000), user_id NUMBER);
   ```

   This table will be used as the target table. Oracle Loader for Hadoop uses table metadata from the Oracle database to identify the column names, data types, partitions, and other information. For simplicity, create this table with the same columns (fields) as the Hive table. After the external table is created, you can remove this table or use it to insert the rows from the external table into the target table. (For more information about target tables, see About the Target Table Metadata.

2. Create the loader configuration file, as in the following example:

   ```
   <?xml version="1.0" encoding="UTF-8" ?>
   <configuration>
   <!--                          Input settings                             -->
   ```

```
<property>
    <name>mapreduce.inputformat.class</name>
    <value>oracle.hadoop.loader.lib.input.HiveToAvroInputFormat</value>
</property>
<property>
    <name>oracle.hadoop.loader.input.hive.databaseName</name>
    <value>default</value>
</property>
<property>
    <name>oracle.hadoop.loader.input.hive.tableName</name>
    <value>CUST_TWEETS_HIVE_TAB</value>
</property>
<!--                          Output settings                             -->
 <property>
   <name>mapreduce.outputformat.class</name>
   <value>oracle.hadoop.loader.lib.output.DataPumpOutputFormat</value>
 </property>
 <property>
   <name>mapred.output.dir</name>
   <value>/user/scott/data_output</value>
 </property>
<!--                          Table information                            -->
<property>
    <name>oracle.hadoop.loader.loaderMap.targetTable</name>
    <value>tweets_t</value>
</property>
<!--                          Connection information                      -->
<property>
  <name>oracle.hadoop.loader.connection.url</name>
  <value>jdbc:oracle:thin:@//myhost:1521/myservicename</value>
</property>
<property>
    <name>oracle.hadoop.loader.connection.user</name>
    <value>scott</value>
</property>
<property>
    <name>oracle.hadoop.loader.connection.password</name>
    <value>thepassword2</value>
    <description> Having the password in cleartext is NOT RECOMMENDED. Use Oracle
Wallet instead. </description>
</property>
</configuration>
```

With this configuration, Data Pump files will be created in HDFS. If you want delimited text files as the output, then replace th following:

```
oracle.hadoop.loader.lib.output.DataPumpOutputFormat
```

with this:

```
oracle.hadoop.loader.lib.output.DelimitedTextOutputFormat
```

3. Name the configuration file `tweets_hive_to_data_pump.xml`.
4. Create the Data Pump files:

```
# Add HIVE_HOME/lib* and the Hive configuration directory to
HADOOP_CLASSPATH.
export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$HIVE_HOME/lib/*:$HIVE_CONF_DIR
# Add Oracle Spatial libraries to HADOOP_CLASSPATH.
export ORACLE_SPATIAL_VECTOR_LIB_PATH=/opt/oracle/oracle-spatial-graph/
```

```
spatial/vector/jlib

export
HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$ORACLE_SPATIAL_VECTOR_LIB_PATH/
ojdbc8.jar:$ORACLE_SPATIAL_VECTOR_LIB_PATH/
sdoutl.jar:$ORACLE_SPATIAL_VECTOR_LIB_PATH/
sdoapi.jar:$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdohadoop-
vector.jar:$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdohadoop-vector-hive.jar

# The Oracle Spatial libraries need to be added to the libjars
option as well.
export LIBJARS=$ORACLE_SPATIAL_VECTOR_LIB_PATH/
ojdbc8.jar,$ORACLE_SPATIAL_VECTOR_LIB_PATH/
sdoutl.jar,$ORACLE_SPATIAL_VECTOR_LIB_PATH/
sdoapi.jar,$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdohadoop-
vector.jar,$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdohadoop-vector-hive.jar

# And the following HIVE jar files have to be added to the libjars
option.
export LIBJARS=$LIBJARS,$HIVE_HOME/lib/hive-exec-
*.jar,$HIVE_HOME/lib/hive-metastore-*.jar,$HIVE_HOME/lib/
libfb303*.jar

hadoop jar ${OLH_HOME}/jlib/oraloader.jar \
           oracle.hadoop.loader.OraLoader \
           -conf /home/oracle/tweets_hive_to_data_pump.xml \
           -libjars $LIBJARS
```

For the preceding example:

- Be sure that the environment variable OLH_HOME has to be set to the installation directory.
- Set the environment variable HIVE_HOME to point to the Hive installation directory (for example, `/usr/lib/hive`).
- Set the environment variable HIVE_CONF_DIR to point to the Hive configuration directory (for example, `/etc/hive/conf`).
- Add the following Hive jar files, in a comma-separated list, to the `-libjars` option of the `hadoop` command. Replace the asterisks (*) with the complete file names on your system:

  ```
  hive-exec-*.jar
  hive-metastore-*.jar
  libfb303*.jar
  ```

- If `oracle.kv.hadoop.hive.table.TableStorageHandler` is used to create the Hive table (with the data coming from Oracle NoSQL Database), you must also add the following jar file to the `-libjars` option of the `hadoop` command: `$KVHOME/lib/kvclient.jar` (where KVHOME is the directory where the Oracle NoSQL Database is installed)
- If `org.apache.hadoop.hive.hbase.HBaseStorageHandler` is used to create the Hive table (with the data coming from Apache HBase), you must also add the following JAR files, in a comma-separated list, to the `-libjars` option of the `hadoop` command:

```
$HIVE_HOME/lib/hbase-server.jar
$HIVE_HOME/lib/hive-hbase-handler.jar
$HIVE_HOME/lib/hbase-common.jar
$HIVE_HOME/lib/hbase-client.jar
$HIVE_HOME/lib/hbase-hadoop2-compat.jar
$HIVE_HOME/lib/hbase-hadoop-compat.jar
$HIVE_HOME/lib/hbase-protocol.jar
$HIVE_HOME/lib/htrace-core.jar
```

3.3.2 Creating the SQL Connector for HDFS

To create the SQL Connector fo HDFS, follow the instructions in this topic.

1. Create the configuration file for the SQL Connector for HDFS), as in the following example:

```
<?xml version="1.0"?>
 <configuration>
    <property>
      <name>oracle.hadoop.exttab.tableName</name>
      <value>TWEETS_EXT_TAB_DP</value>
    </property>
    <property>
      <name>oracle.hadoop.exttab.sourceType</name>
      <value>datapump</value>
    </property>
   <property>
      <name>oracle.hadoop.exttab.dataPaths</name>
      <value>/user/scott/data_output/oraloader-0000*.dat</value>
    </property>
    <property>
      <name>oracle.hadoop.connection.url</name>
      <value>jdbc:oracle:thin:@//myhost:1521/myservicename</value>
    </property>
    <property>
      <name>oracle.hadoop.connection.user</name>
      <value>scott</value>
    </property>
    <property>
      <name>oracle.hadoop.exttab.defaultDirectory</name>
      <value>TWEETS_DT_DIR</value>
    </property>
</configuration>
```

   If the files are delimited text files, follow the steps in Using Oracle SQL Connector for HDFS with Delimited Text Files.

2. Name the configuration file `tweets_ext_from_dp.xml`.

3. Create the external table.

```
hadoop jar $OSCH_HOME/jlib/orahdfs.jar \
           oracle.hadoop.exttab.ExternalTable \
           -conf /home/oracle/tweets_ext_from_dp.xml\
           -createTable
```

   In the preceding command, you can either create the OSCH_HOME environment variable, or replace OSCH_HOME in the command with the full path to the installation directory for Oracle SQL Connector for HDFS. On Oracle Big Data Appliance, this directory is: `/opt/oracle/orahdfs-version`

The table TWEETS_EXT_TAB_DP is now ready to query. It can be queried like any other table in the database. For example:

```
select count(*) from TWEETS_EXT_TAB_DP;
```

You can perform spatial operations on that table, such as the following example to retrieve the users that are tweeting in a quarter-mile radius of a cinema:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry,
8307), 0.5, 'UNIT=YARD'), ci.name, tw.user_id
from CINEMA ci, TWEETS_EXT_TAB_DP tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
'DISTANCE=200 UNIT=MILE') = 'TRUE';
```

This information can be used further to customize advertising.

Note that the SRID of the geometries is 8307. Also, if the spatial data is in GeoJSON format, then the query should be as follows:

```
select sdo_geom.SDO_DISTANCE(ci.geometry,
SDO_UTIL.FROM_GEOJSON(tw.geometry, '', 8307), 0.5, 'UNIT=YARD'),
ci.name, tw.user_id
from CINEMA ci, TWEETS_EXT_TAB_DP tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry,
'', 8307), 'DISTANCE=200 UNIT=MILE') = 'TRUE';
```

3.4 Integrating HDFS Spatial Data with Oracle Database Using Oracle Big Data SQL

You can use Oracle Big Data SQL to facilitate spatial data access between HDFS and Oracle Database.

To enable the spatial features in Oracle Big Data SQL, update the file `bigdata.properties` to add the following lines at the end (replacing $ORACLE_SPATIAL_VECTOR_LIB_PATH with the path to the Oracle Spatial libraries):

```
java.classpath.user=$ORACLE_SPATIAL_VECTOR_LIB_PATH/ojdbc8.jar:
$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdoutl.jar: $ORACLE_SPATIAL_VECTOR_LIB_PATH/
sdoapi.jar:
$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdohadoop-vector.jar:
$ORACLE_SPATIAL_VECTOR_LIB_PATH/sdohadoop-vector-hive.jar
(Also add here jars containing custom SerDe and/or InputFormat specifications.)
```

If the files are in HDFS, you can use the following solutions:

- Creating Oracle External Tables for HDFS Files with Big Data SQL
- Creating Oracle External Tables Using Hive Tables with Big Data SQL

If you are accessing spatial data from Oracle NoSQL Database or Apache HBase, you can use the solution in Creating Oracle External Tables Using Hive Tables with Big Data SQL.

To use Oracle SQL Connector for HDFS (OSCH) with files generated by Oracle Loader for Hadoop (OLH), you must understand how OLH is used to move data from HDFS to Oracle Database.

Modifications are required for moving Big Data Spatial and Graph spatial data into the database. This solution generally applies for any kind of files in HDFS or any kind of Hive data. The spatial information can be in a well known format or a custom format.

First, an example of how to create external tables from files in HDFS containing spatial information in a user defined format. Assume that the files in HDFS have records the following format:

```
{
    "type":"Feature",
    "id":"6703",
    "followers_count":1,
    "friends_count":62,
    "location":"Hong Kong",
    "user_id":3479846,
    "longitude":114.18306,
    "latitude":22.30693
}

{
    "type":"Feature",
    "id":"6702",
    "followers_count":57,
    "friends_count":166,
    "location":"Singapore",
    "user_id":1765655,
    "longitude":103.85387,
    "latitude":1.29498
}
```

The Hive command to create a table for those records is as follows:

```
add jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/ojdbc8.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoutl.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdoapi.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-
vector.jar
     /opt/oracle/oracle-spatial-graph/spatial/vector/jlib/sdohadoop-vector-
hive.jar
     … (add here jars containing custom SerDe and/or InputFormats);
CREATE EXTERNAL TABLE IF NOT EXISTS CUST_TWEETS_HIVE_TAB (id STRING,
geometry STRING, followers_count STRING, friends_count STRING, location
STRING, user_id STRING)
ROW FORMAT SERDE 'mypackage.TweetsSerDe'
STORED AS INPUTFORMAT
'oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION '/user/scott/simple_tweets_data';
```

The `InputFormat` object `oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat` can read those records even if they are not strict GeoJSON. Thus, the preceding example does not need a custom `InputFormat` specification. However, it does require a custom Hive Serializer and Deserializer (SerDe) to transform the latitude and longitude into a WKT or GeoJSON geometry. For that, the Spatial Java API can be used in the deserialize function of the SerDe, as the following example

```
    @Override
    public Object deserialize(Writable w) throws SerDeException {
        Text rowText = (Text) w;
        List<Text> row = new ArrayList<Text>(columnNames.size());

        //default all values to null
        for(int i=0;i<columnNames.size();i++){
            row.add(null);
        }

        // Try parsing row into JSON object
        JsonNode recordNode = null;

        try {
            String txt = rowText.toString().trim();
            recordNode = jsonMapper.readTree(txt);
            row.set(columnNames.indexOf("id"), new
Text(recordNode.get("id").getTextValue()));
            row.set(columnNames.indexOf("followers_count"), new
Text(recordNode.get("followers_count").toString()));
            row.set(columnNames.indexOf("friends_count"), new
Text(recordNode.get("friends_count").toString()));
            row.set(columnNames.indexOf("location"), new
Text(recordNode.get("location").getTextValue()));
            row.set(columnNames.indexOf("user_id"), new
Text(recordNode.get("user_id").toString()));

            Double longitude =
recordNode.get("longitude").getDoubleValue();
            Double latitude =
recordNode.get("latitude").getDoubleValue();

            //use the Spatial API to create the geometry
            JGeometry geom = JGeometry.createPoint(new double[]{
                    longitude,
                    latitude},
                    2, //dimensions
                    8307 //SRID
                    );
            //Transform the JGeometry to WKT
             String geoWKT = new String(wkt.fromJGeometry(geom));
            row.set(columnNames.indexOf("geometry"), new Text(geoWKT));
        } catch (Exception e) {
            throw new SerDeException("Exception parsing JSON: "
+e.getMessage(), e);
        }
```

```
        return row;
    }
```

In the preceding example, to return the geometries in GeoJSON format, replace the following:

```
String geoWKT = new String(wkt.fromJGeometry(geom));
row.set(columnNames.indexOf("geometry"), new Text(geoWKT));
```

with this:

```
row.set(columnNames.indexOf("geometry"), new Text(geom.toGeoJson()));
```

More SerDe examples to transform data in GeoJSON, WKT, or ESRI Shapefiles with the Spatial Java API are available in the folder: `/opt/oracle/oracle-spatial-graph/spatial/vector/examples/hive/java/src/oracle/spatial/hadoop/vector/hive/java/src/serde`

The following example queries the Hive table:

```
select ID, FOLLOWERS_COUNT, FRIENDS_COUNT, LOCATION, USER_ID, GEOMETRY from
CUST_TWEETS_HIVE_TAB limit 10;
```

The output looks like the following:

```
6703     1     62     Hong Kong     3479846     POINT (114.18306 22.30693)
6702     57     166     Singapore     1765655     POINT (103.85387 1.29498)
```

- Creating Oracle External Tables for HDFS Files with Big Data SQL
- Creating Oracle External Tables Using Hive Tables with Big Data SQL

3.4.1 Creating Oracle External Tables for HDFS Files with Big Data SQL

You can create Oracle external tables for any kind of files in HDFS. The spatial information can be in a well known format or a custom format.

If the geometry format is not WKT or GeoJSON, then use one of the provided SerDe examples in the folder `/opt/oracle/oracle-spatial-graph/spatial/vector/examples/hive/java/src/oracle/spatial/hadoop/vector/hive/java/src/serde`, or create a custom SerDe as in the example in Using Oracle SQL Connector for HDFS with Files Generated by Oracle Loader for Hadoop.

After that, create an Oracle external table, as in the following example:

```
CREATE TABLE SAMPLE_TWEETS (id VARCHAR2(4000),
  geometry VARCHAR2(4000),
  followers_count VARCHAR2(4000),
  friends_count VARCHAR2(4000),
  location VARCHAR2(4000), user_id VARCHAR2(4000)) ORGANIZATION EXTERNAL
             (TYPE oracle_hdfs DEFAULT DIRECTORY DEFAULT_DIR
 ACCESS PARAMETERS (
  com.oracle.bigdata.rowformat: \
     SERDE 'mypackage.TweetsSerDe'
  com.oracle.bigdata.fileformat: \
     INPUTFORMAT
```

```
'oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat' \
     OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat' \
  )
LOCATION ('/user/scott/simple_tweets_data/*.log'));
```

The table SAMPLE_TWEETS is now ready to query. It can be queried like any other table in the database. For example:

```
select count(*) from SAMPLE_TWEETS;
```

You can perform spatial operations on that table, such as the following example to retrieve the users that are tweeting in a quarter-mile radius of a cinema:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry,
8307), 0.5, 'UNIT=YARD'), ci.name, tw.user_id
from CINEMA ci, SAMPLE_TWEETS tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
'DISTANCE=200 UNIT=MILE') = 'TRUE';
```

This information can be used further to customize advertising.

Note that the SRID of the geometries is 8307. Also, if the spatial data is in GeoJSON format, then the query should be as follows:

```
select sdo_geom.SDO_DISTANCE(ci.geometry,
SDO_UTIL.FROM_GEOJSON(tw.geometry, '', 8307), 0.5, 'UNIT=YARD'),
ci.name, tw.user_id
from CINEMA ci, SAMPLE_TWEETS tw where
SDO_WITHIN_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry,
'', 8307), 'DISTANCE=200 UNIT=MILE') = 'TRUE';
```

3.4.2 Creating Oracle External Tables Using Hive Tables with Big Data SQL

You can create Oracle external tables using Hive tables with Big Data SQL. The spatial information can be in a well known format or a custom format.

A Hive table used to create an Oracle external table must be created as described in Using Oracle SQL Connector for HDFS with Files Generated by Oracle Loader for Hadoop.

Create an Oracle external table that can be created using the Hive table. For example:

```
CREATE TABLE SAMPLE_TWEETS (id VARCHAR2(4000),  geometry
VARCHAR2(4000),  followers_count VARCHAR2(4000),  friends_count
VARCHAR2(4000),  location VARCHAR2(4000), user_id VARCHAR2(4000))
ORGANIZATION EXTERNAL
(TYPE ORACLE_HIVE
 DEFAULT DIRECTORY DEFAULT_DIR
 ACCESS PARAMETERS (
com.oracle.bigdata.cluster=cluster
```

```
com.oracle.bigdata.tablename=default.CUST_TWEETS_HIVE_TAB)
) PARALLEL 2 REJECT LIMIT UNLIMITED;
```

The table SAMPLE_TWEETS is now ready to query. It can be queried like any other table in the database. For example:

```
select count(*) from SAMPLE_TWEETS;
```

You can perform spatial operations on that table, such as the following example to retrieve the users that are tweeting in a quarter-mile radius of a cinema:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_GEOMETRY(tw.geometry, 8307),
0.5, 'UNIT=YARD'), ci.name, tw.user_id
from CINEMA ci, SAMPLE_TWEETS tw where SDO_WITHIN_DISTANCE(ci.geometry,
SDO_GEOMETRY(tw.geometry, 8307), 'DISTANCE=200 UNIT=MILE') = 'TRUE';
```

This information can be used further to customize advertising.

Note that the SRID of the geometries is 8307. Also, if the spatial data is in GeoJSON format, then the query should be as follows:

```
select sdo_geom.SDO_DISTANCE(ci.geometry, SDO_UTIL.FROM_GEOJSON(tw.geometry,
'', 8307), 0.5, 'UNIT=YARD'), ci.name, tw.user_id
from CINEMA ci, SAMPLE_TWEETS tw where SDO_WITHIN_DISTANCE(ci.geometry,
SDO_UTIL.FROM_GEOJSON(tw.geometry, '', 8307), 'DISTANCE=200 UNIT=MILE') =
'TRUE';
```

4

Using Property Graphs in a Big Data Environment

The property graph analytics feature is installed and configured using the `oracle-graph-hdfs-connector-<ver>.zip` that is available with **Oracle Graph Server and Client** downloads.

You can download Oracle Graph Server and Client from Oracle Software Delivery Cloud. Also, it is important to note that it requires the use of a database for authentication of users connecting to the graph server (PGX).

For more information about creating, storing, and working with property graph data in a Big Data environment, see Oracle Database Graph Developer's Guide for Property Graph.

A Third-Party Licenses for Bundled Software

Oracle Big Data Spatial and Graph installs several third-party products. This appendix lists information that applies to all Apache licensed code, and it lists license information for the installed third-party products.

The following tables show the Provider, Component(s), Version, and Licensing Information for these products.

Table A-1 Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
Apache	TinkerPop Gremlin Server	3.4.1	Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			(an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. == BSD-style Licenses ==

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			The Apache TinkerPop project bundles the following components under the BSD License: jcabi-log (com.jcabi:jcabi-log:0.14 - http://www.jcabi.com/jcabi-log) - for details, see licenses/jcabi-log jcabi-manifests 1.1 (com.jcabi:jcabi-manifests:1.1 - http://www.jcabi.com/jcabi-manifests) - for details, see licenses/jcabi-manifests JLine (jline:jline:2.12 - https://github.com/jline/jline2) - for details, see licenses/jline2 Kryo (com.esotericsoftware:kryo-shaded:3.0.1 - https://github.com/EsotericSoftware/kryo) - shaded in gremlin-shaded to org.apache.tinkerpop.shaded.kryo - for details, see licenses/kryo minlog (com.esotericsoftware:minlog:1.3.0 - https://github.com/EsotericSoftware/minlog) - shaded in gremlin-shaded to org.apache.tinkerpop.shaded.minlog - for details, see licenses/minlog == MIT Licenses == The Apache TinkerPop project bundles the following components under the MIT License: JCL 1.1.1 implemented over SLF4J (org.slf4j:jcl-over-slf4j:1.7.21 - http://www.slf4j.org) - for details, see licenses/slf4j SLF4J API Module (org.slf4j:slf4j-api:1.7.21 - http://www.slf4j.org) - for details, see licenses/slf4j SLF4J LOG4J-12 Binding (org.slf4j:slf4j-log4j12:1.7.21 - http://www.slf4j.org) - for details, see licenses/slf4j Foundation stylesheet for CodeRay (http://foundation.zurb.com) - for details, see licenses/foundation normalize.css 2.1.2 (http://necolas.github.io/normalize.css/) - for details, see licenses/normalize == Other Licenses == The Apache TinkerPop project bundles the following components under the ISC License: jBCrypt (org.mindrot:jbcrypt:0.4 - https://github.com/

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			djmdjm/jBCrypt) - for details, see licenses/jbcrypt Apache TinkerPop (incubating) Copyright 2015-2017 The Apache Software Foundation. ISC License Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. ______ This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- Apache Commons Lang 3.3.1/2.6 (AL ASF) --- This product includes software from the Spring Framework, under the Apache License 2.0 (see: StringUtils.containsWhitespace()) --- Apache Groovy 2.4.7 (AL ASF) --- This product includes/uses ANTLR (http://www.antlr2.org/) developed by Terence Parr 1989-2006 --- Apache Ivy 2.3.0 (AL ASF) --- Portions of Ivy were originally developed by Jayasoft SARL (http://www.jayasoft.fr/) and are licensed to the Apache Software Foundation under the "Software Grant License Agreement" SSH and SFTP support is provided by the JCraft JSch package,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Componen t(s)	Version	Licensing Information
			which is open source software, available under the terms of a BSD style license. The original software and related information is available at http://www.jcraft.com/jsch/. --- JavaTuples 1.2 --- Copyright (c) 2010, The JAVATUPLES team (http:// www.javatuples.org) --- Metrics 3.0.2 --- Copyright 2010-2013 Coda Hale and Yammer, Inc., 2014-2015 Dropwizard Team This product includes software developed by Coda Hale and Yammer, Inc. This product includes code derived from the JSR-166 project (ThreadLocalRandom, Striped64, LongAdder), which was released with the following comments: Written by Doug Lea with assistance from members of JCP JSR-166 Expert Group and released to the public domain, as explained at http://creativecommons.org/publicdomain/zero/1.0/ --- Netty 4.0.34 --- Copyright 2014 The Netty Project foundation Copyright (c) 2013-2016 ZURB, inc. MIT License Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. jbcrypt jBCrypt is subject to the following license: /* * Copyright (c) 2006 Damien Miller * * Permission to use, copy, modify, and distribute this software for any * purpose with or without fee is hereby granted, provided that the above * copyright notice and this permission notice appear in all copies. * * THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES * WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF * MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR * ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES * WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN * ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF * OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			SOFTWARE. */ jcabi-log Copyright (c) 2012-2015, jcabi.com All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3) Neither the name of the jcabi.com nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. jcabi-manifests Copyright (c) 2012-2015, jcabi.com All rights reserved. Redistribution and use in source and binary forms, with or

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Componen t(s)	Version	Licensing Information
			without modification, are permitted provided that the following conditions are met: 1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3) Neither the name of the jcabi.com nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. jline2 Copyright (c) 2002-2012, the original author or authors. All rights reserved. http://www.opensource.org/licenses/bsd-license.php Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of JLine nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. kryo Copyright (c) 2008, Nathan Sweet All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of Esoteric Software nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. minlog Copyright (c) 2008, Nathan Sweet All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of Esoteric Software nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. normalize Copyright (c) Nicolas Gallagher and Jonathan Neal Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. slf4j Copyright (c) 2004-2013 QOS.ch All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. https://github.com/square/javapoet/blob/master/README.md

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			Copyright 2015 Square, Inc. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Componen t(s)	Version	Licensing Information
JS Foundation and other contributors	lodash	4.17.11	The MIT License Copyright JS Foundation and other contributors / js.foundation/> Based on Underscore.js, copyright Jeremy Ashkenas, DocumentCloud and Investigative Reporters & Editors / underscorejs.org/> This software consists of voluntary contributions made by many individuals. For exact contribution history, see the revision history available at https://github.com/lodash/lodash The following license applies to all parts of this software except as documented below: ==== Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			==== Copyright and related rights for sample code are waived via CC0. Sample code is defined as all source code displayed within the prose of the documentation. CC0: http://creativecommons.org/publicdomain/zero/1.0/ ==== Files located in the node_modules and vendor directories are externally maintained libraries used by this software which have their own licenses; we recommend you read them, as their terms may differ from the terms above.

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
Kirollos Risk	fuse.js	3.4.4	fuse.js v. 3.2.1 Source URl: https://github.com/krisk/Fuse/tree/v3.2.1 license URL: https://github.com/krisk/Fuse/blob/v3.2.1/ LICENSE Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "{}" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright 2017 Kirollos Risk Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Copyright URL: https://github.com/krisk/Fuse/blob/v3.2.1/COPYRIGHT.txt

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			Fuse.js {VERSION} - Lightweight fuzzy-search ({HOMEPAGE}) Copyright (c) 2012-2017 Kirollos Risk ({AUTHOR_URL}) All Rights Reserved. Apache Software License 2.0 http://www.apache.org/licenses/LICENSE-2.0 (NO 4th party dependencies)
Mike Bostock	D3 (d3.js)	5.9.0	Copyright 2010-2017 Mike Bostock All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of the author nor the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
Petka Antonov	bluebird	3.5.4	Bluebird 3.5.1 (no 4th party dependencies) Source URL: https://github.com/petkaantonov/bluebird/tree/v3.5.1 MIT License URL: https://github.com/petkaantonov/bluebird/blob/v3.5.1/LICENSE The MIT License (MIT) Copyright (c) 2013-2017 Petka Antonov Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
Tim Wood, Iskren Chernev, Moment.js contributors	moment.js	2.24.0	Copyright (c) JS Foundation and other contributors Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
Tinkerpop	Blueprints	2.6	Copyright (c) 2009-Infinity, TinkerPop [http://tinkerpop.com] All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of the TinkerPop nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL TINKERPOP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use,

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: You must give any other recipients of the Work or Derivative Works a copy of this License; and You must cause any modified files to carry prominent notices stating that You changed the files; and You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and If the Work includes a "NOTICE" text file as part of its

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal

Table A-1 (Cont.) Third-Party License Information (Part 1)

Provider	Component(s)	Version	Licensing Information
			theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS

Table A-2 Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	Apache HC Fluent	4.5.9	Apache HttpComponent Fluent Apache HttpComponents Client Copyright 1999-2018 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). === ======= Licensed to the Apache Software Foundation (ASF) under one or more contributor license agreements. See the NOTICE file distributed with this work for additional information regarding copyright ownership. The ASF licenses this file to you under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. === ======= This project includes Public Suffix List copied from /publicsuffix.org/list/effective_tld_names.dat> licensed under the terms of the Mozilla Public License, v. 2.0 Full license text: /mozilla.org/MPL/2.0/> *** Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS ===

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			This project includes Public Suffix List copied from /publicsuffix.org/list/effective_tld_names.dat> licensed under the terms of the Mozilla Public License, v. 2.0 Full license text: /mozilla.org/MPL/2.0/> Mozilla Public License Version 2.0 ================================== 1. Definitions -------------- 1.1. "Contributor" means each individual or legal entity that creates, contributes to the creation of, or owns Covered Software. 1.2. "Contributor Version" means the combination of the Contributions of others (if any) used by a Contributor and that particular Contributor's Contribution. 1.3. "Contribution" means Covered Software of a particular Contributor. 1.4. "Covered Software" means Source Code Form to which the initial Contributor has attached the notice in Exhibit A, the Executable Form of such Source Code Form, and Modifications of such Source Code Form, in each case including portions thereof. 1.5. "Incompatible With Secondary Licenses" means (a) that the initial Contributor has attached the notice described in Exhibit B to the Covered Software; or (b) that the Covered Software was made available under the terms of version 1.1 or earlier of the License, but not also under the terms of a Secondary License. 1.6. "Executable Form" means any form of the work other than Source Code Form. 1.7. "Larger Work"

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			means a work that combines Covered Software with other material, in a separate file or files, that is not Covered Software. 1.8. "License" means this document. 1.9. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently, any and all of the rights conveyed by this License. 1.10. "Modifications" means any of the following: (a) any file in Source Code Form that results from an addition to, deletion from, or modification of the contents of Covered Software; or (b) any new file in Source Code Form that contains any Covered Software. 1.11. "Patent Claims" of a Contributor means any patent claim(s), including without limitation, method, process, and apparatus claims, in any patent Licensable by such Contributor that would be infringed, but for the grant of the License, by the making, using, selling, offering for sale, having made, import, or transfer of either its Contributions or its Contributor Version. 1.12. "Secondary License" means either the GNU General Public License, Version 2.0, the GNU Lesser General Public License, Version 2.1, the GNU Affero General Public License, Version 3.0, or any later versions of those licenses. 1.13. "Source Code Form" means the form of the work preferred for making modifications. 1.14. "You" (or "Your") means an individual or a legal entity exercising rights under this

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			License. For legal entities, "You" includes any entity that controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity. 2. License Grants and Conditions -------------------------------- 2.1. Grants Each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license: (a) under intellectual property rights (other than patent or trademark) Licensable by such Contributor to use, reproduce, make available, modify, display, perform, distribute, and otherwise exploit its Contributions, either on an unmodified basis, with Modifications, or as part of a Larger Work; and (b) under Patent Claims of such Contributor to make, use, sell, offer for sale, have made, import, and otherwise transfer either its Contributions or its Contributor Version. 2.2. Effective Date The licenses granted in Section 2.1 with respect to any Contribution become effective for each Contribution on the date the Contributor first distributes such Contribution. 2.3. Limitations on Grant Scope The licenses granted in this Section 2 are the only rights granted under this License. No additional rights or licenses will be implied from the distribution or licensing of Covered Software under this License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Notwithstanding Section 2.1(b) above, no patent license is granted by a Contributor: (a) for any code that a Contributor has removed from Covered Software; or (b) for infringements caused by: (i) Your and any other third party's modifications of Covered Software, or (ii) the combination of its Contributions with other software (except as part of its Contributor Version); or (c) under Patent Claims infringed by Covered Software in the absence of its Contributions. This License does not grant any rights in the trademarks, service marks, or logos of any Contributor (except as may be necessary to comply with the notice requirements in Section 3.4). 2.4. Subsequent Licenses No Contributor makes additional grants as a result of Your choice to distribute the Covered Software under a subsequent version of this License (see Section 10.2) or under the terms of a Secondary License (if permitted under the terms of Section 3.3). 2.5. Representation Each Contributor represents that the Contributor believes its Contributions are its original creation(s) or it has sufficient rights to grant the rights to its Contributions conveyed by this License. 2.6. Fair Use This License is not intended to limit any rights You have under applicable copyright doctrines of fair use, fair dealing, or other equivalents. 2.7. Conditions

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Sections 3.1, 3.2, 3.3, and 3.4 are conditions of the licenses granted in Section 2.1. 3. Responsibilities ------------------- 3.1. Distribution of Source Form All distribution of Covered Software in Source Code Form, including any Modifications that You create or to which You contribute, must be under the terms of this License. You must inform recipients that the Source Code Form of the Covered Software is governed by the terms of this License, and how they can obtain a copy of this License. You may not attempt to alter or restrict the recipients' rights in the Source Code Form. 3.2. Distribution of Executable Form If You distribute Covered Software in Executable Form then: (a) such Covered Software must also be made available in Source Code Form, as described in Section 3.1, and You must inform recipients of the Executable Form how they can obtain a copy of such Source Code Form by reasonable means in a timely manner, at a charge no more than the cost of distribution to the recipient; and (b) You may distribute such Executable Form under the terms of this License, or sublicense it under different terms, provided that the license for the Executable Form does not attempt to limit or alter the recipients' rights in the Source Code Form under this License. 3.3. Distribution of a Larger Work You may create and distribute a Larger Work under terms of Your choice, provided that You also comply with the requirements of this License for

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the Covered Software. If the Larger Work is a combination of Covered Software with a work governed by one or more Secondary Licenses, and the Covered Software is not Incompatible With Secondary Licenses, this License permits You to additionally distribute such Covered Software under the terms of such Secondary License(s), so that the recipient of the Larger Work may, at their option, further distribute the Covered Software under the terms of either this License or such Secondary License(s). 3.4. Notices You may not remove or alter the substance of any license notices (including copyright notices, patent notices, disclaimers of warranty, or limitations of liability) contained within the Source Code Form of the Covered Software, except that You may alter any license notices to the extent required to remedy known factual inaccuracies. 3.5. Application of Additional Terms You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, You may do so only on Your own behalf, and not on behalf of any Contributor. You must make it absolutely clear that any such warranty, support, indemnity, or liability obligation is offered by You alone, and You hereby agree to indemnify every Contributor for any liability incurred by such Contributor as a result of warranty, support, indemnity or liability terms You offer. You may include additional disclaimers of warranty and limitations of liability specific to any jurisdiction. 4. Inability to Comply Due to Statute or Regulation ---

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Software due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be placed in a text file included with all distributions of the Covered Software under this License. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it. 5. Termination -------------- 5.1. The rights granted under this License will terminate automatically if You fail to comply with any of its terms. However, if You become compliant, then the rights granted under this License from a particular Contributor are reinstated (a) provisionally, unless and until such Contributor explicitly and finally terminates Your grants, and (b) on an ongoing basis, if such Contributor fails to notify You of the non-compliance by some reasonable means prior to 60 days after You have come back into compliance. Moreover, Your grants from a particular Contributor are reinstated on an ongoing basis if such Contributor notifies You of the non-compliance by some reasonable means, this is the first time You have received notice of non-compliance with this License from such Contributor, and You become compliant prior to 30 days after Your receipt of the notice. 5.2. If You initiate litigation against any entity by asserting a patent infringement claim (excluding declaratory judgment actions, counter-claims, and cross-claims) alleging that a Contributor Version directly or indirectly infringes any patent, then the rights granted to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			You by any and all Contributors for the Covered Software under Section 2.1 of this License shall terminate. 5.3. In the event of termination under Sections 5.1 or 5.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or Your distributors under this License prior to termination shall survive termination. *** * * * 6. Disclaimer of Warranty * * ------------------------- * * * * Covered Software is provided under this License on an "as is" * * basis, without warranty of any kind, either expressed, implied, or * * statutory, including, without limitation, warranties that the * * Covered Software is free of defects, merchantable, fit for a * * particular purpose or non-infringing. The entire risk as to the * * quality and performance of the Covered Software is with You. * * Should any Covered Software prove defective in any respect, You * * (not any Contributor) assume the cost of any necessary servicing, * * repair, or correction. This disclaimer of warranty constitutes an * * essential part of this License. No use of any Covered Software is * * authorized under this License except under this disclaimer. * * * *** *** * * * 7. Limitation of Liability * * -------------------------- * * * * Under no circumstances and under no legal theory, whether tort * * (including negligence), contract, or otherwise, shall any *

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			* Contributor, or anyone who distributes Covered Software as * * permitted above, be liable to You for any direct, indirect, * * special, incidental, or consequential damages of any character * * including, without limitation, damages for lost profits, loss of * * goodwill, work stoppage, computer failure or malfunction, or any * * and all other commercial damages or losses, even if such party * * shall have been informed of the possibility of such damages. This * * limitation of liability shall not apply to liability for death or * * personal injury resulting from such party's negligence to the * * extent applicable law prohibits such limitation. Some * * jurisdictions do not allow the exclusion or limitation of * * incidental or consequential damages, so this exclusion and * * limitation may not apply to You. * * * ** 8. Litigation ------------- Any litigation relating to this License may be brought only in the courts of a jurisdiction where the defendant maintains its principal place of business and such litigation shall be governed by laws of that jurisdiction, without reference to its conflict-of-law provisions. Nothing in this Section shall prevent a party's ability to bring cross-claims or counter-claims. 9. Miscellaneous ---------------- This License represents the complete agreement concerning the subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. Any law or regulation which provides that the language of a contract shall be construed against

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the drafter shall not be used to construe this License against a Contributor. 10. Versions of the License --------------------------- 10.1. New Versions Mozilla Foundation is the license steward. Except as provided in Section 10.3, no one other than the license steward has the right to modify or publish new versions of this License. Each version will be given a distinguishing version number. 10.2. Effect of New Versions You may distribute the Covered Software under the terms of the version of the License under which You originally received the Covered Software, or under the terms of any subsequent version published by the license steward. 10.3. Modified Versions If you create software not governed by this License, and you want to create a new license for such software, you may create and use a modified version of this License if you rename the license and remove any references to the name of the license steward (except to note that such modified license differs from this License). 10.4. Distributing Source Code Form that is Incompatible With Secondary Licenses If You choose to distribute Source Code Form that is Incompatible With Secondary Licenses under the terms of this version of the License, the notice described in Exhibit B of this License must be attached. Exhibit A - Source Code Form License Notice ---

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. If it is not possible or desirable to put the notice in a particular file, then You may include the notice in a location (such as a LICENSE file in a relevant directory) where a recipient would be likely to look for such a notice. You may add additional accurate notices of copyright ownership. Exhibit B - "Incompatible With Secondary Licenses" Notice --- This Source Code Form is "Incompatible With Secondary Licenses", as defined by the Mozilla Public License, v. 2.0. Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner]

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. COPYRIGHT NOTICE ============= Apache Commons Codec Copyright 2002-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). src/test/org/apache/commons/codec/language/DoubleMetaphoneTest.java contains test data from http://aspell.net/test/orig/batch0.tab. Copyright (C) 2002 Kevin Atkinson (kevina@gnu.org) == The content of package org.apache.commons.codec.language.bm has been translated from the original php source code available at http://stevemorse.org/phoneticinfo.htm with permission from the original authors. Original source copyright: Copyright (c) 2008 Alexander Beider & Stephen P. Morse.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	Commons IO	2.6	Apache Commons IO Copyright 2002-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
Apache	Commons Lang	3.9	Apache Commons Lang Copyright 2001-2016 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). This product includes software from the Spring Framework, under the Apache License 2.0 (see: StringUtils.containsWhitespace()) Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			`WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.` `See the License for the specific language governing permissions and`

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	Commons VFS	2.3	Apache Commons VFS Project Copyright 2002-2019 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			permissions and limitations under the License. ------------------------------- org.apache.commons:commons-compress commons-logging:commons-logging Apache Commons Logging Copyright 2003-2016 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/).

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	Groovy	2.4.15	Apache Groovy Copyright 2003-2018 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). This product bundles icons from the famfamfam.com silk icons set http://www.famfamfam.com/lab/icons/silk/ Licensed under the Creative Commons Attribution Licence v2.5 http://creativecommons.org/licenses/by/2.5/ Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s)

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: You must give any other recipients of the Work or Derivative Works a copy of this License; and You must cause any modified files to carry prominent notices stating that You changed the files; and You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work To apply the Apache License to your work, attach the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	HttpComponents HttpClient	4.5.9	--- copyright notice --- Apache HttpComponents Client Copyright 1999-2019 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term --- Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS == This project includes Public Suffix List copied from /publicsuffix.org/list/effective_tld_names.dat> licensed under the terms of the Mozilla Public License, v. 2.0 Full license text: /mozilla.org/MPL/2.0/> Mozilla Public License Version 2.0 ================================== 1. Definitions -------------- 1.1. "Contributor" means each individual or legal entity that creates, contributes to the creation of, or owns Covered Software. 1.2. "Contributor Version" means the combination of the Contributions of others (if any) used by a Contributor and that particular Contributor's Contribution. 1.3. "Contribution" means Covered Software of a particular Contributor. 1.4. "Covered Software" means Source Code Form to which the initial Contributor has attached the notice in Exhibit A, the Executable Form of such Source Code Form, and Modifications of such Source Code Form, in each case including portions thereof.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			1.5. "Incompatible With Secondary Licenses" means (a) that the initial Contributor has attached the notice described in Exhibit B to the Covered Software; or (b) that the Covered Software was made available under the terms of version 1.1 or earlier of the License, but not also under the terms of a Secondary License. 1.6. "Executable Form" means any form of the work other than Source Code Form. 1.7. "Larger Work" means a work that combines Covered Software with other material, in a separate file or files, that is not Covered Software. 1.8. "License" means this document. 1.9. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently, any and all of the rights conveyed by this License. 1.10. "Modifications" means any of the following: (a) any file in Source Code Form that results from an addition to, deletion from, or modification of the contents of Covered Software; or (b) any new file in Source Code Form that contains any Covered Software. 1.11. "Patent Claims" of a Contributor means any patent claim(s), including without limitation, method, process, and apparatus claims, in any patent Licensable by such Contributor that would be infringed, but for the grant of the License, by the making, using, selling, offering for sale, having made, import, or transfer of either its Contributions or its Contributor Version.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			1.12. "Secondary License" means either the GNU General Public License, Version 2.0, the GNU Lesser General Public License, Version 2.1, the GNU Affero General Public License, Version 3.0, or any later versions of those licenses. 1.13. "Source Code Form" means the form of the work preferred for making modifications. 1.14. "You" (or "Your") means an individual or a legal entity exercising rights under this License. For legal entities, "You" includes any entity that controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity. 2. License Grants and Conditions -------------------------------- 2.1. Grants Each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license: (a) under intellectual property rights (other than patent or trademark) Licensable by such Contributor to use, reproduce, make available, modify, display, perform, distribute, and otherwise exploit its Contributions, either on an unmodified basis, with Modifications, or as part of a Larger Work; and (b) under Patent Claims of such Contributor to make, use, sell, offer for sale, have made, import, and otherwise transfer either its Contributions or its Contributor Version. 2.2. Effective Date

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			The licenses granted in Section 2.1 with respect to any Contribution become effective for each Contribution on the date the Contributor first distributes such Contribution. 2.3. Limitations on Grant Scope The licenses granted in this Section 2 are the only rights granted under this License. No additional rights or licenses will be implied from the distribution or licensing of Covered Software under this License. Notwithstanding Section 2.1(b) above, no patent license is granted by a Contributor: (a) for any code that a Contributor has removed from Covered Software; or (b) for infringements caused by: (i) Your and any other third party's modifications of Covered Software, or (ii) the combination of its Contributions with other software (except as part of its Contributor Version); or (c) under Patent Claims infringed by Covered Software in the absence of its Contributions. This License does not grant any rights in the trademarks, service marks, or logos of any Contributor (except as may be necessary to comply with the notice requirements in Section 3.4). 2.4. Subsequent Licenses No Contributor makes additional grants as a result of Your choice to distribute the Covered Software under a subsequent version of this License (see Section 10.2) or under the terms of a Secondary License (if permitted under the terms of Section 3.3). 2.5. Representation

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Each Contributor represents that the Contributor believes its Contributions are its original creation(s) or it has sufficient rights to grant the rights to its Contributions conveyed by this License. 2.6. Fair Use This License is not intended to limit any rights You have under applicable copyright doctrines of fair use, fair dealing, or other equivalents. 2.7. Conditions Sections 3.1, 3.2, 3.3, and 3.4 are conditions of the licenses granted in Section 2.1. 3. Responsibilities ------------------- 3.1. Distribution of Source Form All distribution of Covered Software in Source Code Form, including any Modifications that You create or to which You contribute, must be under the terms of this License. You must inform recipients that the Source Code Form of the Covered Software is governed by the terms of this License, and how they can obtain a copy of this License. You may not attempt to alter or restrict the recipients' rights in the Source Code Form. 3.2. Distribution of Executable Form If You distribute Covered Software in Executable Form then: (a) such Covered Software must also be made available in Source Code Form, as described in Section 3.1, and You must inform recipients of the Executable Form how they can obtain a copy of such Source Code Form by reasonable means in a timely manner, at a charge no more than the cost of distribution to the recipient; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(b) You may distribute such Executable Form under the terms of this License, or sublicense it under different terms, provided that the license for the Executable Form does not attempt to limit or alter the recipients' rights in the Source Code Form under this License. 3.3. Distribution of a Larger Work You may create and distribute a Larger Work under terms of Your choice, provided that You also comply with the requirements of this License for the Covered Software. If the Larger Work is a combination of Covered Software with a work governed by one or more Secondary Licenses, and the Covered Software is not Incompatible With Secondary Licenses, this License permits You to additionally distribute such Covered Software under the terms of such Secondary License(s), so that the recipient of the Larger Work may, at their option, further distribute the Covered Software under the terms of either this License or such Secondary License(s). 3.4. Notices You may not remove or alter the substance of any license notices (including copyright notices, patent notices, disclaimers of warranty, or limitations of liability) contained within the Source Code Form of the Covered Software, except that You may alter any license notices to the extent required to remedy known factual inaccuracies. 3.5. Application of Additional Terms You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, You may do so only on Your own behalf, and not on behalf of any Contributor. You must make it absolutely clear that any

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			such warranty, support, indemnity, or liability obligation is offered by You alone, and You hereby agree to indemnify every Contributor for any liability incurred by such Contributor as a result of warranty, support, indemnity or liability terms You offer. You may include additional disclaimers of warranty and limitations of liability specific to any jurisdiction. 4. Inability to Comply Due to Statute or Regulation --- If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Software due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be placed in a text file included with all distributions of the Covered Software under this License. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it. 5. Termination -------------- 5.1. The rights granted under this License will terminate automatically if You fail to comply with any of its terms. However, if You become compliant, then the rights granted under this License from a particular Contributor are reinstated (a) provisionally, unless and until such Contributor explicitly and finally terminates Your grants, and (b) on an ongoing basis, if such Contributor fails to notify You of the non-compliance by some reasonable means prior to 60 days after You have come back into compliance. Moreover, Your grants from a particular Contributor are reinstated on an ongoing basis if such Contributor

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			notifies You of the non-compliance by some reasonable means, this is the first time You have received notice of non-compliance with this License from such Contributor, and You become compliant prior to 30 days after Your receipt of the notice. 5.2. If You initiate litigation against any entity by asserting a patent infringement claim (excluding declaratory judgment actions, counter-claims, and cross-claims) alleging that a Contributor Version directly or indirectly infringes any patent, then the rights granted to You by any and all Contributors for the Covered Software under Section 2.1 of this License shall terminate. 5.3. In the event of termination under Sections 5.1 or 5.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or Your distributors under this License prior to termination shall survive termination. *** * * * 6. Disclaimer of Warranty * * ------------------------- * * * * Covered Software is provided under this License on an "as is" * * basis, without warranty of any kind, either expressed, implied, or * * statutory, including, without limitation, warranties that the * * Covered Software is free of defects, merchantable, fit for a * * particular purpose or non-infringing. The entire risk as to the * * quality and performance of the Covered Software is with You. * * Should any Covered Software prove defective in any respect, You * * (not any Contributor) assume the cost of any necessary servicing, * * repair, or correction. This disclaimer of warranty constitutes an * * essential part of this License. No use of any Covered Software is *

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			* authorized under this License except under this disclaimer. * * * *** *** * * * 7. Limitation of Liability * * -------------------------- * * * * Under no circumstances and under no legal theory, whether tort * * (including negligence), contract, or otherwise, shall any * * Contributor, or anyone who distributes Covered Software as * * permitted above, be liable to You for any direct, indirect, * * special, incidental, or consequential damages of any character * * including, without limitation, damages for lost profits, loss of * * goodwill, work stoppage, computer failure or malfunction, or any * * and all other commercial damages or losses, even if such party * * shall have been informed of the possibility of such damages. This * * limitation of liability shall not apply to liability for death or * * personal injury resulting from such party's negligence to the * * extent applicable law prohibits such limitation. Some * * jurisdictions do not allow the exclusion or limitation of * * incidental or consequential damages, so this exclusion and * * limitation may not apply to You. * * * *** 8. Litigation ------------- Any litigation relating to this License may be brought only in the courts of a jurisdiction where the defendant maintains its principal place of business and such litigation shall be governed by laws of that jurisdiction, without reference to its conflict-of-law provisions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Nothing in this Section shall prevent a party's ability to bring cross-claims or counter-claims. 9. Miscellaneous ---------------- This License represents the complete agreement concerning the subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not be used to construe this License against a Contributor. 10. Versions of the License --------------------------- 10.1. New Versions Mozilla Foundation is the license steward. Except as provided in Section 10.3, no one other than the license steward has the right to modify or publish new versions of this License. Each version will be given a distinguishing version number. 10.2. Effect of New Versions You may distribute the Covered Software under the terms of the version of the License under which You originally received the Covered Software, or under the terms of any subsequent version published by the license steward. 10.3. Modified Versions If you create software not governed by this License, and you want to create a new license for such software, you may create and use a modified version of this License if you rename the license and remove any references to the name of the license steward (except to note that such modified license differs from this License).

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			10.4. Distributing Source Code Form that is Incompatible With Secondary Licenses If You choose to distribute Source Code Form that is Incompatible With Secondary Licenses under the terms of this version of the License, the notice described in Exhibit B of this License must be attached. Exhibit A - Source Code Form License Notice --- This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at http://mozilla.org/MPL/2.0/. If it is not possible or desirable to put the notice in a particular file, then You may include the notice in a location (such as a LICENSE file in a relevant directory) where a recipient would be likely to look for such a notice. You may add additional accurate notices of copyright ownership. Exhibit B - "Incompatible With Secondary Licenses" Notice --- This Source Code Form is "Incompatible With Secondary Licenses", as defined by the Mozilla Public License, v. 2.0.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
Apache	HttpCore	4.4.11	--- copyright notice of 4-th dependency #1 - httpcore 4.4.11 --- Apache HttpComponents Core Copyright 2005-2019 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #1 - httpcore 4.4.11 --- LICENSE: Apache 2.0 https://github.com/apache/httpcomponents-core/blob/4.4.11/LICENSE.txt --- copyright notice of 4-th dependency #2 - commons-logging 1.2 --- Apache Commons Logging Copyright 2003-2014 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #2 - commons-logging 1.2 --- LICENSE: Apache 2.0 https://github.com/apache/commons-logging/blob/LOGGING_1_2/LICENSE.txt --- copyright term 4-th dependency #3 - commons-codec 1.12 --- Apache Commons Codec Copyright 2002-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). src/test/org/apache/commons/codec/language/DoubleMetaphoneTest.java contains test data from http://aspell.net/test/orig/batch0.tab. Copyright (C) 2002 Kevin Atkinson (kevina@gnu.org) === The content of package org.apache.commons.codec.language.bm has been translated from the original php source code available at http://stevemorse.org/phoneticinfo.htm with permission from the original authors. Original source copyright: Copyright (c) 2008 Alexander Beider & Stephen P. Morse. --- copyright term 4-th dependency #3 - commons-codec 1.12 --- LICENSE: Apache 2.0 https://github.com/apache/commons-codec/

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			blob/commons-codec-1.12/LICENSE.txt Apache HttpComponents Core Copyright 2005-2019 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ============================ LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	Log4J	2.11.2	log4j 2.11.2 COPYRIGHT: Copyright 1999-2017 Apache Software Foundation LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright 1999-2005 The Apache Software Foundation Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. =====================================

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Modules: log4j-1.2-api log4j-api-java9 log4j-api log4j-appserver log4j-bom log4j-cassandra log4j-core-its log4j-core-java9 log4j-core log4j-couchdb log4j-distribution log4j-flume-ng log4j-iostreams log4j-jcl log4j-jdbc-dbcp2 log4j-jmx-gui log4j-jpa log4j-jul log4j-liquibase log4j-mongodb2 log4j-mongodb3 log4j-osgi log4j-perf log4j-samples log4j-slf4j-impl log4j-slf4j18-impl log4j-taglib log4j-to-slf4j log4j-web ===================================== 4P Dependencies --------------- FROM NOTICE FILE: This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ResolverUtil.java Copyright 2005-2006 Tim Fennell LICENSE: Apache 2.0 http://www.apache.org/licenses/ LICENSE-2.0 Dumbster SMTP test server Copyright 2004 Jason Paul Kitchen LICENSE: Apache 2.0 http://www.apache.org/licenses/ LICENSE-2.0 TypeUtil.java Copyright 2002-2012 Ramnivas Laddad, Juergen Hoeller, Chris

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Beams LICENSE: Apache 2.0 http://www.apache.org/licenses/ LICENSE-2.0 picocli (http://picocli.info) Copyright 2017 Remko Popma LICENSE: Apache 2.0 http://www.apache.org/licenses/ LICENSE-2.0 --------------- FROM POM FILES (Compile Dependencies Only) --- com.lmax disruptor 3.4.2 Copyright 2011 - 2018 LMAX Ltd. LICENSE: Apache 2.0 https://github.com/LMAX-Exchange/ disruptor/blob/3.4.2/LICENCE.txt --- org.springframework spring-aop 3.2.18.RELEASE COPYRIGHT: Copyright 2002-2018 the original author or authors. LICENSE: Apache 2.0 http://www.apache.org/licenses/ LICENSE-2.0 --- slf4j-api 1.7.25 slf4j-ext 1.7.25 COPYRIGHT and LICENSE: Copyright (c) 2004-2017 QOS.ch All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. LICENSE TERMS ==

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	Tomcat	9.0.24	Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. APACHE TOMCAT SUBCOMPONENTS: Apache Tomcat includes a number of subcomponents with

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			separate copyright notices and license terms. Your use of these subcomponents is subject to the terms and conditions of the following licenses. For the Eclipse JDT Core Batch Compiler (ecj-x.x.x.jar) component: Eclipse Public License - v 1.0 THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT. 1. DEFINITIONS "Contribution" means: a) in the case of the initial Contributor, the initial code and documentation distributed under this Agreement, and b) in the case of each subsequent Contributor: i) changes to the Program, and ii) additions to the Program; where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program. "Contributor" means any person or entity that distributes the Program. "Licensed Patents" mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Program" means the Contributions distributed in accordance with this Agreement. "Recipient" means anyone who receives the Program under this Agreement, including all Contributors. 2. GRANT OF RIGHTS a) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form. b) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder. c) Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program. d) Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement. 3. REQUIREMENTS A Contributor may choose to distribute the Program in object code form under its own license agreement, provided that: a) it complies with the terms and conditions of this Agreement; and b) its license agreement: i) effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose; ii) effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; iii) states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and iv) states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange. When the Program is made available in source code form: a) it must be made available under this Agreement; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			b) a copy of this Agreement must be included with each copy of the Program. Contributors may not remove or alter any copyright notices contained within the Program. Each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution. 4. COMMERCIAL DISTRIBUTION Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program, the Contributor who includes the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs (collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement. In order to qualify, an Indemnified Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that Commercial Contributor then makes performance claims, or offers warranties related to Product X, those performance claims and warranties are such Commercial Contributor's responsibility alone. Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay any damages as a result, the Commercial Contributor must pay those damages. 5. NO WARRANTY EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement , including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations. 6. DISCLAIMER OF LIABILITY EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM OR THE EXERCISE OF ANY RIGHTS

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			GRANTED HEREUNDER, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. 7. GENERAL If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement, and without further action by the parties hereto, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable. If Recipient institutes patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Program itself (excluding combinations of the Program with other software or hardware) infringes such Recipient's patent(s), then such Recipient's rights granted under Section 2(b) shall terminate as of the date such litigation is filed. All Recipient's rights under this Agreement shall terminate if it fails to comply with any of the material terms or conditions of this Agreement and does not cure such failure in a reasonable period of time after becoming aware of such noncompliance. If all Recipient's rights under this Agreement terminate, Recipient agrees to cease use and distribution of the Program as soon as reasonably practicable. However, Recipient's obligations under this Agreement and any licenses granted by Recipient relating to the Program shall continue and survive. Everyone is permitted to copy and distribute copies of this Agreement, but in order to avoid inconsistency the Agreement is copyrighted and may only be modified in the following manner. The Agreement Steward reserves the right to publish new versions (including revisions) of this Agreement from time to time. No one other than the Agreement Steward has the right to modify this Agreement. The Eclipse Foundation is the initial Agreement Steward. The

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Eclipse Foundation may assign the responsibility to serve as the Agreement Steward to a suitable separate entity. Each new version of the Agreement will be given a distinguishing version number. The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise. All rights in the Program not expressly granted under this Agreement are reserved. This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation. COPYRIGHT NOTICES === Apache Tomcat Copyright 1999-2019 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (https://www.apache.org/). This software contains code derived from netty-native developed by the Netty project (https://netty.io, https://github.com/netty/netty-tcnative/) and from finagle-native developed at Twitter (https://github.com/twitter/finagle). This software contains code derived from jgroups-kubernetes developed by the JGroups project (http://www.jgroups.org/). The Windows Installer is built with the Nullsoft

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Scriptable Install System (NSIS), which is open source software. The original software and related information is available at http://nsis.sourceforge.net. Java compilation software for JSP pages is provided by the Eclipse JDT Core Batch Compiler component, which is open source software. The original software and related information is available at https://www.eclipse.org/jdt/core/. org.apache.tomcat.util.json.JSONParser.jj is a public domain javacc grammar for JSON written by Robert Fischer. https://github.com/RobertFischer/json-parser For portions of the Tomcat JNI OpenSSL API and the OpenSSL JSSE integration The org.apache.tomcat.jni and the org.apache.tomcat.net.openssl packages are derivative work originating from the Netty project and the finagle-native project developed at Twitter * Copyright 2014 The Netty Project * Copyright 2014 Twitter For portions of the Tomcat cloud support The org.apache.catalina.tribes.membership.cloud package contains derivative work originating from the jgroups project. https://github.com/jgroups-extras/jgroups-kubernetes Copyright 2002-2018 Red Hat Inc. The original XML Schemas for Java EE Deployment Descriptors: - javaee_5.xsd - javaee_web_services_1_2.xsd - javaee_web_services_client_1_2.xsd - javaee_6.xsd - javaee_web_services_1_3.xsd - javaee_web_services_client_1_3.xsd - jsp_2_2.xsd - web-app_3_0.xsd - web-common_3_0.xsd - web-fragment_3_0.xsd - javaee_7.xsd - javaee_web_services_1_4.xsd - javaee_web_services_client_1_4.xsd - jsp_2_3.xsd - web-app_3_1.xsd - web-common_3_1.xsd - web-fragment_3_1.xsd - javaee_8.xsd

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			- web-app_4_0.xsd - web-common_4_0.xsd - web-fragment_4_0.xsd may be obtained from: http://www.oracle.com/webfolder/technetwork/jsc/xml/ns/javaee/index.html

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Apache	log4cxx	0.10.0	Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. APACHE TOMCAT SUBCOMPONENTS: Apache Tomcat includes a number of subcomponents with

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			separate copyright notices and license terms. Your use of these subcomponents is subject to the terms and conditions of the following licenses. For the Eclipse JDT Core Batch Compiler (ecj-x.x.x.jar) component: Eclipse Public License - v 1.0 THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT. 1. DEFINITIONS "Contribution" means: a) in the case of the initial Contributor, the initial code and documentation distributed under this Agreement, and b) in the case of each subsequent Contributor: i) changes to the Program, and ii) additions to the Program; where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program. "Contributor" means any person or entity that distributes the Program. "Licensed Patents" mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Program" means the Contributions distributed in accordance with this Agreement. "Recipient" means anyone who receives the Program under this Agreement, including all Contributors. 2. GRANT OF RIGHTS a) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form. b) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder. c) Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program. d) Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement. 3. REQUIREMENTS A Contributor may choose to distribute the Program in object code form under its own license agreement, provided that: a) it complies with the terms and conditions of this Agreement; and b) its license agreement: i) effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose; ii) effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; iii) states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and iv) states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange. When the Program is made available in source code form: a) it must be made available under this Agreement; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			b) a copy of this Agreement must be included with each copy of the Program. Contributors may not remove or alter any copyright notices contained within the Program. Each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution. 4. COMMERCIAL DISTRIBUTION Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program, the Contributor who includes the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs (collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement. In order to qualify, an Indemnified Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that Commercial Contributor then makes performance claims, or offers warranties related to Product X, those performance claims and warranties are such Commercial Contributor's responsibility alone. Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay any damages as a result, the Commercial Contributor must pay those damages. 5. NO WARRANTY EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement , including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations. 6. DISCLAIMER OF LIABILITY EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM OR THE EXERCISE OF ANY RIGHTS

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			GRANTED HEREUNDER, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. 7. GENERAL If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement, and without further action by the parties hereto, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable. If Recipient institutes patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Program itself (excluding combinations of the Program with other software or hardware) infringes such Recipient's patent(s), then such Recipient's rights granted under Section 2(b) shall terminate as of the date such litigation is filed. All Recipient's rights under this Agreement shall terminate if it fails to comply with any of the material terms or conditions of this Agreement and does not cure such failure in a reasonable period of time after becoming aware of such noncompliance. If all Recipient's rights under this Agreement terminate, Recipient agrees to cease use and distribution of the Program as soon as reasonably practicable. However, Recipient's obligations under this Agreement and any licenses granted by Recipient relating to the Program shall continue and survive. Everyone is permitted to copy and distribute copies of this Agreement, but in order to avoid inconsistency the Agreement is copyrighted and may only be modified in the following manner. The Agreement Steward reserves the right to publish new versions (including revisions) of this Agreement from time to time. No one other than the Agreement Steward has the right to modify this Agreement. The Eclipse Foundation is the initial Agreement Steward. The

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Eclipse Foundation may assign the responsibility to serve as the Agreement Steward to a suitable separate entity. Each new version of the Agreement will be given a distinguishing version number. The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise. All rights in the Program not expressly granted under this Agreement are reserved. This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation. COPYRIGHT NOTICES == Apache Tomcat Copyright 1999-2019 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (https://www.apache.org/). This software contains code derived from netty-native developed by the Netty project (https://netty.io, https://github.com/netty/netty-tcnative/) and from finagle-native developed at Twitter (https://github.com/twitter/finagle). This software contains code derived from jgroups-kubernetes developed by the JGroups project (http://www.jgroups.org/). The Windows Installer is built with the Nullsoft

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Scriptable Install System (NSIS), which is open source software. The original software and related information is available at http://nsis.sourceforge.net. Java compilation software for JSP pages is provided by the Eclipse JDT Core Batch Compiler component, which is open source software. The original software and related information is available at https://www.eclipse.org/jdt/core/. org.apache.tomcat.util.json.JSONParser.jj is a public domain javacc grammar for JSON written by Robert Fischer. https://github.com/RobertFischer/json-parser For portions of the Tomcat JNI OpenSSL API and the OpenSSL JSSE integration The org.apache.tomcat.jni and the org.apache.tomcat.net.openssl packages are derivative work originating from the Netty project and the finagle-native project developed at Twitter * Copyright 2014 The Netty Project * Copyright 2014 Twitter For portions of the Tomcat cloud support The org.apache.catalina.tribes.membership.cloud package contains derivative work originating from the jgroups project. https://github.com/jgroups-extras/jgroups-kubernetes Copyright 2002-2018 Red Hat Inc. The original XML Schemas for Java EE Deployment Descriptors: - javaee_5.xsd - javaee_web_services_1_2.xsd - javaee_web_services_client_1_2.xsd - javaee_6.xsd - javaee_web_services_1_3.xsd - javaee_web_services_client_1_3.xsd - jsp_2_2.xsd - web-app_3_0.xsd - web-common_3_0.xsd - web-fragment_3_0.xsd - javaee_7.xsd - javaee_web_services_1_4.xsd - javaee_web_services_client_1_4.xsd - jsp_2_3.xsd - web-app_3_1.xsd - web-common_3_1.xsd - web-fragment_3_1.xsd - javaee_8.xsd

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			- web-app_4_0.xsd - web-common_4_0.xsd - web-fragment_4_0.xsd may be obtained from: http://www.oracle.com/webfolder/technetwork/jsc/xml/ns/javaee/index.html

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Elasticsearch	Java Low Level REST Client	7.2.0	--- copyright notice --- Elasticsearch Copyright 2009-2018 Elasticsearch This product includes software developed by The Apache Software Foundation (http://www.apache.org/). This product includes software developed by Joda.org (http://www.joda.org/). --- license term --- Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. --- copyright notice of 4-th dependency #1 - HttpClient

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			4.5.8 --- Apache HttpComponents Client Copyright 1999-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #1 - HttpClient 4.5.8 --- LICENSE: Apache 2.0 https://github.com/apache/httpcomponents-client/blob/4.5.4/LICENSE.txt --- copyright notice of 4-th dependency #2 - HttpCore 4.4.11 --- Apache HttpComponents Core Copyright 2005-2018 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #2 - HttpCore 4.4.11 --- LICENSE: Apache 2.0 https://github.com/apache/httpcomponents-core/blob/4.4.10/LICENSE.txt --- copyright notice of 4-th dependency #3 - commons codec 1.11 --- Apache Commons Codec Copyright 2002-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). src/test/org/apache/commons/codec/language/DoubleMetaphoneTest.java contains test data from http://aspell.net/test/orig/batch0.tab. Copyright (C) 2002 Kevin Atkinson (kevina@gnu.org) == The content of package org.apache.commons.codec.language.bm has been translated from the original php source code available at http://stevemorse.org/phoneticinfo.htm with permission from the original authors. Original source copyright: Copyright (c) 2008 Alexander Beider & Stephen P. Morse. --- license term of 4-th dependency #3 - commons codec 1.11 --- LICENSE: Apache 2.0 https://github.com/apache/commons-codec/blob/commons-codec-1.11/LICENSE.txt

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			--- copyright notice of 4-th dependency #4 - commons-logging 1.1.3 --- Apache Commons Logging Copyright 2003-2013 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #4 - commons-logging 1.1.3 --- LICENSE: Apache 2.0 https://github.com/apache/commons-logging/blob/LOGGING_1_1_3/LICENSE.txt --- copyright notice of 4-th dependency #5 - httpasyncclient 4.1.4 --- Apache HttpComponents AsyncClient Copyright 2010-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #5 - httpasyncclient 4.1.4 --- LICENSE: Apache 2.0 https://github.com/apache/httpasyncclient/blob/4.1.4/LICENSE.txt --- copyright notice of 4-th dependency #6 - httpcore-nio 4.4.11 --- Apache HttpComponents Core Copyright 2005-2018 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). --- license term of 4-th dependency #6 - httpcore-nio 4.4.11 --- https://github.com/apache/httpcomponents-core/blob/4.4.10/LICENSE.txt

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
FasterXML, LLC	jackson-annotations	2.9.9	Jackson Annotations Copyright (c) 2019 Tatu Saloranta LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
FasterXML, LLC	jackson-core	2.9.9	Jackson Core Copyright ģ2008Ọ19 FasterXML. All rights reserved. This copy of Jackson JSON processor streaming parser/ generator is licensed under the Apache (Software) License, version 2.0 ("the License"). See the License for details about distribution rights, and the specific rights regarding derivate works. You may obtain a copy of the License at: http://www.apache.org/licenses/LICENSE-2.0 NOTICE FILE: =============== # Jackson JSON processor Jackson is a high-performance, Free/Open Source JSON processing library. It was originally written by Tatu Saloranta (tatu.saloranta@iki.fi), and has been in development since 2007. It is currently developed by a community of developers, as well as supported commercially by FasterXML.com. ## Licensing Jackson core and extension components may licensed under different licenses. To find the details that apply to this artifact see the accompanying LICENSE file. For more information, including possible other licensing options, contact FasterXML.com (http://fasterxml.com). ## Credits A list of contributors may be found from CREDITS file, which is included in some artifacts (usually source distributions); but is always available from the source code management (SCM) system project uses. =============== From the LICENSE file: Apache License Version 2.0, January 2004 http://www.apache.org/licenses/

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
FasterXML, LLC	jackson-databind	2.9.9.3	Jackson Databind Copyright (c) 2019 Tatu Saloranta LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			COPYRIGHT NOTICE # Jackson JSON processor Jackson is a high-performance, Free/Open Source JSON processing library. It was originally written by Tatu Saloranta (tatu.saloranta@iki.fi), and has been in development since 2007. It is currently developed by a community of developers, as well as supported commercially by FasterXML.com. ## Licensing -----------------jackson-core 2.9.9 ----------------------- COPYRIGHT: Copyright (c) 2007- Tatu Saloranta, tatu.saloranta@iki.fi LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Jackson core and extension components may be licensed under different licenses. To find the details that apply to this artifact see the accompanying LICENSE file. For more information, including possible other licensing options, contact FasterXML.com (http://fasterxml.com). ## Credits A list of contributors may be found from CREDITS file, which is included in some artifacts (usually source distributions); but is always available from the source code management (SCM) system project uses. -----------------jackson-annotations 2.9.9 ----------------------- COPYRIGHT: Copyright (c) 2007- Tatu Saloranta, tatu.saloranta@iki.fi LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s)

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, r

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
FasterXML, LLC	jackson-dataformat-yaml	2.9.9	jackson-dataformat-yaml 2.9.9 copyright: [1] license: [3] snakeyaml 1.23 copyright: [2] license: [3] =====BEGIN REFERENCE[1]===== This copy of Jackson JSON processor YAML module is licensed under the Apache (Software) License, version 2.0 ("the License"). See the License for details about distribution rights, and the specific rights regarding derivate works. You may obtain a copy of the License at: http://www.apache.org/licenses/LICENSE-2.0 =====END REFERENCE[1]===== =====BEGIN REFERENCE[2]===== /** * Copyright (c) 2008, http://www.snakeyaml.org * * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License. * You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. */ =====END REFERENCE[2]===== =====BEGIN REFERENCE[3]===== Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. 4TH Party Dependency : snakeyaml license: Apache 2.0 ------------------- Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. =====END REFERENCE[3]=====

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
FasterXML, LLC	jackson-dataformats-text	2.9.9	jackson-dataformat-text 2.9.9 COPYRIGHT: Copyright 2007 Tatu Saloranta Copyright 2012 FasterXML.com LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: You must give any other recipients of the Work or Derivative Works a copy of this License; and You must cause any modified files to carry prominent notices stating that You changed the files; and You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS ================================= Modules: csv properties yaml

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			================================ 4P Dependencies: com.fasterxml.jackson.core jackson-core 2.9.9 com.fasterxml.jackson.core jackson-databind 2.9.9 com.fasterxml.jackson.core jackson-annotations 2.9.9 COPYRIGHT: Copyright 2007 Tatu Saloranta Copyright 2012 FasterXML.com LICENSE: Apache 2.0 http://www.apache.org/licenses/LICENSE-2.0 --- org.yaml snakeyaml 1.23 COYRIGHT: Copyright (c) 2008, http://www.snakeyaml.org LICENSE: Apache 2.0 http://www.apache.org/licenses/LICENSE-2.0 Copyright (C) 2019 The Guava Authors Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. === +--- 4th party: com.google.guava:failureaccess:1.0.1 Copyright (C) 2018 The Guava Authors === +--- 4th party: com.google.guava:listenablefuture:9999.0-empty-to-avoid-conflict-with-guava Copyright (C) 2018 The Guava Authors

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
FasterXML, LLC	Guice	4.2.2	Google Guice 4.2.2 ------------------ COPYRIGHT: Copyright (C) 2006-Present Google Inc. LICENSE: Apache 2.0 https://github.com/google/guice/blob/4.2/COPYING Please see below for the Apache License 2.0 text. ********************** Modules: bom core extensions Modules within extensions: assistedinject dagger-adapter grapher jmx jndi multibindings persist servlet spring struts2 testlib throwingproviders ************************** Mandatory Fourth-party Dependency: AOP Alliance 1.0 -------------------------------------- Copyright: none LICENSE: all the source code provided by AOP Alliance is Public Domain. Mandatory Fourth-party Dependency: Guava 25.1-android -- Copyright: Copyright (C) 2007 The Guava Authors License: * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License. * You may obtain a copy of the License at *

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			* http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. Please see below for the Apache License 2.0 text. Mandatory Fifth-party Dependency: Findbugs JSR305 3.0.2 --- Copyright: Copyright (c) 2007-2009, JSR305 expert group All rights reserved. License: The JSR-305 reference implementation (lib/jsr305.jar) is distributed under the terms of the New BSD license: http://www.opensource.org/licenses/bsd-license.php See the JSR-305 home page for more information: http://code.google.com/p/jsr-305/ Please see below for the New BSD License text. Mandatory Fifth-party Dependency: Error Prone Annotations 2.1.3 -- Copyright & License: /* * Copyright 2014 The Error Prone Authors. * * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License. * You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. */ Please see below for the Apache License 2.0 text. Mandatory Fifth-party Dependency: J2ObjC Annotations 1.1 -- Copyright & License: /* * Copyright 2011 Google Inc. All Rights Reserved. * * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License. * You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. */ Please see below for the Apache License 2.0 text. Mandatory (though can be excluded from classpath without ill effect) Fifth-party Dependency: Checker Qual 2.0.0 --- Copyright & License: The Checker Framework Copyright 2004-present by the Checker Framework developers Most of the Checker Framework is licensed under the GNU General Public License, version 2 (GPL2), with the classpath exception. The text of this license appears below. This is the same license used for OpenJDK.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			A few parts of the Checker Framework have more permissive licenses. * The annotations are licensed under the MIT License. (The text of this license appears below.) More specifically, all the parts of the Checker Framework that you might want to include with your own program use the MIT License. This is the checker-qual.jar file and all the files that appear in it: every file in a qual/ directory, plus utility files such as NullnessUtil.java, RegexUtil.java, SignednessUtil.java, etc. In addition, the cleanroom implementations of third-party annotations, which the Checker Framework recognizes as aliases for its own annotations, are licensed under the MIT License. Some external libraries that are included with the Checker Framework have different licenses. * javaparser is dual licensed under the LGPL or the Apache license -- you may use it under whichever one you want. (The javaparser source code contains a file with the text of the GPL, but it is not clear why, since javaparser does not use the GPL.) See file stubparser/LICENSE and the source code of all its files. * JUnit is licensed under the Common Public License v1.0 (see http://www.junit.org/license), with parts (Hamcrest) licensed under the BSD License (see http://hamcrest.org/JavaHamcrest/). * Libraries in plume-lib (https://github.com/plume-lib/) are licensed under the MIT License. The Checker Framework includes annotations for the JDK in directory checker/jdk/, and for some other libraries. Each annotated library uses the same license as the unannotated version of the library. Mandatory Fifth-party dependency: Animal Sniffer Annotations

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			1.14 -- Copyright & License: /* * The MIT License * * Copyright (c) 2008 Kohsuke Kawaguchi and codehaus.org. * * Permission is hereby granted, free of charge, to any person obtaining a copy * of this software and associated documentation files (the "Software"), to deal * in the Software without restriction, including without limitation the rights * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell * copies of the Software, and to permit persons to whom the Software is * furnished to do so, subject to the following conditions: * * The above copyright notice and this permission notice shall be included in * all copies or substantial portions of the Software. * * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN * THE SOFTWARE. * */ Mandatory Fourth-Party Dependency: javax.inject 1 --- Copyright & License: /* * Copyright (C) 2009 The JSR-330 Expert Group * * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			* You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. */ See below for the Apache License 2.0 text. Optional Fourth-party Dependency: CGLib 3.2.9 --- Copyright & License: /* * Copyright 2002,2003,2004 The Apache Software Foundation * * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License. * You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. */ See below for the Apache License 2.0 text. Optional Fifth-party Dependency: Apache Ant Core 1.10.3 --- Copyright & License: Apache Ant Copyright 1999-2018 The Apache Software Foundation This product includes software developed at

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			The Apache Software Foundation (http://www.apache.org/). The task is based on code Copyright (c) 2002, Landmark Graphics Corp that has been kindly donated to the Apache Software Foundation. See below for the Apache License 2.0 text. Optional Sixth-party Dependency: Apache Ant Launcher 1.10.3 --- Copyright & License: Apache Ant Copyright 1999-2018 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). The task is based on code Copyright (c) 2002, Landmark Graphics Corp that has been kindly donated to the Apache Software Foundation. See below for the Apache License 2.0 text. Optional Fourth-party Dependency: ASM 7.0 --- Copyright & License: ASM: a very small and fast Java bytecode manipulation framework Copyright (c) 2000-2011 INRIA, France Telecom All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. Neither the name of the copyright holders nor the names

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. Apache License 2.0 ------------------ Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s)

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS =================================== 4P Dependencies: aopalliance 1.0 COPYRIGHT and LICENSE: All the source code provided by AOP Alliance is Public Domain.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			--- com.google.guava COPYRIGHT: Copyright (C) 2008 The Guava Authors LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
FasterXML, LLC	Protocol Buffers (aka Google protobuf)	3.8.0	Copyright 2008 Google Inc. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. Code generated by the Protocol Buffer compiler is owned by the owner of the input file used when generating it. This code is not standalone and requires a support library to be linked with it. This support library is itself covered by the above license.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
K. Kosako	Onigurama	6.9.2	Oniguruma LICENSE ----------------- Copyright (c) 2002-2019 K.Kosako All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. /*** oniguruma.h - Oniguruma (regular expression library) ***/ /*- * Copyright (c) 2002-2019 K.Kosako

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information

Marc Prud'hommeaux	JLine	2.14.6	`Copyright (c) 2002-2016, the original author or authors.`
			`All rights reserved.`
			
			`http://www.opensource.org/licenses/bsd-license.php`
			
			`Redistribution and use in source and binary forms, with or`
			`without modification, are permitted provided that the`
			`following`
			`conditions are met:`
			
			`Redistributions of source code must retain the above`
			`copyright`
			`notice, this list of conditions and the following disclaimer.`
			
			`Redistributions in binary form must reproduce the above`
			`copyright`
			`notice, this list of conditions and the following disclaimer`
			`in the documentation and/or other materials provided with`
			`the distribution.`
			
			`Neither the name of JLine nor the names of its contributors`
			`may be used to endorse or promote products derived from this`
			`software without specific prior written permission.`
			
			`THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND`
			`CONTRIBUTORS`
			`"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING,`
			`BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY`
			`AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO`
			`EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE`
			`FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,`
			`OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,`
			`PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,`
			`DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED`
			`AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT`
			`LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)`
			`ARISING`
			`IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED`
			`OF THE POSSIBILITY OF SUCH DAMAGE.`
			`=============`
			`jansi 1.12 Apache 2.0`
			`hawtjni 2.4.1 Apache 2.0`
			`Apache License`
			` Version 2.0, January 2004`
			` http://www.apache.org/licenses/`
			
			` TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION`
			
			` 1. Definitions.`
			
			` "License" shall mean the terms and conditions for use,`
			`reproduction,`
			` and distribution as defined by Sections 1 through 9 of this`

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Spoofax Copyright 2016 Delft University of Technology This project includes software developed at the Programming Languages Group at Delft University of Technology (http://www.tudelft.nl). ---------separator--------- Aopalliance 1.0: http://aopalliance.sourceforge.net/ Public Domain ---------separator--------- Capsule 0.6.2: https://github.com/usethesource/capsule/blob/ v0.6.2/LICENSE Copyright (c) Michael Steindorfer and Contributors. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. ---------separator--------- Commons-configuration2 2.2: https://github.com/apache/commons-configuration/blob/CONFIGURATION_2_2/LICENSE.txt Apache Commons Configuration Copyright 2001-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ---------separator--------- Commons-configuration2-jackson 0.7.0: https://github.com/Virtlink/commons-configuration2-jackson/blob/v0.7.0/LICENSE Jackson for Commons Configuration 2 Copyright 2015 - Daniel Pelsmaeker ---------separator--------- Commons-io 2.6: https://github.com/apache/commons-io/blob/commons-io-2.6-RC3/LICENSE.txt Apache Commons IO Copyright 2002-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ---------separator---------

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Commons-lang3 3.8.1: https://github.com/apache/commons-lang/blob/LANG_3_8_1/LICENSE.txt Apache Commons Lang Copyright 2001-2018 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ---------separator--------- Commons-logging 1.2: https://github.com/apache/commons-logging/blob/LOGGING_1_2/LICENSE.txt Apache Commons Logging Copyright 2003-2014 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ---------separator--------- Commons-vfs2 2.2: https://github.com/apache/commons-vfs/blob/commons-vfs2-project-2.2/LICENSE.txt Apache Commons VFS Copyright 2002-2017 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). ---------separator--------- Guava 26.0-jre: https://github.com/google/guava/blob/v26.0/COPYING Copyright (C) 2019 The Guava Authors ---------separator--------- Guice 4.2.2: https://github.com/google/guice/blob/4.2.2/COPYING Copyright (C) 2019 Google Inc. ---------separator--------- Jackson-annotations 2.9.7: https://github.com/FasterXML/

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			jackson-annotations/blob/jackson-annotations-2.9.7/src/main/resources/META-INF/LICENSE Copyright FasterXML.com ---------separator--------- Jackson-core 2.9.7: https://github.com/FasterXML/jackson-core/blob/master/src/main/resources/META-INF/LICENSE Copyright (c) 2007- Tatu Saloranta, tatu.saloranta@iki.fi ---------separator--------- Jackson-databind 2.9.9.3: https://github.com/FasterXML/jackson-databind/blob/jackson-databind-2.9.9.3/src/main/resources/META-INF/LICENSE Copyright FasterXML.com ---------separator--------- Jackson-dataformat-yaml 2.9.7: https://github.com/FasterXML/jackson-dataformats-text/tree/jackson-dataformats-text-2.9.7 Copyright 2017 FasterXML ---------separator--------- Rxjava-core 0.20.7: https://search.maven.org/artifact/com.netflix.rxjava/rxjava-core/0.20.7/jar Copyright 2014 Netflix, Inc. ---------separator--------- Snakeyaml 1.18: https://bitbucket.org/asomov/snakeyaml/src/v1.18/LICENSE.txt Copyright (c) 2008, http://www.snakeyaml.org junixsocket Unix Domain Sockets for Java + RMI-over-AF_UNIX. Copyright (c) 2009-2018 Dr. Christian Kohlsch□□r See https://github.com/kohlschutter/junixsocket for more information. ---------separator---------

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Metaborg	Spoofax	2.5.7	Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
NewsClub, Christian Kohlschütter	junixsocket	2.2.0	junixsocket Copyright 2009-2019 Christian Kohlsch□□r Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. ---------separator--------- Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON- INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Petri Lehtinen	Jansson	2.12	Copyright (c) 2009-2018 Petri Lehtinen Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
QOS.ch	Simple Logging Facade for Java (SLF4J)	1.7.26	Simple Logging Facade for Java (SLF4J) 1.7.26 https://github.com/qos-ch/slf4j/tree/v_1.7.26 Copyright (c) 2004-2017 QOS.ch All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Sabastiano Vigna	fastutil	8.2.3	fastutil 8.2.1 COPYRIGHT: Copyright (C) 2003-2017 Paolo Boldi and Sebastiano Vigna LICENSE: Apache 2.0 Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			`permissions and` `limitations under the License.`

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Skymind	DeepLearning4J	0.9.1	Copyright 2015 Skymind, Inc. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: You must give any other recipients of the Work or Derivative Works a copy of this License; and You must cause any modified files to carry prominent notices stating that You changed the files; and You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. *** As used in PGX, DeepLearning4j includes the following fourth party dependencies nd4j-native-api v. 0.9.1 Copyright 2015 Skymind,Inc. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			deeplearning4j-core v. 0.9.1 Copyright 2015 Skymind,Inc. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. deeplearning4j-nn v. 0.9.1 Copyright 2015 Skymind,Inc. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. nd4j-jackson v. 0.9.1 Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License commons-lang3 v. 3.4

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Copyright 2001-2015 The Apache Software Foundation This product includes software developed at The Apache Software Foundation (http://www.apache.org/). This product includes software from the Spring Framework, under the Apache License 2.0 (see: StringUtils.containsWhitespace()) javacpp v. 1.3.3 You may use this work under the terms of either the Apache License, Version 2.0, or the GNU General Public License (GPL), either version 2, or any later version, with "Classpath" exception (details below). You don't have to do anything special to choose one license or the other and you don't have to notify anyone which license you are using. You are free to use this work in any project (even commercial projects) as long as the copyright header is left intact. == Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. 2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. 6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright [yyyy] [name of copyright owner] Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. GNU GENERAL PUBLIC LICENSE Version 2, June 1991 Copyright (C) 1989, 1991 Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. Preamble The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too. When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			things. To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it. For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights. We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software. Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations. Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all. The precise terms and conditions for copying, distribution and modification follow.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION 0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you". Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does. 1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program. You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions: a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change. b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License. c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably consi

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Skymind	ND4J	0.9.1	Copyright (C) 2017 Skymind --- Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION 1. Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. 3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. 4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. 5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. 7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. 8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. 9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS APPENDIX: How to apply the Apache License to your work. To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "{}" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives. Copyright {yyyy} {name of copyright owner} Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			--- SLF4J (MIT) Copyright (c) 2004-2017 QOS.ch All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. --- Google Guava (Apache) Copyright (C) 2016, The Guava Authors --- Findbugs (Lesser GPL) Copyright (C) 2006, The University of Maryland

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999 Copyright (C) 1991, 1999 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. [This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.] Preamble The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below. When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things. To protect your rights, we need to make restrictions that

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it. For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights. We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library. To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others. ? Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs. When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library. We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances. For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system. Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library. The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run. ? GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION 0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you". A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables. The "Library", below, refers to any such software library

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".) "Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library. Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does. 1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library. You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			exchange for a fee. ? 2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions: a) The modified work must itself be a software library. b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change. c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License. d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful. (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it. Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library. In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License. 3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices. ? Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy. This option is useful when you wish to copy part of the code of the Library into a program that is not a library. 4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange. If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code. 5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License. However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables. When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law. If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.) Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself. ? 6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications. You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things: a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.) b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with. c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution. d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place. e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy. For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute. ? 7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things: a) Accompany the combined library with a copy of the same work based on th

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
SmartBear Software	swagger-annotations	1.5.22	Copyright 2016 SmartBear Software Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at [apache.org/licenses/LICENSE-2.0](http://www.apache.org/licenses/LICENSE-2.0) Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Apache License Version 2.0, January 2004 http://www.apache.org/licenses/ TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION Definitions. "License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document. "Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License. "Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity. "You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License. "Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files. "Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			but not limited to compiled object code, generated documentation, and conversions to other media types. "Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below). "Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof. "Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution." "Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions: (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and (b) You must cause any modified files to carry prominent notices stating that You changed the files; and (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use,

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			reproduction, and distribution of the Work otherwise complies with the conditions stated in this License. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability. END OF TERMS AND CONDITIONS
Tatsuhiro Tsujikawa	Argparse4j	0.8.1	* Copyright (C) 2011-2017 Tatsuhiro Tsujikawa * * Permission is hereby granted, free of charge, to any person * obtaining a copy of this software and associated documentation * files (the "Software"), to deal in the Software without * restriction, including without limitation the rights to use, copy, * modify, merge, publish, distribute, sublicense, and/or sell copies * of the Software, and to permit persons to whom the Software is * furnished to do so, subject to the following conditions: * * The above copyright notice and this permission notice shall be * included in all copies or substantial portions of the Software. * * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, * EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF * MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND * NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS * BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN * ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE * SOFTWARE.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Tipue	Tipue Search	7.1	--- copy right notice --- Tipue Search Copyright (c) 2018 Tipue --- license terms --- Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
Willem-Hendrik Thiart	raft	0.7.0	BSD 3 License URL: https://github.com/willemt/raft/blob/v0.6.0/LICENSE Copyright (c) 2013, Willem-Hendrik Thiart All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. * The names of its contributors may not be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL WILLEM-HENDRIK THIART BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
openssl.org	OpenSSL	1.1.1c	1. Include the following in any advertising and in the about box or other location with copyright notices: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit" "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" 2. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include this additional acknowledgement in any about box or other location with copyright notices: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)" 3. Include the following verbatim in the documentation: The OpenSSL toolkit stays under a double license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. OpenSSL License --------------- /* == * Copyright (c) 1998-2019 The OpenSSL Project. All rights reserved. * * Redistribution and use in source and binary forms, with or without * modification, are permitted provided that the following conditions * are met: * * 1. Redistributions of source code must retain the above copyright * notice, this list of conditions and the following disclaimer. * * 2. Redistributions in binary form must reproduce the above copyright * notice, this list of conditions and the following disclaimer in * the documentation and/or other materials provided with the * distribution. * * 3. All advertising materials mentioning features or use of this * software must display the following acknowledgment: * "This product includes software developed by the OpenSSL

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Componen t(s)	Version	Licensing Information
			Project * for use in the OpenSSL Toolkit. (http://www.openssl.org/)" * * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to * endorse or promote products derived from this software without * prior written permission. For written permission, please contact * openssl-core@openssl.org. * * 5. Products derived from this software may not be called "OpenSSL" * nor may "OpenSSL" appear in their names without prior written * permission of the OpenSSL Project. * * 6. Redistributions of any form whatsoever must retain the following * acknowledgment: * "This product includes software developed by the OpenSSL Project * for use in the OpenSSL Toolkit (http://www.openssl.org/)" * * THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY * EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR * PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED * OF THE POSSIBILITY OF SUCH DAMAGE. * == * * This product includes cryptographic software written by Eric Young * (eay@cryptsoft.com). This product includes software

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			written by Tim * Hudson (tjh@cryptsoft.com). * */ Original SSLeay License ----------------------- /* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) * All rights reserved. * * This package is an SSL implementation written * by Eric Young (eay@cryptsoft.com). * The implementation was written so as to conform with Netscapes SSL. * * This library is free for commercial and non-commercial use as long as * the following conditions are aheared to. The following conditions * apply to all code found in this distribution, be it the RC4, RSA, * lhash, DES, etc., code; not just the SSL code. The SSL documentation * included with this distribution is covered by the same copyright terms * except that the holder is Tim Hudson (tjh@cryptsoft.com). * * Copyright remains Eric Young's, and as such any Copyright notices in * the code are not to be removed. * If this package is used in a product, Eric Young should be given attribution * as the author of the parts of the library used. * This can be in the form of a textual message at program startup or * in documentation (online or textual) provided with the package. * * Redistribution and use in source and binary forms, with or without * modification, are permitted provided that the following conditions * are met: * 1. Redistributions of source code must retain the copyright * notice, this list of conditions and the following disclaimer. * 2. Redistributions in binary form must reproduce the above copyright * notice, this list of conditions and the following disclaimer in the * documentation and/or other materials provided with the

Table A-2 (Cont.) Third-Party License Information (Part 2)

Provider	Component(s)	Version	Licensing Information
			distribution. * 3. All advertising materials mentioning features or use of this software * must display the following acknowledgement: * "This product includes cryptographic software written by * Eric Young (eay@cryptsoft.com)" * The word 'cryptographic' can be left out if the rouines from the library * being used are not cryptographic related :-). * 4. If you include any Windows specific code (or a derivative thereof) from * the apps directory (application code) you must include an acknowledgement: * "This product includes software written by Tim Hudson (tjh@cryptsoft.com)" * * THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF * SUCH DAMAGE. * * The licence and distribution terms for any publically available version or * derivative of this code cannot be changed. i.e. this code cannot simply be * copied and put under another distribution licence * [including the GNU Public Licence.] */

B

Hive and Spark Spatial SQL Functions

This appendix provides reference information about the Hive and Spark spatial SQL functions.
To use these functions, you must understand the concepts and techniques described in whichever of the following apply to your needs:

- Oracle Big Data Spatial Vector Hive Analysis, especially Using the Hive Spatial API,
- Oracle Big Data Spatial Vector Analysis for Spark, especially Spatial Analysis Spark SQL UDFs

The functions are presented alphabetically. However, they can be grouped into the following logical categories: geometry constructors, single-geometry functions, and two-geometry functions.

Geometry constructors:

- ST_Geometry
- ST_LineString
- ST_MultiLineString
- ST_MultiPoint
- ST_MultiPolygon
- ST_Point
- ST_Polygon

Single-geometry functions:

- ST_Area
- ST_AsWKB
- ST_AsWKT
- ST_Buffer
- ST_ConvexHull
- ST_Envelope
- ST_Length
- ST_Simplify
- ST_SimplifyVW
- ST_Volume

Two-geometry functions:

- ST_AnyInteract
- ST_Contains
- ST_Distance

- ST_Inside
- ST_AnyInteract
- ST_Area
- ST_AsWKB
- ST_AsWKT
- ST_Buffer
- ST_Contains
- ST_ConvexHull
- ST_Distance
- ST_Envelope
- ST_Geometry
- ST_Inside
- ST_Length
- ST_LineString
- ST_MultiLineString
- ST_MultiPoint
- ST_MultiPolygon
- ST_Point
- ST_Polygon
- ST_Simplify
- ST_SimplifyVW
- ST_Volume

B.1 ST_AnyInteract

Format

```
ST_AnyInteract(
  geometry1  ST_Geometry,
  geometry1  ST_Geometry,
  tolerance  NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic
geometries));
```

Description

Determines if `geometry1` has any spatial interaction with `geometry2`, returning `true` or `false`.

Parameters

geometry1
A 2D or 3D geometry object.

geometry2
Another 2D or 3D geometry object.

tolerance
Tolerance at which `geometry2` is valid.

Usage Notes

Both geometries must have the same number of dimensions (2 or 3) and the same spatial reference system (SRID, or coordinate system).

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_AnyInteract(
  ST_Point('{ "type": "Point", "coordinates": [2, 3]}', 8307),
  ST_Polygon('{"type": "Polygon","coordinates": [[[1, 2], [5, 2], [5, 6], [1, 6], [1,
2]]]}', 8307))
from hivetable LIMIT 1;
-- return true
```

B.2 ST_Area

Format

```
ST_Area(
  geometry   ST_Geometry
  tolerance  NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic geometries));
```

Description

Returns the area of a polygon or multipolygon geometry.

Parameters

geometry
An ST_Geometry object.

tolerance
Value reflecting the distance that two points can be apart and still be considered the same.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Area(ST_Polygon('{"type": "Polygon","coordinates": [[[1, 2], [5, 2], [5, 7],
[1, 7], [1, 2]]]}', 0))
  from hivetable LIMIT 1;  -- return 20
```

B.3 ST_AsWKB

Format

```
ST_AsWKB(
  geometry ST_Geometry);
```

Description

Returns the well-known binary (WKB) representation of the geometry.

Parameters

geometry
An ST_Geometry object.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_AsWKB( ST_Point('{ "type": "Point", "coordinates": [0, 5]}', 8307))
   from hivetable LIMIT 1;
```

B.4 ST_AsWKT

Format

```
ST_AsWKT(
  geometry ST_Geometry);
```

Description

Returns the well-known text (WKT) representation of the geometry.

Parameters

geometry
An ST_Geometry object.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_AsWKT(ST_Point('{ "type": "Point", "coordinates": [0, 5]}', 8307))
  from hivetable LIMIT 1;
```

B.5 ST_Buffer

Format

```
ST_Buffer(
  geometry     ST_Geometry,
  bufferWidth  NUMBER,
  arcTol       NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic
geometries));
```

Description

Generates a new ST_Geometry object that is the buffered version of the input geometry.

Parameters

geometry
Any 2D geometry object. If the geometry is geodetic, it is interpreted as longitude/latitude values in the WGS84 spatial reference system, and `bufferWidth` and `tolerance` are interpreted as meters.

bufferWidth
The distance value used for the buffer.

arcTol
Tolerance used for geodetic arc densification. (Ignored for nongeodetic geometries.)

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Buffer(ST_Point('{ "type": "Point", "coordinates": [0, 5]}', 0), 3)
  from hivetable LIMIT 1;
-- return {"type":"Polygon", "coordinates": [[[-3,5],[-2.8977774789,4.2235428647],
[-2.5980762114,3.5],[-2.1213203436,2.8786796564],[-1.5,2.4019237886],
[-0.7764571353,2.1022225211],[0,2],[0.7764571353,2.1022225211],[1.5,2.4019237886],
[2.1213203436,2.8786796564],[2.5980762114,3.5],[2.8977774789,4.2235428647],[3,5],
[2.8977774789,5.7764571353],[2.5980762114,6.5],[2.1213203436,7.1213203436],
[1.5,7.5980762114],[0.7764571353,7.8977774789],[0,8],[-0.7764571353,7.8977774789],
[-1.5,7.5980762114],[-2.1213203436,7.1213203436],[-2.5980762114,6.5],
[-2.8977774789,5.7764571353],[-3,5]]],"crs":{"type":"name","properties":
{"name":"EPSG:0"}}}
```

B.6 ST_Contains

Format

```
ST_Contains(
  geometry1  ST_Geometry,
  geometry1  ST_Geometry,
  tolerance  NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic geometries));
```

Description

Determines if `geometry1` contains `geometry2`, returning `true` or `false`.

Parameters

geometry1
A polygon or solid geometry object.

geometry2
Another 2D or 3D geometry object.

tolerance
Tolerance at which `geometry2` is valid.

Usage Notes

Both geometries must have the same number of dimensions (2 or 3) and the same spatial reference system (SRID, or coordinate system).

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Contains(
  ST_Polygon('{"type": "Polygon","coordinates": [[[1, 2], [5, 2], [5, 6], [1, 
6], [1, 2]]]}', 8307),
  ST_Point('{ "type": "Point", "coordinates": [2, 3]}', 8307))
from hivetable LIMIT 1;
-- return true
```

B.7 ST_ConvexHull

Format

```
ST_ConvexHull(
  geometry ST_Geometry);
```

Description

Returns the convex hull of the input geometry as an ST_Geometry object.

Parameters

geometry
A 2D ST_Geometry object.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_ConvexHull(
  ST_MultiPoint(' { "type": "MultiPoint","coordinates": [ [1, 2], [-1, -2], [5, 
```

```
6] ] }', 0))
from hivetable LIMIT 1;
-- return {"type":"Polygon", "coordinates":[[[5,6],[1,2],[-1,-2],[5,6]]],"crs":
{"type":"name","properties":{"name":"EPSG:0"}}}
```

B.8 ST_Distance

Format

```
ST_Distance(
  geometry1  ST_Geometry,
  geometry1  ST_Geometry,
  tolerance  NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic geometries));
```

Description

Determines the distance between two 2D geometries.

Parameters

geometry1
A 2D geometry object.

geometry2
A 2D geometry object.

tolerance
Tolerance at which `geometry2` is valid.

Usage Notes

This function returns thedistance between the two given geometries. For projected data, the distance is in the same unit as the unit of projection. For geodetic data, the distance is in meters.

If an error occurs, the function returns -1.

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Distance(
  ST_Point('{ "type": "Point", "coordinates": [0, 0]}', 0),
  ST_Point('{ "type": "Point", "coordinates": [6, 8]}', 0))
from hivetable LIMIT 1;
-- return 10.0
```

B.9 ST_Envelope

Format

```
ST_Envelope(
  geometry ST_Geometry);
```

Description

Returns the envelope (bounding polygon) of the input geometry as an ST_Geometry object.

Parameters

geometry
A 2D ST_Geometry object.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Envelope(
  ST_MultiPoint(' { "type": "MultiPoint","coordinates": [ [1, 2], [-1, -2], [5, 6] ] }', 0))
from hivetable LIMIT 1;
-- return {"type":"Polygon", "coordinates":[[[-1,-2],[5,-2],[5,6],[-1,6],
[-1,-2]]],"crs":{"type":"name","properties":{"name":"EPSG:0"}}}
```

B.10 ST_Geometry

Format

```
ST_GEOMETRY(
  geometry STRING
  srid INT);
```

or

```
ST_GEOMETRY(
  geometry BINARY
  srid INT);
```

or

```
ST_GEOMETRY(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a GeoJSON string representation of the geometry, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.
The function format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a point using GeoJSON
select ST_Geometry (' { "type": "Point", "coordinates": [100.0, 0.0]}', 8307) from
hivetable LIMIT 1;
-- creates a point using WKT
select ST_Geometry ('point(100.0 0.0)', 8307) from hivetable LIMIT 1;
-- creates the geometries using a HiveRecordInfoProvider
select ST_Geometry (geoColumn, 'hive.samples.SampleHiveRecordInfoProviderImpl') from
hivetable;
```

B.11 ST_Inside

Format

```
ST_Inside(
  geometry1  ST_Geometry,
  geometry1  ST_Geometry,
  tolerance  NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic geometries));
```

Description

Determines if `geometry1` is inside `geometry2`, returning `true` or `false`.

Parameters

geometry1
A 2D or 3D geometry object.

geometry2
A polygon or solid geometry object.

tolerance
Tolerance at which `geometry1` is valid.

Usage Notes

Both geometries must have the same number of dimensions (2 or 3) and the same spatial reference system (SRID, or coordinate system).

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Inside(
  ST_Point('{ "type": "Point", "coordinates": [2, 3]}', 8307),
  ST_Polygon('{"type": "Polygon","coordinates": [[[1, 2], [5, 2], [5, 6], [1,
6], [1, 2]]]}', 8307))
from hivetable LIMIT 1;
-- return true
```

B.12 ST_Length

Format

```
ST_Length(
  geometry   ST_Geometry
  tolerance  NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic
geometries));
```

Description

Returns the length of a line or polygon geometry.

Parameters

geometry
An ST_Geometry object.

tolerance
Value reflecting the distance that two points can be apart and still be considered the same.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Length(ST_Polygon('{"type": "Polygon","coordinates": [[[1, 2], [5, 2],
[5, 6], [1, 6], [1, 2]]]}', 0))
  from hivetable LIMIT 1;  -- return 16
```

B.13 ST_LineString

Format

```
ST_LineString(
  geometry STRING
  srid INT);
```

or

```
ST_LineString(
  geometry BINARY
  srid INT);
```

or

```
ST_LineString(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a line string geometry in GeoJSON format, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.
The function format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a line using GeoJSON
select ST_LineString (' { "type": "LineString","coordinates": [ [100.0, 0.0], [101.0,
1.0] ]} ', 8307) from hivetable LIMIT 1;
-- creates a line using WKT
select ST_LineString (' linestring(1 1, 5 5, 10 10, 20 20)', 8307) from hivetable
LIMIT 1;
-- creates the lines using a HiveRecordInfoProvider
select ST_LineString (geoColumn, 'mypackage.hiveRecordInfoProviderImpl') from
hivetable;
```

B.14 ST_MultiLineString

Format

```
ST_MultiLineString(
  geometry STRING
  srid INT);
```

or

```
ST_MultiLineString(
  geometry BINARY
  srid INT);
```

or

```
ST_MultiLineString(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a multiline string geometry in GeoJSON format, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.
The function format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a MultiLineString using GeoJSON
select ST_ MultiLineString (' { "type": "MultiLineString","coordinates":
[ [ [100.0, 0.0], [101.0, 1.0] ], [ [102.0, 2.0], [103.0, 3.0] ]] }', 8307) from
hivetable LIMIT 1;
-- creates a MultiLineString using WKT
select ST_ MultiLineString ('multilinestring ((10 10, 20 20, 10 40),
(40 40, 30 30, 40 20, 30 10))', 8307) from hivetable LIMIT 1;
-- creates MultiLineStrings using a HiveRecordInfoProvider
select ST_ MultiLineString (geoColumn, 'mypackage.hiveRecordInfoProviderImpl')
from hivetable;
```

B.15 ST_MultiPoint

Format

```
ST_MultiPoint(
  geometry STRING
  srid INT);
```

or

```
ST_MultiPoint(
  geometry BINARY
  srid INT);
```

or

```
ST_MultiPoint(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a multipoint geometry in GeoJSON format, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.
The function format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a MultiPoint using GeoJSON
select ST_MultiPoint (' { "type": "MultiPoint","coordinates": [ [100.0, 0.0], [101.0,
1.0] ] }', 8307) from hivetable LIMIT 1;
-- creates a MultiPoint using WKT
select ST_ MultiPoint ('multipoint ((10 40), (40 30), (20 20), (30 10))', 8307) from
```

```
hivetable LIMIT 1;
-- creates MultiPoints using a HiveRecordInfoProvider
select ST_ MultiPoint (geoColumn, 'mypackage.hiveRecordInfoProviderImpl') from
hivetable;
```

B.16 ST_MultiPolygon

Format

```
ST_MultiPolygon(
  geometry STRING
  srid INT);
```

or

```
ST_MultiPolygon(
  geometry BINARY
  srid INT);
```

or

```
ST_MultiPolygon(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a multipolygon geometry in GeoJSON format, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.
The function format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a MultiPolygon using GeoJSON
select ST_ MultiPolygon (' { "type": "MultiPolygon","coordinates": [[[[102.0, 2.0],
[103.0, 2.0], [103.0, 3.0], [102.0, 3.0], [102.0, 2.0]]], [[[100.0, 0.0], [101.0,
0.0], [101.0, 1.0], [100.0, 1.0], [100.0, 0.0]], [[100.2, 0.2], [100.8, 0.2], [100.8,
0.8], [100.2, 0.8], [100.2, 0.2]]] ] }', 8307) from hivetable LIMIT 1;
-- creates a MultiPolygon using WKT
select ST_ MultiPolygon ('multipolygon(((30 20, 45 40, 10 40, 30 20)),
((15 5, 40 10, 10 20, 5 10, 15 5)))', 8307) from hivetable LIMIT 1;
-- creates MultiPolygons using a HiveRecordInfoProvider
select ST_ MultiPolygon (geoColumn, 'mypackage.hiveRecordInfoProviderImpl') from
hivetable;
```

B.17 ST_Point

Format

```
ST_Point(
  geometry STRING
  srid INT);
```

or

```
ST_Point(
  geometry BINARY
  srid INT);
```

or

```
ST_Point(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a point geometry in GeoJSON format, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.

The format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a point using GeoJSON
select ST_Point (' { "type": "Point", "coordinates": [100.0, 0.0]}', 8307) from
hivetable LIMIT 1;
-- creates a point using WKT
select ST_Point ('point(100.0 0.0)', 8307) from hivetable LIMIT 1;
-- creates the points using a HiveRecordInfoProvider
select ST_Point (geoColumn, 'hive.samples.SampleHiveRecordInfoProviderImpl')
from hivetable;
```

B.18 ST_Polygon

Format

```
ST_Polygon(
  geometry STRING
  srid INT);
```

or

```
ST_Polygon(
  geometry BINARY
  srid INT);
```

or

```
ST_Polygon(
  geometry Object
  hiveRecordInfoProvider STRING);
```

Description

Creates a polygon geometry in GeoJSON format, and returns a GeoJSON string representation of the geometry.

Parameters

geometry
To create a geometry from a GeoJSON or WKT string (first format): Geometry definition in GeoJSON or WKT format.
To create a geometry from a WKB object (second format): Geometry definition in WKB format.
To create a geometry using a Hive object (third format): Geometry definition in any Hive supported type.

srid
Spatial reference system (coordinate system) identifier.

hiveRecordInfoProvider
The fully qualified name of an implementation of the interface `oracle.spatial.hadoop.vector.hive.HiveRecordInfoProvider` to extract the geometry in GeoJSON format.
The function format with the `hiveRecordInfoProvider` parameter does not apply to Spark spatial SQL functions.

Usage Notes

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
-- creates a polygon using GeoJSON
select ST_Polygon (' { "type": "Polygon","coordinates": [ [ [100.0, 0.0], [101.0,
0.0], [101.0, 1.0], [100.0, 1.0], [100.0, 0.0] ]] }', 8307) from hivetable LIMIT 1;
-- creates a polygon using WKT
select ST_ Polygon ('polygon((0 0, 10 0, 10 10, 0 0))', 8307) from hivetable LIMIT 1;
-- creates the polygons using a HiveRecordInfoProvider
select ST_ Polygon (geoColumn, 'mypackage.hiveRecordInfoProviderImpl') from hivetable;
```

B.19 ST_Simplify

Format

```
ST_Simplify(
  geometry   ST_Geometry,
  threshold  NUMBER);
```

Description

Generates a new ST_Geometry object by simplifying the input geometry using the Douglas-Peucker algorithm.

Parameters

geometry
Any 2D geometry object. If the geometry is geodetic, it is interpreted as longitude/latitude values in the WGS84 spatial reference system, and `bufferWidth` and `tolerance` are interpreted as meters.

threshold
Threshold value to be used for the geometry simplification. Should be a positive number. (Zero causes the input geometry to be returned.) If the input geometry is geodetic, the value is the number of meters; if the input geometry is non-geodetic, the value is the number of units associated with the data.
As the threshold value is decreased, the generated geometry is likely to be closer to the input geometry; as the threshold value is increased, fewer vertices are likely to be in the returned geometry.

Usage Notes

Depending on the threshold value, a polygon can simplify into a line or a point, and a line can simplify into a point. Therefore, the output object should be checked for type, because the output geometry type might be different from the input geometry type.

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Simplify(
  ST_POLYGON('{"type": "Polygon","coordinates": [[[1, 2], [1.01, 2.01], [5, 2],
[5, 6], [1, 6], [1, 2]]]}', 0),
  1)
from hivetable LIMIT 1;
-- return {"type":"Polygon", "coordinates":[[[1,2],[5,2],[5,6],[1,6],
[1,2]]],"crs":{"type":"name","properties":{"name":"EPSG:0"}}}
```

B.20 ST_SimplifyVW

Format

```
ST_SimplifyVW(
  geometry   ST_Geometry,
  threshold  NUMBER);
```

Description

Generates a new ST_Geometry object by simplifying the input geometry using the Visvalingham-Whyatt algorithm.

Parameters

geometry
Any 2D geometry object. If the geometry is geodetic, it is interpreted as longitude/latitude values in the WGS84 spatial reference system, and `bufferWidth` and `tolerance` are interpreted as meters.

threshold
Threshold value to be used for the geometry simplification. Should be a positive number. (Zero causes the input geometry to be returned.) If the input geometry is geodetic, the value is the number of meters; if the input geometry is non-geodetic, the value is the number of units associated with the data.
As the threshold value is decreased, the generated geometry is likely to be closer to the input geometry; as the threshold value is increased, fewer vertices are likely to be in the returned geometry.

Usage Notes

Depending on the threshold value, a polygon can simplify into a line or a point, and a line can simplify into a point. Therefore, the output object should be checked for type, because the output geometry type might be different from the input geometry type.

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_SimplifyVW(
  ST_POLYGON('{"type": "Polygon","coordinates": [[[1, 2], [1.01, 2.01], [5, 2],
[5, 6], [1, 6], [1, 2]]]}', 0),
  50)
from hivetable LIMIT 1;
```

```
-- return {"type":"Polygon", "coordinates":[[[1,2],[5,6],[1,6],[1,2]]],"crs":
{"type":"name","properties":{"name":"EPSG:0"}}}
```

B.21 ST_Volume

Format

```
ST_Volume(
  multipolygon  ST_MultiPolygon,
  tolerance     NUMBER DEFAULT 0 (nongeodetic geometries) or 0.05 (geodetic
geometries));
```

Description

Returns the area of a multipolygon 3D geometry. The multipolygon is handled as a solid.

Parameters

multipolygon
An ST_Multipolygon object.

tolerance
Value reflecting the distance that two points can be apart and still be considered the same.

Usage Notes

For projected data, the volume is in the same unit as the unit of projection. For geodetic data, the volume is in cubic meters.

Returns -1 in case of an error.

See also Oracle Big Data Spatial Vector Hive Analysis and Oracle Big Data Spatial Vector Analysis for Spark for conceptual and usage information.

Examples

```
select ST_Volume(
  ST_MultiPolygon (' { "type": "MultiPolygon", "coordinates":
    [[[[0, 0, 0], [0, 0, 1], [0, 1, 1], [0, 1, 0], [0, 0, 0]]],
    [[[0, 0, 0], [0, 1, 0], [1, 1, 0], [1, 0, 0], [0, 0, 0]]],
    [[[0, 0, 0], [1, 0, 0], [1, 0, 1], [0, 0, 1], [0, 0, 0]]],
    [[[1, 1, 0], [1, 1, 1], [1, 0, 1], [1, 0, 0], [1, 1, 0]]],
    [[[0, 1, 0], [0, 1, 1], [1, 1, 1], [1, 1, 0], [0, 1, 0]]],
    [[[0, 0, 1], [1, 0, 1], [1, 1, 1], [0, 1, 1], [0, 0, 1]]]]}',
    0))
  from hivetable LIMIT 1; -- return 1.0
```