Oracle® Fusion Middleware

WebCenter Sites: Analytics Developer's Guide

11*g* Release 1 (11.1.1.6.0)

E39315-01

July 2013

Oracle Fusion Middleware WebCenter Sites: Analytics Developer's Guide 11*g* Release 1 (11.1.1.6.0)

E39315-01

Primary Author: Will Harris

Contributing Author: Sean Cearley, Promila Chitkara, Tatiana Kolubayev, Melinda Rubenau

Contributor: Kannan Appachi, Avadhani Chandrashekar, Saikat Chaudhuri,Joseph Granato, Ravi Khanuja, Patrice Palau, Vijayalakshmi Rajan

Contents

5 Exercise 4: Creating and Preparing a Report for Viewing

Part II Reference

6 Enabling Data Capture for Different Types of Reports

7 WebCenter Sites: Analytics Database Schema

Preface

This guide shows you how to create demographic reports on visitors who browse websites powered by Oracle WebCenter Sites. Part I of this guide is a tutorial for creating reports with custom data capture parameters. Part II is a reference section that discusses enabling data capture for different types of Oracle WebCenter Sites assets and reports. Part II also contains information about the schema of the Analytics database, which stores statistics on site visitors.

Audience

This guide is written primarily for developers. It is assumed that developers have a clear knowledge of their company's business needs, and a basic understanding of their roles in the development of the online site and its back end. This guide is also useful to administrators, who collaborate with developers by setting up content management sites, site users, workflow processes, publishing methods, and Oracle WebCenter Sites client options.

Developers must know Java, JavaServer Pages (JSP), XML, and HTML. Administrators are not required to have programming experience, although a technical background is assumed.

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 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 following documents in the Oracle WebCenter Sites 11*g* Release 1 (11.1.1.6.0) documentation set:

- *Oracle WebCenter Sites Certification Matrix*
- *Oracle WebCenter Sites Release Notes*

- *Oracle Fusion Middleware WebCenter Sites: Analytics User's Guide*
- *Oracle Fusion Middleware WebCenter Sites: Analytics Administrator's Guide*
- *Oracle Fusion Middleware WebCenter Sites: Analytics Tag Reference*

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.

What's New in This Guide

This guide consolidates Oracle WebCenter Sites: Analytics developer's documentation that was once available in separate books. This guide contains the following parts:

- Part I, "Tutorial for Creating Analytics Reports with Custom Parameters" shows developers how to create and configure their own reports for Oracle WebCenter Sites: Analytics.
- Part II, "Reference" contains information about enabling data capture on visitors' usage of different types of published WebCenter Sites assets. This part also outlines the schema of database tables that store statistics on site visitors.

Part I

Tutorial for Creating Analytics Reports with Custom Parameters

This part contains a tutorial to help developers create and configure their own reports for Oracle WebCenter Sites: Analytics.

This part contains the following chapters:

- Chapter 1, "About Creating Reports for Analytics"
- Chapter 2, "Exercise 1: Adding a New Parameter for Data Capture"
- Chapter 3, "Exercise 2: Adding a New Parameter to the DetailDefinition Table"
- Chapter 4, "Exercise 3: Developing a New Analytics Job"
- Chapter 5, "Exercise 4: Creating and Preparing a Report for Viewing"

1 About Creating Reports for Analytics

Part I of this guide is a tutorial with exercises to help you create and configure your own reports for Oracle WebCenter Sites: Analytics.

This chapter outlines the flow of the tutorial and contains the following sections:

- Section 1.1, "About This Tutorial"
- Section 1.2, "The 'NewBrowsers' Report"
- Section 1.3, "Prerequisites for Creating Reports with Custom Parameters"
- Section 1.4, "Quick Reference"

1.1 About This Tutorial

The goal of this tutorial is to provide you with an understanding of how to create and configure reports in Oracle WebCenter Sites: Analytics. To achieve this, you will complete the following exercises:

1. Chapter 2, "Exercise 1: Adding a New Parameter for Data Capture." This exercise walks you through adding a new parameter to Analytics for data capture.
2. Chapter 3, "Exercise 2: Adding a New Parameter to the DetailDefinition Table." This exercise walks you through configuring Analytics to recognize a new parameter. In this exercise, you will add that parameter to the `DETAILDEFINITION` table. An Analytics job can process a parameter for display in a report only if the parameter is defined in the `DETAILDEFINITION` table.
3. Chapter 4, "Exercise 3: Developing a New Analytics Job." This exercise walks you through developing an Analytics job for a new parameter. The Analytics job you create will process the new parameter for display in a report ("NewBrowsers" report in this tutorial).
4. Chapter 5, "Exercise 4: Creating and Preparing a Report for Viewing." This exercise walks you through configuring a custom Analytics report. The report will draw statistical data from the Analytics database and contain the following features:
   - A table
   - A time period selector
   - A chart
   - A data filter

Section 1.4, "Quick Reference" outlines the steps you will complete in each exercise.

Note : Throughout this guide, we assume you are working exclusively with the FirstSite II sample site. Select this site whenever prompted in the Analytics or WebCenter Sites interfaces.

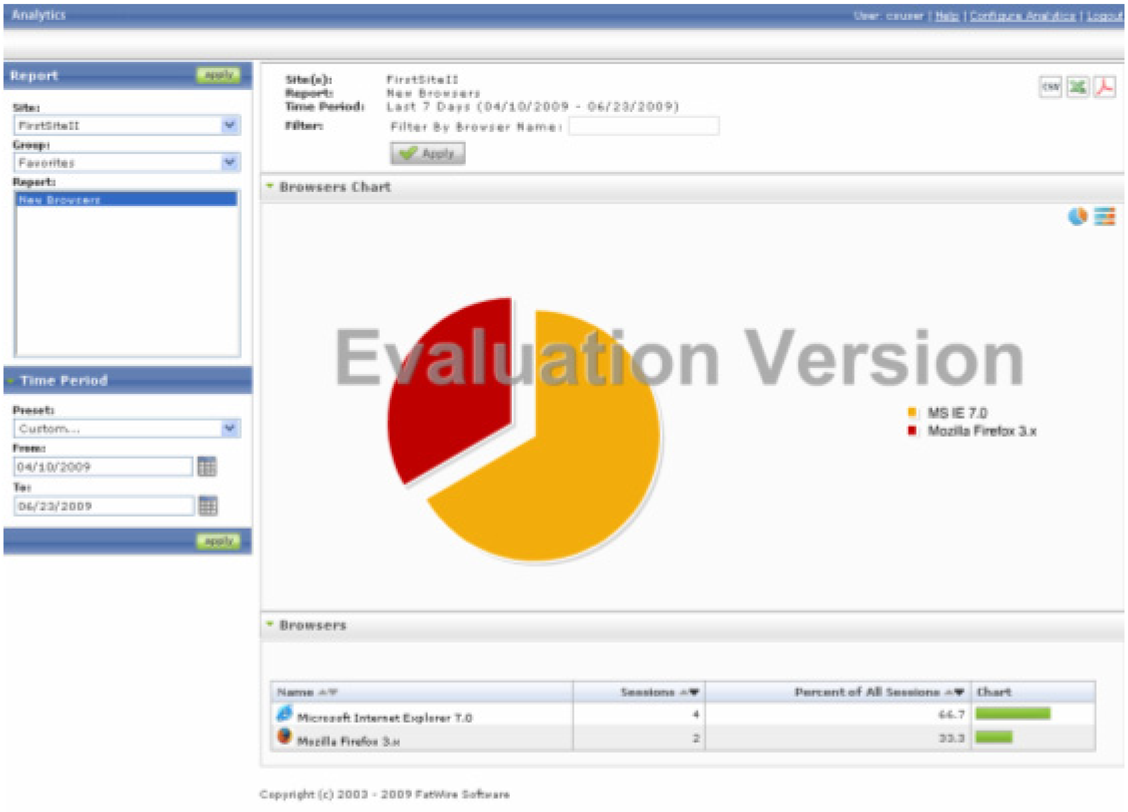
1.2 The 'NewBrowsers' Report

The report you will be creating in this tutorial is called the "NewBrowsers" report.

Note : Your "NewBrowsers" report is a duplicate of the default "Browsers" report in your Analytics installation. For the purposes of this tutorial, the xml file and report name of the "Browsers" report you will be configuring, along with the bean and mapper class names, have been renamed to avoid overwriting the default "Browsers" report. To avoid overwriting the default report, the names of your custom report, its xml file, and bean and mapper classes are unique and used throughout this tutorial.

For a quick reference of the steps you need to complete in order to successfully add a parameter and create a report, see Section 1.4, "Quick Reference." Once you have completed the exercises in this tutorial, your report should be displayed as shown in Figure 1–1.

Figure 1–1 "New Browsers" Report



Site: FirstSiteII

Group: Favorites

Report: New Browsers

Preset: Custom...

From: 04/10/2009

To: 06/23/2009

Site(s): FirstSiteII
Report: New Browsers
Time Period: Last 7 Days (04/10/2009 - 06/23/2009)
Filter: Filter By Browser Name:

Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7.0	4	66.7	🟩🟩🟩🟩
Mozilla Firefox 3.x	2	33.3	🟩🟩

Copyright (c) 2003 - 2009 FatWire Software

1.3 Prerequisites for Creating Reports with Custom Parameters

Before proceeding through the rest of this document, you should:

- Be familiar with configuring and using Oracle WebCenter Sites: Analytics to capture data and generate reports.
- Have a working knowledge of the following aspects of WebCenter Sites:
  - Element code (JSPs). You will be adding a new parameter to the Analytics image tag call in an element asset, as well as the Analytics image tag JSP.
  - Flex and basic asset models. You should be familiar with how your assets are structured in order to know which asset attributes to pass to Analytics for tracking.
  - Database tables. You will be adding a new parameter to the `DETAILDEFINITION` table and extending the Analytics database schema by creating a new `L3` table which will store the processed data collected by the new parameter.
- Know SQL. You will write and modify SQL code that retrieves data in your reports.
- Know XML. Analytics reports are built with XML tags that define report features.

  > **Note :** When writing report code, have the *Oracle Fusion Middleware WebCenter Sites: Analytics Tag Reference* handy. It contains descriptions of the XML tags you will use when coding your reports, as well as the parameters they take.

- Have the following documentation handy in case you need to refer to it when completing the exercises:
  - *Oracle Fusion Middleware WebCenter Sites: Analytics User's Guide*
  - *Oracle Fusion Middleware WebCenter Sites: Analytics Administrator's Guide*
- Be familiar with the basics of the Hadoop Distributed File System and how to create Hadoop Map/Reduce jobs.

  > **Note :** For more information about Hadoop Distributed File Systems and how to create Map/Reduce jobs, refer to the following website:
  >
  > `http://hadoop.apache.org/core/docs/current/mapred_tutorial.html`

1.4 Quick Reference

For your reference, the steps you will follow are outlined below.

Exercise 1: Adding a New Parameter

In Chapter 2, "Exercise 1: Adding a New Parameter for Data Capture" you will add a new parameter to Analytics so Analytics can start capturing data on that parameter. The steps are:

1. Modify the Analytics image tag and its JSP to trigger data capture for the new parameter, as explained in Section 2.2, "Adding the New Parameter to the Image Tag and Its JSP."

2. The next step in this tutorial is to add the new parameter to the `DETAILDEFINITION` table. For instructions, see "Exercise 2: Adding a New Parameter to the DETAILDEFINITON Table."

Exercise 2: Adding a New Parameter to the DETAILDEFINITON Table

In Chapter 3, "Exercise 2: Adding a New Parameter to the DetailDefinition Table" you will configure Analytics to recognize the new parameter by adding that parameter to the `DETAILDEFINITION` table. The steps are:

1. Add the new parameter to the `DETAILDEFINITION` table. This enables Analytics to see the new parameter (created in "Exercise 1: Adding a New Parameter") which, in turn, allows an Analytics job (which you will be creating in "Exercise 3: Developing a New Analytics Job") to process the parameter for display in a report ('NewBrowsers' report in this example).
2. The next step in this tutorial is to develop a new Analytics job. Before you start the next exercise, browse your FSII site to create data for the new Analytics job (you will be developing) to process. Make sure that your Analytics sensor, Hadoop, and HDFS Agent are running before you proceed to "Exercise 3: Developing a New Analytics Job."

Exercise 3: Developing a New Analytics Job

In Chapter 4, "Exercise 3: Developing a New Analytics Job" you will be developing an Analytics job to process the raw data that the new parameter captures. The steps are:

1. Develop a new Analytics job. In this example we use the "NewBrowsers" report (which you will create in "Exercise 4: Creating and Configuring a Report"). The new job enables Analytics to process raw data captured from the given site. The processed data will then be inserted into the database. For instructions, see Section 4.1.2, "Developing an Analytics Job for the 'New Browsers' Report." The steps are outlined below:
   a. Select the location that will be used as the input data for the report. For instructions, see Section 4.1.2.1, "Step 1: Select the Input Location."
   b. Add a new `L3` table to store the processed data. For instructions, see Section 4.1.2.2, "Step 2: Extend the Schema."
   c. Create a new bean class which will store the output data. For instructions, see Section 4.1.2.3, "Step 3: Create the Beans."
   d. Create a new mapper class which will encapsulate the business logic to process the input data for the report. For instructions, see Section 4.1.2.4, "Step 4: Create the Mapper Classes."
   e. Configure the processor to integrate the bean and mapper classes. For instructions, see Section 4.1.2.5, "Step 5: Adding Beans and Mappers to the Processor Definitions."
2. Create and configure the "NewBrowsers" report, which will include the new parameter you create in this exercise. For instructions, see "Exercise 4: Creating and Configuring a Report."

Exercise 4: Creating and Configuring a Report

In Chapter 5, "Exercise 4: Creating and Preparing a Report for Viewing" you will create, from scratch, a "NewBrowsers" report, by completing the following steps:

Note : The "NewBrowsers" report you will be creating in this tutorial, is a duplicate of the default "Browsers" report in your Analytics installation. For the purposes of this tutorial, the xml file and report name of the "Browsers" report you will be configuring, along with the bean and mapper class names, have been renamed to avoid overwriting the default report.

1. Create a new report file and register it with Analytics as explained in Section 5.2.1, "Creating and Registering the Report File" and outlined below:

   a. Create the XML file that will contain the report code. For instructions, see Section 5.2.1.1, "Step 1: Create the XML File."

   b. Place the file in the Analytics reports directory so that Analytics can access the file. For instructions, see Section 5.2.1.2, "Step 2: Place the XML File in the Analytics Reports Directory."

   c. Label the report components, such as module captions, column heads, and so on by defining them in a property file and registering the file with Analytics, as explained in Section 5.2.1.3, "Step 3: Label the Report Components."

   d. Make the report available to Analytics users, as explained in Section 5.2.1.4, "Step 4: Make the Report Available to Analytics Users."

   e. Test the report, as explained in Section 5.2.1.5, "Step 5: Test the New Report" to make sure it has been coded correctly.

2. Add a table, as explained in Section 5.2.2, "Adding a Table." A table is the primary method of presenting statistical data in an Analytics report.

3. Add a time period selector, as explained in Section 5.2.3, "Adding a Time Period Selector." The selector allows users to restrict the statistical data displayed in the report to a specific time period.

4. Add a data filter, as explained in Section 5.3, "Adding a Filter." Filters allow users to restrict the statistical data displayed in the report to specific parameter values. A separate filter is required for each parameter against which you want to filter data.

5. Add a chart, as explained in Section 5.3.3, "Adding a Chart." Charts allow you to visually present statistical data in a report.

6. Test your completed "NewBrowsers" report. At this point, you should have a complete, functional report, as shown in Section 5.3.4, "Testing the Completed 'NewBrowsers' Report."

2

Exercise 1: Adding a New Parameter for Data Capture

This exercise shows you how to add a new parameter to Analytics for data capture.

This exercise contains the following sections:

- Section 2.1, "Overview"
- Section 2.2, "Adding the New Parameter to the Image Tag and Its JSP"
- Section 2.3, "Next Steps"

2.1 Overview

In this exercise, based on the FirstSite II sample site, you will configure Analytics to start capturing data on a new parameter, `agent`. The new `agent` parameter will be used for capturing raw data on the browsers used by visitors of your site.

Once you add the new `agent` parameter to Analytics, you will proceed to the next exercise (Chapter 3, "Exercise 2: Adding a New Parameter to the DetailDefinition Table") in order to configure Analytics to recognize the new `agent` parameter. Then, you will follow instructions in Chapter 4, "Exercise 3: Developing a New Analytics Job" to create an Analytics job that statistically processes the new parameter for display in a report ("NewBrowsers" report in this tutorial). The statistics of the processed `agent` parameter will be displayed as a table column in the "NewBrowsers" report once you create the report, as shown in Chapter 5, "Exercise 4: Creating and Preparing a Report for Viewing."

> **Note:** Throughout this exercise, we assume you are working exclusively with the FirstSite II sample site. Select this site whenever prompted in the Analytics or WebCenter Sites interfaces.

2.2 Adding the New Parameter to the Image Tag and Its JSP

The first step is to make Analytics aware of the new parameter so that it starts capturing data on that parameter. In our example, you will do so by adding the parameter to the Analytics image tag inside the `FSIIWrapper` element and to the `AddAnalyticsImgTag` JSP. When you complete the steps below, the sensor servlet will start capturing the parameter values and storing them as raw data.

1. Add the new parameter to the Analytics image tag call in the `FSIIWrapper` element:

a. Log in to the WebCenter Sites Admin interface as an administrator or another user who has the permissions to edit "CSElement" assets.

b. When prompted, select the FirstSite II sample site.

c. In the button bar, click **Search**.

d. In the list of asset types, click **Find CSElement**.

e. In the "Search" form, enter **FSIIWrapper** into the search field and click **Search**.

f. In the list of results, click the **Edit** (pencil) icon next to the **FSIIWrapper** asset.

g. WebCenter Sites displays the "FSIIWrapper" asset in the "Edit" form.

h. In the section selector at the top of the form, click **Element**.

i. WebCenter Sites displays the "Element" section of the "Edit" form.

j. To make editing the element code easier, select all of the text in the **Element Logic** field, then copy and paste it into a text editor of your choice. Save the code to a temporary file.

– In the element code, locate the call to the Analytics image tag, shown below:

```
<render:callelement elementname="Analytics/AddAnalyticsImgTag">
    <render:argument name="c" value='<%=ics.GetVar("c")%>'/>
    <render:argument name="cid" value='<%=ics.GetVar("cid")%>'/>
    <render:argument name="site" value='<%=ics.GetVar("site")%>'/>
    <render:argument name="pagename"
value='<%=ics.GetVar("childpagename")%>'/>
    <render:argument name="recid" value='<%=ics.GetVar("recid")%>'/>
</render:callelement>
```

– Insert the following statement at the end of the Analytics image tag call, but before the closing `</render:callelement>` tag:

```
<render:argument name="agent" value='<%=
request.getHeader("User-Agent")%>'/>
```

Note: The `agent` variable must be set before the Analytics image tag is called.

– The modified Analytics image tag call should look as follows:

```
<render:callelement elementname="Analytics/AddAnalyticsImgTag">
<render:argument name="c" value='<%=ics.GetVar("c")%>'/>
<render:argument name="cid" value='<%=ics.GetVar("cid")%>'/>
<render:argument name="site" value='<%=ics.GetVar("site")%>'/>
<render:argument name="pagename"
value='<%=ics.GetVar("childpagename")%>'/>
<render:argument name="recid" value='<%=ics.GetVar("recid")%>'/>
<render:argument name="agent" value='<%= request.getHeader(
"User-Agent" )
%>'/>
   </render:callelement>
```

– Save and close the file.

2. Add the new parameter to the Analytics image tag JSP.

a. Start WebCenter Sites Explorer.

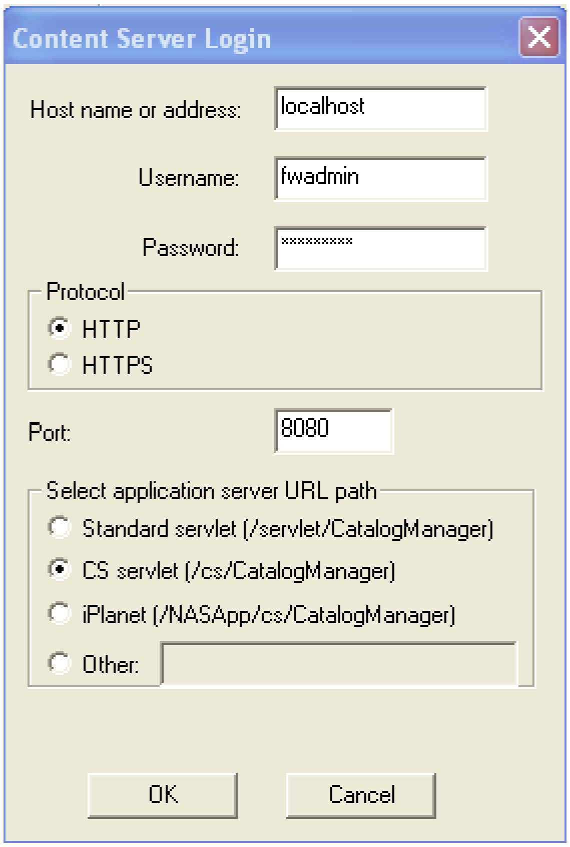
b. In the WebCenter Sites Explorer screen, select **File**, then **Open Content Server** to display the "Content Server Login" dialog box.

c. Enter the following values:

- **Host name or address** – The host name or IP address. You must not leave this blank.
- **Username** – Your WebCenter Sites user name.
- **Password** – Your WebCenter Sites password.
- **Port** – The port number (the default is 8080).
- **Protocol** – Typically, this is HTTP. (You may select HTTPS if the web server is running SSL).
- **Application server URL path** – Select the `CS servlet (/cs/CatalogManager)` option.

Your completed login form should be similar to the one in Figure 2–1:

Figure 2–1 WebCenter Sites Explorer Login Form



Content Server Login

Host name or address: localhost

Username: fwadmin

Password: *********

Protocol

● HTTP

○ HTTPS

Port: 8080

Select application server URL path

○ Standard servlet (/servlet/CatalogManager)

● CS servlet (/cs/CatalogManager)

○ iPlanet (/NASApp/cs/CatalogManager)

○ Other:

d. Click **OK** to log in.

The Sites Explorer window appears.

e. In the **localhost** tree, expand the following: **Tables**, then **ElementCatalog**, and then **Analytics**.

f. In the `Analytics` table, open the `AddAnalyticsImgTag` JSP.

g. Copy and paste the `AddAnalyticsImgTag.jsp` code in a text editor and locate the following lines:

```
String recid = ics.GetVar("recid");
String siteName = ics.GetVar("site");
```

h. Add the following line after the two lines listed in step g (directly above:)

```
String agent = ics.GetVar("agent");
```

The modified section will look as follows:

```
String recid = ics.GetVar("recid");
String siteName = ics.GetVar("site");
String agent = ics.GetVar("agent");
```

i. Locate the following piece of code and insert the bold-type code in the exact locations shown:

```
<script type="text/javascript">
<!--
var jsParam = '';
if(self.StatPixelParamFromPage) {jsParam=StatPixelParamFromPage;};
var color = 'n/a'; var size = 'n/a'; var puri = '';
var java = navigator.javaEnabled();
var nav = navigator.appName;
var agent = navigator.userAgent;
var objValue = '';
var write = (navigator.appName=='Netscape' &&
(navigator.appVersion.charAt(0)=='2' ||
navigator.appVersion.charAt(0)=='3')) ? false:true;
if (write==true) {
size = screen.width + "x" + screen.height;
color = (nav!='Netscape') ? screen.colorDepth:screen.pixelDepth;
objValue= encodeURIComponent('<%= ics.GetVar("assetname") %>');
puri = '?siteName=<%= siteName %>&objType=<%= UobjType %>&objID=<%= id
%>&objName=' +objValue+'&sessionID=<%= TsessionID %>
&agent=<%= agent %>&Referer=<%= referer
%>&nav='+nav+'&size='+size+'&color='+color+'&java=
'+java+'&js=true'+jsParam;
puri += '<%= recparameter%>';
puri += '<%= urlEncparameter%>';
//alert(puri);
document.write('<img style="display:none;" height="1" width="1" border="0"
src="<%= statisticSensorPath%>'+puri+'" alt="pixel" /
>');
}
//-->
</script>
<noscript>
<img style="display:none;" height="1" width="1" src="<%=
statisticSensorPath%>?siteName=<%= siteName%>&objType=<%=
UobjType%>&objID=<%= id%>&objName=<%= ics.GetVar("assetname")%>&URLENC=<%=
ics.GetVar("objUrl")
%>&sessionID=<%= TsessionID%>
&agent=<%= agent %>&Referer=<%= referer%>&js=false"
alt="pixel" />
</noscript>
```

3. When you have made your modifications, select all of the code in your text editor, then copy and paste it back into the Sites Explorer field.

4. **Save** and close the file.
5. Restart the application server for your changes to take effect.

2.3 Next Steps

In Chapter 3, "Exercise 2: Adding a New Parameter to the DetailDefinition Table," you will add the `agent` parameter to the `DETAILDEFINITION` table for data capture. This enables Analytics to process the `agent` parameter which, in turn, makes the parameter available to an Analytics job (which you will create in Chapter 4, "Exercise 3: Developing a New Analytics Job") to process the raw data captured by the new `agent` parameter. The new Analytics job will process the `agent` parameter for display in a report ("NewBrowsers" report in this tutorial). The processed `agent` parameter will be displayed as a table column in the "NewBrowsers" report.

3

Exercise 2: Adding a New Parameter to the DetailDefinition Table

Custom parameters capture data on specific operations visitors perform on your website. This exercise shows you how to add a custom parameter to the DETAILDEFINITION table for data capture.

This exercise contains the following sections:

- Section 3.1, "Overview"
- Section 3.2, "Create a Database Entry for a Custom Parameter"
- Section 3.3, "Next Steps"

3.1 Overview

To create a report based on custom parameters, you must first create the parameters for which you wish to capture data, and then configure Analytics to process those parameters. In this exercise, you will be configuring Analytics to process the agent parameter (created in Chapter 2, "Exercise 1: Adding a New Parameter for Data Capture") by creating a database entry for the parameter in the DETAILDEFINITION table.

Any parameter defined in the DETAILDEFINITION table can be seen by Analytics which, in turn, makes the parameter available to the Analytics job that is created to process that parameter. You will be creating an Analytics job to process the agent parameter in the next exercise (Chapter 4, "Exercise 3: Developing a New Analytics Job").

3.2 Create a Database Entry for a Custom Parameter

For an Analytics job to process a custom parameter for display in your report, you will add a row (record) to the DETAILDEFINITION table in the Analytics database for the custom parameter on which you wish to capture data. In this exercise, you will add a row for the agent parameter (created in Chapter 2, "Exercise 1: Adding a New Parameter for Data Capture").

Since the DETAILDEFINITION table requires you to specify the type of parameter you are adding as an entry to it, you should have an understanding of the types of custom parameters Analytics supports.

Analytics supports the following types of custom parameters:

- **Session related –** These parameters are bound to a particular session. The value of a session-related parameter remains the same for the entire session.

- **OI (Object Impression) related** – These parameters are bound to a particular page-view or object-impression. The values of OI-related parameters are different for each page rendered (object impression).

The `agent` parameter that you will be adding to the `DETAILDEFINITION` table is a session related parameter. The `DETAILDEFINITION` table's schema is shown in Table 3–1.

Table 3–1 The* `DETAILDEFINITION` *table

Column Name	Data Type	Description
`Id`	NUMBER (22)	The primary key, which is a unique number generated by the database for a custom parameter. This key is used to access the row for this parameter.
`Name`	VARCHAR2 (64 Bytes)	The name of the custom parameter.
`Key`	VARCHAR2 (64 Bytes)	The parameter name to be passed via the image tag, which is generally the same value as the `Name` parameter. This name is used in the tag to identify the value being passed.
`Objecttypeid`	NUMBER (22)	The object type identifier which is used by Analytics to track the number of different `objecttypeid` occurrences.
`Siteid`	NUMBER (22)	The `siteid` (foreign key from the site table) for which the parameter is configured. If you set the value to `Null`, the parameter is applicable for all registered sites.
`Type`	NUMBER (22)	Specifies the type of parameter. **Possible values:** 0 for OI-related parameters and 1 for session-related parameters.
`Groupfunction`	VARCHAR2 (64 Bytes)	This field is used only for the `L2_SESSIONDET` table in the Analytics database. In this exercise, the value of this field is `NULL`.
`Autodef`	NUMBER (22)	This field is deprecated. In this exercise, the value of this field is `NULL`.

To store the data collected by the `agent` parameter, you will need to add the new record to the `DETAILDEFINITION` table. Execute the following SQL statement (replacing the sample `Values` with the values for the parameter you wish to insert into the table):

```
insert into DETAILDEFINITION (ID, NAME, KEY, OBBJECTTYPEID, SITEID, TYPE,
GROUPFUNCTION, AUTODEF) Values(249, 'mime', 'mime', NULL, NULL, 0, NULL, NULL);
```

Note: The `ID` must be unique and must not conflict with any existing rows. `SITEID` must be the value for the site as registered in the Analytics database.

3.3 Next Steps

In Chapter 4, "Exercise 3: Developing a New Analytics Job" you will create a new Analytics job to process the raw data that is captured by the new `agent` parameter you added to the `DETAILDEFINITION` table in this exercise. The new Analytics job will process the `agent` parameter for display in a report ("NewBrowsers" report in this tutorial). The processed `agent` parameter will be displayed as a table column in the "NewBrowsers" report.

Before you develop a new Analytics job, keep in mind that:

In order for the new parameter to start capturing data for Analytics, the following components must be installed and running:

- Analytics Sensor
- Hadoop
- hadoop-jobs
- HDFS Agent

Note: Before developing a new Analytics job, browse your FSII site. Browsing ensures that there will be data to process for the new Analytics job (you will be developing in the next exercise).

4

Exercise 3: Developing a New Analytics Job

In order for Analytics to process raw data for a custom report, you must develop a new Analytics job to process that data. The processed data will be inserted into the database.

> **Note :** This exercise walks you through the process of developing a new Analytics job, for the parameter you added (in Exercise 1) to your Analytics installation. The Analytics job you will be developing is a duplicate of the default Analytics job in your Analytics installation. For the purposes of this tutorial, all bean and mapper class names used to develop the Analytics job in this exercise have been renamed to avoid overwriting the default Analytics job.

This exercise contains the following sections:

- Section 4.1, "Overview for Creating an Analytics Job"
- Section 4.2, "Next Steps"

4.1 Overview for Creating an Analytics Job

The following is a brief description of the steps that are required for developing an Analytics job.

1. Select the most appropriate location as the input:

   Select a location from the folders within the `analytics` directory, which must be used by the Analytics job as input data for the report. The criteria for choosing a particular location as an input depends primarily on the data required for a particular report. In most instances, the following location is sufficient for satisfying the data requirements for your report.

   `Sesdata`: This location stores the data for all sessions, along with the details of the visitor, object impressions, and other relevant information associated with a session.

> **Note :** When creating a custom report, do not use the `injected` folders (such as `oiinjected`, `sesinjected`, or `visinjected)` as your input location(s), because the data stored within these locations is used by the database injection processor to store into the database.
>
> For more information about locations and processors, and which type of data is stored in each location, see the *Oracle Fusion Middleware WebCenter Sites: Analytics Administrator's Guide*.

2. Extend the schema:

   To store the processed data, you will add a new `L3` table to the Analytics database.

3. Create a new bean class:

   The main purpose of the bean class is to store the output data. The framework will use the bean class to store the final output of the job. The database injection processors will take the data stored within each bean class and insert the data into the `L3` table (created in the previous step).

   The new bean class extends the pre-defined classes provided by the framework. Implementation details are explained in the following sections: Section 4.1.1, "Example for Developing a New Analytics Job" and Section 4.1.2, "Developing an Analytics Job for the 'New Browsers' Report."

4. Create a new `Mapper` class:

   The `Mapper` class will encapsulate the business logic to process the input data. For every input bean, the Mapper class will create a new instance of the bean class (created in step 3), and then store the processed data in the newly created instance. The output of the Mapper class will be collected by the Analytics framework and further processed before it is finally written to the designated output location.

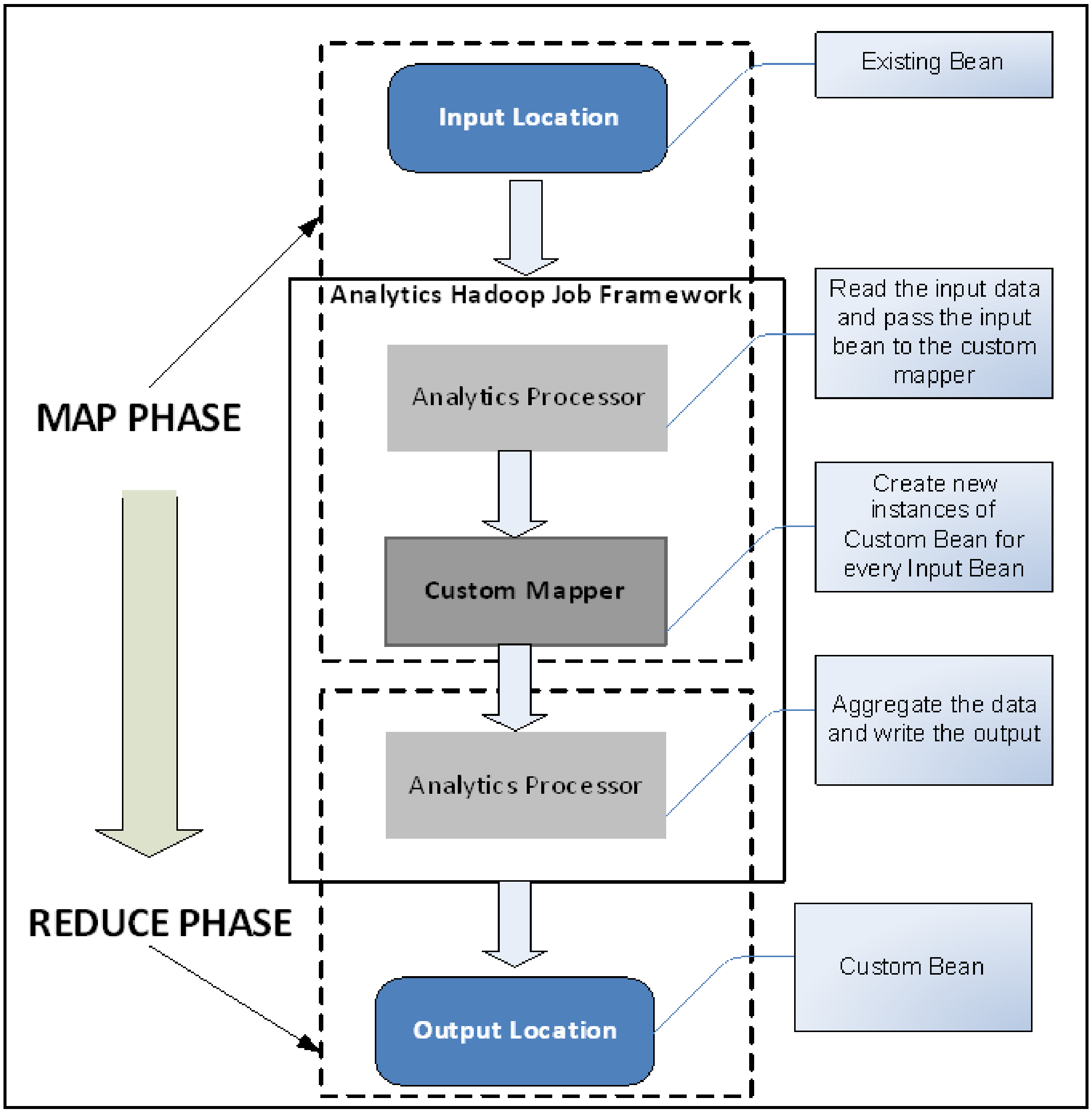
   The new Mapper class will extend the pre-defined classes provided by the framework. The implementation details will be explained in the next section.

5. Configuring the processor.

   To integrate your newly coded beans and Mapper classes with the Analytics framework, you need to add them to the existing processor configuration file (`.xml` file).

Figure 4–1 depicts the job execution flow with the custom mapper integrated into the Analytics-Hadoop job framework.

Figure 4–1 Execution flow chart with custom mapper integrated with the analytics hadoop job framework


Input Location
Existing Bean
Analytics Hadoop Job Framework
Analytics Processor
Read the input data and pass the input bean to the custom mapper
MAP PHASE
Custom Mapper
Create new instances of Custom Bean for every Input Bean
Analytics Processor
Aggregate the data and write the output
REDUCE PHASE
Output Location
Custom Bean

In the **Map Phase**, the processor will read the data from the input location. This data will be passed to the custom mapper. The custom mapper will transform every input bean to custom bean.

In the **Reduce Phase**, the data collected from the Custom Mapper will be further processed before it is written to the output location. The content of the output location will contain the Custom Bean. This aggregated data will be stored into the database by the database injection processor.

4.1.1 Example for Developing a New Analytics Job

This section contains the following topics:

- Section 4.1.1.1, "Designing the 'NewBrowsers' Report"
- Section 4.1.2, "Developing an Analytics Job for the 'New Browsers' Report"
- Section 4.1.3, "Configuring Database Injection"

- Section 4.1.4, "Integrating the New Analytics Job with the Existing Hadoop-Jobs Component"

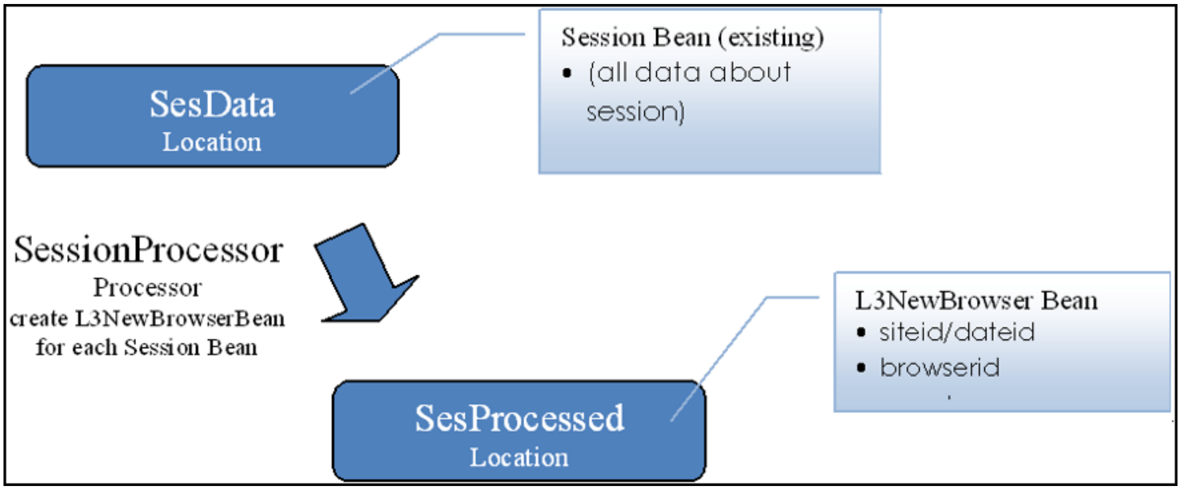
4.1.1.1 Designing the 'NewBrowsers' Report

The "NewBrowsers" report identifies the browsers that visitors used to access a given site's page view within the reported time period.

Counting browsers is simply counting the sessions for each browser. We simply sum up the aggregated values on sessions. Aggregating sums for all possible time ranges that users can select results in too many aggregated values, so we concentrate on daily sums.

Data aggregation is done in a single Map-Reduce phase. (1) The starting point is the session data (`SesData` location). You will use this data to generate intermediate/uncompressed raw data stored as `L3NewBrowserBean` objects in the `SesProcessed` location. (2) The `SessionBean` objects are then used by the `SessionProcessor` processor to create `L3NewBrowserBean` objects, which store aggregated data that can be inserted into the database. The implementation is illustrated in Figure 4–2.

Figure 4–2 "NewBrowsers" Report Implementation



- The `SesData` location has all data on all sessions. You will use this data as the starting point for generating your report data. There is no need to add to or modify that data; you can use the existing `SessionBean` objects.
- Extend the `SessionProcessor` processor to create a new `L3NewBrowserBean` object for each combination of `siteid`, `dateid`, `browserid`, siteid/dateid/browserid combination. This object will be stored by the SessionProcessor processor in the Sesinjected location.
- The `SessionInjection` processor inserts the data into the database. A new bean is not required, but for proper insertion into the database, make sure that you have properly annotated the fields of the `L3` Bean (created in the `SessionProcessor`) with theAggregator.

Note : The Aggregator is a java annotation used to tag fields of a bean that can be aggregated. If you wish to use an Aggregator other than `SumAggregator`, then you can use any of the following Aggregators listed:

- `AvgAggregator`
- `CountAggregator`
- `DistinctCountAggregator`
- `MaxAggregator`
- `NullAggregator`
- `MinAggregator`

4.1.2 Developing an Analytics Job for the 'New Browsers' Report

Follow the general steps, as described in Section 4.1, "Overview for Creating an Analytics Job," for developing an Analytics Job.

Section 4.1.2.1, "Step 1: Select the Input Location"

Section 4.1.2.2, "Step 2: Extend the Schema"

Section 4.1.2.3, "Step 3: Create the Beans"

Section 4.1.2.4, "Step 4: Create the Mapper Classes"

Section 4.1.2.5, "Step 5: Adding Beans and Mappers to the Processor Definitions"

4.1.2.1 Step 1: Select the Input Location

In the "NewBrowsers" report you are aggregating data on a session, so the input location should be the `sesdata` location.

4.1.2.2 Step 2: Extend the Schema

To store the data, you will need to add a new table to store the pre-aggregated `L3` data. Execute the following SQL statement as the `analytics` user:

```
CREATE TABLE L3_DATEXSITEXNEWBROWSERXCOUNT ( DATEID NUMBER NOT NULL , SITEID
NUMBER(6) , BROWSERID NUMBER , COUNT NUMBER);commit;
```

4.1.2.3 Step 3: Create the Beans

For this report, one bean is required:

`L3NewBrowserBean` - This class is mapped to the `L3` table in the database. The `L3NewBrowserBean` is used to store the aggregated count of browser visits. The content of this bean will be injected into the `L3_DATEXSITEXNEWBROWSERXCOUNT` table.

Create the `L3NewBrowserBean` bean class. For sample code, see the example in Section 4–1, "L3NewBrowserBean.java."

Example 4–1 L3NewBrowserBean.java

```
package com.fatwire.analytics.domain.l3;
import javax.persistence.AttributeOverride;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Transient;
import com.fatwire.analytics.domain.AbstractL3Bean;
import com.fatwire.analytics.domain.annotation.Aggregate;
import com.fatwire.analytics.domain.annotation.SumAggregator;
import com.fatwire.analytics.domain.key.DateSiteEntityKey;
/**
* this class represents entries in the L3_dateXsiteXnewbrowserXcount table
*/
@Entity(name="L3_dateXsiteXnewbrowserXcount")
@AttributeOverride(name = "key.firstEntity", column = @Column(name = "BROWSERID", nullable = false, insertable = false, updatable = false))
public class L3NewBrowserBean extends AbstractL3Bean<DateSiteEntityKey> {
private static final long serialVersionUID = 1L;
/** the primary key definition of the object/table */
@Id
private DateSiteEntityKey key = new DateSiteEntityKey();
/** the count value for the entity */
@Aggregate(aggregatorClass = SumAggregator.class)
private Long count;
/** overwrite constructor to set the hadoop keys
public L3NewBrowserBean() {
keys = new String[]{"dateid", "siteid", "browserid", "type"};
}
public Long getBrowserid() {
return key.getFirstEntity();
}
public void setBrowserid(Long browserid) {
key.setFirstEntity(browserid);
}
@Transient
public DateSiteEntityKey getKey() {
return key;
}
public void setKey(DateSiteEntityKey key) {
this.key = key;
}
public Long getSiteid() {
return key.getSiteid();
}
public void setSiteid(Long siteid) {
```

```
45. key.setSiteid(siteid);
46. }
47. public Long getDateid() {
48. return key.getDateid();
49. }
50. public void setDateid(Long dateid) {
51. key.setDateid(dateid);
52. }
53. public Long getCount() {
54. return count;
55. }
56. public void setCount(Long count) {
57. this.count = count;
58.   }
59. }
```

The `L3NewBrowserBean` code is analyzed as follows:

- Lines 2-10: Import all the required packages
- Line 16: Extends the `com.fatwire.analytics.domain.AbstractL3Bean` class.
- Lines 20-23:

  Declare private member variables where:

  – `key`: primary key

  – `count`: number of sessions
- `@Entity` annotation designates this class as persistent entity thereby making it eligible for use by the JPA services (Line 14). The value of the name attribute is the name of the database table to which the entity should be mapped.
- With the `@Attribute` annotation (Line 15), the `L3_DATEXSITEXNEWBROWSERXCOUNT` table would have the `key.firstEntity` attribute of the persistent entity mapped to the `BROWSERID` column.
- Use the `DateSiteEntity` key (Line 20) which is an implementation of an `L3` multi-column primary key.

> **Note :** If you do not wish to use the `DateSiteEntity` implementation, then you can use any of the following multi-column primary key implementations:
>
> - `DateSite`**Entity**`Entitykey`
> - `DateSite`**Entity**`Stringkey`
> - `DateSiteStringkey`
>
> where **Entity** represents a numeric entity.
>
> The choice of implementation will depend solely on the primary key of the table used for storing the contents of the `L3` bean.

- Use the `@Id` annotation (Line 19) to designate `DateSiteEntity` key member variable as the entity's primary key.
- Use the aggregate annotation to annotate the count field with `SumAggregator`. The `SumAggregator` annotation is used to sum the session count for each browser. (Line 22)
- In the constructor specify the key on the basis of which multiple instances of the bean class will be aggregated (Lines 25-26). The `type` signifies the name of the bean class.
- Implement getter/setter methods to expose the private member fields (Lines 28-57).

4.1.2.4 Step 4: Create the Mapper Classes

For this report, one Mapper class `L3NewBrowserMapper` `(L3NewBrowserMapper.java)` is required:

Example 4–2 L3NewBrowserMapper.java

```
1.  package com.fatwire.analytics.report.mapper;
2.  import java.io.IOException;
3.  import org.apache.log4j.Logger;
4.  import com.fatwire.analytics.domain.SessionBean;
5.  import com.fatwire.analytics.domain.l3.L3NewBrowserBean;
6.  import com.fatwire.analytics.mapreduce.AbstractAnalyticsMapper;
7.  import com.fatwire.analytics.mapreduce.AnalyticsOutputCollector;
8.  /**
9.  * L3 mapper on New Browser
10. */
11. public class L3NewBrowserMapper extends AbstractAnalyticsMapper<SessionBean,
    L3NewBrowserBean> {
12.
13. /** initialize logging */
14. private static final Logger logger =
    Logger.getLogger(L3NewBrowserMapper.class);
15.
16. @Override
17. public void map(SessionBean input, AnalyticsOutputCollector<L3NewBrowserBean>
    outputCollector) throws IOException {
18. if(logger.isTraceEnabled()) {
19. logger.trace("mapping input bean '"+ input +"' to L3NewBrowserBean");
20. }
21.
22. L3NewBrowserBean output = new L3NewBrowserBean();
23. output.setDateid(input.getDateid());
24. output.setSiteid(input.getSiteid());
25. output.setBrowserid(input.getBrowserid());
26. output.setCount(1L);
27.
28.         // collect the output bean
29. outputCollector.collect(output);
30.   }
31. }
```

The L3NewBrowserMapper class code is analyzed as follows:

- The L3NewBrowserMapper class will extend the AbstractAnalyticsMapper class (Line 11) and override the map method (Lines 16-17).
- In the map method, every input SessionBean is transformed into L3NewBrowserBean by setting the value of L3NewBrowserBean from the SessionBean (Lines 19-25).
- The count property of the L3NewBrowserBean is set to 1L for every input bean (Line 26).
- Every L3NewBrowserBean created will be collected by the output collector (AnalyticsOutputCollector) (Line 29).
- Add debugging statements (Line 19).

4.1.2.5 Step 5: Adding Beans and Mappers to the Processor Definitions

To enable your newly coded beans and mapper classes, add them to the existing processor definitions. Adding a mapper is done by adding the mapper to the spring-mapper.xml files in the corresponding processor folder.

In this exercise you will be configuring:

- L3NewBrowserMapper
- L3NewBrowserBean

To add beans and mappers to the processor definitions

1. Configure L3NewBrowserMapper:

   a. Open the processors/sesprocessor/spring-mapper.xml file in a text editor.

   b. Add the **bean class** line (shown in bold type, below) to the spring-mapper.xml file:

```
<bean id="AnalyticsMapperConfigBean" class="java.util.ArrayList">
    <constructor-arg>
      <list>
       <bean id="clickstreamMapper"
class="com.fatwire.analytics.report.mapper.L3ClickstreamMapper"/>
<bean id="newBrowserMapper"
class="com.fatwire.analytics.report.mapper.L3NewBrowserMapper"/>
       <bean id="osMapper"
class="com.fatwire.analytics.report.mapper.L3OperatingsystemMapper"/>
       <bean id="sessionEntryidMapper"
class="com.fatwire.analytics.report.mapper.L3SessionEntryMapper"/>
       <bean id="sessionExitidMapper"
class="com.fatwire.analytics.report.mapper.L3SessionExitMapper"/>
       <bean id="ipMapper"
class="com.fatwire.analytics.report.mapper.L3IpMapper"/>
       <bean id="hostnameMapper"
class="com.fatwire.analytics.report.mapper.L3HostnameMapper"/>
       <bean id="jsMapper"
```

```
class="com.fatwire.analytics.report.mapper.L3JsMapper"/>
       <bean id="searchengineMapper"
class="com.fatwire.analytics.report.mapper.L3SearchengineMapper"/>
       <bean id="refererMapper"
class="com.fatwire.analytics.report.mapper.L3RefererMapper"/>
       <bean id="screenresMapper"
class="com.fatwire.analytics.report.mapper.L3ScreenresMapper"/>

       <bean id="sessionQuantilMapper"
class="com.fatwire.analytics.report.mapper.L3SessionQuantilMapper"/>

       <bean id="objectDurationMapper"
class="com.fatwire.analytics.report.mapper.L3ObjectDurationMapper"/>

       <bean id="engageMapper"
class="com.fatwire.analytics.report.mapper.L3EngageMapper"/>
      </list>
    </constructor-arg>
   </bean>
```

2. Configure L3NewBrowserBean:

   a. Open the processor/sesprocessor/spring-combiner_reducer.xml file in a text editor.

   b. Add the entry key line (shown in bold type, below) to the spring- combiner_ reducer.xml file:

```
<util:map id="AnalyticsCombinerReducerConfigBean"
map-class="java.util.HashMap">
<entry key="com.fatwire.analytics.domain.l3.L3ClickstreamBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3OperatingsystemBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3NewBrowserBean"
value-ref="analyticsBeanReducer"/>

<entry key="com.fatwire.analytics.domain.l3.L3SessionEntryBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3SessionExitBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3IpBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3HostnameBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3JsBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3SearchengineBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3RefererBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3ScreenresBean"
value-ref="analyticsBeanReducer"/>

<entry key="com.fatwire.analytics.domain.l3.L3SessionQuantilBean"
value-ref="analyticsBeanReducer"/>

<entry key="com.fatwire.analytics.domain.l3.L3ObjectDurationBean"
value-ref="analyticsBeanReducer"/>

<entry key="com.fatwire.analytics.domain.l3.L3EngageRecBean"
```

```
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3EngageRecSegBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3EngageRecSegObjBean"
value-ref="analyticsBeanReducer"/>
</util:map>
```

4.1.3 Configuring Database Injection

Follow these steps to configure the database injection:

1. Open the `processors/sesinjection/spring-combiner.xml` file in a text editor.
2. Add the entry key line (shown in bold type, below) to the spring-combiner.xml snippet:

```
<util:map id="AnalyticsCombinerConfigBean" map-class="java.util.HashMap">
    <entry key="com.fatwire.analytics.domain.l3.L3ClickstreamBean"
value-ref="analyticsBeanReducer"/>
    <entry key="com.fatwire.analytics.domain.l3.L3NewBrowserBean"
value-ref="analyticsBeanReducer"/>
    <entry key="com.fatwire.analytics.domain.l3.L3OperatingsystemBean"
value-ref="analyticsBeanReducer"/>
    <entry key="com.fatwire.analytics.domain.l3.L3SearchengineBean"
value-ref="analyticsBeanReducer"/>

<entry key="com.fatwire.analytics.domain.l3.L3SessionEntryBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3SessionExitBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3IpBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3HostnameBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3JsBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3SearchengineBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3RefererBean"
value-ref="analyticsBeanReducer"/>
    <entry key="com.fatwire.analytics.domain.l3.L3ScreenresBean"
value-ref="analyticsBeanReducer"/>

    <entry key="com.fatwire.analytics.domain.l3.L3SessionQuantilBean"
value-ref="analyticsBeanReducer"/>

    <entry key="com.fatwire.analytics.domain.l3.L3ObjectDurationBean"
value-ref="analyticsBeanReducer"/>

    <entry key="com.fatwire.analytics.domain.l3.L3EngageRecBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3EngageRecSegBean"
value-ref="analyticsBeanReducer"/>
<entry key="com.fatwire.analytics.domain.l3.L3EngageRecSegObjBean"
value-ref="analyticsBeanReducer"/>
</util:map>
```

3. Open the `processors/sesinjection/spring-reducer.xml` files in a text editor.

4. Add the **entry key** line (shown in bold type, below) to the `spring-reducer.xml` snippet:

```
<util:map id="AnalyticsReducerConfigBean" map-class="java.util.HashMap">
    <entry key="com.fatwire.analytics.domain.l3.L3ClickstreamBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3NewBrowserBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3OperatingsystemBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3SearchengineBean"
value-ref="databaseInjection"/>

    <entry key="com.fatwire.analytics.domain.l3.L3SessionEntryBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3SessionExitBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3IpBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3HostnameBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3JsBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3SearchengineBean"
value-ref="databaseInjection"/>
    <entry key="com.fatwire.analytics.domain.l3.L3RefererBean"
value-ref="databaseInjection"/>
        <entry key="com.fatwire.analytics.domain.l3.L3ScreenresBean"
value-ref="databaseInjection"/>

        <entry key="com.fatwire.analytics.domain.l3.L3SessionQuantilBean"
value-ref="databaseInjection"/>

        <entry key="com.fatwire.analytics.domain.l3.L3ObjectDurationBean"
value-ref="databaseInjection"/>

        <entry key="com.fatwire.analytics.domain.l3.L3EngageRecBean"
value-ref="databaseInjection"/>
       <entry key="com.fatwire.analytics.domain.l3.L3EngageRecSegBean"
value-ref="databaseInjection"/>
      <entry key="com.fatwire.analytics.domain.l3.L3EngageRecSegObjBean"
value-ref="databaseInjection"/>
      </util:map>
```

4.1.4 Integrating the New Analytics Job with the Existing Hadoop-Jobs Component

Once you have developed the new Analytics job, integrate the new job you developed with the existing hadoop-jobs component by recreating a jar file (`hadoop-jobs.jar`). Copy the new `hadoop-jobs.jar` file to the hadoop-jobs directory. Recreating the `jar` file enables the hadoop-jobs component to process the data captured by the new Analytics job you developed in this exercise.

To integrate the new Analytics job with the existing hadoop-jobs component

1. Create the `hadoop-jobs.jar` file.
2. Replace the existing `hadoop-jobs.jar` file, located in the hadoop-jobs installation directory, with the jar file you just created. Then, remove the `._tmp_hadoop-jobs.jar` file, which is the actual jar used by the run command. By

deleting this jar, the run command will rebuild it from the new `hadoop-jobs.jar`.

3. Run the hadoop-jobs component in order to process the data captured by the parameter (added in Chapter 2, "Exercise 1: Adding a New Parameter for Data Capture").

4.2 Next Steps

Chapter 5, "Exercise 4: Creating and Preparing a Report for Viewing" of this tutorial walks you through the process of creating a new report in the reporting interface. As an example, you will create the "NewBrowsers" report, which displays the number of visitors for each browser.

Note : The "NewBrowsers" report you will be creating in this tutorial, is a duplicate of the default "Browsers" report in your Analytics installation. For the purposes of this tutorial, the xml file and report name of the "Browsers" report you will be configuring, along with the bean and mapper class names, have been renamed to avoid overwriting the default "Browsers" report.

5

Exercise 4: Creating and Preparing a Report for Viewing

Your goal in this exercise is to learn how to configure a complete, usable Analytics report. To accomplish that, you will configure a custom report called "NewBrowsers" that will display the browsers that visitors used to gain access to the given site's page view within the reported time period.

> **Note:** The "NewBrowser" report you will be creating in this tutorial, is a duplicate of the default "Browsers" report in your Analytics installation. For the purposes of this tutorial, the xml file and report name of the "Browsers" report you will be configuring, along with the bean and mapper class names, have been renamed to avoid overwriting the default "Browsers" report.

The report will display data that has already been captured on the FirstSite II sample site and stored in the Analytics database.

When building the report, you will first create a simple report with a test module that displays a "Hello World" greeting. You will remove this test module when you begin adding features to your report.

> **Note:** Throughout this exercise, we assume that you are working exclusively with the FirstSite II sample site. Select this site whenever prompted in the Analytics or WebCenter Sites interfaces.

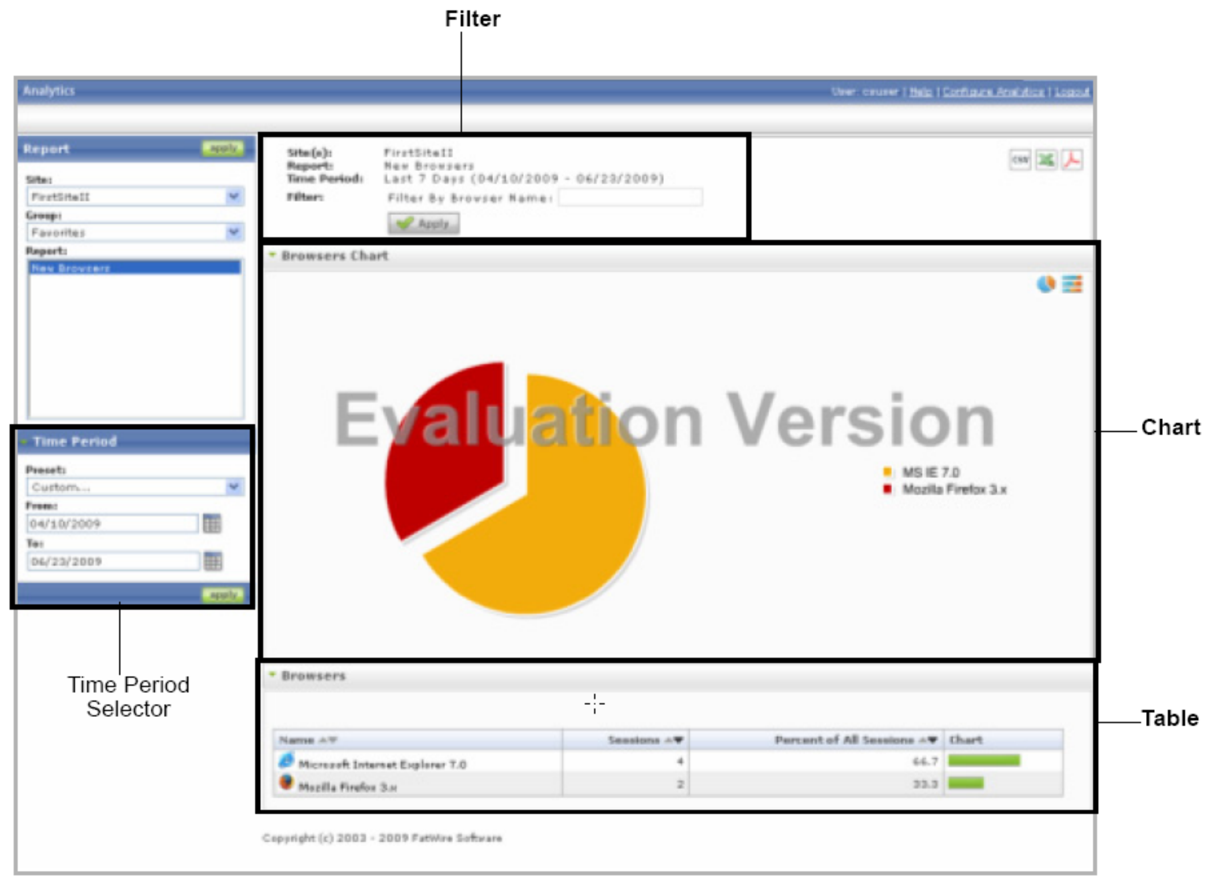
This exercise consists of the following sections:

- Section 5.1, "Report Design"
- Section 5.2, "Creating and Preparing the 'NewBrowsers' Report for Viewing"
- Section 5.3, "Adding a Filter"

5.1 Report Design

An Analytics report is composed of modules. A module is a piece of code that implements a specific feature in the report, such as a table, a chart, or a filter. For example, the "NewBrowsers" report you will build in this exercise contains the following features: filter, chart, table, and time period selector, shown in Figure 5–1.

Figure 5–1 New Browsers Report



Analytics

User: csuser | Help | Configure Analytics | Logout

Report apply

Site: FirstSiteII

Group: Favorites

Report: New Browsers

Time Period

Preset: Custom...

From: 04/10/2009

To: 06/23/2009

apply

Site(s): FirstSiteII
Report: New Browsers
Time Period: Last 7 Days (04/10/2009 - 06/23/2009)
Filter: Filter By Browser Name:

Apply

Browsers Chart

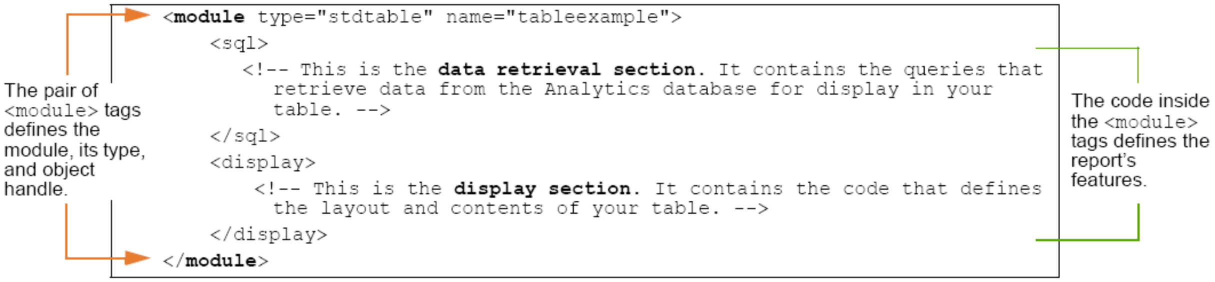
MS IE 7.0
Mozilla Firefox 3.x

Browsers

Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7.0	4	66.7	🟩🟩
Mozilla Firefox 3.x	2	33.3	🟩

Copyright (c) 2003 - 2009 FatWire Software

Each of these features, except for the time period selector, is implemented as a separate and, in most cases, self-contained module. (Some features, such as filters, require you to modify the code of other modules in order to function.) Let's look at the structure of the table module. It has a module declaration, a data retrieval section, and a display section:



```
<module type="stdtable" name="tableexample">
    <sql>
      <!-- This is the data retrieval section. It contains the queries that
        retrieve data from the Analytics database for display in your
        table. -->
    </sql>
    <display>
       <!-- This is the display section. It contains the code that defines
        the layout and contents of your table. -->
    </display>
</module>
```

The pair of <module> tags defines the module, its type, and object handle.

The code inside the <module> tags defines the report's features.

Modules that query the database to display data contain the `<sql>` and `<display>` tags, as shown in the sample code above. Modules that do not display data, such as filter modules, contain the following structure:

```
<filter name="filter-browsername" required="false" type="text" captions="name"
key="filter-browsername" />
```

The chart module has the same structure as described above.

Tables, charts, and other features of your report contain areas (such as headings, field names and column heads) that must be filled in by having their values defined in a property file accessible within your application server's classpath. For example, the following statement defines the heading for the "Browser Name" column in your table:

```
report_newbrowser_module_browser_column_broname=Name
```

Note: For your reference, the code for the "NewBrowsers" report, annotated module by module, is shown in Section 5.1.1, "'NewBrowsers' Report Code."

Now that you know how reports are built, let's go ahead and create the "NewBrowsers" report. Continue to Section 5.2, "Creating and Preparing the 'NewBrowsers' Report for Viewing."

5.1.1 'NewBrowsers' Report Code

Below is the code that powers the "NewBrowsers" report. The code is annotated for your reference.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--  <!DOCTYPE report SYSTEM "report.dtd">  -->

<report type="std" visible="true" name="newbrowser" >

<!-- Global parameters (such as sitenames), are defined here-->

<globalparam name="sitenames" />

<!-- After declaring global parameters, start declaring several request parameters
like rgrpid, timepreset, sdate, edate etc. sdate, edate enable the time range
selector in the report-->
<param type="request" name="rgrpid" key="rgrpid" required="false" />
<param type="request" name="timepreset" key="timepreset" required="false">
    <restriction type="listvalue">
       <restrictionvalue value="yesterday" />
               <restrictionvalue value="lastweek" />
               <restrictionvalue value="lastmonth" />
               <restrictionvalue value="last7days" />
               <restrictionvalue value="last4weeks" />
               <restrictionvalue value="last12months" />
       </restriction>
</param>
<param type="request" name="siteid" key="siteid" required="true" />
<param type="request" name="sdate" key="sdate" required="true">
    <restriction type="isDate" value="dd.MM.yyyy" />
</param>
<param type="request" name="edate" key="edate" required="true">
    <restriction type="isDate" value="dd.MM.yyyy" />
</param>

<!—a filter is getting configured here, that will filter the report data by
browser name. -->
<filter name="filter-browsername"
        required="false"
        type="text"
        captions="name"
        key="filter-browsername"
/>

<!—the following parameter will generate the sum of all sessions that have come
from different browsers-->
<param type="sql" name="browsertotal" >
    <query>
       select
          nvl(sum(count), 0) as count
       from
          help_dates dates
          join l3_datexsitexnewbrowserxcount l3 on (dates.id = l3.dateid)
       where<![CDATA[
          siteid = #siteid# and
          dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
          dates.dat  < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>
    </query>
</param>
<!-- The chart module defines your pie chart. -->
<module type="stdchart" name="chartbrowser">
    <param type="request" name="charttype" key="chartbrowser_charttype"
       default="pie_labeled">
       <restriction type="listvalue">
             <restrictionvalue value="pie_labeled" />
             <restrictionvalue value="bar_labeled" />
       </restriction>
    </param>

    <param type="string" name="chart_max_display" value="10"/>
    <param type="string" name="chart_display_rest" value="true"/>
    <sql>
    <query>
       select
           bro.id as broid,
           bro.name as broname,
           nvl(count, 0) as count,
           <equals name="browsertotal" value="0">
              0 as percent
           </equals>
           <notEquals name="browsertotal" value="0">
              (nvl(count, 0)/#browsertotal#)*100 as percent
           </notEquals>
       from
           l2_browser bro
           join (
           select
              bro2.id as broid,
              bro2.name as broname,
              sum(count) as count
           from
              help_dates dates
              join l3_datexsitexnewbrowserxcount l3 on
                      (dates.id = l3.dateid)
                      join l2_browser bro2 on (l3.browserid = bro2.id
           where<![CDATA[
              siteid = #siteid# and
              dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
              dates.dat  < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>
           group by
              bro2.id, bro2.name
           ) on (bro.id = broid)
           <dynamicOrderBy default="count-desc" />
    </query>
    </sql>
    <display type="html">
       <value name="xaxis" type="string">
           <valueparam name="format" parse="false"/>
           <valueparam name="value" value="#broname#" />
       </value>

       <value name="yaxis" type="number">
              <valueparam name="format" parse="false" value="#####0" />
              <valueparam name="value" value="#count#" />
       </value>
    </display>

    </module>
<!-- This module defines the table that will show statistics for each browser to
the target page.-->
<module type="stdtable" name="browser" >
    <sql>
       <query>
           select
              bro.id as broid,
              bro.name as broname,
              bro.iconpath as broiconpath,
              nvl(count, 0) as count,
              <equals name="browsertotal" value="0">
                 0 as percent
              </equals>
              <notEquals name="browsertotal" value="0">
                 (nvl(count, 0)/#browsertotal#)*100 as percent
              </notEquals>
           from
              l2_browser bro
              join (
              select
                 bro2.id as broid,
                 bro2.name as broname,
                 sum(count) as count
             from
                 help_dates dates
                 join l3_datexsitexnewbrowserxcount l3 on (dates.id = l3.dateid)
                 join l2_browser bro2 on (l3.browserid = bro2.id)
              where<![CDATA[
<!-- The CDATA statement allows the time period selector to limit the data
displayed in the table to a specific time period. -->
                 siteid = #siteid# and
                 dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
                 dates.dat < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>

<!-- The <notNull> tag ensures that its contents are added to the main query only
if they are not null. -->

                 <notNull name="filter-browsername">
                   and lower(bro2.name) like lower(replace('#filter-
                   browsername#','*','%'))
                 </notNull>
           group by
```

```
              bro2.id, bro2.name

           ) on (bro.id = broid)
           <dynamicOrderBy default="count-desc" />
       </query>
       <count type="simple" />
    </sql>

    <display type="html">
    <!-- Table columns are defined here.
    Each <column> statement defines one column.-->

    <column name="broname" columntype="text" sortrscolumn="broname">
           <value type="image">
              <valueparam name="src" value="#imgpath#/browser/#broiconpath#" />
              <valueparam name="alt" value="#broname#" />
              <valueparam name="text" value="#broname#" />
              <valueparam name="width" value="16" />
              <valueparam name="height" value="16" />
           </value>
           <value type="string">
              <valueparam name="value" value=" #broname#" />
           </value>
           </column>
           <column name="count" columntype="number" sortrscolumn="count">
              <value type="number">
                 <valueparam name="format" parse="false" value="#####0" />
                 <valueparam name="value" value="#count#" />
              </value>
           </column>
           <column name="percent" columntype="number" sortrscolumn="count">
              <value type="number">
                 <valueparam name="format" parse="false" value="#####0.0" />
                 <valueparam name="value" value="#percent#" />
              </value>
           </column>
           <column name="chart" columntype="string">
              <value type="chart">
                 <valueparam name="format" parse="false" value="#####0" />
                 <valueparam name="value" value="#percent#" />
                 <valueparam name="maxvalue" value="100" />
                 <valueparam name="width" value="100" />
                 <valueparam name="image" value="#imgpathstyle_branding#graph_
                  blue.gif" />
              </value>
           </column>
    </display>
</module>
</report>
```

5.2 Creating and Preparing the 'NewBrowsers' Report for Viewing

You will now create the "NewBrowsers" report (shown in Section 5.1, "Report Design").

Note: In this exercise, you will add features to your report in the order shown below. Once you are familiar with report code, you can build your report in the order that's most convenient for you. Be aware, however, that certain features, such as filters, require you to modify the code in other modules.

1. Section 5.2.1, "Creating and Registering the Report File." The first task is to create the XML file that will hold the report code and register it with Analytics.
2. Section 5.2.2, "Adding a Table." The table displays a set of statistics for each browser (Browser Name; Sessions; Percent of all sessions; and the chart that shows the percent graphically).
3. Section 5.2.3, "Adding a Time Period Selector." This selector allows you to limit the data displayed in the report to a specific time period. (This feature does not require a module.)
4. Section 5.3, "Adding a Filter." Using filters, you can restrict the data displayed in the report to specific browsers and session counts.
5. Section 5.3.3, "Adding a Chart." The "NewBrowsers" chart shows how often a given browser was used to access the site's page view during the reported time period.
6. Section 5.3.4, "Testing the Completed 'NewBrowsers' Report." Test your report to make sure it looks and behaves as intended.

5.2.1 Creating and Registering the Report File

Your first task is to create the foundation for the report – the XML file that will hold the report code. In this section you will create a report file containing the "Hello World" test module that will display only text, and you will register the report with Analytics.

The steps for creating and registering a report file are:

Section 5.2.1.1, "Step 1: Create the XML File"

Section 5.2.1.2, "Step 2: Place the XML File in the Analytics Reports Directory"

Section 5.2.1.3, "Step 3: Label the Report Components"

Section 5.2.1.4, "Step 4: Make the Report Available to Analytics Users"

Section 5.2.1.5, "Step 5: Test the New Report"

5.2.1.1 Step 1: Create the XML File

The first step is to create an XML file called `report_newbrowser.xml`.

1. In a text editor, create a new file named `report_newbrowser.xml`.
2. Paste the following code into the file:

```
<report type="std" name="newbrowser">

<param type="request" name="rgrpid" key="rgrpid" required="false" />
<param type="request" name="siteid" key="siteid" required="true" />

<module type="simpletext" name="helloworld" >
<display type="html">
<text>
```

```
<![CDATA[
  <div>
    Hello World! FatWire is greeting you!
  </div>
]]>
</text>
</display>
</module>
</report>
```

3. Save and close the file.

5.2.1.2 Step 2: Place the XML File in the Analytics Reports Directory

Copy the report file to the directory defined by the `report_instdir` parameter in the `global.xml` configuration file, so that Analytics can access the report file. For information on the `global.xml` file and its location, see the *Oracle Fusion Middleware WebCenter Sites Installation Guide*.

5.2.1.3 Step 3: Label the Report Components

You must now define the labels for areas such as the report name, module headings, and so on. The labels you define will be displayed in the reporting interface when the report is generated.

> **Note:** Component labeling in Analytics is implemented using the `ResourceBundle` Java class. For more information on this class, see the following URL:
>
> `http://java.oracle.com/developers/technicalArticles/Intl/ResourceBundles/`

1. In a text editor, create a new file named `NewBrowsersLocalization.properties`.
2. Save the file in a directory that is within your application server's classpath. In this exercise, place it in the `WEB-INF/classes` directory inside the `analytics` web application directory on your application server.
3. In the property file, add a statement for each parameter string that you want to label in the reporting interface. For now, you will give the report a name, and give the "Hello World" module a heading. Add the following statements to the property file:

```
report_newbrowser=New Browsers
report_newbrowser_module_helloworld=My first report in Analytics!
```

> **Note:** You must use the following syntax when adding statements to the property file:
>
> - Report name: `report_`*`reportName`*
> - Module names: `report_`*`reportName`*`_module_`*`moduleName`*
> - Column heads in a table: `report _`*`reportName`*`_module_column_`*`columnName`*

4. When you have added the two statements, save and close the file.
5. Register the property file with Analytics by adding its name to the `global.xml` file as follows:
   a. Open the `global.xml` file (usually located in the `WEB-INF/classes` directory inside the `analytics` web application directory on your application server) in a text editor.
   b. Locate the `<locales></locales>` section and insert the following statement inside it:

      ```
      <locale name="NewBrowserLocalization" />
      ```
   c. Save and close the file.
6. Restart the application server for your changes to take effect.

5.2.1.4 Step 4: Make the Report Available to Analytics Users

In order to make the new report available to your Analytics users, you must register the report with Analytics, add it to a report group, and grant users access to the report.

1. Log in to the Analytics administration interface as `csuser/csuser` via the following URL:

   ```
   http://<hostname>:<port>/analyticsadmin/Admin?advmode=true
   ```

   > **Note:** The `advmode=true` parameter gives you access to advanced configuration options normally unavailable in the administration interface.

2. Register the new report with Analytics:
   a. In the "Report" section of the left-hand pane, click **Register**.
   b. Analytics displays the "Add/Edit Report" form.
   c. In the **Name** field, enter `newbrowser`.

      This is the object handle for your report. The value you enter here will be used to refer to the report in code (you used this name when you labeled the components of your report in Section 5.2.1.3, "Step 3: Label the Report Components"). This name will **not** be displayed in the reporting interface.
   d. In the **Config-file path** field, enter `report_newbrowser.xml`.
   e. Click **Save**.
3. Add the new report to the "Favorites" report group:
   a. In the "Report to Reportgroup" section of the left-hand pane, click **Assign**.

      Analytics displays the "Report Group" form.
   b. In the "Report Group" drop-down list, select **user**.
   c. In the list of reports, select the check box next to **newbrowser**.
   d. In the "Report Group" drop-down list, select **favorite**.
   e. In the list of reports, select the check box next to **newbrowser**.
   f. Click **Save**.
4. Grant users access to the report:

a. In the "Access Rights" section of the left-hand pane, click **Grant rights**.

b. Analytics displays the "User Groups" form.

c. In the form, select the check box next to **newbrowser**.

d. Click **Save**.

5.2.1.5 Step 5: Test the New Report

Generate the report to check that it is behaving as intended.

1. Log in to the Analytics reporting interface as `csuser/csuser` via the following URL:

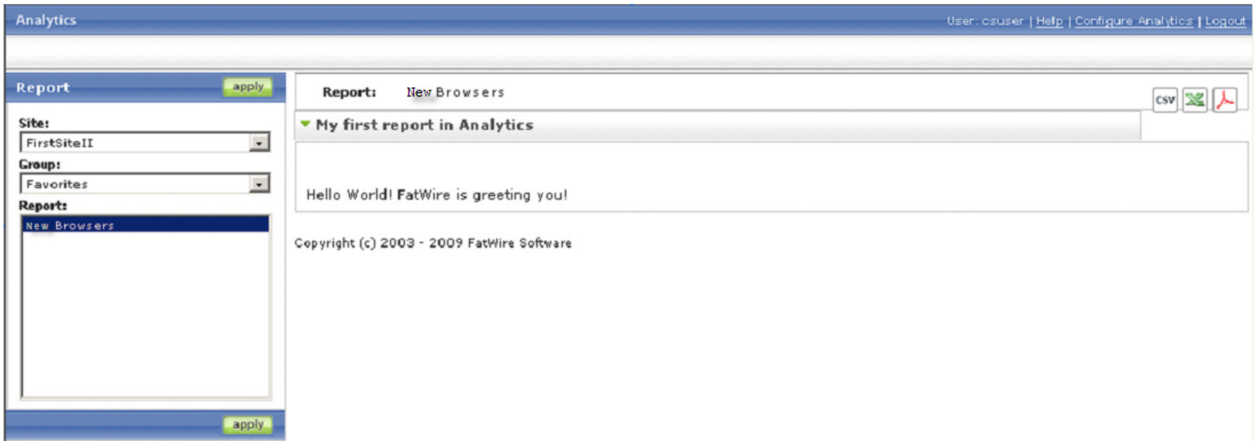
   ```
   http://<hostname>:<port>/analytics/Reports.do
   ```

2. In the "Group" drop-down list in the left-hand pane, select **Favorites**.

3. In the "Report" list, double-click **New Browsers**.

   Analytics displays your report. The report should look similar to the one in Figure 5–2:

Figure 5–2 Generated New Browsers Report



Hello World! FatWire is greeting you!

Copyright (c) 2003 - 2009 FatWire Software

4. If your report does not appear as in the above figure, or if an error is displayed, retrace your steps and check your code for errors.

   Once you have verified your report is behaving as intended, continue on to Section 5.2.2, "Adding a Table."

5.2.2 Adding a Table

A table is the primary way of presenting data in an Analytics report. In this section, you will add a table to your "NewBrowsers" report that will display several categories of information on different browsers.

How Do I Add a Table?

You add a table to your report by inserting a table module into your report's code. A table module has two sections (as shown in the example in the Section 5.1, "Report Design"):

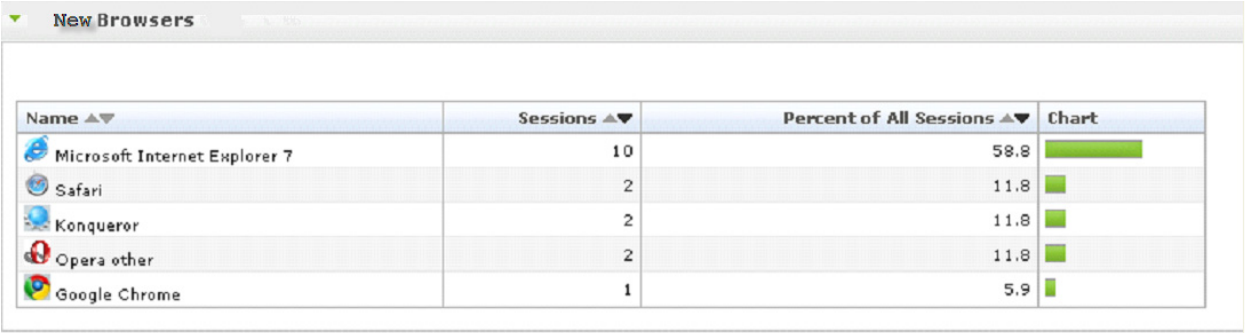
- **Data retrieval section** (enclosed within the `<sql>` tag, lines b–ao below). This section contains the SQL code that retrieves the data you want to display in your table from the Analytics database.
- **Display section** (enclosed within the `<display>` tag, lines aq–by below). This section contains the code that formats and displays the data retrieved from the Analytics database by the code in the data retrieval section. The code in the display section defines the layout of the table and the formatting applied to each column.

In this section, you will add the "NewBrowsers" table to your report. The table will display the following columns:

- Name
- Sessions
- Percent of All Sessions
- Chart

The table will look like the one in Figure 5–3:

Figure 5–3 Table for the "New Browsers" Report



Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7	10	58.8	🟩
Safari	2	11.8	🟩
Konqueror	2	11.8	🟩
Opera other	2	11.8	🟩
Google Chrome	1	5.9	🟩

To add a table to your report

1. Open your `report_newbrowser.xml` file and replace the "Hello World" test module with the following code:

```
a.  <module type="stdtable" name="browser" >
b.  <sql>
c.    <query>
d.      select
e.        bro.id as broid,
f.        bro.name as broname,
g.        bro.iconpath as broiconpath,
h.        nvl(count, 0) as count,
i.        <equals name="browsertotal" value="0">
j.          0 as percent
k.        </equals>
l.        <notEquals name="browsertotal" value="0">
m.          (nvl(count, 0)/#browsertotal#)*100 as percent
n.        </notEquals>
```

```
    from
      l2_browser bro
      join (
    select
      bro2.id as broid,
      bro2.name as broname,
      sum(count) as count
    from
      help_dates dates
      join l3_datexsitexnewbrowserxcount l3 on (dates.id = l3.dateid)
      join l2_browser bro2 on (l3.browserid = bro2.id)
    where<![CDATA[
      siteid = #siteid# and
      dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
      dates.dat < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>
      <notNull name="filter-browsername">
      and lower(bro2.name) like lower(replace('#filter-
      browsername#','*','%'))
      </notNull>
    group by
      bro2.id, bro2.name

    ) on (bro.id = broid)
    <dynamicOrderBy default="count-desc" />
  </query>
  <count type="simple" />
</sql>

<display type="html">
<column name="broname" columntype="text" sortrscolumn="broname">
  <value type="image">
    <valueparam name="src" value="#imgpath#/browser/#broiconpath#" />
    <valueparam name="alt" value="#broname#" />
    <valueparam name="text" value="#broname#" />
    <valueparam name="width" value="16" />
    <valueparam name="height" value="16" />
  </value>
  <value type="string">
    <valueparam name="value" value=" #broname#" />
  </value>
</column>
<column name="count" columntype="number" sortrscolumn="count">
```

```
be.   <value type="number">
bf.   <valueparam name="format" parse="false" value="#####0" />
bg.   <valueparam name="value" value="#count#" />
bh.   </value>
bi. </column>
bj. <column name="percent" columntype="number" sortrscolumn="count">
bk.   <value type="number">
bl.   <valueparam name="format" parse="false" value="#####0.0" />
bm.   <valueparam name="value" value="#percent#" />
bn.   </value>
bo. </column>
bp. <column name="chart" columntype="string">
bq.   <value type="chart">
br.   <valueparam name="format" parse="false" value="#####0" />
bs.   <valueparam name="value" value="#percent#" />
bt.   <valueparam name="maxvalue" value="100" />
bu.   <valueparam name="width" value="100" />
bv.   <valueparam name="image" value="#imgpathstyle_branding#graph_blue.gif" />
bw.   </value>
bx. </column>
by. </display>
bz. </module>
```

The code above is analyzed as follows:

– In the data retrieval section, a select query retrieves the required rows (broid, broname, broiconpath, count, percent) from the L3_DATEXSITEXNEWBROWSERXCOUNT and L2_BROWSER tables (lines d-ao). You access these values from the display section by calling them enclosed within hash (#) signs, (for example, value="#broname#" in line ba).

– The <dynamicOrderBy> tag in line al defines how the rows should be ordered. In our example, the value "count-desc" indicates the rows are ordered by the number of sessions, in descending order.

– In the display section, the <column> statements define columns that constitute your table (lines ar-bc, bd-bi, bj-bo, and bp-bx). Each column definition is responsible for one column in the table and takes the following parameters:

  name (required) – specifies the object handle for the column. You use this handle to specify a label for the column in the NewBrowserLocalization.properties file.

  columntype (required) – specifies the type of the column. Your table uses number, text, and string columns.

  sortrscolumn (required) – specifies whether the column should be sortable, and if so, how the data should be sorted.

Note: A sortable column displays a small arrow in its heading that indicates the direction in which its values are being sorted. Clicking the arrow reverses the sort order.

– The following columns are defined in our example:

The broname column of type text (lines ar-bc) displays the name of the browser along with the corresponding icon. Columns of type text may have a value parameter (lines as-bb) of type image, number, string, date etc. (parameter type is defined in the value tag). Here the broname column is composed of two different types of values namely image and string. The value of type image can have several value-parameters, including, but not limited to: src (source path), alt (alternative text), width, and height, which collectively define the rendition of the image on the report.

The count column (lines bd-bi) of type number displays the number of sessions that have come from the browser. Columns of type number must have a value parameter (lines as-bb) of type number (parameter type is defined in the value tag) which specifies what data should be displayed (value parameter, line ba, and how the data should be formatted (format parameter, line az).

The percent column (lines bj-bo) of type number displays the percent of sessions with respect to all sessions that have come from all different browsers.

The chart column (lines bp-bx) of type chart is the graphic representation of the percent column.

2. Save and close the file.

3. Label the table and column headings as follows:

   a. Open the NewBrowserLocalization.properties file (in this exercise, located in the WEB-INF/classes directory in the analytics application directory on your application server) in a text editor.

   b. Add the following statements at the end of the file:

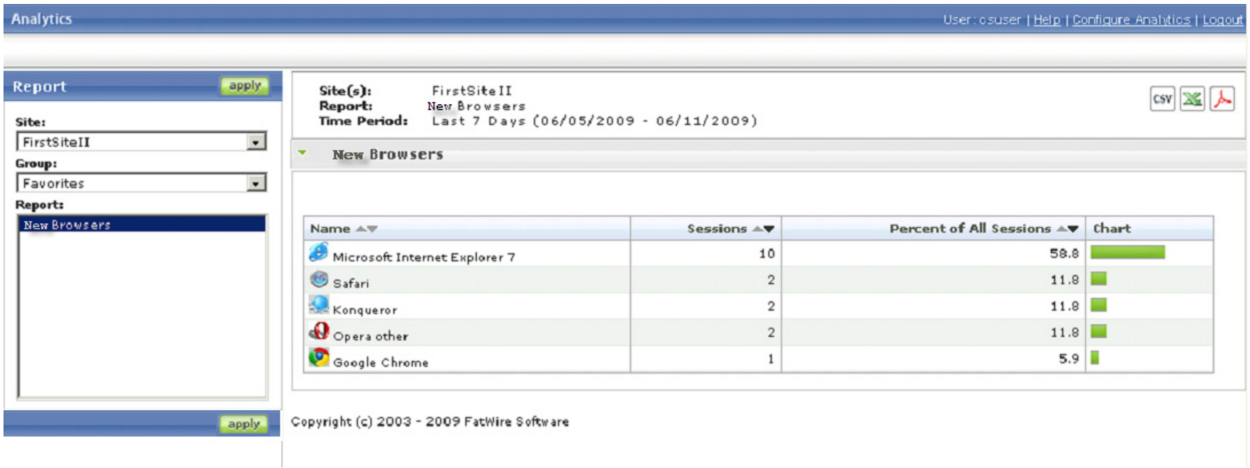
```
# Report New Browser
report_newbrowser=Browsers
report_newbrowser_module_browser_column_broname=Name
report_newbrowser_module_browser_column_count=Sessions
report_newbrowser_module_browser_column_percent=Percent of All Sessions
report_newbrowser_module_browser_column_chart=Chart
report_newbrowser_module_browser=Browsers
```

   c. Save and close the file.

   d. Restart your application server for your changes to take effect.

4. Test your report. At this point, it should look similar to the one in Figure 5–4:

Figure 5–4 "New Browsers" Report Displaying the Added Table



Site:
FirstSiteII

Group:
Favorites

Report:
New Browsers

Site(s): FirstSiteII
Report: New Browsers
Time Period: Last 7 Days (06/05/2009 - 06/11/2009)

New Browsers

Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7	10	58.8	🟩
Safari	2	11.8	🟩
Konqueror	2	11.8	🟩
Opera other	2	11.8	🟩
Google Chrome	1	5.9	🟩

Copyright (c) 2003 - 2009 FatWire Software

5. If your report does not appear as in Figure 5–4, or if an error is displayed, retrace your steps and check your code for errors.

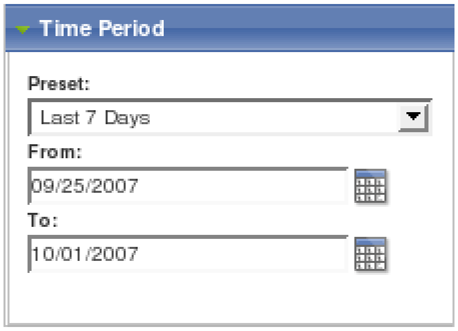
6. Continue on to Section 5.2.3, "Adding a Time Period Selector."

5.2.3 Adding a Time Period Selector

Now that your "NewBrowsers" report displays data, you will add a time period selector. This selector is a very useful feature, as it allows you to limit the amount of data displayed in the report to a specific time period. For example, you might want to view visitor activity that happened during a specific day or even hour, instead of all activity captured to date.

The time period selector will appear in the navigation pane in the reporting interface when you access the report:

Figure 5–5 Time Period Selector



Preset:
Last 7 Days

From:
09/25/2007

To:
10/01/2007

To enable time period selection in your report

1. Open the `report_newbrowser.xml` file (located in the Analytics reports directory described in Section 5.2.1.2, "Step 2: Place the XML File in the Analytics Reports Directory") in a text editor.

2. Locate the global parameter section and add the parameters (shown in bold type, below):

```
<param type="request" name="rgrpid" key="rgrpid" required="false" />
      <param type="request" name="timepreset" key="timepreset"
required="false">
```

```
   <restriction type="listvalue">
     <restrictionvalue value="yesterday" />
     <restrictionvalue value="lastweek" />
     <restrictionvalue value="lastmonth" />
     <restrictionvalue value="last7days" />
     <restrictionvalue value="last4weeks" />
     <restrictionvalue value="last12months" />
   </restriction>
</param>
<param type="request" name="siteid" key="siteid" required="true" />
<param type="request" name="sdate" key="sdate" required="true">
   <restriction type="isDate" value="dd.MM.yyyy" />
</param>
<param type="request" name="edate" key="edate" required="true">
   <restriction type="isDate" value="dd.MM.yyyy" />
</param>
```

The new parameters cause the "Time Period" panel to appear in the left-hand navigation pane when the report is accessed.

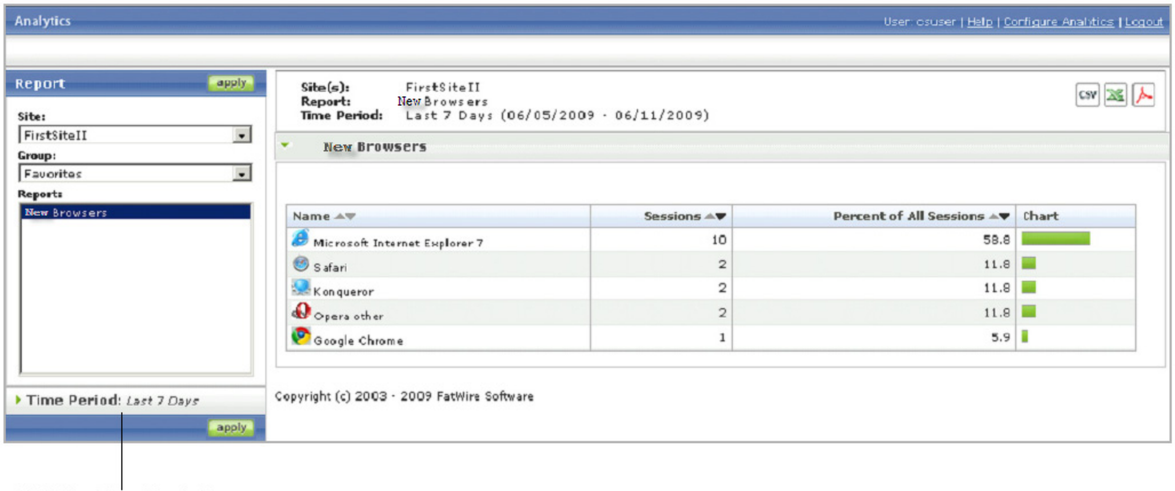
3. Insert the code (shown in bold type, below) into the data retrieval section of the table module at the exact locations shown:

```
select
bro.id as broid,
bro.name as broname,
bro.iconpath as broiconpath,
nvl(count, 0) as count,
<equals name="browsertotal" value="0">
  0 as percent
</equals>
<notEquals name="browsertotal" value="0">
  (nvl(count, 0)/#browsertotal#)*100 as percent
</notEquals>
from
l2_browser bro
join (
  select
  bro2.id as broid,
  bro2.name as broname,
  sum(count) as count
  from
  help_dates dates
  join l3_datexsitexnewbrowserxcount l3 on (dates.id = l3.dateid)
  join l2_browser bro2 on (l3.browserid = bro2.id)
  where<![CDATA[
        siteid = #siteid# and
        dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
        dates.dat  < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>
  group by
  bro2.id, bro2.name
) on (bro.id = broid)
```

4. Save and close the file.

5. Test your report. The report should now look like the one in Figure 5–6:

Figure 5–6 New Browsers Resport Displaying the Time Period Selector



Site: FirstSiteII

Group: Favorites

Reports: New Browsers

Time Period: *Last 7 Days*

Site(s): FirstSiteII
Report: New Browsers
Time Period: Last 7 Days (06/05/2009 - 06/11/2009)

New Browsers

Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7	10	58.8	🟩
Safari	2	11.8	🟩
Konqueror	2	11.8	🟩
Opera other	2	11.8	🟩
Google Chrome	1	5.9	🟩

Copyright (c) 2003 - 2009 FatWire Software

Click the **Time Period** bar to expand the time period selector panel.

6. Test the time period selector.

   a. Click the **Time Period** bar to expand the time period selector panel.

   b. Select the desired start date and end date.

   c. Click **Apply**.

   The data displayed in the table should change depending on the selected time period. If it does not, or if an error is displayed, retrace your steps and check your code for errors.

7. Continue on to Section 5.3, "Adding a Filter."

5.3 Adding a Filter

In the previous section you added a time period selector to your report to enable restriction of the displayed data to a specific time period. However, what if you want to restrict the displayed data by a parameter other than time? Your solution is to add a data filter.

5.3.1 What Does a Filter Do?

A filter module adds the "Filter by" field in the summary section of the report, and allows the users to:

- Search for a specific data point or value (for example, a specific browser).
- Restrict the displayed data to a specific range of a particular parameter (such as the number of recorded sessions)

You will add the following filter to your "NewBrowsers" report:

- **A "Browser Name"** filter, which will allow you to search for specific browsers or to restrict the displayed data to a specific range of browser names.

5.3.2 How Do I Add a Filter?

To implement a filter, you must:

- Add the filter module that defines your filter(s) to the report code.
- Modify the data retrieval sections of your display modules (such as table or chart modules) to support query filtering.

To add filters to your report

1. Open the `report_newbrowser.xml` file (located in the Analytics reports directory described in Section 5.2.1.2, "Step 2: Place the XML File in the Analytics Reports Directory") in a text editor.
2. Add the following module code after the table module but before the closing `</report>` element:

```
<filter name="filter-browsername" required="false" type="text" captions="name"
key="filter-browsername" />
```

The code above is analyzed as follows:

> **Note:** For detailed information on the tags shown in this example, and the parameters they take, see the *Oracle Fusion Middleware WebCenter Sites: Analytics Tag Reference*.

You define the filter by using a `filter` tag, which takes the following parameters:

- `name` (required) – specifies an object handle for the filter. The following conditions apply to this parameter:
  - The value of this parameter must begin with `filter-` so that Analytics treats this definition as a filter definition.
  - The value must be identical to the value of the `key` parameter (explained below).

> **Note:** If either of these conditions is not met, the filter will not function.

- `required` (required) – specifies whether the `key` parameter (explained below) must be assigned a value for the report to function. In our example, `required` is set to `false`.
- `type` (required) – specifies the type of the filter and, simultaneously, the filter's input method (for example, how the filter will be presented in the reporting interface).
- In this example, you are using a filter of type `text`, which manifests the filter as a text field into which the user can enter one or more filtering criteria. Other available types include `yesno`, `radio`, `dbselect`, `date`, and others.
- `key` (optional, see `required` above) – specifies the name used for the parameter in the URL of the report page to pass the value of the filter entered by the user when the user clicks **Apply**. The following conditions apply to this parameter:
  - The value of this parameter must begin with `filter-` so that Analytics treats this definition as a filter definition.

- The value must be identical to the value of the `name` parameter (explained above).

Note: If either of these two conditions is not met, the filter will not function.

3. Insert the code (shown in bold type, below) into the data retrieval section at the exact locations shown:

```
select
bro.id as broid,
bro.name as broname,
bro.iconpath as broiconpath,
nvl(count, 0) as count,
<equals name="browsertotal" value="0">
 0 as percent
</equals>
<notEquals name="browsertotal" value="0">
 (nvl(count, 0)/#browsertotal#)*100 as percent
</notEquals>
from
l2_browser bro
join (
  select
  bro2.id as broid,
  bro2.name as broname,
  sum(count) as count
  from
  help_dates dates
  join l3_datexsitexnewbrowserxcount l3 on (dates.id = l3.dateid)
  join l2_browser bro2 on (l3.browserid = bro2.id)
  where<![CDATA[
        siteid = #siteid# and
        dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
        dates.dat  < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>

  <notNull name="filter-browsername">
    and lower(bro2.name) like lower(replace('#filter-browsername#','*','%'))
  </notNull>

  group by
  bro2.id, bro2.name
) on (bro.id = broid)
```

The code above is analyzed as follows:

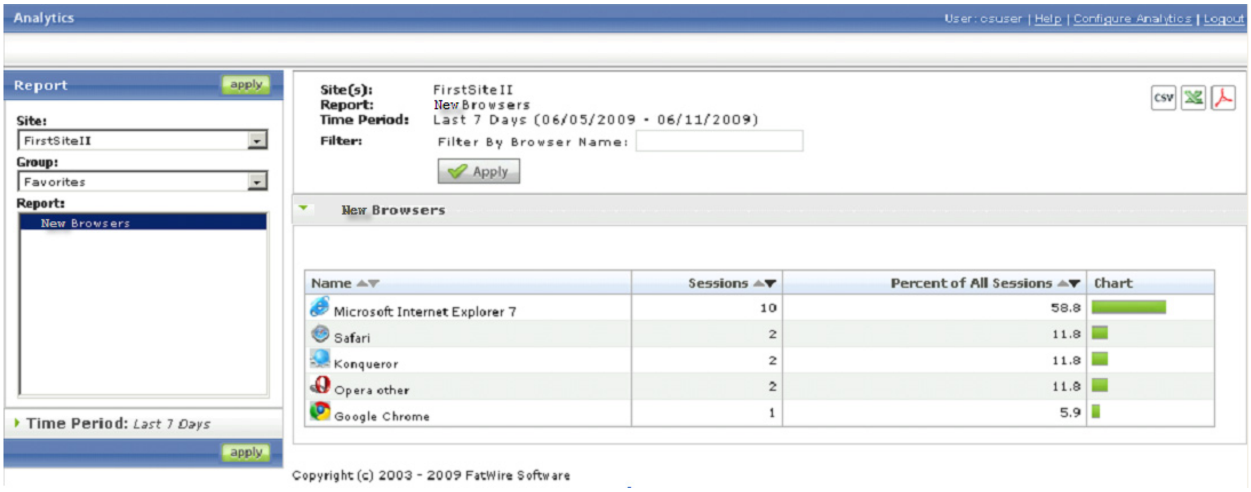
- The inserted code limits the results returned by the SQL query based on the criteria provided to the filter (for example, the filter value).
- The code inside each `<notNull>` tag is added to the query only if the value of the parameter called by the `<notNull>` tag is not null.

4. Save and close the report file. Open the `NewBrowserLocalization.properties` file (in this exercise, located in the `WEB-INF/classes` directory in the `analytics` application directory on your application server) in a text editor. Add the following line to the end of the file:

```
report_newbrowser_filter-browsername=Filter By Browser Name
```

5. Test your report. The "Filter" field should appear in the summary section at the top of the report, as shown in Figure 5–7:

Figure 5–7 "New Browsers" Report Displaying the Filter field



Site:
FirstSiteII

Group:
Favorites

Report:
New Browsers

Time Period: *Last 7 Days*

Site(s): FirstSiteII
Report: New Browsers
Time Period: Last 7 Days (06/05/2009 - 06/11/2009)
Filter: Filter By Browser Name:

Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7	10	58.8	🟩
Safari	2	11.8	🟩
Konqueror	2	11.8	🟩
Opera other	2	11.8	🟩
Google Chrome	1	5.9	🟩

Copyright (c) 2003 - 2009 FatWire Software

If your report does not appear as in the above figure, or if an error is displayed, retrace your steps and check your code for errors.

6. Continue on to Section 5.3.3, "Adding a Chart."

5.3.3 Adding a Chart

The final step in this exercise will be to add a chart to your "NewBrowsers" report. A chart allows you to graphically present statistical data in your report.

You add a chart to your report by inserting a chart module into your report's code. A chart module is structured in a way similar to the table module. It has two sections:

- **Data retrieval section** (see step 5). This section contains the SQL code that retrieves the data you want to display in your pie chart from the Analytics database.
- **Display section** (enclosed within the `<display>` tag in step 6). This section contains the code that formats and displays the data retrieved from the Analytics database by the code in the data retrieval section. The code in the display section defines the type of the chart and assigns data to the chart's axes.

In our example, you will create a pie chart (XY chart, refer to step 2 below) displaying session-count of all browsers in percentage format. Each section of the pie chart will display browser data, along with a legend section at the right of the chart. If you hover the mouse on the pie section, the percentage of session count will be displayed along with the browser's name.

To add the chart to your report

1. Open the `report_newbrowser.xml` file (located in the Analytics reports directory described in Section 5.2.1.2, "Step 2: Place the XML File in the Analytics Reports Directory") in a text editor.

2. Add the following module declaration at the beginning of the report code, right after the last global parameter definition, but before the table module. This will cause the chart to appear above the table when the report is generated.

> **Note:** The order in which modules appear in your code is the order in which they will appear in the reporting interface. (The exception to this is the filter module which has a fixed location in the summary section at the top of the report.)

```
<module type="stdchart" name="chartbrowser">
<param type="request"
       name="charttype"
       key="chartbrowser_charttype"
       default="pie_labeled">
  <restriction type="listvalue">
    <restrictionvalue value="pie_labeled" />
    <restrictionvalue value=" bar_labeled " />
  </restriction>
</param>
```

The code above is analyzed as follows:

- Module of type `stdchart` defines this as a chart module. You assign an object handle to the module using the `name` parameter. You will use this handle to give your chart module a heading later on in this procedure by declaring it in the report's localization properties file.
- The `<param>` tag defines the desired style for your chart. Parameter of type `request` and name `charttype` specify the chart type by assigning a value to the `key` parameter. The value of the key parameter must follow the syntax, `moduleName_chartstyle`. In our example, you are using the `chartbrowser_charttype` chart type.

> **Note:** The available chart styles are defined in the Swiff Chart Generator software and are referenced by Analytics. The following default Swiff Chart Generator chart styles are supported: area, bar, column, line, and pie.

- The `<restriction>` tag enables validation of request parameters. Only parameter values defined by each `<restrictionvalue>` tag are allowed; an error message will be displayed for all other values.

3. Now, add the following parameters to the chart module:

```
<param type="string" name="chart_max_display" value="10"/>
<param type="string" name="chart_display_rest" value="true"/>
```

4. The `chart_max_display` parameter denotes that the pie chart will display ten different colored pie sections corresponding to the top 10 popular browsers. The `chart_display_rest` parameter denotes that the rest of the browser session counts will be shown as a single pie chart section. These two parameters are optional, and in the absence of these parameters the reporting engine takes the default values for chart rendering.

5. Add the data retrieval section to your chart module. Insert the following code after the chart module declaration you added in the previous step:

```
select
bro.id as broid,
bro.name as broname,
nvl(count, 0) as count,
<equals name="browsertotal" value="0">
  0 as percent
</equals>
<notEquals name="browsertotal" value="0">
  (nvl(count, 0)/#browsertotal#)*100 as percent
</notEquals>
from
l2_browser bro
join (
  select
  bro2.id as broid,
  bro2.name as broname,
  sum(count) as count
  from
  help_dates dates
  join l3_datexsitexnewbrowserxcount l3 on (dates.id = l3.dateid)
  join l2_browser bro2 on (l3.browserid = bro2.id)
  where<![CDATA[
        siteid = #siteid# and
        dates.dat >= to_date('#sdate#', 'DD-MM-YYYY') and
        dates.dat < to_date('#edate#', 'DD-MM-YYYY')+1 ]]>
  group by
  bro2.id, bro2.name
) on (bro.id = broid)
```

The code above is analyzed as follows

The SQL query retrieves the data that will be displayed on the pie chart. Note that the data retrieval query in the chart module is similar to the query of the table module. The only difference is that the data retrieval query for the chart module lacks the filter part, (for example, the data on the chart will not be filtered by the filter input, only table data will be filtered). However you can add the same filter in the query to filter the chart data.

6. Add the display section code to your chart module. Insert the following code after the data retrieval section you added in the previous step:

```
<display type="html">
  <value name="xaxis" type="string">
      <valueparam name="format" parse="false"/>
      <valueparam name="value" value="#broname#" />
  </value>

  <value name="yaxis" type="number">
      <valueparam name="format" parse="false" value="#####0" />
      <valueparam name="value" value="#count#" />
  </value>
</display>
```

The code above is analyzed as follows:

In the display section, you define the axes for your chart and assign the appropriate data to each axis.

Analytics distinguishes between two categories of charts:

- XY charts are charts that have no axes, such as pie charts, and therefore require no legend for the data series (and thus, no Z axis declaration)
- XYZ charts are charts that have axes, (such as line, bar, area, and column charts) and therefore require that the data series legend be defined through a Z axis declaration.

In our example you are defining a pie chart, which is an XY chart so there will be no Z axis. However, we assign the different categories/data points (browser names in this exercise) to the X axis (value name="xaxis") and values for each categories (here session count) to Y axis (value name="yaxis")

7. Save and close the file.
8. Give your chart a heading. Do the following:
    a. Open the `NewBrowserLocalization.properties` file (in our example, located in the `WEB-INF/classes` directory in the `analytics` application directory on your application server) in a text editor.
    b. Add the following statement at the end of the file:

    ```
    # Chart Module
    report_newbrowser_module_chartvisitordetail=Browsers Chart
    ```

    c. Save and close the file.
    d. Restart your application server for your changes to take effect.
9. Your report is now complete! Continue on to the final step in this exercise, Section 5.3.4, "Testing the Completed 'NewBrowsers' Report," to test your finished report and check your code.

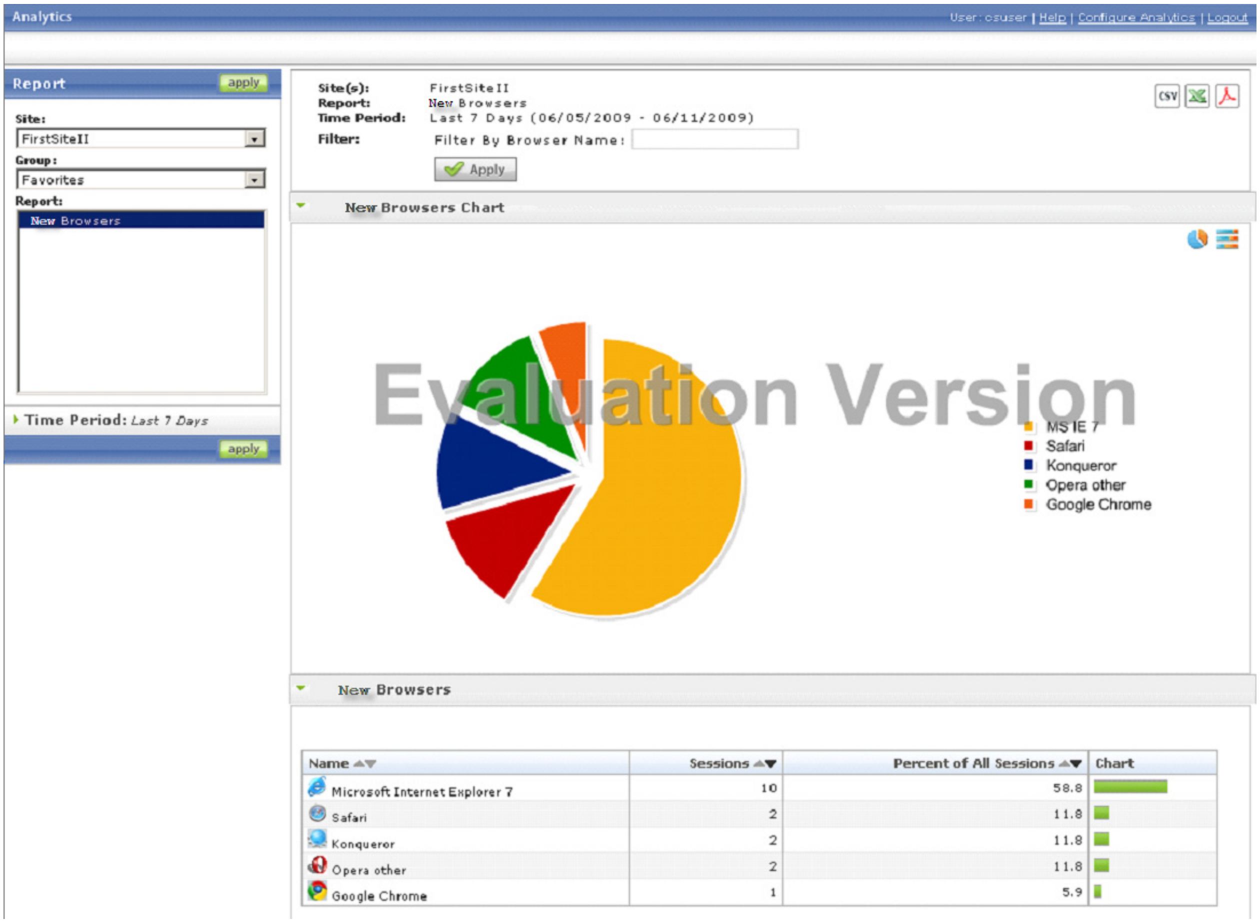
5.3.4 Testing the Completed 'NewBrowsers' Report

Congratulations! You have successfully built your first Analytics report. Test your report to make sure there are no errors in the code.

1. Log in to the Analytics reporting interface and generate the "NewBrowsers" report.

    Your report should look like the one in Figure 5–8:

Figure 5–8 Completed "New Browsers" Report



Site: FirstSiteII
Group: Favorites
Report: New Browsers

Site(s): FirstSiteII
Report: New Browsers
Time Period: Last 7 Days (06/05/2009 - 06/11/2009)
Filter: Filter By Browser Name:

Name	Sessions	Percent of All Sessions	Chart
Microsoft Internet Explorer 7	10	58.8	
Safari	2	11.8	
Konqueror	2	11.8	
Opera other	2	11.8	
Google Chrome	1	5.9	

2. If your report does not appear as in the above figure, or if an error is displayed, retrace your steps and check your code for errors by comparing it to the sample code in Section 5.1.1, "'NewBrowsers' Report Code."

Part II

Reference

This part contains information about enabling data capture for different types of published WebCenter Sites assets. This part also outlines the schema of database tables in which statistics on site visitors are stored.

This part contains the following chapters:

6 Enabling Data Capture for Different Types of Reports

This chapter shows you how to enable data capture on website visitors and their usage of pages that are published on Oracle WebCenter Sites. In this chapter, you will implement the `AddAnalyticsImgTag` and the associated code to enable data capture. Once the captured data is statistically processed, the results can be made available for various Analytics reports such as General Information, Content Information, User Information, and Engage reports.

This chapter contains the following sections:

- Section 6.1, "Enabling Data Capture"
- Section 6.2, "Testing Your Analytics Installation"

6.1 Enabling Data Capture

Note: The steps in this section require you to have a working knowledge of WebCenter Sites elements, Java, and JSP. If you plan to enable data capture on Engage assets, you must also have a thorough understanding of recommendations and segments.

Before starting the steps in this chapter, ensure WebCenter Sites administrators have completed the procedures for integrating Analytics with Web Center Sites, as described in the *Oracle Fusion Middleware WebCenter Sites: Analytics Administrator's Guide*.

Analytics can capture data on the usage of WebCenter Sites assets, on their visitors, and the visitors' searches only if published pages are tagged for data capture. In the case of Engage assets and internal searches, the assets themselves must be tagged for data capture.

This section shows you how to enable data capture for a variety of reports:

- Section 6.1.1, "Enabling Data Capture for General, Content, and User Information Reports"
- Section 6.1.2, "Enabling Data Capture for the "Internal Searches" Report"
- Section 6.1.3, "Enabling Data Capture for Engage Reports"

Completing the steps enables data capture for all reports in all report groups.

6.1.1 Enabling Data Capture for General, Content, and User Information Reports

In this section, you will enable data capture for all report groups except "Internal Searches" and "Engage."

To enable data capture for General, Content, and User Information Reports

Tag the pages on which the assets are displayed. Use the following code:

```
<%-- Analytics Image tag
     Data capture code for asset uses and user information
--%>
<ics:if condition='<%="true".equalsIgnoreCase(ics.GetProperty("analytics.enabled",
"futuretense_xcel.ini", true))%>'>
<ics:then>

<ics:if condition='<%=ics.GetVar("packedargs") != null%>'>

<ics:then>
<render:unpackarg unpack="recid" remove="true"
packed='<%=ics.GetVar("packedargs")%>' outvar="packedargs"/>

</ics:then>

</ics:if>
<render:callelement elementname="Analytics/AddAnalyticsImgTag">
<render:argument name="c" value='<%=ics.GetVar("c")%>'/>
<render:argument name="cid" value='<%=ics.GetVar("cid")%>'/>
<render:argument name="site" value='<%=ics.GetVar("site")%>'/>
<render:argument name="pagename" value='<%=ics.GetVar("childpagename")%>'/>
<render:argument name="recid"
value='<%=ics.GetVar("recid")%>'/>
</render:callelement>
</ics:then>
</ics:if>
```

Parameters

- `c:`

  Type of asset being visited

- `cid:`

  ID of an asset of a given type

- `site:`

  Name of the content management site hosting the page

- `pagename:`

  Name of the page being visited

- `pageurl:`

  URL of the page being visited

Note:

- Be sure to tag pages that display the assets whose asset types are registered. If the pages remain untagged, empty reports will be generated. For registration procedures, see "Configuring an Asset Report" in the *Oracle Fusion Middleware WebCenter Sites: Analytics Administrator's Guide*.
- If you are using the FirstSite II sample site, note that it uses a wrapper page that is called by all other pages. Add the data capture code (in this section) to the wrapper element (for example, `ElementCatalog/FSIIWrapper)`. Do the same for any other site that uses a wrapper page. For any site that does not use a wrapper page, make sure that the `c`, `cid`, `site` and `pagename` parameters are properly passed from the page to the data capture code.
- To enable generation of a site's "Entry - Exit Pages" report (which identifies a session's first and last pages), you can code either all pages on that site, or a wrapper page that is called by all other pages.

For each page request, the `FatWire/Analytics/AddAnalyticsImgTag` invokes the `sensor` servlet in Analytics to capture data on visitors browsing the page. The data includes visitors' geographic locations, IP addresses, browsing technologies, and session parameters. The remaining code captures usage information on WebCenter Sites assets displayed on the same page. The captured data is later processed statistically. The statistics are then available for "General Information" reports, "Content Information" reports, and "User Information" reports.

6.1.2 Enabling Data Capture for the "Internal Searches" Report

Enable data capture for "Internal Searches" report by calling the HTML `img` tag on every page where search data must be captured. Pass the following parameters:

```
<img
style="display:none;"
height="1"
width="1"
border="0"
src="<%=statisticsUrl%>
?siteName=<%= ics.GetVar("site")%>
&objType=SEARCH
&query=<SQL_query>
&objID=1
&directhits=<number_of_results>
&maxscore=<relevance_score>
&sessionID=<%=sessionId%>"
alt="pixel"
/>
```

Parameters

- `src`

  URL of the data capture servlet for Analytics. The URL is configured in the `analytics.datacaptureurl` property, in `futuretense.ini`. Get the URL as follows:

  ```
  <property:get param="analytics.datacaptureurl" inifile="futuretense.ini"
  ```

```
varname="datacaptureurl"/>
<% String statisticsUrl=
  ics.GetVar("datacaptureurl");%>
```

- `siteName`

  Name of the content management site where the query is entered.

- `objType`

  Object name for the internal SQL query. The value must be `SEARCH`.

- `query`

  The query that is entered in the search field.

- `objID`

  A random value (pass a value of "1").

- `directhits`

  Number of results returned by the search query.

- `maxscore`

  (Optional) the relevance score of the most relevant result.

  For example, `88.0` for a result that is 88% relevant to the keywords entered.

- `sessionID`

  Browser session ID.

Example:

The following example shows how the `img` tag could look after the variables are replaced:

```
<img
style="display:none;"
height="1"
width="1"
border="0"
src="http://10.120.12.68:7001/analytics/statistics
?siteName=FirstSiteII
&objType=SEARCH
&query=Select * from Activities
&objID=1
&directhits=10
&maxscore=88.0
&sessionID=5E7B9B22F84F0FAB0E68108089AA1609"
alt="pixel"
/>
```

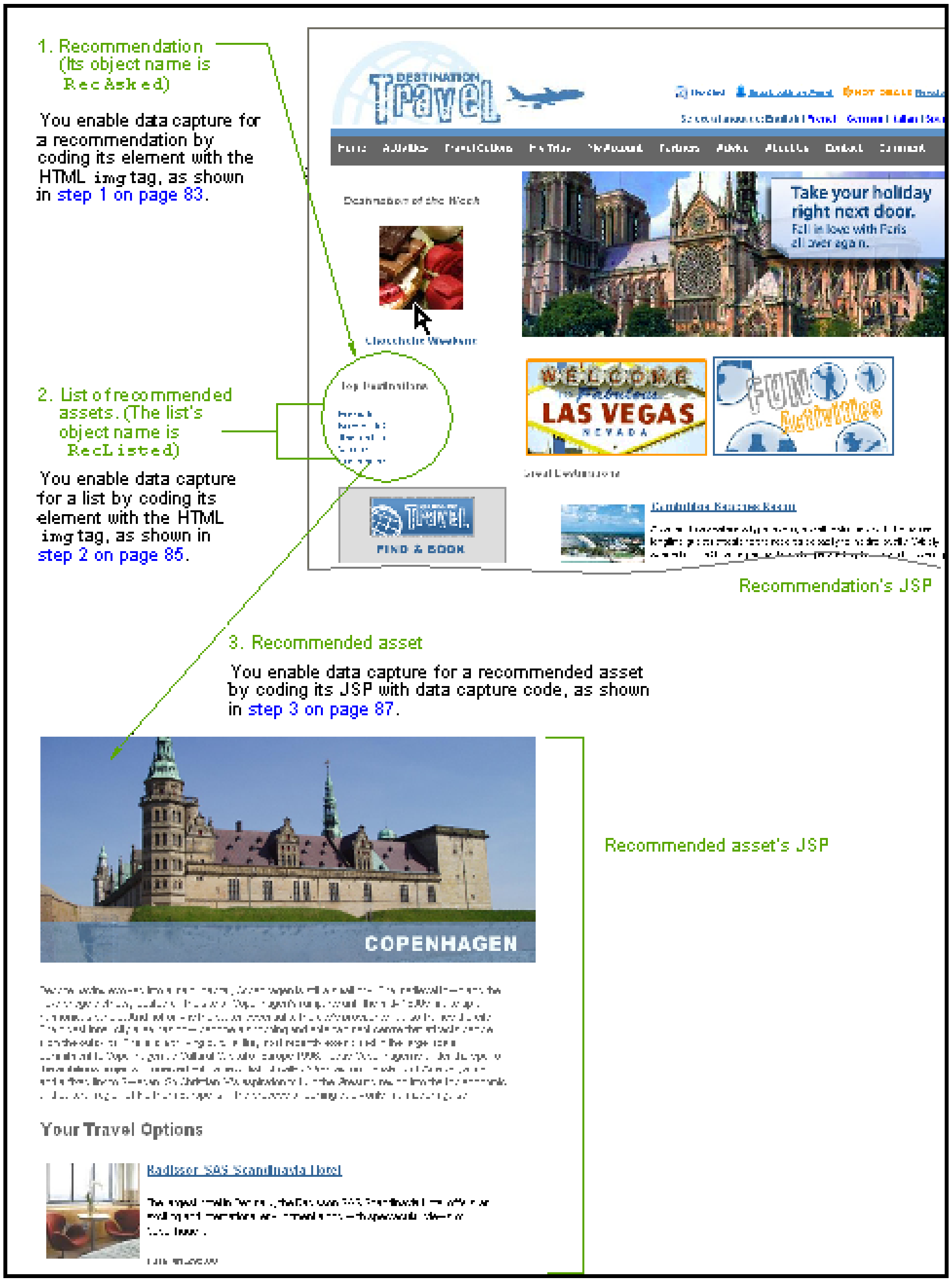
If your pages contain Engage assets and you wish to enable data capture, continue with Section 6.1.3, "Enabling Data Capture for Engage Reports." Otherwise, you have completed configuring Analytics. Test the system by following the steps in Section 6.2, "Testing Your Analytics Installation."

6.1.3 Enabling Data Capture for Engage Reports

Engage assets include recommendations, recommended assets (advertised in the recommendations), and segments to which the recommendations are made. Before starting the steps in this section, familiarize yourself with their general flow by referring to the steps in Figure 6–1. The same figure provides examples of Engage assets and their counterpart objects RecAsked and RecListed.

Note: Data capture on segments is implicit via the segID parameter in the img tag, used to enable data capture.

Figure 6–1 Definitions of Engage Assets



1. Recommendation (Its object name is RecAsked)

You enable data capture for a recommendation by coding its element with the HTML img tag, as shown in step 1 on page 83.

2. List of recommended assets. (The list's object name is RecListed)

You enable data capture for a list by coding its element with the HTML img tag, as shown in step 2 on page 85.

Recommendation's JSP

3. Recommended asset

You enable data capture for a recommended asset by coding its JSP with data capture code, as shown in step 3 on page 87.

Recommended asset's JSP

Destination Travel
Take your holiday right next door.
Fall in love with Paris all over again.
Welcome to Fabulous Las Vegas Nevada
Fun Activities
Travel
FIND & BOOK
COPENHAGEN
Your Travel Options
Radisson SAS Scandinavia Hotel

Note: To complete the steps below, you must have a thorough understanding of recommendations and segments, as well as experience with Java and JSP.

To enable data capture on Engage assets

1. Enable data capture on each recommendation (for an example of a recommendation, see Figure 6–1):

   a. Add the lines below to the recommendation's element:

```
<%@ taglib prefix="property" uri="futuretense_cs/property.tld"%>
<%@ page import="java.util.*, java.text.*, java.io.*"%>
<%@ page import="at.onetoone.esa.tools.*"%>
```

> **Note:** The `at.onetoone.esa.tools` files are located in `analyticscs.jar`.

   b. Insert the HTML `img` tag below into the recommendation's element. The `img` tag will capture the ID and name of the recommendation as soon as the page is displayed to the visitor.

> **Note:** If you are using the FirstSite II sample site, add the `img` tag to `ElementCatalog/AdvCols/FSIIDetail`.

```
<script type="text/javascript">

pixelreccalled = new Image();
      pixelreccalled.src = "<%=statisticsUrl%>?siteName
          =<%= ics.GetVar("site")%>
&objType=RecAsked&objID=<%=ics.GetVar("cid")%>
&objName=<%=ics.GetVar("reconame")%>

  &segID=<%=segmentidlist%>

  &sessionID=<%=sessionId%>

  &Referer=<%= referer %>

  &size=<%=screenResolution%>";
</script>
<noscript>
<img style="display:none;"
height="1"
width="1"
src="<%=statisticsUrl%>?siteName=<%= ics.GetVar("site")%>
&objType=RecAsked
&objID=<%=ics.GetVar("cid")%>
&objName=<%=ics.GetVar("reconame")%>
&segID=<%=segmentidlist%>
&sessionID=<%=sessionId%>
&Referer=<%= referer %>
&size=<%=screenResolution%>
alt="pixel" />
</noscript>
```

Parameters

– `src`

URL of the data capture servlet for Analytics. The URL is configured in the `analytics:datacaptureurl` property, in `futuretense.ini`. Get the URL as follows:

```
<property:get param="analytics.datacaptureurl" inifile="futuretense.ini"
varname="datacaptureurl"/>

<% String statisticsUrl=
    ics.GetVar("datacaptureurl");%>
```

– `siteName`

  Name of the content management site where the recommendation is displayed.

– `objType`

  Object name for the displayed recommendation. The value must be `RecAsked`. (For an example of a recommendation, see the first step in Figure 6–1.) The `objType` is used to identify Engage assets (recommendations and segments), capture data on the assets, and create reports in the "Engage" report group.

– `objID`

  Asset ID of the recommendation.

– `objName`

  Name of the recommendation.

– `segID`

  Either a comma-separated list of IDs of the segments to which the visitor belongs, or `0` if the visitor does not belong to a segment.

– `sessionID`

  Browser session ID.

– `size`

  Browser screen resolution.

Example

The following example shows how the `img` tag could look after the variables are replaced:

```
<img
style="display:none;"
height="1"
width="1"
src="http://etravel.fatwire.com:80/analytics/statistics
?siteName =FirstSiteII
&objType=RecAsked&objID=1120593564193
&objName=FSIIItemOfTheWeek
&segID=0
&sessionID=B28822803C3843B8CC
&size=1280x1024"
alt="pixel"/>
```

2. Enable data capture on each recommendation's list of recommended assets (for an example of such a list, see Figure 6–1).

This step requires you to first encode three parameters, then insert the HTML `img` tag into the same recommendation element as in the previous step, but pass parameters which are specific to the list of recommended assets. Do the following:

a. Encode the parameters `objListName`, `objListID`, and `objListType` as follows, using the Base64 file located in `analyticscs.jar`:

```
String objListNameValue = Base64.encodeBytes(<comma-separated list of asset
names in the recommendation>.getBytes(), Base64.DONT_BREAK_LINES);
String objListIDValue = Base64.encodeBytes(<comma-separated list of asset
IDs in the recommendation>.getBytes(), Base64.DONT_BREAK_LINES);
String objListTypeValue = Base64.encodeBytes(<comma-separated list of asset
types in the recommendation>.getBytes(), Base64.DONT_BREAK_LINES);
```

b. Insert the HTML `img` tag with the following parameters into the JSP:

Note: If you are using the FirstSite II sample site, add the `img` tag to `ElementCatalog\AdvCols\FSIIDetail`.

For example code, see `FSIIDetail.jsp` (in the following directory: `FatWire Analytics\examples\FirstSiteII\ElementCatalog/AdvCols`).

```
<script type="text/javascript">

imagereclisted = new Image();

imagereclisted.src = "<%=statisticsUrl%>?siteName=<%
 = ics.GetVar("site")%>
 &objType=RecListed
 &objID=<%=ics.GetVar("cid")%>
 &objName=<%=ics.GetVar("reconame")%>
 &objListID=<%=objListIDValue%>
 &objListName=<%=objListNameValue%>
 &objListType=<%=objListTypeValue%>
 &segID=<%=segmentidlist%>
 &sessionID=<%=sessionId%>
 &size=<%=screenResolution%>";
</script>
<noscript>

<img style="display:none;"
height="1"
width="1"
src="<%=statisticsUrl%>
?siteName=<%= ics.GetVar("site")%>
&objType=RecListed
&objID=<%=ics.GetVar("cid")%>
&objName=<%=ics.GetVar("reconame")%>
&objListID=<%=objListIDValue%>
&objListName=<%=objListNameValue%>
&objListType=<%=objListTypeValue%>
&segID=<%=segmentidlist%>
&sessionID=<%=sessionId%>
&size=<%=screenResolution%>"
alt="pixel"
/>

</noscript>
```

Parameters

- `src`

  URL of the data capture servlet for Analytics. The URL is configured in the `analytics.datacaptureurl` property, in `futuretense.ini`. Get the URL as follows:

  ```
  <property:get param="analytics.datacaptureurl"
  inifile="futuretense.ini" varname="datacaptureurl"/>

  <% String statisticsUrl=
    ics.GetVar("datacaptureurl");%>
  ```

- `siteName`

  Name of the content management site where the recommendation is listed.

- `objType`

  Object name for the list of recommended assets. The value must be `RecListed`. (For an example of a list of recommended assets, see the second step in Figure 6–1.) The `objType` is used to identify Engage assets (recommendations and segments), capture data on the assets, and create reports in the "Engage" report group.

- `objID`

  Asset ID of the recommendation.

- `objName`

  Name of the recommendation.

- `objListName`

  Comma-separated list of asset names in the recommendation. The value for this parameter is the value that was encoded in the previous step.

- `objListType`

  Comma-separated list of asset types in the recommendation. The value for this parameter is the value that was encoded in the previous step.

- `segID`

  Either a comma-separated list of IDs of the segments to which the visitor belongs, or `0` if the visitor does not belong to a segment.

- `sessionID`

  Browser session ID.

- `size`

  The browser screen resolution.

Example:

The following example shows how the `img` tag could look after the variables are replaced:

```
<img
style="display:none;"
height="1"
width="1"
src="http://etravel.fatwire.com:80/analytics/statistics
```

```
?siteName=FirstSiteII
&objType=RecListed
&objID=1120593564193
&objName=FSIIItemOfTheWeek
&objListID=MTEzODY2MDI1NjExMQ===
&objListName=Q2hvY29ob2xpYyBXZWVrZW5k=
&objListType=UHJvZHVjdF9D=
&segID=0&sessionID=B28822803C3843B8CC496743D5B132B0
&size=1280x1024"
alt="pixel"
/>
```

c. Add the `recid` parameter to the element that generates the links which make up the list of recommended assets (each link points from the recommendation to a recommended asset). Also, add the `recid` parameter to the `pagecriteria` variable for the element's `SiteCatalog` entry.

> **Note:** `recid` is the ID of the recommendation to which an asset belongs. For example code, see `FSIISummary.jsp` for `Product_C` (in the following directory: `FatWire Analytics\examples\FirstSiteII\ElementCatalog\Product_C`).

3. Enable data capture on each recommended asset (for an example of a recommended asset, see Figure 6–1):

a. If the data capture code from Section 6.1.1, "Enabling Data Capture for General, Content, and User Information Reports" does not exist in the recommended asset's JSP, add the code.

> **Note:** Data capture code can exist on a page displaying an Engage asset if that page was already coded for data capture on WebCenter Sites assets.

b. Pass the `recid` parameter to the data capture element `"FatWire/Analytics/`AddAnalyticsImgTag" by inserting the following line into the data capture code in Section 6.1.1, "Enabling Data Capture for General, Content, and User Information Reports":

```
<render:argument name="recid" value='<%=ics.GetVar
     ("recid")%>'/
```

This enables Analytics to record which Engage asset has been selected.

4. At this point, you have enabled data capture for each recommendation, its list of recommended assets, and its individual recommended assets. You have also implicitly enabled data capture for segments, via the `segID` parameter in the `img` tags.

Continue to the next step to test your installation.

6.2 Testing Your Analytics Installation

You can test your code by testing your Analytics installation either on your own or with the help of the WebCenter Sites administrator and other users.

To test your Analytics installation

1. Go to the web site and visit the pages you have tagged for data capture.
2. Allow the data analysis jobs to finish processing the captured data.

> **Note:** The data analysis jobs process data that was captured within the past 24 hours. Bear in mind that a data analysis job is a resource-intensive process that can take a significant amount of time to complete, depending on the amount of raw data that was captured.

3. Generate all the reports that users will be generating. For descriptions of the reports and instructions on generating them, see the *Oracle Fusion Middleware WebCenter Sites: Analytics User's Guide*.

Assuming a successful outcome, you are now ready to use Analytics.

7

WebCenter Sites: Analytics Database Schema

This appendix contains the following sections:

- Section 7.1, "L3 Tables"
- Section 7.2, "HELP Tables"
- Section 7.3, "System Configuration and Maintenance Tables"
- Section 7.4, "Database Indexes"

7.1 L3 Tables

Level 3 database tables store statistically processed, aggregated data. When aggregated in advance, the data becomes available on demand at report generation time.

Table names indicate which type of data is stored. For example:

- The table named `L3_DATEXSITEXCITYXCOUNT` stores *daily* data for a site that is browsed by visitors from a given city.
- The table named `L3_WEEKXSITEXCITYXCOUNT` stores the same type of data as `L3_DATEXSITEXCITYXCOUNT`, but for the given week (that is, `L3_WEEKXSITEXCITYXCOUNT` stores *weekly* data for a site that is browsed by visitors from a given city).
- The table named `L3_MONTHXSITEXCITYXCOUNT` stores the same type of data as `L3_DATEXSITEXCITYXCOUNT`, but for the given month (that is, `L3_MONTHXSITEXCITYXCOUNT` stores *monthly* data for a site that is browsed by visitors from a given city).

> **Note:** The term **object impression** is used throughout this reference. An object impression is a single invocation of the sensor servlet. An object impression can also be thought of as a "snapshot" of raw site visitor data that is captured for analysis. For more information about object impressions, see the *Oracle Fusion Middleware WebCenter Sites: Analytics Administrator's Guide*.

- Section 7.1.1, "CS_L3DATEXSITEXOBJXCOUNT"
- Section 7.1.2, "L3_DATEXSITEXCITYXCOUNT"
- Section 7.1.3, "L3_WEEKXSITEXCITYXCOUNT"
- Section 7.1.4, "L3_MONTHXSITEXCITYXCOUNT"

7.1.1 CS_L3DATEXSITEXOBJXCOUNT

This table stores data for the chart in the "Clickstream" report. It stores the number of clicks that were made from one object to another on a specific date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table to reference the date of the data.
SITEID	NUMBER	Foreign key to the SITE table.
OBJECTFROMID	NUMBER	From which object.
OBJECTTOID	NUMBER	To which object.
COUNT	NUMBER	Number of clicks.

7.1.2 L3_DATEXSITEXCITYXCOUNT

Reserved for internal use. This table stores the number of object impressions, sessions, and visitors from a specific city on a specific day. See Section 7.2.1, "HELP_CITY" for more information.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
CITYID	NUMBER	Foreign key to the HELP_CITY table.
OICOUNT	NUMBER	Number of object impressions for the day, for the site, for the city.
VISCOUNT	NUMBER	Number of visitors for the day, for the site, for the city.
SESCOUNT	NUMBER	Number of sessions for the day, for the site, for the city.

7.1.3 L3_WEEKXSITEXCITYXCOUNT

Reserved for internal use. This table stores the number of object impressions, sessions, and visitors from a specific city on a specific week. See Section 7.2.1, "HELP_CITY" for more information.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
CITYID	NUMBER	Foreign key to the HELP_CITY table.

Column Name	Type	Description
OICOUNT	NUMBER	Number of object impressions for the week for the site, for the city.
VISCOUNT	NUMBER	Number of visitors for the week for the site, for the city
SESCOUNT	NUMBER	Number of sessions for the week for the site, for the city.

7.1.4 L3_MONTHXSITEXCITYXCOUNT

Reserved for internal use. This table stores the number of object impressions, sessions, and visitors from a specific city on a specific month. See the HELP_CITY table for more information.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
CITYID	NUMBER	Foreign key to the HELP_CITY table.
OICOUNT	NUMBER	Number of object impressions for the month site, and city.
VISCOUNT	NUMBER	Number of visitors for the month, site, and city.
SESCOUNT	NUMBER	Number of sessions for the month site, and city.

7.1.5 L3_DATEXSITEXDLXMIMEXCOUNT

This table stores the number of object impressions for the given object and its MIME type.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OBJECTID	NUMBER (12)	Foreign key to the L2_OBJECT table.
MIMETYPE	VARCHAR2 (32 BYTE)	The MIME type.
COUNT	NUMBER	Number of object impressions for the day, for the site, and for the object with that MIME type.

7.1.6 L3_DATEXSITEXGROUPXOICOUNT

This table stores how often a group was visited (number of object impressions).

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
GROUPID	NUMBER	Foreign key to the L2_GROUP table.
COUNT	NUMBER	Number of object impressions.

7.1.7 L3_DATEXSITEXGROUPXSESCOUNT

This table stores how often a group was visited (number of sessions).

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
GROUPID	NUMBER	Foreign key to the L2_GROUP table.
SESCOUNT	NUMBER	Number of sessions.

7.1.8 L3_DATEXSITEXGROUPXVISCOUNT

This table stores the number of visitors a group receives on a daily basis.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
GROUPID	NUMBER	Foreign key to the L2_GROUP table.
VISCOUNT	NUMBER	Number of visitors.

7.1.9 L3_WEEKXSITEXGROUPXVISCOUNT

This table stores the number of visitors a group receives on a weekly basis.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
GROUPID	NUMBER	Foreign key to the L2_GROUP table.
VISCOUNT	NUMBER	Number of visitors.

7.1.10 L3_MONTHXSITEXGROUPXVISCOUNT

This table stores the number of visitors a group receives on a monthly basis.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
GROUPID	NUMBER	Foreign key to the L2_GROUP table.

Column Name	Type	Description
VISCOUNT	NUMBER	Number of visitors.

7.1.11 L3_DATEXSITEXREGIONXCOUNT

Reserved for internal use. This table stores the number of object impressions, sessions, and visitors that occur daily for each region.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
REGIONID	NUMBER	Foreign key to the HELP_REGION table.
OICOUNT	NUMBER	Number of object impressions from that region on the site for the day.
VISCOUNT	NUMBER	Number of visitors from that region on the site for the day.
SESCOUNT	NUMBER	Number of sessions from that region on the site for the day.

7.1.12 L3_WEEKXSITEXREGIONXCOUNT

Reserved for internal use. This table stores the number of object impressions, sessions, and visitors that occur weekly for each region.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
REGIONID	NUMBER	Foreign key to the HELP_REGION table.
OICOUNT	NUMBER	Number of object impressions from that region on the site for the week.
VISCOUNT	NUMBER	Number of visitors from that region on the site for the week.
SESCOUNT	NUMBER	Number of sessions from that region, on the site for the week.

7.1.13 L3_MONTHXSITEXREGIONXCOUNT

Reserved for internal use. This table stores the number of object impressions, sessions, and visitors that occur monthly for each region.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.

Column Name	Type	Description
SITEID	NUMBER (6)	Foreign key to the SITE table.
REGIONID	NUMBER	Foreign key to the HELP_REGION table.
OICOUNT	NUMBER	Number of object impressions from that region on the site for the month.
VISCOUNT	NUMBER	Number of visitors from that region on the site for the month.
SESCOUNT	NUMBER	Number of sessions from that region on the site for the month.

7.1.14 L3_DATEXSITEXJS

This table stores the number of clients that use JavaScript.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
VALUE	NUMBER	This stores either `true` or `false`.
COUNT	NUMBER	Number of object impressions with that `VALUE`.

7.1.15 L3_DATEXSITEXOBJECTXDURATION

This table stores the length of time that users view an object.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OBJECTID	NUMBER (12)	Foreign key to the L2_OBJECT table.
DURATION	NUMBER	Number of seconds until the user called another page.
COUNT	NUMBER	Number of data rows from which the duration is calculated.

7.1.16 L3_DATEXSITEXBROWSERXCOUNT

This table stores the number of sessions a type of browser has completed.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
BROWSERID	NUMBER (12)	Foreign key to the L2_BROWSER table.

Column Name	Type	Description
COUNT	NUMBER	Number of sessions a browser has completed.

7.1.17 L3_DATEXSITEXCOUNTRYXCOUNT

This table stores aggregated data for each country, for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
COUNTRYID	NUMBER (12)	Foreign key to the HELP_COUNTRY table.
COUNT	NUMBER	Number of sessions for this site, on this date.
OICOUNT	NUMBER	Number of object impressions for this site, on this date.
VISCOUNT	NUMBER	Number of visitors for this site, on this date.

7.1.18 L3_WEEKXSITEXCOUNTRYXCOUNT

This table stores aggregated data for each country, for the given week.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
COUNTRYID	NUMBER (12)	Foreign key to the HELP_COUNTRY table.
COUNT	NUMBER	Number of sessions for this site, within this week.
OICOUNT	NUMBER	Number of object impressions for this site, within this week.
VISCOUNT	NUMBER	Number of visitors for this site, within this week.

7.1.19 L3_MONTHXSITEXCOUNTRYXCOUNT

This table stores aggregated data for each country, for the given month.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
COUNTRYID	NUMBER (12)	Foreign key to the HELP_COUNTRY table.

Column Name	Type	Description
COUNT	NUMBER	Number of sessions for this site, on this date.
OICOUNT	NUMBER	Number of object impressions for this site, on this date.
VISCOUNT	NUMBER	Number of visitors for this site, on this date.

7.1.20 L3_DATEXSITEXENTRYIDXCOUNT

This table stores entry page data for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OBJECTID	NUMBER	Foreign key to the L2_OBJECT table.
COUNT	NUMBER	How often this object was used as an entry page.

7.1.21 L3_DATEXSITEXEXITIDXCOUNT

This table stores exit page data for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OBJECTID	NUMBER	Foreign key to the L2_OBJECT table.
COUNT	NUMBER	How often this object was used as an exit page.

7.1.22 L3_DATEXSITEXHOSTNAMEXCOUNT

This table stores the number of sessions for the given host name on the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
HOSTNAME	VARCHAR2 (64 BYTE)	Name of the host.
COUNT	NUMBER	Number of sessions for this host.

7.1.23 L3_DATEXSITEXINTERNALSEARCH

This table stores aggregated data of internal searches for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
QUERY	VARCHAR2 (2049 BYTE)	Query string that was searched.
COUNT	NUMBER	How often the query was searched.
AVGHITS	NUMBER	Average number of search results.
MAXHITS	NUMBER	Maximum number of search results.
MINHITS	NUMBER	Minimum number of search results.
AVGMAXSCORE	NUMBER	Average maximum score.
MINSCORE	NUMBER	Minimum score.
MAXSCORE NUMBER	NUMBER	Maximum score.
OBJECTTYPEID	NUMBER	Foreign key to L2_OBJECTTYPE of the query.

7.1.24 L3_DATEXSITEXIPXCOUNT

This table stores the number of sessions for the given IP address and date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
IP	VARCHAR2 (15 BYTE)	IP address.
COUNT	NUMBER	Number of sessions for the IP address on the site for the day.

7.1.25 L3_DATEXSITEXISPXCOUNT

This table stores for each ISP daily the number of object impressions, sessions and visitors.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER(6)	Foreign key to the SITE table.
ISPID	NUMBER	Foreign key to the HELP_ISP table.
OICOUNT	NUMBER	Number of object impressions for that ISP on that site on that day.
VISCOUNT	NUMBER	Number of visitors for that ISP on that site on that day.
SESCOUNT	NUMBER	Number of sessions for that ISP on that site on that day.

7.1.26 L3_WEEKXSITEXISPXCOUNT

This table stores for each ISP weekly the number of object impressions, sessions and visitors.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER(6)	Foreign key to the SITE table.
ISPID	NUMBER	Foreign key to the HELP_ISP table.
OICOUNT	NUMBER	Number of object impressions for that ISP on that site on that week.
VISCOUNT	NUMBER	Number of visitors for that ISP on that site on that week.
SESCOUNT	NUMBER	Number of sessions for that ISP on that site on that week.

7.1.27 L3_MONTHXSITEXISPXCOUNT

This table stores for each ISP monthly the number of object impressions, sessions and visitors.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER(6)	Foreign key to the SITE table.
ISPID	NUMBER	Foreign key to the HELP_ISP table.
OICOUNT	NUMBER	Number of object impressions for that ISP on that site on that month.
VISCOUNT	NUMBER	Number of visitors for that ISP on that site on that month.
SESCOUNT	NUMBER	Number of sessions for that ISP on that site on that month.

7.1.28 L3_DATEXSITEXOBJECTTYPEXCOUNT

This table stores aggregated data for each object type for the given date and hour.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OBJECTTYPEID	NUMBER (12)	Foreign key to the L2_OBJECTTYPE table.
COUNT	NUMBER	Number of object impressions of that object type for the site, for the day in that hour.

Column Name	Type	Description
HOUR	VARCHAR2 (2 BYTE)	Hour of the object impressions.

7.1.29 L3_DATEXSITEXOBJECTXCOUNT

This table stores aggregated data for each object, for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OBJECTID	NUMBER (12)	Foreign key to the L2_OBJECT table.
COUNT	NUMBER	Number of object impressions.

7.1.30 L3_AUDIT_TRAIL

This table provides audit trail tracking of asset modifications.

Column Name	Type	Description
DATEID	NUMBER	Foreign key in the HELP_DATES table.
OBJECTID	NUMBER	Asset identifier.
SITEID	NUMBER	Foreign key in the SITE table.
CSUSERID	VARCHAR2(255)	User name of the WebCenter Sites user.
OPTYPE	VARCHAR2(255)	Type of operation: CREATE|EDIT|DELETE
TIMESTAMP	DATE	When the modification occurred.

7.1.31 L3_DATEXSITEXOPTYPEXCOUNT

This table stores the number of daily asset modifications.

Column Name	Type	Description
DATEID	NUMBER	Foreign key in the HELP_DATES table.
SITEID	NUMBER	Foreign key in the SITE table.
OPTYPE	VARCHAR2(255)	Type of operation: CREATE|EDIT|DELETE
COUNT	NUMBER	Number of incidents.

7.1.32 L3_WEEKXSITEXOPTYPEXCOUNT

This table stores the number of weekly asset modifications.

Column Name	Type	Description
DATEID	NUMBER	Foreign key in the HELP_DATES table.
SITEID	NUMBER	Foreign key in the SITE table.
OPTYPE	VARCHAR2(255)	Type of operation: CREATE|EDIT|DELETE
COUNT	NUMBER	Number of incidents.

7.1.33 L3_MONTHXSITEXOPTYPEXCOUNT

This table stores the number of monthly asset modifications.

Column Name	Type	Description
DATEID	NUMBER	Foreign key in the HELP_DATES table.
SITEID	NUMBER	Foreign key in the SITE table.
OPTYPE	VARCHAR2(255)	Type of operation: CREATE|EDIT|DELETE
COUNT	NUMBER	Number of incidents.

7.1.34 L3_DATEXSITEXOSXCOUNT

This table stores the number of sessions for each operating system, for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
OSID	NUMBER (12)	Foreign key to the L2_OS table.
COUNT	NUMBER	Number of sessions for the operating system.

7.1.35 L3_DATEXSITEXREFERERXCOUNT

This table stores the number of sessions for each referrer URL, for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
REFERER	VARCHAR2 (500 BYTE)	The referrer URL.
COUNT	NUMBER	Number of sessions from the referrer URL.

7.1.36 L3_DATEXSITEXSCREENRESXCOUNT

This table stores the number of sessions for each screen resolution, for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
SCREENRESOLUTIONID	NUMBER (12)	Screen resolution as a single number `VVVVHHHH`, where `VVVV` is vertical resolution, and `HHHH` is horizontal resolution.
COUNT	NUMBER	Number of sessions with that screen resolution.

7.1.37 L3_DATEXSITEXSESSION

This table stores session information for each site, for the given date.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
AVG	NUMBER	Average duration of all sessions included in the line.
DURRATIONQUANTIL	NUMBER	■ `Null` for sessions without a valid session ID. ■ `0` for sessions with a length of 0 seconds. ■ `1` for sessions between 1 and 30 seconds. ■ `30` for sessions between 30 and 120 seconds. ■ `120` for sessions between 120 and 300 seconds. ■ `300` for sessions between 300 and 900 seconds. ■ `900` for sessions between 900 and 1800 seconds. ■ `1800` for sessions between 1,800 and 3,600 seconds. ■ `3600` for sessions between 3,600 and 86,400 seconds.
QUANTILCOUNT	NUMBER	Number of sessions in the quantile.
PICOUNT	NUMBER	Number of object impressions in the quantile.

7.1.38 L3_DATEXSITEXSESSIONXQUANTIL

This table stores session information based on quantile.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
SUM	NUMBER	Number of sessions.
AVG	NUMBER	Average duration of these sessions.
QUAN0	NUMBER	Number of sessions with a length of '0'.
QUAN0AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN1	NUMBER	Number of sessions with a length between 1 and 30 seconds.
QUAN1AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN30	NUMBER	Number of sessions with a length between 31 and 120 seconds (2 minutes).
QUAN30AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN120	NUMBER	Number of sessions with a length between 121 and 300 seconds (5 minutes).
QUAN120AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN300	NUMBER	Number of sessions with a length between 301 and 900 seconds (15 minutes).
QUAN300AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN900	NUMBER	Number of sessions with a length between 901 and 1800 seconds (30 minutes).
QUAN900AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN1800	NUMBER	Number of sessions with a length between 1801 and 3600 seconds (one hour).
QUAN1800AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN3600	NUMBER	Number of sessions with a length between 3601 and 86400 seconds (one week).
QUAN3600AVG	NUMBER	Average duration of all sessions from this quantile.
QUAN86400	NUMBER	Number of sessions with a length exceeding 86400 seconds (more than one week).
QUAN86400AVG	NUMBER	Average duration of all sessions from this quantile.

Column Name	Type	Description
QUANNULL	NUMBER	Number of sessions with a length of '0' and a SESSIONID of null (no cookie sessions).
QUANNULLAVG	NUMBER	Reserved for internal use.
PICOUNT	NUMBER	Number of object impressions in the quantile.

7.1.39 L3_DATEXSITEXSEXKEYWORDXCOUNT

This table stores the daily number of sessions that start with a given keyword.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
SEARCHENGINEID	NUMBER	Foreign key to the L2_SEARCHENGINE table.
KEYWORD	VARCHAR2 (128 BYTE)	The search phrase.
COUNT	NUMBER	Number of sessions started with the referrer and the phrase from the search engine.

7.1.40 L3_DATEXSITEXVISITORXCOUNT

This table stores the daily number of sessions for each visitor.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
COUNT	NUMBER	Number of visitors to the site for the day.
SESCOUNT	NUMBER	Number of sessions for the site for the day.
PICOUNT	NUMBER	Number of object impressions on the site for the day.
NEW	NUMBER	Number of new visitors on the site for the day.

7.1.41 L3_WEEKXSITEXVISITORXCOUNT

This table stores the weekly number of sessions for each visitor.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.

Column Name	Type	Description
COUNT	NUMBER	Number of visitors on the site for the week
SESCOUNT	NUMBER	Number of sessions on the site for the week.
PICOUNT	NUMBER	Number of object impressions on the site for the week.
NEW	NUMBER	Number of visitors on the site for the week.

7.1.42 L3_MONTHXSITEXVISITORXCOUNT

This table stores the monthly number of sessions for each visitor.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER (6)	Foreign key to the SITE table.
COUNT	NUMBER	Number of visitors on the site for the month.
SESCOUNT	NUMBER	Number of sessions on the site for the month.
PICOUNT	NUMBER	Number of object impressions on the site for the month.
NEW	NUMBER	Number of new visitors on the site for the month.

7.1.43 REC_L3_RECXCOUNT

This table stores the number of recommendations that were displayed and clicked on a given day.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER	Foreign key to the SITE table.
RECID	NUMBER (6)	Foreign key to the REC_RECOMMENDATION table.
VIEWCOUNT	NUMBER (12)	Number of times the recommendation was displayed.
CLICKCOUNT	NUMBER (12)	Number of times the recommendation was clicked.

7.1.44 REC_L3_RECXSEGXCOUNT

This table stores the number of recommendations that were displayed and clicked by a given segment on a given day.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER	Foreign key to the SITE table.
RECID	NUMBER (6)	Foreign key to the REC_RECOMMENDATION table.
SEGID	NUMBER (6)	Foreign key to the REC_SEGMENT table.
VIEWCOUNT	NUMBER (12)	Number of times the recommendation was displayed by that segment.
CLICKCOUNT	NUMBER (12)	Number of times the recommendation was clicked by that segment.

7.1.45 REC_L3_RECXSEGXOBJXCOUNT

This table stores the number of times an object was accessed on a given day by a specific segment of a specific recommendation.

Column Name	Type	Description
DATEID	NUMBER	Foreign key to the HELP_DATES table.
SITEID	NUMBER	Foreign key to the SITE table.
RECID	NUMBER (6)	Foreign key to the REC_RECOMMENDATION table.
SEGID	NUMBER (6)	Foreign key to the REC_SEGMENT table.
OBJID	NUMBER (24)	Foreign key to the L2_OBJECT table.
OBJVIEWEDCOUNT	NUMBER (12)	How often the object was displayed during a recommendation list.
OBJCLICKEDCOUNT	NUMBER (12)	How often the object was clicked during a recommendation list.

7.2 HELP Tables

Tables in this section store static data that is used by various jobs and reports. The tables are populated at installation time.

- Section 7.2.1, "HELP_CITY"
- Section 7.2.2, "HELP_REGION"
- Section 7.2.3, "HELP_COUNTRY"
- Section 7.2.4, "HELP_DATES"
- Section 7.2.5, "HELP_HOURS"
- Section 7.2.6, "HELP_ISP"

7.2.1 HELP_CITY

Reserved for internal use.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64)	Name of the city.
REGIONID	NUMBER	Foreign key to the HELP_REGION table.
COUNTRYID	NUMBER	Foreign key to the HELP_COUNTRY table.

7.2.2 HELP_REGION

Reserved for internal use.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (128 BYTE)	Name of the region.
COUNTRYID	NUMBER	Foreign key to the HELP_COUNTRY table.
REGIONCODE	VARCHAR2 (2)	Foreign key to the HELP_REGION table.

7.2.3 HELP_COUNTRY

This table stores detailed information about countries.

Column Name	Type	Description
ID	NUMBER	The primary key.
CODE	VARCHAR2 (5 BYTE)	Two-letter country code.
NAME	VARCHAR2 (50 BYTE)	Country name
COUNTRY	VARCHAR2(50)	This column is not used.
POPULATION	NUMBER (14)	Population of the country.

7.2.4 HELP_DATES

This table has one entry for each day. It is used and referenced for performance reasons.

Column Name	Type	Description
ID	NUMBER	The primary key.
DAT	DATE	The date (midnight).
STRDAT	VARCHAR2 (10 BYTE)	The date as yyyy-mm-dd string.

7.2.5 HELP_HOURS

This table has one entry for hour (00–23). It is used and referenced for performance reasons.

Column Name	Type	Description
HOUR	NUMBER	The hour (0-based).
STRHOUR	VARCHAR2 (2 BYTE)	A String representing an hour (any value between 00 and 23).

7.2.6 HELP_ISP

This tables stores all available ISP in the Internet Service Provider report.

Column Name	Type	Description
ID	NUMBER	Primary key.
NAME	VARCHAR2(255)	Name of the ISP.

7.3 System Configuration and Maintenance Tables

Tables in this section are used to configure and maintain the system.

- Section 7.3.1, "ACCESSRIGHT"
- Section 7.3.2, "BROWSERGROUP"
- Section 7.3.3, "BROWSERIDENTIFICATION"
- Section 7.3.4, "IP2COUNTRYCITY_BLOCKS"
- Section 7.3.5, "IP2ISP_BLOCKS"
- Section 7.3.6, "IPQUICKLIST"
- Section 7.3.7, "ESAUSER"
- Section 7.3.8, "IP2COUNTRY"
- Section 7.3.9, "OBJECTTYPEGROUP"
- Section 7.3.10, "SETTING"
- Section 7.3.11, "OSGROUP"
- Section 7.3.12, "OSIDENTIFICATION"
- Section 7.3.13, "REC_RECOMMENDATION"
- Section 7.3.14, "REC_SEGMENT"
- Section 7.3.15, "REPORT"
- Section 7.3.16, "REPORT2REPORTGROUP"
- Section 7.3.17, "REPORTGROUP"
- Section 7.3.18, "REPORTPARAMETER"
- Section 7.3.19, "SEARCHENGINEGROUP"
- Section 7.3.20, "SEARCHENGINEIDENT"
- Section 7.3.21, "SITE"

- Section 7.3.22, "SITE2OBJECTTYPEGROUP"
- Section 7.3.23, "SITEGROUP"
- Section 7.3.24, "PI_SITEXOBJECTTYPE"
- Section 7.3.25, "USER2USERGROUP"
- Section 7.3.26, "USERGROUP"
- Section 7.3.27, "L2_BROWSER"
- Section 7.3.28, "L2_GROUP"
- Section 7.3.29, "L2_OBJECT"
- Section 7.3.30, "L2_OBJECTNAMEHISTORY"
- Section 7.3.31, "L2_OBJECTTYPE"
- Section 7.3.32, "L2_OS"
- Section 7.3.33, "L2_PIOBJECTTYPE"
- Section 7.3.34, "L2_SEARCHENGINE"
- Section 7.3.35, "L2_VISITOR"
- Section 7.3.36, "TASKSTATUSHISTORY"
- Section 7.3.37, "HDFSAGENTSTATUSHISTORY"
- Section 7.3.38, "DETAILDEFINITION"

7.3.1 ACCESSRIGHT

This table defines access rights. A user gains access when one of its user groups has access to reports on the site.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
SITE	NUMBER (12)	Foreign key to the SITE table.
REPORT	NUMBER (12)	Foreign key to the REPORT table.
USERGROUP	NUMBER (12)	Foreign key to the USERGROUP table.

7.3.2 BROWSERGROUP

This table stores the names of browser groups. For example, group "Internet Explorer" includes Internet Explorer 6 and 7. Each browser is individually listed in the L2_BROWSER table.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64 BYTE)	Name of the group. **Legal values:** Any value up to 64 characters.

7.3.3 BROWSERIDENTIFICATION

This table is used to identify browsers based on the browser's agent string. If the agent string contains the pattern of an entry for this table, then the agent is assigned to the browser. If an agent string matches more than one record of this table, the record with the highest priority is taken.

Column Name	Type	Description
ID	INTEGER	The primary key.
BROWSERID	INTEGER	Foreign key to an L2_BROWSER record which assigns the agent to a browser if the pattern applies.
PATTERN	VARCHAR2 (50 BYTE)	The pattern to identify the browser with.
PRIORITY	INTEGER	The priority to use if an agent string applies to more than one record.

7.3.4 IP2COUNTRYCITY_BLOCKS

Reserved for internal use. This table stores information related to an IP address.

Column Name	Type	Description
IPFROM	NUMBER	Limit this information for addresses above this value.
IPTO	NUMBER	Limit this information for addresses below this value.
LOCATIONID	NUMBER	This column is not used.
COUNTRYID	NUMBER	Foreign key to a HELP_COUNTRY record for a given IP address.
CITYID	NUMBER	Foreign key to a HELP_CITY record for a given IP address.
REGIONID	NUMBER	Foreign key to a HELP_CITY record for a given IP address.
ISPID	NUMBER	Used for custom report development only.

7.3.5 IP2ISP_BLOCKS

This tables stores information related to an IP address.

Column Name	Type	Description
IPFROM	NUMBER	Limit this information for addresses above this value.
IPTO	NUMBER	Limit this information for addresses below this value.
ISPID	NUMBER	Foreign key to a HELP_ISP record to which the IP address belongs.
ISPNAME	VARCHAR2(255)	Internet Service Provider name.

7.3.6 IPQUICKLIST

This table caches information calculated from the IP2COUNTRYCITY_BLOCKS table for fast access.

Column Name	Type	Description
IP	NUMBER	The IP address the information is related to.
ISPID	NUMBER	Used for custom report development only.
REGIONID	NUMBER	Foreign key to a HELP_REGION record that the IP address belongs to.
CITYID	NUMBER	Foreign key to a HELP_CITY record that the IP address belongs to.
COUNTRYID	NUMBER	Foreign key to a HELP_COUNTRY record that the IP address belongs to.

7.3.7 ESAUSER

This table stores all valid users who are able to log in to Analytics.

Column Name	Type	Description
ID	NUMBER	The primary key.
USERNAME	VARCHAR2 (255 BYTE)	The username to log in.
FIRSTNAME	VARCHAR2 (255 BYTE)	First name of the user.
LASTNAME	VARCHAR2 (255 BYTE)	Fast name of the user.
PASSWORD	VARCHAR2 (255 BYTE)	Password of the user (`md5hash`).
DISABLED	NUMBER	Specifies whether the user account is disabled.
DELETED	NUMBER	Disables the account.
ISADMIN	NUMBER	Defines if the user is able to use the administrator's interface. Use `1` to allow this.
DEFAULTSITEID	NUMBER	Foreign key to a SITE record to use when the user logs in.

7.3.8 IP2COUNTRY

This table stores the IP to country mappings.

Column Name	Type	Description
IPFROM	NUMBER (16)	The start IP address of the mapping.
IPTO	NUMBER (16)	The end IP address of the mapping.
REGISTRY	VARCHAR2 (16 BYTE)	Reserved for internal use.

Column Name	Type	Description
ASSIGNED	VARCHAR2 (16 BYTE)	Reserved for internal use.
CTRY	VARCHAR2 (16 BYTE)	Reserved for internal use.
CNTRY	VARCHAR2 (16 BYTE)	Reserved for internal use.
COUNTRY	VARCHAR2 (32 BYTE)	Foreign key to the HELP_COUNTRY table.

7.3.9 OBJECTTYPEGROUP

This table defines object type groups.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64 BYTE)	Name of the object type group.

7.3.10 SETTING

This table stores various settings in a `key=value` schema, which are used by database jobs or other parts of the application.

Column Name	Type	Description
NAME	VARCHAR2 (255 BYTE)	Name of the setting.
VALUE	VARCHAR2 (255 BYTE)	Value of the setting.
ID	NUMBER	The primary key.

7.3.11 OSGROUP

This table stores the names of operating system groups. For example, group "Windows NT" contains Windows NT4, 2000, and XP. Each operating system is individually listed in the L2_OS table.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64 BYTE)	Name of the group. **Legal values:** Any string up to 64 characters.
ICONPATH	VARCHAR2 (64 BYTE)	Name and path of the icon shown in the "Operating Systems" report. This path is appended to the `IMGPATH` parameter which is defined in the `global.xml` configuration file.

7.3.12 OSIDENTIFICATION

This table is used to identify operating systems based on the browser's agent string. If the agent string contains the pattern of an entry in this table, the agent is assigned to the operating system. If an agent string matches more than one record in this table, the record with the highest priority is taken.

Column Name	Type	Description
ID	INTEGER	The primary key.
OSID	INTEGER	Foreign key to an L2_OS record to the operating system the agent is assigned to if the pattern applies.
PATTERN	VARCHAR2 (50 BYTE)	The pattern to identify the operating system.
PRIORITY	INTEGER	The priority to use if an agent string applies to more than one record.

7.3.13 REC_RECOMMENDATION

This table defines all recommendations for the Engage reports.

Column Name	Type	Description
ID	NUMBER (6)	The primary key.
NAME	VARCHAR2 (125 BYTE)	Name of the recommendation.
DELETED	NUMBER (1)	Marks the record as deleted. (0= false, 1= true).
SITEID	NUMBER (12)	Foreign key to the SITE table to match this recommendation to a site.
RECOMMENDATIONID	VARCHAR2 (21 BYTE)	The WebCenter Sites recommendation asset ID of this recommendation.
DISABLED	NUMBER (1)	Must be set to '0'. Reserved for internal use.

7.3.14 REC_SEGMENT

This table defines all segments for the Engage reports.

Column Name	Type	Description
ID	NUMBER (6)	The primary key.
NAME	VARCHAR2 (124 BYTE)	Name of the segment.
DELETED	NUMBER (1)	Mark this record as deleted. ('0'= False, '1'= True).
SITEID	NUMBER (12)	Foreign key to the SITE table, to match this segment to a site.
SEGMENTID	VARCHAR2 (21 BYTE)	The WebCenter Sites segment asset ID of this segment.
DISABLED	NUMBER (1)	Must be set to '0'. Reserved for internal use.

7.3.15 REPORT

Reports are defined in this table (one line per report).

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR2 (256 BYTE)	Name of the report.
CONFIGPATH	VARCHAR2 (256 BYTE)	This points to the XML configuration file of the report that is relative to the REPORT_INSTDIR parameter in the global.xml configuration file.

7.3.16 REPORT2REPORTGROUP

This table stores the assignment of a report to its group in the report selection panel.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
REPORT	NUMBER (12)	Foreign key to the REPORT table.
REPORTGROUP	NUMBER (12)	Foreign key to the REPORTGROUP record.

7.3.17 REPORTGROUP

This table defines the report groups that are available in the report selection panel.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR2 (32 BYTE)	Name of the group.
PRIORITY	NUMBER	The priority of the group. If a report is assigned to more than one group, then the group with the highest priority is used by default.

7.3.18 REPORTPARAMETER

This table stores additional parameters to a report in a key=value schema.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (128 BYTE)	Name of the parameter.
VALUE	VARCHAR2 (128 BYTE)	Value of the parameter.
REPORTID	NUMBER	Foreign key to a REPORT record.

7.3.19 SEARCHENGINEGROUP

This table stores the names of search engine groups (for example, group "Google" contains google.com/.ca/.de). Each search engine is individually listed in the L2_SEARCHENGINE table.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64 BYTE)	Name of the group. **Legal values:** Any value up to 64 characters.
ICONPATH	VARCHAR2 (64 BYTE)	Name and path of the icon shown in the search engine report. This path is appended to the `imgpath` parameter defined in the `global.xml` configuration file.

7.3.20 SEARCHENGINEIDENT

This table is used to identify search engine referrers. If the referrer contains the pattern of an entry for this table, the session is assigned to the search engine. If a referrer's string matches more than one record in this table, the record with the highest priority is taken.

Column Name	Type	Description
ID	INTEGER	The primary key.
SEARCHENGINEID	INTEGER	Foreign key to an L2_SEARCHENGINE record for the search engine to which the session is assigned, if the pattern applies.
PRIORITY	INTEGER	The priority to use if a referrer string applies to more than one record.
PATTERN	VARCHAR2 (50 BYTE)	The pattern to identify the search engine.

7.3.21 SITE

This table stores all registered sites.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64 BYTE)	Name of the site written exactly as collected from the `image` tag.
LINK	VARCHAR2 (256 BYTE)	URL of the site starting with `http://`. This URL will be used to link to the site from Analytics and to identify the internal referrer.
SCREENX	NUMBER	The screen width that the site is designed for. This is used for the "Screen Resolution" report.
SITEGROUPID	NUMBER	This column is used for custom reports only.
COUNTRYID	NUMBER	Foreign key to the HELP_COUNTRY record for the site.

Column Name	Type	Description
STARTDATE	DATE	Stores the date when the site was registered with Analytics.
LONGNAME	VARCHAR2 (64 BYTE)	This is an optional, longer, more readable name for the site.
USERIDMETHOD	VARCHAR2 (64 BYTE)	Defines how to identify users.
DELETED	NUMBER (1)	This column is not used.
DISABLED	NUMBER (1)	This column is not used.

7.3.22 SITE2OBJECTTYPEGROUP

This table defines which object types are part of which object type group. This setting is site specific.

Column Name	Type	Description
ID	NUMBER	The primary key.
SITEID	NUMBER	Foreign key to a SITE record.
OBJECTTYPEGROUPID	NUMBER	Foreign key to the OBJECTTYPEGROUP table.
OBJECTTYPEID	NUMBER	Foreign key to an L2_OBJECTTYPE record.

7.3.23 SITEGROUP

This table is used by custom reports only.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (64 BYTE)	Name of the site group.

7.3.24 PI_SITEXOBJECTTYPE

This table defines whether the performance indicator is enabled for a site's object type.

Column Name	Type	Description
ID	NUMBER (21)	The primary key.
SITEID	NUMBER (21)	Foreign key to a SITE record.
OBJECTTYPEID	NUMBER (12)	Foreign key to an L2_OBJECTTYPE record.
ENABLE	NUMBER (1)	This defines if the performance indicator is enabled: '0' for false, '1' for true.

7.3.25 USER2USERGROUP

This table stores ESAUSER to USERGROUP relations.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
ESAUSER	NUMBER (12)	Foreign key to an ESAUSER record.
USERGROUP	NUMBER (1)	Foreign key to a USERGROUP record.

7.3.26 USERGROUP

This table defines user groups.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR (128 BYTE)	Name of the user group.

7.3.27 L2_BROWSER

This table stores information about all known browsers used by the "Browsers" report.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR2 (500 BYTE)	Name of the browser.
BROWSERGROUPID	NUMBER	Foreign key to the BROWSERGROUP table. Reserved for internal use.
ICONPATH	VARCHAR2 (64 BYTE)	Image path of the browser.

7.3.28 L2_GROUP

This table is used by custom reports only.

Column Name	Type	Description
ID	NUMBER	The primary key.
NAME	VARCHAR2 (255 BYTE)	Name of the group.
PATH	VARCHAR2 (1024 BYTE)	The path string of the group.
PARENTID	NUMBER	The parent node, which is the foreign key to the L2_GROUP table.
EDATE	DATE	Date when added.
SITEID	NUMBER	Foreign key to the SITE table.

7.3.29 L2_OBJECT

This table stores information about all objects (assets). This table is populated on demand by the "normalize" database job.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
OBJECTTYPEID	NUMBER (24)	Foreign key to the L2_OBJECTTYPE table.
NAME	VARCHAR2 (255 BYTE)	Name of the object (asset). This value will be updated as soon as it changes.
OBJECTID	VARCHAR2 (255 BYTE)	The identification string of the object. Usually this is the WebCenter Sites AssetID.
URL	VARCHAR2 (256 BYTE)	The URL to link to this object.
SITEID	NUMBER	Foreign key to the SITE table.

7.3.30 L2_OBJECTNAMEHISTORY

This table stores the previous names of the L2_OBJECT records.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
OBJECTID	NUMBER (24)	Foreign key to the L2_OBJECT table.
EDATE	DATE	The timestamp when the name was changed.
NAME	VARCHAR2 (255 BYTE)	The previous value.

7.3.31 L2_OBJECTTYPE

This table stores the names of all object types. These object types are referred to as "asset types" in WebCenter Sites.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR2 (128 BYTE)	Name of the object type.
DISABLED	NUMBER (1)	This column is not used.
DELETED	NUMBER (1)	This column is not used.

7.3.32 L2_OS

This table defines the operating systems in the "Operating Systems" report.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR2 (512 BYTE)	Name of the operating system.
OSGROUPID	NUMBER	Foreign key to the OSGROUP table.

7.3.33 L2_PIOBJECTTYPE

This table defines the sites and object types for which the performance indicator is enabled.

Column Name	Type	Description
SITEID	NUMBER	Foreign key to the site to enable the performance indicator.
OBJECTTYPEID	NUMBER	Foreign key to the L2_OBJECTTYPE table, to enable the performance indicator.

7.3.34 L2_SEARCHENGINE

This table stores information about search engines that are used by the "External Search Engines" report.

Column Name	Type	Description
ID	NUMBER (12)	The primary key.
NAME	VARCHAR2 (512 BYTE)	Name of the search engine.
SEARCHENGINEGROUPID	NUMBER	Foreign key to the SEARCHENGINEGROUP table.
KEY	VARCHAR2 (16 BYTE)	Name of the parameter.

7.3.35 L2_VISITOR

This table stores visitors.

Column Name	Type	Description
ID	NUMBER(12)	Primary key.
VISITORID	VARCHAR2(1024 BYTE)	Unique visitor identification string.

7.3.36 TASKSTATUSHISTORY

This table is used to maintain information about work packages as they are processed by the Hadoop-jobs scheduler.

Column Name	Type	Description
IDENTFIER	VARCHAR2(255)	Job name identifier.
ENDDATE	VARCHAR2(128)	Date and time job ended or failed.
INTERNALIDENTIFIER	VARCHAR2(255)	Internal Hadoop job ID.
STARTDATE	VARCHAR2(128)	Date and time job started.
STATUS	VARCHAR2(128)	Current status of the task.
TARGET	VARCHAR2(1024)	Target path of output data.
TYPE	VARCHAR2(128)	Type of job.
WORKPACKAGEPATH	VARCHAR2(1024)	Work package path.

7.3.37 HDFSAGENTSTATUSHISTORY

This table is used to maintain information about files moved from local file systems to HDFS for analytics processing.

Column Name	Type	Description
ID	VARCHAR2(256)	Identifier.
DESTFILENAME	VARCHAR2(1020)	Output raw work package file on HDFS.
ENDDATE	VARCHAR2(20)	End date.
IDENTIFIER	VARCHAR2(128)	Internal identifier.
FILESIZE	NUMBER	Size of the file.
SRCFILENAME	VARCHAR2(1020)	Source file taken from local file system.
STARTDATE	VARCHAR2(20)	Date when transfer started.
STATUSNR	NUMBER	Internal status code.
LOGTIMESTAMP	NUMBER	Timestamp when history registered.

7.3.38 DETAILDEFINITION

This table is used to identify URL parameters that can be used by the system. Unlisted parameters are ignored. This table is especially important if you are creating custom reports. You must enter tag-generated URL parameters into this table in order for the parameters to be retained for processing and reporting purposes.

Column Name	Type	Description
ID	NUMBER(12)	Primary key
NAME	VARCHAR2(64 BYTE)	Name of the parameter.
KEY	VARCHAR2(64 BYTE)	This column must have the same value as the `name` column.
OBJECTTYPEID	NUMBER	This field is deprecated.
SITEID	NUMBER	This field is deprecated.
TYPE	NUMBER	Type of `DETAILDEFINITION` entry.
GROUPFUNCTION	NVARCHAR2(12)	Only for `L2_SESSIONDET` (type 1): the value can be `first`, to use the first value of the session, `last` to use the last value, or `all` to use all values of the session.
AUTODEF	NUMBER	This field is deprecated.

7.4 Database Indexes

This section lists the Analytics database tables and index names.

Database Index Name	Table	Columns
ACCESSRIGHT_PK_IDX	ACCESSRIGHT	ID

Database Index Name	Table	Columns
IDX$$_55B10002	ACCESSRIGHT	SITE, REPORT
BROWSERGROUP_PK_IDX	BROWSERGROUP	ID
BROWSERIDENTIFICATION_PK_IDX	BROWSERIDENTIFICATION	ID
PK_USER	ESAUSER	ID
CITY_COUNTRYID_IDX	HELP_CITY	COUNTRYID
CITY_ID_IDX	HELP_CITY	ID
CITY_NAME_IDX	HELP_CITY	NAME
COUNTRY_CODE_IDX	HELP_COUNTRY	CODE
COUNTRY_ID_IDX	HELP_COUNTRY	ID
HELP_DATES_DAT_IDX	HELP_DATES	DAT
IDX$$_4C6E0002	HELP_DATES	ID
<AUTO GENERATED VALUE>	HELP_DATES	STRDAT
REGION_COUNTRYID_CODE_IDX	HELP_REGION	COUNTRYID, REGIONCODE
REGION_ID_IDX	HELP_REGION	ID
PK_IPQUICKLIST_IDX	IPQUICKLIST	IP
IP2COUNTRY_IPFROM_IDX	IP2COUNTRY	IPFROM
IP2COUNTRY_IPTO_IDX	IP2COUNTRY	IPTO
PK_IP2COUNTRYCITY_BLOCKS_IDX	IP2COUNTRYCITY_BLOCKS	IPTO,IPFROM
PK_L2_BROWSER	L2_BROWSER	ID
GROUP_ID_IDX	L2_GROUP	ID
L2_OBJECT_NAME_IDX	L2_OBJECT	NAME
L2_OBJECT_OBJECTID_IDX	L2_OBJECT	OBJECTID
L2_OBJECT_OBJECTTYPEID_IDX	L2_OBJECT	OBJECTTYPEID
PK_L2_OBJECT	L2_OBJECT	ID
L2_OJECTNH_OBJECTID_IDX	L2_OBJECTNAMEHISTORY	OBJECTID
PK_L2_OBJECTNAMEHISTORY	L2_OBJECTNAMEHISTORY	ID
L2_OBJECTTYPE_NAME_IDX	L2_OBJECTTYPE	NAME
PK_L2_OBJECTTYPE	L2_OBJECTTYPE	ID
INDEX PK_L2_OS	L2_OS	ID
IDX44_4BFF0003	L3_DATEXSITEXBROWSERXCOUNT	DATEID
L3_CITY_DATE_IDX	L3_DATEXSITEXCITYXCOUNT	DATEID
L3_COUNTRY_DATE_IDX	L3_DATEXSITEXCOUNTRYXCOUNT	DATEID
L3_DL_MIME_IDX	L3_DATEXSITEXDLXMIMEXCOUNT	DATEID
IDX$$_4BFF0005	L3_DATEXSITEXENTRYIDXCOUNT	DATEID
IDX$$_4C6D0001	L3_DATEXSITEXENTRYIDXCOUNT	SITEID
IDX$$_4BFF0006	L3_DATEXSITEXEXITIDXCOUNT	DATEID
IDX$$_4C6E0001	L3_DATEXSITEXEXITIDXCOUNT	SITEID

Database Index Name	Table	Columns
L3_GROUP_OI_IDX	L3_DATEXSITEXGROUPXOICOUNT	DATEID
L3_GROUP_SES_IDX	L3_DATEXSITEXGROUPXSESCOUNT	DATEID
L3_GROUP_VIS_DATE_IDX	L3_DATEXSITEXGROUPXVISCOUNT	DATEID
IDX$$_4BFF0007	L3_DATEXSITEXHOSTNAMEXCOUNT	DATEID
L3_ISP_DATE_IDX	L3_DATEXSITEXREGIONXCOUNT	DATEID
L3_JS_IDX	L3_DATEXSITEXJS	DATEID
IDX$$_4BFF0002	L3_DATEXSITEXOBJECTTYPEXCOUNT	DATEID
IDX$$_4BFF0001	CS_L3DATEXSITEXOBJXCOUNT	DATEID
IDX$$_4C1D0001	CS_L3DATEXSITEXOBJXCOUNT	OBJECTID
L3_DURATION_IDX	L3_DATEXSITEXOBJECTXDURATION	DATEID
L3_REGION_DATE_IDX	L3_MONTHXSITEXREGIONXCOUNT	DATEID
<AUTO GENERATED VALUE>	L3_DATEXSITEXSESSIONXQUANTIL	DATEID, SITEID
L3_CITY_MONTH_IDX	L3_MONTHXSITEXCITYXCOUNT	DATEID
L3_COUNTRY_MONTH-IDX	L3_MONTHXSITEXCOUNTRYXCOUNT	DATEID
L3_GROUP_VIS_MONTH_IDX	L3_MONTHXSITEXGROUPXVISCOUNT	DATEID
L3_REGION_MONTH_IDX	L3_MONTHXSITEXREGIONXCOUNT	DATEID
L3_CITY_WEEK_IDX	L3_WEEKXSITEXCITYXCOUNT	DATEID
L3_COUNTRY_WEEK_IDX	L3_WEEKXSITEXCOUNTRYXCOUNT	DATEID
L3_GROUP_VIS_WEEK_IDX	L3_WEEKXSITEXGROUPXVISCOUNT	DATEID
OSIDPRIM	OSIDENTIFICATION	ID
<AUTO GENERATED VALUE>	REC_L3_RECXSEGXCOUNT	DATEID, SEGID, RECID
<AUTO GENERATED VALUE>	REC_L3_RECXSEGXOBJXCOUNT	DATEID, SEGID, RECID, OBJID
PK_REC_RECOMMENDATION	REC_RECOMMENDATION	ID
<AUTO GENERATED VALUE>	REC_RECOMMENDATION	RECOMMENDATION ID
REC_RECOMMENDATION	REC_SEGMENT	ID
<AUTO GENERATED VALUE>	REC_SEGMENT	SEGMENTID
PK_REPORT	REPORT	ID
UNIQUE_REPORT_NAME	REPORT	NAME
PK_REPORTPARAMETER	REPORTPARAMETER	ID
UNIQUE_NAME_REPORTID	REPORTPARAMETER	REPORTID, NAME
PK_REPORT2REPORTGROUP	REPORT2REPORTGROUP	ID
PK_SETTING	SETTING	ID
PK_SITE	SITE	ID
<AUTO GENERATED VALUE>	SITE	NAME
PK_STATUSPARAMETER	PI_SITEXOBJECTTYPE	ID
<AUTO GENERATED VALUE>	PI_SITEXOBJECTTYPE	NAME

Database Index Name	Table	Columns
PK_USERGROUP	USERGROUP	ID
IDX$$_55B10001	USER2USERGROUP	ESAUSER
PK_USER2USERGROUP	USER2USERGROUP	ID