

# Oracle® NoSQL Database Plugins Guide



Release 24.3

G12046-02

October 2024

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

Copyright © 2024, 2024, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# Contents

## 1 About IntelliJ Plugin

---

Setting Up IntelliJ Plug-in	1-1
Creating a NoSQL Project in IntelliJ	1-2
Connecting to Oracle NoSQL Database from IntelliJ	1-2
Managing Tables Using the IntelliJ Plugin	1-4
Perform DDL operations using IntelliJ	1-5
Perform DML operations using IntelliJ	1-9

## 2 About Oracle NoSQL Database Visual Studio Code Extension

---

Installing Oracle NoSQL Database Visual Studio Code Extension	2-1
Connecting to Oracle NoSQL Database from Visual Studio Code	2-3
Managing Tables Using Visual Studio Code Extension	2-4
Perform DDL operations using Visual Studio Code	2-5
Perform DML operations using Visual Studio Code	2-8
Removing a Connection	2-10

## Index

---

## List of Tables

---

1-1	Connection Parameters	1-3
1-2	Program Arguments	1-5
2-1	Oracle NoSQL Database Connection Parameters	2-4
2-2	Create an Oracle NoSQL Database Table	2-6

# 1

## About IntelliJ Plugin

Browse tables and execute queries on your Oracle NoSQL Database data store from IntelliJ.

The Oracle NoSQL Database IntelliJ plugin connects to a running instance of your data store and allows you to:

- View the tables in a well-defined tree structure with Table Explorer.
- View information on columns, indexes, primary key(s), and shard key(s) for a table.
- View column data in a well-formatted JSON Structure.
- Create tables using form-based schema entry or supply DDL statements.
- Drop tables.
- Add new columns using form-based entry or supply DDL statements.
- Drop Columns.
- Create Indexes.
- Drop Indexes.
- Execute SELECT SQL queries on a table and view query results in tabular format.
- Execute DML statements to update, insert, and delete data from a table.

### Topics:

- [Setting Up IntelliJ Plug-in](#)
- [Creating a NoSQL Project in IntelliJ](#)
- [Connecting to Oracle NoSQL Database from IntelliJ](#)
- [Managing Tables Using the IntelliJ Plugin](#)

## Setting Up IntelliJ Plug-in

Learn how to set up the IntelliJ plug-in for your data store.

Perform the following steps:

1. Download the latest Oracle NoSQL Database Java SDK from [GitHub](#).
2. Extract (unzip) the downloaded file in a local repository. Make a note of the path where you extracted the Oracle NoSQL Java SDK. You will need to provide this path when you connect to the Oracle NoSQL Database from IntelliJ.
3. Install the IntelliJ plugin, and restart the IDE.

You have two options to install the plugin:

- Search the Oracle NoSQL Database Connector in the JetBrains plug-in repository, and install it, or
- Download the IntelliJ plugin from Oracle Technology Network, and install the plugin from disk.

 **Tip:**

Don't extract the downloaded plugin zip file. Select the plugin in the zip format while installing it from disk.

After you successfully set up your IntelliJ plugin, create a NoSQL project, and connect it to your data store.

## Creating a NoSQL Project in IntelliJ

Learn how to create a NoSQL project in IntelliJ.

Perform the following steps:

1. Open IntelliJ IDEA. Click **File > New > Project**.
2. Enter a value for **Project Name** and **Project Location**, and click **Create**.
3. Once your NoSQL project is created, you can browse the example java files from the Project Explorer window.
4. Make sure Notifications are enabled for your Oracle NoSQL project. To enable Notifications, press `Alt+⌨` to open Main Menu. Click **View**, expand **Tool Windows >**

**Notifications.** A Notification icon  appears on the right tool window bar.

After you successfully create a NoSQL project in IntelliJ, connect your project to your data store.

## Connecting to Oracle NoSQL Database from IntelliJ


Learn how to connect your NoSQL project to Oracle NoSQL Database the data store using the IntelliJ plugin.

Prerequisites:




To create a successful connection to your Oracle NoSQL Database data store, ensure that:

- The data store is deployed and running.
- The Oracle NoSQL Database Proxy is started. See [Configuring the Proxy](#). Starting the release 19.5, Proxy is bundled along with the Oracle NoSQL Database download package.

Perform the following steps:

1. Open your NoSQL project in IntelliJ.
2. Click the **task** icon  in the **Schema Explorer** window to open the **Settings** dialog for the plugin.
3. Expand **Tools > Oracle NoSQL** in the Settings Explorer, and click **Connections**. You can view all the existing connections for the on-premises profile type under the Connections dropdown.
4. Select **Onprem** from the drop-down menu for the Profile type.
5. Click **Add Connection**. Enter values for the following connection parameters, and click **OK**.

**Table 1-1 Connection Parameters**

Parameter	Description
Connection Name	<p>A unique name, that is given to a specific connection specification is mandatory from the plugin version 1.5.1. Updating the Connection Name field is recommended after upgrading the plugin from version 1.4.0 or lower.</p> <div style="border: 1px solid #0070C0; padding: 10px; margin-top: 10px;"> <p> <b>Note:</b></p> <p>You can add multiple connections and the stored connection specifications are persistent.</p> </div>
Proxy URL	<p><code>http://&lt;proxy_host&gt;:&lt;proxy_http_port&gt;</code> or <code>https://</code> <code>&lt;proxy_host&gt;:&lt;proxy_http_port&gt;</code> where:</p> <ul style="list-style-type: none"> <li>• <code>http</code> or <code>https</code> indicates the store security. For a secure data store, the proxy URL begins with <code>https</code>.</li> <li>• <code>proxy_host</code> is the host name of the machine to host the proxy service. This should match the host you configured earlier.</li> </ul> <p>See <a href="#">Configuring the Proxy</a>.</p>
SDK Path	<p>Complete path to the directory where you extracted the Oracle NoSQL Java SDK. For example, <code>D:\oracle-nosql-java-sdk-5.4.14\oracle-nosql-java-sdk</code></p>
Namespace	<p>Provide the name of your namespace. This is optional. If no value is provided then the default namespace <code>sysdefault</code> is used.</p> <div style="border: 1px solid #0070C0; padding: 10px; margin-top: 10px;"> <p> <b>Note:</b></p> <p>You can add one or more namespaces to your store, create tables within them, and grant permission for users to access namespaces and tables.</p> </div>
Security	<p>Select <b>SSL</b> for secure data store. In case, you are creating connection to a non-secure KVStore, select <b>None</b>. The default value is <b>SSL</b>.</p> <div style="border: 1px solid #0070C0; padding: 10px; margin-top: 10px;"> <p> <b>Note:</b></p> <p>In case of secure data store, the proxy URL must begin with <code>https</code>.</p> </div>

**Table 1-1 (Cont.) Connection Parameters**

Parameter	Description
Username	User name to connect to the secure store. This value is required only if you select <b>SSL</b> for the Security parameter.
Password	Password to connect to the secure store. This value is required only if you select <b>SSL</b> for the Security parameter.
TrustStore	Browse to the location where the certificate trust file is placed. See Using the Proxy in a secure data store.
Passphrase	Give a value for the passphrase. A passphrase refers to a secret used to protect an encryption key. This is optional.

 **Note:**

If you are updating the plugin from version 1.4.0 or lower, all the stored connections migrate to the new version. In this case, the Connection Name will be the same as Proxy URL. Follow the below step to change the Connection Name.

- The IntelliJ Plugin saves the connection details in the connection name specified. To modify the connection details, choose the connection name in the drop-down for **Connections**. Click **Modify Connection**. You can change any of the connection parameters (mentioned above) and click **OK** to save the settings. To remove a connection name from the plugin, choose the connection name and click **Delete Connection**. Once you confirm the action to delete, the connection name is removed from the plugin.
- Click the Web icon in the Schema Explorer. The list of existing connections is displayed in the dropdown box. The connection will be displayed in the NoSQL tool window in the following format: `Connection Name:Proxy URL:Namespace(if specified)`. Choose the connection and click **OK**. The IntelliJ plugin connects your project to the Oracle NoSQL Database KVStore and displays its schema in the Schema Explorer window.
- Click the **Database** icon in the Schema Explorer. A calculator tool is opened in a new window. You can use this tool to calculate the optimalJE cache size, storage size, and the number of shards required, based on the inputs you provide. You first need to provide the following inputs - Table Id (Any unique integer to identify the table), primary key size, row size, row count, and optionally the size of the indexes and click **Add Table**. You can then select the table added and click **CALCULATE** to view the JE Cache size (in GB), number of shards and Storage size (in TB).

After you successfully connect your project to your Oracle NoSQL Database data store, you can manage the tables and data in your schema.

## Managing Tables Using the IntelliJ Plugin

Learn how to create tables and view table data in your data store from IntelliJ.

After connecting to the Oracle NoSQL Database, you can execute the examples downloaded with Oracle NoSQL Java SDK to create a sample table. With the help of the IntelliJ Plugin, you can view the tables and their data in the Schema Explorer window.

Execute an example program:



1. Open the NoSQL project connected to your Oracle NoSQL Database.
2. Locate and click `BasicTableExample` in the Project Explorer window. You will find this in the **examples** folder under `oracle-nosql-java-sdk`. By looking at the code, you can notice that this program creates a table called `audienceData`, puts two rows into this table, queries the inserted rows, deletes the inserted rows, and finally drops the `audienceData` table.
3. To pass the required arguments, click **Run > Edit Configurations**. Enter the following program arguments, and click **OK**.

**Table 1-2 Program Arguments**

Program Arguments	More Information
<code>http://&lt;proxy_host&gt;:&lt;proxy_http_port&gt; -useKVProxy</code>	For example, if your Proxy URL is <code>http://&lt;proxy_host&gt;:8080</code> , the program argument must be <code>http://&lt;proxy_host&gt;:8080 -useKVProxy</code> .

4. To execute this program, click **Run > Run 'BasicExampleTable'** or press **Shift + 10**.
5. Verify the logs in the terminal to confirm that the code executed successfully. You can see the display messages that indicate table creation, rows insertion, and so on.

 **Tip:**

As the `BasicExampleTable` deletes the inserted rows and drops the `audienceData` table, you can't view this table in the Schema Explorer. If you want to see the table in the Schema Explorer, comment the code that deletes the inserted rows and drops the table, and rerun the program.

6. To view the tables and their data:
  - a. Locate the Schema Explorer, and click the **Refresh** icon to reload the schema.
  - b. Locate the `audienceData` table under your tenant identifier, and expand it to view its columns, primary key, and shard key details.
  - c. Double-click the table name to view its data. Alternatively, you can right-click the table and select **Browse Table**.
  - d. A record viewer window appears in the main editor. Click **Execute** to run the query and display table data.
  - e. To view individual cell data separately, double-click the cell.

## Perform DDL operations using IntelliJ

You can use IntelliJ to perform DDL operations.

Some of the DDL operations that can be performed from inside the IntelliJ plugin are

- [CREATE TABLE](#)
- [DROP TABLE](#)
- [CREATE INDEX](#)
- [DROP INDEX](#)

- [ADD COLUMN](#)
- [DROP COLUMN](#)
- [EXECUTE DDL](#)

### CREATE TABLE

- Locate the Schema Explorer, and click the **Refresh** icon to reload the schema.
- Right click the connection name and choose **Create Table**.
- In the prompt, enter the details for your new table. You can create the Oracle NoSQL Database table in two modes:
  - **\*\*Simple DDL Input\*\*** : You can use this mode to create the table declaratively, that is, without writing a DDL statement.
  - **\*\*Advanced DDL Input\*\*** : You can use this mode to create the table using a DDL statement.
- You have the option to view the DDL statement before creating. Click **Show DDL** to view the DDL statement formed based on the values entered in the fields in the Simple DDL input mode. This DDL statement gets executed when you click Create.
- Click **Create** to create the table.
- To create a child table, right click on the desired table and choose **Create Child Table**. You can create a child table in two modes:
  - **\*\*Simple DDL Input\*\*** : You can use this mode to create a child table by simply entering a table name along with other required details.
  - **\*\*Advanced DDL Input\*\*** : You can use this mode to create a child table using a DDL statement.

For more details on child tables, see Table Hierarchies in *Developer's Guide*.

- Click **Create** to create a child table.
- You have an option to view the DDL statement after creating a table. Right click on the existing table. Choose **View Table DDL**. To copy the DDL statement, click **Copy to Clipboard**. Click **OK** to close the dialog box.

### DROP TABLE

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Right click on the table that you want to drop. Choose **Drop Table**.
- A confirmation window appears, click **Ok** to confirm the drop action.

### CREATE INDEX

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Right click on the table where index need to be created. Choose **Create Index**.
- In the Create Index panel, you have an option to create index in two modes:
  - **\*\*Form Based Index Creation(Simple DDL Input)\*\*** : Enter the details for creating an index without writing any DDL statement. Specify the name of the index and the columns to be part of the index. If the column is of JSON data type, you see an additional field called "JSON Path to Index Field" appear. Enter the path to the location of the JSON field, and choose the data type for it.

- **\*\*Create Index as DDL Statement (For Advanced DDL input)\*\*** : Enter a valid DDL statement to create an index. It can also include complex data type i.e. array, map, and record.
- Click **Add Index**.

### DROP INDEX

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Click on the target table to see the listed columns, Primary Keys, Indexes and Shard Keys.
- Locate the target-index which has to be dropped and right-click on it. Click **Drop Index**.
- A confirmation window appears, click **Ok** to confirm the drop action.

### ADD COLUMN

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Right click on the table where column needs to be added. Choose **Add Column**.
- You can add new COLUMNS in two modes:
  - Simple DDL Input : You can use this mode to add new columns without writing a DDL statement. In case of binary or fixed binary select the data type as `Binary`. For fixed binary, enter the size of the file in the `Size` field and keep the field null in case of binary data type.
  - Advanced DDL Input : You can use this mode to add new columns into the table by supplying a valid DDL statement. This mode can also create columns of complex data type. For example, array, map, or record and also in nested format.
- In both the modes, specify the name of the column and define the column with its properties - datatype, default value and whether it is nullable.
- Click **Add Column**.

### DROP COLUMN

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Click on the target table to see the listed columns, Primary Keys, Indexes and Shard Keys.
- Locate the target-column which has to be dropped and right-click on it. Click **Drop Column**.
- A confirmation window appears, click **Ok** to confirm the drop action.

### EXECUTE DDL

You can execute any table-specific DDL statements and table-independent operations like managing namespaces, regions, and roles using the Execute DDL option.

1. Click on the Execute icon (a triangle) in the Schema Explorer to open the **Execute DDL** window. Alternatively, you can also right-click the connection name and choose **Execute DDL**.
2. In the new window, enter the SQL statement which is a System Request (like creating or dropping region/role/namespace, etc). Click **Execute**. The result of the DDL is displayed in the lower portion of the window.

 **Note:**

You can also perform any other DDL operation like CREATE TABLE from this window.

Operations supported using Execute DDL:

- **Create a Namespace:** You can add one or more namespaces to your store, create tables within them, and grant permission for users to access namespaces and tables. Example:

```
CREATE NAMESPACE ns1
```

- **Create a remote region:** You can create remote regions from each local region in a Multi-Region NoSQL Database. Example:

```
CREATE REGION fra
```

- **Add a local region:** You can set a name to the local region in a Multi-Region NoSQL Database. Example:

```
SET LOCAL REGION lnd
```

- **Drop a region:** You can remove a region. Example:

```
DROP REGION fra
```

- **Create an on-premises Multi-Region table:** You can create an on-premises Multi-Region table on a data store and specify the list of regions that the table should span. You can also expand a Multi-Region table by adding more regions or contract a Multi-Region table by removing the table from any existing region. Example: Create a Multi-Region Table

```
CREATE TABLE users(id INTEGER, name STRING, team STRING,  
PRIMARY KEY(id)) IN REGIONS fra,lnd
```

Example: Expand a Multi-Region Table

```
CREATE REGION dub;  
ALTER TABLE users ADD REGIONS dub;
```

Example: Contract a Multi-Region Table

```
ALTER TABLE users DROP REGIONS dub;
```

 **Note:**

Before using the IntelliJ plugin to create/expand or contract a Multi-Region table, you should complete all the tasks for configuring a Multi-Region data store. This includes deploying the data store, configuring the XRegion service, starting the XRegion agents, and running the proxy. See [Configuring Multi-Region Data Stores](#) for more details.

## Perform DML operations using IntelliJ

You can add data, modify existing data and query data from tables using IntelliJ plugin.

### Insert data

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Right click on the table where a row needs to be inserted. Choose **Insert Row**.
- In the Insert Row panel, enter the details for inserting a new row. You can INSERT a new ROW in two modes:
  - Simple Input : You can use this mode to insert the new row without writing a DML statement. Here a form based row fields entry is loaded, where you can enter the value of every field in the row.
    - \* For binary data type, the string typed in should be a valid Base64 encoding of a binary value or select the file to upload in the desired column.
    - \* For fixed binary data type, the string typed in should be a valid Base64 encoding of a binary value or upload the file of size defined during the creation of the particular column.

 **Note:**

The file format you upload for binary data type should only have `.bin` extension.

- Advanced JSON Input : You can use this mode to insert a new row into the table by supplying a JSON Object containing the column name and its corresponding value as key-value pairs. The input can also be of complex data type i.e. array, map, record.
- Click **Insert Row**.

### Modify Data - UPDATE ROW/DELETE ROW

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Right click on the table where a row needs to be inserted. Choose **Browse Table**.
- In the textbox on the left, enter the SQL statement to fetch data from your table. Click **Execute** to run the query.
- To view individual cell data separately, click the table cell.
- To perform DML operations like Update and Delete Row, right-click on the particular row. Pick your option from the context-menu that appears.
  - Delete Row : A confirmation window appears, click **Ok** to delete the row.
  - Update Row : A separate HTML panel opens below the listed rows, containing the column names and its corresponding value in a form-based entry and as a JSON key-pair object. You can choose either of the two methods and supply new values.

 **Note:**

In any row, PRIMARY KEY and GENERATED ALWAYS AS IDENTITY columns cannot be updated.

### Query tables

- Locate the Schema Explorer, and click the Refresh icon to reload the schema.
- Right click on the table and choose **Browse Table**.
- In the textbox on the left, enter the SELECT statement to fetch data from your table.
- Click **Execute** to run the query. The corresponding data is retrieved from the table.
- Right click on any row and click **Download JSON**. In the dialog box, navigate to the location where you want to save the file and click **Save**. Once the file is downloaded, a notification appears at the bottom-right of the screen. Click the link to open the downloaded file. The file opens in the browser.
  - In case of Binary data type, simply click **Download Binary Object** in the output.
- Click **Download Query Result**, to download all the data in the query result. In the dialog box, navigate to the location where you want to save the file and click **Save**. In case of multiple rows, a progress bar appears on the bottom-right of the screen to showing the number of rows downloaded in real time. Once the file is downloaded, a notification appears at the bottom-right of the screen. Click the link to open the downloaded file. The file opens in the browser.
- Click **Show Query Plan** to view the execution plan of the query.
- Click the **Previous Commands** dropdown, to view the top 20 recently executed SQL statements that had provided an output.

 **Note:**

The dropdown will only show SQL statements related to the table you are working on.

### Schema Explorer

- In the Schema Explorer window, you can verify the full data type of a particular column. Locate the particular column and the data type is followed by the column name.

# 2

## About Oracle NoSQL Database Visual Studio Code Extension

The Oracle NoSQL Database provides an extension for [Microsoft Visual Studio Code](#) which lets you connect to a running instance of Oracle NoSQL Database.

You can use Oracle NoSQL Database Visual Studio (VS) Code extension to:

- View the tables in a well-defined tree structure with Table Explorer.
- View information on columns, indexes, primary key(s), and shard key(s) for a table.
- View column data in a well-formatted JSON Structure.
- Create tables and child tables using form-based schema entry or supply DDL statements.
- Drop tables.
- Add new columns using form-based entry or supply DDL statements.
- Drop Columns.
- Create Indexes.
- Drop Indexes.
- Execute SELECT SQL queries on a table and view query results in tabular format.
- Execute DML statements to update, insert, and delete data from a table.
- Download the Query Result after running the SELECT query into a JSON file.
- Download each row of the result obtained after running the SELECT query into a JSON file.

## Installing Oracle NoSQL Database Visual Studio Code Extension

You can install the Oracle NoSQL Database VS Code extension in two ways. Install from the Visual Studio Marketplace for online installation or install from the VSIX package using \*.vsix file for offline installation.

Before you can install the Oracle NoSQL Database Visual Studio (VS) Code extension, you must install Visual Studio Code. You can download Visual Studio Code from [here](#).

- 
- [Install from Visual Studio Marketplace](#)
  - [Install from a VSIX](#)

### Install from Visual Studio Marketplace

1. In Visual Studio Code, click the **Extensions** icon in the left navigation.



Alternatively, you can open the **Extensions** view by pressing:

- (Windows and Linux) Control + Shift + X
  - (macOS) Command + Shift + X.
2. Search Oracle NoSQL Database Connector in the extension marketplace.
  3. Click Install on the Oracle NoSQL Database Connector extension

## Install from a VSIX

1. Download the VSIX file for Oracle NoSQL Database from Oracle NoSQL Database Downloads site.
2. In Visual Studio Code, click the **Extensions** icon in the left navigation.



Alternatively, you can open the **Extensions** view by pressing:

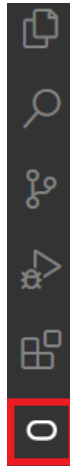
- (Windows and Linux) Control + Shift + X
  - (macOS) Command + Shift + X.
3. In the **Extensions** view, Click the **More Actions (...)** menu and then click **Install from VSIX...**
  4. Browse to the location where the \*.vsix file is stored and click **Install**.
-



# Connecting to Oracle NoSQL Database from Visual Studio Code

Oracle NoSQL Database Visual Studio (VS) Code extension allows you to connect to Oracle NoSQL Database by filling in the connection information in the specific fields.

1. In Visual Studio Code, click the **Oracle NoSQL DB** view in the **Activity Bar**.



2. Open the Oracle NoSQL DB **Show Connection Settings** page from the Command Palette or the **Oracle NoSQL DB** view in the **Activity Bar**.

- Open from Command Palette
  - a. Open the **Command Palette** by pressing:
    - (Windows and Linux) Control + Shift + P
    - (macOS) Command + Shift + P
  - b. From the Command Palette, select **OracleNoSQL: Show Connections Settings**.

 **Tip:**

Enter `oraclenosql` in the Command Palette to display all of the Oracle NoSQL DB commands you can use.

- Open from Oracle NoSQL DB View
  - a. Expand the **Table Explorer** pane in the left navigation if it's collapsed.
  - b. Click **Add Connection** to open the Oracle NoSQL DB **Show Connection Settings** page.
- 3. In the **Show Connection Settings** page, click **Onprem** to connect to Oracle NoSQL Database.
- 4. Enter the connection information.

**Table 2-1 Oracle NoSQL Database Connection Parameters**

Field	Description
Endpoint:	<p>Service URL of the Oracle NoSQL Database Proxy.</p> <ul style="list-style-type: none"> <li>• <code>http://&lt;proxy_host&gt;:&lt;proxy_http_port&gt;</code></li> <li>• <code>https://&lt;proxy_host&gt;:&lt;proxy_https_port&gt;</code></li> </ul> <p>where,</p> <ul style="list-style-type: none"> <li>• <code>http</code> or <code>https</code> indicates the store security. For a secure KVStore, the proxy URL begins with <code>https</code>.</li> <li>• <code>proxy_host</code> is the host name of the machine to host the Oracle NoSQL Database Proxy service. For more information, see <a href="#">Configuring the Proxy</a>.</li> </ul>
Security:	<p>Select SSL for secure KVStores. In case you are creating the connection to a non-secure KVStore, select <code>None</code>. The default value is <code>None</code>.</p>
Username:	<p>User name to connect to the secure store. This value is required only if you select SSL for the Security parameter.</p>
Password:	<p>Password to connect to the secure store. This value is required only if you select SSL for the Security parameter.</p>
Certificate File:	<p>Browse to the location of the SSL certificate file (for example, <code>certificate.pem</code>). See <a href="#">Using the Proxy in a secure data store</a>.</p>

5. Click **Connect**.
6. Click **Reset** to clear the saved connection details from the workspace.

 **Note:**

When you want to connect to a on-premise secure cluster using Visual Studio Code Extension, you need to do one of the following if you have self-signed certificates:

- Add the self-signed certificate to the trusted root store in your Windows configuration (using Microsoft Management Console).
- Open Visual Studio Code. Go to File > Preferences > Settings > Application > Proxy. In the **Proxy Support** menu select **off**.

## Managing Tables Using Visual Studio Code Extension

Once you connect to your deployment using Oracle NoSQL Database Visual Studio (VS) Code extension, use the **TABLE EXPLORER** located on the left navigation to:

- Explore your tables, columns, indexes, primary keys, and shard keys.

- Create new tables.
- Drop existing tables.
- Create Indexes.
- Drop Indexes.
- Add columns.
- Drop Columns.
- Insert data into table.
- Execute SELECT SQL queries.

### Explore tables, columns, indexes and keys

When you expand an active connection, Oracle NoSQL Database VS Code shows the tables in that deployment.

- Click the table name to view its columns, indexes, primary key(s), and shard key(s). The column name displays along with its data type.
- You can refresh the schema or table at any time to re-query your deployment and populate Oracle NoSQL Database with the most up-to-date data.
  - In the **TABLE EXPLORER**, locate the connection and click the Refresh icon to reload the schema. Alternatively, you can right-click the connection and select **Refresh Schema**.
  - In the **TABLE EXPLORER**, locate the table name and click the Refresh icon to reload the table. Alternatively, you can right-click the table name and select **Refresh Table**.

## Perform DDL operations using Visual Studio Code

You can use Visual Studio Code to perform DDL operations.

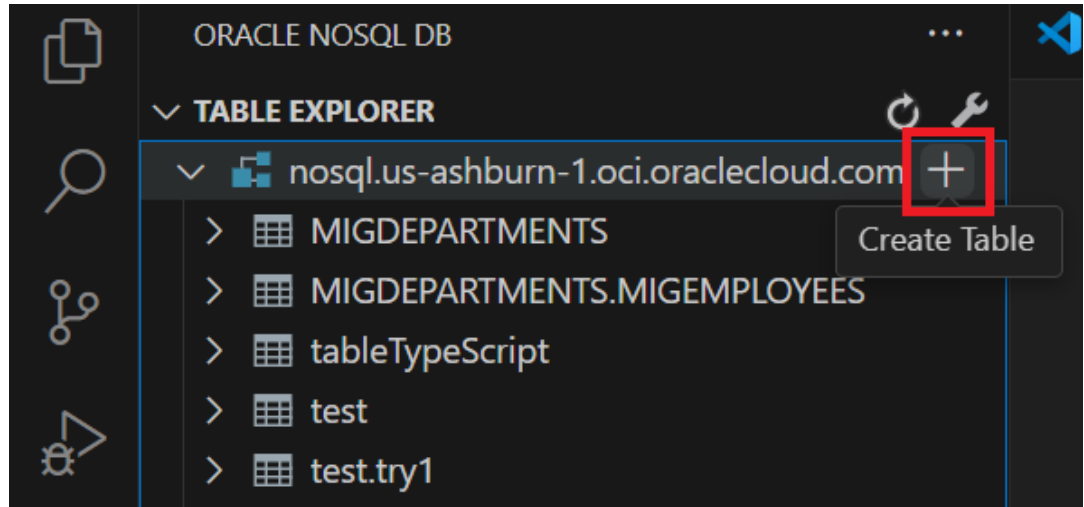
Some of the DDL operations that can be performed from inside the Visual Studio Code plugin are:

- [CREATE TABLE](#)
- [DROP TABLE](#)
- [CREATE INDEX](#)
- [DROP INDEX](#)
- [ADD COLUMN](#)
- [DROP COLUMN](#)

### CREATE TABLE


You can create the Oracle NoSQL Database table in two modes:

- **Simple DDL Input:** You can use this mode to create the Oracle NoSQL Database table declaratively, that is, without writing a DDL statement.
  - **Advanced DDL Input:** You can use this mode to create the Oracle NoSQL Database table using a DDL statement.
1. Hover over the Oracle NoSQL Database connection to add the new table.
  2. Click the **Plus** icon that appears or right-click on the database connection name and click **Create Table**.



3. In the **Create Table** page, select **Simple DDL Input**.

**Table 2-2 Create an Oracle NoSQL Database Table**

Field	Description
<b>Table Name:</b>	Specify a unique table name.
<b>Column Name</b>	Specify a column name for the primary key in your table.
<b>Column Type</b>	Select the data type for your primary key column.
<b>Set as Shard Key</b>	Select this option to set this primary key column as shard key. Shard key is to distribute data across the Oracle NoSQL Database cluster for increased efficiency, and to position records that share the shard key locally for easy reference and access. Records that share the shard key are stored in the same physical location and can be accessed atomically and efficiently.
<b>Remove</b>	Click this button to delete an existing column.
<b>+ Add Primary Key Column</b>	Click this button to add more columns while creating a composite (multi-column) primary key.
<b>Column Name</b>	Specify the column name.
<b>Column Type</b>	Select the data type for your column.
<b>Default Value</b>	(optional) Specify a default value for the column.
<div style="border: 1px solid #0070C0; padding: 5px; margin: 5px 0;"> <p> <b>Note:</b> Default values can not be specified for binary and JSON data type columns.</p> </div>	
<b>Not Null</b>	Select this option to specify that a column must always have a value.
<b>Remove</b>	Click this button to delete an existing column.
<b>+ Add Column</b>	Click this button to add more columns.

**Table 2-2 (Cont.) Create an Oracle NoSQL Database Table**

Field	Description
<b>Unit</b>	Select the unit ( <b>Days</b> or <b>Hours</b> ) to use for TTL value for the rows in the table.
<b>Value</b>	Specify expiration duration for the rows in the table. After the number of days or hours, the rows expire automatically, and are no longer available. The default value is zero, indicating no expiration time.

 **Note:**

Updating Table Time to Live (TTL) does not change the TTL value of any existing data in the table. The new TTL value applies *only* to those rows that are added to the table after this value is modified and to the rows for which no overriding row-specific value has been supplied.

4. Click **Show DDL** to view the DDL statement formed based on the values entered in the fields in the **Simple DDL input** mode. This DDL statement gets executed when you click **Create**.
5. Click **Create**.
  - To create a child table, right click on the desired table and choose **Create Child Table**. You can create a child table in two modes:
    - **Simple DDL Input**: You can use this mode to create a child table by simply entering a table name along with other required details.
    - **Advanced DDL Input**: You can use this mode to create a child table using a DDL statement.

For more details on child tables, see Table Hierarchies in *Developer's Guide*.

- Click **Create** to create a child table.
- You have an option to view the DDL statement after creating a table. Right click on the existing table. Choose **View Table DDL**. To copy the DDL statement, click **Copy to Clipboard**. Click **OK** to close the dialog box.

#### DROP TABLE

1. Right-click the target table.
2. Click **Drop Table**.
3. Click **Yes** to drop the table.

#### CREATE INDEX

- Locate the Table Explorer, and click the Refresh Schema icon to reload the schema.
- Right click on the table where index need to be created. Choose **Create Index**.
- In the Create Index panel, you have an option to create index in two modes:

- **Simple Input:** Specify the name of the index and the columns to be part of the index. If the column type is JSON, you see an additional field called "JSON Path to Index Field". Enter the path to the location of the JSON field, and choose the data type for it.
- **Using Advance DDL:** Enter a valid DDL statement to create an index on any column(s). It can also include complex data type i.e. array, map, and record.
- Click **Add Index**.

### DROP INDEX

- Locate the Table Explorer, and click the Refresh Schema to reload the schema.
- Click on the table where the index needs to be removed. The list of indexes are displayed below the column names.
- Right click on the index to be dropped. Click **Drop Index**.
- A confirmation window appears, click **Ok** to confirm the drop action.

### ADD COLUMN

- Locate the Table Explorer, and click the Refresh Schema to reload the schema.
- Right click on the table where column needs to be added. Click **Add columns**.
- In the Add Column(s) Panel, you have an option to add column in two modes:
  - **Simple DDL Input:** Specify the name of the column and define the column with its properties - datatype, default value and whether it is nullable. In case of binary or fixed binary select the data type as `Binary`. For fixed binary enter the size of the file in the `Size` field and keep the field null in case of binary data type.
  - **Advanced DDL Input:** You can use this mode to add new columns into the table by supplying a valid DDL statement, as well as, create columns with complex data type (e.g. array, map, or record and also in nested format).
- Click **Add New Columns**.

### DROP COLUMN

- Locate the Table Explorer, and click the Refresh Schema to reload the schema.
- Expand the table where column needs to be removed.
- Right click the column to be removed and choose **Drop Column**.
- A confirmation window appears, click **Ok** to confirm the drop action.

## Perform DML operations using Visual Studio Code

You can add data, modify existing data and query data from tables using Visual Studio Code plugin.

### Insert Data

- Locate the Table Explorer, and click the Refresh Schema to reload the schema.
- Right click on the table where a row needs to be inserted. Choose **Insert Row**.
- In the Insert Row panel, enter the details for inserting a new row. You can INSERT a new ROW in two modes:

- Simple Input : You can use this mode to insert the new row without writing a DML statement. Here a form based row fields entry is loaded, where you can enter the value of every field in the row.
  - \* For binary data type, the string typed in should be a valid Base64 encoding of a binary value or select the file to upload in the desired column.
  - \* For fixed binary data type, the string typed in should be a valid Base64 encoding of a binary value or upload the file according to the size specified for the column.

 **Note:**

The file format you upload for binary data type should have the .bin extension.

- Advanced JSON Input : You can use this mode to insert a new row into the table by supplying a JSON Object containing the column name and its corresponding value as key-value pairs. The input can also be a complex data type i.e. array, map, record.
- Click **Insert Row**.

**Modify Data - UPDATE ROW/DELETE ROW:**

- Locate the Table Explorer, and click the Refresh Schema to reload the schema.
- Click on the table where data needs to be modified.
- In the textbox on the right under SQL>, enter the SQL statement to fetch data from your table. Click > to run the query.
- To view individual cell data separately, click the table cell.
- To perform DML operations like Update and Delete Row, right-click on the particular row. Pick your option from the context-menu that appears.
  - Delete Row : A confirmation window appears, click **Ok** to delete the row.
  - Update Row : A separate HTML panel opens below the listed rows, containing the column names and its corresponding value in a form-based entry or provide the input as ON key-pair object. You can choose either of the two methods and supply new values.

 **Note:**

In any row, PRIMARY KEY and GENERATED ALWAYS AS IDENTITY columns cannot be updated.

**Executing SQL Queries for a Table**

- Locate the Table Explorer, and click the Refresh Schema to reload the schema.
- Right click on the table and choose **Browse Table**.
- In the textbox on the right under SQL>, enter the SELECT statement to fetch data from your table.
- Click > to run the query. The corresponding data is retrieved from the table.

- Right click on any row and click **Download JSON**. The single row gets downloaded into a JSON file. In the dialog box, navigate to the location where you want to save the file and click **Save**.
- Click **Download Query Result** to save the complete result of the SELECT statement as a JSON file. In the dialog box, navigate to the location where you want to save the file and click **Save**.
- Click **Fetch All Records** to retrieve all data from the table.
- Click **Show Query Plan** to view the execution plan of the query.
- Click the **Previous Commands** dropdown, to view the recently executed SQL statements that had provided an output.

 **Note:**

The dropdown will only show SQL statements related to the table you are dealing with.

### Table Explorer

- In the Table Explorer window, you can verify the data type of a particular column. Locate the particular column and the data type is followed by the column name.

## Removing a Connection

Oracle NoSQL Database Connector provides two methods to remove a connection from Visual Studio (VS) Code.

You can:

- Remove a connection with the Command Palette, or
- Remove a connection from the Oracle NoSQL DB view in the Activity Bar.

 **Note:**

Removing a connection from Visual Studio Code deletes the persisted connection details from the current workspace.

- 
- [Remove Connection from Oracle NoSQL DB View](#)
  - [Remove Connection with Command Palette](#)

### Remove Connection from Oracle NoSQL DB View

1. Expand the **TABLE EXPLORER** pane in the left navigation if it's collapsed.
2. Right-click the connection you want to remove, then click **Remove Connection**.

### Remove Connection with Command Palette

1. Open the **Command Palette** by pressing:



- (Windows and Linux) Control + Shift + P
  - (macOS) Command + Shift + P
2. From the Command Palette, select **OracleNoSQL: Remove Connection**.



**Tip:**

Enter `oraclenosql` in the Command Palette to display all of the Oracle NoSQL DB commands you can use.

---

# Glossary

# Index