Using Oracle® SQL Developer Web

Release 20.2
F32156-01
July 2020

Using Oracle® SQL Developer Web, Release 20.2

F32156-01

Primary Author: Celin Cherian

Contributing Authors: Mike Fitch, Chuck Murray

Contributors: Jeff Smith, Syme Kutz, Victor Martinez, Renato Uriel Gutierrez Salas, Antonio Hernandez, Sofia Isadora Padilla Munoz, Miguel Sanchez Villafan, Viktor Perelmouter, Tulika Das, Eric Paapanen

Contents

Preface

1 Changes in Release 20.2 for Using Oracle SQL Developer Web

2 About Oracle SQL Developer Web

3 The Database Actions Page

4 The Actions Overview Page

5 The SQL Worksheet Page

6 The Data Modeler Page

7 Creating and Editing Database Objects

8 The User Management Page

9 The Monitoring Pages

Preface

This document provides information about Oracle SQL Developer Web, a web-based interface of Oracle SQL Developer.

Audience

This online help is intended for those using Oracle SQL Developer Web included with Oracle REST Data Services.

Documentation Accessibility

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

Access to Oracle Support

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

Product Accessibility

Oracle SQL Developer Web provides features to support accessibility. See *SQL Developer Web Accessibility Guide*.

Related Documents

Oracle REST Data Services Installation, Configuration, and Development Guide.

Oracle SQL Developer User's Guide.

To download release notes, installation documentation, white papers, or other collateral for SQL Developer, go to the Oracle Technology Network (OTN) at

`http://www.oracle.com/technetwork/`

For the PL/SQL page on OTN, see `http://www.oracle.com/technetwork/database/features/plsql/`

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.

Third-Party License Information

See the Third-Party License Information section in *Oracle REST Data Services Installation, Configuration, and Development Guide*

1 Changes in Release 20.2 for *Using Oracle SQL Developer Web*

This sections lists the changes in this release.

New Features

New features in this release include:

- A new landing page for SQL Developer Web.

  See The Database Actions Page

- Instance Viewer in the Monitoring menu enables a DBA user to see a graphical representation of information about the instance associated with a connection.

  See The Instance Viewer Page

- Load data from JSON, XML, or AVRO files into an existing or new table.

  See Loading Data Using the SQL Worksheet

2 About Oracle SQL Developer Web

Oracle SQL Developer Web is a web-based application that uses ORDS (Oracle REST Data Services) to provide many of the database development and administration features of desktop-based Oracle SQL Developer.The main features include running SQL statements and scripts in the worksheet, exporting data, monitoring databases, and creating Data Modeler diagrams.

> **Note:**
>
> Some features of Oracle SQL Developer Web are only available if you sign in as a user with the DBA role. For such features, a "restricted availability" statement appears at the start of the feature description; for example:
>
> 👁 Available only if you signed in as a database user with the DBA and PDB_DBA roles.

For creating users and assigning roles, see The Create/Edit User Dialog.

For system requirements, see ORDS System Requirements.

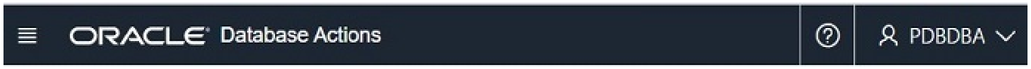
2.1 About the SQL Developer Web User Interface

The SQL Developer Web user interface has three components:

- The Header at the top
- The page body, whose content varies depending on which page you are viewing
- The Status Bar at the bottom

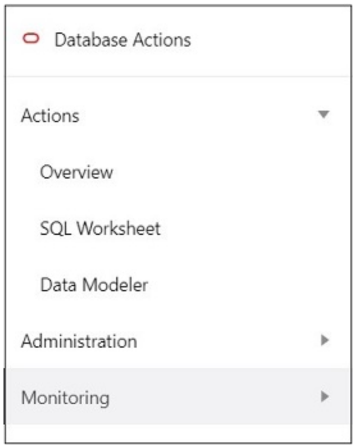
Header

The header contains the Selector icon, the help icon, and the user drop-down list.



- **Selector Icon**

  Click the Selector icon ≡ to see the main menu in a left navigation pane. Use the Selector to switch between the Database Actions page and the navigation pane.



- **Help Icon**

  Click the help icon to open the contextual or online help for the page you are viewing.

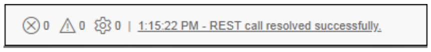
- **User Drop-Down List**

  The user drop-down list shows the database user you are signed in as, and provides the following items when you open it:

  - **Preferences**: Opens a dialog that lets you set your preferred time zone and user interface language. The languages available are English, German, Spanish, French, Italian, Japanese, Koren, Portuguese, and Chinese.
  - **Log**: Opens a dialog that shows the list of HTTP calls made during your SQL Developer Web session.
  - **About**: Opens a dialog providing version information for the database and other components as well as copyright and licensing information.
  - **Sign Out**: Signs you out of your database session.

Status Bar

The status bar contains icons that link to log files. The three icons (Errors, Warnings, Processes) are filters that have been applied to the log file.



1:15:22 PM - REST call resolved successfully.

Errors, **Warnings**: Displays an Errors or Warnings dialog, which lists log entries from unsuccessful REST calls or from any other problem in the application.

Processes: Displays a Processes dialog, which logs REST calls that are either finished or ongoing.

Log notification link: Displays a Log dialog, containing log entries of the following types: Errors, Warnings, Processes, SQL History and SQL Result.

2.2 Accessing Oracle SQL Developer Web

Oracle SQL Developer Web is included with Oracle REST Data Services.

To access SQL Developer Web:

1. Enable SQL Developer Web in Oracle REST Data Services. For more information, see Advanced Installation Using Command-Line Prompts in *Oracle REST Data Services Installation, Configuration, and Development Guide*.
2. To use SQL Developer Web, you must sign in as a database user whose schema has been REST-enabled for SQL Developer Web.

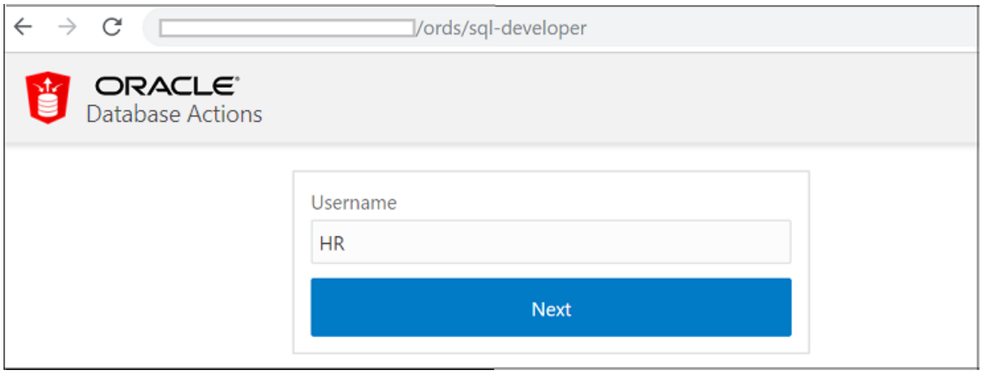
   Execute the following code as a database user with the DBA role:

   ```
   BEGIN
    ords_admin.enable_schema(
     p_enabled => TRUE,
     p_schema => 'schema-name',
     p_url_mapping_type => 'BASE_PATH',
     p_url_mapping_pattern => 'schema-alias',
     p_auto_rest_auth => TRUE
    );
    commit;
   END;
   ```

   where:

   - *schema-name* is the database schema name in all-uppercase.
   - *schema-alias* is an alias for the schema name that will appear in the URL the user will use to access SQL Developer Web. Oracle recommends that you do not use the schema name itself as a security measure to keep the schema name from being exposed.
   - *p_auto_rest_auth* specifies that the REST /metadata-catalog/ endpoint requires authorization. REST uses the metadata-catalog to get a list of published services on the schema.
3. In the login page (`http://`*`domain name`*`/ords/sql-developer`), enter the name of the user and click **Next**.

Figure 2-1 Login Page



Username: HR

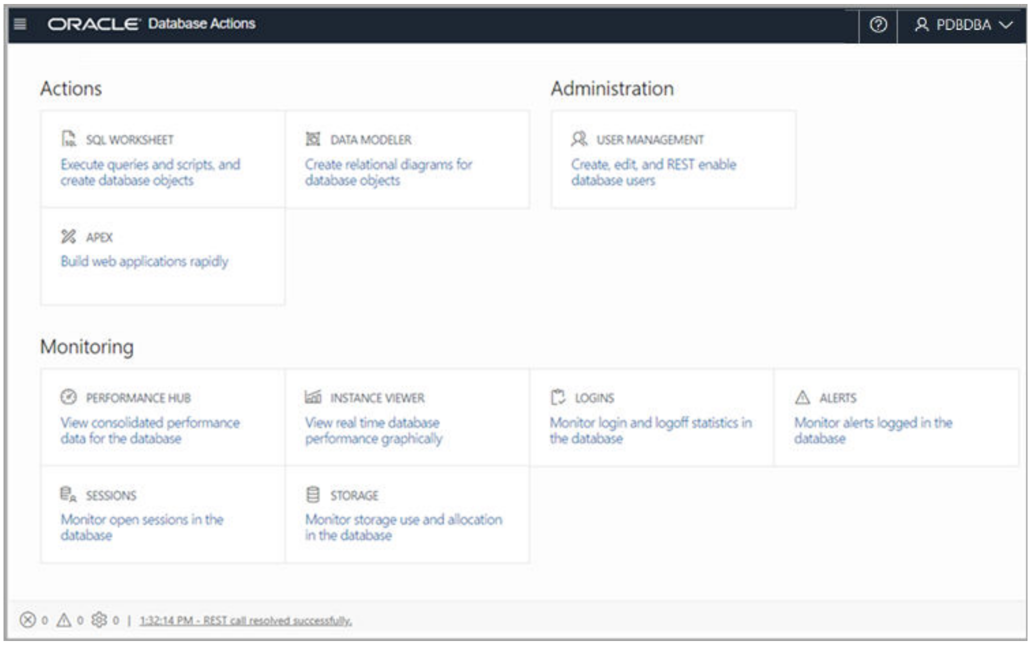
You see the URL change to `/ords/<`*`Username`*`>`. Enter the password, and click **Sign in**. The Oracle Database Actions page appears.

When you log out, you return to the login page (`http://`*`domain name`*`/ords/sql-developer`) and are prompted for the user name again.

3
The Database Actions Page

When you log in to Oracle SQL Developer Web, the Database Actions page appears. This page shows the main features that you can navigate to in the application.

Figure 3-1 The Database Actions Page



Actions

SQL WORKSHEET
Execute queries and scripts, and create database objects

DATA MODELER
Create relational diagrams for database objects

APEX
Build web applications rapidly

Administration

USER MANAGEMENT
Create, edit, and REST enable database users

Monitoring

PERFORMANCE HUB
View consolidated performance data for the database

INSTANCE VIEWER
View real time database performance graphically

LOGINS
Monitor login and logoff statistics in the database

ALERTS
Monitor alerts logged in the database

SESSIONS
Monitor open sessions in the database

STORAGE
Monitor storage use and allocation in the database

0 0 0 | 1:32:14 PM - REST call resolved successfully.

The Database Actions page contains three main groups: Actions, Administration, and Monitoring. Each group consists of feature-based icons that when clicked go to the respective page in the application. The Administration and Monitoring groups are available only to administrator user roles.

The main features are:

- **SQL Worksheet**: Enter and execute SQL and PL/SQL commands, and create database objects. See The SQL Worksheet Page
- **Data Modeler**: Create diagrams from existing database schemas, generate DDL statements, and create reports. See The Data Modeler Page
- **APEX**: Link to the Oracle Application Express sign-in page. Application Express is a rapid web application development platform for the Oracle database. See Oracle Application Express
- **User Management**: Perform user management tasks such as create, edit, and REST enable users. User Management is available only if you are signed in as a database user with DBA and PDB_DBA roles. See The User Management Page

- **Monitoring**: Monitor database activity and performance using various tools. Monitoring is available only if you are signed in as a database user with DBA and PDB_DBA roles. See The Monitoring Pages

4
The Actions Overview Page

The Overview page contains widgets that provide a general overview of the activity in Worksheet and Data Modeler.

To reach this page, click Selector ≡ , click the down arrow to expand **Actions** in the navigation pane, and then click **Overview**.

- **My Worksheets**: Displays your saved worksheets. You can click the name of the worksheet to open it in the Worksheet page.
- **My Diagrams**: Displays the Data Modeler diagrams that have been saved. You can click the name of the diagram to open it in the Data Modeler page.
- **Recently Modified Objects**: Displays a timeline of the created, modified, and dropped objects in the database. You can zoom in and out using the + and – icons. You can also move horizontally by dragging the cursor to the right or left.
- **Invalid Objects**: Displays the invalid objects in your schema.
- **Table Stats Freshness**: Displays the time period since the tables were last analyzed.

You can right-click the header in Invalid Objects, Table Stats Freshness, My Worksheets, or My Diagrams to manage or sort columns:

- **Columns**: Enables you to select columns to show or hide.
- **Sort**: Displays a dialog box for selecting columns to sort by. For each column, you can specify ascending or descending order, and you can specify that null values be displayed first.

Right-click the body of the display table to count rows or to view records:

- **Count Rows**: Displays the number of rows in the table.
- **Single Record View**: Enables you to view data for a table or view, one record at a time.
- **Copy**: Enables you to copy data from a cell or a row or a range of rows. To copy from more than one row, select the rows you want to copy, right-click by pressing the SHIFT or CTRL key, and select **Copy**.

5
The SQL Worksheet Page

The SQL Worksheet is a code editor that enables you to enter and execute SQL and PL/SQL statements, and create database objects.

To reach the SQL Worksheet page, click **SQL Worksheet** in the Database Actions page, or click Selector ≡ and select **SQL Worksheet** from the Actions menu in the navigation pane.

You can use SQL and PL/SQL statements in the worksheet to create a table, insert data, create and edit a trigger, select data from a table, and save that data to a file. Some other features are syntax highlighting and error detection.

The SQL Worksheet page consists of the left pane for navigating worksheets and objects, the editor for executing SQL statements, and the output pane for viewing the results. These panes are described in the following sections:

- Navigating Objects and Worksheets
- Executing SQL Statements in the Worksheet Editor
  - Keyboard Shortcuts in the SQL Worksheet
- Viewing the Worksheet Output
- Loading Data Using the SQL Worksheet

5.1 Navigating Objects and Worksheets

The Navigator tab in the left pane displays the saved objects for the selected schema.

The drop-down menus enable you to select the schema for which you want to see the objects, and filter the results by object type.

The Worksheets tab displays the worksheets that are saved in the browser. You can open or delete an existing worksheet by using the context (right-click) menu.

> **Note:**
>
> The worksheets created are dependent on the browser used (Internet Explorer, Mozilla Firefox, or Google Chrome) for the session at the time of saving the worksheet, and they will not be accessible from any other browser.

You can also search the contents of a saved worksheet or search for objects in the Navigator tab by name. The search functionality is not case-sensitive, retrieves all matching entries, and does not require the use of wildcard characters.

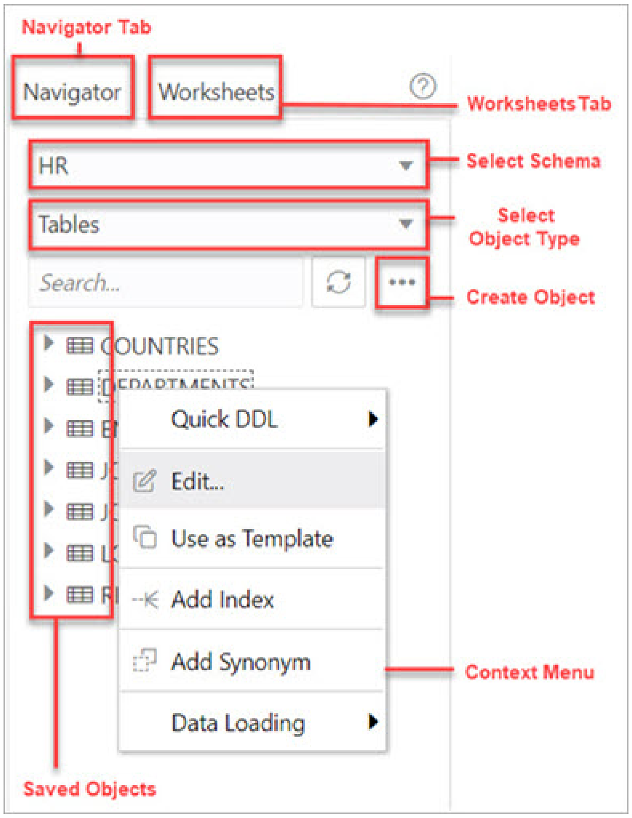
Using the context (right-click) menu for objects in the Navigator tab, you can create and edit objects, add indexes and synonyms, and upload data from local files into existing tables.

To refresh the objects or worksheets list, click Refresh ⟳ . To create a new object, click Object submenu ⊞ .

For context-related help, click Help ⓘ .

Figure 5-1 shows the various elements in the worksheet left pane.

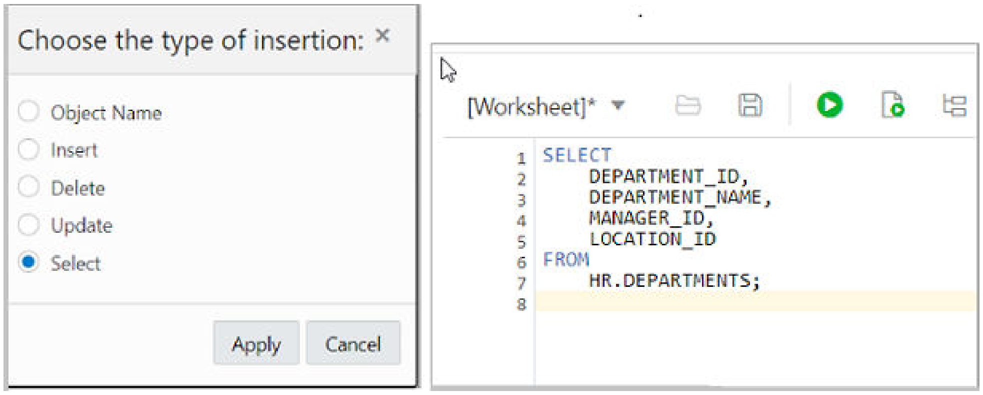
Figure 5-1   Left Pane in the SQL Worksheet



You can drag objects from the Navigator and drop them into the worksheet editor in the right pane:

- If you drag and drop a table or view, you are prompted to select one of the following SQL statements: Insert, Update, Select, or Delete. For example, if you choose Select, a Select statement is constructed with all columns in the table or view. You can then edit the statement, for example, modifying the column list or adding a WHERE clause.

Figure 5-2   Insert Select Query



Choose the type of insertion:

○ Object Name
○ Insert
○ Delete
○ Update
● Select

```
SELECT
    DEPARTMENT_ID,
    DEPARTMENT_NAME,
    MANAGER_ID,
    LOCATION_ID
FROM
    HR.DEPARTMENTS;
```

If you choose Object Name, the name of the object prefixed by the schema name is added to the worksheet.

- If you drag and drop a function or procedure, you can choose to insert the name or the PL/SQL code of the function or procedure in the worksheet. If you select the PL/SQL code, you can enter the parameters before inserting the code into the worksheet.

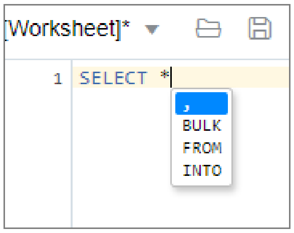
5.2 Executing SQL Statements in the Worksheet Editor

The SQL worksheet editor enables you to enter the SQL statements that you intend to execute.

You can use SQL and PL/SQL statements to specify actions such as creating a table, inserting data, selecting data or deleting data from a table. For multiple statements, each non-PL/SQL statement must be terminated with either a semicolon or (on a new line) a slash (/), and each PL/SQL statement must be terminated with a slash (/) on a new line. SQL keywords are automatically highlighted.

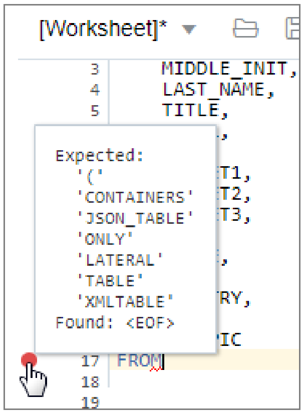
For SQL*Plus and SQLcl statements supported in the worksheet, see Supported SQL*Plus and SQLcl Commands.

If you press **Ctrl+Space**, the worksheet provides you with a list of possible completions at the insertion point that you can use to autocomplete code that you are editing. This list is based on the code context at the insertion point. Also, you can select multiple options in the list using **Ctrl+Click**.



```
SELECT *
```

An error in the code is signified by a red dot or squiggle line. When you hover over it, you see a pop-up displaying possible fixes for resolving the error.



```
    MIDDLE_INIT,
    LAST_NAME,
    TITLE,
```

```
Expected:
   '('
   'CONTAINERS'
   'JSON_TABLE'
   'ONLY'
   'LATERAL'
   'TABLE'
   'XMLTABLE'
Found: <EOF>
```

```
FROM
```

The SQL Worksheet toolbar contains icons for the following operations:

- **Worksheet** enables you to create a worksheet, open or delete an existing worksheet, and save a worksheet.

  > **Note:**
  >
  > The worksheets are saved in the browser. Therefore, if you do sensitive work on a computer in a public network, remember to clear the browser cache before you leave. Additionally, the saved worksheets will only be displayed in the browser that was used at the time of creating the worksheet.

- **Run Statement** executes the selected statements or the statement at the mouse pointer in the worksheet editor. The SQL statements can include bind variables and substitution variables of type VARCHAR2 (although in most cases, VARCHAR2 is automatically converted internally to NUMBER if necessary). A dialog box is displayed for entering variable values.
- **Run Script** executes all statements in the worksheet editor using the Script Runner. The SQL statements can include bind variables (but not substitution variables) of type VARCHAR2 (although in most cases, VARCHAR2 is automatically converted internally to NUMBER if necessary). A dialog box is displayed for entering bind variable values.
- **Explain Plan** generates the execution plan for the statement (internally executing the EXPLAIN PLAN statement). The execution plan is automatically displayed in the Explain Plan tab in the worksheet output pane.
- **Autotrace** runs the statement and collects runtime statistics and the actual execution plan. The Autotrace output is displayed in the Autotrace tab in the worksheet output pane. Currently, there are no preferences available.
- **Download Editor Content** downloads the content of the worksheet as a SQL file to the local system.
- **Format** formats the SQL statement in the editor, such as capitalizing the names of statements, clauses, keywords, and adding line breaks and indentation.
- **Clear** removes the statements from the editor.
- **Tour** provides a guided tour of the worksheet highlighting salient features and providing information that is useful if you are new to the interface.
- **Help** provides context-related help and provides a link to the help documentation.

For a list of shortcuts used in the worksheet, see Keyboard Shortcuts in the SQL Worksheet.

5.2.1 Keyboard Shortcuts in the SQL Worksheet

The following table lists the keyboard shortcuts for various user actions in the SQL worksheet.

Table 5-1 Keyboard Shortcuts for User Actions in the Worksheet

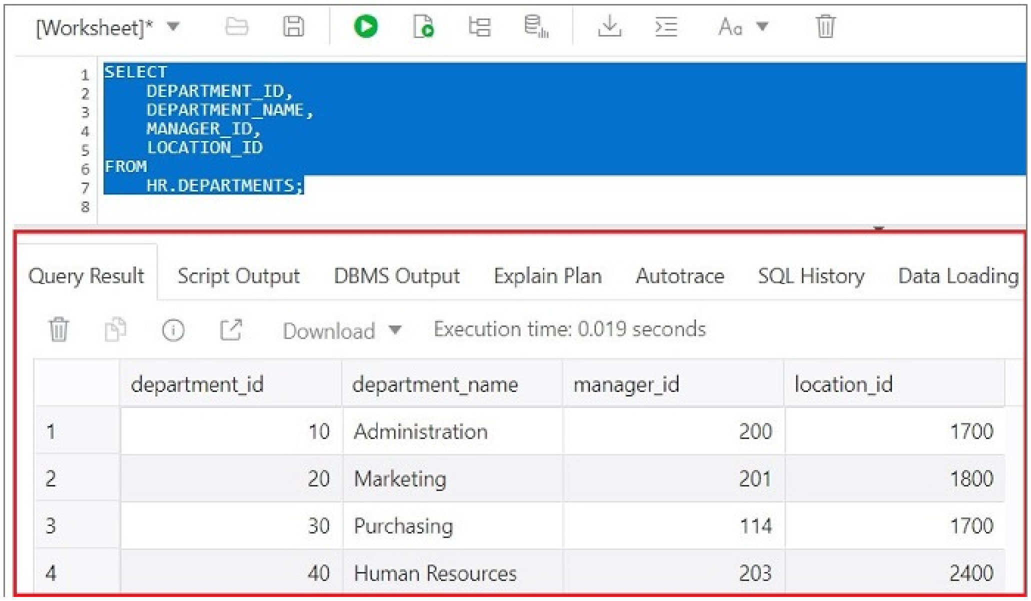
Keyboard Shortcut	Action
Ctrl+Enter/ Cmd+Enter	Runs the code as query.
Ctrl+Down Arrow/ Cmd+Down Arrow	Moves to the next SQL code from history.
Ctrl+Up Arrow/ Cmd+Up Arrow	Moves to the previous SQL code from history.
Ctrl+D/ Cmd+D	Clears the editor.
Ctrl+S/ Cmd+S	Saves the current worksheet.
Ctrl+O/ Cmd+O	Opens the worksheet browser dialog.
Ctrl+I/ Cmd+I	Downloads the content of the editor.
F1	Opens the help topic.
F5	Runs code as script.
F6	Shows Autotrace.
F10	Shows Explain Plan.
Ctrl+F7/ Cmd+F7	Formats code in the editor.
Ctrl+Space/ Cmd+Space	Autocompletes code (shows hints).
Windows+Esc/ Cmd+Esc	Focuses outside the editor and navigates to the rest of the application using the Tab key.

5.3 Viewing the Worksheet Output

The lower right pane in the SQL worksheet displays the output of the operation executed in the worksheet editor.

Figure 5-3 shows the output pane in the worksheet.

Figure 5-3 Output Pane in SQL Worksheet



```
SELECT
    DEPARTMENT_ID,
    DEPARTMENT_NAME,
    MANAGER_ID,
    LOCATION_ID
FROM
    HR.DEPARTMENTS;
```

	department_id	department_name	manager_id	location_id
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	30	Purchasing	114	1700
4	40	Human Resources	203	2400

The output pane has the following tabs:

- **Query Result**: Displays the results of the most recent Run Statement operation in a display table.
- **Script Output**: Displays the text output from your statements executed as a script using the script engine.
- **DBMS Output**: Displays the output of DBMS_OUTPUT package statements.
- **Explain Plan**: Displays the plan for your query using the Explain Plan command.
- **Autotrace**: Displays the session statistics and execution plan from `v$sql_plan` when executing a SQL statement using the Autotrace feature. Displays the output if you clicked the Autotrace icon.
- **SQL History**: Displays the SQL statements and scripts that you have executed. To re-enter a previously executed query in the worksheet, double-click the query in the history list. You can search for specific statements by clicking the Search icon. The Search functionality is case-sensitive, retrieves all entries that contain the search text, and does not require wildcard characters.
- **Data Loading**: Displays a report of the total rows loaded and failed for all visible tables (including tables from other schemas).

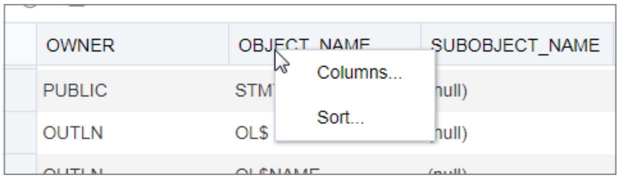
The icons in this pane are:

- **Clear output**: Clears the output.
- **Show info**: Displays the SQL statement for which the output is displayed.
- **Open in new tab**: Opens the query result or explain plan in a new window.
- **Download**: This is applicable only for Query Result. Enables you to download the query result to your local computer in CSV, JSON, XML, or TEXT (.tsv) format.

In the Query Result tab, in the display table, the context menu (right-click) for the row header consists of the following:

- **Columns** enables you to select columns to hide.
- **Sort** displays a dialog box for selecting columns to sort by. For each column, you can specify ascending or descending order, and you can specify that null values be displayed first.

Figure 5-4 Context Menu for Row Header



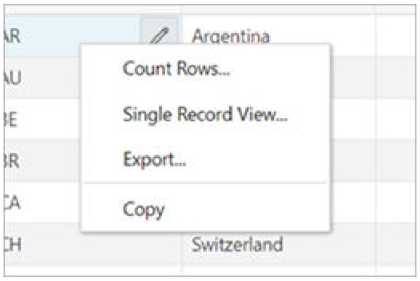
The context menu for the rest of the display table consists of the following commands:

- **Count Rows** displays the number of rows in the result set for your query.
- **Single Record View** enables you to view data for a table or view, one record at a time.
- **Export** generates the file for download based on the format selected, which can be XML, CSV (comma-separated values including a header row for column identifiers), Insert , Delimited, Fixed, HTML, JSON, or TEXT.
  - **Format**: Select the format to export from the drop-down list.
  - **Line Terminator**: Identifies the terminator for each line. The line terminator is not included in the data exported. If the preview page shows the data in one single row, the correct terminator is not specified.
  - **Header**: Controls whether the first row is a header row or the first row of data.
  - **Left and Right Enclosure**: Enclosures are used for character data and are optional. Enclosures are not included in the data exported.

> **Note:**
>
> If a popup blocker is enabled, it will prevent the file from downloading.

- **Copy** copies data from a cell or a row or a range of rows.

Figure 5-5 Context Menu



5.4 Loading Data Using the SQL Worksheet

In the SQL Worksheet page, you can load data from a local file into an existing table or into a new table.

The file formats that you can load are CSV, XLS, XLSX, TSV, TXT, XML, JSON, and AVRO. For XML, JSON, and AVRO files, see Format Specifications for JSON, AVRO, and XML Files.

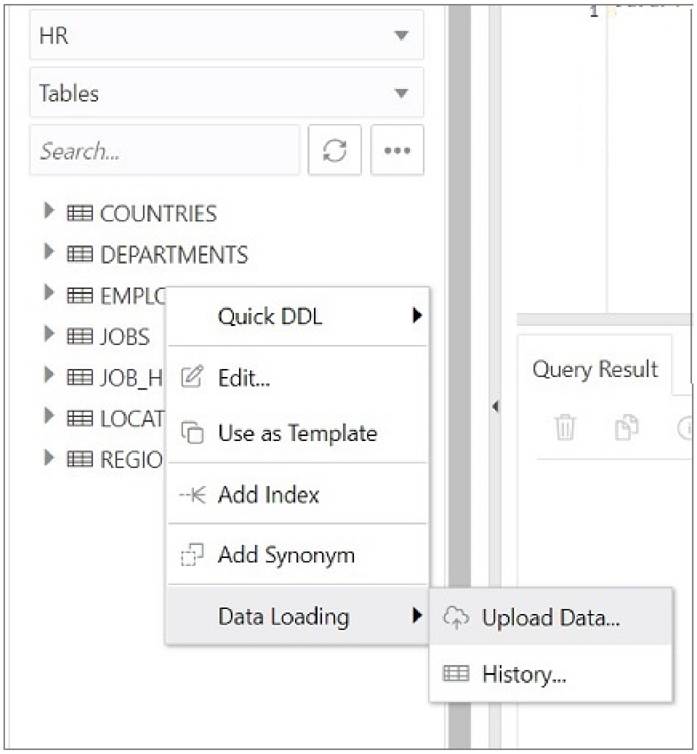
- Loading Data from a Local File to an Existing Table
- Loading Data from a Local File to a New Table

5.4.1 Loading Data from a Local File to an Existing Table

To load data from a local file to an existing table:

1. In the Navigator tab, in the left pane, right-click the table that you want to load data into, select **Data Loading**, and then select **Upload Data**.

Figure 5-6 Upload Data Option for an Existing Table



The Import data dialog is displayed.

2. Drag and drop the file from your system into the dialog, or click **Select Files** to browse for the file and open it.

   A preview of the data is displayed in a grid format.

3. Click Show/Hide options ⚙ ▾ to display options that you can modify for data preview:

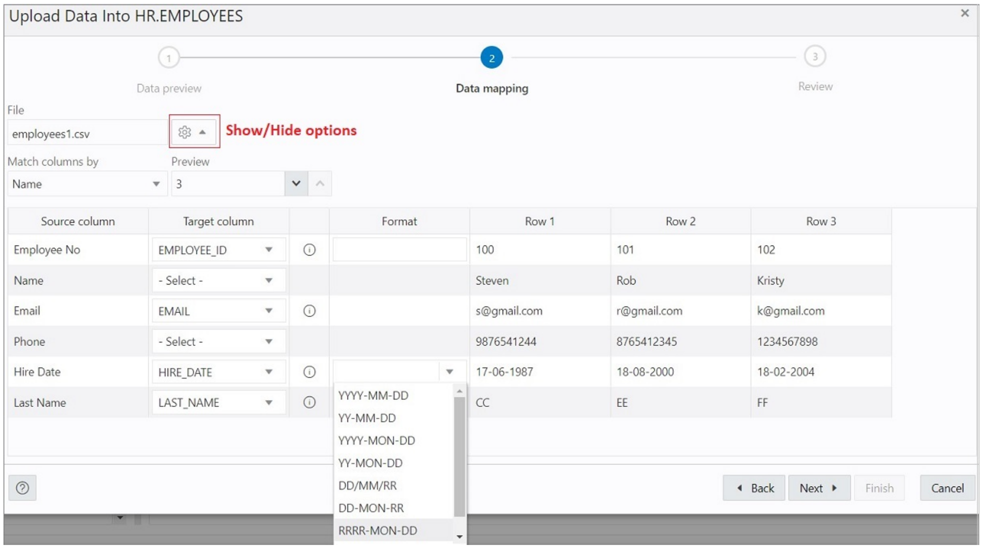
   - **Column names**: Select **Get from file** to display column headers in the first row.
   - **Text enclosure** and **Field delimiter**: These options are visible only when the selected file is in a plain text format (CSV, TSV, or TXT). Select or enter the character used in the source file for text enclosure and field delimiter.
   - **Rows to skip**: Enter or use the up and down arrows to select the number of rows to skip.
   - **Rows to load**: Enter or use the up and down arrows to select the number of rows to load.
   - **Preview size**: Enter or use the up and down arrows to select the number of rows to preview.

   To remove the options selected and the data preview, click **Clear**.

   After selecting the required options, click **Apply**, and then click **Next**.

4. In Data mapping, match the data in the file to the appropriate columns in the target table. By default, the matching is done using column name.

Figure 5-7 Data Mapping


Upload Data Into HR.EMPLOYEES

File: employees1.csv

Match columns by: Name

Preview: 3

Source column	Target column		Format	Row 1	Row 2	Row 3
Employee No	EMPLOYEE_ID	ⓘ		100	101	102
Name	- Select -			Steven	Rob	Kristy
Email	EMAIL	ⓘ		s@gmail.com	r@gmail.com	k@gmail.com
Phone	- Select -			9876541244	8765412345	1234567898
Hire Date	HIRE_DATE	ⓘ		17-06-1987	18-08-2000	18-02-2004
Last Name	LAST_NAME	ⓘ		CC	EE	FF

To modify, click Show/Hide ⚙ ▾. In Match columns by:

- Select **Name** to match columns based on the name of the column in the target table.
- Select **Position** if you want to match columns based on the position of the column in the target table.
- Select **None** to remove the current selections and to select the target column for each source column from the drop-down list.

> **Note:**
>
> Based on the data in the file, SQL Developer Web attempts to retrieve the correct format mask of date-based columns. You can also change the suggested format by entering it directly into the target cell.

If there are any issues to resolve, you see a notification such as ⓘ 1 pending actions on the top right of the dialog.

Click **Next**.

5. A summary of the previous screens is displayed. Click **Finish**.

   The data will start uploading to the target table. After it is completed, an entry is added to the Log with the status of the operation. To view the Log, click the timestamp notification at the bottom of the page. If the operation is successful, a `Data Import Completed` notification is displayed.

6. For a detailed summary of the upload process, right-click the table in the Navigator tab, select **Data Loading**, and then select **History**. A summary of the data loaded is displayed in the Data Loading History dialog.

   If any data failed to load, you can view the number of rows in the Failed Rows column. Click the column to open a dialog showing the failed rows.

   In the Data Loading History dialog, you can also search for files loaded by schema name, table name, or file name. To remove the loaded files, click Remove all history 🗑.

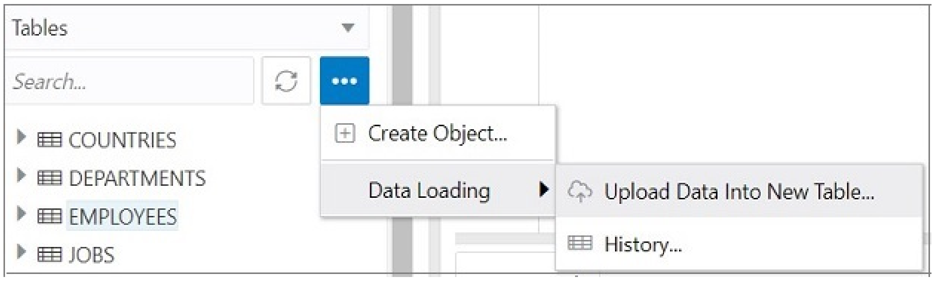
5.4.2 Loading Data from a Local File to a New Table

To load data from a local file to a new table:

1. You can start in one of the following ways:

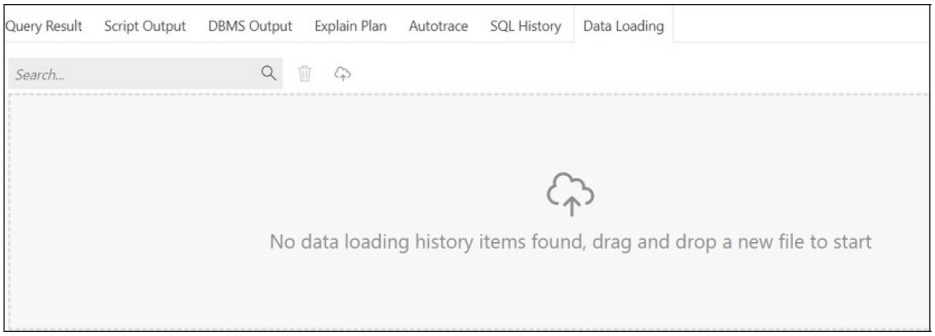
   - In the Navigator tab, in the left pane, click Object submenu ••• , select **Data Loading**, and then select **Upload Data into New Table**.

     Figure 5-8 Upload Data to New Table Option

   - In the Navigator tab, drag and drop the local file into the left pane. When you drag a file into the pane, the following message is displayed `Drop the file here to start`.
   - In the worksheet output pane, select the **Data Loading** tab and drag and drop the local file into the output pane.

Figure 5-9 Data Loading Tab



No data loading history items found, drag and drop a new file to start

The Upload Data into New Table is displayed. A preview of the data is displayed in a grid format.

2. Click ⚙ ▾ to display options that you can modify for data preview:
   - **Column names**: Select **Get from file** to display column headers in the first row.
   - **Text enclosure** and **Field delimiter**: This option is visible only when the selected file is in a plain text format (CSV, TSV, or TXT). Select or enter the character used in the source file for text enclosure and field delimiter.
   - **Rows to skip**: Enter or use the up and down arrows to select the number of rows to skip.
   - **Preview size**: Enter or use the up and down arrows to select the number of rows to preview.
   - **Limit rows to upload**: If you select this option, you need to specify the rows to load. Use the up and down arrows to select the number of rows to load.

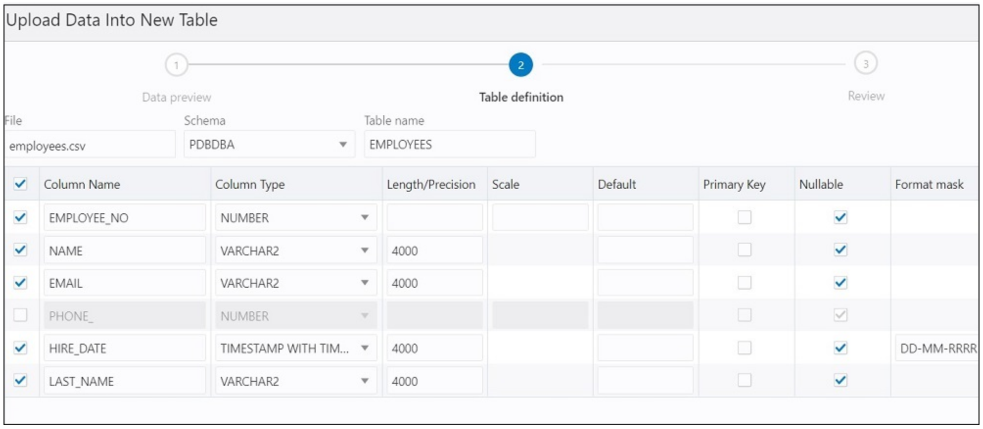
   To remove the options selected and the data preview, click **Clear**.

   After selecting the required options, click **Apply**, and then click **Next**.

3. In Table Definition, do the following:
   - In the **Table Name** field, enter a name for the target table.
   - Select the check box at the beginning of a row to add the column to the target table.
   - Select or enter the column attributes such as Column Name, Column Type, Precision, Scale, Default, Primary Key and Nullable.
   - The Format Mask column appears for date, timestamp and numeric types of data. For date and timestamp types, you must select a value from the drop-down list or type the value in the Format Mask field. For numeric types, the format mask is optional.

     For a date and timestamp column, you need to supply a compatible format mask that describes the data being uploaded. SQL Developer Web will try to determine this automatically, but you need to review and modify it appropriately. For example, if the date data looks like `12-FEB-2021 12.21.30`, you need to supply a date mask of `DD-MON-YYYY HH.MI.SS`.

Figure 5-10 Table Definition Step in Upload Data into New Table



File: employees.csv

Schema: PDBDBA

Table name: EMPLOYEES

☑	Column Name	Column Type	Length/Precision	Scale	Default	Primary Key	Nullable	Format mask
☑	EMPLOYEE_NO	NUMBER				□	☑	
☑	NAME	VARCHAR2	4000			□	☑	
☑	EMAIL	VARCHAR2	4000			□	☑	
□	PHONE_	NUMBER				□	☑	
☑	HIRE_DATE	TIMESTAMP WITH TIM...	4000			□	☑	DD-MM-RRRR
☑	LAST_NAME	VARCHAR2	4000			□	☑	

Click **Next**.

4. Review the generated DDL code based on the selections made in the previous screens. The mapping of the source to target columns are also displayed.

   Click **Finish**. After the data is successfully loaded, the new table is displayed in the Navigator pane.

5. For a detailed report of the total rows loaded and failed, do one of the following:

   - Right-click the table in the Navigator tab, select **Data Loading**, and then select **History**. This displays the report for a specific table.
   - In the Navigator pane, select Object submenu ••• , select **Data Loading**, and then select **History**. This displays the report for all tables in the schema that is currently selected in the object Navigator.
   - In the worksheet output pane, select the **Data Loading** tab. This displays the report for all visible tables (including tables from other schemas).

   A summary of the data loaded is displayed in the History dialog. If any data failed to load, you can view the number of rows in the Failed Rows column. Click the failed rows column to open a dialog showing the failed rows. In the History dialog, you can also search for files loaded by schema name, table name, or file name. To remove the loaded files, click Remove all history 🗑.

5.4.3 Format Specifications for JSON, AVRO, and XML Files

Data has to be stored in a particular format for JSON, AVRO, and XML files to load them successfully into a table.

The format specifications are described in the following sections:

- JSON and AVRO Files
- XML Files

5.4.3.1 JSON and AVRO Files

For JSON and AVRO files, the conversion for primitive types to table columns is supported only for top-level data. Nested objects are saved as JSON strings such as VARCHAR2 (JSON) or CLOB (JSON).

> **Note:**
>
> JSON check constraints are available only for Oracle Database 12c and later releases.

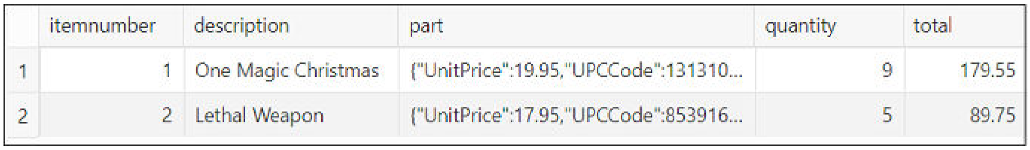
Consider the following JSON file:

```
[
  {
    "ItemNumber": 1,
    "Description": "One Magic Christmas",
    "Part": {
      "UnitPrice": 19.95,
      "UPCCode": 13131092899
    },
    "Quantity": 9,
    "Total": 179.55
  },
  {
    "ItemNumber": 2,
    "Description": "Lethal Weapon",
    "Part": {
      "UnitPrice": 17.95,
      "UPCCode": 85391628927
    },
    "Quantity": 5,
    "Total": 89.75
  }
]
```

The AVRO schema for this file:

```
{
     "type": "array",
     "items": {
          "type": "record",
          "fields": [
               {
                    "name": "ItemNumber",
                    "type": "int"
               },
               {
                    "name": "Description",
                    "type": "string"
               },
               {
                    "name": "Part",
                    "type": {
                         "type": "record",
                         "fields": [
                              {
                                   "name": "UnitPrice",
                                   "type": "float"
                              },
                              {
                                   "name": "UPCCode",
                                   "type": "float"
                              }
                         ]
                    }
               },
               {
                    "name": "Quantity",
                    "type": "int"
               },
               {
                    "name": "Total",
                    "type": "float"
               }
          ]
     }
}
```

Load the example file into SQL Developer Web, and it is converted to the following table with two rows.



	itemnumber	description	part	quantity	total
1	1	One Magic Christmas	{"UnitPrice":19.95,"UPCCode":131310...	9	179.55
2	2	Lethal Weapon	{"UnitPrice":17.95,"UPCCode":853916...	5	89.75

In this example, "part" is a nested object that is assigned the column type CLOB (JSON) during data mapping.

5.4.3.2 XML Files

This section lists the specifications for loading XML files.

- Attributes will have their own columns

  If the XML data is structured as:

  ```
  <?xml version="1.0"?>
  <catalog>
    <book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <publisher>John Doe</publisher>
  ```

```
   </book>
</catalog>
```

The generated columns in the table are `id`, `author`, `title`, `genre`, and `publisher`.

- Two or more levels of nesting are needed to parse the data

  In the following example, the data that needs to be parsed will not be located because it has only one level of nesting (catalog).

```
<?xml version="1.0"?>
<catalog>
  <author>Ralls, Kim</author>
  <title>Midnight Rain</title>
  <genre>Fantasy</genre>
  <publisher>John Doe</publisher>
</catalog>
```

  However, the following examples will work:

```
<?xml version="1.0"?>
<catalog>
  <book id="bk102">
    <author>Ralls, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <publisher>John Doe</publisher>
  </book>
</catalog>
```

  or

```
<?xml version="1.0"?>
<catalog>
  <bookstore>
    <book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <publisher>John Doe</publisher>
    </book>
  </bookstore>
</catalog>
```

  or

```
<?xml version="1.0"?>
<catalog>
  <bookstore>
    <shelf>
      <book id="bk102">
        <author>Ralls, Kim</author>
        <title>Midnight Rain</title>
        <genre>Fantasy</genre>
        <publisher>John Doe</publisher>
      </book>
    </shelf>
  </bookstore>
</catalog>
```

- Special characters such as hyphen (-) and period (.) in tag names are replaced by underscore (_) in the column name

  XML tag names can contain hyphens and periods. Since the parser is converting XML to JSON, these characters become invalid object keys.

  ```
  <?xml version="1.0"?>
  <catalog>
    <book id="bk102">
      <author-name>Ralls, Kim</author-name>
      <title.1>Midnight Rain</title.1>
      <genre>Fantasy</genre>
      <publisher>John Doe</publisher>
     </book>
  </catalog>
  ```

  The generated columns are `id`, `author_name`, `title_1`, `genre`, and `publisher`.

- First-level only-text tags are ignored

  ```
  <?xml version="1.0"?>
  <catalog>
    <library> New Age Library </library>
    <book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <publisher>John Doe</publisher>
    </book>
  </catalog>
  ```

  The `<library>` tag is ignored and only the content of the `<book>` tag is taken into account. The generated columns are `id`, `author`, `title`, `genre`, and `publisher`.

- First-level repetitive data is interpreted as an array of values

  ```
  <?xml version="1.0" encoding="UTF-8"?>
  <items id="orders">
      <item_number>1</item_number>
      <description>One Magic Christmas</description>
      <part>
        <unit_price>19.95</unit_price>
        <upccode>13131092899</upccode>
      </part>
      <quantity>9</quantity>
      <total>179.55</total>
      <item_number>2</item_number>
      <description>Lethal Weapon</description>
  ```

```
    <part>
      <unit_price>17.95</unit_price>
      <upccode>85391628927</upccode>
    </part>
    <quantity>5</quantity>
    <total>89.75</total>
</items>
```

The generated columns include item_number, description, part,
and each column will have only one row with the
following values respectively ([1,2], ["One Magic Christmas","Lethal Weapon"], [{" unit_price":19.95,"upccode":13131092899}, {"unit_price":17.95,"upccode":85391628927}] and so on for the remaining columns.

- Tags containing values and attributes are converted to object

```
<?xml version="1.0"?>
<catalog>
  <book id="bk102">
    <author country="ca">Ralls, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <publisher>John Doe</publisher>
  </book>
</catalog>
```

The <author> tag is converted to a column and will have an object as value that is structured as:

```
{
      "_":"Ralls, Kim",
      "country":"ca"
    }
```

Note that the value of the tag has underscore ("_") as key and the attributes as "attribute_name": "attribute_value".

6

The Data Modeler Page

The Data Modeler page provides an integrated version of Oracle SQL Developer Data Modeler with basic reporting features. You can create diagrams from existing schemas, retrieve data dictionary information, generate DDL statements, and export diagrams.

To reach the Data Modeler page, click **Data Modeler** in the Database Actions page, or click Selector ☰ and select **Data Modeler** from the Actions menu in the navigation pane.

> **Note:**
>
> You can create and edit objects for your schema in the Data Modeler interface.

The Data Modeler page consists of the left pane for navigating objects and diagrams, the editor pane for working with relational diagrams, and right pane for viewing the properties of the object selected in the diagram. These panes are described in the following sections:

- Navigating Diagrams and Objects
- About the Data Modeling Editor

Related Topics

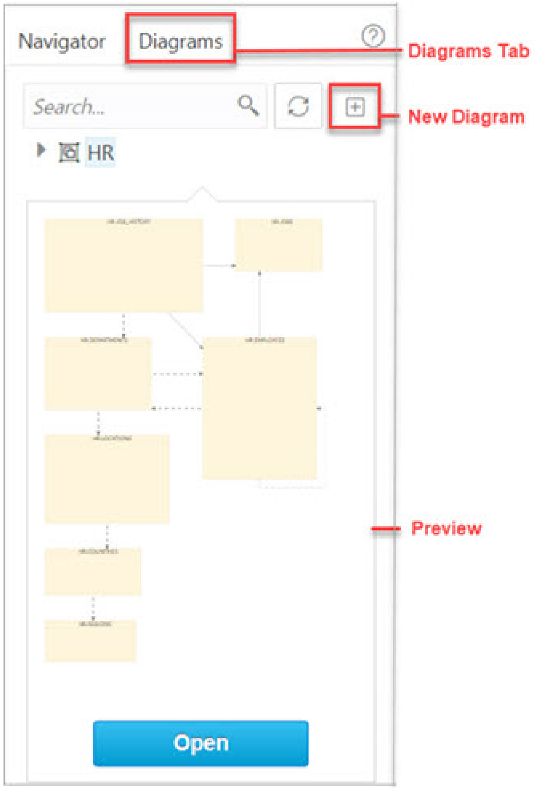
- *Oracle SQL Developer Data Modeler User's Guide*

6.1 Navigating Diagrams and Objects

The **Diagrams** tab lists the Data Modeler diagrams that have been saved.

When you right-click a diagram, you have options to open, save, delete, and view properties. Figure 6-1 shows a thumbnail preview that is displayed when you click the name of a diagram in the left pane.

Figure 6-1 Diagram Preview in Data Modeler



Diagrams Tab

New Diagram

Preview

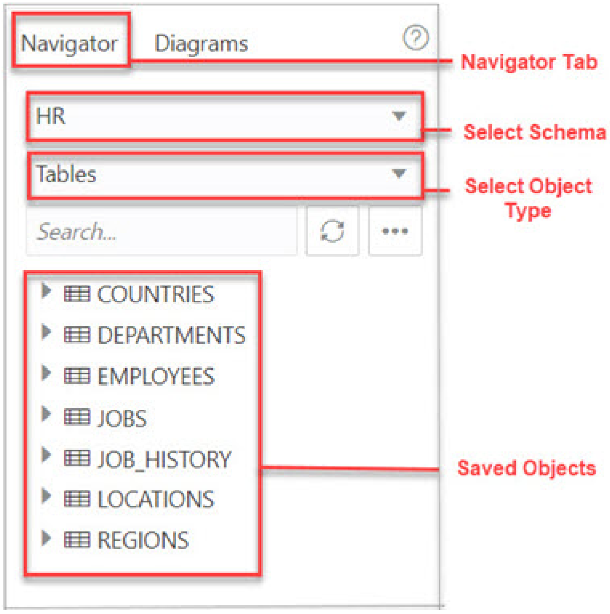
- To create a new diagram, in the Diagrams tab, click New Diagram ⊞.
- To open an existing diagram, in the Diagrams tab, right-click the diagram and click **Open**.

> **Note:**
>
> Diagrams are stored in the database, in the SQL Developer Web user schema. When you use Data Modeler for the first time, a `OSDDMW_DIAGRAMS` table is created in your schema, to store the diagrams.
>
> Oracle SQL Developer Data Modeler can import from or export to the `OSDDMW_DIAGRAMS` table if a connection is provided. For more information, see Sharing Diagrams with SQL Developer Data Modeler.

The **Navigator** tab lists the objects that are available for each schema. You can select the schema and object type from the drop-down lists. For a particular schema, if a tables object is selected, the left pane displays all the tables that belong to the particular schema. You can expand a table to view its columns. Figure 6-2 indicates the important elements in the Navigator tab.

In a schema, you can search for objects in the Navigator tab by entering a few characters. You can also search the contents of a saved diagram. The search functionality is not case-sensitive, retrieves all matching entries, and does not require the use of wildcard characters.

Figure 6-2 Navigator Tab



The context menu for a table or view consists of:

Add Object to Diagram: Adds the selected object to the selected diagram. Alternatively, you can drag and drop an object into a selected diagram.

Add Object with dependencies to Diagram: Adds the selected object along with parent and child tables related to the object to the selected diagram.

Add Object as Star Schema to Diagram: Adds the selected object to the diagram and searches the data dictionary for foreign keys and implied foreign keys related to the object. The related tables or views are added to the diagram and the star schema layout is applied. See Implied Foreign Keys

Edit, Use as Template: Opens the Table Properties Dialog. Use to edit an existing object (such as a table or view) for a specific schema, or create a new object by using an existing one for the initial content. See The Table Properties Dialog and The View Properties Dialog.

Add/Edit Sequence: Opens the Sequence Properties Dialog. Use to create or edit a sequence for a selected schema. See The Sequence Properties Dialog.

Add Index: Opens the Index Properties Dialog. Use to create an index for a table. See The Index Properties Dialog.

Add Synonym: Opens the Synonym Properties Dialog. Use to create a synonym for a table or view in a selected schema. See The Synonym Properties Dialog

6.2 About the Data Modeling Editor

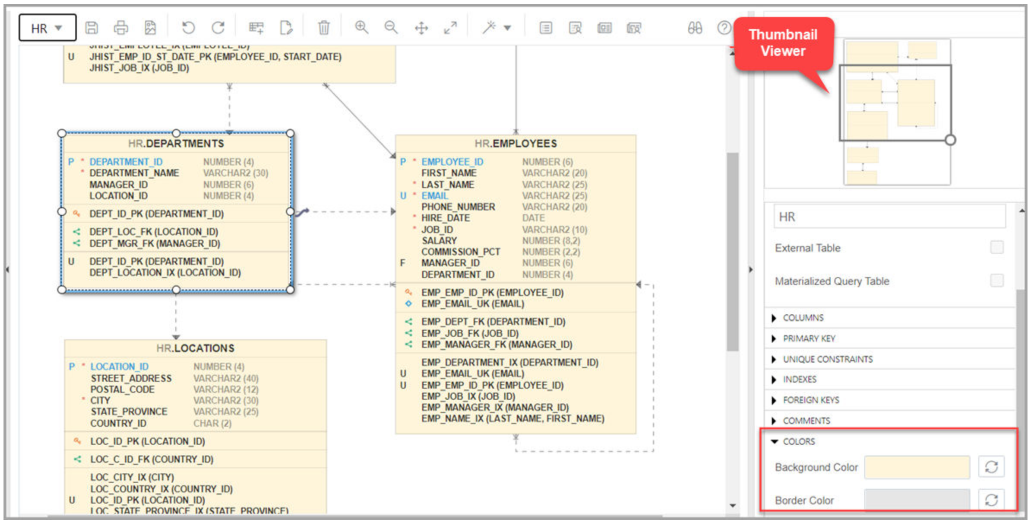
You can create and work with relational diagrams in the editor pane.

When an object is selected in the diagram, you can inspect the properties of the object in the right pane. For a table, the properties displayed are Columns, Primary Key, Unique Constraints, Indexes, Foreign Keys, Comments, and Colors. The only properties that you can edit are background and border color. Click Reset to return to the default colors.

Figure 6-3 shows the properties of an object selected in the editor pane.

You can drag the thumbnail viewer in the upper right corner to display the appropriate area in the editor pane. This is useful when you are working with long diagrams in the editor pane.

Figure 6-3 Inspect Properties of Object Selected in Editor Pane


HR
U JHIST_EMP_ID_ST_DATE_PK (EMPLOYEE_ID, START_DATE)
JHIST_JOB_IX (JOB_ID)
HR.DEPARTMENTS
P * DEPARTMENT_ID NUMBER (4)
* DEPARTMENT_NAME VARCHAR2 (30)
MANAGER_ID NUMBER (6)
LOCATION_ID NUMBER (4)
DEPT_ID_PK (DEPARTMENT_ID)
DEPT_LOC_FK (LOCATION_ID)
DEPT_MGR_FK (MANAGER_ID)
U DEPT_ID_PK (DEPARTMENT_ID)
DEPT_LOCATION_IX (LOCATION_ID)
HR.EMPLOYEES
P * EMPLOYEE_ID NUMBER (6)
FIRST_NAME VARCHAR2 (20)
* LAST_NAME VARCHAR2 (25)
U * EMAIL VARCHAR2 (25)
PHONE_NUMBER VARCHAR2 (20)
* HIRE_DATE DATE
* JOB_ID VARCHAR2 (10)
SALARY NUMBER (8,2)
COMMISSION_PCT NUMBER (2,2)
F MANAGER_ID NUMBER (6)
DEPARTMENT_ID NUMBER (4)
EMP_EMP_ID_PK (EMPLOYEE_ID)
EMP_EMAIL_UK (EMAIL)
EMP_DEPT_FK (DEPARTMENT_ID)
EMP_JOB_FK (JOB_ID)
EMP_MANAGER_FK (MANAGER_ID)
EMP_DEPARTMENT_IX (DEPARTMENT_ID)
U EMP_EMAIL_UK (EMAIL)
U EMP_EMP_ID_PK (EMPLOYEE_ID)
EMP_JOB_IX (JOB_ID)
EMP_MANAGER_IX (MANAGER_ID)
EMP_NAME_IX (LAST_NAME, FIRST_NAME)
HR.LOCATIONS
P * LOCATION_ID NUMBER (4)
STREET_ADDRESS VARCHAR2 (40)
POSTAL_CODE VARCHAR2 (12)
* CITY VARCHAR2 (30)
STATE_PROVINCE VARCHAR2 (25)
COUNTRY_ID CHAR (2)
LOC_ID_PK (LOCATION_ID)
LOC_C_ID_FK (COUNTRY_ID)
LOC_CITY_IX (CITY)
LOC_COUNTRY_IX (COUNTRY_ID)
U LOC_ID_PK (LOCATION_ID)
LOC_STATE_PROVINCE_IX (STATE_PROVINCE)
Thumbnail Viewer
COLUMNS
PRIMARY KEY
UNIQUE CONSTRAINTS
INDEXES
FOREIGN KEYS
COMMENTS
COLORS

HR

External Table ☐

Materialized Query Table ☐

Background Color

Border Color

When creating diagrams, you can do the following:

- Select an object and then drag to move it around.
- Adjust or move objects with the relationships intact.
- Add elbows to relationship lines to avoid intersecting with lines from other objects. Right-click the relationship line and drag to create the elbow. Click Remove vertex ☒ to restore the original shape.
- Resize objects by dragging the handles that are positioned around the box.
- Select and then right-click an object for the following options:
  - **DDL Preview**: Shows the DDL statements for the object.
  - **Update**: Updates any actions performed on the object.
  - **Delete**: Deletes the object.
  - **Edit**: Opens the Table Properties Dialog for editing a table object or the View Properties Dialog for a view object.
  - **Implied Foreign Keys**: Opens the Implied Foreign Keys Dialog. Use this option to define implied foreign keys for the object. See Implied Foreign Keys

The icons on the toolbar are:

- **Save Diagram**: Saves the currently selected diagram. Diagrams are stored in a table that is created in the schema of the user.
- **Print Diagram**: Prints the selected diagram.
- **Save to SVG**: Saves the currently selected diagram to an image file in SVG format.
- **Add Objects to Diagram**: A dialog is displayed where you can select one or more objects of type tables or views from a specific schema into the selected diagram. Type * to list all the objects of an object type (tables or views) in the schema. You can also search by typing a few characters.
- **Add Note**: Adds notes to the selected diagram. Select the note to add text and to see the associated properties in the right pane, such as Font Size and Colors.
  - To enter text, expand TEXT and type information. Click **Apply**.
  - To select the background, border, or text color for the note, click the box and select the required color. To revert to the default color, click Reset ↻ .
- **Delete**: Deletes the selected object or objects from the diagram. To select multiple objects, press the CTRL key and select the objects.
- **Zoom In**: Displays more detail, and potentially fewer objects, in the currently selected diagram.
- **Zoom Out**: Displays less detail, and potentially more objects, in the currently selected diagram.
- **Fit Screen**: Makes all relevant objects fit in the window for the currently selected diagram, adjusting the sizes of shapes and text labels as needed.
- **Actual Size**: Adjusts the shapes and text labels in the currently selected diagram to the default sizes.
- **Layout: Auto Layout**: Rearranges the objects in the diagram to a layout that may be more meaningful and attractive. If you do not like the rearrangement, you can restore the previous layout by clicking Undo.
- **Layout: Star Layout**: Rearranges the objects in the diagram to a star schema layout, where the fact table is in the center and the associated dimension tables surround the fact table.
- **DDL Preview**: Shows the DDL statements for the object. You have the option to save or send the DDL statements to the worksheet. To choose what you want to display, click **Options**.
- **DDL Preview for Current Schema**: Shows the DDL statements for the current schema. You can send the DDL statements to the worksheet.
- **Diagram Report**: Generates a data dictionary report of everything in the diagram.
- **Schema Report**: Generates a data dictionary report of everything in the schema.
- **Help**: Displays the help for the Data Modeling editor.

7

Creating and Editing Database Objects

You can create and edit objects using Create and Edit Object wizards available from the Navigator tab in SQL Worksheet and Data Modeler.

The wizards for various object types are described in the following sections:

- The Table Properties Dialog
- The Index Properties Dialog
- The Sequence Properties Dialog
- The View Properties Dialog
- The Synonym Properties Dialog
- Implied Foreign Keys
- The Materialized View Log Properties Dialog

If you do not have any existing objects for a object type (such as tables), you can create objects by using the Object submenu ••• icon, after first selecting the object type from the drop-down list. For all objects, the context menu includes **Quick DDL**, which saves the DDL statements for creating the object to a worksheet or to a text file. The various object types and the corresponding options are:

- **Tables**: Tables are used to hold data. Each table has multiple columns that describe the attributes of the database entity associated with the table, and each column has an associated data type.

  You can create new tables, edit existing tables, and add indexes by using the options available in the context menu. Or you can select **Use as Template**, which displays the dialog box with information for the selected object filled in. You should then change the object name and any other aspects of the definition, as needed, to create a new object of that type. For a description of the fields, see The Table Properties Dialog.

- **Views**: Views are virtual tables that select data from one or more underlying tables. For information about creating and editing a view, see The View Properties Dialog.

- **Indexes**: An index contains an entry for each value that appears in the indexed column of the table and provides direct, fast access to rows. For information about creating and editing an index for a table, see The Index Properties Dialog.

- **Packages**, **Functions**, **Procedures**: A package contains subprograms, which are programming objects that can be stored and executed in the database server, and called from other programming objects or applications. A package can contain functions or procedures, or both. Select **Run** from the context menu to insert the code for a selected function, procedure, or package into the worksheet.

- **Triggers**: Triggers are stored PL/SQL blocks associated with a table, a schema, the database, or anonymous PL/SQL blocks or calls to a procedure implemented in PL/SQL or Java.

- **Types**: A data type associates a fixed set of properties with the values that can be used in a column of a table or in an argument of a function or procedure.
- **Sequences**: Sequences are used to generate unique integers. You can use sequences to automatically generate primary key values. For information about creating and editing a sequence, see The Sequence Properties Dialog.
- **Materialized View**: A materialized view is a database object that contains the results of a query. The FROM clause of the query can name tables, views, and other materialized views. For information about creating and editing a materialized view, see Materialized View Pane.
- **Materialized Views Logs**: A materialized view log is a table associated with the master table of a materialized view. When DML changes are made to master table data, Oracle Database stores rows describing those changes in the materialized view log and then uses the materialized view log to refresh materialized views based on the master table. For information about creating and editing a materialized view log, see The Materialized View Log Properties Dialog.
- **Synonyms, Public Synonyms**: Synonyms provide alternative names for tables, views, sequences, procedures, stored functions, packages, materialized views, Java class database objects, user-defined object types, or other synonyms. For information about creating a synonym, see The Synonym Properties Dialog.
- **Database Links**: A database link is a database object in one database that enables you to access objects on another database. After you have created a database link, you can use it to refer to tables and views in the other database.

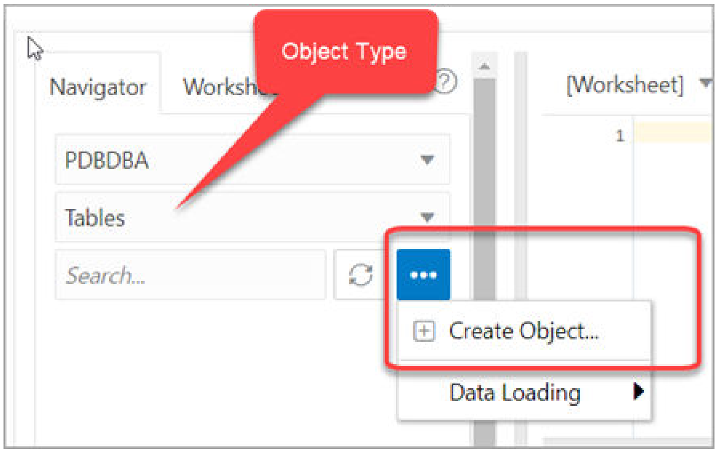
7.1 The Table Properties Dialog

The Table Properties Dialog is displayed when you create a table, edit an existing table, or create a table using an existing one as a template.

You can open the Table Properties dialog from the Navigator tab in SQL Worksheet or Data Modeler.

To create a table for a specific schema, in the Navigator tab, select **Tables** from the object type drop-down list, click Object submenu ••• , and select **Create Object**.

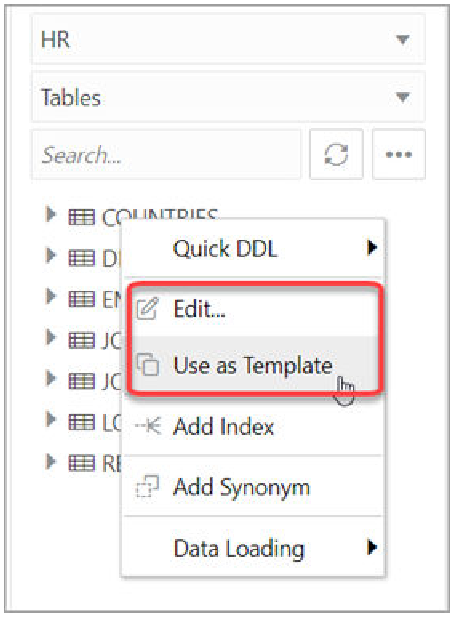
Figure 7-1   Create Object



To create a table from an existing one for a specific schema, right-click the table object in the Navigator tab, and select **Use as Template** as shown in Figure 7-2.

To edit a table for a specific schema, right-click a table object in the Navigator tab, and select **Edit**.

Figure 7-2 Edit Option for Existing Table



The table properties are grouped in several panes.

If you are editing an existing table, you can visit the panes in any order. If you click Create before you finish creating the table, right-click the table name, select **Edit**, and continue creating the table.

> **Note:**
>
> Editing a partitioned table is not recommended. To identify whether a table is partitioned or not, right-click the table name and select **Edit**. If the table is partitioned, a warning message will be displayed.

Schema: Database schema in which to create the table. By default, a new table is created in the existing schema or the schema that you are logged into.

Name: Name for the table.

The different panes in the dialog are described in the following sections:

- Columns Pane
- Primary Key Pane
- Unique Keys Pane
- Indexes Pane

- Foreign Keys Pane
- Table Constraints Pane
- Comments Pane
- Storage Pane
- External Table Properties Pane
- Materialized View Pane
- DDL Pane
- Output Pane

Related Topics

- *Oracle Database SQL Language Reference*

7.1.1 Columns Pane

Specifies properties for each column in the table.

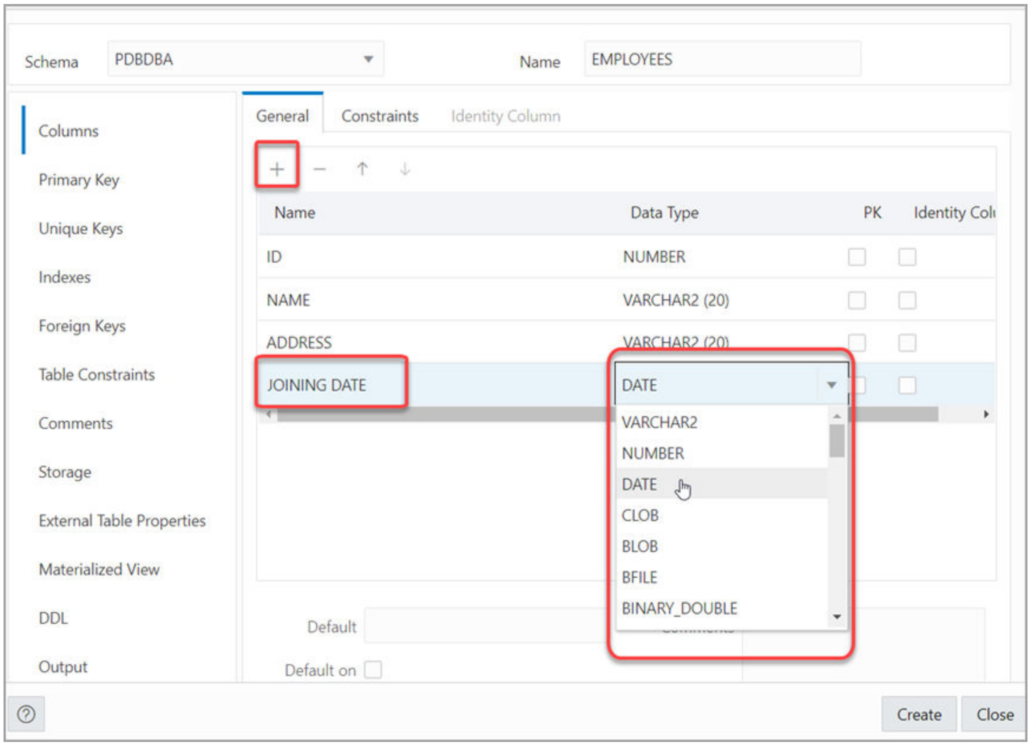
General tab

Lists the columns available in the table.

To add a column, click Add Column (+). A new row is added to the table below. Select the row and enter the details for the column.

To delete a column, select the row and click Remove Column (-). To move a column up or down in the table, select it and use the up-arrrow and down-arrow icons.

Figure 7-3 Add a Column



Schema PDBDBA

Name EMPLOYEES

Name	Data Type	PK	Identity Col
ID	NUMBER	□	□
NAME	VARCHAR2 (20)	□	□
ADDRESS	VARCHAR2 (20)	□	□
JOINING DATE	DATE	□	□

- **Name**: Name for the column.
- **Datatype**: Data type for the column.
- **Default**: If no value is specified, the default value inserted into the column when a row is inserted.
- **Default on NULL**: Applicable for Oracle Database 12c and later releases. If this option is selected, when a row is inserted into the table and the value specified for the column is NULL, the default value is inserted into the column.
- **Expression**: Expression for computing the value in the column.
- **Comments**: Optional descriptive comments about the column. Use this field to provide descriptions for the attributes.

In the table:

- **PK**: If this option is selected, the column becomes the primary key.
- **Identity Column**: If this option is selected, the column becomes an identity column. This is applicable only for Oracle Database 12c and later releases. For more details, see the Identity Column tab.

Constraints tab

Displays the Not Null and Check Constraints for a column. A check constraint requires values in a column to comply with a specified condition.

- **Not Null Constraint: Name**: Name for the Not Null constraint.
- **Not Null Constraint: Not Null**: If this option is selected, the column must contain data. You cannot specify no value or an explicit null value for this column when you insert a row. If this option is not checked, the column can contain either data or no data. A primary key column cannot be null.
- **Check Constraint: Name**: Name for the check constraint definition.
- **Check Constraint: Constraint**: Condition that must be met for a column to fulfill the check constraint. You can use any valid CHECK clause (without the CHECK keyword). For example, to indicate that the value in a numeric column named RATING must be from 1 to 10, you can specify: rating >=1 and rating <= 10.
- **Enabled**: If this option is selected, the constraint is checked when data is entered or updated in the column.
- **Deferrable**: If this option is selected, you can defer checking the validity of the constraint until the end of a transaction.
- **Initially Immediate**: If this option is selected, the constraint is checked whenever you add, update, or delete data from the column.
- **Validate**: If this option is selected, the existing data is checked to see if it conforms to the constraint.

Identity Column tab

Applicable for Oracle Database 12c and later releases. The Identity Column tab lists the properties of the identity column. This tab becomes available only after the Identity Column checkbox is selected for the column in the General tab. An identity column is an autoincrement column that can be used to identify a table row. Only one identity column can be specified for a table.

- **Generate**: Always means that values cannot be explicitly included for the identity column in INSERT OR UPDATE statements, By Default means values for the identity column are generated automatically if no values are specified explicitly, By Default on Null means values are generated for the column only when a NULL value is supplied.

- **Start with**: Starting value of the sequence.
- **Increment**: Interval between successive numbers in a sequence.
- **Min value**: Lowest possible value for the sequence. The default is 1 for an ascending sequence and -(10^26) for a descending sequence.
- **Max value**: Highest possible value for the sequence. The default is 10^27 for an ascending sequence and -1 for a descending sequence.
- **Cache** and **Cache size**: Cache causes sequence values to be preallocated in cache, which can improve application performance; Cache size indicates the number of sequence values preallocated in cache. No Cache causes sequence values not to be preallocated in cache.
- **Cycle**: Indicates whether the sequence "wraps around" to reuse numbers after reaching its maximum value (for an ascending sequence) or its minimum value (for a descending sequence). If cycling of values is not enabled, the sequence cannot generate more values after reaching its maximum or minimum value.
- **Order**: Indicates whether sequence numbers are generated in the order in which they are requested. If No Order is specified, sequence numbers are not guaranteed to be in the order in which they were requested.

7.1.2 Primary Key Pane

Specifies the primary key for the table.

The primary key is the column, or set of columns, that uniquely identifies each row in the table. If the Primary Key checkbox is selected for a column in the General tab, the corresponding fields are automatically populated in the Primary Key pane. You can make changes to the properties as required.

An index is automatically created on the primary key.

- **Name**: Name of the constraint to be associated with the primary key definition.
- **Enabled**: If this option is checked, the primary key constraint is enforced: that is, the data in the primary key column (or set of columns) must be unique and not null.
- **Index**: Name of the index to which the primary key refers.
- **Tablespace**: Name of the tablespace associated with the index.
- **Available Columns**: Lists the columns that are available to be added to the primary key definition. You can select multiple attributes, if required, for the primary key.
- **Selected Columns**: Lists the columns that are included in the primary key definition.

To add a column to the primary key definition, select it in Available Columns and click the Add (**>**) icon; to remove a column from the primary key definition, select it in Selected Columns and click the Remove (**<**) icon. To move all columns from available to selected (or the reverse), use the Add All (**>>**) or Remove All (**<<**) icon. To move a column up or down in the primary key definition, select it in Selected Columns and use the arrow buttons.

7.1.3 Unique Keys Pane

Specifies one or more unique constraints for the table.

A unique constraint specifies a column, or set of columns, whose data values must be unique: each data value must not be null, and it must not be the same as any other value in the column.

To add a unique constraint, click the Add button; to delete a unique constraint, select it and click the Remove button.

- **Name**: Name of the unique constraint.
- **Enabled**: If this option is selected, the unique constraint is enforced.
- **Rely**: If this option is selected, the constraint in NOVALIDATE mode is taken into account during query rewrite.
- **Deferrable**: If this option is selected, in subsequent transactions, constraint checking can be deferred until the end of the transaction using the SET CONSTRAINT(S) statement.
- **Initially Immediate**: If this option is selected, the constraint is checked at the end of each subsequent SQL statement.
- **Validate**: If the option is selected, the existing data is checked to see if it conforms to the constraint.
- **Index**: Name of the index to which the unique key refers.
- **Tablespace**: Name of the tablespace associated with the index.
- **Available Columns**: Lists the columns that are available to be added to the unique constraint definition.
- **Selected Columns**: Lists the columns that are included in the unique constraint definition.

To add a column to the unique constraint definition, select it in Available Columns and click the Add (**>**) icon; to remove a column from the unique constraint definition, select it in Selected Columns and click the Remove (**<**) icon. To move all columns from available to selected (or the reverse), use the Add All (**>>**) or Remove All (**<<**) icon. To move a column up or down in the unique constraint definition, select it in Selected Columns and use the arrow buttons.

7.1.4 Indexes Pane

Lists the indexes defined for the table.

To add an index, click Add Index (+); to delete an index, select it and click Remove Index (-).

- **Name**: Name of the index.
- **Type**: The type of Oracle index. `Non-unique` means that the index can contain multiple identical values; `Unique` means that no duplicate values are permitted; `Bitmap` stores rowids associated with a key value as a bitmap.
- **Tablespace**: Name of the tablespace for the index.

- **Expression**: A column expression is an expression built from columns, constants, SQL functions, and user-defined functions. When you specify a column expression, you create a function-based index.
- **Available Columns** and **Selected Columns**: Columns selected for the index. To select a column, click the column in the Available Columns box, and then click the click the Add Selected Columns icon to move it to the Selected Columns box.

7.1.5 Foreign Keys Pane

Specifies one or more foreign keys for the table.

A foreign key specifies a column ("local column"), whose data values match values in the primary key or unique constraint of another table.

- **Name**: Name of the foreign key definition.
- **Enabled**: If this option is checked, the foreign key is enforced.
- **Rely**, **Deferrable**, **Initially Immediate**, **Validate**: See the description of these fields in the Unique Keys pane.
- **Referenced Constraint: Schema**: Name of the schema containing the table with the primary key or unique constraint to which this foreign key refers.
- **Referenced Constraint: Table**: Name of the table with the primary key or unique constraint to which this foreign key refers.
- **Referenced Constraint: Constraint**: Name of the primary key or unique constraint to which this foreign key refers.
- **Referenced Constraint: On Delete**: Action to take automatically when a row in the referenced table is deleted and rows with that value exist in the table containing this foreign key: `NO ACTION` (shown by a crossing line in diagrams) performs no action on these rows; `CASCADE` (shown by an "X") deletes these rows; `SET NULL` (shown by a small circle) sets null all columns in those rows that can be set to a null value.
- **Assocations: Local Column**: Lists the column in the currently selected (local) table that is included in the foreign key definition. For each referenced column in the foreign key definition, select the name of a column in the edited table.
- **Associations: Referenced Column**: For each local column, identifies the column in the other (foreign) table that must have a value matching the value in the local column.

7.1.6 Table Constraints Pane

Specifies one or more check constraints for the table.

A check constraint specifies a condition that must be met when a row is inserted into the table or when an existing row is modified.

- **Name**: Name of the check constraint definition.
- **Check Condition**: Condition that must be met for a row to fulfil the check constraint. You can use any valid CHECK clause (without the CHECK keyword). For example, to indicate that the value in a numeric column named RATING must be from 1 to 10, you can specify rating >=1 and rating <= 10.
- **Enabled**: If this option is checked, the check constraint is enforced.

7.1.7 Comments Pane

Enter descriptive comments in this pane. This is optional.

7.1.8 Storage Pane

Enables you to specify storage options for the table.

When you create or edit a table or an index, you can override the default storage options.

- **Organization**: Specifies that the table is stored and organized with (Index) or without an index (Heap) or as an external table (External).
- **Tablespace**: Name of the tablespace for the table or index.
- **Logging**: ON means that the table creation and any subsequent INSERT operations against the table are logged in the redo log file. OFF means that these operations are not logged in the redo log file.
- **Row Archival**: YES enables in-database archiving, which allows you to archive rows within the table by marking them as invisible.

7.1.9 External Table Properties Pane

Specifies options for an external table.

An external table is a read-only table whose metadata is stored in the database but whose data in stored outside the database.

External Table

- **Access Driver Type**: Specifies the type of external table.
  - ORACLE_LOADER: Extracts data from text data files. This is the default access driver, which loads data from external tables to internal tables.
  - ORACLE_DATAPUMP: Extracts data from binary dump files. This access driver can perform both loads and unloads.
  - ORACLE_BIGDATA: Extracts data from Oracle Big Data Appliance.
  - ORACLE_HDFS: Extracts data stored in a Hadoop Distributed File System (HDFS).
  - ORACLE_HIVE: Extracts data stored in Apache HIVE.
- **Default Directory**: Specifies the default directory to use for all input and output files that do not explicitly name a directory object. The location is specified with a directory object, not a directory path.
- **Access Params**: Assigns values to the parameters of the specific access driver for the external table. Access parameters are optional.
  - OPAQUE_FORMAT_SPEC: The opaque_format_spec specifies all access parameters for the ORACLE_LOADER, ORACLE_DATAPUMP, ORACLE_HDFS, and ORACLE_HIVE access drivers. For descriptions of the access parameters, see *Oracle Database Utilities*. Field names specified in

the opaque_format_spec must match columns in the table definition, else Oracle Database ignores them.

  - USING CLOB: Enables you to derive the parameters and their values through a subquery. The subquery cannot contain any set operators or an ORDER BY clause. It must return one row containing a single item of data type CLOB.

- **Reject Limit**: The number of conversion errors that can occur during a query of the external data before an Oracle Database error is returned and the query is aborted.
- **Project Column**: Determines how the access driver validates the rows of an external table in subsequent queries.
  - ALL: Processes all column values, regardless of which columns are selected, and validates only those rows with fully valid column entries. If any column value raises an error, such as a data type conversion error, the row is rejected even if that column was not referenced in the select list of the query.
  - REFERENCED: Processes only those columns in the select list of the query.

  The ALL setting guarantees consistent result sets. The REFERENCED setting can result in different numbers of rows returned, depending on the columns referenced in subsequent queries, but is faster than the ALL setting. If a subsequent query selects all columns of the external table, then the settings behave identically.
- **Location**: Specifies the data files for the external table. Use the Add (+) icon to add each location specification.
  - For ORACLE_LOADER and ORACLE_DATAPUMP, the files are named in the form `directory:file`. The directory portion is optional. If it is missing, then the default directory is used as the directory for the file. If you are using the ORACLE_LOADER access driver, then you can use wildcards in the file name. An asterisk (*) signifies multiple characters and a question mark (?) signifies a single character.
  - For ORACLE_HDFS, LOCATION is a list of Uniform Resource Identifiers (URIs) for a directory or for a file. There is no directory object associated with a URI.
  - For ORACLE_HIVE, LOCATION is not used. Instead, the Hadoop HCatalog table is read to obtain information about the location of the data source (which could be a file or another database).

Opaque Format Spec

Specifies all access parameters for the ORACLE_LOADER, ORACLE_DATAPUMP, ORACLE_HDFS, and ORACLE_HIVE access drivers.

For example:

```
RECORDS DELIMITED BY NEWLINE CHARACTERSET US7ASCII
    TERRITORY AMERICAN
    BADFILE log_file_dir:'ext_1v3.bad'
    LOGFILE log_file_dir:'ext_1v3.log'
    FIELDS TERMINATED BY "|" OPTIONALLY ENCLOSED BY '^' LDRTRIM
    ( PROD_ID,
      CUST_ID ,
      TIME_ID DATE(10) "YYYY-MM-DD",
      CHANNEL_ID ,
      PROMO_ID ,
      QUANTITY_SOLD ,
      AMOUNT_SOLD ,
      UNIT_COST ,
      UNIT_PRICE
    )
```

and the full statement:

```
CREATE TABLE SH.SALES_TRANSACTIONS_EXT
    (
      PROD_ID NUMBER ,
      CUST_ID NUMBER ,
      TIME_ID DATE ,
      CHANNEL_ID NUMBER ,
      PROMO_ID NUMBER ,
      QUANTITY_SOLD NUMBER ,
     AMOUNT_SOLD NUMBER (10,2) ,
     UNIT_COST NUMBER (10,2) ,
     UNIT_PRICE NUMBER (10,2)
     )
     ORGANIZATION EXTERNAL
     (
     TYPE ORACLE_LOADER
     DEFAULT DIRECTORY DATA_FILE_DIR
     ACCESS PARAMETERS
     (
     RECORDS DELIMITED BY NEWLINE CHARACTERSET US7ASCII
    TERRITORY AMERICAN
    BADFILE log_file_dir:'ext_1v3.bad'
    LOGFILE log_file_dir:'ext_1v3.log'
    FIELDS TERMINATED BY "|" OPTIONALLY ENCLOSED BY '^' LDRTRIM
    ( PROD_ID ,
      CUST_ID ,
      TIME_ID DATE(10) "YYYY-MM-DD",
      CHANNEL_ID ,
      PROMO_ID ,
      QUANTITY_SOLD ,
      AMOUNT_SOLD ,
      UNIT_COST ,
      UNIT_PRICE
    )
    )
     LOCATION ( "DATA_FILE_DIR":'sale1v3.dat')
     )
     REJECT LIMIT 100
 ;
```

CLOB Subquery

Type or copy and paste the query.

> **Note:**
>
> For more information about the external table fields, see *Oracle Database Utilities* and *Oracle Database SQL Language Reference*

7.1.10 Materialized View Pane

Specifies options for a materialized view.

Query: Contains the SQL code for the query part of the view definition. Type or copy and paste the query.

General

- **On Pre-built Table**: If **Yes**, an existing table is registered as a preinitialized materialized view. This option is particularly useful for registering large materialized views in a data warehousing environment. The table must have the same name and be in the same schema as the resulting materialized view, and the table should reflect the materialization of a subquery.
- **Reduced Precision**: **Yes** authorizes the loss of precision that will result if the precision of the table or materialized view columns do not exactly match the precision returned by the subquery. If **No** , the precision of the table or materialized view columns must exactly match the precision returned by the subquery, or the create operation will fail.
- **For Update**: Select **Yes** to allow a subquery, primary key, object, or rowid materialized view to be updated. When used in conjunction with Advanced Replication, these updates will be propagated to the master.
- **Real Time MV**: Select **Yes** to create a real-time materialized view or a regular view. A real-time materialized view provides fresh data to user queries even when the materialized view is not in sync with its base tables due to data changes. Instead of modifying the materialized view, the optimizer writes a query that combines the existing rows in the materialized view with changes recorded in log files (either materialized view logs or the direct loader logs). This is called on-query computation.
- **Query Rewrite**: If **Enable**, the materialized view is enabled for query rewrite, which transforms a user request written in terms of master tables into a semantically equivalent request that includes one or more materialized views.
- **Build**: Specifies when to populate the materialized view. **Immediate** indicates that the materialized view is to be populated immediately. **Deferred** indicates that the materialized view is to be populated by the next refresh operation. If you specify Deferred, the first (deferred) refresh must always be a complete refresh; until then, the materialized view has a staleness value of unusable, so it cannot be used for query rewrite.
- **Use Index**: If **Yes**, a default index is created and used to speed up incremental (fast) refresh of the materialized view. If **No**, this default index is not created. (For example, you might choose to suppress the index creation now and to create such an index explicitly later.)
- **Index Tablespace**: Specifies the tablespace in which the materialized view is to be created. If a tablespace is not selected, the materialized view is created in the default tablespace of the schema containing the materialized view.

- **Cache**: If **Yes**, the blocks retrieved for this table are placed at the most recently used end of the least recently used (LRU) list in the buffer cache when a full table scan is performed. This setting is useful for small lookup tables. If **No**, the blocks are placed at the least recently used end of the LRU list.

Refresh Clause

- **Refresh**: Select Yes to enable refresh operations.
- **Refresh Type**: The method of refresh operation to be performed:
  - Complete Refresh: Executes the defining query of the materialized view, even if a fast refresh is possible.
  - Fast Refresh: Uses the incremental refresh method, which performs the refresh according to the changes that have occurred to the master tables. The changes for conventional DML changes are stored in the materialized view log associated with the master table. The changes for direct-path INSERT operations are stored in the direct loader log.
  - Force Refresh: Performs a fast refresh if one is possible; otherwise, performs a complete refresh.
- **Action**: The type of refresh operation to be performed:
  - On Demand: Performs a refresh when one of the DBMS_MVIEW refresh procedures are called.
  - On Commit: Performs a fast refresh whenever the database commits a transaction that operates on a master table of the materialized view. This may increase the time taken to complete the commit, because the database performs the refresh operation as part of the commit process.
  - Specify: Performs refresh operations according to what you specify in the Start on and Next fields.
- **Start Date**: Starting date and time for the first automatic refresh operation. Must be in the future.
- **Next Date**: Time for the next automatic refresh operation. The interval between the Start on and Next times establishes the interval for subsequent automatic refresh operations. If you do not specify a value, the refresh operation is performed only once at the time specified for Start on.
- **With**: Refresh type, which determines the type of materialized view:
  - Primary Key: Creates a primary key materialized view, which allows materialized view master tables to be reorganized without affecting the eligibility of the materialized view for fast refresh.
  - Row ID: Creates a rowid materialized view, which is useful if the materialized view does not include all primary key columns of the master tables.
- **Default Storage**: If Yes, DEFAULT specifies that Oracle Database will choose automatically which rollback segment to use. If you specify DEFAULT, you cannot specify the rollback_segment. DEFAULT is most useful when modifying, rather than creating, a materialized view.
- **Storage Type**: MASTER specifies the remote rollback segment to be used at the remote master site for the individual materialized view. LOCAL specifies the remote rollback segment to be used for the local refresh group that contains the materialized view. This is the default.
- **Rollback Segment**: Enter the name of the rollback segment.

- **Using Constraint**: If this option is checked, more rewrite alternatives can be used during the refresh operation, resulting in more efficient refresh execution. The behavior of this option is affected by whether you select Enforced or Trusted.
  - Enforced: Causes only enforced constraints to be used during the refresh operation.
  - Trusted: Enables the use of dimension and constraint information that has been declared trustworthy by the database administrator but that has not been validated by the database. If the dimension and constraint information is valid, performance may improve. However, if this information is invalid, then the refresh procedure may corrupt the materialized view even though it returns a success status.

7.1.11 DDL Pane

You can review and save the SQL statements that are generated when creating or editing the object. If you want to make any changes, go back to the relevant panes and make the changes there.

- For a new table, click **CREATE** to view the generated DDL statements.
- When you edit table properties, click **UPDATE** to view the generated ALTER statements. For a new table, the UPDATE tab will not be available.

When you are finished, click **Apply**.

7.1.12 Output Pane

Displays the results of the DDL commands. If there are any errors, go to the appropriate pane, fix the errors, and run the commands again. You can save to a text file or clear the output.

7.2 The Index Properties Dialog

The Index Properties dialog box is displayed when you create or edit an index.

To create an index for a selected schema, in SQL Worksheet, in the Navigator tab, select **Indexes** from the object type drop-down list, click Object submenu ••• , and select **Create Object**.

To edit an index for a selected schema, right-click a table object in the Navigator tab, and select **Edit**.

Definition pane

- **Schema**: Database schema that owns the table associated with the index.
- **Table**: Name of the table associated with the index.
- **Schema**: Database in which to create the index.
- **Tablespace**: Tablespace for the index.
- **Name**: Name of the index.
- **Type**: The type of Oracle index.
  - **Non-unique** means that the index can contain multiple identical values.

  - **Unique** means that no duplicate values are permitted.
  - **Bitmap** stores rowids associated with a key value as a bitmap.
- **Expression**: A column name or column expression. A column expression is an expression built from columns, constants, SQL functions, and user-defined functions. When you specify a column expression, you create a function-based index.
- **Available Columns**: Columns available in the table.
- **Selected Columns**: Columns selected for the index. Click Add Selected Columns > to move columns from the Available Columns list.
- **Order**: **ASC** for an ascending index (index values sorted in ascending order); **DESC** for a descending index (index values sorted in descending order).

DDL pane

You can review and save the SQL statements that are generated when creating or editing the index. If you want to make any changes, go back to the Definition pane and make the changes there.

- For a new index, click **CREATE** to view the generated DDL statements.
- When you edit index properties, click **UPDATE** to view the generated ALTER statements. For a new index, the UPDATE tab will not be available.

When you are finished, click **Apply**.

Output pane

Displays the results of the DDL commands. If there are any errors, go to the Definition pane, fix the errors, and run the commands again. You can save to a text file or clear the output.

7.3 The Sequence Properties Dialog

The Sequence Properties Dialog is displayed when you create or edit a sequence.

To create a sequence for a selected schema, in SQL Worksheet, in the Navigator tab, select **Sequences** from the object type drop-down list, click Object submenu ••• , and select **Create Object**.

To edit a sequence for a selected schema, right-click a sequence object in the Navigator tab and select **Edit**.

A sequence is an object from which multiple users may generate unique integers. You can use sequences to automatically generate primary key values.

Properties pane

- **Schema**: Database schema in which to create the sequence.
- **Name**: Name of the sequence.
- **Start with**: Starting value of the sequence.
- **Increment**: Interval between successive numbers in a sequence.
- **Min value**: Lowest possible value for the sequence. The default is 1 for an ascending sequence and -(10^26) for a descending sequence.

- **Max value**: Highest possible value for the sequence. The default is 10^27 for an ascending sequence and -1 for a descending sequence.
- **Cache** and **Cache size**: `Cache` causes sequence values to be preallocated in cache, which can improve application performance; Cache size indicates the number of sequence values preallocated in cache. `No Cache` causes sequence values not to be preallocated in cache.
- **Cycle**: Indicates whether the sequence "wraps around" to reuse numbers after reaching its maximum value (for an ascending sequence) or its minimum value (for a descending sequence). If cycling of values is not enabled, the sequence cannot generate more values after reaching its maximum or minimum value.
- **Order**: Indicates whether sequence numbers are generated in the order in which they are requested. If **No Order** is specified, sequence numbers are not guaranteed to be in the order in which they were requested.

DDL pane

You can review and save the SQL statements that are generated when creating or editing the sequence. If you want to make any changes, go back to the Properties pane and make the changes there.

- For a new sequence, click **CREATE** to view the generated DDL statements.
- When you edit a sequence, click **UPDATE** to view the generated ALTER statements. For a new sequence, the UPDATE tab will not be available.

When you are finished, click **Apply**.

Output pane

Displays the results of the DDL commands. If there are any errors, go to the Properties pane, fix the errors, and run the commands again. You have save to a text file or clear the output.

7.4 The View Properties Dialog

The View Properties Dialog is displayed when you create or edit a view.

You can open the View Properties Dialog from the Navigator tab in SQL Worksheet or Data Modeler.

To create a view for a selected schema, in SQL Worksheet, in the Navigator tab, select **Views** from the object type drop-down list, click Object submenu ••• , and select **Create Object**.

To create a view from an existing template for a selected schema, in the Navigator tab, select the view to create from, right-click and select **Use as Template**.

To edit a view for a selected schema, right-click a view object in the Navigator pane, and select **Edit**.

Schema: Database schema in which to create the view.

Name: Name of the view.

The different panes in the dialog are described in the following sections:

SQL Query pane

Enter or copy and paste the SQL query for the view, using the SELECT and FROM keywords along with the syntax needed to retrieve the desired information. A semicolon is not required after the query.

Columns pane

Click Refresh Columns ⟳ to automatically populate the columns in this pane. You can edit the columns by selecting the required row and making changes in the Header Alias and Comments fields.

Storage pane

- **Force on Create**: Select **Yes** to create the view regardless of whether the base tables of the view or the referenced object types exist or the owner of the schema containing the view has privileges on them. These conditions must be true before any SELECT, INSERT, UPDATE, or DELETE statements can be issued against the view. If the view definition contains any constraints, CREATE VIEW ... FORCE fails if the base table does not exist or the referenced object type does not exist. CREATE VIEW ... FORCE also fails if the view definition names a constraint that does not exist.
- **Query Restriction**: **Read Only** prevents the view from being used to add, delete, or change data in the underlying table. **Check Option** prohibits any changes to the underlying table that would produce rows that are not included in this view.

Use the Primary Key, Unique Keys, Foreign Keys, and Comments panes to add or edit properties as required.

DDL pane

Based on the inputs provided, the DDL statements are generated. You can review and save the SQL statements. If you want to make any changes, go back to the relevant pane and make the changes there.

- For a new view, click **CREATE** to view the generated DDL statements.
- When you edit a view, click **UPDATE** to view the generated ALTER statements. For a new view, the UPDATE tab will not be available.

When you are finished, click **Apply**.

Output pane

Displays the results of the DDL commands. If there are any errors, go to the respective pane, fix the errors, and run the commands again. You can save to a text file or clear the output.

7.5 The Synonym Properties Dialog

The Synonym Properties Dialog is displayed when you create a synonym.

There are two ways of creating a synonym for a selected schema:

- In SQL Worksheet, in the Navigator tab, right-click the object for which you want to create the synonym, and select **Add Synonym**. In this case, the only fields that you can edit in the Properties pane are Public and Synonym Name. The values of the remaining fields are predetermined by the object selected.

- In SQL Worksheet, in the Navigator tab, select the object type as **Synonyms** or **Public Synonyms** from the drop-down list. Click Object submenu ••• , and select **Create Object**. All the fields in the Properties dialog are available for edit.

The different panes in the dialog are described in the following sections:

Properties pane

- **Public**: If this option is checked, the synonym is accessible to all users. However, each user must have appropriate privileges on the underlying object to use the synonym. If this option is not checked, the synonym is a private synonym, and is accessible only within its schema.
- **Synonym Schema**: Database schema in which to create the synonym.
- **Synonym Name**: Name of the synonym. A private synonym must be unique within its schema; a public synonym must be unique within the database.
- **Object Type**: Specify the type of object to which this synonym refers.
- **Object Schema**: Schema containing the object or name to which this synonym refers.
- **DB Filter**: After selecting the Object Type and Object Schema, the list of objects of the selected type may be very long. To filter the object names, enter the search entry and click Refresh ⟳ . The Object Name field is auto-filled with appropriate object names in the drop-down list.
- **Object Name**: Select the name of the object to which this synonym refers.
- **DB Link**: Enter a complete or partial database link to create a synonym for a schema object on a remote database where the object is located. If you specify DB Link and omit schema, then the synonym refers to an object in the schema specified by the database link. Oracle recommends that you specify the schema containing the object in the remote database. If you omit DB Link, then Oracle Database assumes the object is located on the local database.

DDL pane

Based on the inputs provided, the DDL statements are generated. You can review and save the SQL statements. If you want to make any changes, go back to the relevant pane and make the changes there.

- For a new view, click **CREATE** to view the generated DDL statements.
- When you edit a view, click **UPDATE** to view the generated ALTER statements. For a new view, the UPDATE tab will not be available.

When you are finished, click **Apply**.

Output pane

Displays the results of the DDL commands. If there are any errors, go to the respective pane, fix the errors, and run the commands again. You can save to a text file or clear the output.

7.6 Implied Foreign Keys

Implied foreign keys are dependencies that exist between tables but are not defined in the database. In a data warehouse environment, it is a common practice not to create foreign keys. However, it becomes necessary to show these dependencies for presentation or reporting purposes.

You can display implied foreign keys for objects in a star schema by defining them or by discovering them in the data dictionary. It is possible to have more than one source for implied foreign keys.

Define Implied Foreign Keys

You can defined implied foreign keys in two ways:

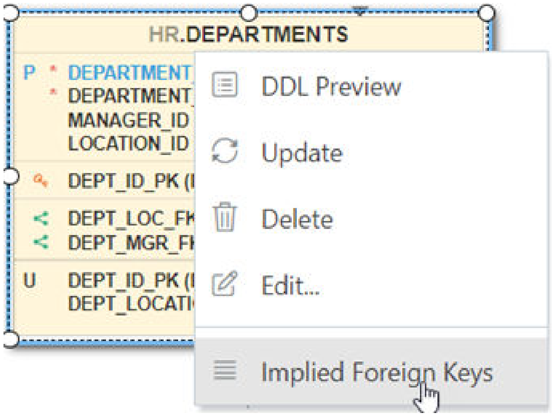
- Using the Implied Foreign Keys dialog
- By dragging the arrow to the referenced object in the diagram

Using the Implied Foreign Keys Dialog

You can define implied foreign keys using the Implied Foreign Keys dialog in Data Modeler.

1. In a Data Modeler diagram, right-click an object (table or view) and select **Implied Foreign Keys**.

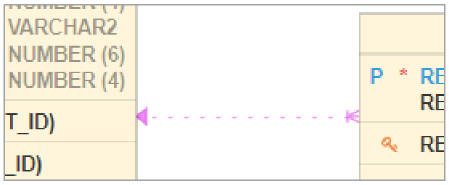
   Figure 7-4 Select Implied Foreign Keys for an Object

   The Implied Foreign Keys dialog is displayed.

2. In the Implied Foreign Keys dialog, click + to add an entry in the grid.
3. Select the entry in the grid to enable and enter values in the following fields:
   - **Referenced Object**: Object in the diagram that has a dependency to the source object.
   - **Local Column**: Name of the column in the source object.
   - **Referenced Column**: Name of the column in the targeted object.
   - **Discovery Sources**: Automatically prefilled, displays whether the implied foreign keys have been defined or were discovered in the data dictionary.
4. Click **OK**. The implied foreign key dependency is displayed with a dotted line on the diagram.

Figure 7-5 Dotted Line Between Two Objects

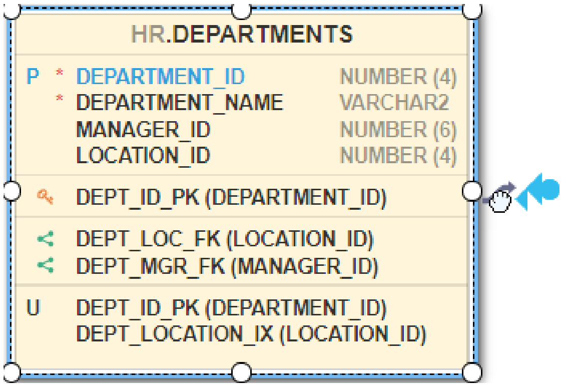


By Dragging the Arrow to the Referenced Object in the Diagram

You can also define an implied foreign key in the following way:

1. Select the source object on the diagram.
2. Click and drag the small curved arrow with a blue indicator to the referenced object. The dependency will be displayed with a dotted line on the diagram.

Figure 7-6 Drag Blue Indicator



HR.DEPARTMENTS

P * DEPARTMENT_ID NUMBER (4)
* DEPARTMENT_NAME VARCHAR2
MANAGER_ID NUMBER (6)
LOCATION_ID NUMBER (4)

DEPT_ID_PK (DEPARTMENT_ID)

DEPT_LOC_FK (LOCATION_ID)
DEPT_MGR_FK (MANAGER_ID)

U DEPT_ID_PK (DEPARTMENT_ID)
DEPT_LOCATION_IX (LOCATION_ID)

3. Right-click the source object and enter the column names in the Implied Foreign Keys dialog.

Discover Implied Foreign Keys in the Data Dictionary

A star schema can be discovered by searching for several types of definitions in the data dictionary.

In the Navigator tab, right-click an object (table or view) and select **Add Object as Star Schema to Diagram**. The object must be a fact table. The data dictionary is then searched for joins and dependencies related to the object, such as:

- Foreign keys defined for the selected table to other tables. If implied foreign keys are later discovered for the same columns, they will not be displayed on the diagram.
- Joins used in the definitions of bitmap join indexes, materialized views with aggregates, and attribute clustering.
- Dependencies based on dimension definitions and column name matching in fact table.
- Fact and dimension definitions for Analytic views and OLAP cube and dimensions.

7.7 The Materialized View Log Properties Dialog

The Materialized View Log Properties dialog is displayed when you create or edit a materialized view log, which is a table associated with the master table of a materialized view.

To create a materialized view log for a selected schema, in SQL Worksheet, in the Navigator tab, select **Materialized View Logs** from the object type drop-down list, click Object submenu ••• , and select **Create Object**.

To edit, right-click a materialized view log object in the Navigator pane and select **Edit**.

Schema: Database schema in which to create the materialized view log.

Table: Name of the master table of the materialized view to be associated with this materialized view log.

Properties tab

- **Row ID Logged**: **Yes** indicates that the rowid of all rows changed should be recorded in the materialized view log. **No** indicates that the rowid of all rows changed should not be recorded in the materialized view log.
- **PK Logged**: **Yes** indicates that the primary key of all rows changed should be recorded in the materialized view log; **No** indicates that the primary key of all rows changed should not be recorded in the materialized view log.
- **New values**: **Yes** saves both old and new values for update DML operations in the materialized view log; **No** disables the recording of new values in the materialized view log. If this log is for a table on which you have a single-table materialized aggregate view, and if you want the materialized view to be eligible for fast refresh, you must specify Yes.
- **Object ID Logged**: For a log on an object table only: **Yes** indicates that the system-generated or user-defined object identifier of every modified row should be recorded in the materialized view log. **No** indicates that the system-generated or user-defined object identifier of every modified row should not be recorded in the materialized view log.
- **Cache**: For data that will be accessed frequently, **CACHE** specifies that the blocks retrieved for this log are placed at the most recently used end of the least recently used list in the buffer cache when a full table scan is performed. This attribute is useful for small lookup tables. **NOCACHE** specifies that the blocks are placed at the least recently used end of the LRU list.
- **Parallel**: If **YES**, parallel operations will be supported for the materialized view log.
- **Sequence Logged**: **Yes** indicates that a sequence value providing additional ordering information should be recorded in the materialized view log. **No** indicates that a sequence value providing additional ordering information should not be recorded in the materialized view log. Sequence numbers (that is, Yes for this option) are necessary to support fast refresh after some update scenarios.
- **Commit SCN**: If this option is enabled, the database is instructed to use commit SCN data rather than timestamps.
- **Available Columns** and **Selected Columns**: Additional columns, which are non-primary-key columns referenced by subquery materialized views, to be recorded

in the materialized view log. To select one or more filter columns, use the arrow buttons to move columns from Available to Selected.

Storage tab

- **Tablespace**: Tablespace in which the materialized view log is to be created
- **Logging**: **YES** or **NO**, to establish the logging characteristics for the materialized view log.
- **Buffer Mode**: Select **KEEP** to put blocks from the segment into the KEEP buffer pool. Select **RECYCLE** to put blocks from the segment into the RECYCLE pool. Select **DEFAULT** to indicate the default buffer pool.
- **Percent Free**: Specify a whole number representing the percentage of space in each data block of the database object reserved for future updates to rows of the object. The value of PCTFREE must be a value from 0 to 99.
- **Percent Used**: Specify a whole number representing the minimum percentage of used space that Oracle maintains for each data block of the database object. PCTUSED is specified as a positive integer from 0 to 99 and defaults to 40.
- **Initrans**: Specify the initial number of concurrent transaction entries allocated within each data block allocated to the database object. This value can range from 1 to 255 and defaults to 1.
- **Freelists**: In tablespaces with manual segment-space management, for objects other than tablespaces and rollback segments, specify the number of free lists for each of the free list groups for the table, partition, cluster, or index. The default and minimum value for this parameter is 1, meaning that each free list group contains one free list.
- **Freelist Groups**: In tablespaces with manual segment-space management, specify the number of groups of free lists for the database object you are creating.
- **Initial Extent**: Specify the size of the first extent of the object.
- **Next Extent**: Specify in bytes the size of the next extent to be allocated to the object.
- **Percent Increase**: In locally managed tablespaces, Oracle Database uses the value of PCTINCREASE during segment creation to determine the initial segment size and ignores this parameter during subsequent space allocation.
- **Min Extent**: In locally managed tablespaces, Oracle Database uses the value of MINEXTENTS in conjunction with PCTINCREASE, INITIAL and NEXT to determine the initial segment size.
- **Max Extent**: This storage parameter is valid only for objects in dictionary-managed tablespaces. Specify the total number of extents, including the first, that Oracle can allocate for the object.
- **Unlimited**: Select this option if you want extents to be allocated automatically as needed. Oracle recommends this setting as a way to minimize fragmentation.

Purge tab

- **Type**: In **IMMEDIATE SYNCHRONOUS**, the materialized view log is purged immediately after refresh. This is the default. In **IMMEDIATE ASYNCHRONOUS**, the materialized view log is purged in a separate Oracle Scheduler job after the refresh operation.

- **Deferred, Start With, Next, Repeat Interval**: Sets up a scheduled purge that is independent of the materialized view refresh and is initiated during CREATE or ALTER MATERIALIZED VIEW LOG statement.

Refresh tab

- **Type**: **Synchronous Refresh** creates a staging log that can be used for synchronous refresh. Specify the name of the staging log to be created. The staging log will be created in the schema in which the master table resides. **Fast Refresh** creates a materialized view log that can be used for fast refresh. The materialized view log will be created in the same schema in which the master table resides. This is the default.

DDL pane

Based on the inputs provided, the DDL statements are generated. You can review and save the SQL statements. If you want to make any changes, go back to the relevant pane and make the changes there.

- For a new materialized view log, click **CREATE** to view the generated DDL statements.
- When you edit a materialized view log, click **UPDATE** to view the generated ALTER statements. For a new materialized view log, the UPDATE tab will not be available.

When you are finished, click **Apply**.

Output pane

Displays the results of the DDL commands. If there are any errors, go to the respective pane, fix the errors, and run the commands again. You can save to a text file or clear the output.

8

The User Management Page

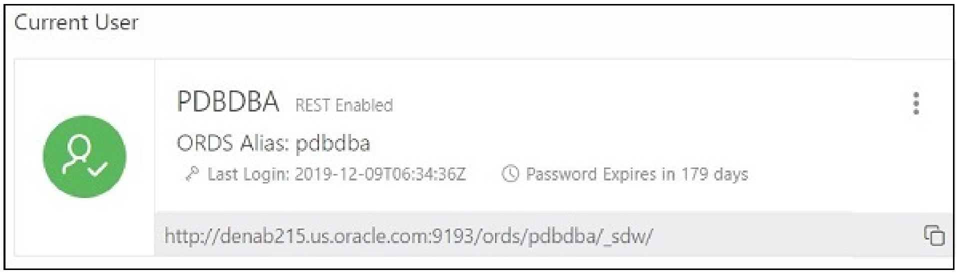
👁 Available only if you signed in as a database user with the DBA and PDB_DBA roles.

The User Management page enables you to perform user management tasks such as create, edit, enable REST, and delete users.

To reach the User Management page, click **User Management** in the Database Actions page, or click Selector ≡ and select **User Management** from the Administration menu in the navigation pane.

The User Management page consists of two parts: Current User and All Users.

Current User



Current User

PDBDBA REST Enabled

ORDS Alias: pdbdba

Last Login: 2019-12-09T06:34:36Z

Password Expires in 179 days

http://denab215.us.oracle.com:9193/ords/pdbdba/_sdw/

Displays information about the current user such as user name, whether REST Enabled or not, REST Alias, account expiration (in days), and the last login date and time. The icon on the left displays the user status with one of the following colours: green (Open), blue (Locked), and red (Expired).

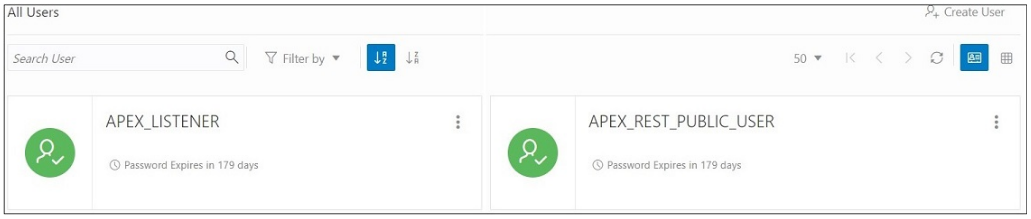
The URL at the bottom is displayed only if the user is REST Enabled. It provides the URL to the SQL Developer Web user login page. Use Copy to Clipboard ⧉ to copy the URL to the user's clipboard.

Click Object submenu ⋮ to open the context menu. The actions available are:

- **Edit**: Opens the Edit User Dialog, where you can edit current user information. See The Create/Edit User Dialog.
- **Enable REST**: Enables REST for a user where disabled. When this option is selected the first time, it opens the Enable REST dialog. See The REST Enable User Dialog.
- **Disable REST**: Disables REST where enabled for a user.
- **Drop REST Services**: Removes the REST data for a user, such as REST Alias, Base Path and so on, which is stored in Oracle REST Data Services (ORDS).

- **Delete**: Opens the Delete User dialog, where you can delete the user. See The Delete User Dialog.

All Users



Displays information about all other users that have been created in the database. You can use the Search field, which is case insensitive, to search for users, or sort the users in ascending or descending order using the sort icons, or filter by user status or REST status.

To create a user, click **Create User** to open the Create User dialog. For more information, see The Create/Edit User Dialog.

There are two views available:

▤ (Card View): Displays the user information in a card view. This is the default display view. Each user card provides details such as user status, password expiry, user name, and the context menu.

▦ (Grid View): Displays the user information in a tabular format. The last column in each row contains the context menu icon.

8.1 The Create/Edit User Dialog

This dialog is used for creating a new database user or editing an existing database user.

- To create a new database user, click **Create User** in the User Management page.
- To edit an existing database user, select **Edit** from the context menu for the associated user.

The user properties are grouped under two tabs: User and Granted Roles.

User Tab

Specifies general properties for the database user.

- **User Name**: The user name string. For an existing user, this field is read-only. To change the name, you must drop the user and create a new user with the desired name.
- **New Password**: Password string for the new user, or new password for an existing user. You must also type the same password string for **Confirm Password**.

- **Password Expired**: If this option is selected, the password is marked as expired, and the user must change the password before being permitted to connect to the database.
- **Account is Locked**: If this option is selected, the user will not be permitted to connect to the database until a DBA user unlocks the account associated with this user.
- **REST Enable**: If this option is selected, the user is enabled for REST. Enter the related fields: Alias, URL Mapping Type and Authorization required. These options are also available in the REST Enable dialog.

Granted Roles Tab

Specifies roles to be granted to the user. For each role, you can check **Granted** to grant the role, **Admin** to permit the user to grant the role to other users, and **Default** to use the default settings for Granted and Admin. A new user is granted **CONNECT** and **RESOURCE** roles when REST Enable is selected.

8.2 The REST Enable User Dialog

The dialog is used to enable REST for a user that has not been REST enabled.

To open this dialog, in the User Management page, select **Enable REST** from the user's context menu

When you select Enable REST for a user for the first time, the REST Enable User dialog is displayed. Subsequently, if you disable REST and then select Enable REST again, you receive a message stating that REST is enabled. In this case, the REST data previously provided is used for enabling REST. To enter new REST data, select **Drop REST Services** and then select **Enable REST** again.

Schema Alias: Enter the alias for the schema name to use in the URL.

URL Mapping Type: Select **BASE_PATH** or **BASE_URL**.

Authorization Required: For a schema, controls whether Oracle REST Data Services should require user authorization before allowing access to the Oracle REST Data Services metadata catalog of this schema.

8.3 The Delete User Dialog

This dialog is used to delete users.

In the User Management page, select **Delete** from the user's context menu to delete a user.

- **Cascade**: If this option is selected, all dependent objects are also deleted.
- **Drop REST Services**: If this option is selected, all user REST data is removed from ORDS.

> **Note:**
>
> If you do not select this option when deleting a user, the next time you create a user with the same user name, it will still retain the REST-enabled property.

Click **Delete User** and a confirmation or error notification is displayed.

9
The Monitoring Pages

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Monitoring menu provides access to several pages to view the performance and other characteristics of your database.

- The Monitoring Overview Page
- The Performance Hub Page
- The Instance Viewer Page
- The Logins Page
- The Alerts Page
- The Sessions Page
- The Storage Page
- The Parameters Page
- The Real Time SQL Monitoring Page
- The Top SQL Page
- The Waits Page

9.1 The Monitoring Overview Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Monitoring Overview page displays general information about the database.

To reach this page, click Selector ≡ to see the navigation pane, click the down arrow to expand **Monitoring**, and then click **Overview**.

The widgets on this page show snapshot information about the database status, online database storage, sessions, wait events, user accounts, alerts, and expiring passwords. Click a widget to go to its page where you can see a more detailed view of the data.

- **Database Status**: Shows the status of the database.
- **Used Online Database Storage**: Displays how much storage is being used by the database. You can click the title to open The Storage Page.
- **Sessions**: Displays the number of sessions by session status. You can click the title to open The Sessions Page.
- **Waits**: Displays how many wait events are occurring in the database by wait event class. When you click the title:

  - For Oracle Database 19c and later releases, the Performance Hub page is displayed.
  - For Oracle Database 18c and previous releases, the Activity-Waits page is displayed.
- **Users**: Displays how many user accounts are in the open, locked and expired statuses. You can hover over a status to see a list of the user accounts with that status.
  - **Open**: This status indicates that the user's account is unlocked and access to the database is enabled.
  - **Locked**: This status indicates that the user's account is locked and access to the database is disabled. The account must be unlocked to enable access to the database.
  - **Expired**: This status indicates that the user's password has expired and must be changed before the user can log in to the database.
- **Alerts**: Displays a summary of alerts over the last 7 days. You can click the title to open The Alerts Page
- **Expiring Passwords**: Shows a list of user accounts and whether a user account password has expired or the number of days before it will expire.

9.2 The Performance Hub Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles for Oracle Database 19c and later releases.

The Performance Hub page shows performance data for a time period you specify.

To reach the Performance Hub page, click **Performance Hub** in the Database Actions page, or click Selector ≡ and select **Performance Hub** from the Monitoring menu in the navigation pane.

> **Note:**
>
> The Performance Hub page is available in the following user interface languages: French, Japanese, Korean, Traditional Chinese, and Simplified Chinese. If you change the language to German, Spanish, Italian, or Portuguese in Preferences, the Performance Hub page reverts to English.

The Performance Hub page consists of these parts:

- Time Range Area: Use the controls in time range area at the top of the page to specify the time period for which you want to view performance data.
- ASH Analytics Tab: Use this tab to explore ASH (Active Session History) information across a variety of different dimensions for the specified time period.
- SQL Monitoring Tab: Use this tab to view the top 100 SQL statement executions by different dimensions for the specified time period, and to view details of SQL statement executions you select.

Time Range Area

Use these controls in the time range area to specify the time period for which you want to view performance data:

- **Select Duration**: Use this drop-down list, located in the top right of the time range area, to set the timeframe displayed in the timeline. You can choose **Last hour**, **Last 8 hours**, **Last 24 hours**, **Last Week**, or you can choose **Custom** and define your own timeframe.
- **Timeline**: The timeline displays a graph spanning the timeframe selected in the timeframe dropdown, showing Waits, User I/O and CPU usage during the period. At its end is the time slider.
- **Time Slider**: The time slider is a box you can drag back and forth along the current timeline. Use it to pick the specific time period within the timeframe for which you want to view performance data. You can also drag the side handles on the time slider to make it wider or narrower to encompass a longer or shorter time period.

ASH Analytics Tab

The ASH Analytics tab consists of the Average Active Sessions chart and two secondary tables below it.

- **Average Active Sessions Chart**: This chart shows performance information for the time period defined by the time slider. You can choose to chart different dimensions of information, such as **Wait Class**, **Wait Event**, or **Service**, by selecting the dimension from the drop-down list to the right of the chart title.

  You can download an AWR (Automatic Workload Repository) report for the current time period by right-clicking in the Average Active Sessions chart area and choosing **Generate AWR Report**.

  For more information about ASH and AWR, see Active Session History (ASH) and Automatic Workload Repository (AWR) in *Oracle Database Concepts*.
- **Secondary Tables**: The two tables below the Average Active Sessions chart show the information dimension chosen in the chart filtered by another dimension you choose. For example, if the Average Active Sessions chart is showing Wait Class, you could show SQL ID and User Session dimension information by Wait Class, one in each of the two secondary tables.

  In the secondary tables, the SQL ID and User Session dimensions provide links to SQL Details and Session Details pages, respectively, for the dimension data listed in the table.

SQL Monitoring Tab

The SQL Monitoring tab shows a table of the top 100 monitored SQL statements that were executing or that completed during the selected time period.

The table displays information about monitored SQL statement executions. If there is a green spinning icon in the Status column, then the monitored statement did not complete during the selected time period. A red cross indicates that the SQL did not complete either due to an error or due to the session getting terminated. If there is a check mark in the Status column, then the statement completed its execution during the selected time period.

SQL statements are monitored only if they have consumed at least 5 seconds of CPU or I/O time.

You can view information such as the status of a statement, its duration, its type (SQL, PL/SQL, or DBOP), its SQL ID, its SQL plan hash, the user who issued it, whether it executed as a serial or parallel statement, the time the database spent performing CPU activity, I/O, or other activity for the statement, the read and write requests and bytes associated with the statement, and the start and end time for the statement.

Click a SQL ID to display the SQL Details page with more information about that SQL statement.

9.3 The Instance Viewer Page

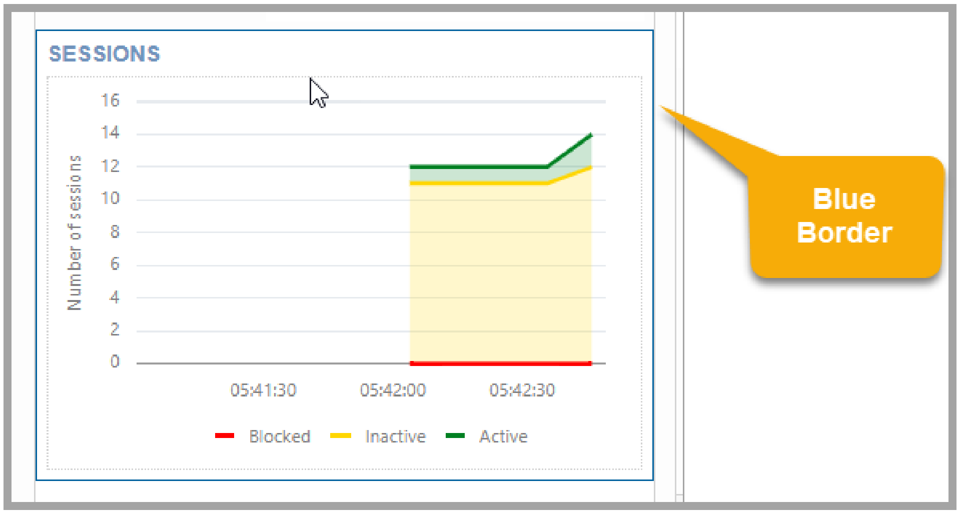
Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Instance Viewer is a performance dashboard that enables a DBA user to see a graphical representation of information about the instance associated with a connection.

To reach the Instance Viewer page, click **Instance Viewer** in the Database Actions page, or click Selector ☰ and select **Instance Viewer** from the Monitoring menu in the navigation pane.

When hovering over a chart, a dark blue border indicates that you can click to drill down to the relevant page for more detailed information.

Figure 9-1 Dark Blue Border to Drill Down



The types of information displayed include:

- Database
- Clients
- Sessions

- Processes (Counts, Execution Rate, Parse Rate, Open Cursors, Commit Rate)
- Waits
- Memory (DB Block Rate, Logical Reads, Allocation, Redo Generation)
- Storage (Files, Redo Log)
- DB CPU Ratio (database operations as a percentage of CPU activity)
- Top SQL (provides performance metrics on SQL operations, which are sorted based on the CPU time required)

9.4 The Logins Page

👁 Available only if the following three conditions are met:

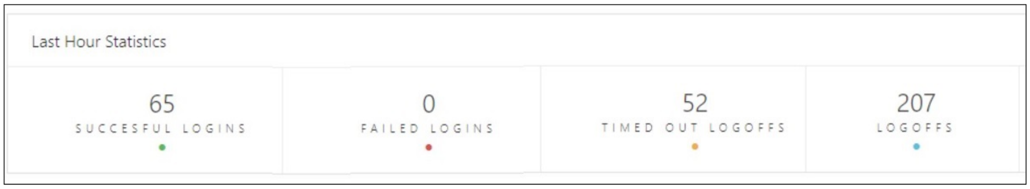
- You are signed in as a database user with the DBA and PDB_DBA roles for Oracle Database 12c and later releases.
- Auditing is enabled in the database.
- At least one of the four actions displayed on the Logins page must have an audit policy that is enabled. By default, Failed Logins is enabled.

To reach the Logins page, click **Logins** in the Database Actions page, or click Selector ≡ and select **Logins** from the Monitoring menu in the navigation pane.

The Logins page shows the number of succesful logins, failed logins, timed-out logoffs, and logoffs that have occurred within the last hour. The Last Hour Statistics widget only displays the actions being audited. If any of the tabs (succesful logins, failed logins, timedout logoffs or logoffs) are disabled, the database is currently not auditing that action. If you want to activate the action, you must create an audit policy with the desired action. To create an audit policy, see Auditing Activities.

Click a tab in Last Hour Statistics to view the corresponding entries in the table.

Figure 9-2 Tabs in Last Hour Statistics



Last Hour Statistics

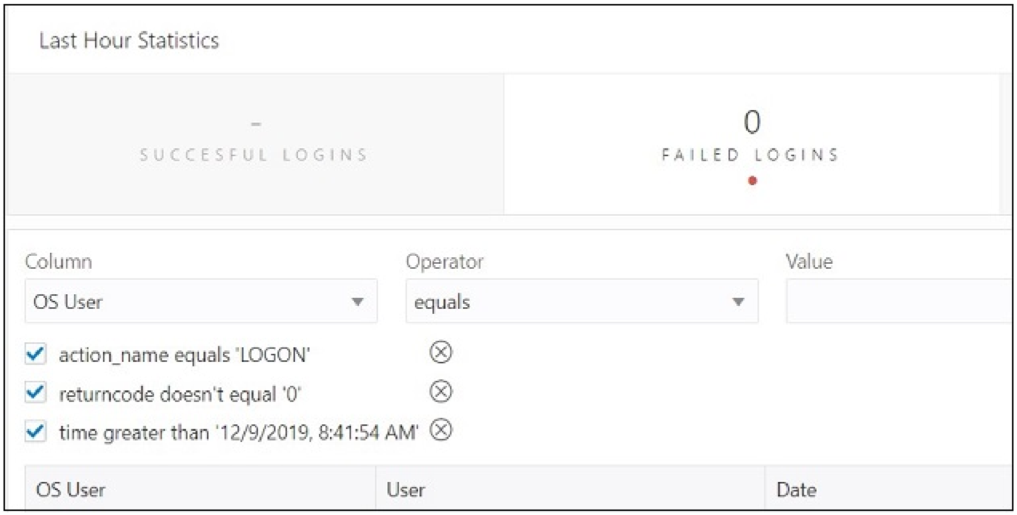
65 SUCCESFUL LOGINS	0 FAILED LOGINS	52 TIMED OUT LOGOFFS	207 LOGOFFS

You can search for results in the table by entering values in the Column, Operator and Value fields and clicking Filter 🔍. You can also filter by selecting one or more of the available filter options for each action.

Click Refresh ⟳ to refresh the data in the grid.

The following figure shows the three filters for the Failed Logins action.

Figure 9-3   Filters for Failed Logins



Last Hour Statistics

- SUCCESFUL LOGINS

0 FAILED LOGINS

Column: OS User ▼

Operator: equals ▼

Value:

☑ action_name equals 'LOGON' ⊗

☑ returncode doesn't equal '0' ⊗

☑ time greater than '12/9/2019, 8:41:54 AM' ⊗

OS User	User	Date

To return to the previous page, remove filters by clicking Remove Filter ⊗ or refresh the browser.

9.5 The Alerts Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Alerts page is a chronological log of messages and errors and is commonly used to learn whether the background processes have encountered errors. You can review the alert log periodically to verify that your database system is operating normally.

To reach the Alerts page, click **Alerts** in the Database Actions page, or click Selector ≡ and select **Alerts** from the Monitoring menu in the navigation pane.

The alert log includes the following:

- Nondefault initialization parameters used at startup
- Administrative operations, such as STARTUP, SHUTDOWN, ARCHIVE LOG, RECOVER, and CREATE/ALTER/ DROP DATABASE/TABLESPACE
- Messages and errors relating to the functions of certain background processes, such as LGWR
- Internal errors (ORA-600), block corruption errors (ORA-1578), and deadlock errors (ORA-60)

Click Refresh ↻ at the top right of the page to refresh the data.

You can search for a specific value in the log by selecting the display column in the first drop-down list, selecting the condition in the second drop-down list, entering the search value in the box, and clicking the search icon.

In the display table, if you right-click the header row, you see:

Columns: Enables you to select columns to show or hide.

Sort: Displays a dialog box for selecting columns to sort by. For each column, you can specify ascending or descending order, and you can specify that null values be displayed first.

If you right-click any other part of the display table, you see:

Count Rows: Displays the number of rows in the table.

Single Record View: Enables you to view data for a table or view, one record at a time.

Copy: Enables you to copy data from a cell or a row or a range of rows. To copy from more than one row, select the rows you want to copy, right-click by pressing the SHIFT or CTRL key, and select **Copy**.

9.6 The Sessions Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Sessions page shows information about all currently open sessions in the database.

To reach the Sessions page, click **Sessions** in the Database Actions page, or click Selector ≡ and select **Sessions** from the Monitoring menu in the navigation pane.

The data is automatically refreshed at intervals ranging from 10 seconds to 2 minutes. You can also refresh the data by clicking Refresh ↻ at the top right of the screen.

The table shows summarized data about each open session. Select a session in the table to see more detailed data in the Session Details table below, such as the last SQL statement, explain plan, waits, contention, and so on. You can use the Column, Operator and Value fields to search for the required sessions.

In the display table, if you right-click the header row, you see:

Columns: Enables you to select columns to show or hide.

Sort: Displays a dialog box for selecting columns to sort by. For each column, you can specify ascending or descending order, and you can specify that null values be displayed first.

If you right-click any other part of the display table, you see:

Count Rows: Displays the number of rows in the table.

Single Record View: Enables you to view data for a table or view, one record at a time.

Copy: Enables you to copy data from a cell or a row or a range of rows. To copy from more than one row, select the rows you want to copy by pressing the SHIFT or CTRL key, right-click and select **Copy**.

9.7 The Storage Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Storage page shows the storage used based on the current allocation of tablespaces along with additional drill-down capabilities to view segments.

To reach the Storage page, click **Storage** in the Database Actions page, or click Selector ≡ and select **Storage** from the Monitoring menu in the navigation pane.

You can refine the list of segments shown by using the filter feature. Click **View Datafiles** to view the datafiles in each tablespace.

You can view tablespace and segment space usage.

To view space usage information

1. From the Database drop-down menu, click **Storage**.

   The Storage page displays. If the Oracle database is version 12c or later, the Storage page shows the used and allocated storage space for tablespaces in any pluggable database. If the Oracle database is version 11g, the Storage page shows the used and allocated space for the entire database.

2. You can click a tablespace to view its storage information. An interactive report appears, showing the segments that exist within the tablespace. Most segments are user objects, and they include tables, LOBs, and indexes.

3. On the Segments page, you can refine the list of segments shown by using the filter feature.

   For example, you can search for all the segments for a specific owner (schema) by selecting OWNER from the first drop-down list, entering the owner (schema) name in the box, and clicking the search icon.

9.8 The Parameters Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Parameters pages displays initialization parameters, which are used to configure the database instance, including memory structures, and define locations for database files.

To reach the Parameters page, click **Parameters** in the Database Actions page, click Selector ≡ and select **Parameters** from the Monitoring menu in the navigation pane.

Values for initialization parameters are stored in a text-based initialization parameter file (PFILE) or binary server parameter file (SPFILE). The initialization parameter file is read at database instance startup.

Click Refresh ↻ at the top right of the page to refresh the data.

To perform a search, enter values in the search criteria columns and click the search icon to locate the initialization parameter.

In the display table, if you right-click the header row, you see:

Columns: Enables you to select columns to show or hide.

Sort: Displays a dialog box for selecting columns to sort by. For each column, you can specify ascending or descending order, and you can specify that null values be displayed first.

If you right-click any other part of the display table, you see:

Count Rows: Displays the number of rows in the table.

Single Record View: Enables you to view data for a table or view, one record at a time.

Copy: Enables you to copy data from a cell or a row or a range of rows. To copy data from more than one row, select the rows you want to copy by pressing Shift or Ctrl, right-click and select **Copy**.

9.9 The Real Time SQL Monitoring Page

Available only if you signed in as a database user with DBA and PDB_DBA roles for Oracle Database 18c and previous releases.

The Real Time SQL Monitoring page shows in real time the SQL statements that are being monitored in the database.

To reach the Real Time SQL Monitor page, click **Real Time SQL Monitor** in the Database Actions page, or click Selector ☰ and select **Real Time SQL Monitor** from the Monitoring menu in the navigation pane.

In the Auto Refresh drop-down list, you can select the time (in seconds) to periodically refresh the data. Select `0 seconds` to disable the auto-refresh.

This tool helps identify run-time issues for SQL statements by providing two major functions:

- General view of monitored statements
- View of SQL execution details

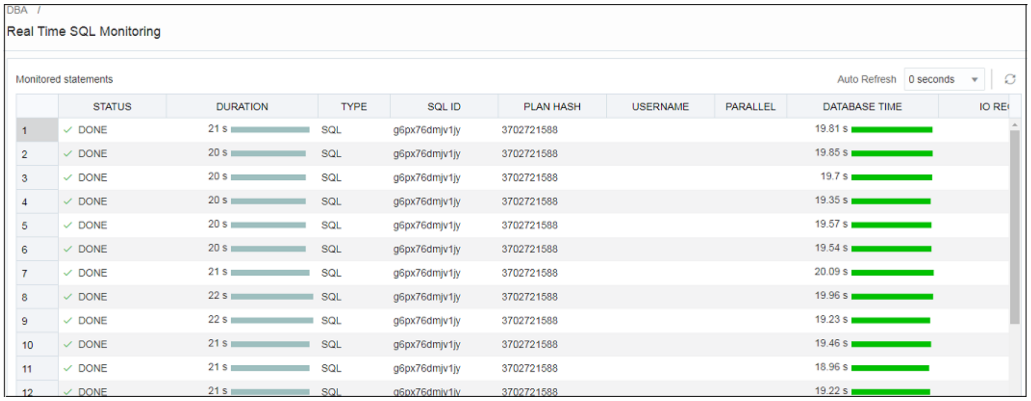
General View of Monitored Statements

The page contains a table of SQL statements currently running. This table shows the following information:

- **Status**: Current state of the SQL statement execution. For example, a SQL statement that has already finished its execution will show a status of "DONE".
- **Duration**: Amount of time a SQL statement is taking, or has taken, to execute.
- **SQL ID**: SQL identifier of the statement being monitored.
- **Session ID**: Session identifier that is executing, or has executed, the SQL statement.
- **Session Serial Number**: Uniquely identifies a session's objects.

- **Instance Degree of Parallelism**: This Degree of Parallelism (DOP) column shows how many instances and parallel execution servers are allocated. It is shown in the form of “number of instances” | “number of parallel servers”.
- **Database Time**: Place the cursor over the database time to see a breakdown of the time and wait events.
- **CPU Time**: CPU time consumed by the execution of the query.
- **I/O Time**: I/O time consumed by the execution of the query.
- **Start Time**: Time in which the execution of the SQL statement started.
- **SQL Statement**: SQL statement being monitored.

Figure 9-4   Real Time SQL Monitoring Table



DBA /

Real Time SQL Monitoring

Monitored statements

Auto Refresh 0 seconds

	STATUS	DURATION	TYPE	SQL ID	PLAN HASH	USERNAME	PARALLEL	DATABASE TIME	IO REI
1	✓ DONE	21 s	SQL	g6px76dmjv1jy	3702721588			19.81 s	
2	✓ DONE	20 s	SQL	g6px76dmjv1jy	3702721588			19.85 s	
3	✓ DONE	20 s	SQL	g6px76dmjv1jy	3702721588			19.7 s	
4	✓ DONE	20 s	SQL	g6px76dmjv1jy	3702721588			19.35 s	
5	✓ DONE	20 s	SQL	g6px76dmjv1jy	3702721588			19.57 s	
6	✓ DONE	20 s	SQL	g6px76dmjv1jy	3702721588			19.54 s	
7	✓ DONE	21 s	SQL	g6px76dmjv1jy	3702721588			20.09 s	
8	✓ DONE	22 s	SQL	g6px76dmjv1jy	3702721588			19.96 s	
9	✓ DONE	22 s	SQL	g6px76dmjv1jy	3702721588			19.23 s	
10	✓ DONE	21 s	SQL	g6px76dmjv1jy	3702721588			19.46 s	
11	✓ DONE	21 s	SQL	g6px76dmjv1jy	3702721588			18.96 s	
12	✓ DONE	21 s	SQL	g6px76dmjv1jy	3702721588			19.22 s	

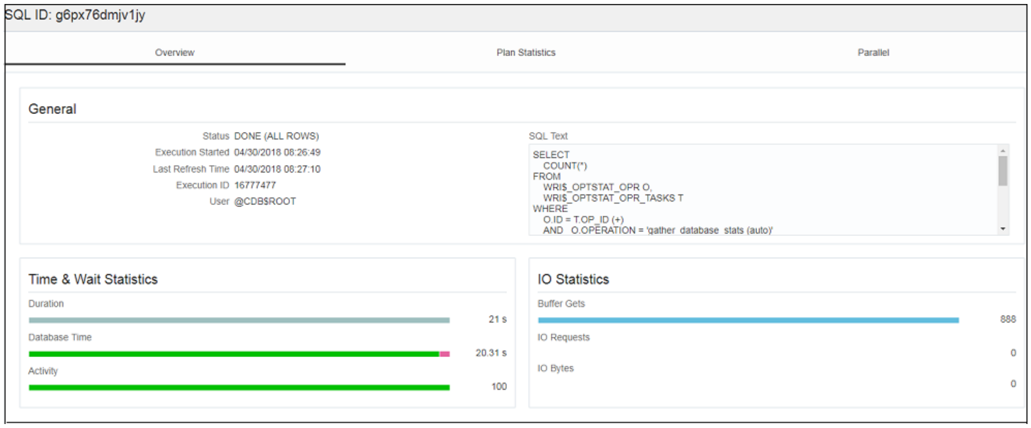
View of SQL Execution Details

When a SQL statement is drilled down from the main monitor table, a detailed view is shown. The SQL ID, Start Time and the SQL Execution ID represent the execution key that uniquely identify this SQL statement. A detail view consists of the general characteristics that integrate the execution of a SQL statement.

General information about the query execution is provided:

- **Execution Plan**: Degree of Parallelism of the SQL statement
- **Execution Started**: Time that the SQL statement execution started
- **Last Refresh Time**: Last update time of the SQL monitor registry for the SQL statement
- **Execution ID**: Execution identifier
- **User**: User in the format USER@CONTAINER
- **SQL Text**: Formatted view of the SQL statement that is being executed.

Figure 9-5   SQL Execution Details in Real Time SQL Monitor



SQL ID: g6px76dmjv1jy

General

Status DONE (ALL ROWS)
Execution Started 04/30/2018 08:26:49
Last Refresh Time 04/30/2018 08:27:10
Execution ID 16777477
User @CDB$ROOT

SQL Text

```
SELECT
   COUNT(*)
FROM
   WRI$_OPTSTAT_OPR O,
   WRI$_OPTSTAT_OPR_TASKS T
WHERE
   O.ID = T.OP_ID (+)
   AND   O.OPERATION = 'gather_database_stats (auto)'
```

Time & Wait Statistics

Duration 21 s
Database Time 20.31 s
Activity 100

IO Statistics

Buffer Gets 888
IO Requests 0
IO Bytes 0

General statistics of the SQL statement are provided: total duration of execution, the number of buffered gets, number of Input/Output requests and bytes.

Detailed information of the statement: This space holds the information corresponding to the explain plan, parallel behavior and CPU activity involved in the execution of the statement:

- **Plan Statistics**: Explain plan of the execution of the SQL statement in the form of a table. Each row is a different operation involved in the execution of the SQL statement and it shows hierarchy dependency by adding a space at the beginning of the text in the Operation column.
- **Parallelism Details for the SQL statement**: Each execution consists of a parallel coordinator and one or more parallel sets. Each set can have one or more processes. When a row has dependents, each of its columns will be the sum of the values of its dependants. When this happens, a sigma symbol will appear to show that a value consists of the sum of others.

> **Note:**
>
> For more information, see "Monitoring the Database" in the *Oracle Database Administrator’s Guide*.

9.10 The Top SQL Page

👁 Available only if you signed in as a database user with DBA and PDB_DBA roles.

The Top SQL page displays SQL statements executed in the database based on CPU time consumed. The SQL statement that consumes the maximum CPU time is right at the top, and the remaining statements continue in descending order of CPU time.

To reach the Top SQL page, click **Top SQL** in the Database Actions page, or click Selector ☰ and select **Top SQL** from the Monitoring menu in the navigation pane.

This feature enables you to focus your SQL tuning efforts on the statements that can have the most impact on database performance.

Click a row to see a formatted view of the SQL statement, the execution plan, the runtime history from Active Session History, and a SQL Tuning Advisor report. Active Session History is part of the Diagnostics Pack, and SQL Tuning Advisor is part of the Tuning Pack for Oracle Database.

9.11 The Waits Page

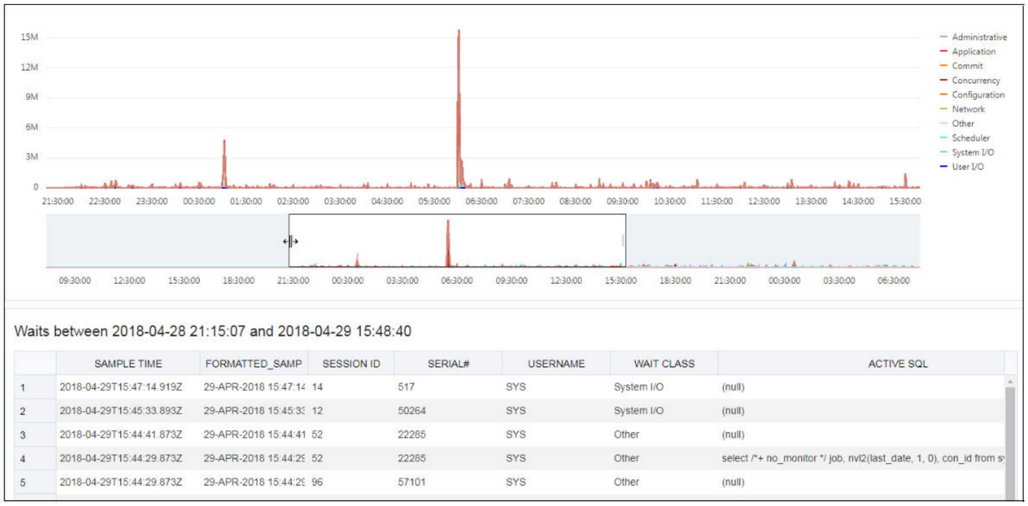
Available only if you signed in as a database user with DBA and PDB_DBA roles for Oracle Database 18c and previous releases.

The Waits page shows a chart with the distribution of wait events in real time.

To reach the Waits page, click **Waits** in the Database Actions page, or click Selector ☰ and select **Waits** from the Monitoring menu in the navigation pane.

Use the slider controls to zoom in on a specific time period. To use the slider controls, place the cursor over the handles at both sides of the box and drag the sides to the time period required. The chart above will refresh to the selected time period. The table will also automatically refresh and the wait events will filter to that period of time enabling you to easily identify the problem SQL statement.

Figure 9-6 Distribution of Wait Events Chart

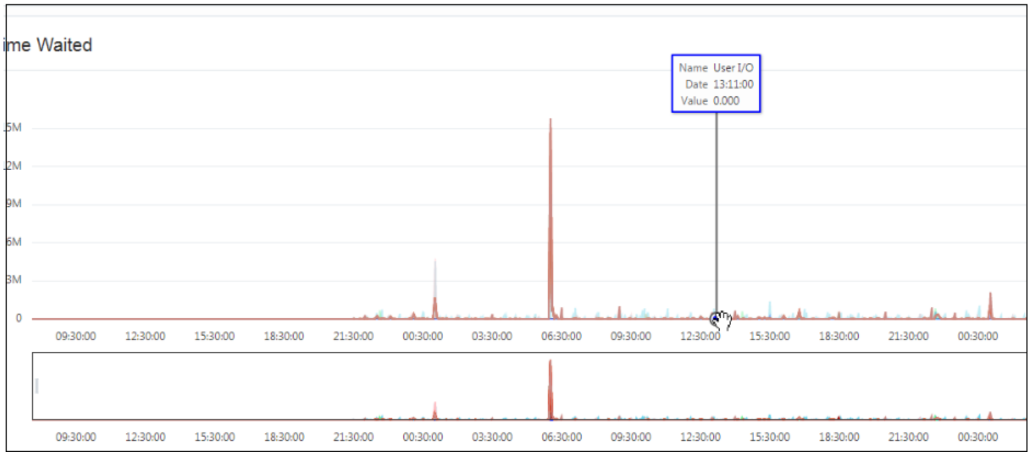


Waits between 2018-04-28 21:15:07 and 2018-04-29 15:48:40

	SAMPLE TIME	FORMATTED_SAMP	SESSION ID	SERIAL#	USERNAME	WAIT CLASS	ACTIVE SQL
1	2018-04-29T15:47:14.919Z	29-APR-2018 15:47:14	14	517	SYS	System I/O	(null)
2	2018-04-29T15:45:33.893Z	29-APR-2018 15:45:3:	12	50264	SYS	System I/O	(null)
3	2018-04-29T15:44:41.873Z	29-APR-2018 15:44:41	52	22285	SYS	Other	(null)
4	2018-04-29T15:44:29.873Z	29-APR-2018 15:44:2!	52	22285	SYS	Other	select /*+ no_monitor */ job, nvl2(last_date, 1, 0), con_id from sy
5	2018-04-29T15:44:29.873Z	29-APR-2018 15:44:2!	96	57101	SYS	Other	(null)

When you place the cursor over data points in the chart, a pop-up box displays details about the wait event.

Figure 9-7   Display Wait Event Details



A
Supported SQL*Plus and SQLcl Commands

This following sections list the SQL*Plus and SQLcl commands supported in the worksheet.

A.1 Supported SQL*Plus Commands

SQL Developer Web supports most of the SQL*Plus commands except those statements that are related to formatting.

- `/ (slash)`
- `@@ { url | file_name[.ext] } [arg ...]`
- `ACC[EPT] variable [NUM[BER] | CHAR | DATE | BINARY_FLOAT | BINARY_DOUBLE] [FOR[MAT] format] [DEF[AULT] default] [PROMPT text | NOPR[OMPT]] [HIDE]`
- `ARCHIVE LOG LIST`
- `BRE[AK] [ON report_element [action [action]]] ...`
- `BTI[TLE] [printspec [text | variable] ...] | [ON | OFF]`
- `C[HANGE] sepchar old [sepchar [new [sepchar]]`
- `CL[EAR] option ...`
- `COL[UMN] [{column | expr} [option ...]]`
- `COMP[UTE] [function [LAB[EL] text] ... OF {expr | column | alias} ...ON {expr | column | alias | REPORT | ROW} ...]`
- `COPY {FROM database | TO database | FROM database TO database} {APPEND | CREATE | INSERT | REPLACE} destination_table[(column, column, column, ...)] USING query`
- `DEF[INE] [variable] | [variable = text]`
- `DESC[RIBE] {[schema.]object[@connect_identifier]}`
- `DISC[ONNECT]`
- `EXEC[UTE] statement`
- `{EXIT | QUIT} [SUCCESS | FAILURE | WARNING | n | variable | :BindVariable] [COMMIT | ROLLBACK]`
- `GET [FILE] file_name[.ext] [LIST | NOLIST]`
- `HO[ST] [command]`
- `L[IST] [n | n m | n * | n LAST | * | * n | * LAST | LAST]`
- `PAU[SE] [text]`

- `PRINT [variable ...]`
- `PRO[MPT] [text]`
- `{QUIT | EXIT} [SUCCESS | FAILURE | WARNING | n | variable | :BindVariable] [COMMIT | ROLLBACK]`
- `R[UN]`
- `SAV[E] [FILE] file_name[.ext] [CRE[ATE] | REP[LACE] | APP[END]]`
- `SET system_variable value`
- `SHO[W] [option]`
- `SHUTDOWN [ABORT | IMMEDIATE | NORMAL | TRANSACTIONAL [LOCAL]]`
- `STA[RT] { url | file_name[.ext] } [arg ...]`
- `STARTUP db_options | cdb_options | upgrade_options`
- `TIMI[NG] [START text | SHOW | STOP]`
- `TTI[TLE] [printspec [text | variable] ...] | [ON | OFF]`
- `UNDEF[INE] variable ...`
- `VAR[IABLE] [variable [type][=value]]`
- `XQUERY xquery_statement`

A.2 Supported SQLcl Commands

SQL Developer Web supports many of the SQLcl commands.

- `ALIAS`
- `APEX`
- `BRIDGE`
- `CTAS`
- `DDL`
- `FORMAT`
- `INFORMATION`
- `LOAD`
- `NET`
- `OERR`
- `RESERVED_WORDS`
- `SCRIPT`
- `SETERRORL`
- `SODA` (See SODA Commands)
- `TNSPING`
- `TOSUB`
- `WHICH`

A.2.1 SODA Commands

Starting from SQL Developer Web release 19.4.2, SODA commands are now supported in SQL Developer Web. SODA allows schemaless application development using the JSON data model. The commands are:

- `SODA create <collection_name>` — Creates a new collection
- `SODA list` — Lists all the collections
- `SODA get <collection_name> [-all | -f | -k | -klist] [{<key> | <k1> <k2> ... | <qbe>}]` — Lists documents in the collection. Optional arguments:
  - `all`: Lists the keys of all documents in the collection
  - `k`: Lists documents matching the specific *<key>*
  - `klist`: Lists documents matching the list of keys
  - `f`: Lists documents matching the *<qbe>*
- `SODA insert <collection_name> <json_str | filename>` — Inserts a new document within a collection
- `SODA drop <collection_name>` — Deletes existing collection
- `SODA count <collection_name> [<qbe>]` — Counts number of documents inside collection. Optional parameter <`qbe`> returns number of matching documents
- `SODA replace <collection_name> <oldkey> <new_{str | doc}>` — Replaces one document with another
- `SODA remove <collection_name> [-k | -klist | -f] {<key> | <k1> <k2> ...| <qbe>}` — Removes documents from collection. Optional arguments:
  - `k`: Removes document in collection matching the specific *<key>*
  - `klist`: Removes document in collection matching the list *<key1> <key2>* ...
  - `f`: Removes document in collection matching *<qbe>*