Oracle® Database

Getting Started with Oracle Developer Tools for VS Code

Release 21.10.0
F93942-02
June 2024

Oracle Database Getting Started with Oracle Developer Tools for VS Code, Release 21.10.0

F93942-02

Primary Authors: Christian Shay, Maitreyee Chaliha

Contents

1 Getting Started with Oracle Developer Tools for VS Code

1

Getting Started with Oracle Developer Tools for VS Code

Welcome! Oracle Developer Tools for VS Code enables Visual Studio Code developers to connect to Oracle Database and Oracle Autonomous Database; edit SQL and PL/SQL with IntelliSense, breadcrumbs, Hover, and Go To/Peek; execute SQL and PL/SQL and view and save the results; and debug PL/SQL using VS Code's native debugging features. The Oracle Database Explorer tree control allows you to quickly explore your database schema, view table data, and edit, debug, execute and save PL/SQL. Create and manage Oracle Autonomous Databases (ADBs) using Oracle Cloud Infrastructure Explorer tree control. Create, start, stop and terminate ADB instances, and automatically create a walletless database connection or download credentials files.

This Quick Start will help you install, connect, configure and get started using Oracle Developer Tools for VS Code.

> **Note:**
>
> This document assumes you are using version 21.10.0 (released May 2024). Follow the instructions for updating in Installing and Updating Oracle Developer Tools for VS Code if you are using an older version.

Installing and Updating Oracle Developer Tools for VS Code

Installing Oracle Developer Tools for VS Code

To install Oracle Developer Tools for VS Code:

1. Upgrade Visual Studio Code to version **1.75** or later. You can check the version by using the **Help->About** menu item. To upgrade select **Help->Check for Updates**.
2. Click on the Extensions icon in the Activity Bar on the side of VS Code or use the **View: Show Extensions** command (Ctrl+Shift+X)
3. Type **Oracle Developer Tools** in the extension search bar.
4. Locate Oracle Developer Tools for VS Code and click **Install**.
5. When the installation is complete, restart Visual Studio Code.

During the installation, the .NET Runtime is now automatically installed if no compatible runtime is currently installed. If there is a problem installing the .NET Runtime, then you may be prompted to manually install it. Installation instructions are as follows:

- Linux
- macOS(x64, Arm64/Apple M1)

- Windows (x64)

Updating this Extension

To be able to update to new versions of this extension, upgrade Visual Studio Code to version **1.75** or later. You can check the version by using the **Help->About** menu item.

- To upgrade to the latest version of Visual Studio Code, select **Help->Check for Updates**.
- If you have disabled automatic extension updates, press F1 to open the Command Palette and enter the **Check for Extension Updates** command and click the **Update...** button which will appear in the list of installed extensions.

> **See Also:**
>
> Extension Marketplace

Creating and Managing Oracle Autonomous Databases (Oracle Cloud)

Follow these instructions to create an Oracle Cloud account, generate the required config file and key files, and connect the **Oracle Cloud Infrastructure Explorer** tree control to the Oracle Cloud. Once the Oracle Cloud Infrastructure Explorer is connected to the Oracle Cloud, you can create and manage Oracle Autonomous Databases:

- To change the compartment or region: Right click on the profile name (usually **DEFAULT**) and select **Change Compartment or Region** from the menu
- To create a new Autonomous Database (ADB), including an Always Free ADB, right click on **Autonomous Transaction Processing Databases**, **Autonomous Data Warehouses**, or **Autonomous JSON Databases** and select **Create New**.
- To view an existing database, expand **Autonomous Transaction Processing Databases**, **Autonomous Data Warehouses**, or **Autonomous JSON Databases** and view the list of databases. Change the region or compartment if your databases are not shown. Database icons with a red ball are not available, those with a yellow ball are starting or stopping, and icons with no ball are available. Icons with a green star are Always Free databases. Icons containing a **D** are dedicated instances.
- To easily and automatically create a connection in Database Explorer, right click on database icon and select **Create Connection in Database Explorer**. For more information, see Connecting Without Using Credentials Files or Connecting Using Oracle Cloud Infrastructure Explorer to Automatically Download Credentials Files (Wallets).
- To download credentials files, right click on database icon and select **Download Credentials Files**.
- To obtain a connection string for this database, right click on database icon and select **Get Connection Strings**. Select the **Authentication** type and **TNS Name (Service Level)** from the drop down lists.
- To configure walletless connectivity and network access right click on database icon and select **Configure Walletless Connectivity and Network Access**. For more information,

see Configuring Walletless Connectivity and Network Access for Oracle Autonomous Databases (Oracle Cloud).

- To Start an ADB, right click on database icon and select **Start**.
- To Stop an ADB, right click on database icon and select **Stop**.
- To Terminate an ADB, right click on database icon and select **Terminate**.
- To change the ADMIN password for an ADB, right click on database icon and select **Change Administrator Password**.
- To refresh the tree control, click on any item and select **Refresh**

Configuring Walletless Connectivity and Network Access for Oracle Autonomous Databases (Oracle Cloud)

To connect to Oracle Autonomous Database without using credentials files (walletless), the database must be configured to allow this and Network Access Control rules should be set up.

- See Creating and Managing Oracle Autonomous Databases (Oracle Cloud) for instructions on how to configure Oracle Cloud Infrastructure Explorer tree control for first use.
- Expand Autonomous Transaction Processing Databases, Autonomous Data Warehouses, or Autonomous JSON Databases and view the list of databases.
- Right click on database icon and select **Configure Walletless Connectivity and Network Access**.
- If you wish to enable walletless access, in the **Network Access Type** drop down list, select **Secure access from allowed IPs and VCNs only (wallet optional)** and check the **Enable walletless connectivity (TLS)** checkbox
- To disable walletless access, select **Secure access from everywhere (wallet required)**.
- To modify Access Control Rules, click the **Update Accesss Control** button.
- In the **Update Accesss Control** dialog, click **Add Access Control Rule** for a new rule, or update an existing rule. Select a **IP Notation Type** and populate the **Values** field, then press **OK**. For example, select a type of **IP address**, then click the **Add my IP Address** Button to populate the field with the public IP address of your system as visible on the public internet.
- Press the **Apply Changes** button in the **Configure Walletless Connectivity and Network Access** dialog. The database will temporarily be offline while the change is being processed.

Exploring Database Schema with Oracle Database Explorer

To view Database Explorer, click the database icon in the Activity Bar on the far left side of Visual Studio Code.

- Connect: Click the plus (+) sign to create a new connection, following the steps in **Connecting to Oracle Database** or **Connecting to Oracle Autonomous Database (ADB)**. Once connection nodes display, you can click on the node to view the database schema.
- Delete Connection: Right click on any connection node and select ***Delete***
- Update Connection info: Right click on any connection node and select ***Update***

- Set a connection to be the default used when opening a SQL or PL/SQL file: Right click on any connection and select **Set as Default Connection**. The icon for the default connection will contain an asterix(*) in the upper left hand corner.
- Unset a connection as a default connection: Right click on the default connection and select **Unset as Default Connection**.
- Disconnect: Right click on any connection node and select ***Disconnect***
- View Other Users/Other Schemas: Click to expand the ***Other Users*** node. Select the schema you wish to view. Alternately, click on the connection node and select ***Update***. In the connection dialog, check the ***Show more options*** checkbox then select a different schema in the ***Current Schema*** dropdown. Press the ***Update Connection*** button.
- View and Save Table/View Data: On a Table and View node's menu, select ***Show Data.*** For more information see **Viewing and Saving Result Sets** below.
- Generate SELECT, INSERT, or DELETE statements for a table: On a Table node, right click and select ***Generate Select SQL, Generate Insert SQL, or Generate Delete SQL*** from the menu. The SQL will be added to an open SQL script file that is selected, or a new SQL file will be opened containing the SQL.
- View metadata for a schema object (describe): On any schema object node, right click and select the ***Describe*** menu item.
- Edit PL/SQL: On stored procedures, functions, packages, or trigger nodes select ***Open/ Open Package Body/Open Specification*** to open PL/SQL into a file for editing. See **Editing PL/SQL in your Database** for more details.
- Save PL/SQL to database: If the PL/SQL was opened with the ***Open*** menu item in Database Explorer, you can save changes to the database by right clicking in the PL/SQL code and selecting ***Save***. If there are errors when saving, they will be listed in the Problems panel, and you can click on an error to go to the line that contains the error. (Note: these menu names were previously named ***Edit PL/SQL*** and ***Save to Database*** in earlier releases.)
- Save PL/SQL to file: In Oracle Database Explorer right click a package, procedure, function, or trigger and from the menu and select ***Download***. Alternatively, after opening the package, procedure, function, or trigger from Database Explorer select the tab with the PL/SQL code in it. In Visual Studio Code menu select ***File->Save As*** and then click ***Show Local***.
- Execute PL/SQL:From menu, select ***Run*** to execute a stored procedure or function
- Debug PL/SQL: Right click on a connection name and select ***PL/SQL Debugger and Compiler Settings*** to set the ip address and port number range that will be used. A script to grant privileges and configure the database is also provided in the dialog. To compile with debug information, right click on a procedure/function/package and select ***Compile Debug*** from the menu. To begin debugging, from the same menu select ***Step Into*** or ***Run Debug*** (if the PL/SQL is already open and breakpoints set). To listen for and debug PL/SQL procedures and functions called by applications or SQL scripts, right click on a connection name and select ***Start External Application Debugger.*** For more details on PL/SQL debugging, see the **Debugging PL/SQL from Oracle Database Explorer** and **Debugging PL/SQL Called By an Application or SQL Scripts** sections.
- Refresh: Right click on any node and select ***Refresh*** to refresh that node and all child nodes
- Filter: Right click on the connection node and select ***Filters*** to open the Filters dialog. See Applying Filters to Oracle Database Explorer and IntelliSense Suggestions.

Connecting to Oracle Database

Connecting Using Host Name/IP Address and Service Name

If your database is in the Oracle Cloud go to **Connecting to Oracle Autonomous Database (ADB)**. For additional help using the connection dialog, visit the connection dialog help.

1. You can connect to the Oracle Database using one of the following ways:
   - To connect to Oracle Database from a .SQL or .PL/SQL file, press F1 to open Command Palette and select **Oracle:Connect** from the dropdown. Then select **New Connection**.

     Or,
   - To connect from Oracle Database Explorer, click the plus (+) sign:
2. A connection dialog will open. In the **Connection Type** dropdown, select **Basic (Host, Port, Service Name)**.
3. Enter the database hostname or IP Address, port number, and service name.
4. Select the database role from the **Role** drop down list.
5. Enter the username and password.
6. If you are using Proxy Authentication, check the **Show more options** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

   ```
   proxyusername[username]/[proxypassword]
   ```

   > **Note:**
   >
   > For more details about connecting with a proxy, see Connecting Using a Proxy User.
7. If you wish to use a different schema than the default schema associated with your username, check the **Show more options** checkbox and select the schema name from the **Current Schema** dropdown.
8. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere.
9. Click the **Create Connection** button.

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Using a TNSNAMES.ORA Connection Alias

If your database is in the Oracle Cloud go to **Connecting to Oracle Autonomous Database (ADB)**. For additional help using the connection dialog, visit the connection dialog help.

1. Copy the TNSNAMES.ORA file that you want to use into the directory that is set as **Config Files Folder** in the Oracle Developer Tools for VS Code Extension Settings, or change this setting as desired. By default this location is **~/Oracle/network/admin** on Linux and Mac and **%USERPROFILE%\Oracle\network\admin** on Windows.
2. If you don't have a TNSNAMES.ORA but would like to create one, see the example located in **~/.vscode/extensions/oracle.oracledevtools-21.7.1/sample/network** on Linux and Mac and **%USERPROFILE% \.vscode\extensions\oracle.oracledevtools-21.7.1\sample\network** on Windows .
3. You can connect to the Oracle Database using one of the following ways:
   - To connect to Oracle Database from a .SQL or .PL/SQL file, press F1 to open Command Palette and select **Oracle:Connect** from the dropdown. Then select **New Connection**.

     Or,
   - To connect from Oracle Database Explorer, click the plus (+) sign:
4. A connection dialog will open. In the **Connection Type** dropdown, select **Use TNSNAMES.ORA**.
5. Make sure the **TNS Admin Location** field is set to the directory where your TNSNAMES.ORA file is located. If not, change it.
6. Select an alias from the **TNS Alias** dropdown list.
7. Select the database role from the **Role** drop down list.
8. Enter the username and password.
9. If you are using Proxy Authentication, check the **Show more options** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

   ```
   proxyusername[username]/[proxypassword]
   ```

   > **Note:**
   >
   > For more details about connecting with a proxy, see Connecting Using a Proxy User.
10. If you wish to use a different schema than the default schema associated with your username, check the **Show more options** checkbox and select the schema name from the **Current Schema** dropdown.
11. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere.
12. Click the **Create Connection** button.

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Using a Proxy User

If your database is in the Oracle Cloud go to **Connecting to Oracle Autonomous Database (ADB)**. For additional help using the connection dialog, visit the connection dialog help.

- Follow the steps in Connecting to Oracle Database to open the connection dialog and provide the connection information.
- If you are using Proxy Authentication, check the ***Show more options*** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

  ```
  proxyusername[username]/[proxypassword]
  ```

> **Note:**
>
> See the connection dialog help for more information about other proxy scenarios such as using external authentication (that is, OS/Kerberos/Certificate) or if using Secure External Password Store (SEPS) wallet for database username and password.

Connecting Using a Directory Server (LDAP)

If your database is in the Oracle Cloud go to **Connecting to Oracle Autonomous Database (ADB)**. For additional help using the connection dialog, visit the connection dialog help.

Connections using a directory server (LDAP) are now supported on all platforms (Windows, MacOS, and Linux).

1. Copy the LDAP.ORA and SQLNET.ORA files that you want to use into the directory that is set as ***Config Files Folder*** in the Oracle Developer Tools for VS Code Extension Settings, or change this setting as desired. By default this location is ***~/Oracle/network/admin*** on Linux and Mac and ***%USERPROFILE%\Oracle\network\admin*** on Windows.
2. To connect to Oracle Database from a .SQL or .PL/SQL file, press F1 to open Command Palette and select ***Oracle:Connect*** from the dropdown. Then select ***New Connection***.
3. To connect from Oracle Database Explorer, click the plus sign button
4. A connection dialog will open. In the ***Connection Type*** dropdown, select ***LDAP Directory Server***
5. Make sure the ***TNS Admin Location*** field is set to the directory where your LDAP.ORA and SQLNET.ORA files are located. If not, change it.
6. In the ***TNS Alias*** field, type in the database service name.
7. Select the database role from the ***Role*** drop down list

8. Enter the username and password
9. If you are using Proxy Authentication, check the ***Show more options*** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

   ```
   proxyusername[username]/[proxypassword]
   ```

   > **Note:**
   >
   > For more details about connecting with a proxy, see Connecting Using a Proxy User.

10. If you wish to use a different schema than the default schema associated with your username, check the ***Show more options*** checkbox and select the schema name from the ***Current Schema*** dropdown
11. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere
12. Click the ***Create Connection*** button

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information, see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Using OS Authentication

If your database is in the Oracle Cloud go to **Connecting to Oracle Autonomous Database (ADB)**. For additional help using the connection dialog, visit the connection dialog help.

If using both Windows Client and Windows Database Server, set **SQLNET.AUTHENTICATION_SERVICES=(NTS)** in a SQLNET.ORA file located in the directory that is set as ***Config Files Folder*** in the Oracle Developer Tools for VS Code Extension Settings. By default this location is ***%USERPROFILE%\Oracle\network\admin***

Follow the steps in **Connecting Using Host Name/IP Address and Service Name** or **Connecting Using a TNSNAMES.ORA Connection Alias** above, except

- In the ***User name*** field, enter a forward slash: **/**
- The ***Password*** field should be left empty.

> **Note:**
>
> For more details about connecting with a proxy, see Connecting Using a Proxy User.

Connecting Using Kerberos Authentication

If your database is in the Oracle Cloud go to **Connecting to Oracle Autonomous Database (ADB)**. For additional help using the connection dialog, visit the connection dialog help.

Before connecting using Kerberos, server and client side configuration is required.

For instructions on how to configure Kerberos, see sections File Based Credential Cache and MSLSA and Configuring Kerberos Authentication with ODP.NET in the *Oracle Data Provider for .NET Developer's Guide*.

> **Note:**
>
> - Oracle Developer Tools for VS Code includes Kerberos.NET. MIT Kerberos does not need to be installed.
> - If using File Based Credential Cache instead of MSLSA, the full Oracle client will need to be installed on the system where the credential file is generated in order to obtain the executables okinit.exe, oklist.exe and okdstry.exe.
>
> The full Oracle client can be found here:
>
> – Windows: Oracle Database 21c (21.3) for Microsoft Windows x64 (64-bit)
>
> – Linux: Oracle Database 21c (21.3) for Linux x86-64 (RPM)
> - The instructions in the link above assume a Windows KDC. If using a Linux KDC, please see Extract a Service Key Table from Kerberos

To connect to the Oracle Database using Kerberos Authentication, follow the steps in Connecting Using a TNSNAMES.ORA Connection Alias, except

1. Ensure the `SQLNET.ORA` file is in the same directory as the `TNSNAMES.ORA`
2. In the **User name** field, enter a forward slash: **/**
3. The **Password** field should be left empty.

> **Note:**
>
> For more details about connecting with a proxy, see the connection dialog help.

Connecting to Oracle Autonomous Database (ADB)

Connecting Using Oracle Cloud Infrastructure Explorer to Connect Without Using Credentials Files (Walletless)

If your Oracle Database is On-prem, then go to **Connecting to Oracle Database**. For additional help using the connection dialog, visit the connection dialog help.

Oracle Cloud Infrastructure Explorer makes connecting to your Autonomous Database easier by automatically obtaining the connection string pre-populating the connection dialog.

Follow the steps in the Creating and Managing Oracle Autonomous Databases (Oracle Cloud) and Configuring Walletless Connectivity and Network Access for Oracle Autonomous Databases (Oracle Cloud) to connect Oracle Cloud Infrastructure Explorer tree control to the Oracle Cloud, locate the Autonomous Database instance you wish to connect to and configure the Autonomous database for walletless access if necessary.

1. Right click on the Autonomous Database instance name and select **Create Connection in Database Explorer**. The ***Select Authentication and Connection String*** will open. If this dialog does not open but instead a ***Download Credentials Files*** dialog opens, this means that the database is not configured for walletless access.
2. Set Authentication to **TLS(walletless)**, select a choice from the ***TNS Name (Service Level)*** drop down, and press ***OK***.

   The connection dialog will open.
3. Select **Default** from the Role drop down list
4. Enter the username and password. (If you are new to Oracle Autonomous Database, use username ADMIN).
5. If you are using Proxy Authentication, check the **Show more options** checkbox and provide the proxy username and password.

   > **Note:**
   >
   > For more details about connecting with a proxy, see Connecting Using a Proxy User.

6. If you want to use a different schema than the default schema associated with your username, check the **Show more options** checkbox and select the schema from the Current Schema dropdown.
7. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere.
8. Click the **Create Connection** button.
9. If you are using macOS, and receive a Connection request timed out error when trying to connect, please update .NET Runtime to version 6.0.

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Using Oracle Cloud Infrastructure Explorer to Automatically Download Credentials Files (Wallets)

If your Oracle Database is On-prem, then go to **Connecting to Oracle Database**. For additional help using the connection dialog, visit the connection dialog help.

Follow the steps in **Create and Manage Oracle Autonomous Databases** to connect Oracle Cloud Infrastructure Explorer tree control to the Oracle Cloud and locate the Autonomous Database instance you wish to connect to.

Oracle Cloud Infrastructure Explorer makes connecting to your Autonomous Database easier by automatically downloading credentials files if needed, and by pre-populating the connection dialog.

1. Right click on the Autonomous Database instance name and select ***Create Connection in Database Explorer***. Depending on hour your database is configured, you may see a ***Select Authentication and Connection String*** dialog. If you do, Set **Authentication** to `TLS(walletless)`, select a choice from the ***TNS Name (Service Level)*** drop down, and press ***OK***.
2. When the Download Credentials Files dialog opens, verify the file path and press OK. A prepopulated connection dialog will open.
3. Select the alias name you wish to connect to, for example ***mydb_high***, from the ***TNS Alias*** dropdown list.
4. Select ***Default*** from the ***Role*** drop down list
5. Enter the username and password. (If you are new to Oracle Autonomous Database, use username ***ADMIN***)
6. If you are using Proxy Authentication, check the ***Show more options*** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

   ```
   proxyusername[username]/[proxypassword]
   ```

   > **Note:**
   >
   > For more details about connecting with a proxy, see Connecting Using a Proxy User.

7. If you want to use a different schema than the default schema associated with your username, check the **Show more options** checkbox and select the schema from the **Current Schema** dropdown.
8. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere
9. Click the ***Create Connection*** button
10. If you are using macOS, and receive a ***Connection request timed out*** error when trying to connect, please update .NET Runtime to version 6.0.

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Using Credentials Files Obtained Elsewhere

If your Oracle Database is On-prem, then go to **Connecting to Oracle Database**. For additional help using the connection dialog, visit the connection dialog help.

The easiest way to connect to Autonomous Database is to use Oracle Cloud Infrastructure Explorer . Follow the steps described in **Connecting Using Oracle Cloud Infrastructure Explorer to Automatically Download Credentials Files (Wallets)** . However, if you have obtained the credentials files from another source, such as the Administration Console or from your administrator, follow these steps:

1. Unzip the credentials files into the directory that is set as ***Config Files Folder*** in the Oracle Developer Tools for VS Code Extension Settings. By default this location is ***~/Oracle/network/admin*** on Linux and Mac and ***%USERPROFILE%\Oracle\network\admin*** on Windows.
2. To connect from Oracle Database Explorer, click the plus sign button
3. To connect to Oracle Database from a .SQL or .PL/SQL file, press F1 to open Command Palette and select ***Oracle:Connect*** from the dropdown. Then select ***New Connection***.
4. A connection dialog will open. In the ***Connection Type*** dropdown, select ***TNS Alias***
5. Make sure the ***TNS Admin Location*** field is set to the directory containing your credentials files. If not, change it.
6. Check the ***Use Wallet File*** checkbox
7. Make sure the ***Wallet File Location*** field is set to the directory containing your credentials files. If not, change it.
8. Select the alias name you wish to connect to, for example ***mydb_high***, from the ***TNS Alias*** dropdown list.
9. Select ***Default*** from the ***Role*** drop down list
10. Enter the username and password. (If you are new to Oracle Autonomous Database, use username ***ADMIN***)
11. If you are using Proxy Authentication, check the ***Show more options*** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

    ```
    proxyusername[username]/[proxypassword]
    ```

    > **Note:**
    >
    > For more details about connecting with a proxy, see Connecting Using a Proxy User.

12. If you want to use a different schema than the default schema associated with your username, then check the ***Show more options*** checkbox and select the schema from the **Current Schema** dropdown.
13. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere
14. Click the ***Create Connection*** button
15. If you are using macOS, and receive a ***Connection request timed out*** error when trying to connect, please update .NET Runtime to version 6.0.

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Without Using Credentials Files

If your Oracle Database is On-prem, then go to **Connecting to Oracle Database**. For additional help using the connection dialog, visit the connection dialog help.

The easiest way to connect to Autonomous Database is to use Oracle Cloud Infrastructure Explorer. See Connecting Using Oracle Cloud Infrastructure Explorer to Connect Without Using Credentials Files (Walletless). However, if you have obtained the connection string from another source, such as the Administration Console or from your administrator, and the database is configured for walletless connectivity, perform the following steps.

1. You can connect to the Oracle Database using one of the following ways:
   - To connect to Oracle Database from a .SQL or .PL/SQL file, press F1 to open Command Palette and select **Oracle:Connect** from the dropdown. Then select **New Connection**.

     Or,
   - To connect from Oracle Database Explorer, click the plus (+) sign:
2. A connection dialog will open. In the ***Connection Type*** dropdown, select ***Advanced***
3. Paste the connect string into the ***Connect String*** box.
4. Select ***Default*** from the ***Role*** drop down list
5. Enter the username and password. (If you are new to Oracle Autonomous Database, use username ***ADMIN***)
6. If you are using Proxy Authentication, check the ***Show more options*** checkbox and provide the proxy username and password. To guide you in entering the fields, note that in other tools, you may have connected using this format:

   ```
   proxyusername[username]/[proxypassword]
   ```

> **Note:**
>
> For more details about connecting with a proxy, see Connecting Using a Proxy User.

7. If you want to use a different schema than the default schema associated with your username, then check the ***Show more options*** checkbox and select the schema from the ***Current Schema*** dropdown
8. Provide a connection name to be used to reference this connection in Database Explorer and elsewhere
9. Click the ***Create Connection*** button
10. For information on how to enable walletless connectivity from Visual Studio Code, go to Configuring Walletless Connectivity and Network Access for Oracle Autonomous Databases (Oracle Cloud).

> **Note:**
>
> Connection information and settings may be stored with a User, Workspace, or Folder Scope. For more information see Organizing Connections by User, Workspace, or Folder Scope.

Connecting Using OCI IAM Single Sign-on

If your Oracle Database is On-prem, then go to **Connecting to Oracle Database**. For additional help using the connection dialog, visit the connection dialog help.

> **Note:**
>
> This connection method requires that your administrator has enabled OCI IAM Single Sign-on with ADB. You must also have installed .NET Runtime version 5.0 or later.

- Follow the instructions in the following documentation section to use OCI CLI to create a token, and to modify your sqlnet.ora and/or tnsnames.ora to use that token: Configuring a Client Connection for SQL*Plus That Uses an IAM Token. Disregard the portions of those instructions specific to SQL*Plus.
- Follow the steps from the quickstart sections in Connecting to Oracle Autonomous Database (ADB) as applicable, except:
  - In the ***User name*** field, enter a forward slash "**/**" (without the quotes)
  - The ***Password*** field should be left empty.

Organizing Connections by User, Workspace, or Folder Scope

Workspace and Folder scoped connection information and settings can be stored in a project along with the code for that project. This can also allow multiple Visual Studio Code users to share connection information via source controlled projects.

Connection information and settings can be stored at a user scope in the Visual Studio Code `settings.json` file for the user. If a workspace is currently open, then it can also be stored at a Workspace scope in a settings file in the workspace. And it can also be stored at a Workspace Folder scope in a settings file in a folder in the workspace.

Connections created with a workspace or folder scope will appear in Oracle Database Explorer when the Visual Studio Code workspace is open. When the workspace is closed or another workspace opened, those connections in Oracle Database Explorer will disappear.

For more information about Visual Studio Code User and Workspace settings see:

https://code.visualstudio.com/docs/getstarted/settings

To set the scope of a connection:

1. Create a connection in Oracle Database Explorer by clicking the plus (**+**) sign to create a new connection.
2. Select **User**, **Workspace**, or **Folder** tab at the top of the connection dialog. The **Workspace** scope will only be visible if a workspace is opened in Visual Studio Code. If the workspace contains folders, the folder names will be selectable from the **Folders** dropdown.

> **Note:**
>
> The scope of the connection can only be set when it is being created. It cannot be modified later. For example, you cannot modify it later using the **Update** Connection menu item.

3. Check **Append 'workspace' to connection name** or **Append folder name to connection name** if you wish the workspace or folder name to be included alongside the name of the connection node for easier identification.
4. Continue to populate the connection dialog, following the steps in Connecting to Oracle Database or Connecting to Oracle Autonomous Database (ADB).

> **Note:**
>
> Passwords cannot be saved for connections with Workspace or Folder scope.

Applying Filters to Oracle Database Explorer and IntelliSense Suggestions

You can apply filters to limit the amount of database schema objects that are shown in the Oracle Database Explorer tree control and that are offered as IntelliSense (autocomplete) suggestions.

Filters are applied on existing Oracle Database Explorer connections. To create a filter, follow these steps:

1. If needed, first create a connection by clicking the plus (+) sign to create a new connection, and following the steps in Connecting to Oracle Database or Connecting to Oracle Autonomous Database (ADB).
2. Right click on the connection node and select **Filters** to open the Filters dialog.
3. From the list on the left side of the dialog, select the filter type, for example **Tables** or **Procedures**. If you wish to create a filter that applies to all object types, then select the **Connection** filter type.
4. Under **Filter Properties** select **Match Any** or **Match All**. **Match Any** means that any filter property, if evaluated to be true, results in the object being displayed or suggested by IntelliSense (autocomplete). **Match All** means that all Filter Properties must be evaluated to be true for the object to be displayed or suggested in IntelliSense (autocomplete).
5. If no Filter Properties are shown click **Add Filter** to begin adding one.
6. Select the **Property** and **Condition** drop down boxes and enter a value in the **Value** field. If the value is a string, check the **Case Sensitive** check box if the value should be used exactly as entered (without being uppercased first).
7. Once you have entered the value, click the **Add Filter** button if you wish to add add additional Filter Properties.
8. To delete filter properties, click the **X** on the right of the filter property.
9. Check or uncheck the checkboxes labelled **Enable filters for database explorer** and **Enable filters for intellisense** as desired. You can enable/disable all of your filters across all object types for database explorer, intellisense or both by going to the top of the dialog and clicking the **Enable/Disable All Filters** dropdown and selecting one of the options.
10. If you wish this filter to override any connection filter, then check the **Override filter properties checkbox specified at the Connection level**.
11. When you are finished, save your work by pressing the **Save** button.
12. A Filter icon (funnel) will appear next to Filter Types shown on the left side of the Filters dialog if the Filter Type has an enabled filter on it.
13. Icons in Oracle Database Explorer will change to include a small funnel image to indicate that a filter is being applied
14. To permanently delete all filters and start over fresh, click the **Delete All Filters** button near the top right portion of this filters dialog.

Changing the Default Schema for a Connection

To use a different schema than the default for the user you logged in as (for example, you logged in as ADMIN but want to default to using the HR schema when executing SQL or when browsing with Oracle Explorer), follow the steps in **Connecting to Oracle Database** or **Connecting to Oracle Autonomous Database (ADB)** to open the connection dialog and provide the connection information.

- In the connection dialog, check the ***Show more options*** checkbox and then select a different schema in the ***Current Schema*** dropdown.

Selecting a Default Connection to be used when Opening SQL files

You can select a connection in Oracle Database Explorer to be the default connection. This connection will be used when a SQL file is opened that is not explicitly associated with a connection. For example, when opening folders containing many SQL scripts this will associate the connection with all of them, saving you from having to associate each file with a connection.

To set a connection as default (replacing any existing default connection):

1. Create a connection in Oracle Explorer. See Connecting to Oracle Database.
2. Right click on the connection and choose **Set as Default Connection**. The icon will change to indicate that the connection is now default.
3. Alternatively, in the connection dialog, check the **Set as default connection** check box.
4. To unset the default connection (so that there is no default), right click on the connection and choose **Unset as Default Connection**.

Running a SQL Script Each Time a Connection is Established

Each time a connection is established, you may wish to run a SQL script (login SQL script). This could be used to alter session settings to modify NLS parameters or anything else. A login SQL script may be used for all connections, and it can be overridden on a per connection basis.

- To set a login SQL script for all connections, go to the Extension Settings for Oracle Developer Tools for VS Code and set the **Connection Configuration: Login Script** value to the location of a sql script. For example, `/myscripts/login.sql`. Leave this setting blank if you only wish a log script to run for certain connections.
- To set a login SQL script for a particular connection in Oracle Explorer, open the connection dialog. For example, select the connection and choose the **Update** menu item. In the connection dialog, check the **Show more options** checkbox and then populate the **Login Script** field. This script will override any login script set in the extension settings.
- To view any output of the login SQL script, view the Results tab.

Colorizing Oracle Explorer Connections and Code Windows

You can colorize Oracle Database Explorer connection node names and their corresponding code window to make it easier to identify which code window is using which database connection.

To set a color for a connection in Oracle Database Explorer, open the connection dialog. For example, select an existing connection and choose the **Update** menu item, or click the Plus (**+**) icon to open a new connection.

In the connection dialog, click the **Color** button next to the **Connection Name** field. Select a color from the picker dialog and press the **Done** button. If you do not wish to use color you can click the **No Color** button in the picker dialog. Press the **Update Connection** or **Create Connection** on the connection dialog to save the changes.

Viewing other Schemas in Oracle Explorer

- In Oracle Explorer, click to expand the ***Other Users*** node. Select the schema you wish to view.
- If you will mostly be using the alternate schema and wish it to be the default schema shown in Oracle Explorer, follow the steps in **Changing the Default Schema for a Connection**.

Editing PL/SQL in your Database

- View Database Explorer by clicking the database icon in the Activity Bar on the far left side of Visual Studio Code.
- If the Database Explorer pane is empty (no connection nodes showing), click the plus (+) sign to create a new connection, following the steps in **Connecting to Oracle Database** or **Connecting to Oracle Autonomous Database (ADB)**. Once connection nodes display, you can click on the node to view the database schema.
- Navigate in the tree control to the PL/SQL package, stored procedure/function, or trigger you want to edit.
- Right click on the PL/SQL package, stored procedure/function, or trigger and select ***Open/ Open Package Body/Open Specification*** to open PL/SQL for editing. (Note: this menu item was previously named ***Edit/Edit Package Body/Edit Specification*** in earlier releases.)
- When done editing, right click in the PL/SQL code and select ***Save***. (Note: this menu item was previously named ***Save to Database*** in earlier releases.)
- Errors that occur when saving will be listed in the Problems Panel. Click on an error to go to the line that contains the error.
- If you are using Visual Studio Code Auto Save, we recommend that you follow the steps in **Saving PL/SQL in Database to a File**. You can use ***Download*** instead of ***Open***, or adjust the VS Code ***Auto Save Delay*** setting.

Saving PL/SQL in Database to a File

- In Oracle Database Explorer you can right click a package, procedure, function, or trigger and from the menu and select ***Download/Download Package Body/Download Package Specification***.
- To change the default directory where the file is saved, go to the settings and change ***Download: Folder Type*** to ***Other Folder***. Then, change the setting ***Download:Other Folder*** to the path to the folder you wish downloads to be saved in.
- Alternatively, after opening the package, procedure, function, or trigger from Database Explorer select the tab with the PL/SQL code in it. In Visual Studio Code menu select ***File->Save As*** and then click ***Show Local***.

Opening .SQL or .PL/SQL File

- In Oracle Explorer, right click on a connection node and select ***Open Existing SQL File***.
- If a connection does not exist, create a new one by pressing the plus (+) icon and using the steps in **Connecting to Oracle Database** or **Connecting to Oracle Autonomous Database (ADB)**.

Changing Database Connection for an Open .SQL or .PL/SQL File

- Press F1 to open Command Palette and select ***Oracle:Update Connection*** from the dropdown.
- Select an existing connection profile from the list or create a new one using the steps in **Connecting to Oracle Database** or **Connecting to Oracle Autonomous Database (ADB)**.

Creating a New .SQL or .PLSQL File

- In Oracle Explorer, right click on a existing connection node and select ***Open New SQL File***. If a connection does not exist, create a new one by pressing the plus (+) icon and using the steps in **Connecting to Oracle Database** or **Connecting to Oracle Autonomous Database (ADB)**.
- Alternately, Press F1 to open Command Palette and select ***Oracle:Develop New SQL or PLSQL*** from the dropdown. Select an existing connection profile from the list or create a new one
- When done editing, in the VS Code menu select ***File->Save As*** to save the file.
- Alternately, you may execute the SQL*Plus **CONNECT** command. This command will associate the file with the connection specified in the command.

Using Auto Save when Editing PL/SQL

- Visual Studio Code Auto Save is enabled by going to the VS Code menu and selecting ***File->Auto Save***.
- We recommend that you do not use the ***Open*** menu item in Database Explorer when Auto Save is enabled as it results in very frequent updates to the database. Instead, use the ***Download*** menu item as described in the **Saving PL/SQL in Database to a File** section.
- Alternatively, you can adjust the VS Code ***Auto Save Delay*** setting to a larger value.

Executing SQL and PL/SQL

1. Type some SQL or PL/SQL you wish to execute into the .SQL or .PLSQL file
2. While typing, autocomplete suggestions will appear as you type, for example column names. You can click the ***i*** icon in the suggestions to get more details such as the schema objects the suggestion is associated with. Use the arrow keys to navigate through the suggestions, and the enter key to select a suggestion. If you wish to use lower case characters with autocomplete suggestions, go to the settings and change ***Intellisense › Suggestions: Keyword Casing*** and ***Intellisense › Suggestions: Object Name Casing*** to ***lowercase***.
3. Type a schema name followed by a period, for example ***HR.*** to use intellisense to view and select database objects.
4. While typing, some code snippet suggestions may appear. You can view more snippets by typing ***oracle*** on a new line.
5. Position the cursor on a line that contains the SQL or PL/SQL that you wish to execute. Alternately, you can select (highlight) one or more SQL or PL/SQL statements. Right click and select ***Execute SQL*** from the menu. Select ***Execute All*** if you wish to execute all SQL and PL/SQL in the current file.
6. After executing SQL, a new document (Results Window) will open or the results will be appended to an open Results window. The maximum number of rows that can be returned is controlled by the ***Max Rows*** extension setting. As you scroll through the rows, more will be fetched - up to this maximum.
7. To clear the results window, click the ***Clear Results Window*** button located in the area near the document tabs. To change the default to always clear the results window after every execution, change the ***Clear Results Window*** extension setting.
    - Errors that occur when running the script will be listed in the Problems Panel. Click on an error to go to the line that contains the error.
    - By default, SQL statements will automatically commit (auto commit is on). To change this, see **Disable/Enable Auto Commit** in the section below.
    - In addition to SQL and PL/SQL, you can enter SQL*Plus commands as well. For more information see the **Executing SQL*Plus Commands** section and visit Using SQL*Plus Commands with Oracle Developer Tools for VS Code

Using Select AI: Natural Language to SQL

About Select AI

Select AI is a feature of Oracle Autonomous Database that enables you to query your data using natural language. Select AI uses generative AI with Large Language Models (LLMs) to convert a user's input text into Oracle SQL. Select AI processes the natural language prompt, supplements the prompt with metadata from your database, and then generates and (optionally) runs a SQL query.

Configuring Select AI for First Use

Perform the following steps to check the requirements and configure Select AI for first use.

1. Select AI requires the following:
   - Access to an Autonomous Database instance, such as an Always Free Autonomous Database
   - A paid API account for a supported AI provider (currently either Cohere or OpenAI).
2. The Select AI feature requires that the database allow network access to an AI Provider for a database user. The database user also needs execute privileges on the `DBMS_CLOUD` and `DBMS_CLOUD_AI` packages.

   Perform the following steps:

   a. In Database Explorer, Connect to the Autonomous Database as the `ADMIN` user. See Connecting to Oracle Autonomous Database (ADB).

   b. Right click on the connection in Database Explorer and select **Configure Select AI Provider Network Access**.

   c. Choose your **AI Provider** and the **Database Username** from the drop down lists and click the **Save** button.

   If you do not have access to the `ADMIN` account, then you can connect to your database user account and perform the preceding steps. Instead of clicking **Save**, check the **show SQL** checkbox and provide the SQL to the person with access to the `ADMIN` user.
3. You will need to create a credential that stores the API key from your AI provider for use by Oracle Database. The API key is obtained by logging into the AI provider's web site. You can create more than one credential.

   Perform the following steps to create a new credential:

   a. In Database Explorer, Connect to the Autonomous Database as your database user. See Connecting to Oracle Autonomous Database (ADB).

   b. Right click on the connection in Database Explorer and select **Configure Select AI Profile**

   c. Go to the **Manage Credential** tab.

   d. Enter the API Key.

   e. Click **Save** to create the credential.

> **Note:**
>
> You can only create a new credential. An error will be raised if you provide an existing credential name.

4. You will need to create a profile that specifies which AI provider, credential, and model will be used and which database schema object metadata will be provided to the LLM. You may have more than one profile.
   a. In Database Explorer, right click on the connection in Database Explorer and select **Configure Select AI Profile**.
   b. Under the **Manage Profile** tab, enter a profile name.

   > **Note:**
   >
   > You can only create a new profile. An error will be raised if you provide an existing profile name.

   c. Choose a **Credential Name** from the drop down list.
   d. Select the **AI Provider** from the dropdown list
   e. Select the **Model** from the dropdown list
   f. Click on the button with three dots next to the **Object List** field. The **Object List** dialog will open.
   g. In the **Object List** dialog, select the **Schema**.
   h. Using the radio button choose to show either **Tables** or **Views**.
   i. Choose the tables or views for which you would like to send metadata to the LLM.
   j. Cick **OK** to accept the Object List.
   k. Click **Save** to create the profile.

> **See Also:**
>
> For more information about configuring Select AI, see:
>
> - Use Select AI to Generate SQL from Natural Language Prompts
> - DBMS_CLOUD_AI Package

Setting an AI Profile on a Database Connection

You must associate one of the AI Profiles that you have created with the Oracle Database connection that you wish to use with Select AI.

1. In Database Explorer, connect to the Autonomous Database as your database user. See Connecting to Oracle Autonomous Database (ADB).
2. In the connection dialog, optionally check **Show More Options**, and then select a profile from the **Default AI Profile** dropdown.

3. Right click on the connection and select **Open New SQL File**, or **Open Existing SQL File**.
4. In the SQL file, if you did not choose a **Default AI Profile** in the connection dialog, then run this PL/SQL block (you will need to do this each time you connect):

```
begin
 dbms_cloud_ai.SET_PROFILE('yourprofilename');
end;
/
```

Translating and Executing Natural Language Queries

Once your AI Profile has been set on a connection, you can execute natural language queries by executing a statement such as:

```
SELECT AI <natural language query>
```

For example (if using the HR schema):

```
SELECT AI What are the departments and where are they located;

SELECT AI Rank the departments by how many employees work in each department;

SELECT AI Who works in the IT department;
```

You may wish to follow a workflow where you first translate the natural language to SQL, inspect it, and (optionally) edit the SQL before executing it.

1. Place the natural language in a commented line (without including the ***SELECT AI*** prefix)

   For example:

   ```
   -- What are the departments and where are they located

   -- Rank the departments by how many employees work in each department;

   -- Who works in the IT department;
   ```

2. Place the cursor at the end of the line that contains the commented natural language query
3. Right click and choose **Select AI->Translate to SQL** from the menu. Alernatively, you may use the keyboard shortcut or the Select AI toolbar icon.

   The translated SQL will appear.
4. To keep the SQL in the file, press the tab key. Otherwise it will disappear when you press any key or click elsewhere.
5. Edit the SQL if desired and then execute it.
6. To see an explanation of the SQL, place the cursor at the end of the generated SQL and choose **Select AI ->Explain** from the menu.
7. To generate both the translated SQL and an explanation, place the cursor at the end of the commented line that contains the natural language query and choose **Translate to SQL and Explain** from the menu.

Viewing an Explain Plan/Execution Plan for a SQL Statement

You can view an Explain Plan for a SQL Statement in the code window to see the predicted plan for its execution. Similarly, you can view the Execution Plan after SQL has already been executed to see the actual plan used.

Both grid based and text based plan output are available and are configurable using the settings.

To view an Explain Plan for SQL you have entered into the code window, perform the following steps:

1. Position the cursor on the line that contains the SQL and press the **Explain Plan (grid)** or the **Explain Plan (text)** icon in the toolbar. Alternatively, right click and select **Explain Plan (grid)** or the **Explain Plan (text)** from the menu.
2. To configure the results of Explain Plan, right click on the connection node and select **Explain Plan and Execution Plan Settings**

> **Note:**
>
> Explain Plan (grid) uses `PLAN_TABLE` while Explain Plan (text) uses the `DBMS_XPLAN` PL/SQL package.

To view an Execution Plan for SQL you have executed in the code window, perform the following steps:

1. In order to view an execution plan you must have **SELECT** privileges on `V$SQL_PLAN`. This can be granted using the statement:

   ```
   GRANT SELECT on V_$SQL_PLAN to <user>
   ```

2. Type a SQL statement you are interested in and execute it. The statement must be executed before you can request an Execution Plan.
3. Position the cursor on the line that contains the SQL and click the **Execution Plan (grid)** or the **Execution Plan (text)** icon in the toolbar. Alternatively, right click and select **Execution Plan (grid)** or the **Execution Plan (text)** from the menu.
4. To configure the results of Execution plan, right click on the connection node and select **Explain Plan and Execution Plan Settings**

> **Note:**
>
> Execution Plan (grid) uses `V$SQL_PLAN` while Execution Plan (text) uses the `DBMS_XPLAN` PL/SQL package.

Using Real-Time SQL Monitoring

Real-Time SQL Monitoring provides automatic monitoring of SQL statements, PL/SQL blocks, or composite database operations that are considered expensive. You can view a list of

monitored SQL, view and save SQL Monitor Active Reports, and generate an Active Report for ad hoc SQL. The information provided by Real-Time SQL Monitoring is especially useful for long-running SQL statements.

> **Note:**
>
> The use of Real-time SQL Monitoring feature requires an Oracle Tuning Pack license (except for Oracle Database Free which requires no license). For more details, please refer to "Database Licensing Information User Manual" for the database version being used.

- Connect to a database in Oracle Explorer. Any user can view complete Active Reports for SQL they execute using schema objects they own. If your user has been granted the `SELECT_CATALOG_ROLE`, then you will be able to view Active Reports for SQL executed by other users. When this role has not been granted, and the schema objects involved are not owned by the user, the Active Report may be missing information, such as the Explain Plan.
- To view a list of monitored SQL, right click on a connection in Oracle Explorer and select **Real Time SQL Monitoring** from the menu. In the list of monitored SQL that appears, click the **Activity Last:** drop down and select the time frame you are interested in. Click the **Top 100 by:** drop down and select the metric you are interested in. Click the arrows in the column headers to sort by that column.
- To view an Active Report for a SQL statement in the list, click on the SQL ID you are interested in.
- To save a report, click the **Save** icon to choose to save as XML or as an Active Report.
- To generate an Active Report for ad-hoc SQL, type some SQL in the SQL editor, right click and select **Execute and Monitor SQL**.

> **Note:**
>
> The ad-hoc SQL will be executed in order for it to be monitored, however the query results for the SQL will not be displayed as it usually is when you execute SQL.

Formatting SQL and PL/SQL with a Configurable Formatter

This extension includes a built in and configurable SQL and PL/SQL formatter that you can use to enforce style guidelines on your code.

- To modify the formatter settings, right click on an Oracle Explorer Connection and select **SQL and PL/SQL Formatter Settings**. Alternatively, in the SQL editor, click the **SQL and PL/SQL Formatter Settings** icon.
- To format SQL or PL/SQL, in the editor, click the **Format Document** icon, or right click and select **Format Document with..** and then select **Oracle Developer Tools for VS Code (SQL and PL/SQL)**
- To undo a formatting attempt, use the editor's "Undo" function (**Edit->Undo**).

Using Lowercase/Title Case Characters with Autocomplete/Intellisense

- While typing SQL or PL/SQL, autocomplete suggestions will appear as you type, for example column names. You can also type a schema name followed by a period, for example ***HR.*** to use intellisense to view and select database objects.
- You can click the ***i*** icon in the suggestions to get more details such as the schema objects the suggestion is associated with. Use the arrow keys to navigate through the suggestions, and the enter key to select a suggestion.
- By default upper case characters will be used with autocomplete or intellisense suggestions. If you wish to use lower case or title case instead, go to the settings and change ***Intellisense › Suggestions: Keyword Casing*** and ***Intellisense › Suggestions: Object Name Casing*** to ***lowercase*** or ***titlecase***.

Navigating through SQL Scripts and PL/SQL Packages

- To navigate through long SQL scripts using breadcrumbs, click on the breadcrumb at the top of the editor (it will usually be a file name or schema object). This will show a drop down of all SQL commands, PL/SQL packages, anonymous blocks, etc in the script. For some items like PL/SQL packages, procedures or functions you can further expand the breadcrumb to see additional details.
- To make breadcrumbs easier to use, you may wish to modify the Visual Studio Code settings. In particular, setting ***Breadcrumbs: File Path*** to ***last*** or ***off*** decreases the amount of space taken by the file path.
- You can also navigate through PL/SQL packages/procedures/functions that are in the database by using Oracle Explorer to right click on the procedure or function and choosing **Open** from the menu.
- While viewing SQL or PL/SQL scripts, you can right click on schema object names and from the menu select **Go to Definition/Peek Definition**, **Go to/Peek Type Definition** and **Go to/Peek Implementation**. In the case of **Go to**, these will open a new window with the PL/SQL code where the object is defined or implemented. For **Peek**, it will allow you to view and edit the code in place in the current editor window.

Executing SQL*Plus Commands

- SQL*Plus commands provide useful functionality such as connecting and disconnecting, enabling and disabling autocommit, running other SQL*Plus scripts , describing database object metadata, defining and using substitution variables, defining and using bind variables, controlling output display size, saving script output to a file, and more. To execute SQL*Plus commands do the following:

  1. Type a SQL*Plus command into the .SQL or .PLSQL file.

     While typing, autocompletion suggestions will appear as you type
  2. Select (highlight) one or more SQL*Plus commands and SQL or PL/SQL statements.
  3. Right click and select ***Execute SQL*** from the menu.

4. Select ***Execute All*** if you wish to execute all SQL*Plus commands, SQL, and PL/SQL in the current file.

   If one more more SQL*Plus commands are not supported, you will receive a warning message

   The SQL*Plus **CONNECT** command will associate the file with the connection specified in the command. After executing this command, intellisense/autocomplete will use the new connection and schema.

- For more information, including a reference guide of supported SQL*Plus commands, visit Using SQL*Plus Commands with Oracle Developer Tools for VS Code

Disabling and Enabling Auto Commit

When executing multiple SQL statements (for example, running a script), auto commit automatically commits work after each statement. If auto commit is disabled, an explicit COMMIT or ROLLBACK statement can be used to control the behavior

1. In your SQL file, enter **SET AUTOCOMMIT OFF** or **SET AUTOCOMMIT ON** and then right click and select ***Execute SQL*** from the menu. This autocommit setting will be in effect for the life of your connection and will override the default extension setting.
2. To set the default for all SQL files opened with this extension, Press F1 to open the Command Palette and enter ***Preferences: Open Settings (UI)*** command.
3. Expand the ***Extensions*** node and select ***Oracle Developer Tools for VS Code Configuration***.
4. Check the ***SQLPlus:Auto Commit*** checkbox to enable auto commit and uncheck it to disable it.

Viewing and Saving Result Sets

After executing SQL, a new document (Results Window) will open with the result set or the results will be appended to an open Results window. The maximum number of rows that can be returned is controlled by the ***Max Rows*** extension setting. As you page through the rows, more will be fetched - up to this maximum.

1. To save results, choose the format: ***.CSV or JSON***. To change the delimiter or text qualifier in a .CSV file, go to the extension settings and set ***Query›Result Set›CSV:Delimiter*** and/or ***Query›Result Set›CSV:Text Qualifier***. Then click the icon to ***Save Selected Rows*** or ***Save All Rows***. After providing a file name and location, the file will open in Visual Studio Code.
2. To copy the selected rows to the clipboard, click the ***Copy Selected Row(s) to Clipboard*** icon
3. To clear the results window, click the ***Clear Results Window*** button located in the area near the document tabs. To change the default to always clear the results window after every execution, change the ***Clear Results Window*** extension setting.

Debugging PL/SQL

Debugging PL/SQL from Oracle Database Explorer

> **Note:**
>
> PL/SQL debugging in Visual Studio Code works with both on-premises and databases in Oracle Cloud. However, Oracle Autonomous Database is not currently supported.

When first using PL/SQL debugging, configure both the PL/SQL debugger and the database:

1. In Oracle Database Explorer, right click on a connection name and select ***PL/SQL Debugger and Compiler Settings*** and set the ip address and port number range that will be used by the debugger (the Oracle Database will connect back to your machine using that IP address and one of the ports)
2. A script to grant privileges and configure the database is provided in the same dialog. Copy the script and then right click on the connection node for the database and select ***Open New SQL File***. Paste the script into the new file, modify it as needed, and then right click in the editor and select ***Execute All*** from the menu.
3. The PL/SQL procedure/function/packages you wish to debug must be compiled with debug information: Right click on the procedure/function/package and select ***Compile Debug*** from the menu. The Oracle Explorer icons will change to alert you and also to remind you to issue a ***Compile*** when done debugging to restore them to their non-debug state.
4. Right click on the procedure/function/package and select ***Open***, or ***Open Package Body*** from the menu. Set breakpoints as desired (note: conditional breakpoints are not currently supported).
5. To begin debugging, from the same menu select ***Step Into*** or ***Run Debug***.

> **See Also:**
>
> Debugging PL/SQL with Visual Studio Code (and more), for a walkthrough showing how to use PL./SQL debugging.

Debugging PL/SQL Called By an Application or SQL Scripts

> **Note:**
>
> PL/SQL debugging in Visual Studio Code works with both on-premises and databases in Oracle Cloud. However, Oracle Autonomous Database is not currently supported.

- The PL/SQL Debugger can listen for calls to a procedure/function/package and start debugging them when they are called. Any application (web app or command line, using

any programming language) or SQL script can make the PL/SQL call, and the application or script can be located on any machine.

- When first using PL/SQL debugging, configure both the PL/SQL debugger and the database:
- In Oracle Database Explorer, right click on a connection name and select ***PL/SQL Debugger and Compiler Settings*** and set the ip address and port number range that will be used by the debugger (the Oracle Database will connect back to your machine using that IP address and one of the ports)
- A script to grant privileges and configure the database is provided in the same dialog. Copy the script and then right click on the connection node for the database and select ***Open New SQL File***. Paste the script into the new file, modify it as needed, and then right click in the editor and select ***Execute All*** from the menu.
- The PL/SQL procedure/function/packages you wish to debug must be compiled with debug information: Right click on the procedure/function/package and select ***Compile Debug*** from the menu. The Oracle Explorer icons will change to alert you and also to remind you to issue a ***Compile*** when done debugging to restore them to their non-debug state.
- Right click on the procedure/function/package and select ***Open***, or ***Open Package Body*** from the menu. Set breakpoints as desired (note: conditional breakpoints are not currently supported).
- To start the debugger, right click on any connection name and select ***Start External Application Debugger***. The ***Debug Console*** will display the IP address and port number that is being used. Make a note of these.
- If the PL/SQL call is being made by an application (using Oracle data access driver other than the Oracle JDBC thin client), set a ***ORA_DEBUG_JDWP*** environment variable in the environment of the application with the value `host=ipaddress;port=portnumber`. For example: `host=127.0.0.1;port=65000`. The IP address and port number should be the same values reported in the Debug Console. This environment variable must be set before the connection is made by the application to the Oracle Database. For example, this can be set in a Visual Studio Code project in the launch.json file in the configurations section (eg `"env": { "ORA_DEBUG_JDWP": "host=127.0.0.1;port=65000" }`) or it can be set on a command line before launching the application.
- If the PL/SQL call is being made by an application using the Oracle JDBC thin client, or if you cannot enable debugging at the time the connection to the Oracle database is made, modify your code to make a call to the DBMS_DEBUG_JDWP.CONNECT_TCP package procedure at some point in your code before the call is made to the stored procedure/ function/or package that you wish to debug. In this call to CONNECT_TCP provide the same IP address and port number reported in the Debug Console. For example, if using JDBC: `String sql = "CALL DBMS_DEBUG_JDWP.CONNECT_TCP(?,?)"; CallableStatement callStmt = conn.prepareCall(sql)); callStmt.setString(1, host); callStmt.setString(2, port); callStmt.execute();`. At a point in your code after the PL/SQL call is made and debugging is finished, add a call to DBMS_DEBUG_JDWP.DISCONNECT as well.
- If the PL/SQL call is being made by an SQL script in Visual Studio Code, add the following line in the SQL script at some point before the call is made to the stored procedure/ function/or package that you wish to debug: `exec (DBMS_DEBUG_JDWP.CONNECT_TCP('ip address', 'port'));`. For example: `exec (DBMS_DEBUG_JDWP.CONNECT_TCP('127.0.0.1', '65000'));` The IP address and port number should be the same values reported in the Debug Console. At a point in the script after the PL/SQL call is made and debugging is finished, add `exec (DBMS_DEBUG_JDWP.DISCONNECT();)`.

- When the application or SQL script is run, a message will appear in the Debug Console stating *A database has connected to the PL/SQL Debugger on host: youripaddress and port: yourport.* This indicates that the Oracle Database has successfully connected back to Visual Studio Code and debugging can begin.

> **See Also:**
>
> Debugging PL/SQL with Visual Studio Code (and more), for a walkthrough showing how to use PL./SQL debugging.

Viewing SQL History

1. In Oracle Explorer, click on the ***History*** control to open it.
2. Expand the connection name for which you want to see the history. Previously executed SQL commands from this session will be listed.
3. Hover over a long SQL statement to view a tooltip containing the complete statement.
4. To execute a SQL statement directly from the history, right click on it and select ***Run***.
5. To copy the SQL to an open editor, right click on it and select ***Copy to Editor***.
6. To copy the SQL to an new SQL File and open it in an editor, right click on it and select ***Open in Editor***. Then press F1 to open Command Palette and select ***Oracle:Connect*** from the dropdown.
7. SQL history does not persist across sessions. For permanent storage, save the SQL as a bookmark by right clicking and selecting ***Bookmark SQL*** from the menu.

Creating a SQL Bookmark

1. Select one or more lines of SQL or PL/SQL in the editor, right click and select ***Bookmark SQL*** from the menu
2. Provide a bookmark name and a folder name
3. In Oracle Explorer, click on the ***Bookmarks*** control to open it.
4. Expand the folder name. Bookmarks will be listed.
5. Hover over a bookmark to view a tooltip containing the complete statement.
6. To execute the SQL statement directly, right click on the bookmark and select ***Run***.
7. To copy the SQL to an open editor, right click on it and select ***Copy to Editor***.
8. To copy the SQL to a new SQL file, right click on the bookmark and choose ***Open in Editor*** from the menu.

Configure the Location of the Query Results Window

To modify the default location of the Query Results Window, for example, to have the SQL code editor in an upper pane and the Query Results window in a lower pane, open the extension settings and modify the **Query: Open Results Window** setting as shown below:

1. Select **Right** to open results window to the right of the currently active editor.
2. Select **Below** to open results window below the currently active editor.
3. Select **Workbench > Editor Management: Open Positioning** to open the results window using the Visual Studio Code **Workbench > Editor Management: Open Positioning** setting.

Creating and Modifying Keyboard Shortcuts

1. To create keyboard shortcuts (keybindings) or to modify existing ones, go to the ***File*** menu in VS Code and select ***Preferences->Keyboard Shortcuts***
2. In the search text box, enter a portion of the name of the extension whose shortcut you would like to modify, for example "oracle".
3. A list of commands and their shortcuts will appear. Click the pencil icon on the left side of a row to edit a shortcut, or the plus (**+**) sign to create a new one.
4. When prompted, enter the key combination and if it is not being used by another extension, press enter. If it is being used by another extension, press escape and try again.
5. To delete a shortcut, right click on the shortcut and select ***Remove Keybinding***

2

Learn What's New in This Version

To see a list of new features in this release, visit the Visual Studio Code Marketplace

3

Get Help or Provide Feedback

Visit the Oracle Developer Tools for VS Code help forums to ask questions and provide feedback about this extension

Glossary

Index