Oracle® Database

JDBC Quick Start Guide

Release 21c
F77234-01
November 2023

Oracle Database JDBC Quick Start Guide, Release 21c

F77234-01

Primary Author: Gunjan Jain

Contributing Authors: Nirmala Sundarappa, Tulika Das

Contributors: Kuassi Mensah

Contents

Preface

This quick start guide shows how to successfully establish a connection to an Oracle Database (On-Premises) or Oracle Cloud Database that uses TCP connections.

Audience

The *Oracle® Database JDBC Quick Start Guide* is intended for Java developers who use Oracle JDBC driver to connect to the database in their application. This book can be read by anyone with an interest in JDBC programming, but assumes at least some prior knowledge of the following:

- Oracle PL/SQL

To use this document, you must be familiar with

- Relational database concepts
- Your current Oracle Database release
- Your operating system environment

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see these Oracle resources:

- *Oracle Database JDBC Developer's Guide*
- *Oracle Universal Connection Pool Developer's Guide*

1
Prerequisites

The following sections provide information about the tasks that you need to perform before establishing a connection to Oracle Database.

- Installing Oracle Database
  This section lists a few resources that you can use to install Oracle Database Express Edition (XE) 21c. If you have already installed Oracle Database and have the credentials, then you can skip this step.
- Installing JDK 8
  This section lists the step to install JDK 8.
- Creating a Database User (Optional)
  This section lists the steps to create a database user.
- Downloading a Sample Java Program from Github
  This section lists the steps to download a sample Java Program from Github.

1.1 Installing Oracle Database

This section lists a few resources that you can use to install Oracle Database Express Edition (XE) 21c. If you have already installed Oracle Database and have the credentials, then you can skip this step.

- **Windows or Linux platform**: Download and install Oracle Database Release 21c Oracle XE QuickStart.
- **Docker on Mac**: Install and run the Oracle XE on Docker following the instructions from this blog Oracle Database XE Release 21c (21.3.0.0) Docker Image Documentation.
- **More Resources**: Video showing how to install Oracle Database XE on Windows and conduct basic administration.

1.2 Installing JDK 8

This section lists the step to install JDK 8.

Install JDK 8 or a higher JDK version that is a long-term release. For example, JDK 11 or JDK 17.

1.3 Creating a Database User (Optional)

This section lists the steps to create a database user.

> **Note:**
>
> If you already have a database user and password, then you can skip this step.

It is recommended to create a new database user when verifying the database connection. If you are not familiar with creating a database user and assigning privileges, then you can use the CreateUser.java file to create a new database user.

1. Download and edit the file CreateUser.java, provide the connection string, admin user, and admin password along with the new database user and password to be created.
2. Download the latest Oracle JDBC driver and UCP.
3. Make sure that Oracle Database is running before compiling and running the sample.

```
# Compile the Java program
 javac -classpath ./test/ojdbc8.jar:./test/ucp.jar CreateUser.java
# Run the Java program
 java -classpath ./test/ojdbc8.jar:./test/ucp.jar:. CreateUser
```

Alternately, you can use the following commands to create a new database user if you are using Docker.

```
# These are instructions to create a new database user
#Step 1: Make sure docker is running before running the below
docker command
#Step 2: Enter the admin password for your database on the console
#Step 3: Provide the new database user and password to be created

# Create a session within the container with Oracle as user
docker exec -it --user oracle \
   -e 'ORACLE_HOME=/opt/oracle/product/18c/dbhomeXE' \
   $(docker ps --format '{{.ID}}') \
   /bin/bash -c  'exec ${ORACLE_HOME}/bin/sqlplus "sys@xepdb1 as
sysdba"'

# Create a new database user and grant required privileges.
# Replace "{newdbuser}"  and "{newdbpassword}" with the new DB user
and password that you want to create
grant CREATE SESSION, CREATE VIEW, CREATE SEQUENCE,CREATE
PROCEDURE, CREATE TABLE,
CREATE TRIGGER, CREATE TYPE, CREATE MATERIALIZED VIEW to
{newdbuser} identified by {newdbpassword};
```

1.4 Downloading a Sample Java Program from Github

This section lists the steps to download a sample Java Program from Github.

1. Download the QuickStart.java file from Github. This sample application creates a table `todoitem` that contains the tasks and the task completion status, inserts a few rows, and displays the contents of the table.
2. Edit the `QuickStart.java` file to include the following database connection information:
   - **DB_USER**: Use the database user
   - **DB_PASSWORD**: Use the database password

- **DB_URL**: Enter the connection string for Oracle Database. For Oracle Database XE, use the following connection string:

  ```
  Example : DB_URL = "jdbc:oracle:thin:@//localhost:1521/XEPDB1"
  ```

> **See Also:**
>
> Creating a Database User (Optional)

3. Save the changes to the `QuickStart.java` file.

2
Using Maven and Gradle

The following sections provide information about how to connect to Oracle Database using a Maven project or a Gradle project.

- Using a Maven Project
  You can establish a connection to Oracle Database using the Maven Project.
- Using a Gradle Project
  You can establish a connection to Oracle Database using the Gradle project.

2.1 Using a Maven Project

You can establish a connection to Oracle Database using the Maven Project.

- Setting Up a Maven Project
  This section lists the steps to set up a Maven project.
- Building and Running a Sample Java Program
  This section lists the steps to build and run a sample Java program.

2.1.1 Setting Up a Maven Project

This section lists the steps to set up a Maven project.

Make sure to complete all the steps from the Prerequisites section.

1. Create a Maven project.

   Download Apache Maven and set the **PATH** before using mvn commands. Use the following Maven command to create a project:

   ```
   mvn archetype:generate -DgroupId=com.oracle.jdbctest -DartifactId=jdbc-test-project -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
   ```

2. Copy the QuickStart.java file to the `src/main/java/com/oracle/jdbctest` directory.
3. Modify the `pom.xml` file with the following changes:
   - Add Oracle JDBC driver as a dependency.

> **Note:**
>
> `ojdbc8-production` downloads Oracle JDBC driver (`ojdbc8.jar`) along with UCP, a client side connection pool (`ucp.jar`). Refer to the Maven Central Guide for more details.

```
<properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
</properties>
<dependencies>
  <dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>ojdbc8-production</artifactId>
    <version>19.9.0.0</version>
    <type>pom</type>
  </dependency>
</dependencies>
```

2.1.2 Building and Running a Sample Java Program

This section lists the steps to build and run a sample Java program.

Make sure that you are in the directory where the `pom.xml` file is present.

1. Clean and compile the Java code.

   Use the following commands:

   ```
   mvn clean
   ```

   ```
   mvn compile
   ```

2. Run the sample Java program:

   ```
   mvn exec:java -Dexec.cleanupDaemonThreads=false -
   Dexec.mainClass="com.oracle.jdbctest.QuickStart"
   ```

Sample Output:

You will see the queried rows returned from the new table `todoitem` and a message **Congratulations! You have successfully used Oracle Database** as shown in the following screen:

```
New table 'todoitem' is created
New records are inserted

New table 'todoitem' contains:
DESCRIPTION     DONE
-------------------------
Task 1          0
Task 2          0
Task 3          1
Task 4          0
Task 5          1

Congratulations!! You have successfully used Oracle Database
```

2.2 Using a Gradle Project

You can establish a connection to Oracle Database using the Gradle project.

- Setting Up a Gradle Project
  This section lists the steps to set up a Gradle project.
- Building and Running the Gradle Application
  This section lists the steps to run the Gradle application.

2.2.1 Setting Up a Gradle Project

This section lists the steps to set up a Gradle project.

Make sure to complete all the steps from the Prerequisites section.

1. Create a Gradle project.

   Follow instructions from the Gradle Guide for Gradle download and build instructions and set the **PATH** before using Gradle commands. As a first step, create a Gradle project using the following command. Make sure to choose `2:application` for **Select type of project to generate**. Also, for **Source package (default:temp):**, use `com.oracle.jdbctest`.

   ```
   gradle init
   ```

2. Copy the `QuickStart.java` file to the `src/main/java/com/oracle/jdbctest` directory.
3. Modify the `build.gradle` file with the following changes:
   - Add `mavenCentral()` as a repository.
   - Add the Oracle JDBC driver as a dependency.

> **Note:**
>
> *ojdbc8-production* downloads Oracle JDBC driver (`ojdbc8.jar`) along with UCP as a client side connection pool (`ucp.jar`). Refer to the Maven Central Guide for more details.

- Update the `mainClassName` to `QuickStart`.

```
repositories {
  // Maven Central
   mavenCentral()
 }
dependencies {
  // Get the 21.1.0.0 Oracle JDBC driver along with other companion
jars
  implementation("com.oracle.database.jdbc:ojdbc8-
production:19.9.0.0")
 }
application {
  // Define the main class for the application
  mainClassName ='{your_project_directory}.QuickStart'
}
```

2.2.2 Building and Running the Gradle Application

This section lists the steps to run the Gradle application.

Make sure you are in the directory where the `build.gradle` file is present.

1. Compile the Jave code using the following command:

```
./gradlew build
```

2. Run the sample Java Program.

```
./gradlew run
```

Sample Output: You will see the queried rows returned from the new table `todoitem` and a message **Congratulations! You have successfully used Oracle Database** as shown in the following screen.

```
New table 'todoitem' is created
New records are inserted

New table 'todoitem' contains:
DESCRIPTION     DONE
-------------------------
Task 1          0
Task 2          0
Task 3          1
Task 4          0
Task 5          1

Congratulations!! You have successfully used Oracle Database
```

3
Using Eclipse and IntelliJ

The following sections provide information about how to connect to Oracle Database using Eclipse and IntelliJ IDEs:

- Using the Eclipse IDE
  You can establish a connection to Oracle Database using the Eclipse IDE.
- Using the IntelliJ IDE
  You can establish a connection to Oracle Database using the IntelliJ IDE.

3.1 Using the Eclipse IDE

You can establish a connection to Oracle Database using the Eclipse IDE.

- Setting Up a Maven Project
  This section lists the steps to set up a Maven project.
- Building and Running a Sample Java Program
  This section lists the steps to build and run a Java program.

3.1.1 Setting Up a Maven Project

This section lists the steps to set up a Maven project.

Make sure to complete all the steps from the Prerequisites section.

1. Create a Maven project.
   a. From the **File** menu, select **New**, and then select **Project**. You can either use Maven archetype or select **Create a simple project** (skip archetype selection).
   b. Choose GAV for your Maven project. These will appear in the `pom.xml` file for the project.
      - **Group Id:** com.oracle
      - **Artifact Id:** JDBCquickstart
      - **Version:** leave it as 0.0.1-SNAPSHOT
2. Create the `QuickStart.java` file.
   a. Right-click on `src/main/java`, select **New**, and then select **Class**.
   b. Enter the following values, and then click **Finish**.
      - **Package:** com.oracle.jdbctest
      - **Name:** QuickStart

> **Note:**
>
> Make sure to copy the contents of the QuickStart.java file to this new file created.

3. Modify the `pom.xml` file with the following changes:

   Add Oracle JDBC driver as a dependency.

> **Note:**
>
> `ojdbc8-production` will download Oracle JDBC driver (`ojdbc8.jar`) along with `ucp.jar` (it is a JAR file required for using UCP as a client side connection pool). Refer to the Maven Central Guide for more details.

```
<properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
</properties>
<dependencies>
  <dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>ojdbc8-production</artifactId>
    <version>21.1.0.0</version>
    <type>pom</type>
  </dependency>
</dependencies>
```

3.1.2 Building and Running a Sample Java Program

This section lists the steps to build and run a Java program.

Make sure you do not have any compilation error in the Java code and you are using the latest JDK version in Eclipse. Also, make sure that Oracle Database is running on Docker.

1. Right-click **QuickStart.java**.
2. Click **Run As**, and then click **Java Application** to run the sample Java program.

Sample Output:
You will see the queried rows returned from the new table `todoitem` and a message **Congratulations! You have successfully used Oracle Database** as shown in the following screen:

```
New table 'todoitem' is created
New records are inserted

New table 'todoitem' contains:
DESCRIPTION     DONE
-------------------------
Task 1          0
Task 2          0
Task 3          1
Task 4          0
Task 5          1

Congratulations!! You have successfully used Oracle Database
```

3.2 Using the IntelliJ IDE

You can establish a connection to Oracle Database using the IntelliJ IDE.

- Setting Up a Maven Project
  This section lists the steps to set up a Maven project.
- Building and Running QuickStart
  This section lists the steps to build and run QuickStart.

3.2.1 Setting Up a Maven Project

This section lists the steps to set up a Maven project.

Make sure to complete all the steps from the Prerequisites section.

1. Create a Maven project.
   a. From the **File** menu, select **New**, and then select **Project**.
   b. Choose **Maven** on the left side and choose the latest version of JDK as **Project SDK**. You can use any Maven archetype or add your own.
   c. Click **Next**.
   d. Give **Name** as `Quickstart`.
2. Create `QuickStart.java` file.
   a. Right-click on `src/main/java`.
   b. Select **New**, and then select **Java Class**.
   c. Enter `com.oracle.jdbctest.QuickStart.java`. This will create the required package structure as well. Make sure to copy contents of QuickStart.java file to this new file.
3. Modify the `pom.xml` file with the following changes:

   Add Oracle JDBC driver as a dependency.

> **Note:**
>
> `ojdbc8-production` will download the Oracle JDBC driver (`ojdbc8.jar`) along with `ucp.jar` (it is a JAR file required for using UCP as a client side connection pool). Refer to the Maven Central Guide for more details.

```
<properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
</properties>
<dependencies>
  <dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>ojdbc8-production</artifactId>
    <version>21.1.0.0</version>
    <type>pom</type>
  </dependency>
</dependencies>
```

3.2.2 Building and Running QuickStart

This section lists the steps to build and run QuickStart.

1. Compile the Java code.

   Right-click **QuickStart.java**, and then click **`Build Module 'QuickStart'`**.

   Make sure that there are no compilation errors. Also, ensure that Oracle Database is running on Docker and is accessible.

2. Run the sample Java program.

   Right-click **QuickStart.java**, and then click **`Run 'QuickStart.main()'`**.

Sample Output:
You will see the queried rows returned from the database and a message **Congratulations! You have successfully used Oracle Database** as shown in the following screen:

```
New table 'todoitem' is created
New records are inserted

New table 'todoitem' contains:
DESCRIPTION     DONE
-------------------------
Task 1          0
Task 2          0
Task 3          1
Task 4          0
Task 5          1

Congratulations!! You have successfully used Oracle Database
```

4
Connecting to Oracle Database Manually

The following sections describe establishing a connection to Oracle Database manually without any build tool:

- Downloading JDBC Driver and Other JARs
  This section lists the steps to download JDBC drivers and other JARs.
- Building and Running a Sample Java Program
  This section lists the step to build and run a sample Java program.

4.1 Downloading JDBC Driver and Other JARs

This section lists the steps to download JDBC drivers and other JARs.

Make sure to complete all the steps from the Prerequisites section.

Download ojdbc8-full.tar.gz from OTN and unzip the contents to your classpath.

> **Note:**
>
> `ojdbc8-full.tar.gz` contains the latest JDBC driver (`ojdbc8.jar`) and UCP as a client side connection pool (`ucp.jar`).

4.2 Building and Running a Sample Java Program

This section lists the step to build and run a sample Java program.

1. Compile the Java program.

   Make sure to provide the correct path for the required JARs in the classpath.

   ```
   javac -classpath ./lib/ojdbc8.jar:./lib/ucp.jar com/oracle/jdbctest/
   QuickStart.java
   ```

2. Run the sample Java program.

   Make sure to provide the correct path for the required JARs in the classpath.

   ```
   java -classpath ./lib/ojdbc8.jar:./lib/ucp.jar:.
   com.oracle.jdbctest.QuickStart
   ```

Sample Output:
You will see the queried rows returned from the new table `todoitem` and a message **Congratulations! You have successfully used Oracle Database** as shown in the following screen:

```
New table 'todoitem' is created
New records are inserted

New table 'todoitem' contains:
DESCRIPTION     DONE
-------------------------
Task 1          0
Task 2          0
Task 3          1
Task 4          0
Task 5          1

Congratulations!! You have successfully used Oracle Database
```