Oracle® Autonomous Database

JDBC Quick Start Guide

F70770-02
July 2024

Oracle Autonomous Database JDBC Quick Start Guide,

F70770-02

Primary Author: Gunjan Jain

Contributing Authors: Nirmala Sundarappa, Tulika Das

Contributors: Kuassi Mensah

Contents

Preface

Audience v
Documentation Accessibility v
Conventions v

1 Prerequisites

1.1 Prerequisites (one-way TLS) 1-1
1.1.1 Provisioning an Oracle Autonomous Database Instance 1-1
1.1.2 Disabling Mutual TLS (mTLS) Requirement 1-2
1.1.3 Installing JDK 8 1-3
1.1.4 Downloading a Sample Program from Github 1-4
1.1.5 Recent Changes to Database Connectivity 1-5
1.2 Prerequisites (mutual TLS) 1-5
1.2.1 Provisioning an Oracle Autonomous Database Instance 1-6
1.2.2 Obtaining Client Credentials 1-6
1.2.3 Installing JDK 8 1-7
1.2.4 Downloading a Sample Program from Github 1-7
1.3 Additional Step for Changed Connection String 1-8

2 Using Maven and Gradle

2.1 Using a Maven Project 2-1
2.1.1 Setting Up a Maven Project 2-1
2.1.2 Building and Running a Sample Java Program 2-2
2.2 Using a Gradle Project 2-3
2.2.1 Setting Up a Gradle Project 2-4
2.2.2 Building and Running the Gradle Application 2-5

3 Using Eclipse and IntelliJ

3.1 Using the Eclipse IDE 3-1
3.1.1 Setting Up a Maven Project 3-1
3.1.2 Building and Running a Sample Java Program 3-2

3.2 Using the IntelliJ IDE 3-3
3.2.1 Setting Up a Maven Project 3-3
3.2.2 Building and Running ADBQuickStart 3-4

4 Connecting to Oracle Database Manually

4.1 Downloading JDBC Driver and other JARs 4-1
4.2 Building and Running a Sample Java Program 4-1

Preface

This quick start guide shows how to connect your Java applications to Oracle Autonomous Database using Oracle JDBC driver and Universal Connection Pool. Oracle Autonomous Database allows both one-way TLS as well as mutual TLS (mTLS) for connection, the default being the later. With one-way TLS, also called as TLS, you no longer need either Oracle Wallets or Java KeyStore (JKS) to connect to Oracle Autonomous Database. When you disable the mTLS requirement, connections can be established. The Prerequisites section addresses both of these approaches.

If you want to connect to Oracle Database (On-Premises) or Oracle Cloud Database that uses TCP connections then see QuickStart Java applications with Oracle Database (On-premises).

- Audience
- Documentation Accessibility
- Conventions

> **See Also:**
>
> Develop Java applications with Oracle Database for other technical briefs, blogs, and videos.

Audience

This document is intended for developers and architects who use Java to build applications using Oracle Cloud Services, such as Oracle Autonomous Database.

Documentation Accessibility

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

Access to Oracle Support

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

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Prerequisites

The following sections provide information about the tasks that you need to perform before connecting your Java applications to Oracle Autonomous Database using Oracle JDBC driver and Universal Connection Pool.

- Prerequisites (one-way TLS)
  This section lists prerequisites for one-way TLS authentication.
- Prerequisites (mutual TLS)
  This section lists prerequisites for mutual TLS authentication.
- Additional Step for Changed Connection String
  This section includes an additional step that is required to be performed when you change the Autonomous Database Serverless connection string.

1.1 Prerequisites (one-way TLS)

This section lists prerequisites for one-way TLS authentication.

- Provisioning an Oracle Autonomous Database Instance
  This section describes how to provision an Oracle Autonomous Database Instance if you have not provisioned it already.
- Disabling Mutual TLS (mTLS) Requirement
  This section lists the steps to disable the mutual TLS (mTLS) requirement and to enable both mTLS and one-way TLS connections.
- Installing JDK 8
  This section lists the step to install JDK 8.
- Downloading a Sample Program from Github
  This section lists the steps to download a sample program from Github.
- Recent Changes to Database Connectivity
  This section lists different versions of JDBC-thin that support host name-based matching.

1.1.1 Provisioning an Oracle Autonomous Database Instance

This section describes how to provision an Oracle Autonomous Database Instance if you have not provisioned it already.

Click these links to walk through the steps to provision an Oracle Autonomous Database Instance.

- Create an Oracle Cloud Account
- Provision an Oracle Autonomous Database Instance

Remember the password that you set while provisioning the Autonomous Database for the `ADMIN` user. For demonstration purposes, the `ADMIN` user is used, but it is recommended to create other database users either using Oracle SQL Developer or Database Actions.

1.1.2 Disabling Mutual TLS (mTLS) Requirement

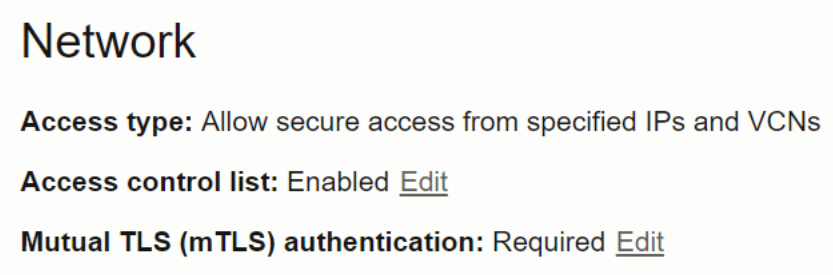
This section lists the steps to disable the mutual TLS (mTLS) requirement and to enable both mTLS and one-way TLS connections.

After the creation of Oracle Autonomous Database, mutual TLS is enabled as a default option.

Follow these instructions to enable both mTLS and one-way TLS connections. After TLS is enabled, you can use both TLS and mTLS to connect to Oracle Autonomous Database.

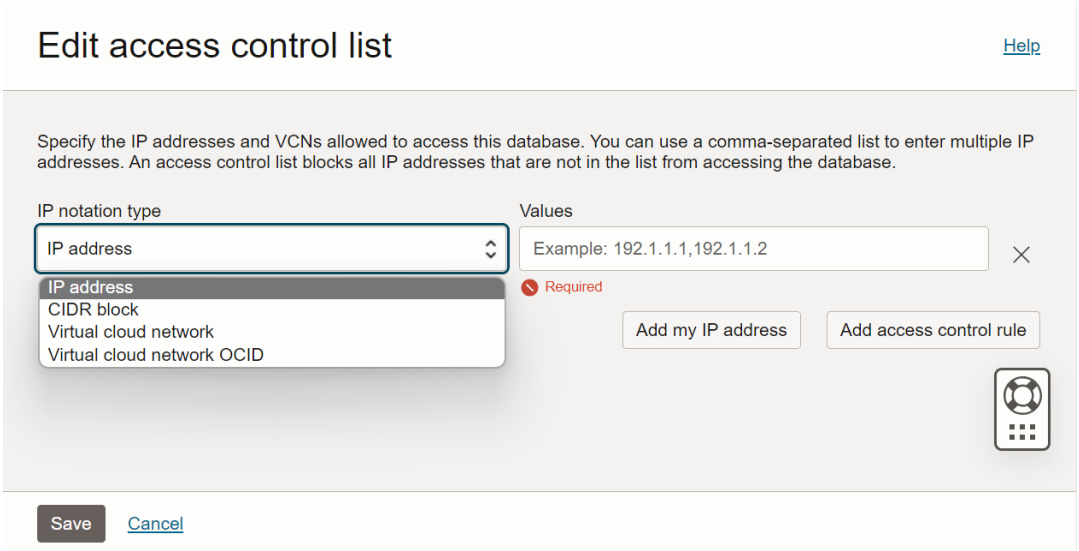
> **See Also:**
>
> *Using Oracle Autonomous Database Serverless* for more information.

1. From the Oracle Cloud Infrastructure left navigation menu, click **Oracle Database**, and then click **Autonomous Database**.
2. On the **Autonomous Databases** page, select your Autonomous Database from the links under the **Display name** column.
3. On the **Autonomous Database details** page, find the section titled **Network** and click **Edit** next to the **Access control list**.

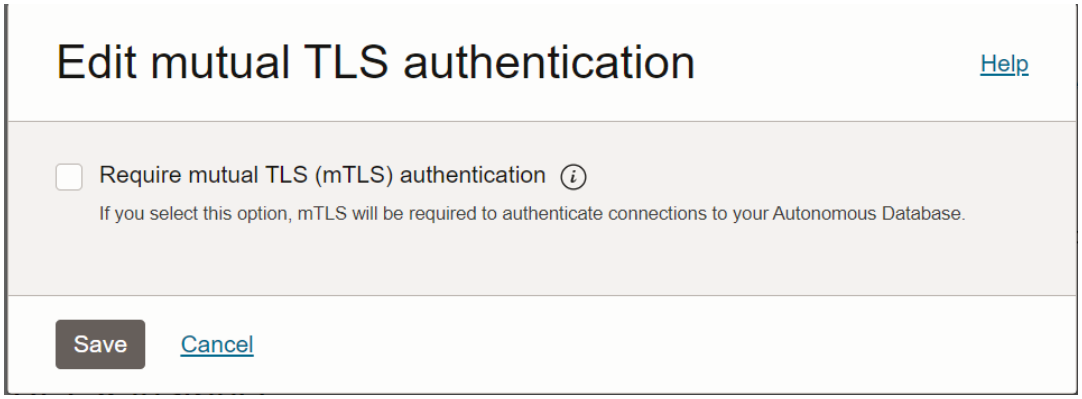


4. In the **Edit access control list** dialog box, select the value for the type of entry that you want to make and enter the value.

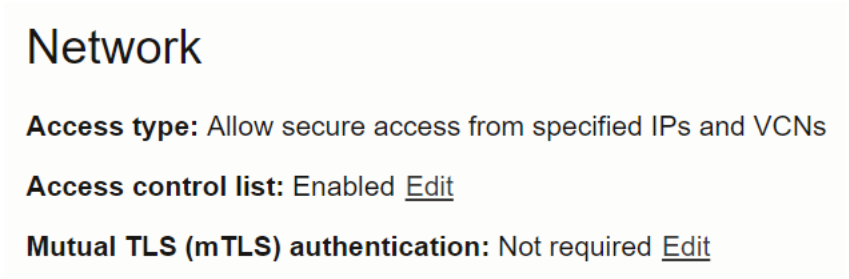
   You can add entries by IP address (your local IP), CIDR block, and VCN (by name or OCID). Add as many as necessary.



5. Click **Edit** in the **Mutual TLS (mTLS) authentication** field, deselect the **Require mutual TLS (mTLS) authentication** check box, and then click **Save**.



After some time the status will change to *Available*. Your network settings should look as shown in the following screenshot:



1.1.3 Installing JDK 8

This section lists the step to install JDK 8.

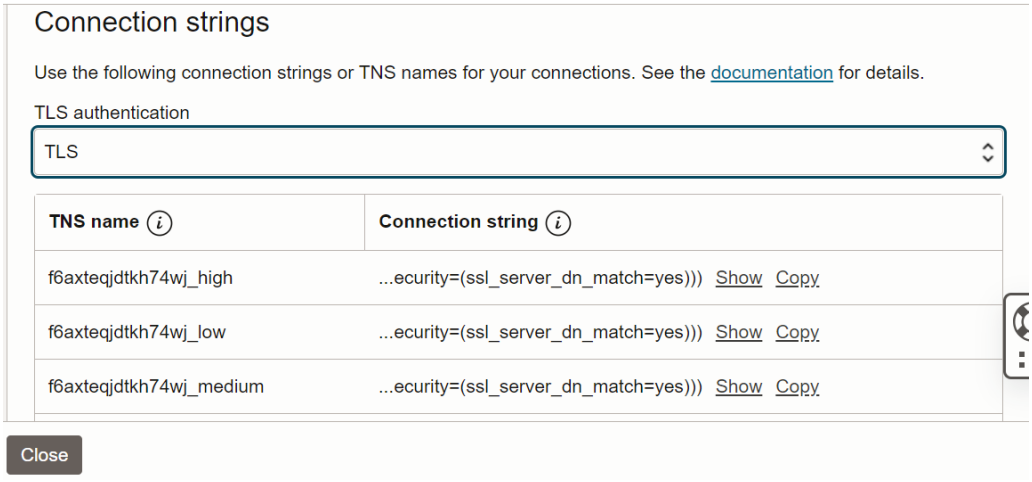
Download latest JDK 8 or a higher JDK versions.

> **Note:**
>
> Ensure that you use `JDK8u162` or a later version. Use `java -version` to check the JDK version that you have installed. To check the JDBC driver version, type `java -jar ojdbc8.jar`.

1.1.4 Downloading a Sample Program from Github

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

1. Download the ADBQuickStart.java file from Github. This sample application uses the Sales History (SH) sample schema and displays 20 records from the `SH.CUSTOMERS` table.
2. Modify the `ADBQuickStart.java` file to include your Oracle Autonomous Database connection information:
   - **DB_USER**: You can use `ADMIN`, the user created by default when the Oracle Autonomous Database is created (if you create another Oracle Autonomous Database user, you can use that user instead).
   - **DB_PASSWORD**: Use the database user's password. If connecting as the `ADMIN` user, set this to the password you chose during the **Create Autonomous Database** step while provisioning Oracle Autonomous Database. For security reasons, you need to enter the password through the console when you run the sample.
   - **DB_URL:** Click **Database connection** on the Oracle Cloud Infrastructure Console. In the **Connection strings** section, choose *TLS* from the **TLS authentication** drop-down list and copy the appropriate connection string based on your requirements. If you are directly using in the Java program, you need to escape **"** in the connection string with **\"**



Example:

```
DB_URL = "jdbc:oracle:thin:@jdbc:oracle:thin:@(description=
      (retry_count=20)(retry_delay=3)(address=(protocol=tcps)(port=1521)
(host=adb.us-sanjose-1.oraclecloud.com))
(connect_data=(service_name=mydemo.adb.oraclecloud.com))
(security=(ssl_server_cert_dn=\"CN=adb.us-sanjose-1.oraclecloud.com,
      OU=Oracle ADB SANJOSE, O=Oracle Corporation, L=Redwood City,
```

```
ST=California,
      C=US\")))"
DB_USER="ADMIN" and DB_PASSWORD="enter_it_from_console"
```

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

   Troubleshooting and Debugging: If you encounter any error, refer the Troubleshooting Tips page for some helpful hints.

1.1.5 Recent Changes to Database Connectivity

This section lists different versions of JDBC-thin that support host name-based matching.

DigiCert retired the Organizational Unit (OU) field for all public TLS/SSL certificates to comply with industry standards as of August 2022 per their announcement. This means that the public TLS/SSL certificates issued by DigiCert will no longer have an OU field.

> **See Also:**
>
> MOS Note **2911553.1**

To avoid disruption to applications connecting to Oracle Autonomous Database Serverless during the server side certificate change while preserving security, you must use host name-based matching (Domain Name (DN) matching) of the server certificate (for TLS server-authentication).

The following versions of JDBC-thin support host name-based matching:

- 21.6 (or later) or 19.15 (or later). These are the recommended versions that support `(security=(ssl_server_dn_match=yes))` in the TNS connection string.
- Other versions that require turning on host name-based matching explicitly are:
  - 21.5 (or before) and 19.14 (or before)
  - 18.21.0.0-patched-for-bug-28492769 and 12.2.0.1-Patched-for-bug-28492769
  - 12.1.0.2 and 11.2.0.4 with patch for bugs 28492769 and 19030178 (for host name-based matching and TLS v1.2 support)
  - You must also explicitly turn on DN matching using one of the following methods:
    * Using a program: `prop.setProperty("oracle.net.ssl_server_dn_match", "true")`
    * Setting a Java system property: `-Doracle.net.ssl_server_dn_match=true`

> **See Also:**
>
> Oracle JDBC Drivers Archive Page

1.2 Prerequisites (mutual TLS)

This section lists prerequisites for mutual TLS authentication.

- Provisioning an Oracle Autonomous Database Instance
  This section describes how to provision an Oracle Autonomous Database Instance if you have not provisioned it already.
- Obtaining Client Credentials
  This section lists the steps to obtain client credentials.
- Installing JDK 8
  This section lists the step to install JDK 8.
- Downloading a Sample Program from Github
  This section lists the steps to download a sample program from Github.

1.2.1 Provisioning an Oracle Autonomous Database Instance

This section describes how to provision an Oracle Autonomous Database Instance if you have not provisioned it already.

Click these links to walk through the steps to provision an Oracle Autonomous Database Instance.

- Create an Oracle Cloud Account
- Provision an Oracle Autonomous Database Instance

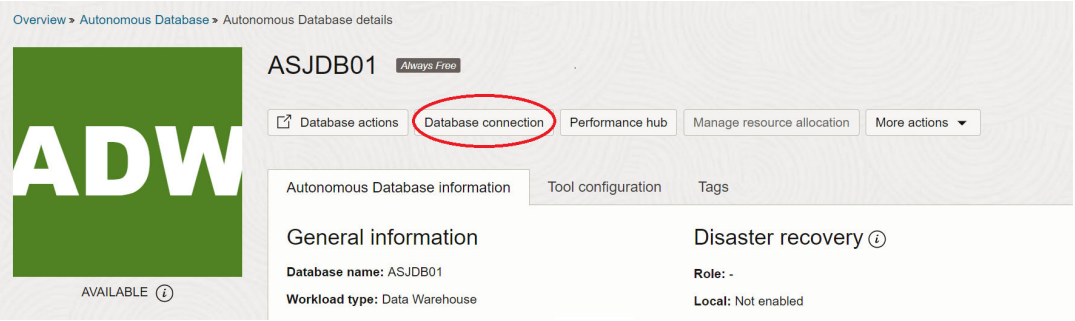
Remember the password that you set while provisioning the Autonomous Database for the `ADMIN` user. For demonstration purposes, the `ADMIN` user is used, but it is recommended to create other database users either using Oracle SQL Developer or Database Actions.

1.2.2 Obtaining Client Credentials

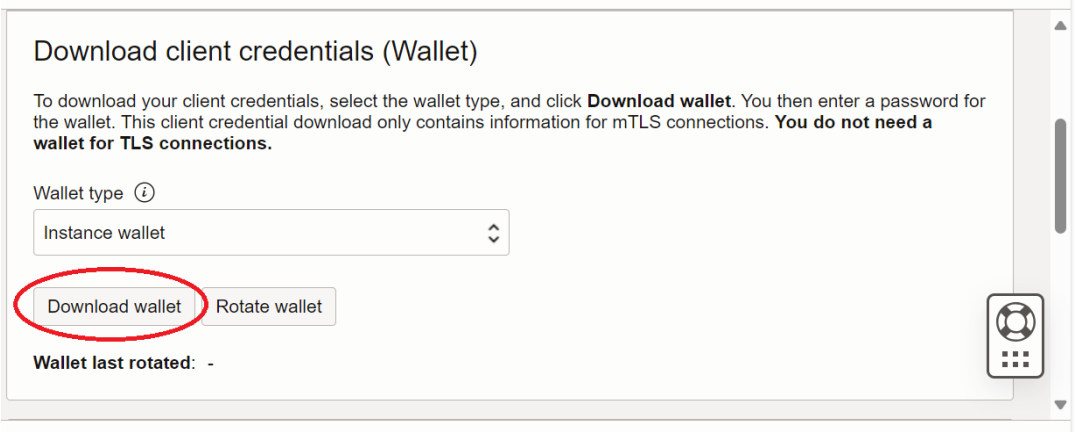
This section lists the steps to obtain client credentials.

After the creation of Oracle Autonomous Database, follow these instructions to download client credentials from the Oracle Cloud Infrastructure Console. The client credentials (`wallet_[dbname].zip`) contain the required wallet files and the `tnsnames.ora` file that allows you to connect using a mTLS connection, providing enhanced security for authentication and encryption.

1. From the **Oracle Cloud Infrastructure Console**, open the navigation menu, select **Oracle Database**, and then select **Autonomous Database**.
2. Go to the **Autonomous Database details** page of your Oracle Autonomous Database Instance.
3. Click **Database connection**.



4. Click **Download wallet**. Leave the **Wallet type** to be *Instance wallet*.



5. Enter a wallet password in the **Password** field and confirm the password in the **Confirm password** field. Then, click **Download**.

   The password must be at least eight characters long and must include at least one letter, and either one numeric character or one special character.

6. Save and unzip the client credentials zip (`wallet_[dbname].zip`) file to a secure directory. You will need this directory location later on.

1.2.3 Installing JDK 8

This section lists the step to install JDK 8.

Download latest JDK 8 or a higher JDK versions.

> **Note:**
>
> Ensure that you use `JDK8u162` or a later version. Use `java -version` to check the JDK version that you have installed. To check the JDBC driver version, type `java -jar ojdbc8.jar`.

1.2.4 Downloading a Sample Program from Github

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

1. Download the ADBQuickStart.java file from Github. This sample application uses the Sales History (SH) sample schema and displays 20 records from the `SH.CUSTOMERS` table.
2. Modify the `ADBQuickStart.java` file to include the Oracle Autonomous Database connection information:
   - **DB_USER:** You can use `ADMIN`, the user created by default when the Oracle Autonomous Database is created (if you create another Oracle Autonomous Database user, you can use that user instead).
   - **DB_PASSWORD:** Use the database user's password. If connecting as the `ADMIN` user, set this to the password you chose during the **Create Autonomous Database** step while provisioning Oracle Autonomous Database. For security reasons, you need to enter the password through the console when you run the sample.

- **DB_URL:** Enter the net service name (TNS Alias) **DBName_medium** where **DBName** is the Oracle Autonomous Database Name entered during the **Create Autonomous Database** step while provisioning Oracle Autonomous Database. The available net service names can be seen in the `tnsnames.ora` file, which is a part of the client credentials zip file.
  `TNS_ADMIN` should point to the location where you have unzipped the client credentials of Oracle Autonomous Database.

Example:

```
DB_URL = "jdbc:oracle:thin:@DBName_medium?TNS_ADMIN=/Users/test/
wallet_DBName" DB_USER="ADMIN" and DB_PASSWORD="enter_it_from_console"
```

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

> **See Also:**
>
> - Troubleshooting and Debugging
> - https://www.youtube.com/watch?v=icexaj5k3qw

1.3 Additional Step for Changed Connection String

This section includes an additional step that is required to be performed when you change the Autonomous Database Serverless connection string.

Autonomous Database Serverless connect strings contain the host name `...(HOST=xyz)...`, which depends on the region. For example, in the Chicago region, the host name would be `adb.us-chicago-1.oraclecloud.com`. If you have replaced the default host name with an IP address or a custom host name, then host name-based DN matching will fail.

The solution is to add a new entry to your `/etc/hosts` file using the original Autonomous Database Serverless domain suffix. Your connection string should then use this new name. For example, add an entry `localtunnel.adb.us-chicago-1.oraclecloud.com` that resolves to your custom IP address and use that name in the JDBC connection string.

2
Using Maven and Gradle

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

Topics:

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

2.1 Using a Maven Project

You can establish a connection to Oracle Autonomous Database using a 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 variable 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 ADBQuickStart.java file to the `src/main/java/com/oracle/jdbctest` directory.
3. Modify the `pom.xml` file and add Oracle JDBC driver as a dependency.

> **Note:**
>
> `ojdbc8-production` will download the Oracle JDBC driver (`ojdbc8.jar`) along with `ucp.jar` (required for using UCP as a client-side connection pool), `oraclepki.jar, osdt_core.jar, and osdt_cert.jar`. These JARs are required for using Oracle Wallets while connecting to Oracle Autonomous Database.

> **See Also:**
>
> Maven Central Guide

```
<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>19.18.0.0</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 `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.ADBQuickStart"
   ```

Sample Output:

The screenshot below shows the queried rows, along with a success message:

```
Enter the password for Autonomous Database:
Available connections after checkout: 4
Borrowed connections after checkout: 1

 Query is SELECT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME, CUST_CITY,CUST_CREDIT_LIMIT FROM SH.CUSTOME
RS WHERE ROWNUM < 20 order by CUST_ID

CUST_ID CUST_FIRST_NAME CUST_LAST_NAME CUST_CITY CUST_CREDIT_LIMIT
-------------------------------------------------------------
3228 Abigail Ruddy Hoofddorp 7000
4117 Abner Everett Clermont-l'Herault 15000
6783 Abigail Ruddy Schimmert 11000
7673 Abner Everett Schwaebisch Gmuend 11000
10338 Abigail Ruddy Scheveningen 1500
13894 Abigail Ruddy Joinville 9000
17449 Abigail Ruddy Nagoya 9000
21005 Abigail Ruddy Santos 3000
24561 Abigail Ruddy Yokohama 7000
25470 Abner Everett Stuttgart 15000
28116 Abigail Ruddy Haarlem 11000
31671 Abigail Ruddy Bolton 1500
35227 Abigail Ruddy Lelystad 9000
36117 Abner Everett Wolverhampton 15000
39672 Abner Everett Murnau 11000
43228 Abner Everett Los Angeles 7000
47006 Abner Everett Montara 11000
49671 Abigail Ruddy Ede 1500
50561 Abner Everett Neuss 7000

Congratulations! You have successfully used Oracle Autonomous Database

Available connections after checkin: 5
Borrowed connections after checkin: 0
```

> **Note:**
>
> If you connect to the Oracle Database from behind a firewall, you will likely encounter a connection timeout error. Make sure to be outside the firewall while running this sample or update the `tnsnames.ora` file to use an HTTPS proxy.

> **See Also:**
>
> *Using Oracle Autonomous Database Serverless* for more information.

2.2 Using a Gradle Project

You can establish a connection to Oracle Autonomous Database using a 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 build and 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 the instructions from the Gradle Guide for Gradle download and build instructions, and set the PATH variable before using Gradle commands. As a first step, create a Gradle project using the following command:

   ```
   gradle init
   ```

   Make sure to choose *2:application* for **Select type of project to generate**. Also, for **Source package (default:temp):** use `com.oracle.jdbctest`.

2. Copy the ADBQuickStart.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 Oracle JDBC driver as a dependency.

   > **Note:**
   >
   > `ojdbc8-production` will download the Oracle JDBC driver (`ojdbc8.jar`) along with `ucp.jar` (required for using UCP as a client-side connection pool), `oraclepki.jar, osdt_core.jar, and osdt_cert.jar`. These JARs are required for using Oracle Wallets while connecting to Oracle Autonomous Database.

   > **See Also:**
   >
   > Maven Central Guide

   - Update the `mainClassName` to *ADBQuickStart*.
   - Add a `run` block to read the password from the console.

```
repositories {
  // Maven Central
   mavenCentral()
 }
dependencies {
  // Get the 19.18.0.0 Oracle JDBC driver along with other companion jars
  implementation("com.oracle.database.jdbc:ojdbc8-production:19.18.0.0")
 }
application {
  // Define the main class for the application
  mainClassName ='{your_project_directory}.ADBQuickStart'
}
```

```
// To pause to read the password from console
run {
  standardInput = System.in
}
```

2.2.2 Building and Running the Gradle Application

This section lists the steps to build and run the Gradle application.

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

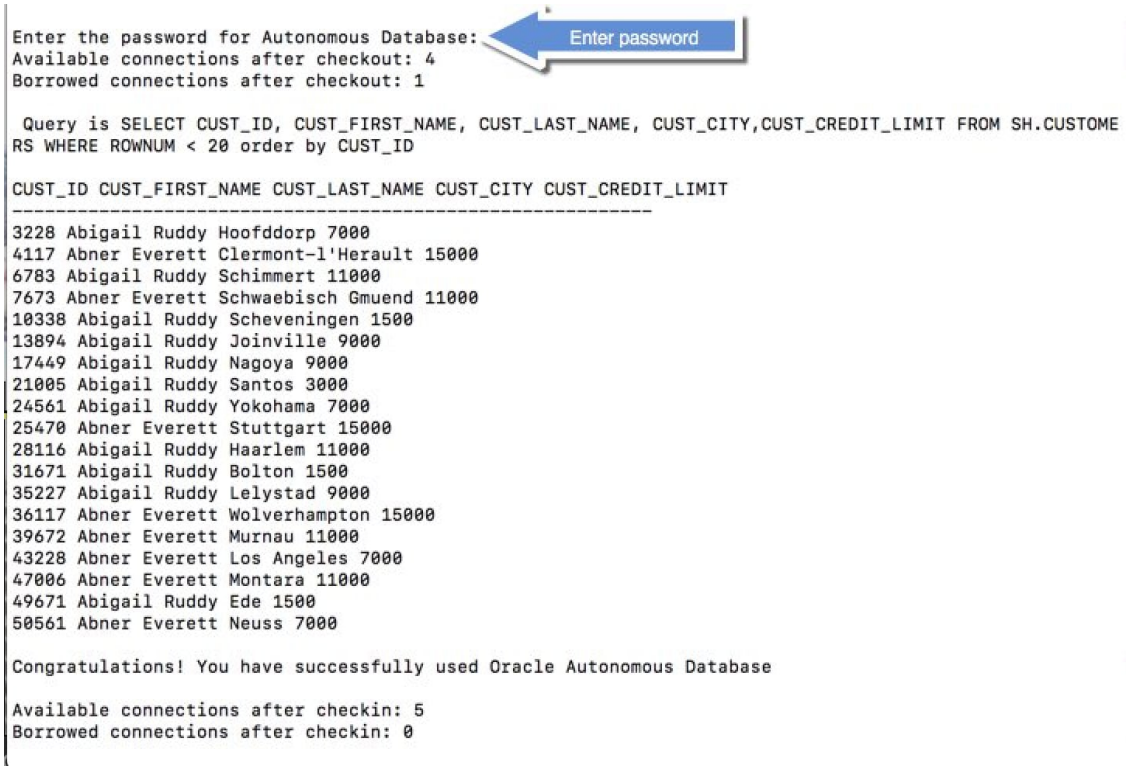
1. Compile the Jave code using the following command:

   ```
   ./gradlew build
   ```

2. Run the sample Java program.

   ```
   ./gradlew run
   ```

Sample Output:
The screenshot below shows the queried rows, along with a success message:



> **Note:**
>
> If you connect to Oracle Autonomous Database from behind a firewall, you will likely encounter a connection timeout error. Make sure to be outside the firewall while running this sample or update the `tnsnames.ora` file to use an HTTPS proxy.

> **See Also:**
>
> *Using Oracle Autonomous Database Serverless* for more information.

3
Using Eclipse and IntelliJ

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

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

3.1 Using the Eclipse IDE

You can establish a connection to Oracle Autonomous 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 **New Maven 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:** `ADBquickstart`
      - **Version:** Leave it as `0.0.1-SNAPSHOT`
2. Create the `ADBQuickStart.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:** ADBQuickStart

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

3. Modify the `pom.xml` file and add Oracle JDBC driver as a dependency.

> **Note:**
>
> `ojdbc8-production` will download the Oracle JDBC driver (`ojdbc8.jar`) along with `ucp.jar` (required for using UCP as a client-side connection pool), `oraclepki.jar, osdt_core.jar, and osdt_cert.jar`. These JARs are required for using Oracle Wallets while connecting to Oracle Autonomous Database.

> **See Also:**
>
> Maven Central Guide

```
<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>19.18.0.0</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 that you do not have any compilation errors in the Java code and you are using the latest JDK version.

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

   You are prompted to enter the database password. Once you enter the password, the results are displayed.

Sample Output:
The screenshot below shows the queried rows, along with a success message:

```
Enter the password for Autonomous Database:
Available connections after checkout: 4
Borrowed connections after checkout: 1

 Query is SELECT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME, CUST_CITY,CUST_CREDIT_LIMIT FROM SH.CUSTOME
RS WHERE ROWNUM < 20 order by CUST_ID

CUST_ID CUST_FIRST_NAME CUST_LAST_NAME CUST_CITY CUST_CREDIT_LIMIT
-------------------------------------------------------------
3228 Abigail Ruddy Hoofddorp 7000
4117 Abner Everett Clermont-l'Herault 15000
6783 Abigail Ruddy Schimmert 11000
7673 Abner Everett Schwaebisch Gmuend 11000
10338 Abigail Ruddy Scheveningen 1500
13894 Abigail Ruddy Joinville 9000
17449 Abigail Ruddy Nagoya 9000
21005 Abigail Ruddy Santos 3000
24561 Abigail Ruddy Yokohama 7000
25470 Abner Everett Stuttgart 15000
28116 Abigail Ruddy Haarlem 11000
31671 Abigail Ruddy Bolton 1500
35227 Abigail Ruddy Lelystad 9000
36117 Abner Everett Wolverhampton 15000
39672 Abner Everett Murnau 11000
43228 Abner Everett Los Angeles 7000
47006 Abner Everett Montara 11000
49671 Abigail Ruddy Ede 1500
50561 Abner Everett Neuss 7000

Congratulations! You have successfully used Oracle Autonomous Database

Available connections after checkin: 5
Borrowed connections after checkin: 0
```

Enter password

Note:

If you connect to Oracle Autonomous Database from behind a firewall, you will likely encounter a connection timeout error. Make sure to be outside the firewall while running this sample or update the `tnsnames.ora` file to use an HTTPS proxy.

See Also:

Using Oracle Autonomous Database Serverless for more information.

3.2 Using the IntelliJ IDE

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

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

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. Click from **File** menu, select **New**, and then select **Project**.
   b. Select **Maven** on the left hand side and select the latest version of JDK as **Project SDK**.
   c. Click **Next**.
   d. Give **Name** as `ADBQuickstart`.
2. Create the `ADBQuickStart.java` file.
   a. Right-click on `src/main/java`.
   b. Select **New**, and then select **Java Class**.
   c. Enter `com.oracle.jdbctest.ADBQuickStart.java`. This will create the required package structure as well. Make sure to copy contents of the `ADBQuickstart.java` file to this new file.
3. Modify the `pom.xml` file and add Oracle JDBC Driver as a dependency.

> **Note:**
>
> `ojdbc8-production` will download the Oracle JDBC driver (`ojdbc8.jar`) along with `ucp.jar` (required for using UCP as a client-side connection pool), `oraclepki.jar, osdt_core.jar, and osdt_cert.jar`. These JARs are required for using Oracle Wallets while connecting to Oracle Autonomous Database.

> **See Also:**
>
> Maven Central Guide

```
<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>19.18.0.0</version>
    <type>pom</type>
  </dependency>
</dependencies>
```

3.2.2 Building and Running ADBQuickStart

This section lists the steps to build and run ADBQuickStart.

1. Compile the Java code.

   Right-click **ADBQuickStart.java**, and then click **Build Module ADBQuickStart**.

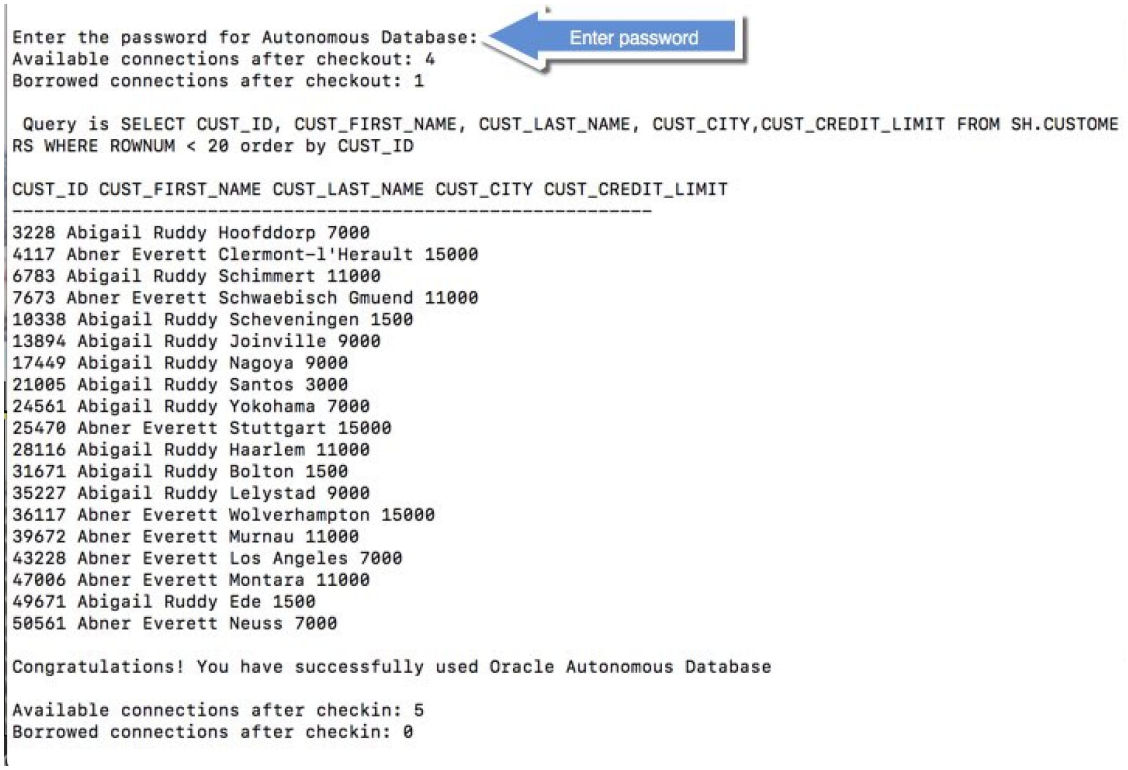
   Make sure that there are no compilation errors.

2. Run the sample Java program.

   Right-click **ADBQuickStart.java**, and then click **Run ADBQuickStart.main()**.

   Make sure to enter the database password on the console.

Sample Output:
The screenshot below shows the queried rows, along with a success message:



> **Note:**
>
> If you connect to Oracle Autonomous Database from behind a firewall, you will likely encounter a connection timeout error. Make sure to be outside the firewall while running this sample or update the `tnsnames.ora` file to use an HTTPS proxy.

> **See Also:**
>
> *Using Oracle Autonomous Database Serverless* for more information.

4
Connecting to Oracle Database Manually

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

Topics:

- 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 the 19c version of ojdbc8-full.tar.gz from OTN and unzip the contents to your classpath.

> **Note:**
>
> `ojdbc8-full.tar.gz` contains the latest JDBC drivers: `ojdbc8.jar`, `ucp.jar` (required JAR for using UCP as a client-side connection pool), `oraclepki.jar`, `osdt_core.jar`, and `osdt_cert.jar`. These JARs are required for using Oracle Wallets while connecting to Oracle Autonomous Database.

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:/lib/oraclepki.jar:./lib/
   osdt_core.jar:./lib/osdt_cert.jar com/oracle/jdbctest/ADBQuickStart.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:/lib/oraclepki.jar:./lib/
   osdt_core.jar:./lib/osdt_cert.jar:. com.oracle.jdbctest.ADBQuickStart
   ```

Sample Output:
The screenshot below shows the queried rows, along with a success message:

```
Enter the password for Autonomous Database:
Available connections after checkout: 4
Borrowed connections after checkout: 1

 Query is SELECT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME, CUST_CITY,CUST_CREDIT_LIMIT FROM SH.CUSTOME
RS WHERE ROWNUM < 20 order by CUST_ID

CUST_ID CUST_FIRST_NAME CUST_LAST_NAME CUST_CITY CUST_CREDIT_LIMIT
-------------------------------------------------------------
3228 Abigail Ruddy Hoofddorp 7000
4117 Abner Everett Clermont-l'Herault 15000
6783 Abigail Ruddy Schimmert 11000
7673 Abner Everett Schwaebisch Gmuend 11000
10338 Abigail Ruddy Scheveningen 1500
13894 Abigail Ruddy Joinville 9000
17449 Abigail Ruddy Nagoya 9000
21005 Abigail Ruddy Santos 3000
24561 Abigail Ruddy Yokohama 7000
25470 Abner Everett Stuttgart 15000
28116 Abigail Ruddy Haarlem 11000
31671 Abigail Ruddy Bolton 1500
35227 Abigail Ruddy Lelystad 9000
36117 Abner Everett Wolverhampton 15000
39672 Abner Everett Murnau 11000
43228 Abner Everett Los Angeles 7000
47006 Abner Everett Montara 11000
49671 Abigail Ruddy Ede 1500
50561 Abner Everett Neuss 7000

Congratulations! You have successfully used Oracle Autonomous Database

Available connections after checkin: 5
Borrowed connections after checkin: 0
```

Enter password

Note:

If you connect to Oracle Autonomous Database from behind a firewall, you will likely encounter a connection timeout error. Make sure to be outside the firewall while running this sample or update the `tnsnames.ora` file to use an HTTPS proxy.

See Also:

Using Oracle Autonomous Database Serverless for more information.