Oracle Database Setup with Docker

A Step-by-Step Guide for Installation and SQL Execution

Get Hands-On Experience with Oracle Database and Object-Relational Mapping

Contents

1 Introduction **2**
1.1 Assumptions 2

2 Prerequisites **2**

3 Installing Oracle Database in Docker **2**
3.1 Pull the Docker Image 2
3.2 Run the Docker Container 3
3.3 Verify Container Status 3
3.4 Create a Sample Application User 3

4 Setting Up GUI Tools **4**
4.1 Primary Tool: Oracle SQL Developer 4
4.1.1 Download and Install 4
4.1.2 Connect to the Database 4
4.1.3 Execute SQL Commands 5
4.2 Alternative Tool: DBeaver 5
4.2.1 Download and Install 5
4.2.2 Connect to the Database 5
4.2.3 Execute SQL Commands 6

5 Executing SQL Commands: Tips **6**

6 Managing the Container **6**
6.1 Starting the Container After Computer Restart 7
6.2 Best Practices for Stopping the Container 7

7 Next Steps **8**

1 Introduction

This guide provides a comprehensive specification for installing Oracle Database 23ai Free (the latest free developer edition, superseding Oracle XE) in a Docker container on your laptop. It covers setting up the database, installing GUI tools (Oracle SQL Developer and DBeaver), and executing SQL commands for hands-on experience, including object-relational mapping (ORM). The setup uses a community-maintained Docker image (`gvenzl/oracle-free:23-full`), which is free and requires no Oracle account.

1.1 Assumptions

- Operating System: Windows, macOS, or Linux.
- Hardware: Minimum 4GB RAM (8GB recommended), 10GB free disk space.
- Internet: Required for downloading Docker images and tools.
- Time Estimate: 30–60 minutes, plus download times.

2 Prerequisites

Before starting, ensure the following are installed:

- **Docker Desktop**: Download from https://www.docker.com/products/docker-desktop. Install and verify with:

  ```
  docker --version
  ```

  Expected output: `Docker version 27.2.0` or similar. Start Docker Desktop.

- **Java JDK 11+**: Required for SQL Developer. Download from https://www.oracle.com/java/technologies/downloads if needed.

3 Installing Oracle Database in Docker

3.1 Pull the Docker Image

Open a terminal (Command Prompt/PowerShell on Windows, Terminal on macOS/Linux) and run:

```
docker pull gvenzl/oracle-free:23-full
```

- Downloads the full edition (5–6 GB) with sample schemas (e.g., HR) for ORM practice.
- Alternatives: `23-slim` (2 GB, fewer schemas) or `latest`.

- Wait 5–30 minutes based on your internet speed.

3.2 Run the Docker Container

Start the container with a secure password (8+ characters, including uppercase, lowercase, digit, special character). Replace `MySecurePassword123` with your choice:

```
docker run --name oracle-db -p 1521:1521 -e ORACLE_PASSWORD=
   MySecurePassword123 -d -v oracle-data:/opt/oracle/oradata gvenzl/oracle-
   free:23-full
```

- `-name oracle-db`: Names the container.
- `-p 1521:1521`: Maps Oracles port to localhost.
- `-e ORACLE`$_P ASSWORD$: $SetsSYS/SYSTEMpassword$.`-d` : $Runsinbackground$.
- `-v oracle-data:/opt/oracle/oradata`: Persists data.

- Optional: Use `-e ORACLE`$_R ANDOM_P ASSWORD = truefo rarandompassword(checklogs)$. Initialization takes 3–10 minutes. Monitor with:

```
docker logs -f oracle-db
```

Look for `DATABASE IS READY TO USE!`.

3.3 Verify Container Status

Check if the container is running:

```
docker ps
```

Confirm `oracle-db` is listed with status `Up`. If not:

- Port conflict: Use `-p 1522:1521` and adjust connections.
- Memory issues: Allocate 4GB+ RAM in Docker settings.

- macOS M-series: Use Colima (`brew install colima; colima start -arch x86`$_6 4 - -memory 4$).

3.4 Create a Sample Application User

Create a non-admin user for ORM practice:

```
docker exec -it oracle-db sqlplus sys/MySecurePassword123@localhost:1521/
   FREEPDB1 as sysdba
```

In SQL*Plus:

```
CREATE USER appuser IDENTIFIED BY app_password;
GRANT CONNECT, RESOURCE TO appuser;
GRANT UNLIMITED TABLESPACE TO appuser;
EXIT;
```

Use appuser/app$_p$*assword for connections*.

4 Setting Up GUI Tools

4.1 Primary Tool: Oracle SQL Developer

4.1.1 Download and Install

- Download from https://www.oracle.com/database/sqldeveloper/technologies/download (e.g., version 24.1).
- Choose ZIP with/without JDK based on Java installation.
- Unzip and run:
  - Windows: sqldeveloper.exe
  - macOS/Linux: sqldeveloper.sh
- If prompted, provide JDK path (Java 11+).

4.1.2 Connect to the Database

- Open SQL Developer, click + to create a connection.
- Settings:
  - Connection Name: OracleDocker
  - Username: appuser (or SYS)
  - Password: app$_p$*assword* (*or* MySecurePassword123) *Role* : SYSDBA *for SYS; blank otherwise*
  - Connection Type: Basic
  - Hostname: localhost
  - Port: 1521
  - Service Name: FREEPDB1
- Click Test Connection, then Connect if successful.

4.1.3 Execute SQL Commands

- Open a SQL Worksheet (File > New > SQL Worksheet).
- Example commands:

```
-- Create table for ORM
CREATE TABLE employees (
  id NUMBER PRIMARY KEY,
  name VARCHAR2(100),
  salary NUMBER
);
-- Insert data
INSERT INTO employees VALUES (1, 'John Doe', 50000);
COMMIT;
-- Query
SELECT * FROM employees;
```

- Run: F9 (single statement) or F5 (script).
- Browse schemas in the Connections panel.

4.2 Alternative Tool: DBeaver

4.2.1 Download and Install

- Download Community Edition from https://dbeaver.io/download (e.g., version 24.2).
- Install:
  - Windows/macOS: Run installer.
  - Linux: Use DEB/RPM or snap/flatpak.
- Add Oracle JDBC driver:
  - Download ojdbc11.jar from https://www.oracle.com/database/technologies/appdev/jdbc-downloads.html.
  - In DBeaver: Database > Driver Manager > Oracle > Libraries > Add File > Select ojdbc11.jar.

4.2.2 Connect to the Database

- Create new connection: Database > New Database Connection > Oracle.
- Settings:
  - Host: localhost

  - Port: 1521
  - Database/SID: Service Name, FREEPDB1
  - Username: appuser (or SYS)
- Password: app$_{p}assword$(orMySecurePassword123)
- Test and connect.

4.2.3 Execute SQL Commands

- Open SQL Editor (right-click connection > SQL Editor).
- Run same example commands as above.
- Execute: Ctrl+Enter (statement) or Ctrl+Shift+Enter (script).
- Use Database Navigator for schema exploration.

5 Executing SQL Commands: Tips

- **Basic Commands**:
  - Create: CREATE TABLE ...
  - Insert: INSERT INTO ... VALUES ...; COMMIT;
  - Query: SELECT * FROM ...;
  - Update/Delete: UPDATE ... SET ...; DELETE FROM ...; COMMIT;
- **ORM Integration**: Use JDBC URL jdbc:oracle:thin:@localhost:1521/FREEPDB1 for Java (Hibernate) or Python (cx_Oracle/SQLAlchemy).
- **Troubleshooting**:
  - Verify container: docker ps
  - Check logs: docker logs oracle-db
  - Firewall: Allow port 1521.
  - Credential errors: Check password case sensitivity.

6 Managing the Container

- **Stop**: To stop the container safely:

```
docker stop oracle-db
```

- **Start**: To restart the container (e.g., after rebooting your computer):

```
docker start oracle-db
```

- **Remove** (caution: only if you want to delete the container; data persists if using volume):

```
docker rm -f oracle-db
```

- **Remove Image** (to free disk space after removing container):

```
docker rmi gvenzl/oracle-free:23-full
```

6.1 Starting the Container After Computer Restart

Since you turn off your computer daily, follow these steps each day to resume working with the Oracle Database:

1. **Start Docker Desktop**: Open Docker Desktop on your computer (Windows, macOS, or Linux). Ensure its running before proceeding.

2. **Start the Container**: Open a terminal (Command Prompt/PowerShell on Windows, Terminal on macOS/Linux) and run:

```
docker start oracle-db
```

   - This restarts the existing oracle-db container, reusing the persisted data (thanks to the -v oracle-data:/opt/oracle/oradata volume).
   - Wait 1-3 minutes for the database to become available. Check status:

```
docker logs oracle-db
```

     Look for DATABASE IS READY TO USE!.

3. **Verify**: Confirm the container is running:

```
docker ps
```

   Ensure oracle-db is listed with status Up.

4. **Connect**: Use SQL Developer or DBeaver with the same connection settings (localhost:1 service name FREEPDB1, user appuser/app$_p$*asswordor*SYS/MySecurePassword123).

6.2 Best Practices for Stopping the Container

To ensure data integrity and system efficiency when stopping the container:

- **Stop Gracefully**: Always use docker stop oracle-db to allow the database to shut down cleanly, preserving data and avoiding corruption.

- **Avoid Forceful Termination**: Do not use docker kill oracle-db or docker rm -f oracle-db unless absolutely necessary (e.g., container is unresponsive), as this may interrupt database processes.
- **Stop Before Shutdown**: If you plan to shut down your computer, stop the container first:

```
docker stop oracle-db
```

  This ensures the database closes properly.
- **Monitor Resources**: If youre not using the database, stop the container to free up memory and CPU (the Oracle container uses 2GB RAM when running).
- **Backup Data**: The oracle-data volume persists data across restarts. To back up, export the database using SQL Developer (Tools > Database Export) or DBeaver (right-click database > Tools > Backup).

7 Next Steps

- Explore sample schemas (e.g., SELECT * FROM HR.EMPLOYEES;).
- Build ORM apps with Hibernate (Java) or SQLAlchemy (Python).
- Contact for help with error messages.