

## How to Install Oracle Database 21c on Docker

To install Oracle Database Enterprise Edition or Standard Edition on Docker, use this procedure.

- [About this Docker Image for Oracle Database](#)  
Review your deployment options for this image.
- [Accessing the Oracle Database Image on Docker](#)  
To access the database image on Docker, you start the container, and then run commands through Docker to access the database.
- [Oracle Enterprise Manager Database Express](#)  
The Oracle Database image in the container includes Oracle Enterprise Manager Database Express (EM Express).
- [Running Scripts After Setup and on Startup](#)  
Learn about how to configure Oracle Database images on Docker to run scripts after setup, and on database startup.
- [Restrictions and Requirements for Oracle Database on Docker](#)  
Be aware of the restrictions that apply to the Docker image, and ensure that your system meets minimum requirements.

## About this Docker Image for Oracle Database

Review your deployment options for this image.

The Oracle Database Enterprise Edition Docker image contains Oracle Database 21c Enterprise Edition, with the option to deploy either Enterprise Edition or Standard Edition, running on Oracle Linux 7 (x86-64). This image contains a default database in a multitenant configuration, with one pluggable database.

### Related Topics

- [Oracle Database 21c Documentation](#)

## Accessing the Oracle Database Image on Docker

To access the database image on Docker, you start the container, and then run commands through Docker to access the database.

 **Note:**

This document provides an overview of the container process for Docker. To obtain the latest information about container images, build container commands, and other procedures for building your own Docker image, refer to the Oracle Database Docker images Github site:

[Oracle Database container images Linux x86-64](#)

- [Connecting to the Oracle Database Server Container](#)  
After the Oracle Database server indicates that the container has started, and the **STATUS** field shows **(healthy)**, client applications can connect to the database.
- [How to Change the Administrative Users Password for the Database](#)  
To change the SYS, SYSTEM, and PDBADMIN user passwords from the defaults to one of your choosing, you use the `setPassword.sh` script provided in the container.
- [Custom Configuration Parameters for Oracle Database on Docker](#)  
To customize your configuration at the time that you start up the Oracle Database image on Docker, you can use this list of custom parameters.
- [How to Use Custom Configuration Parameters for the Database on Docker](#)  
The Oracle Database Docker image enables you to customize the configuration of your database, and to provide initialization parameters for when the database starts.
- [Starting an Oracle Database Server Instance](#)  
To start an Oracle Database server instance you use the `docker run` command, specifying the Docker container that contains the database.

## Connecting to the Oracle Database Server Container

After the Oracle Database server indicates that the container has started, and the **STATUS** field shows **(healthy)**, client applications can connect to the database.

- [Connecting to the Database from Within the Container](#)  
You can connect to Oracle Database server by executing a SQL\*Plus command from within the container
- [Connecting to the Database from Outside the Container](#)  
You can connect to the database outside the container by connecting either to the default port (1521), or to the port you have set to be the exposed port.

## Connecting to the Database from Within the Container

You can connect to Oracle Database server by executing a SQL\*Plus command from within the container

To connect, use one of the following commands, where *dbname* is the database name, *cdb-user-password* is the password for a database user with SYSDBA or SYSTEM system privileges on the CDB, *cdb-sid* is the system identifier of the CDB, *pdb-password* is a user with the PDBADMIN system privileges, and *pdname* is the name of the PDB to which you are connecting.

```
$ docker exec -it dbname sqlplus / as sysdba
```

```
$ docker exec -it dbname sqlplus sys/cdb-user-password@cdb-sid as sysdba
```

```
$ docker exec -it dbname sqlplus system/cdb-user-password@cdb-sid
```

```
$ docker exec -it dbname sqlplus pdbadmin/pdb-user-password@pdname
```

## Connecting to the Database from Outside the Container

You can connect to the database outside the container by connecting either to the default port (1521), or to the port you have set to be the exposed port.

By default, Oracle Database server opens port 1521 for Oracle client connections over the Oracle SQL\*Net protocol. Clients outside the container can connect using either SQL\*Plus, or any Java Database Connectivity (JDBC) interface.

To connect from outside of the container, start the container with the `-p` option, as described in the section "Custom Configurations" in this document.

You can either use the default port as the exposed external port, or map another port as the Docker port. To identify the Docker port to which the Oracle Database port has been mapped, enter the following command, where *db-name* is the Oracle Database name:

```
$ docker port db-name
```

To connect from outside the container using SQL\*Plus, use one of the following commands, where *dbname* is the database name, *cdb-user-password* is the password for a database user with SYSDBA or SYSTEM system privileges on the CDB, *cdb-sid* is the system identifier of the CDB, *pdb-password* is a user with the PDBADMIN system privileges, and *pdname* is the name of the PDB to which you are connecting. In this example, the exposed Docker port for the database is 1521.

```
$ sqlplus sys/cdb-user-password@//localhost:1521/cdb-sid as sysdba
```

```
$ sqlplus system/cdb-user-password@//localhost:1521/cdb-sid
```

```
$ sqlplus pdbadmin/pdb-password@//localhost:1521/pdname
```

## How to Change the Administrative Users Password for the Database

To change the SYS, SYSTEM, and PDBADMIN user passwords from the defaults to one of your choosing, you use the `setPassword.sh` script provided in the container.

On the first startup of the container, if you have not run a custom configuration and provided passwords for database administrator accounts using by using the `-e`, then by default a single random password is generated for these administrative users. You can find this password in the following output line of the database logs:

```
ORACLE PASSWORD FOR SYS, SYSTEM AND PDBADMIN:
```

To change the password for these three database administration accounts, use the Docker `exec` command to run the `setPassword.sh` script that Oracle provides in the container.

### Note:

To use this procedure, the container must be running.

For example, where *dbname* is the database name, and *password* is the password:

```
$ docker exec dbname ./setPassword.sh pdb-password
```

## Custom Configuration Parameters for Oracle Database on Docker

To customize your configuration at the time that you start up the Oracle Database image on Docker, you can use this list of custom parameters.

The supported configuration options are:

- `ORACLE_SID`  
This parameter changes the Oracle system identifier (SID) of the database. This parameter is optional. The default value is set to `ORCLCDB`.
- `ORACLE_PDB`  
This parameter modifies the name of the pluggable database (PDB). This parameter is optional. The default value is set to `ORCLPDB1`.
- `ORACLE_PWD`  
This parameter modifies the password for the SYS, SYSTEM and PDBADMIN administration users. This parameter is optional. The default value is randomly generated. After configuration, you can change the administration user password as described in his password can be changed later as described in "How to Change the Administrative Users Password for the Database."

- `INIT_SGA_SIZE`

This parameter modifies the memory in MB that should be used for all SGA components. This parameter is optional. If you have not provided an SGA size value, then the default value is calculated during database creation. After the database is configured, you can change the SGA value later as described in "How to Set the SGA and PGA Memory."

- `INIT_PGA_SIZE`

This parameter modifies the target aggregate memory in MB of the Program Global Area, or PGA, that you want to be used for all server processes attached to the instance. This parameter is optional. If you have not provided a PGA value, then the default value is calculated during database creation. After the database is configured, you can change the PGA value later as described in "This parameter is optional. After the database is configured, you can change the PGA value later as described in "How to Set the SGA and PGA Memory."

- `ORACLE_EDITION`

This parameter selects the Oracle Database edition when the container is started for the first time. This parameter is optional. The options are `enterprise`, for Enterprise Edition, and `standard`, for Standard Edition. The default value is `enterprise`.

- `ORACLE_CHARACTERSET`

This parameter modifies the national language character set of the database. This parameter is optional. The default value is set to `AL32UTF8`.

## How to Use Custom Configuration Parameters for the Database on Docker

The Oracle Database Docker image enables you to customize the configuration of your database, and to provide initialization parameters for when the database starts.

Oracle Database server container also provides configuration parameters that can be used when starting the container. The following example provides the syntax for a detailed docker run command supporting all custom configurations, with variable values in *Italics font*, which you can replace with values for your deployment:

```
docker run -d --name container_name \
  -p host_port:1521 -p host_port:5500 \
  -e ORACLE_SID=cdb-system-identifer \
  -e ORACLE_PDB=pdb-name \
  -e ORACLE_PWD=oracle-user-password \
  -e INIT_SGA_SIZE=cdb-database-sga-memory-in-mb \
  -e INIT_PGA_SIZE=cdb-database-pga-memory-in-mb \
  -e ORACLE_EDITION=ee-or-se-database-edition \
  -e ORACLE_CHARACTERSET=character-set \
  -e ENABLE_ARCHIVELOG=[true|false] \
  -v [host-mount-point]:/opt/oracle/oradata \
  container-registry.oracle.com/database/enterprise:21.3.0
```

Parameters:

- name: The name of the container. (Default: auto-generated)
- p: The port mapping of the host port to the container port.
  - Two ports are exposed: 1521 (Oracle Listener), 5500 (OEM Express)
- e ORACLE\_SID: The Oracle Database SID that should be used. (Default:ORCLCDB)
- e ORACLE\_PDB: The Oracle Database PDB name that should be used. (Default: ORCLPDB1)
- e ORACLE\_PWD: The Oracle Database SYS, SYSTEM and PDBADMIN password. (Default: auto-generated)
- e INIT\_SGA\_SIZE: The total memory in MB that should be used for all SGA components (Optional)
- e INIT\_PGA\_SIZE: The target aggregate PGA memory in MB that should be used for all server processes attached to the instance (Optional)
- e ORACLE\_EDITION: The Oracle Database Edition [enterprise|standard]. (Default: enterprise)
- e ORACLE\_CHARACTERSET: The character set that you want used when creating the database. (Default: AL32UTF8)
- e ENABLE\_ARCHIVELOG The ARCHIVELOG mode. By default, set to false.
  - If set to true, then ARCHIVLOG mode is enabled in the database (for fresh database creation only)
- v /opt/oracle/oradata
  - The data volume that you want used for the database. Must be writable by the oracle user (uid: 54321) inside the container
  - If omitted, then the database will not be persisted over container recreation.
- v /opt/oracle/scripts/startup | /docker-entrypoint-initdb.d/startup
  - Optional: A volume with custom scripts to be run after database startup.
  - For further details see the section "Running scripts after setup and on startup" section below.
- v /opt/oracle/scripts/setup | /docker-entrypoint-initdb.d/setup
  - Optional: A volume with custom scripts that you want run after database setup.
  - For further details see the "Running scripts after setup and on startup" section below.

## Starting an Oracle Database Server Instance

To start an Oracle Database server instance you use the docker run command, specifying the Docker container that contains the database.

To start the container, run the following command from the command prompt, where *oracle-db* is the name of the container that contains the Oracle Database:

```
$ docker run -d --name oracle-db
  container-registry.oracle.com/database/enterprise:21.3.0.0
```

To find the automatically generated default password for connecting to the Oracle Database server instance, enter the following command: can be found from the logs using

```
$ docker logs oracle-db
```

To obtain the SYS user password, use the `setPassword.sh` script provided in the container.

### Related Topics

- [How to Change the Administrative Users Password for the Database](#)  
To change the SYS, SYSTEM, and PDBADMIN user passwords from the defaults to one of your choosing, you use the `setPassword.sh` script provided in the container.

## Oracle Enterprise Manager Database Express

The Oracle Database image in the container includes Oracle Enterprise Manager Database Express (EM Express).

You can use EM Express to assist you with administering the database.

To access EM Express, start your browser in the container, and open the following URL:

<https://localhost:5500/em/>

- [How to Reuse the Existing Database](#)  
After you initially start up the database, you can reuse the Docker data volumes created during initial configuration.
- [SGA and PGA Memory Configuration for the Database on Docker](#)  
You can choose either to size the SGA and PGA when the initial database is created, or use the automatic values generated during database creation.

## How to Reuse the Existing Database

After you initially start up the database, you can reuse the Docker data volumes created during initial configuration.

The Oracle Database server image uses Docker data volumes to store data files, redo logs, audit logs, alert logs, and trace files. The data volume is mounted within the container at `/opt/oracle/oradata`.

To start the Oracle Database image with a data volume, use the `docker run` command, where `dbname` is the database name:

```
$ docker run -d --name dbname -v
OracleDBData:/opt/oracle/oradata
container-registry.oracle.com/database/enterprise:21.3.0
```

In this example, `OracleDBData` is the data volume that is created by Docker. The data volume is mounted within the container in the path `/opt/oracle/oradata`.

During the first instance creation, data files are created in the new data volume. After the container is destroyed, the data files are kept persistent, and can be reused to start another container with the same data files if the same `ORACLE_SID` parameter is provided.

 **Note:**

When you use data volumes created by Docker, the file system that the Docker container uses affects the performance of Oracle Database. For better performance, Oracle recommends that you select the `ext4` file system, instead of the `btfs` file system.

To use a directory on the host system for the data volume, run the following command, where `dbname` is the CDB name, and `writable-directory-path` is the directory path where you want to place the directory:

```
$ docker run -d --name dbname -v
writable-directory-path:/opt/oracle/oradata
container-registry.oracle.com/database/enterprise:21.3.0
```

 **Note:**

If you provide `standard` as the value for the `ORACLE_EDITION` parameter while creating the data files for the first time, then you must provide the same value when reusing those data files to start a new container.

## SGA and PGA Memory Configuration for the Database on Docker

You can choose either to size the SGA and PGA when the initial database is created, or use the automatic values generated during database creation.

If you want to set the values for the system global area (SGA) and program global area (PGA) memory yourself, then you can define these sizes by specifying the values in megabytes (MB) the first time the database is created. To set the values yourself, specify the `INIT_SGA_SIZE` and `INIT_PGA_SIZE` parameters at the time that you run

the docker run command. You must provide the values in MB, without any units appended to the values. For example:

```
-e INIT_SGA_SIZE=1536
```

These parameters are optional. If you don't provide values for the SGA and PGA, then the values are calculated automatically during database creation.

After the initial database is created, even if you specify `INIT_SGA_SIZE` and `INIT_PGA_SIZE` in the docker run command while reusing existing data files, and even though the values are visible in the container environment, the values are not set inside the database. The values used at the time of the initial database creation continue to be used.

## Running Scripts After Setup and on Startup

Learn about how to configure Oracle Database images on Docker to run scripts after setup, and on database startup.

At the time of this release, you can run shell scripts (`.sh`) and SQL scripts (`.sql`) during database setup. To run these scripts, mount the volume `/opt/oracle/scripts/setup`, and include the scripts in this directory.

To run scripts after database startup, mount the volume `/opt/oracle/scripts/startup`, and include to include scripts in this directory.

Both the setup directory and startup directory locations are also represented by the symbolic link `/docker-entrypoint-initdb.d`. This symbolic link provides consistency with other database docker images. You can decide whether to put the setup and startup scripts under `/opt/oracle/scripts`, or `/docker-entrypoint-initdb.d`.

### Note:

Startup scripts are run after the initial database creation is complete. Setup scripts are run only at the time of the initial database creation.

After the database is set up or started, the scripts in those folders are run on the database in the container. SQL scripts are run using the `SYSDBA` privileges (`as sysdba`), and shell scripts are run with the group privileges of the current user.

To ensure that scripts run in correct order, Oracle recommends that you prefix your scripts with a number. For example:

```
01_users.sql  
02_permissions.sql
```

In the following example, the docker command starts up the database named `dbtest`, and mounts the local directory `myScripts` to `/opt/oracle/scripts/startup`, which is then searched for custom startup scripts:

```
$ docker run -d --name dbtest -v
/home/oracle/myScripts:/opt/oracle/scripts/startup
container-registry.oracle.com/database/enterprise:21.3.0
```

## Restrictions and Requirements for Oracle Database on Docker

Be aware of the restrictions that apply to the Docker image, and ensure that your system meets minimum requirements.

### Restrictions for the Oracle Database Docker Image

This docker image has the following restrictions:

- This docker image release supports only a single database instance.
- Oracle Data Guard is not supported.

### Requirements for the Oracle Database Docker Image

The Docker container environment must meet the following minimum requirements:

- 21 GB of available disk space
- 2 GB of available memory.

## 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.

Copyright © 2018, 2024, Oracle and/or its affiliates

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.