APPX LIBRARY

Oracle 19c with APPX 6.2.1 on Rocky Linux

Oracle 19c (19.3) · Rocky Linux 9.8 · APPX 6.2.1 (64-bit engine)

> Set up Oracle as an APPX file system using FMS type 5. Start with one file, then save a record in APPX and check it in Oracle.

APPX Software, Inc.

Made on November 18, 2026 from the pages of the APPX Library, https://wiki.appx.com/manuals/library/rdbms-oracle-19-3-rocky-9-8-appx-621-x64/. The pages are the current version.

Contents

Chapter 1 **What to install** 3

Chapter 2 **Choose a pluggable database and create a login** 4

Chapter 3 **Set up Oracle Instant Client** 5

Chapter 4 **Set up the FMS group and identity** 6

Chapter 5 **Assign one file and create the table** 8

Chapter 6 **Save and check a record** 10

Chapter 7 **Troubleshooting** 11

CHAPTER 1

What to install

This example was tested with the 64-bit APPX 6.2.1 engine on Rocky Linux 9.8, Oracle Database 19c Enterprise Edition (19.3), and Oracle Instant Client 19.28 Basic Light. APPX and Oracle ran on the same host. If Oracle already runs elsewhere, install Instant Client on the APPX server and use the Oracle host in the connection alias.

Component	Download
APPX server	Install the 64-bit Linux engine. APPX 6.2.1 Linux server installer · Installation instructions
APPX Desktop Client	Use the 6.2.1 Desktop Client to open the APPX session. Linux Desktop Client installer · Other desktop platforms
Oracle Database	For a new database server, Oracle supplies `LINUX.X64_193000_db_home.zip` and the 19c Enterprise Edition RPM. Check the platform and release update requirements before installing. Oracle Database 19c for Linux x86-64 · Oracle installation guide
Oracle Instant Client	Install the 64-bit packages on the APPX server. The test used Basic Light and SQL*Plus 19.28. 19.28 Basic Light ZIP · 19.28 SQL*Plus ZIP · Oracle Instant Client downloads

The public Database Home download is the 19.3 base image. For a new installation on Oracle Linux 9 or Red Hat Enterprise Linux 9, follow Oracle's release update instructions. The Rocky Linux 9.8 and 19.3 combination above describes the lab used for this guide. Oracle's Linux 9 installation and release update instructions

CHAPTER 2

Choose a pluggable database and create a login

Ask your DBA for the listener host and port, the service name for the pluggable database (PDB), and the tablespace for APPX tables. On a local server, a DBA can check the service names with `lsnrctl status`.

Example PDB	Database character set	Use
APPXMB	`AL32UTF8`	Multibyte database used in this walkthrough.
APPXSB	`WE8MSWIN1252`	Separate single-byte database for an application that needs that character set.

Connect as a DBA. Run these commands in SQL*Plus from the container database, substituting your PDB name and a password for your test schema. This example uses the `USERS` tablespace and a small quota.

```
ALTER SESSION SET CONTAINER = APPXMB;
SHOW CON_NAME

SELECT value FROM nls_database_parameters
WHERE parameter = 'NLS_CHARACTERSET';

CREATE USER APPX_ORA_DEMO IDENTIFIED BY "DemoOra19_42"
    DEFAULT TABLESPACE USERS QUOTA 25M ON USERS;
GRANT CREATE SESSION, CREATE TABLE TO APPX_ORA_DEMO;
```

Use the same PDB for the schema and the APPX connection. In this example the character set query returns `AL32UTF8`. The record test below uses ASCII text. Oracle CREATE USER

CHAPTER 3

Set up Oracle Instant Client

1. On the APPX server, install `libaio` and unpack the 64-bit Basic Light and SQL*Plus ZIPs into the same directory. The ZIPs create `instantclient_19_28`. This example uses `/opt/oracle/instantclient_19_28`. Oracle Instant Client ZIP installation
2. Put `tnsnames.ora` in a directory the APPX engine can read, such as `/opt/appx/oracle-net`. Define the alias below. Change the host when Oracle runs elsewhere. The service name in the descriptor must match the PDB service registered with the listener.

```
APPXMB =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 127.0.0.1)(PORT = 1521))
    (CONNECT_DATA = (SERVICE_NAME = appxmb))
  )
```

For a separate single-byte PDB, use an `APPXSB` alias with that PDB's service name, such as `appxsb`.

3. Set these values in the APPX engine's `appx.env`. Use the path where you unpacked Instant Client.

```
APPX_OCI_DIR=/opt/oracle/instantclient_19_28
APPX_OCI_LIB=libclntsh.so.19.1
TNS_ADMIN=/opt/appx/oracle-net
```

The Linux loader also needs to find the client's dependent libraries. Use the loader configuration on the APPX server. On a dedicated client host, you can register the Instant Client directory in `/etc/ld.so.conf.d/` and run `ldconfig`. For a shell test, set `LD_LIBRARY_PATH` for that command.

```
IC=/opt/oracle/instantclient_19_28
TNS_ADMIN=/opt/appx/oracle-net LD_LIBRARY_PATH="$IC" \
  "$IC/sqlplus" -L APPX_ORA_DEMO@APPXMB
```

Enter the schema password at the prompt. Confirm the connection goes to the right PDB before opening APPX.

```
SELECT USER, SYS_CONTEXT('USERENV', 'CON_NAME') AS pdb
FROM dual;
```

CHAPTER 4

Set up the FMS group and identity

1. Open a new APPX Desktop Client session. Choose **System Administration → File Management → FMS Groups**.
2. Add `APPX-ORACLE` and enter FMS type **5**. Click **FMS Group Attributes**.
3. Enter these values and save the group options. Leave the other options at their defaults for this example.

FMS option	Value
Server Name	`APPXMB`
Table Naming Scheme	`APPX_$db_$appl_$file`
Tablespace	`USERS`

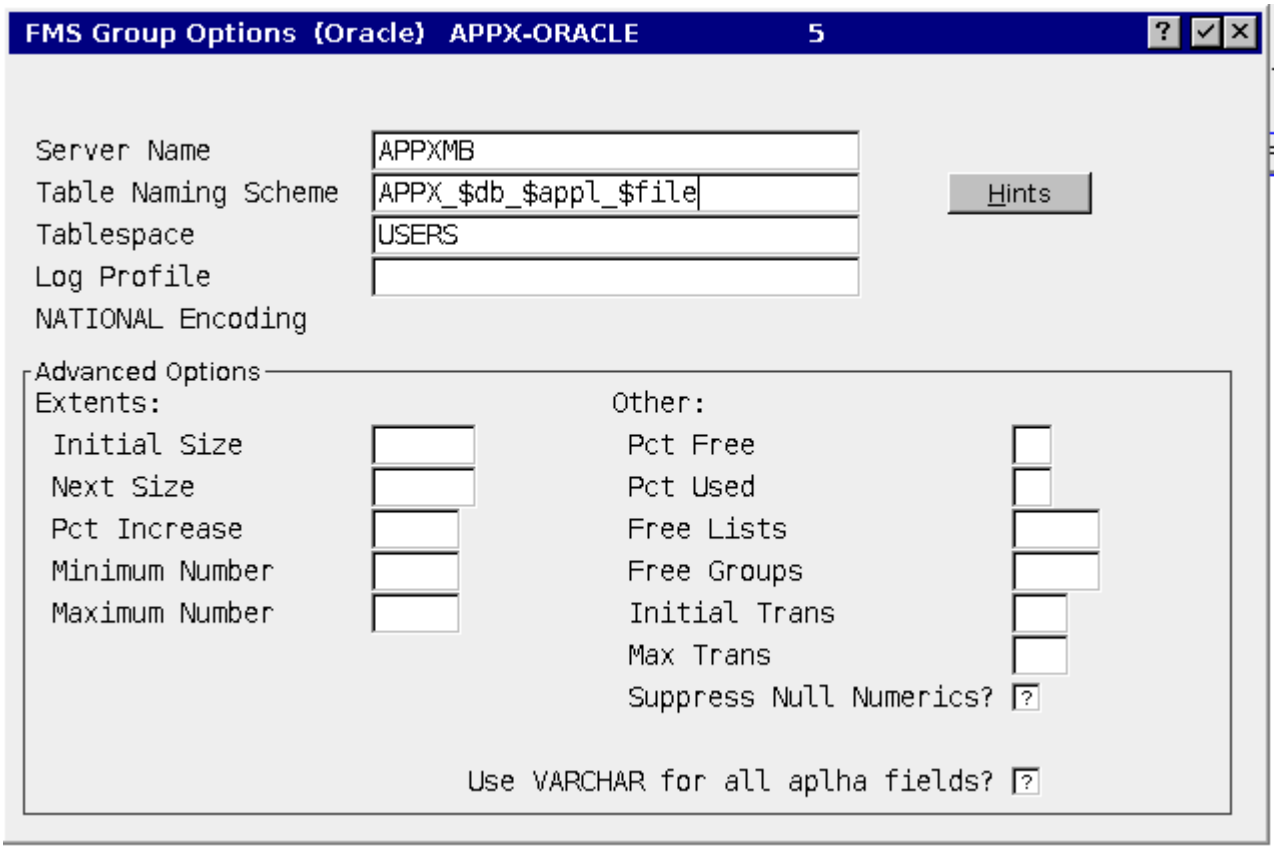


Figure 1. Oracle FMS Group Options with APPXMB, the table naming scheme, and the USERS tablespace.

4. On the group row, click **Identities**. Leave **APPX User** blank for the default identity. Press `Enter` to accept the blank key and again to open the details. To set credentials for just one APPX user, enter that user's ID instead.

5. Enter `APPX_ORA_DEMO` in **User Id** and leave **Realm** blank. Click **Change Password**, enter the schema password twice, and click **Change Password** again. When APPX reports **Password Changed, END to Exit**, close the password dialog and save the identity.
6. Open a new APPX session before testing. APPX can keep the earlier settings for the life of a session.

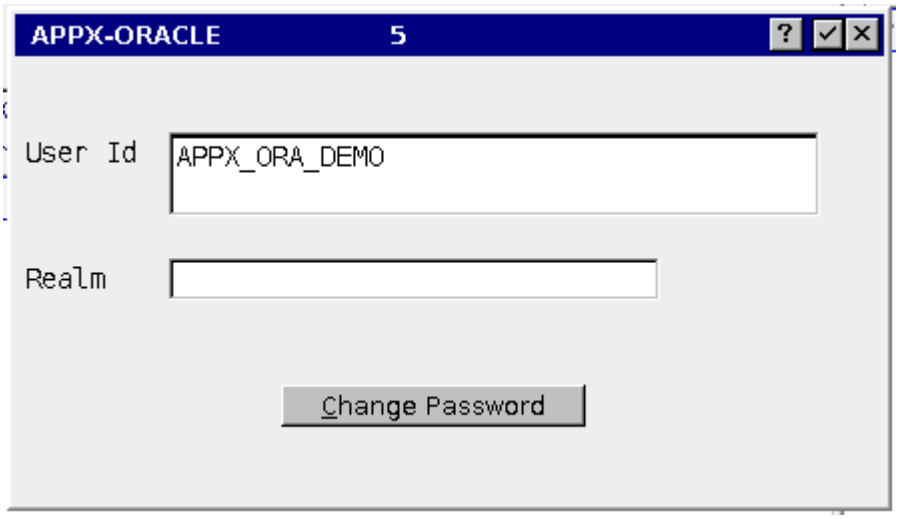


Figure 2. The Oracle identity uses APPX_ORA_DEMO and a blank Realm.

> **FMS type 5 uses Oracle row locks.** No proxy database ID is needed. In the test, a second Oracle session could read the Canada row while APPX held it in Change mode. A request to lock that row with `FOR UPDATE NOWAIT` returned `ORA-00054` until APPX released it.

CHAPTER 5

Assign one file and create the table

Use a test database and a file you can create empty. This example uses the Demo Application's `COUNTRY` file.

1. Open **System Administration → Runtime Configuration → Database Definitions.** Add `TST` with the company name **Oracle Test**. Set the startup application to `DMO`, version `00`. Save and accept the report defaults.
2. For `TST`, open **Related Applications.** Add `DMO` in **Live Operations.** Select the `DMO` row and click **Database Management.**
3. In **File Selection**, click **Unselect All**, then **Select.** Enter `COUNTRY` in **Name** and accept the selection. Return to **Data File Management** and check that it says **1 of 11 Files Currently Selected.**
4. Open **File Specifications** and set **FMS** to **5** and **FMS Group** to `APPX-ORACLE` for `COUNTRY`. Save and return.
5. Choose **Create Files.** The result for `COUNTRY` should say **Created.**

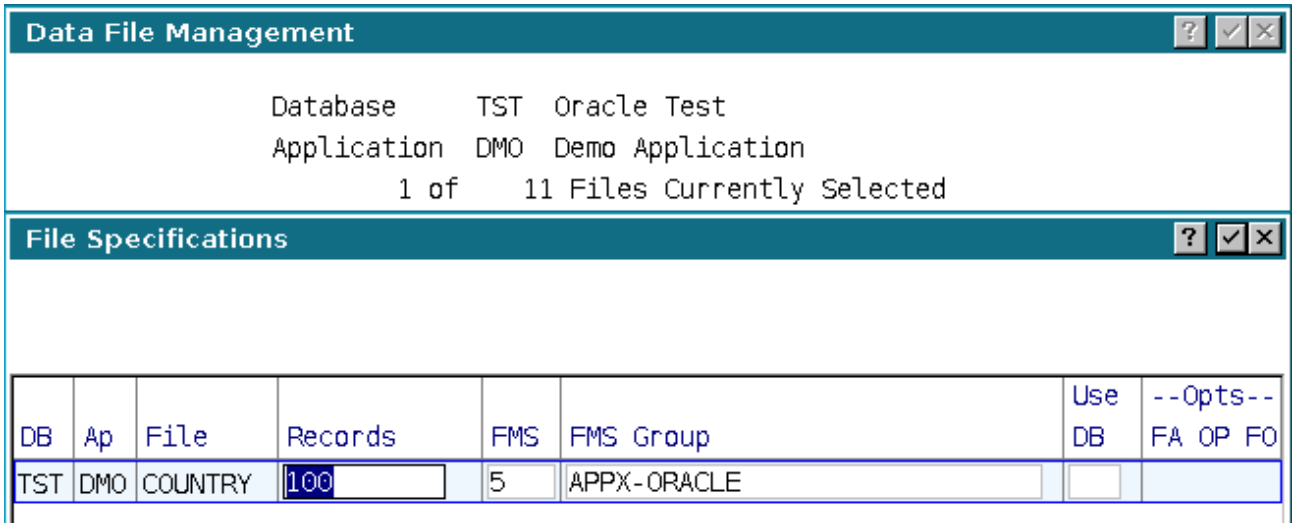


Figure 3. File Specifications for TST DMO COUNTRY show FMS 5 and group APPX-ORACLE.

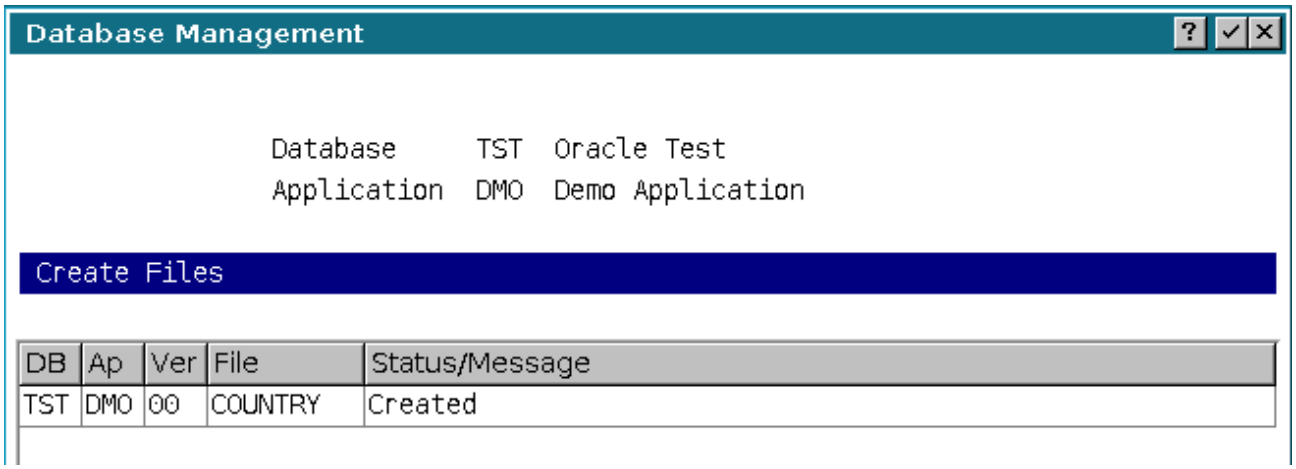


Figure 4. Database Management reports Created for COUNTRY.

Oracle stores the table as `APPX_ORA_DEMO.APPX_TST_DMO_COUNTRY`. APPX also created a unique index on `COUNTRY_NAME`.

To move an existing APPX/IO file, back it up and export it before changing its FMS type. Rehearse the move on a copy. Assign the Oracle FMS group, create the target table and import the export. Compare the record count and check a few keys in APPX before using the converted file. Move an existing APPX/IO file into Oracle

Field types in Oracle

See APPX field type mappings for the column types APPX created. In SQL*Plus, run `DESC APPX_TST_DMO_COUNTRY` for a quick check. Query `USER_TAB_COLUMNS` for character length semantics, numeric precision, and scale.

CHAPTER 6

Save and check a record

1. From the APPX main menu, choose **Run an Application** and execute database `TST`. In the demo's top menu, choose **Setup → Countries**.
2. In **Country Address Format Rules**, choose **Add**. Enter `CANADA` for **Country Name**, `PROVINCE` for **Region Label**, and `2` for **# Words In Postal Code**. Leave **Postal Code Location** at **AFTER REGION**. Click **Save**.
3. Open a fresh Desktop Client session, run `TST`, and go back to **Setup → Countries**. In **Inquire** mode, look up `CANADA` and check that the saved values come back.

With only `COUNTRY` created, the demo may report **Cannot Access Required File** for `PROSPECT`. Click **Cancel** on that message to continue the Countries test. Create the full demo database if you want to run the rest of the application.

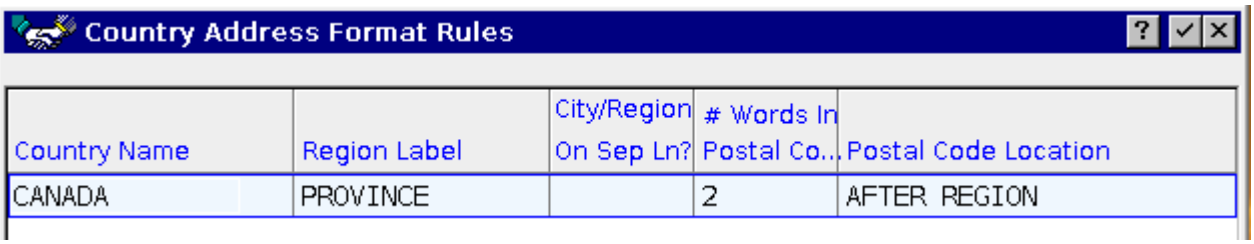


Figure 5. A fresh APPX session displays CANADA, PROVINCE, and 2 in Inquire mode.

In SQL*Plus, connect as `APPX_ORA_DEMO` through `APPXMB` and check the same key.

```
SELECT RTRIM(country_name) AS country_name,
       RTRIM(country_region_label) AS region_label, country_no_words
FROM appx_tst_dmo_country
WHERE country_name = 'CANADA';
```

The lab returned one row with `CANADA`, `PROVINCE`, and `2`.

CHAPTER 7

Troubleshooting

Check the APPX engine

This guide uses **FMS type 5 (Oracle)** through the Oracle Call Interface (OCI).

In APPX **Release Information**, record **Engine Release** and **Compile Date**. The application release numbers on the same screen identify the APPX system applications. Match the database client or ODBC driver to the bitness of the engine that runs the operation. Check the settings on the APPX server and under the Windows or Linux account that runs the engine.

Test the connection outside APPX

On the APPX server, use SQL*Plus with the same Oracle alias and database login as APPX. Check `TNS_ADMIN` and the alias for `ORA-12154`. For `ORA-12514`, check the service name with `lsnrctl status` on the database server and confirm the PDB is open.

If APPX cannot initialize Oracle, check `APPX_OCI_DIR` and `APPX_OCI_LIB`. Use `file` to confirm that the engine and `libclntsh.so.19.1` are both 64-bit. Use `ldd` to check loader dependencies such as `libaio`.

Check the FMS group and Identity

Confirm the FMS group's **Server Name** matches the alias that worked in SQL*Plus.

Under **Identities**, look for an active, unexpired entry for the APPX user running the operation. That entry takes precedence over the blank default. Check **User Id** and re-enter the database password with **Change Password**. Save the Identity and open a fresh APPX session. APPX can cache connection and Identity settings for the life of the session.

For `ORA-01017`, check the password and confirm the user exists in the PDB behind the alias.

Check where APPX creates the table

If the connection succeeds, read the **Create Files** result or open **View Log** for the database error. Check the database, login, and target schema or database with the same connection settings as APPX. Compare the APPX database ID, application ID, file name, and **Table Naming Scheme** with the SQL table name.

Check the PDB, the Oracle user's current schema, `CREATE TABLE`, and the schema's quota on the selected tablespace. `ORA-01950` points to a quota problem.

Capture one failing operation

1. Make a copy of the `appx.env` under the effective `APPXPATH` of the engine that runs the process. Create a private log directory that the engine account can write to, then temporarily set the variables below. Keep `$(pid)` as written. The SQL logger replaces it with the engine process ID.

   ```
   APPX_SQL_CMD=/var/tmp/appx-trace/appx-rdbms-$(pid).log
   APPX_DBG_CODE=0x1F
   ```

   The mask includes generated SQL, return statuses, fetch attempts, and engine and connection operations. Use `0xFFFFFFFF` for one attempt if support needs more detail. The engine reads these settings on startup. `APPX_ODBC_LOG` and `APPX_ODBC_CODE` apply to the driver used by external programs to read APPX data.

2. Keep the Oracle client alias and FMS group you want to diagnose. Use the APPX SQL log and the Oracle error from the failing operation. APPX Oracle debugging
3. Open a fresh APPX session and reproduce the failure once. For **Create Files**, select only the intended test file. For a failed `READ` with `HOLD`, note the APPX error and flags before the next `WRITE`. Try a plain `READ` of the same key in another fresh session. If the log is missing, check `APPXPATH`, the effective variables, and the engine account's access to the log directory.
4. Exit the test sessions and restore the previous `appx.env` settings.

If an insert follows a failed held read, keep the first error. A later duplicate-key error indicates that the key existed when the insert ran. Use the insert branch only when the read explicitly reports that the record was absent. Handle lock and database errors before trying another write.

Share the SQL statement, fetch or lock result, and the first SQLSTATE, native error number, and message when available. Include the APPX flags, file and operation, timestamp and time zone, APPX engine release, build date and bitness, operating system, driver name and version, and database versions before and after an upgrade. Keep the SQL structure and error codes intact and replace passwords and record values with placeholders. The APPX SQL log can contain the engine environment and full connection strings. APPX environment settings