APPX LIBRARY

Oracle 19c with APPX 5.5.4 on Rocky Linux

Oracle 19c (19.3) · Rocky Linux 9.8 · APPX 5.5.4 (32-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-554-x86/. The pages are the current version.

Contents

Chapter 1 **What to install** 3

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

Chapter 3 **Set up the 32-bit Oracle connection** 5

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

Chapter 5 **Assign one file and create the table** 7

Chapter 6 **Save and check a record** 8

Chapter 7 **Troubleshooting** 9

CHAPTER 1

What to install

The example uses a 32-bit APPX 5.5.4 engine on Rocky Linux 9.8 and a 64-bit Oracle 19c (19.3) database. Install the 32-bit Oracle Instant Client 19c client on the APPX server.

Component	Download
APPX server	Install the 32-bit APPX 5.5.4 engine. On x86-64 Rocky Linux, use the installer for 64-bit Linux. It installs the 32-bit APPX engine. APPX 5.5.4 Linux server installer · Installation instructions
APPX Desktop Client	Use the 5.5.4 Desktop Client to open an APPX session. Desktop Client installer
Oracle	Install Oracle 19c or use an existing server with a dedicated test database. Database installation
Oracle client	Match the client to the APPX engine's 32-bit architecture. Oracle Instant Client 19c 19.25

Check the APPX product registration for the Oracle interface. The example uses OCI and FMS type `5`.

CHAPTER 2

Choose a database and create a login

Ask your DBA for the host, port, and service name. This single-byte example uses `APPXSB` with `WE8MSWIN1252`. Use a dedicated schema or database to rehearse the setup. Replace the example password with a generated value.

Connect as a DBA to the `APPXSB` service. If this is a pluggable database, connect to the PDB's service. Check its character set before creating the schema owner.

```
SELECT value FROM nls_database_parameters
WHERE parameter = 'NLS_CHARACTERSET';

CREATE USER APPX_ORA554_DEMO IDENTIFIED BY "use_your_test_password"
    DEFAULT TABLESPACE USERS QUOTA 25M ON USERS;
GRANT CREATE SESSION, CREATE TABLE TO APPX_ORA554_DEMO;
```

CHAPTER 3

Set up the 32-bit Oracle connection

Set the APPX engine's OCI directory to the 32-bit Instant Client. Create a `tnsnames.ora` alias named `APPXSB` for the tested service under `TNS_ADMIN`.

```
APPX_OCI_DIR=/opt/oracle/instantclient_19_25
APPX_OCI_LIB=libclntsh.so.19.1
TNS_ADMIN=/opt/appx/oracle-net
NLS_LANG=AMERICAN_AMERICA.WE8MSWIN1252
```

```
APPXSB =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 127.0.0.1)(PORT = 1521))
    (CONNECT_DATA = (SERVICE_NAME = appxsb))
  )
```

Put this Instant Client directory on `LD_LIBRARY_PATH` for the APPX engine and SQL*Plus. Sign in with `sqlplus -L APPX_ORA554_DEMO@APPXSB`. Use `SHOW USER` and `SELECT SYS_CONTEXT('USERENV', 'SERVICE_NAME') FROM dual;` to confirm the schema and service.

ORA-12154. Confirm that the running APPX engine sees the same `TNS_ADMIN` and alias as SQL*Plus. Restart the engine session after correcting its environment.

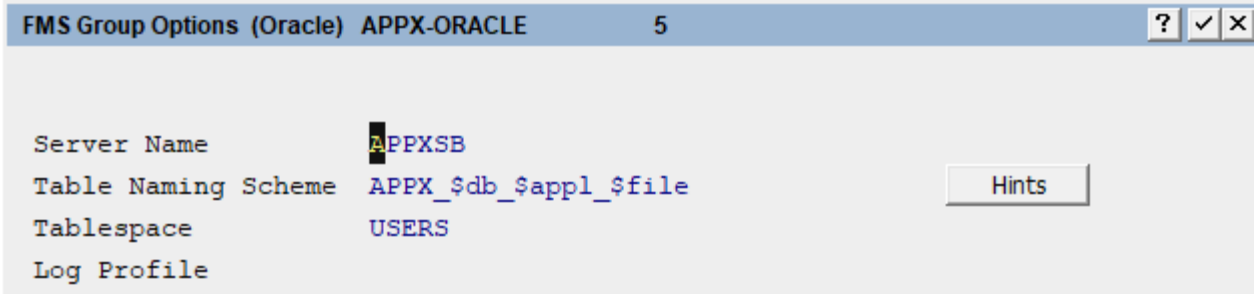
CHAPTER 4

Set up the FMS group and identity

Open a new APPX Desktop Client session. Choose **System Administration → Configuration → File System Groups** under **File Maintenance**.

1. Add `APPX-ORACLE` and enter FMS type `5`.
2. Click **FMS Group Attributes.** Enter the values below and save.

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



The FMS group options for this connection.

On the group row, click **Identities.** Enter your APPX user ID to give that user a database login. Leave **APPX User** blank to set a default identity for the group. Press Enter to accept the key and again to open the details.

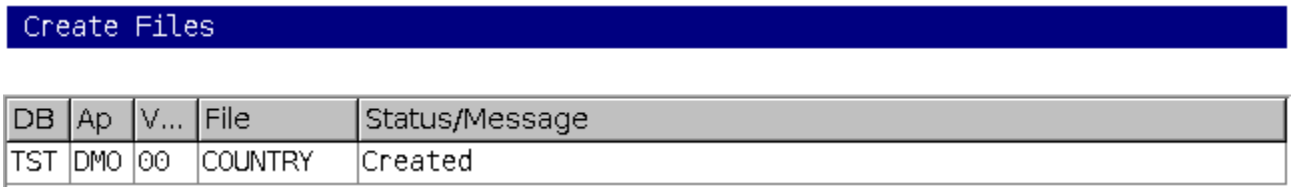
Enter `APPX_ORA554_DEMO` in **User Id** and leave **Realm** blank. Save the identity and reopen it in **Change** mode. Click **Change Password**, enter the database password, and press `Tab` to move to **Confirm Password.** Enter it again and click **Change Password.** When APPX reports **Password Changed, END to Exit**, close the password dialog and save the identity. Open a fresh APPX session before testing.

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 → Database/Applications → Databases / Applications** under **File Maintenance**. 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** if it is absent. Select the `DMO` row and click **Database Management**. If APPX prompts for a **Language ID**, set `ENG` and reopen **Database Management**.
3. In **File Selection**, click **Unselect All**. Highlight `COUNTRY` and press `F3`. If you use the **Select** button, enter `COUNTRY` in **Name** and confirm. Return to **Data File Management** and confirm that one file is selected.
4. Open **File Specifications**. 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**.



Create Files reports Created for COUNTRY.

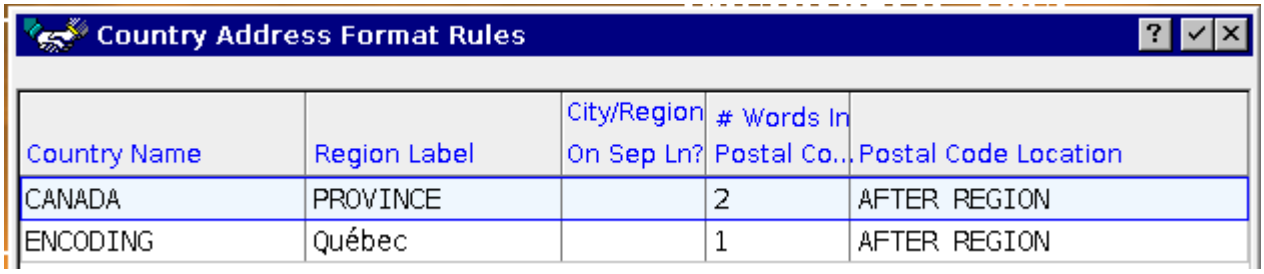
This naming scheme creates `APPX_TST_DMO_COUNTRY` in the selected database or schema. For an existing file, follow the APPX/IO migration and rollback guide.

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. Repeat with `ENCODING` for **Country Name**, `Québec` for **Region Label**, and `1` for **# Words In Postal Code**. Save.
4. Open a fresh Desktop Client session, run `TST`, and go back to **Setup → Countries**. Choose **Inquire** and click **First Record** if the key field is empty. Find the two keys and confirm the saved values, including the accent in `Québec`.

Cannot Access Required File for PROSPECT. In a test database with only `COUNTRY` created, click **Cancel** on this message to continue the Countries test. Create the full demo database to run the rest of the application.



Both records in a fresh Desktop Client session.

Connect independently in SQL*Plus as `APPX_ORA554_DEMO`. Check the records, encoded value, and unique index on `country_name`.

```
SELECT RTRIM(country_name), RTRIM(country_region_label), country_no_words,
       DUMP(RTRIM(country_region_label),16) AS stored_bytes
FROM APPX_TST_DMO_COUNTRY ORDER BY country_name;

SELECT i.uniqueness, c.column_name FROM user_indexes i
JOIN user_ind_columns c ON c.index_name=i.index_name
WHERE i.table_name='APPX_TST_DMO_COUNTRY';
```

Expect `CANADA / PROVINCE / 2` and `ENCODING / Québec / 1`. In the Windows-1252 text, `é` is byte `E9`.

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 32-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

If the SQL query returns a different result from the APPX screen, verify the effective FMS group, database, schema, and table name. Restart the APPX session after changing any of them.