

Multi-Tenant Patch-set Deployment
Oracle Banking Trade Finance
Release 14.5.5.0.0
Part No. F58077-01

[May] [2022]



Table of Contents

1.	Overview of Applications in an Application Container	3
1.1	Managing Applications in an Application Container	3
1.2	Application Maintenance	3
1.2.1	Application Installation	3
1.2.2	Application Upgrade	3
2.	Patch-set Application Steps	6
2.1	Application Upgrade	6
2.1.1	Purpose	6
2.1.2	Steps to be followed	6
2.2	Synchronize application PDBs	9
2.2.1	Purpose	9
2.2.2	Steps to be followed	9
3.	Step by Step Execution	10
3.1.1	Pre- Requisites.....	10
3.1.2	Patch-set Application Step by Step with Screenshots	10

1. Overview of Applications in an Application Container

1.1 Managing Applications in an Application Container

In an application container, an application is the named, versioned set of application common objects stored in the application root. In this context, “application” means “application back-end.” Application common objects include user accounts, tables, PL/SQL packages, and so on. An application can be shared with the application PDBs that belong to the application root.

On performing application changes, application PDBs can synchronize with the application in the application root. The application container also manages the versions of the application and the patches to the application:

- While installing an application, user must specify the application version number.
- While upgrading an application, user must specify the old application version number and the new application version number.

As the application evolves, the application container maintains all of the versions that are applied.

1.2 Application Maintenance

Application maintenance refers to installing, uninstalling, upgrading, or patching an application.

Perform application installation, upgrade, and patching operations using an ALTER PLUGGABLE DATABASE APPLICATION statement.

The basic steps for application maintenance are as follows:

1. Log in to the application root.
2. Begin the operation with an ALTER PLUGGABLE DATABASE APPLICATION ... BEGIN statement in the application root.
3. Execute the application maintenance statements.
4. End the operation with an ALTER PLUGGABLE DATABASE APPLICATION ... END statement.

These statements can be issued in the same user session or in different user sessions.

1.2.1 Application Installation

An application installation is the initial creation of a master application definition. A typical installation creates user accounts, tables, and PL/SQL packages.

Refer [Multi-Tenant_Deployment.pdf](#) for more details on the application installation.

1.2.2 Application Upgrade

An application upgrade is a major change to an installed application.

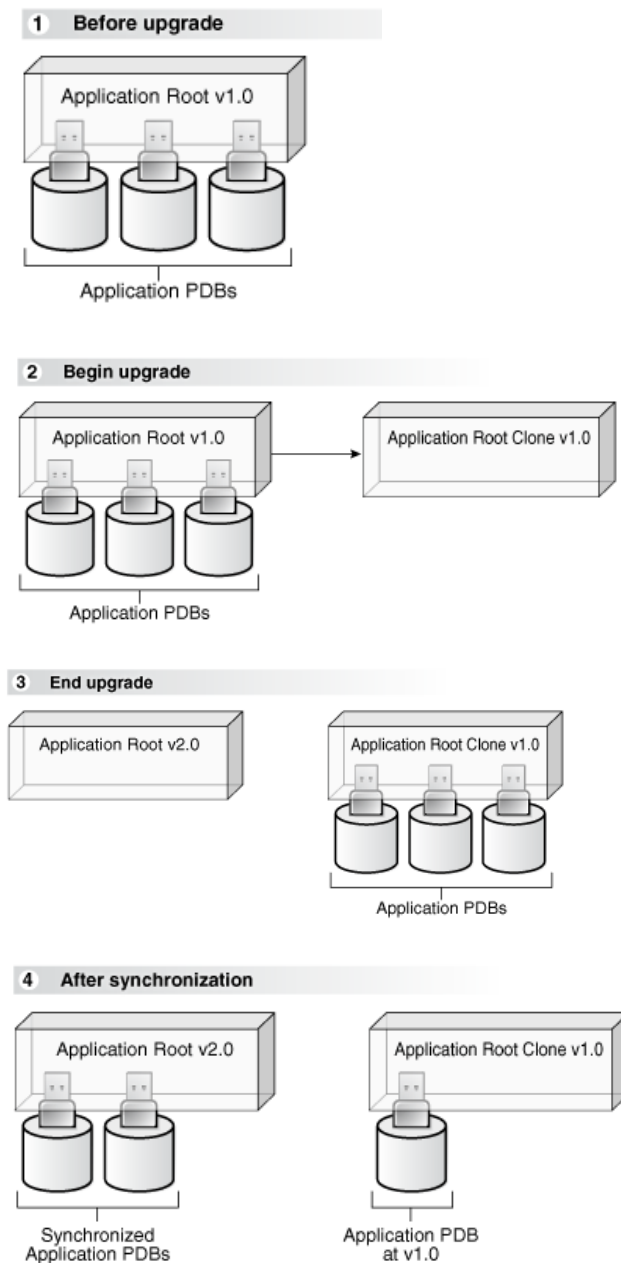
Typically, an upgrade changes the physical architecture of the application. For example, an upgrade might add new tables, and packages, or alter the definitions of existing objects.

To upgrade the application, specify the following in the ALTER PLUGGABLE DATABASE APPLICATION statement:

- Name of the application
- Old application version number
- New application version number

During an application upgrade, the application remains available. To make this availability possible, Oracle Database clones the application root.

The following figure gives an overview of the application upgrade process.



When an application is upgraded, Oracle Database automatically clones the application root.

During the upgrade, application PDBs point to the clone and applications continue to run during the upgrade. Application PDBs can perform DML on metadata-linked tables and views and query data-linked tables.

After the upgrade, the application root clone remains and continues to support any application PDB that still uses the pre-upgrade version of the application in the clone.

Application PDBs that are synchronized are pointed to the upgraded application root. Application PDBs that are not synchronized might continue to use the clone.

2. Patch-set Application Steps

Multi entity application root/PDB based setup has to be available to perform 18c database application upgrade for applying the patch-set. Refer **Multi-Tenant_Deployment.docx** for the deployment and installation steps.

Patch-set can be applied by following below steps in sequential order, and detail of each steps explained as separate sections subsequently.

- Application Upgrade
- Synchronize application PDBs

Patch-set Deployment Pre-requisites:

- ✓ Download the required patch-set zip file and unzip it in a local path.
- ✓ Verify whether the property files (fcubs.properties and env.properties) have the application root schema details where the application is available, if not update the approot schema details through installer (Refer OBTF_Property_File_Creation.docx for more details) and re-generate the files.
- ✓ Make sure to set the flag PATCHSET_INSTALLATION to 'Y'.

2.1 Application Upgrade

2.1.1 Purpose

Major changes to an application constitute application upgrades. During the upgrade, Oracle Database automatically clones the application root and the application PDBs point to the clone.

Application upgrade can be performed in the application root only, and application PDBs applies the changes in the upgrade when they synchronize with the application.

2.1.2 Steps to be followed

Below steps to be followed to initiate application upgrade

- ✓ Start Application upgrade
- ✓ Compiling Incremental Units
- ✓ Recompilation of invalids
- ✓ End Application upgrade

- ✓ Start Application upgrade
- ✓ Application Root objects conversion for new objects
- ✓ Application Root objects conversion for existing objects
- ✓ Recompilation of invalids
- ✓ End Application upgrade

2.1.2.1 Start Application upgrade

An ALTER PLUGGABLE DATABASE APPLICATION statement has to be issued to upgrade an application in the application root.

Each upgrade must be associated with an application name, starting version number, and ending version number.

Pre-requisites:

- The common user must have the DBA privilege, and the privilege must be commonly granted in the application root.

- The application root must be in open read/write.

Run the below script for initiating an application upgrade. This will initiate the application from current version to the next version (patch-set version).



01_Start_Upgrade.sql

Input sample for the script:

Spool Path	<< Any local path>>
Application next version	14.4.0.0.0

2.1.2.2 Compiling Incremental Units

Patch-set objects have to be loaded using bat file [E.g.: SMSDBCompileRun.bat, TFDBCompileRun.bat] by silent installer for respective product processor.

Compile the incremental SMS units using /INSTALLER/SOFT/SMSDBCompileRun.sh for UNIX installations or /INSTALLER/SOFT/SMSDBCompileRun.bat for Windows installations.

Compile the incremental OBTF units using /INSTALLER/SOFT/TFDBCompileRun.sh for UNIX installations or /INSTALLER/SOFT/TFDBCompileRun.bat for Windows installations.

2.1.2.3 Recompilation of invalids

As the sharing property of most of the objects are modified other than NONE, recompilation of objects is not allowed outside an application.

Recompilation of objects will be initiated inside the application upgrade for sanity with zero invalids with the below script:



03_Invalids_Recompilation_Inside_Upgrade

2.1.2.4 End Application upgrade

Application upgrade can be performed in the application root only and end of the upgrade is performed with an ALTER PLUGGABLE DATABASE APPLICATION END UPGRADE statement.

Run the below script for ending an application upgrade for patch-set.



04_End_Upgrade.sql

And run the invalid script by connecting to the common user in approot outside the upgrade.



04_Invalids_Recompilation_Outside_Upgrade.sql

2.1.2.5 Start Application upgrade

Run the below script for initiating another application upgrade for object conversion. This will initiate the application from current version to the next version (patch-set version).



05_Start_Upgrade.sql

Input sample for the script:

Spool Path	<< Any local path>>
Application next version	14.4.0.0.0

2.1.2.6 Application Root objects conversion for new objects

As part of patch-set when there are new tables added which has to be converted as DL or when there is a new function id which is identified to be an approot function is provided, otherwise no conversion will happen as part of this step

Below script takes care of converting the new DL objects during patch-set based on the deployment model of the application during installation.



06_New_Object_Conversion.sql

Input sample for the script:

Spool Path	<< Any local path>>
Approot User (In Caps)	HUBUSER (common user name)

2.1.2.7 Application Root objects conversion for existing objects

Various Sharing types of objects during installation:

- A static table will hold the information of selected table sharing as Data link. Other tables will be treated as Meta Data Link
- Sharing of object types such as INDEX, LOB, TABLE PARTITION, SEQUENCE, and DYNAMIC PACKAGES will remain as NONE.
- All other object types such as Packages, Procedures, Functions, and Synonyms would be converted as Meta Data Link sharing.

Sharing during upgrade:

Sharing of existing database objects will remain the same.

Below script takes care of converting the modified MDL objects when there is a re-creation [objects with Create or Replace command during creation] happens during patch-set



07_Object_Conversion.sql

Input sample for the script:

Spool Path	<< Any local path>>
Approot User (In Caps)	HUBUSER (common user name)

When there are new tables introduced as part of patch-set which has to be converted into DL will be done separately. The recommendation for the same will be provided as part of patch-set instructions for this case.

2.1.2.8 Recompilation of invalids

As the sharing property of most of the objects are modified other than NONE, recompilation of objects is not allowed outside an application.

Recompilation of objects will be initiated inside the application upgrade for sanity with zero invalids with the below script:



08_Invalids_Recompilation_Inside_Upgrade

2.1.2.9 End Application upgrade

Application upgrade can be performed in the application root only and end of the upgrade is performed with an ALTER PLUGGABLE DATABASE APPLICATION END UPGRADE statement.

Run the below script for ending an application upgrade for patch-set.



09_End_Upgrade.sql

And run the invalid script by connecting to the common user in approot outside the upgrade.



09_Invalids_Recompilation_Outside_Upgrad

2.2 Synchronize application PDBs

2.2.1 Purpose

- Synchronizing an application updates the application in the application PDB to the latest version in the application root. When an application is upgraded in an application root, an application PDB that belongs to the application root is not changed until it is synchronized.
- Application PDBs synchronize with an application by running an ALTER PLUGGABLE DATABASE statement with the SYNC clause.

2.2.2 Steps to be followed

Prerequisites

- The current user must have ALTER PLUGGABLE DATABASE system privilege.
- Ensure that the current container is the application PDB.
- Run an ALTER PLUGGABLE DATABASE APPLICATION statement with the SYNC clause.

Run the below script to synchronize the PDBs with the latest application changes in the application root.



10_PDB_Sync.sql

3. Step by Step Execution

3.1.1 Pre- Requisites

1) Before applying the patch-set, we have to make sure the release is updates with the base version of the patch-set.

For Example, If the first patch-set of 14.2 is yet to be applied, the release has to be updated as '14.2.0.0.0'. It can be verified with the below queries

```
select param_name, param_val from CSTB_PARAM WHERE PARAM_NAME = 'RELEASE';
select module_group_id, release from SMTB_MODULES_GROUP;
```

2) Another significant parameter is the values of application name and deployment type in CSTB_PARAM.

This value will be updated from the installer during Approot Object Conversion utility as part of deployment.

```
select param_name, param_val from cstb_param where PARAM_NAME in
('MULTI_TENANT_APP_NAME', 'MULTI_TENANT_DEPLOYMENT_MODEL');
```

The Application name of multi-tenant deployment will be stored in CSTB_PARAM as

Param_Name	Param_Val
MULTI_TENANT_APP_NAME	OBTF

The type of object conversion will be stored in CSTB_PARAM as

Param_Name	Param_Val
MULTI_TENANT_DEPLOYMENT_MODEL	SA (or) SAUA (or) SASDD (or) SASDC

SA → Shared Application

SAUA → Shared Application User Authentication

SASDD → Shared Application Shared Data - Default

SASDC → Shared Application Shared Data – Custom

3.1.2 Patch-set Application Step by Step with Screenshots

Step 1: Start Application upgrade

- Login into the Approot Schema as Common user.
- Run **01_Start_Upgrade.sql** for initiating the application upgrade.
- User input has to be inputted for the below:

Spool Path	<< Any local path>>
Application next version	14.4.0.0.0

- Script will be executed as in the screen shot below and keep the SQL Plus session open for upcoming steps.

Execution Screenshot:

```

C:\app\client\primalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> SPOOL ON
SQL> SET SQLBLANKLINES ON
SQL> SET SERVEROUTPUT ON
SQL> SET ERRORLOGGING ON
SQL> SET ECHO ON
SQL> prompt Welcome to Application PDB Configuration
Welcome to Application PDB Configuration
SQL> SPOOL "&SPOOL_PATH"
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\01Spool.txt
SQL>
SQL> DECLARE
2      l_app_name          VARCHAR2(128);
3      l_app_currver       VARCHAR2(30);
4      l_sql               VARCHAR2(256);
5  BEGIN
6
7      BEGIN
8          SELECT app_name
9              INTO l_app_name
10             FROM dba_applications
11             WHERE app_implicit <> 'Y'
12             AND app_name = (SELECT param_val FROM cstb_param WHERE Param_name = 'MULTI_TENANT_APP_NAME');
13  EXCEPTION
14      WHEN NO_DATA_FOUND THEN
15          dbms_output.put_line('Error Nodata--->' || SQLERRM);
16      WHEN OTHERS THEN
17          dbms_output.put_line('Error others--->' || SQLERRM);
18  END;
19  SELECT MAX(app_version)
20      INTO l_app_currver
21      FROM dba_app_versions
22      WHERE app_name = l_app_name;
23
24  l_sql := 'ALTER PLUGGABLE DATABASE APPLICATION ' || l_app_name || ' BEGIN UPGRADE ''' || l_app_currver || ''' TO ''' || '&P_APPLICATION_NEXTVER' || '''';
25  dbms_output.put_line('l_sql: ' || l_sql);
26  EXECUTE IMMEDIATE l_sql;
27
28  l_sql := 'ALTER SYSTEM SET DEFAULT_SHARING = NONE';
29  dbms_output.put_line('l_sql: ' || l_sql);
30  EXECUTE IMMEDIATE l_sql;
31
32  EXCEPTION
33      WHEN OTHERS THEN
14      WHEN NO_DATA_FOUND THEN
15          dbms_output.put_line('Error Nodata--->' || SQLERRM);
16      WHEN OTHERS THEN
17          dbms_output.put_line('Error others--->' || SQLERRM);
18  END;
19  SELECT MAX(app_version)
20      INTO l_app_currver
21      FROM dba_app_versions
22      WHERE app_name = l_app_name;
23
24  l_sql := 'ALTER PLUGGABLE DATABASE APPLICATION ' || l_app_name || ' BEGIN UPGRADE ''' || l_app_currver || ''' TO ''' || '&P_APPLICATION_NEXTVER' || '''';
25  dbms_output.put_line('l_sql: ' || l_sql);
26  EXECUTE IMMEDIATE l_sql;
27
28  l_sql := 'ALTER SYSTEM SET DEFAULT_SHARING = NONE';
29  dbms_output.put_line('l_sql: ' || l_sql);
30  EXECUTE IMMEDIATE l_sql;
31
32  EXCEPTION
33      WHEN OTHERS THEN
34          dbms_output.put_line('Error --->' || SQLERRM);
35  END;
36  /
Enter value for p_application_nextver: 14.2.0.0.2
old 24:  l_sql := 'ALTER PLUGGABLE DATABASE APPLICATION ' || l_app_name || ' BEGIN UPGRADE ''' || l_app_currver || ''' TO ''' || '&P_APPLICATION_NEXTVER' || '''';
new 24:  l_sql := 'ALTER PLUGGABLE DATABASE APPLICATION ' || l_app_name || ' BEGIN UPGRADE ''' || l_app_currver || ''' TO ''' || '14.2.0.0.2' || '''';
l_sql: ALTER PLUGGABLE DATABASE APPLICATION FCUBS BEGIN UPGRADE '14.2.0.0.1' TO '14.2.0.0.2'
l_sql: ALTER SYSTEM SET DEFAULT_SHARING = NONE
PL/SQL procedure successfully completed.
SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>

```

Step 2: Compiling Incremental Units

- Make sure that the fcubs.properties and env.properties are updated with approot schema details.
- Run the <Product Processor>DBCompileRun.bat from <Patchset>\INSTALLER\SOFT directory. DDL Compilation, Object Compilation and Static Data load will be done.

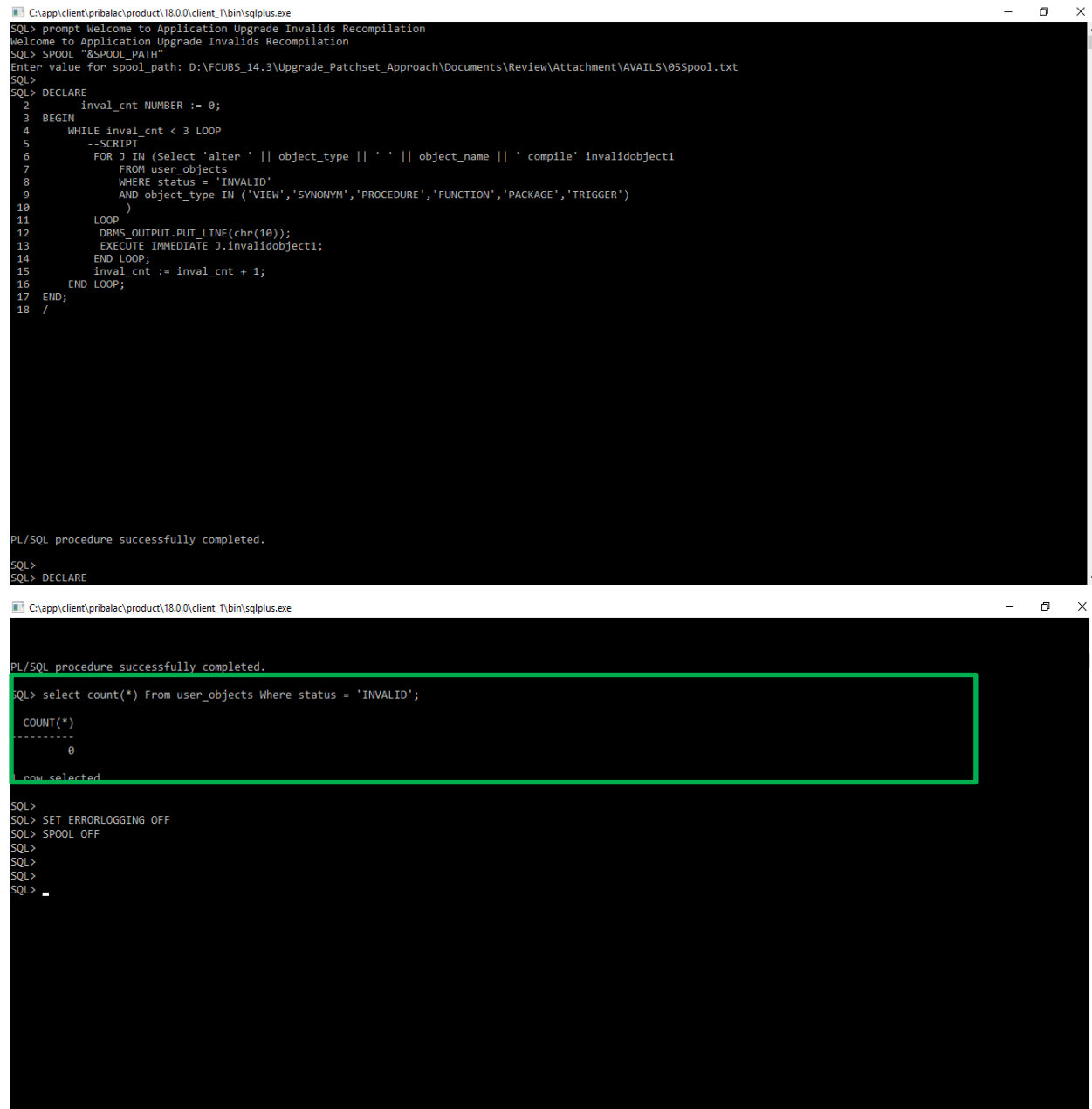
For Example: OBTF INSTALLATION-

First load SMS objects first and then OBTF objects. i.e. Run SMSDBCompileRun.bat and after SMS object loading is completed, then initiate OBTF compilation Run TFDDBCompileRun.bat

Step 3: Recompilation of invalids

- Login into the Approot Schema as Common user.
- Run **03_Invalids_Recompilation.sql** for recompiling the invalids during application upgrade.
- No user input is required for this step.
- Script will be executed as in the screen shot below and keep the SQL Plus session open for upcoming steps.

Execution Screenshot:



```
C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> prompt Welcome to Application Upgrade Invalids Recompilation
Welcome to Application Upgrade Invalids Recompilation
SQL> SPOOL "ASPOOL_PATH"
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\05Spool.txt
SQL>
SQL> DECLARE
2   inval_cnt NUMBER := 0;
3 BEGIN
4   WHILE inval_cnt < 3 LOOP
5     --SCRIPT
6     FOR J IN (Select 'alter ' || object_type || ' ' || object_name || ' compile' invalidobject1
7               FROM user_objects
8               WHERE status = 'INVALID'
9               AND object_type IN ('VIEW', 'SYNONYM', 'PROCEDURE', 'FUNCTION', 'PACKAGE', 'TRIGGER'))
10    )
11    LOOP
12      DBMS_OUTPUT.PUT_LINE(chr(10));
13      EXECUTE IMMEDIATE J.invalidobject1;
14    END LOOP;
15    inval_cnt := inval_cnt + 1;
16  END LOOP;
17 END;
18 /

PL/SQL procedure successfully completed.

SQL>
SQL> DECLARE

C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
PL/SQL procedure successfully completed.

SQL> select count(*) From user_objects Where status = 'INVALID';

COUNT(*)
-----
         0

1 row selected

SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>
SQL>
SQL>
SQL>
```

Step 4: End Application upgrade

- Login into the Approot Schema as Common user.
- Run **06_End_Upgrade.sql** for recompiling the invalids during application upgrade.

- c. No user input is required for this step.
- d. Script will be executed as in the screen shot below.

Execution Screenshot:

```

C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> prompt Welcome to Application PDB Configuration
Welcome to Application PDB Configuration
SQL> SPOOL "&SPOOL_PATH"
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\06Spool1.txt
SQL>
SQL> DECLARE
2   l_app_name      VARCHAR2(128);
3   l_sql           VARCHAR2(256);
4 BEGIN
5   BEGIN
6     SELECT app_name
7     INTO l_app_name
8     FROM dba_applications
9     WHERE app_implicit <> 'Y'
10    AND app_name = (SELECT param_val FROM cstab_param WHERE param_name = 'MULTI_TENANT_APP_NAME');
11   EXCEPTION
12     WHEN NO_DATA_FOUND THEN
13       dbms_output.put_line('Error1 Nodata---')||SQLERRM;
14     WHEN OTHERS THEN
15       dbms_output.put_line('Error1 others---')||SQLERRM;
16   END;
17   l_sql := 'ALTER PLUGGABLE DATABASE APPLICATION ' || l_app_name || ' END UPGRADE ';
18   dbms_output.put_line('l_sql: ' || l_sql);
19   EXECUTE IMMEDIATE l_sql;
20   EXCEPTION
21     WHEN OTHERS THEN
22       dbms_output.put_line('Error ---')||SQLERRM;
23   END;
24 /
l_sql: ALTER PLUGGABLE DATABASE APPLICATION FCUBS END UPGRADE
PL/SQL procedure successfully completed.
SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>

```

Step 5: Start Application upgrade

- a. Login into the Approot Schema as Common user.
- b. Run **05_Start_Upgrade.sql** for initiating the application upgrade.
- c. User input has to be inputted for the below:

Spool Path	<< Any local path>>
Application next version	14.4.0.0.0

- d. Script will be executed similar to step 1 above and keep the SQL Plus session open for upcoming steps.

Step 6: Application Root objects conversion for new objects

- a. Login into the Approot Schema as Common user.
- b. Run **06_New_Object_Conversion.sql** for converting new approot objects added during patch-set as DL
- c. User input has to be inputted for the below:

Spool Path	<< Any local path>>
Approot User (In Caps)	HUBUSER (common user name)

- d. Script will be executed as in the screen shot below and keep the SQL Plus session open for upcoming steps.

Execution Screenshot:

```

C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> prompt Welcome to Application PDB Configuration
Welcome to Application PDB Configuration
SQL> SPOOL 'ASPOOL_PATH'
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\02Spool.txt
SQL>
SQL> DECLARE
2   l_count NUMBER;
3   l_app_deployment VARCHAR2(30);
4 BEGIN
5   SELECT count(*)
6   INTO l_count
7   FROM user_objects
8   WHERE sharing = 'NONE' --to get the new set of DL approot objects if any
9   AND object_name IN (SELECT DISTINCT a.object_name
10                      FROM cstm_approot_objects a
11                      WHERE sharing = 'DL'
12                      AND UPPER(object_type) = 'TABLE'
13                      AND EXISTS (SELECT 1
14                                FROM user_objects b
15                                WHERE b.object_name = a.object_name)
16                      AND EXISTS (SELECT 1
17                                FROM cstm_approot_functions_menu c
18                                WHERE c.function_id = a.function_id
19                                AND c.modifiable IN ('Y', 'S')));
20   dbms_output.put_line('l_count: ' || l_count);
21   IF l_count > 0 THEN
22     dbms_output.put_line('New DL objects are available');
23     SELECT param_val
24     INTO l_app_deployment
25     FROM cstm_param
26     WHERE param_name = 'MULTI_TENANT_DEPLOYMENT_MODEL';
27     dbms_output.put_line('l_app_deployment: ' || l_app_deployment);
28     IF l_app_deployment IS NOT NULL AND l_app_deployment = 'SAUA' THEN
29       UPDATE smtb_menu menu
30       SET menu.approot_flg = 'Y'
31       WHERE menu.function_id IN
32       (SELECT function_id
33        FROM cstm_approot_functions_menu
34        WHERE modifiable = 'S'
35        UNION
36        SELECT summary_fn_id
37        FROM cstm_approot_functions_menu
38        WHERE modifiable = 'S'
39        AND summary_fn_id IS NOT NULL) --SMS function id 'S'
40
41   )
42   LOOP
43     DBMS_OUTPUT.PUT_LINE(chr(10));
44     EXECUTE IMMEDIATE I.sqlobject;
45     DBMS_OUTPUT.PUT_LINE(I.sqlobject);
46   END LOOP;
47   EXCEPTION
48   WHEN OTHERS THEN
49     DBMS_OUTPUT.PUT_LINE('Error --->'||SQLERRM);
50   END;
51 ELSE
52   dbms_output.put_line('No new DL objects available');
53 END IF;
54 EXCEPTION
55 WHEN OTHERS THEN
56   dbms_output.put_line('Error --->'||SQLERRM);
57 END;
58 /
Enter value for p_approot_user: HUBUSER
old 77:      FOR I IN (SELECT 'BEGIN ' || chr(10) || 'DBMS_PDB.SET_DATA_LINKED('&p_approot_user' ||
new 77:      FOR I IN (SELECT 'BEGIN ' || chr(10) || 'DBMS_PDB.SET_DATA_LINKED(''HUBUSER'' ||
l_count: 1
New DL objects are available
l_app_deployment: SAS00

BEGIN
DBMS_PDB.SET_DATA_LINKED('HUBUSER','CSTM_CHECK2',1);
EXCEPTION
WHEN OTHERS then
DBMS_OUTPUT.PUT_LINE('ERROR ->'|| SQLERRM);
END;

PL/SQL procedure successfully completed.

SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>

```

Step7: Application Root objects conversion for existing objects

- Login into the Approot Schema as Common user.
- Run **07_Object_Conversion.sql** for initiating the application upgrade.
- User input has to be inputted for the below:

Spool Path	<< Any local path>>
Approot User (In Caps)	HUBUSER (common user name)

- Script will be executed as in the screen shot below and keep the SQL Plus session open for upcoming steps.

Execution Screenshot:

```

C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> SET SERVEROUTPUT ON
SQL> SET ERRORLOGGING ON
SQL> SET ECHO ON
SQL> prompt Welcome to Upgrade object conversion
Welcome to Upgrade object conversion
SQL> SPOOL "&SPOOL_PATH"
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\04Spool1.txt
SQL>
SQL> DECLARE
2   l_app_name          VARCHAR2(128);
3   BEGIN
4       BEGIN
5           SELECT app_name
6               INTO l_app_name
7           FROM dba_applications
8           WHERE app_implicit <> 'Y'
9           AND app_name = (SELECT param_val FROM cstb_param WHERE Param_name = 'MULTI_TENANT_APP_NAME');
10          EXCEPTION
11              WHEN NO_DATA_FOUND THEN
12                  dbms_output.put_line('Error1 Nodata-->'||SQLERRM);
13              WHEN OTHERS THEN
14                  dbms_output.put_line('Error1 others-->'||SQLERRM);
15          END;
16
17          FOR I IN (SELECT 'BEGIN ' || chr(10) ||
18                      'DBMS_PDB.SET_METADATA_LINKED(''&P_APPROOT_USER'' || ', '''' ||
19                      Object_Name || ', '' || Namespace || '); ' || chr(10) ||
20                      'EXCEPTION ' || chr(10) || 'WHEN OTHERS then ' || chr(10) ||
21                      'DBMS_OUTPUT.PUT_LINE(''ERROR ->' || SQLERRM); ' ||
22                      chr(10) || 'END;' sqlobject
23              FROM user_objects
24              WHERE sharIng = 'NONE'
25              AND object_type NOT IN ('INDEX', 'LOB', 'TABLE PARTITION', 'SEQUENCE')
26              AND (object_name NOT LIKE '%#%')
27              AND (created_appid = (select app_id from dba_applications where app_name = l_app_name) OR
28                  modified_appid = (select app_id from dba_applications where app_name = l_app_name)) ) LOOP
29              DBMS_OUTPUT.PUT_LINE(chr(10));
30              EXECUTE IMMEDIATE I.sqlobject;
31          END LOOP;
32          EXCEPTION
33              WHEN OTHERS THEN
34                  DBMS_OUTPUT.PUT_LINE('Error --->' || SQLERRM);
35          END;
36  /

```

```

C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
5       SELECT app_name
6           INTO l_app_name
7       FROM dba_applications
8       WHERE app_implicit <> 'Y'
9       AND app_name = (SELECT param_val FROM cstb_param WHERE Param_name = 'MULTI_TENANT_APP_NAME');
10      EXCEPTION
11          WHEN NO_DATA_FOUND THEN
12              dbms_output.put_line('Error1 Nodata-->'||SQLERRM);
13          WHEN OTHERS THEN
14              dbms_output.put_line('Error1 others-->'||SQLERRM);
15      END;
16
17      FOR I IN (SELECT 'BEGIN ' || chr(10) ||
18                  'DBMS_PDB.SET_METADATA_LINKED(''&P_APPROOT_USER'' || ', '''' ||
19                  Object_Name || ', '' || Namespace || '); ' || chr(10) ||
20                  'EXCEPTION ' || chr(10) || 'WHEN OTHERS then ' || chr(10) ||
21                  'DBMS_OUTPUT.PUT_LINE(''ERROR ->' || SQLERRM); ' ||
22                  chr(10) || 'END;' sqlobject
23          FROM user_objects
24          WHERE sharIng = 'NONE'
25          AND object_type NOT IN ('INDEX', 'LOB', 'TABLE PARTITION', 'SEQUENCE')
26          AND (object_name NOT LIKE '%#%')
27          AND (created_appid = (select app_id from dba_applications where app_name = l_app_name) OR
28              modified_appid = (select app_id from dba_applications where app_name = l_app_name)) ) LOOP
29              DBMS_OUTPUT.PUT_LINE(chr(10));
30              EXECUTE IMMEDIATE I.sqlobject;
31          END LOOP;
32          EXCEPTION
33              WHEN OTHERS THEN
34                  DBMS_OUTPUT.PUT_LINE('Error --->' || SQLERRM);
35          END;
36  /
Enter value for p_approot_user: HUBUSER
old 18:          'DBMS_PDB.SET_METADATA_LINKED(''&P_APPROOT_USER'' || ', '''' ||
new 18:          'DBMS_PDB.SET_METADATA_LINKED(''HUBUSER'' || ', '''' ||

PL/SQL procedure successfully completed.

SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>

```

Step8: Recompilation of invalids

- Login into the Approot Schema as Common user.
- Run **08_Invalids_Recompilation.sql** for recompiling the invalids during application upgrade.
- No user input is required for this step.
- Script will be executed as in the screen shot below and keep the SQL Plus session open for upcoming steps.

Execution Screenshot:

```
C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> prompt Welcome to Application Upgrade Invalids Recompilation
Welcome to Application Upgrade Invalids Recompilation
SQL> SPOOL "&SPPOOL_PATH"
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\05Spool.txt
SQL>
SQL> DECLARE
2   inval_cnt NUMBER := 0;
3 BEGIN
4   WHILE inval_cnt < 3 LOOP
5     --SCRIPT
6     FOR J IN (Select 'alter ' || object_type || ' ' || object_name || ' compile' invalidobject1
7               FROM user_objects
8               WHERE status = 'INVALID'
9               AND object_type IN ('VIEW','SYNONYM','PROCEDURE','FUNCTION','PACKAGE','TRIGGER'))
10    LOOP
11      DBMS_OUTPUT.PUT_LINE(chr(10));
12      EXECUTE IMMEDIATE J.invalidobject1;
13    END LOOP;
14    inval_cnt := inval_cnt + 1;
15  END LOOP;
16 END;
17 /
PL/SQL procedure successfully completed.
SQL>
SQL> DECLARE

PL/SQL procedure successfully completed.
SQL> select count(*) From user_objects Where status = 'INVALID';
COUNT(*)
-----
0
1 row selected.
SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>
SQL>
SQL>
SQL>
```

Step 9: End Application upgrade

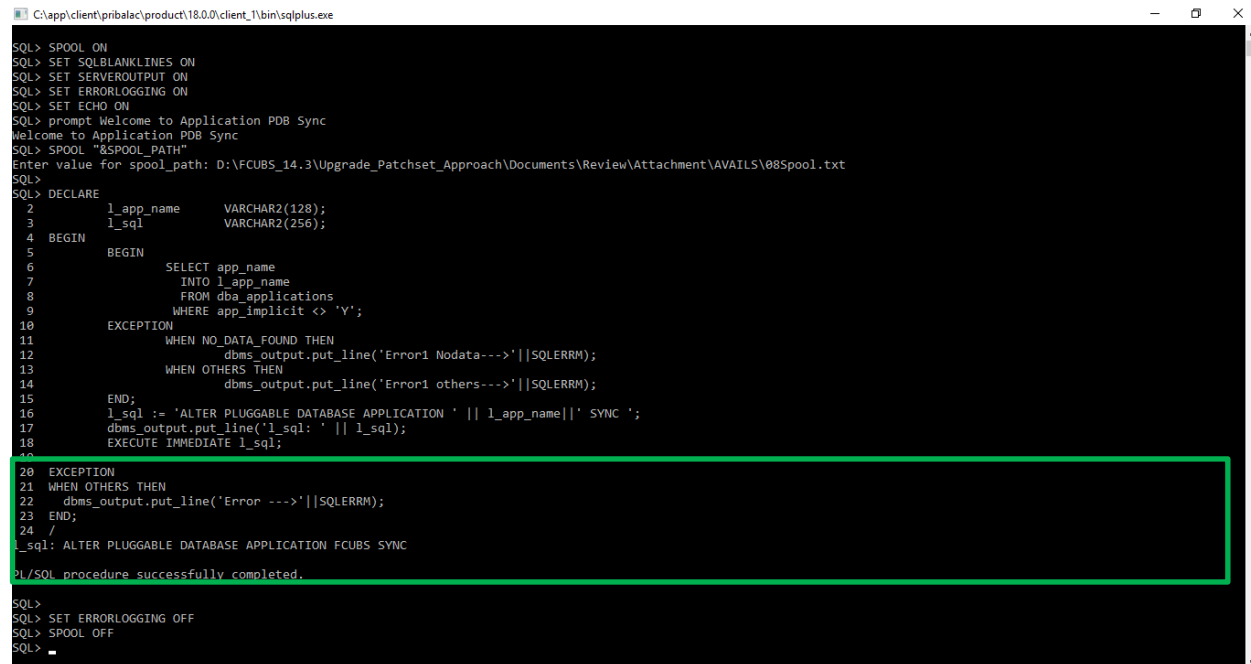
- Login into the Approot Schema as Common user.
- Run **06_End_Upgrade.sql** for recompiling the invalids during application upgrade.
- No user input is required for this step.
- Script will be executed as that of step 4.

Step 10: Synchronize application PDBs

- Login into the PDB Schema as Common user. For each PDB, this steps has to be done individually
- Run **07_PDB_Sync.sql** for synching the application upgrade with PDBs.

- c. No user input is required for this step.
- d. Script will be executed as in the screen shot below.

Execution Screenshot:



```
C:\app\client\pribalac\product\18.0.0\client_1\bin\sqlplus.exe
SQL> SPOOL ON
SQL> SET SQLBLANKLINES ON
SQL> SET SERVEROUTPUT ON
SQL> SET ERRORLOGGING ON
SQL> SET ECHO ON
SQL> prompt Welcome to Application PDB Sync
Welcome to Application PDB Sync
SQL> SPOOL "&SPOOL_PATH"
Enter value for spool_path: D:\FCUBS_14.3\Upgrade_Patchset_Approach\Documents\Review\Attachment\AVAILS\08Spool.txt
SQL>
SQL> DECLARE
2   l_app_name      VARCHAR2(128);
3   l_sql           VARCHAR2(256);
4 BEGIN
5   BEGIN
6       SELECT app_name
7         INTO l_app_name
8       FROM dba_applications
9       WHERE app_implicit <> 'Y';
10  EXCEPTION
11  WHEN NO_DATA_FOUND THEN
12      dbms_output.put_line('Error1 Nodata-->'||SQLERRM);
13  WHEN OTHERS THEN
14      dbms_output.put_line('Error1 others-->'||SQLERRM);
15  END;
16  l_sql := 'ALTER PLUGGABLE DATABASE APPLICATION ' || l_app_name || ' SYNC ';
17  dbms_output.put_line('l_sql: ' || l_sql);
18  EXECUTE IMMEDIATE l_sql;
19
20 EXCEPTION
21 WHEN OTHERS THEN
22     dbms_output.put_line('Error --->'||SQLERRM);
23 END;
24 /
l_sql: ALTER PLUGGABLE DATABASE APPLICATION FCUBS SYNC
PL/SQL procedure successfully completed.

SQL>
SQL> SET ERRORLOGGING OFF
SQL> SPOOL OFF
SQL>
```



Multi-Tenant Patch-set Deployment

[May] [2022]

Version 14.5.5.0.0

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)ss

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax: +91 22 6718 3001

<https://www.oracle.com/industries/financial-services/index.html>

Copyright © [2007], [2022], Oracle and/or its affiliates. All rights reserved.

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 failsafe, 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.

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.

This software or hardware and documentation may provide access to or information on 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. 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.