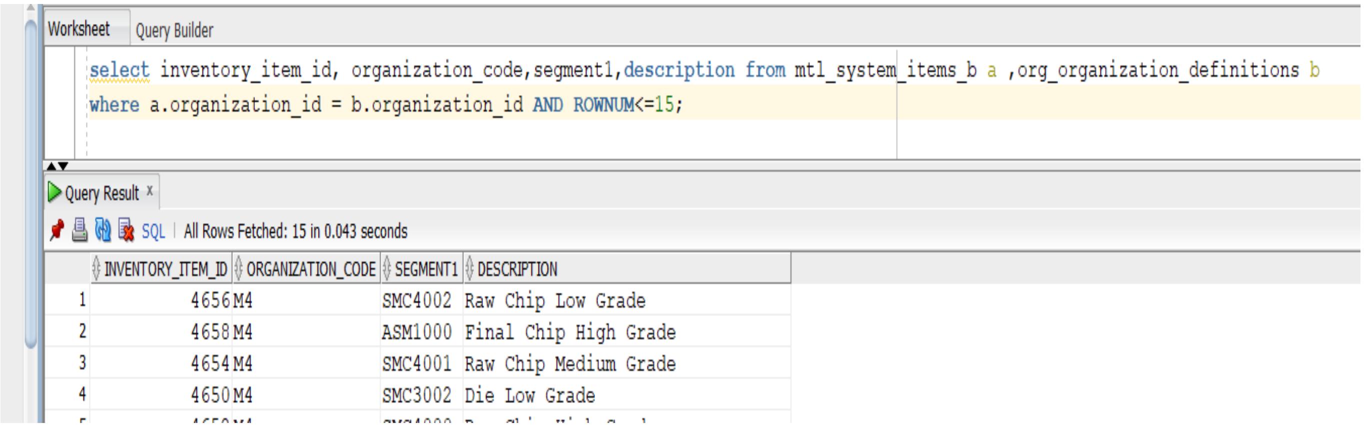
XML REPORTS USING PLSQL APPROACH:

WRITE PLSQL PROGRAM/PROCEDURE TO GENERATE XML DATA FILE.

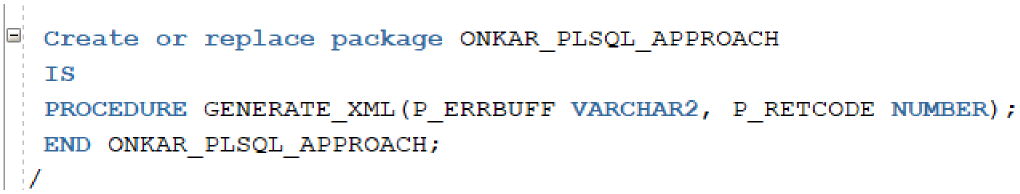
1. FIRST WRITE SQL QUERY:



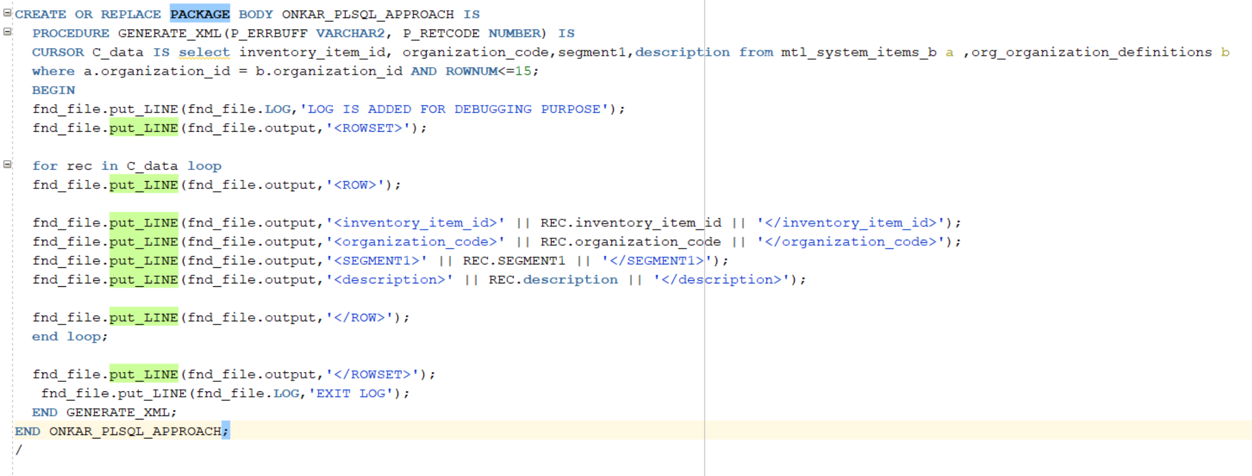
```sql
select inventory_item_id, organization_code,segment1,description from mtl_system_items_b a ,org_organization_definitions b
where a.organization_id = b.organization_id AND ROWNUM<=15;
```

	INVENTORY_ITEM_ID	ORGANIZATION_CODE	SEGMENT1	DESCRIPTION
1	4656	M4	SMC4002	Raw Chip Low Grade
2	4658	M4	ASM1000	Final Chip High Grade
3	4654	M4	SMC4001	Raw Chip Medium Grade
4	4650	M4	SMC3002	Die Low Grade

2. WRITE PLSQL API(PACKAGE & PROCEDURE) TO GENERATE XML FILE:

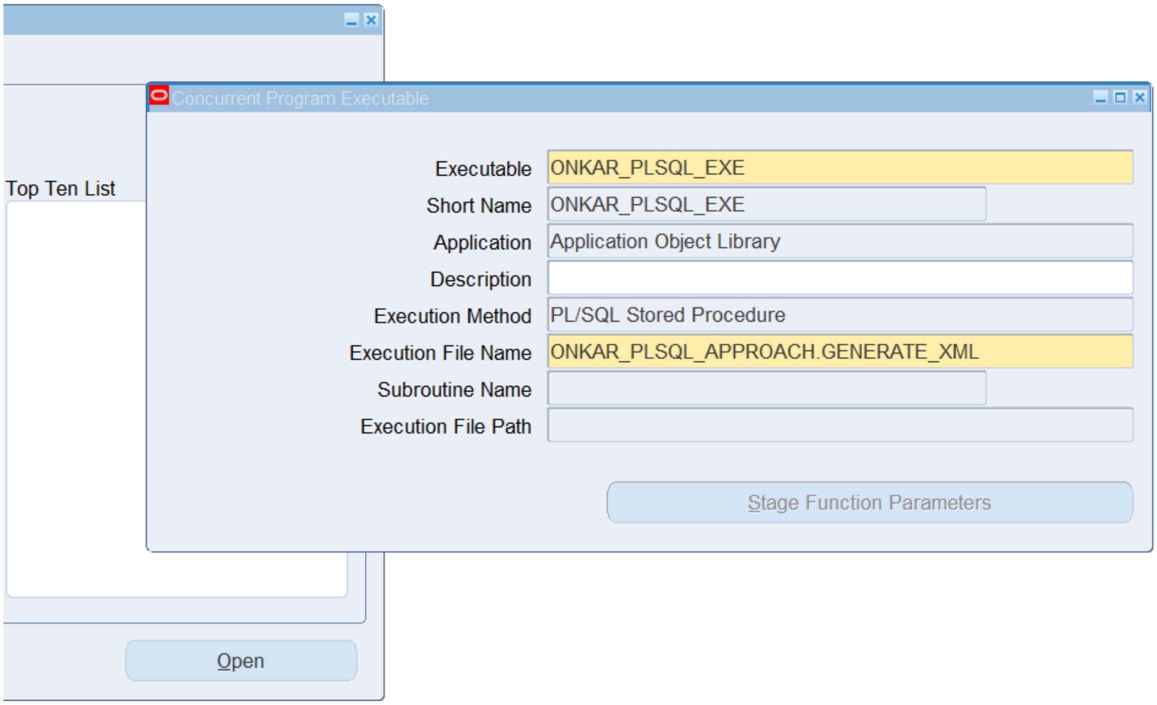


```sql
Create or replace package ONKAR_PLSQL_APPROACH
IS
PROCEDURE GENERATE_XML(P_ERRBUFF VARCHAR2, P_RETCODE NUMBER);
END ONKAR_PLSQL_APPROACH;
/
```



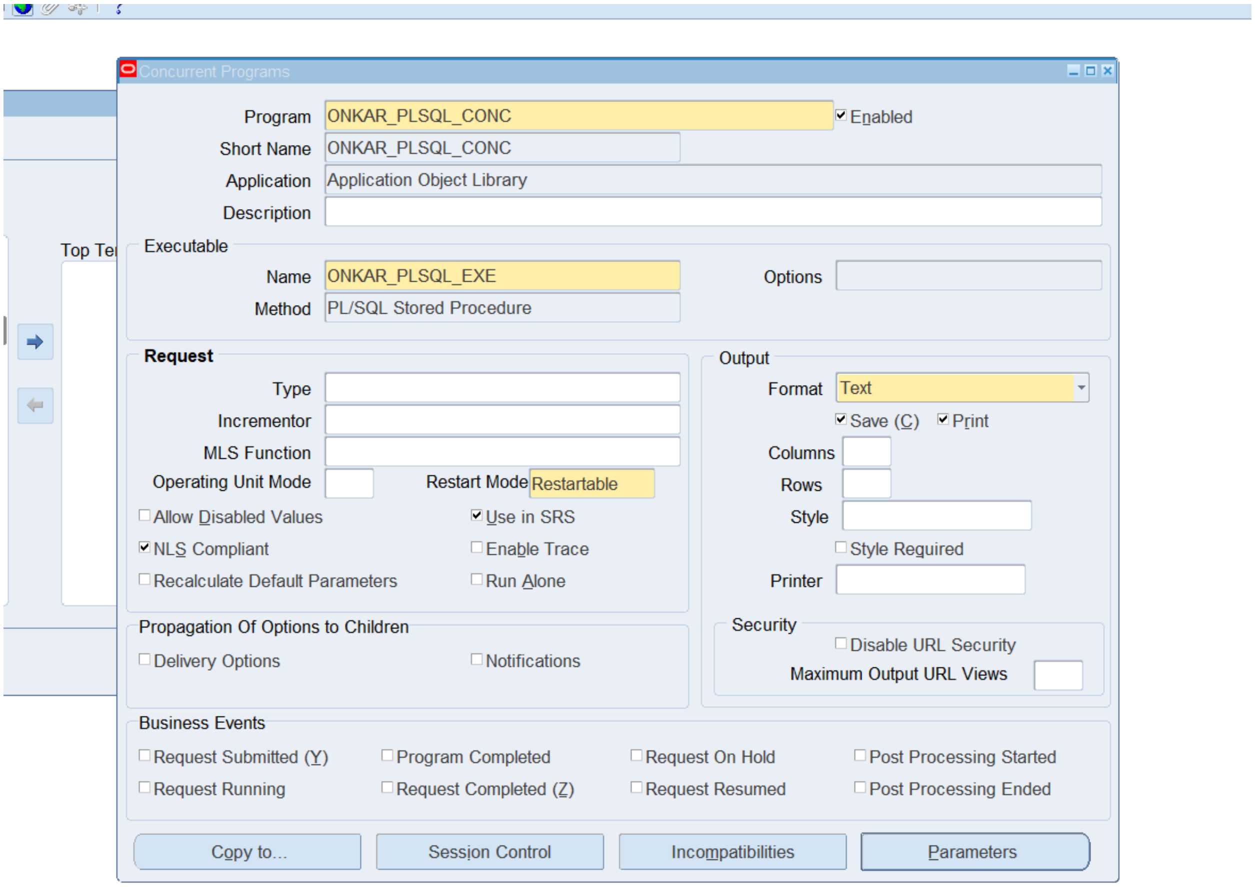
```sql
CREATE OR REPLACE PACKAGE BODY ONKAR_PLSQL_APPROACH IS
  PROCEDURE GENERATE_XML(P_ERRBUFF VARCHAR2, P_RETCODE NUMBER) IS
  CURSOR C_data IS select inventory_item_id, organization_code,segment1,description from mtl_system_items_b a ,org_organization_definitions b
  where a.organization_id = b.organization_id AND ROWNUM<=15;
  BEGIN
  fnd_file.put_LINE(fnd_file.LOG,'LOG IS ADDED FOR DEBUGGING PURPOSE');
  fnd_file.put_LINE(fnd_file.output,'<ROWSET>');

  for rec in C_data loop
  fnd_file.put_LINE(fnd_file.output,'<ROW>');

  fnd_file.put_LINE(fnd_file.output,'<inventory_item_id>' || REC.inventory_item_id || '</inventory_item_id>');
  fnd_file.put_LINE(fnd_file.output,'<organization_code>' || REC.organization_code || '</organization_code>');
  fnd_file.put_LINE(fnd_file.output,'<SEGMENT1>' || REC.SEGMENT1 || '</SEGMENT1>');
  fnd_file.put_LINE(fnd_file.output,'<description>' || REC.description || '</description>');

  fnd_file.put_LINE(fnd_file.output,'</ROW>');
  end loop;

  fnd_file.put_LINE(fnd_file.output,'</ROWSET>');
   fnd_file.put_LINE(fnd_file.LOG,'EXIT LOG');
  END GENERATE_XML;
END ONKAR_PLSQL_APPROACH;
/
```

3. CREATE CONC PRO EXECUTABLE:



Concurrent Program Executable

Executable	ONKAR_PLSQL_EXE
Short Name	ONKAR_PLSQL_EXE
Application	Application Object Library
Description	
Execution Method	PL/SQL Stored Procedure
Execution File Name	ONKAR_PLSQL_APPROACH.GENERATE_XML
Subroutine Name	
Execution File Path	



Concurrent Programs

Program	ONKAR_PLSQL_CONC
Short Name	ONKAR_PLSQL_CONC
Application	Application Object Library
Description	

☑ Enabled

Executable

Name	ONKAR_PLSQL_EXE
Method	PL/SQL Stored Procedure
Options	

Request

Type	
Incrementor	
MLS Function	
Operating Unit Mode	
Restart Mode	Restartable

□ Allow Disabled Values ☑ Use in SRS
☑ NLS Compliant □ Enable Trace
□ Recalculate Default Parameters □ Run Alone

Output

Format	Text
Columns	
Rows	
Style	
Printer	

☑ Save (C) ☑ Print
□ Style Required

Propagation Of Options to Children

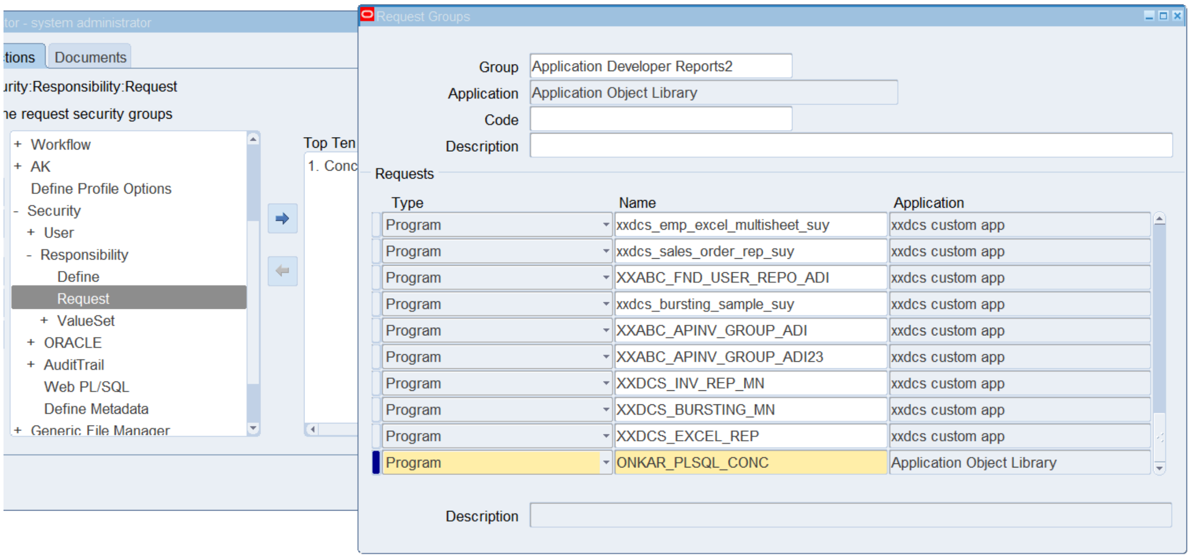
□ Delivery Options □ Notifications

Security

□ Disable URL Security
Maximum Output URL Views

Business Events

□ Request Submitted (Y) □ Program Completed □ Request On Hold □ Post Processing Started
□ Request Running □ Request Completed (Z) □ Request Resumed □ Post Processing Ended

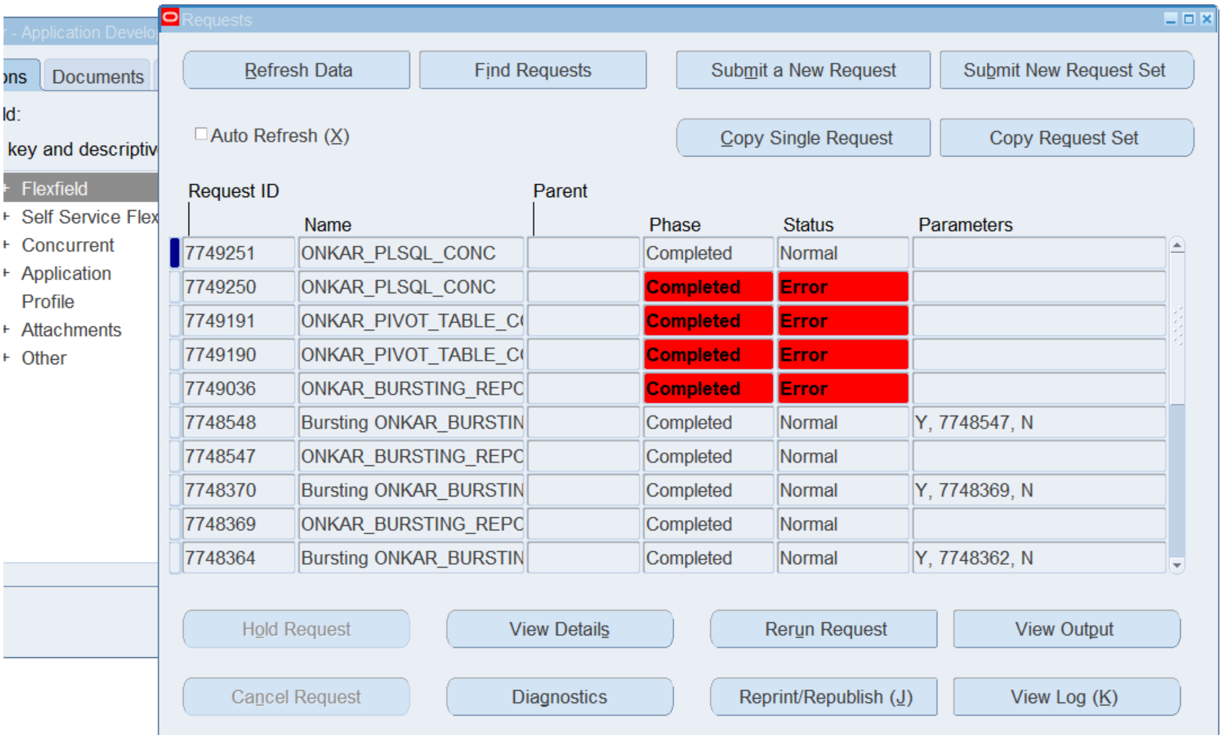


Group	Application Developer Reports2
Application	Application Object Library
Code	
Description	

Requests

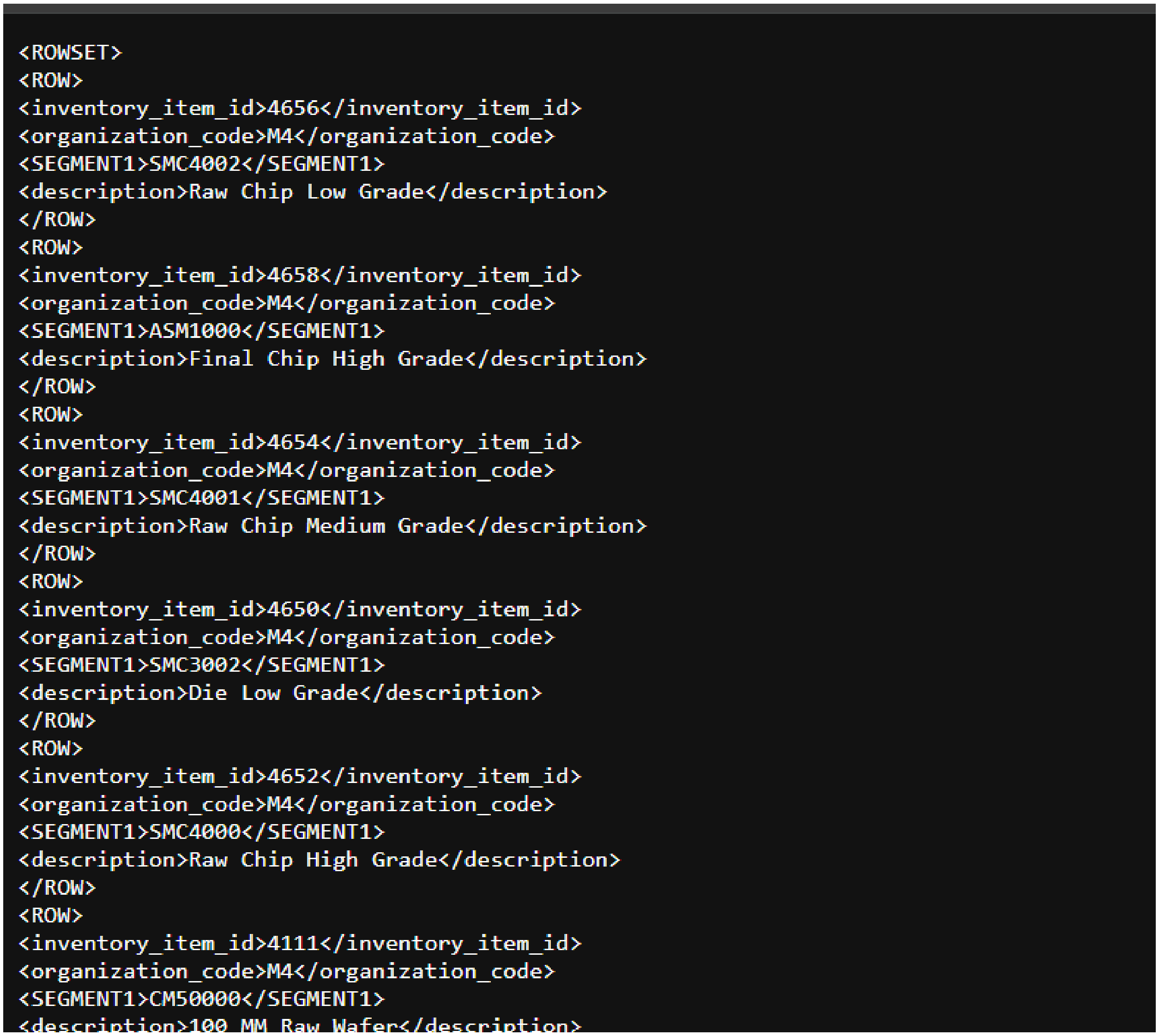
Type	Name	Application
Program	xxdcs_emp_excel_multisheet_suy	xxdcs custom app
Program	xxdcs_sales_order_rep_suy	xxdcs custom app
Program	XXABC_FND_USER_REPO_ADI	xxdcs custom app
Program	xxdcs_bursting_sample_suy	xxdcs custom app
Program	XXABC_APINV_GROUP_ADI	xxdcs custom app
Program	XXABC_APINV_GROUP_ADI23	xxdcs custom app
Program	XXDCS_INV_REP_MN	xxdcs custom app
Program	XXDCS_BURSTING_MN	xxdcs custom app
Program	XXDCS_EXCEL_REP	xxdcs custom app
Program	ONKAR_PLSQL_CONC	Application Object Library

Description



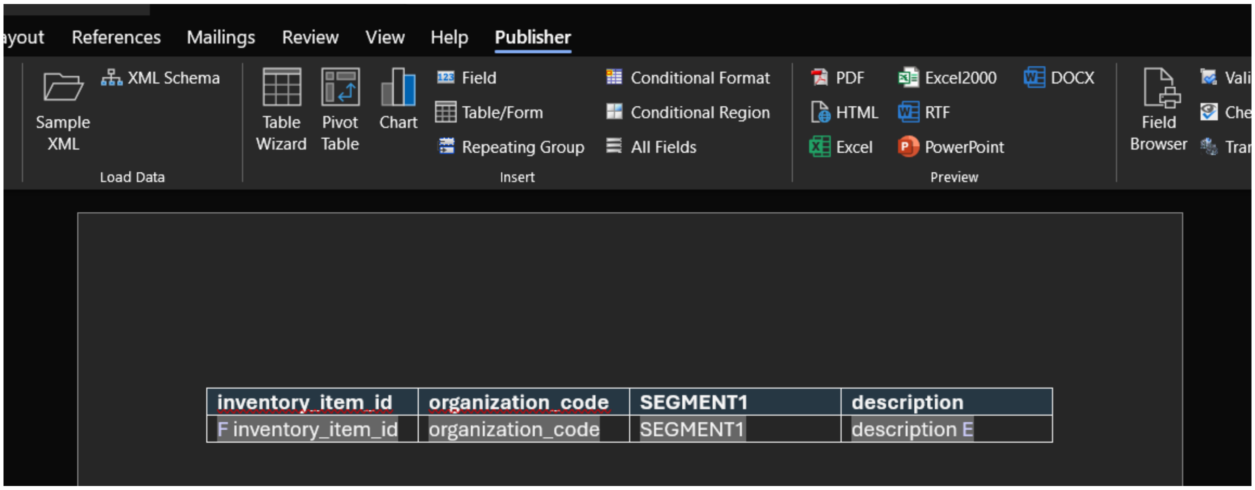
☐ Auto Refresh (X)

Request ID	Name	Parent	Phase	Status	Parameters
7749251	ONKAR_PLSQL_CONC		Completed	Normal	
7749250	ONKAR_PLSQL_CONC		Completed	Error	
7749191	ONKAR_PIVOT_TABLE_C(		Completed	Error	
7749190	ONKAR_PIVOT_TABLE_C(		Completed	Error	
7749036	ONKAR_BURSTING_REPC		Completed	Error	
7748548	Bursting ONKAR_BURSTIN		Completed	Normal	Y, 7748547, N
7748547	ONKAR_BURSTING_REPC		Completed	Normal	
7748370	Bursting ONKAR_BURSTIN		Completed	Normal	Y, 7748369, N
7748369	ONKAR_BURSTING_REPC		Completed	Normal	
7748364	Bursting ONKAR_BURSTIN		Completed	Normal	Y, 7748362, N



```
<ROWSET>
<ROW>
<inventory_item_id>4656</inventory_item_id>
<organization_code>M4</organization_code>
<SEGMENT1>SMC4002</SEGMENT1>
<description>Raw Chip Low Grade</description>
</ROW>
<ROW>
<inventory_item_id>4658</inventory_item_id>
<organization_code>M4</organization_code>
<SEGMENT1>ASM1000</SEGMENT1>
<description>Final Chip High Grade</description>
</ROW>
<ROW>
<inventory_item_id>4654</inventory_item_id>
<organization_code>M4</organization_code>
<SEGMENT1>SMC4001</SEGMENT1>
<description>Raw Chip Medium Grade</description>
</ROW>
<ROW>
<inventory_item_id>4650</inventory_item_id>
<organization_code>M4</organization_code>
<SEGMENT1>SMC3002</SEGMENT1>
<description>Die Low Grade</description>
</ROW>
<ROW>
<inventory_item_id>4652</inventory_item_id>
<organization_code>M4</organization_code>
<SEGMENT1>SMC4000</SEGMENT1>
<description>Raw Chip High Grade</description>
</ROW>
<ROW>
<inventory_item_id>4111</inventory_item_id>
<organization_code>M4</organization_code>
<SEGMENT1>CM50000</SEGMENT1>
<description>100 MM Raw Wafer</description>
```

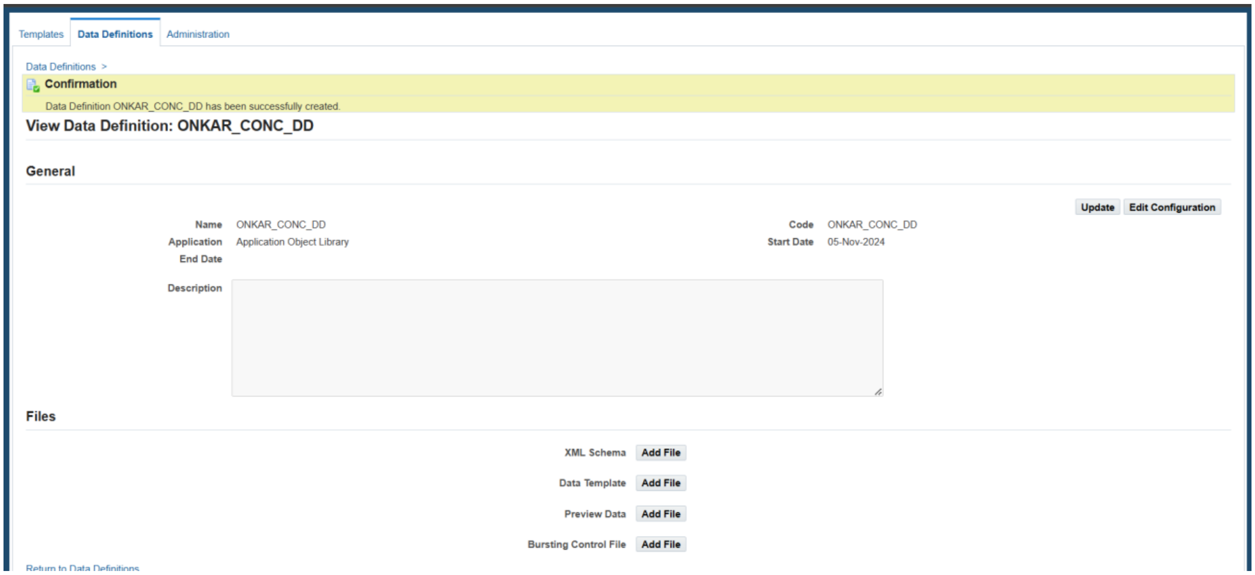
4. SAVE THE ABOVE FILE IN .XML AND CREATE TEMPLATE:



inventory_item_id	organization_code	SEGMENT1	description
F inventory_item_id	organization_code	SEGMENT1	description E

5. NOW CREATE DATA DEFINITION AND TEMPLATE:

IN PLSQL APPROACH WE DO NOT NEED TO MATCH THE DD CODE WITH CONC PROGRAM SHORT NAME.



Confirmation

Data Definition ONKAR_CONC_DD has been successfully created.

View Data Definition: ONKAR_CONC_DD

General

Name	ONKAR_CONC_DD	Code	ONKAR_CONC_DD
Application	Application Object Library	Start Date	05-Nov-2024
End Date			
Description			

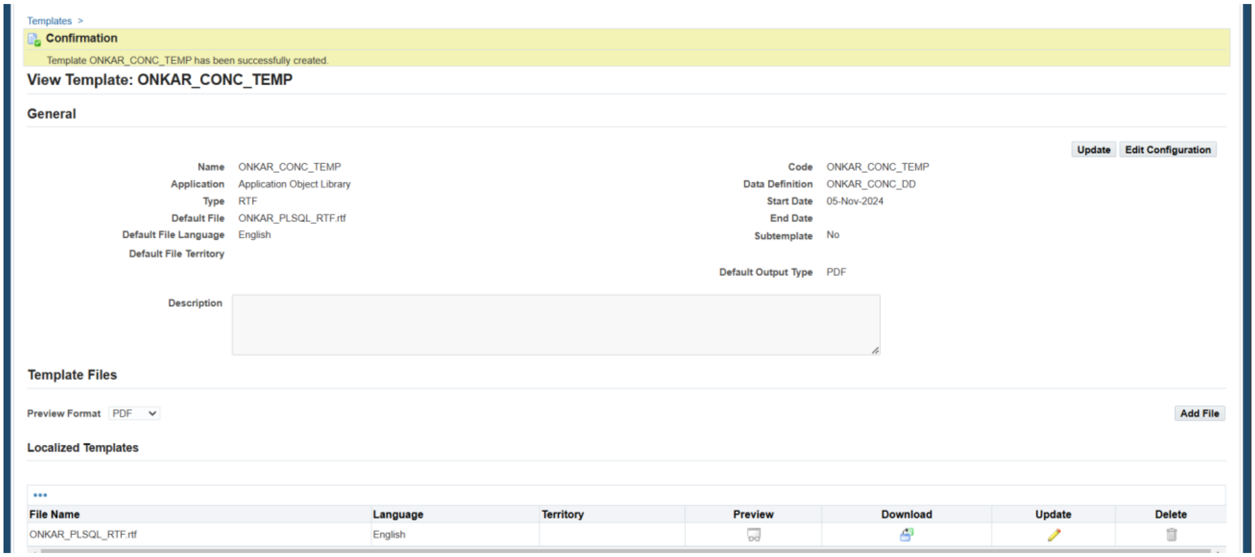
Files

XML Schema

Data Template

Preview Data

Bursting Control File



Confirmation

Template ONKAR_CONC_TEMP has been successfully created.

View Template: ONKAR_CONC_TEMP

General

Name	ONKAR_CONC_TEMP	Code	ONKAR_CONC_TEMP
Application	Application Object Library	Data Definition	ONKAR_CONC_DD
Type	RTF	Start Date	05-Nov-2024
Default File	ONKAR_PLSQL_RTF.rtf	End Date	
Default File Language	English	Subtemplate	No
Default File Territory			
		Default Output Type	PDF
Description			

Template Files

Preview Format PDF

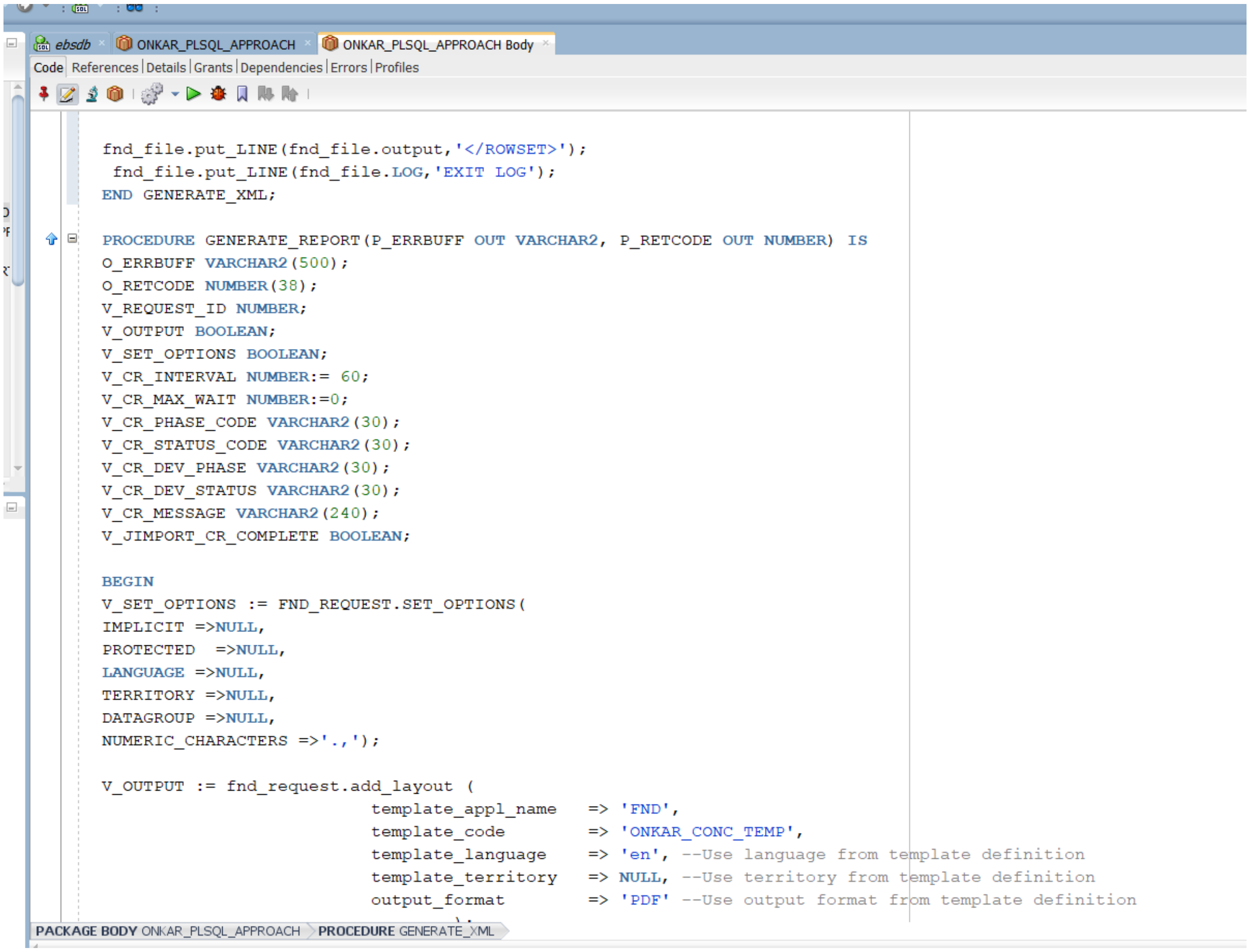
Localized Templates

File Name	Language	Territory	Preview	Download	Update	Delete
ONKAR_PLSQL_RTF.rtf	English				✎	

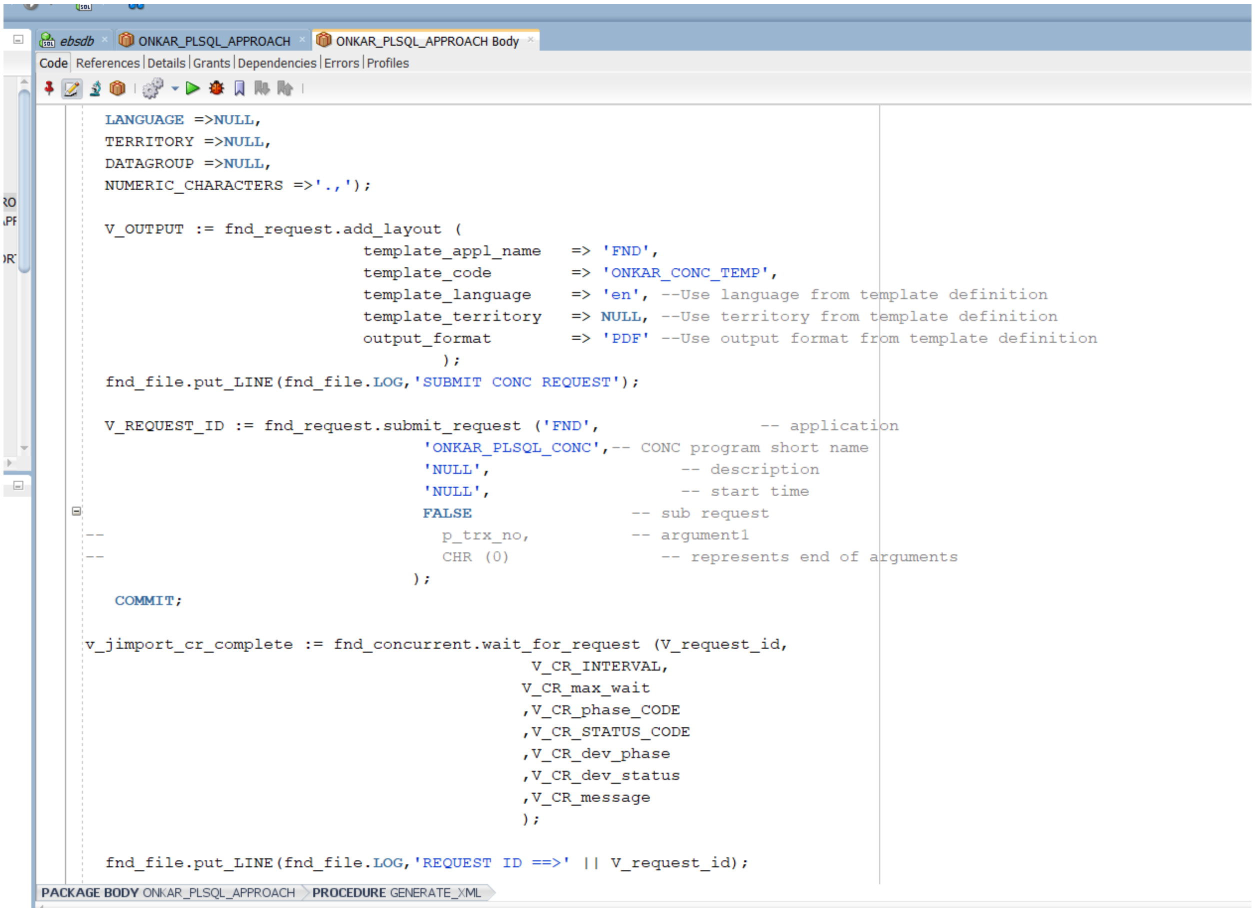
NOW RERUN THE REQUEST BUT NO CREATION OF FORM TAKES PLACE. WE ONLY GET XML DATA AS OUTPUT.

6. CREATE PLSQL API TO INVOKE OUR TEMPLATE:

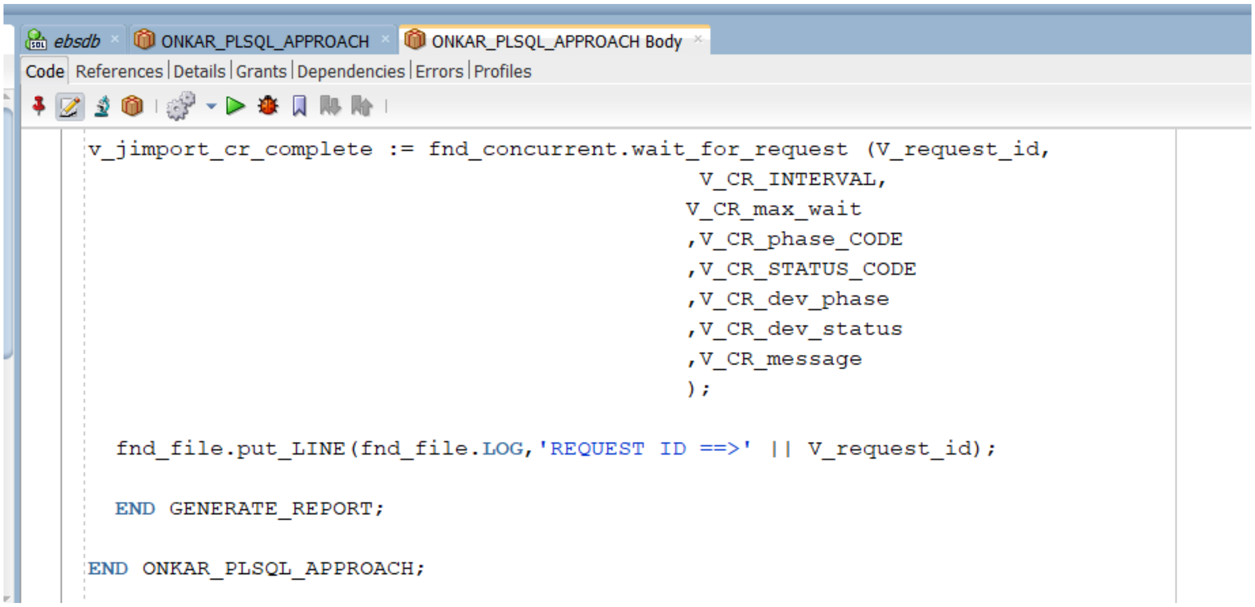
WE ARE STILL GETTING ONLY XML OUTPUT SO WE NEED TO USE **‘ADD LAYOUT METHOD’**. AS WE ARE CALLING ‘XML_GENERATE’ PROCEDURE FIRST THUS WE CANNOT ABLE TO GET LAYOUT SO TO RESOLVE THIS WE MUST CALL THE RTF LAYOUT FIRST BEFORE EVEN GENRATING THE XML DATA. TO DO THAT CREATE PROCEDURE WITH ‘ADD LAYOUT’ LOGIC.



```
fnd_file.put_LINE(fnd_file.output,'</ROWSET>');
 fnd_file.put_LINE(fnd_file.LOG,'EXIT LOG');
END GENERATE_XML;

PROCEDURE GENERATE_REPORT(P_ERRBUFF OUT VARCHAR2, P_RETCODE OUT NUMBER) IS
O_ERRBUFF VARCHAR2(500);
O_RETCODE NUMBER(38);
V_REQUEST_ID NUMBER;
V_OUTPUT BOOLEAN;
V_SET_OPTIONS BOOLEAN;
V_CR_INTERVAL NUMBER:= 60;
V_CR_MAX_WAIT NUMBER:=0;
V_CR_PHASE_CODE VARCHAR2(30);
V_CR_STATUS_CODE VARCHAR2(30);
V_CR_DEV_PHASE VARCHAR2(30);
V_CR_DEV_STATUS VARCHAR2(30);
V_CR_MESSAGE VARCHAR2(240);
V_JIMPORT_CR_COMPLETE BOOLEAN;

BEGIN
V_SET_OPTIONS := FND_REQUEST.SET_OPTIONS(
IMPLICIT =>NULL,
PROTECTED  =>NULL,
LANGUAGE =>NULL,
TERRITORY =>NULL,
DATAGROUP =>NULL,
NUMERIC_CHARACTERS =>'.,');

V_OUTPUT := fnd_request.add_layout (
                          template_appl_name   => 'FND',
                          template_code        => 'ONKAR_CONC_TEMP',
                          template_language    => 'en', --Use language from template definition
                          template_territory   => NULL, --Use territory from template definition
                          output_format        => 'PDF' --Use output format from template definition
```

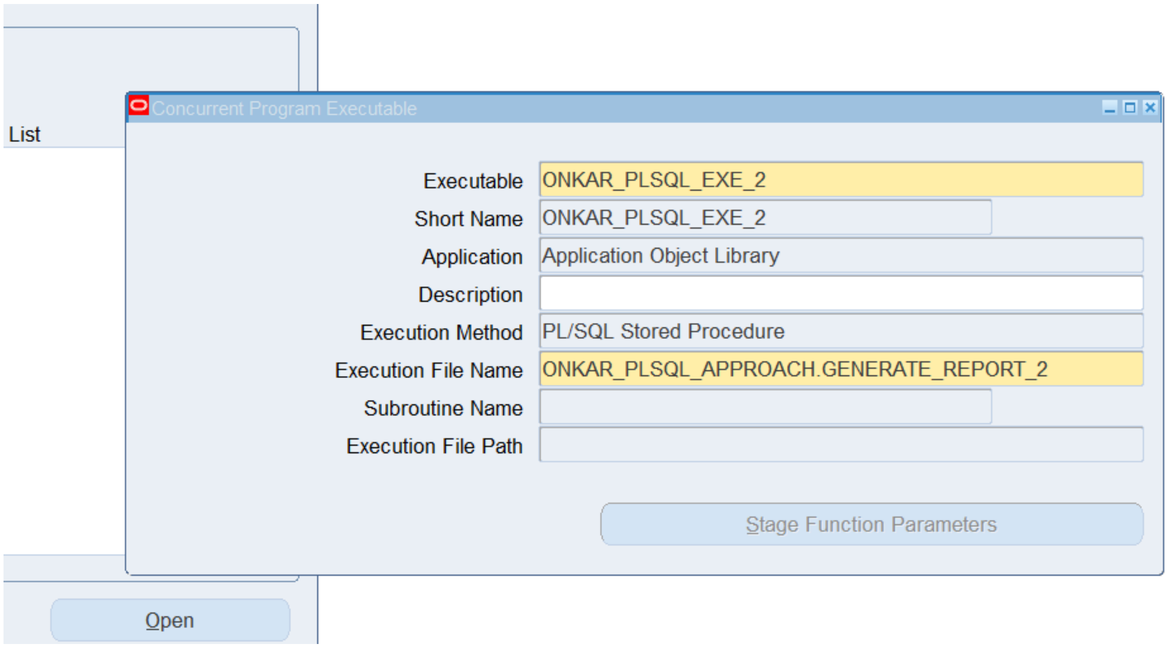


```
   LANGUAGE =>NULL,
   TERRITORY =>NULL,
   DATAGROUP =>NULL,
   NUMERIC_CHARACTERS =>'.,');

   V_OUTPUT := fnd_request.add_layout (
                          template_appl_name   => 'FND',
                          template_code        => 'ONKAR_CONC_TEMP',
                          template_language    => 'en', --Use language from template definition
                          template_territory   => NULL, --Use territory from template definition
                          output_format        => 'PDF' --Use output format from template definition
                                  );
   fnd_file.put_LINE(fnd_file.LOG,'SUBMIT CONC REQUEST');

   V_REQUEST_ID := fnd_request.submit_request ('FND',                  -- application
                                'ONKAR_PLSQL_CONC',-- CONC program short name
                                'NULL',                      -- description
                                'NULL',                      -- start time
                                FALSE                   -- sub request
--                                p_trx_no,           -- argument1
--                                CHR (0)                -- represents end of arguments
                               );
    COMMIT;

v_jimport_cr_complete := fnd_concurrent.wait_for_request (V_request_id,
                                             V_CR_INTERVAL,
                                            V_CR_max_wait
                                            ,V_CR_phase_CODE
                                            ,V_CR_STATUS_CODE
                                            ,V_CR_dev_phase
                                            ,V_CR_dev_status
                                            ,V_CR_message
                                            );

   fnd_file.put_LINE(fnd_file.LOG,'REQUEST ID ==>' || V_request_id);
```

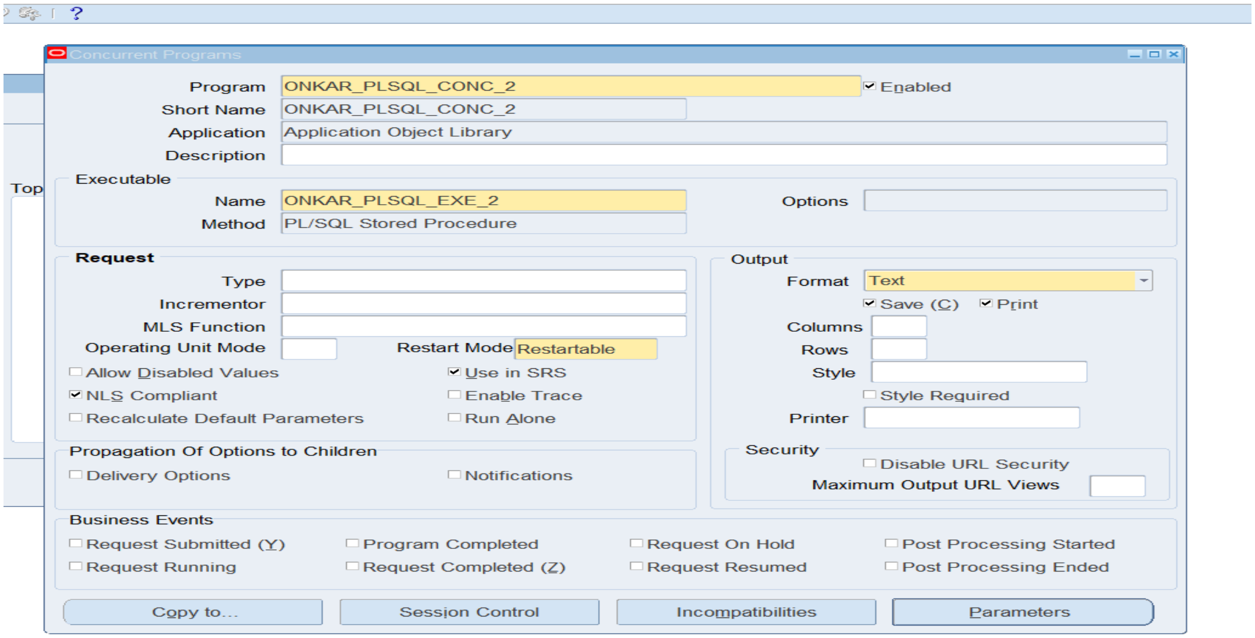


```
v_jimport_cr_complete := fnd_concurrent.wait_for_request (V_request_id,
                                             V_CR_INTERVAL,
                                            V_CR_max_wait
                                            ,V_CR_phase_CODE
                                            ,V_CR_STATUS_CODE
                                            ,V_CR_dev_phase
                                            ,V_CR_dev_status
                                            ,V_CR_message
                                            );

  fnd_file.put_LINE(fnd_file.LOG,'REQUEST ID ==>' || V_request_id);

  END GENERATE_REPORT;

END ONKAR_PLSQL_APPROACH;
```

7. NOW RUN THE REQUEST AGAIN:



Executable	ONKAR_PLSQL_EXE_2
Short Name	ONKAR_PLSQL_EXE_2
Application	Application Object Library
Description	
Execution Method	PL/SQL Stored Procedure
Execution File Name	ONKAR_PLSQL_APPROACH.GENERATE_REPORT_2
Subroutine Name	
Execution File Path	



Program: ONKAR_PLSQL_CONC_2 ☑ Enabled
Short Name: ONKAR_PLSQL_CONC_2
Application: Application Object Library
Description:

Executable
Name: ONKAR_PLSQL_EXE_2
Options:
Method: PL/SQL Stored Procedure

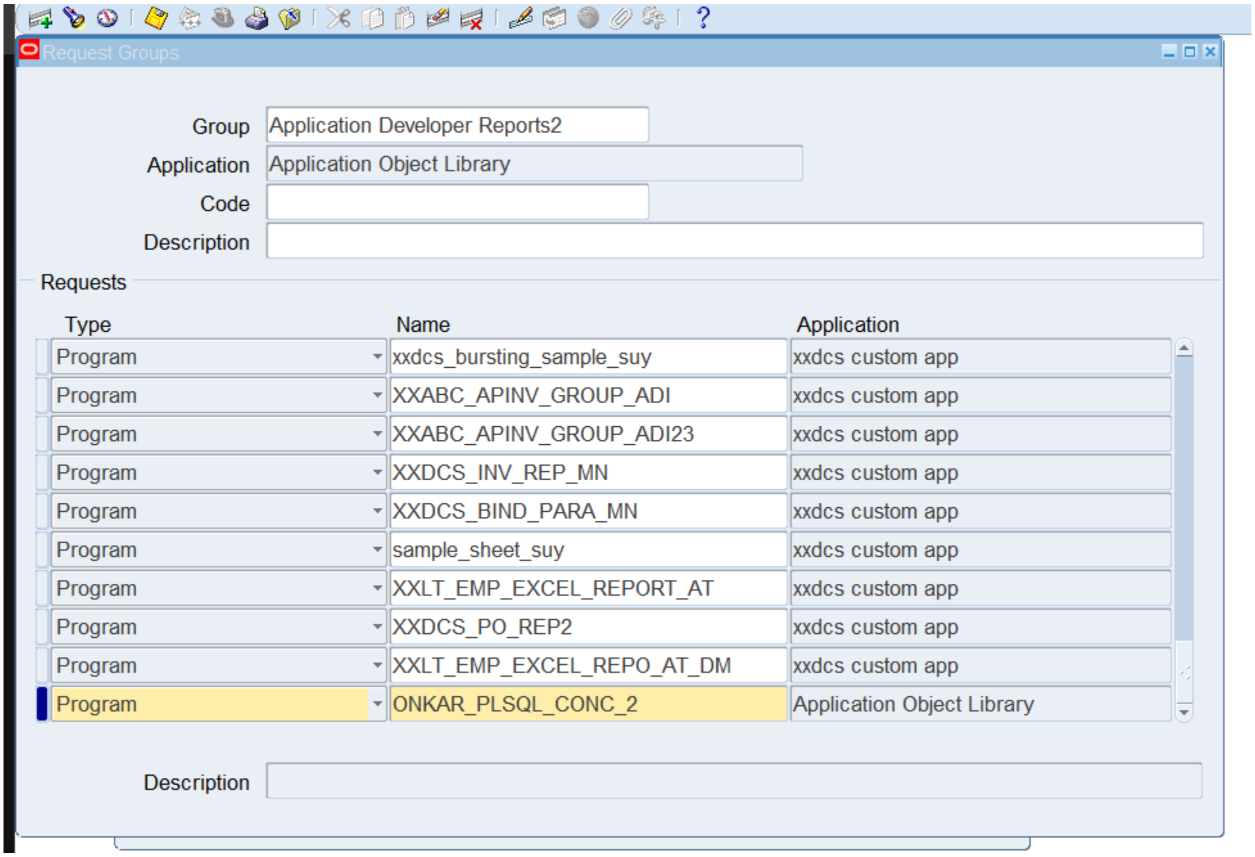
Request
Type:
Incrementor:
MLS Function:
Operating Unit Mode:
Restart Mode: Restartable
□ Allow Disabled Values
☑ Use in SRS
☑ NLS Compliant
□ Enable Trace
□ Recalculate Default Parameters
□ Run Alone

Output
Format: Text
☑ Save (C) ☑ Print
Columns:
Rows:
Style:
□ Style Required
Printer:

Propagation Of Options to Children
□ Delivery Options
□ Notifications

Security
□ Disable URL Security
Maximum Output URL Views:

Business Events
□ Request Submitted (Y)
□ Program Completed
□ Request On Hold
□ Post Processing Started
□ Request Running
□ Request Completed (Z)
□ Request Resumed
□ Post Processing Ended



Group: Application Developer Reports2
Application: Application Object Library
Code:
Description:

Requests

Type	Name	Application
Program	xxdcs_bursting_sample_suy	xxdcs custom app
Program	XXABC_APINV_GROUP_ADI	xxdcs custom app
Program	XXABC_APINV_GROUP_ADI23	xxdcs custom app
Program	XXDCS_INV_REP_MN	xxdcs custom app
Program	XXDCS_BIND_PARA_MN	xxdcs custom app
Program	sample_sheet_suy	xxdcs custom app
Program	XXLT_EMP_EXCEL_REPORT_AT	xxdcs custom app
Program	XXDCS_PO_REP2	xxdcs custom app
Program	XXLT_EMP_EXCEL_REPO_AT_DM	xxdcs custom app
Program	ONKAR_PLSQL_CONC_2	Application Object Library

Description:

```
create or replace PACKAGE BODY ONKAR_PLSQL_APPROACH IS
```

```
 PROCEDURE GENERATE_XML(P_ERRBUFF OUT VARCHAR2, P_RETCODE OUT NUMBER)
IS
 CURSOR C_data IS select inventory_item_id, organization_code,segment1,description
from mtl_system_items_b a ,org_organization_definitions b
 where a.organization_id = b.organization_id AND ROWNUM<=15;
 BEGIN
 fnd_file.put_LINE(fnd_file.LOG,'LOG IS ADDED FOR DEBUGGING PURPOSE');
 fnd_file.put_LINE(fnd_file.output,'<ROWSET>');

 for rec in C_data loop
 fnd_file.put_LINE(fnd_file.output,'<ROW>');

 fnd_file.put_LINE(fnd_file.output,'<inventory_item_id>' || REC.inventory_item_id ||
'</inventory_item_id>');
 fnd_file.put_LINE(fnd_file.output,'<organization_code>' || REC.organization_code ||
'</organization_code>');
 fnd_file.put_LINE(fnd_file.output,'<SEGMENT1>' || REC.SEGMENT1 || '</SEGMENT1>');
 fnd_file.put_LINE(fnd_file.output,'<description>' || REC.description || '</description>');

 fnd_file.put_LINE(fnd_file.output,'</ROW>');
 end loop;

 fnd_file.put_LINE(fnd_file.output,'</ROWSET>');
  fnd_file.put_LINE(fnd_file.LOG,'EXIT LOG');
 END GENERATE_XML;

 PROCEDURE GENERATE_REPORT_2(P_ERRBUFF OUT VARCHAR2, P_RETCODE OUT
NUMBER) IS
 O_ERRBUFF VARCHAR2(500);
 O_RETCODE NUMBER(38);
 V_REQUEST_ID NUMBER;
 V_OUTPUT BOOLEAN;
 V_SET_OPTIONS BOOLEAN;
 V_CR_INTERVAL NUMBER:= 60;
 V_CR_MAX_WAIT NUMBER:=0;
 V_CR_PHASE_CODE VARCHAR2(30);
 V_CR_STATUS_CODE VARCHAR2(30);
 V_CR_DEV_PHASE VARCHAR2(30);
 V_CR_DEV_STATUS VARCHAR2(30);
 V_CR_MESSAGE VARCHAR2(240);
 V_JIMPORT_CR_COMPLETE BOOLEAN;

 BEGIN
 V_SET_OPTIONS := FND_REQUEST.SET_OPTIONS(
 IMPLICIT =>NULL,
 PROTECTED  =>NULL,
 LANGUAGE =>NULL,
 TERRITORY =>NULL,
 DATAGROUP =>NULL,
 NUMERIC_CHARACTERS =>'.,');

 V_OUTPUT := fnd_request.add_layout (
                  template_appl_name   => 'FND',
                  template_code       => 'ONKAR_CONC_TEMP',
                  template_language    => 'en', --Use language from template definition
                  template_territory   => NULL, --Use territory from template definition
                  output_format       => 'PDF' --Use output format from template definition
                       );
 fnd_file.put_LINE(fnd_file.LOG,'SUBMIT CONC REQUEST');

   V_REQUEST_ID := fnd_request.submit_request ('FND',             -- application
                      'ONKAR_PLSQL_CONC_2',-- CONC program short name
                      'NULL',               -- description
                      'NULL',               -- start time
                      FALSE            -- sub request
--                      p_trx_no,        -- argument1
--                      CHR (0)           -- represents end of arguments
                     );
  COMMIT;

v_jimport_cr_complete := fnd_concurrent.wait_for_request (V_request_id,
                            V_CR_INTERVAL,
                           V_CR_max_wait
                           ,V_CR_phase_CODE
                           ,V_CR_STATUS_CODE
                           ,V_CR_dev_phase
                           ,V_CR_dev_status
                           ,V_CR_message
                           );

 fnd_file.put_LINE(fnd_file.LOG,'REQUEST ID -->' || V_request_id);

 END GENERATE_REPORT_2;

END ONKAR_PLSQL_APPROACH;
```