ORACLE EBS

MINI PROJECT REPORT

MINI PROJECT 2

RAJDEEP DAS

EMPLOYEE ID: 2152683

Email: Rajdeep.Das5@cognizant.com

Email: inquiry@cognizant.com

Acknowledgement

I would like to thank my Trainer **Dr. Geeta Mandar Mete**, who gave me a golden opportunity to work on this project. I'd also like to express my gratitude to Cognizant Technology Solution wholeheartedly.

I must also thank my project group members for the immense support and help during this project. Without their help, completing this project would have been very difficult.

Date: 03-07-2022

Rajdeep Das
Employee Id: 2152683
Oracle EBS Track Intern,
Cognizant Technology Solution

Signature, Trainer

START

Q1. Create a Master Detail BI Publisher to display TYPE LOOKUP CODE wise list of
Purchase Orders, Quotations, RFQ.s etc.

STEPS:-

Step 1:

A. I have taken three tables: PO_LINES_ALL, PO_HEADERS_ALL, MTL_SYSTEM_ITEMS_B.
B. I have taken total 4 TYPES LOOKUP CODE.

Code:

```
CREATE OR REPLACE PROCEDURE XXCTS_RAJDEEP_MD_PROJECT(
   ERRBUF OUT VARCHAR2,
   RETCODE OUT VARCHAR2)
IS
  CURSOR ITEM_CUR
  IS
   SELECT PHA.SEGMENT1,
     MSIB.SEGMENT1 AS ITEM_NAME,
     PLA.ITEM_DESCRIPTION,
     PHA.TYPE_LOOKUP_CODE,
     PLA.UNIT_PRICE,
     PLA.QUANTITY
   FROM PO_LINES_ALL PLA,
```

```
    PO_HEADERS_ALL PHA,
    MTL_SYSTEM_ITEMS_B MSIB
  WHERE PLA.PO_HEADER_ID    = PHA.PO_HEADER_ID
  AND PLA.ITEM_ID          = MSIB.INVENTORY_ITEM_ID
  AND PHA.TYPE_LOOKUP_CODE IN ('RFQ', 'QUOTATION', 'BLANKET',
'STANDARD')
  AND MSIB.SEGMENT1 LIKE 'XXCTS_R%';
 ITEM_REC ITEM_CUR%ROWTYPE;
 V_ERROR_CODE  NUMBER;
 V_ERR_MESSAGE VARCHAR2(200);
BEGIN
 OPEN ITEM_CUR;
 FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<?xml version="1.0"?>');
 FND_FILE.PUT_LINE(fnd_file.output,'<ItemRoot>');
 LOOP
  FETCH ITEM_CUR INTO ITEM_REC;
  EXIT
 WHEN ITEM_CUR%NOTFOUND;
  FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<ITEM>');
  FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<SEGMENT1>' || ITEM_REC.SEGMENT1
|| '</SEGMENT1>');
  FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<ITEM_NAME>' ||
ITEM_REC.ITEM_NAME || '</ITEM_NAME>');
```

```
   FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<ITEM_DESCRIPTION>'||
ITEM_REC.ITEM_DESCRIPTION || '</ITEM_DESCRIPTION>');

   FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<TYPE_LOOKUP_CODE>'||
ITEM_REC.TYPE_LOOKUP_CODE || '</TYPE_LOOKUP_CODE>');

   FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<UNIT_PRICE>'||
ITEM_REC.UNIT_PRICE || '</UNIT_PRICE>');

   FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'<QUANTITY>'|| ITEM_REC.QUANTITY
|| '</QUANTITY>');

   FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'</ITEM>');

  END LOOP;

  FND_FILE.PUT_LINE(FND_FILE.OUTPUT,'</ItemRoot>');

  CLOSE ITEM_CUR;

EXCEPTION

WHEN OTHERS THEN

  V_ERROR_CODE  := SQLCODE;

  V_ERR_MESSAGE := SUBSTR(SQLERRM,1,200);

  FND_FILE.PUT_LINE(FND_FILE.LOG, 'Error Code :' || V_ERROR_CODE);

  FND_FILE.PUT_LINE(FND_FILE.LOG,'Error Message :' || V_ERR_MESSAGE);

END;
```

```
CREATE OR REPLACE PROCEDURE XXCTS_RAJDEEP_MD_PROJECT(ERRBUF OUT VARCHAR2, RETCODE OUT
VARCHAR2)
IS
  CURSOR ITEM_CUR
IS
SELECT PHA.SEGMENT1, MSIB.SEGMENT1 AS ITEM_NAME, PLA.ITEM_DESCRIPTION, PHA.TYPE_LOOKUP_CODE,
PLA.UNIT_PRICE, PLA.QUANTITY
FROM PO_LINES_ALL PLA, PO_HEADERS_ALL PHA, MTL_SYSTEM_ITEMS_B MSIB
WHERE PLA.PO_HEADER_ID = PHA.PO_HEADER_ID
AND PLA.ITEM_ID = MSIB.INVENTORY_ITEM_ID
AND PHA.TYPE_LOOKUP_CODE IN ('RFQ', 'QUOTATION', 'BLANKET', 'STANDARD')
AND MSIB.SEGMENT1 LIKE 'XXCTS_R%';
```

Script Output | Query Result

Task completed in 0.953 seconds

```
Procedure XXCTS_RAJDEEP_MD_PROJECT compiled
```

Step 2: Open EBS Application.

Navigate to Application Developer → Concurrent → Executable.

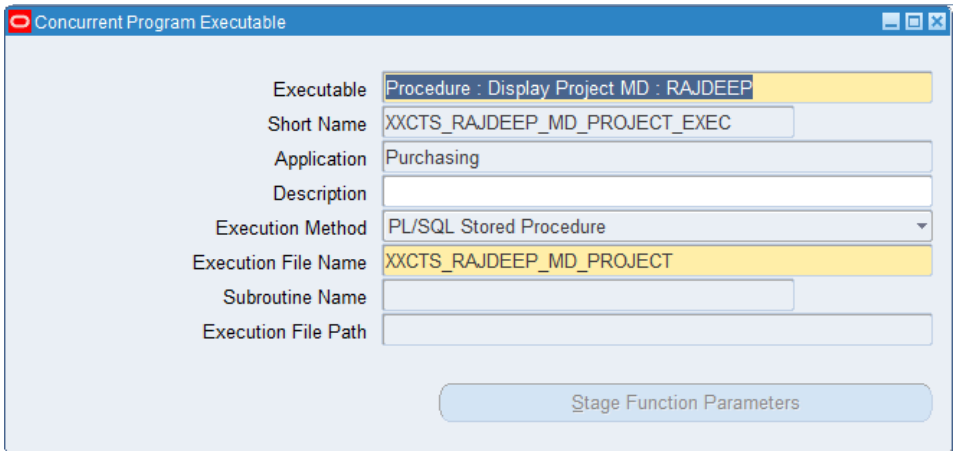
The Concurrent Program Executable dialogue box will appear on the screen.

Executable name: Procedure Display Project MD : RAJDEEP

Short Name: XXCTS_RAJDEEP_MD_PROJECT_EXEC

Application: Purchasing

Execution File name: apps.(Procedure_Name). Here it's apps. XXCTS_RAJDEEP_MD_PROJECT



Click on Save. Close it.

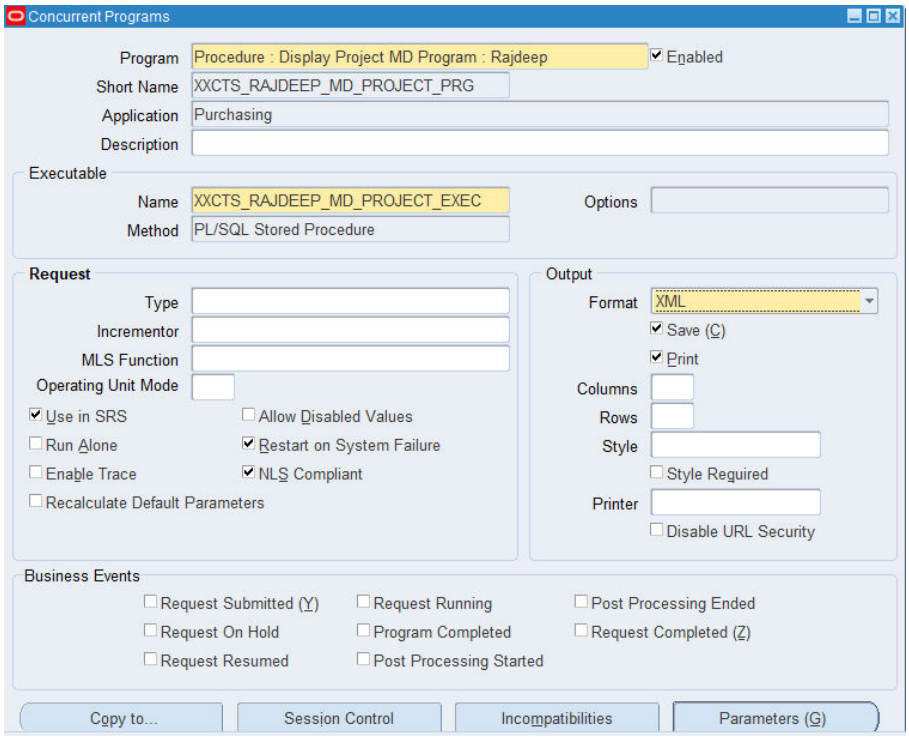
Step 3: Navigate to Application Developer → Concurrent→ Program

Concurrent Programs dialogue box will appear.

Program name:- Procedure : Display Project MD Program : Rajdeep

Short name: XXCTS_RAJDEEP_MD_PROJECT_PRG

Application: Purchasing. Choose Executable name as the previously mentioned in the Executable dialogue box. Choose Format as XML and save it.



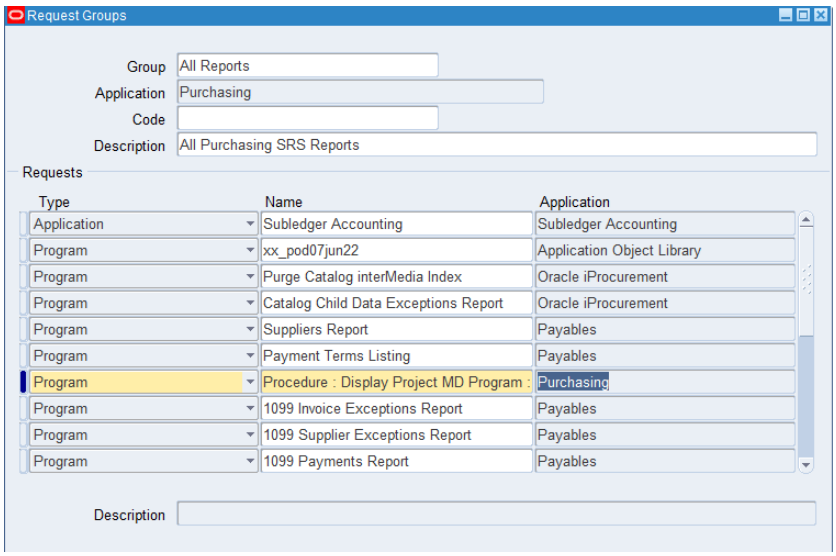
Step 4: Switch to System Administrator responsibility→ Security→ Responsibility→ Request

The Request Groups Dialogue box appears.

I have applied query here.

Group: All Reports

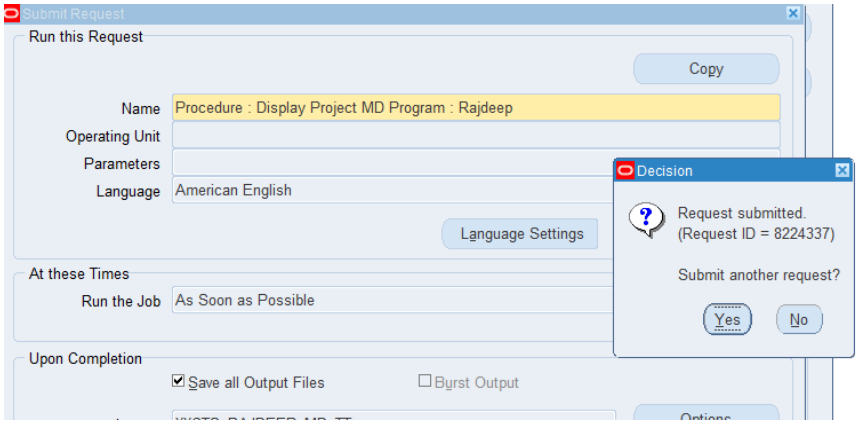
Application: Purchasing



Click on Add row in the Toolbar. And add Procedure: Display Project MD Program: Rajdeep in that row and save it.

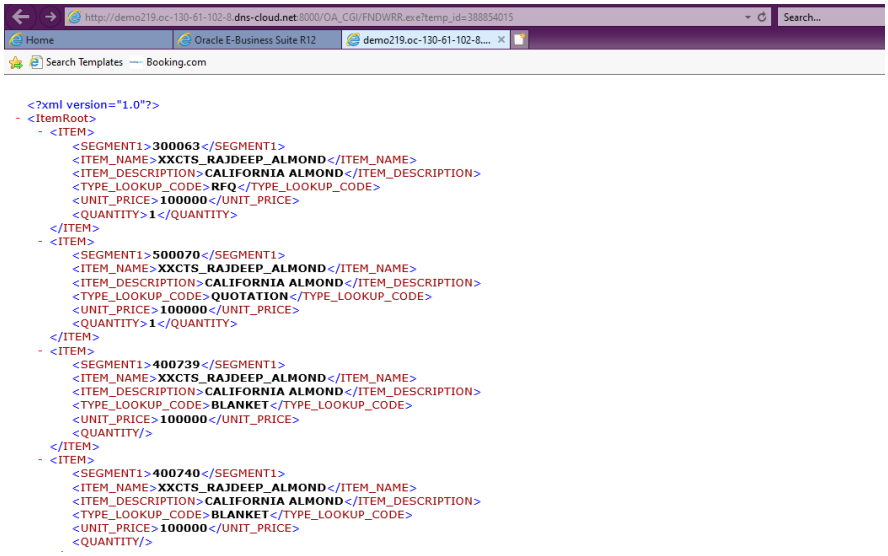
Step 5: Go to View → Requests

Put the name of the Procedure Program name i.e. Procedure: Display Project MD Program : Rajdeep. Click Submit. Click No.



Step 6: The Procedure got compiled and the Output of XML I got on the Internet Explorer.

Below is the screenshot.

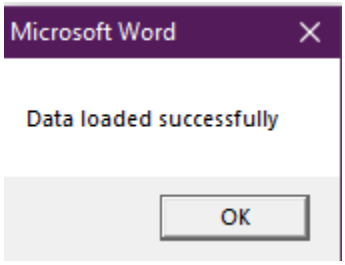


Step 7: Saved it as Procedure_Display_Project_MD.xml.

Step 8: Created a word document file.

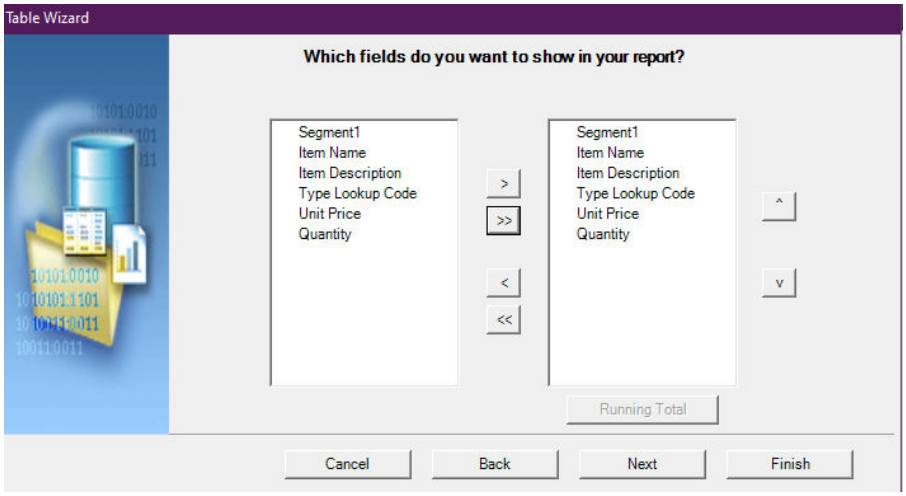
Step 9: Clicked on BI Publisher tab on menu bar.

Step 10: Click on Sample XML option and uploaded that XML file. It will pop up like this.

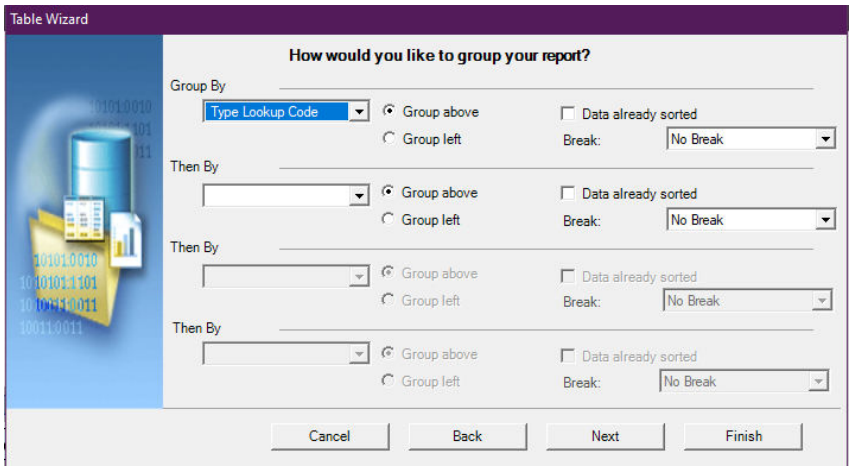


Step 11: Clicked on Table Wizard option on the Menu bar.

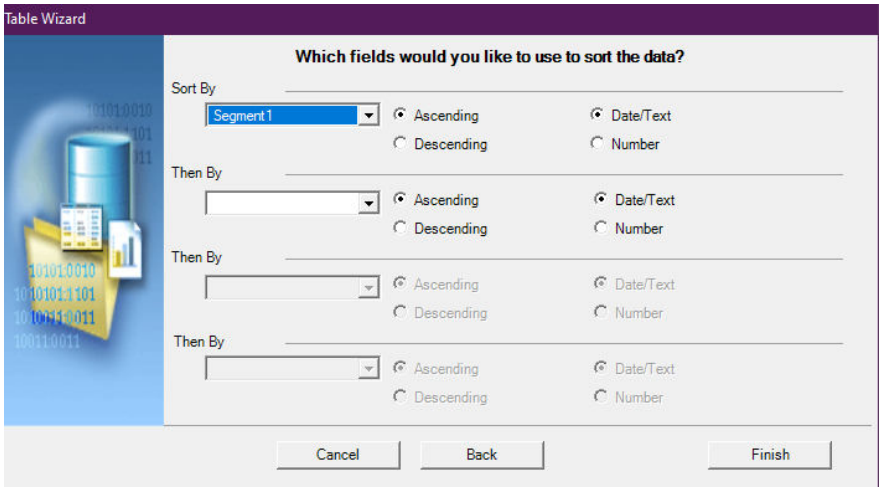
Step 12: Took all the XML tags at one side and created the template using **Group by** condition by applying it on **'Type Lookup Code'** and applying **Sorted by** condition on **'Segment1'**.



Applied Group by Condition on Type_Lookup_Code.



Next Sorted by Segment1



Clicked Finish and the table got created.

MASTER DETAILS PROJECT REPORT

group ITEM by TYPE_LOOKUP_CODE

TYPE_LOOKUP_CODE

Segment1	Item Name	Item Description	Unit Price	Quantity
FOR LOOP SEGMENT1	C ITEM_NAME	ITEM_DESCRIPTION	UNIT_PRICE	QUANTITY END LOOP

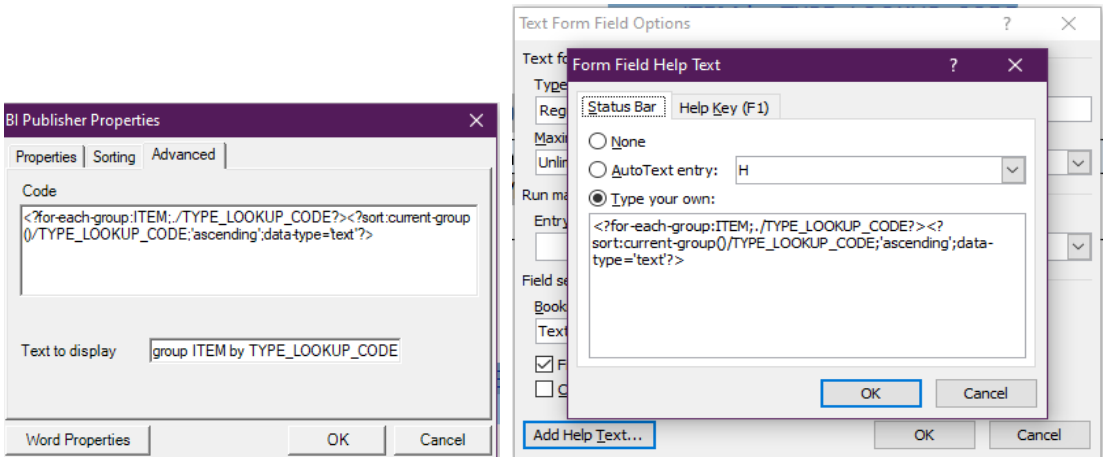
end ITEM by TYPE_LOOKUP_CODE

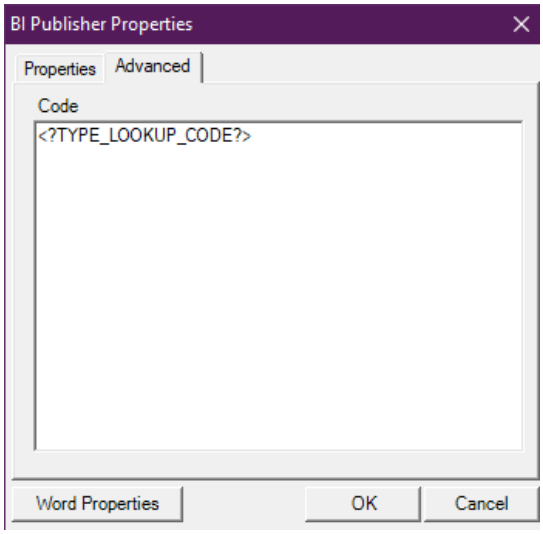
Now I right-clicked on each of the column-name in the table and go to the BI Publisher → Properties option,

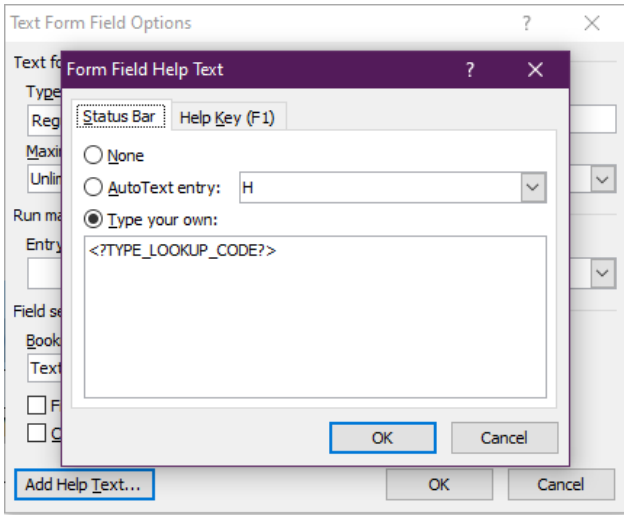
Step 13: In the coming dialogue box, click on 'Advanced' tab, I have just copied the column-tag (<?tagname?>) and then clicked on 'Word Properties' button blow. In the next coming dialogue box, click on 'Add Help Text' button below. Paste the copied text under the 'Status' bar and click on OK.

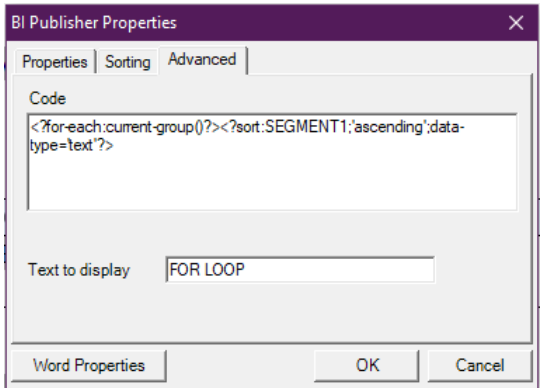
(Only for F, change the label text as "FOR LOOP" and for E it will be END LOOP)

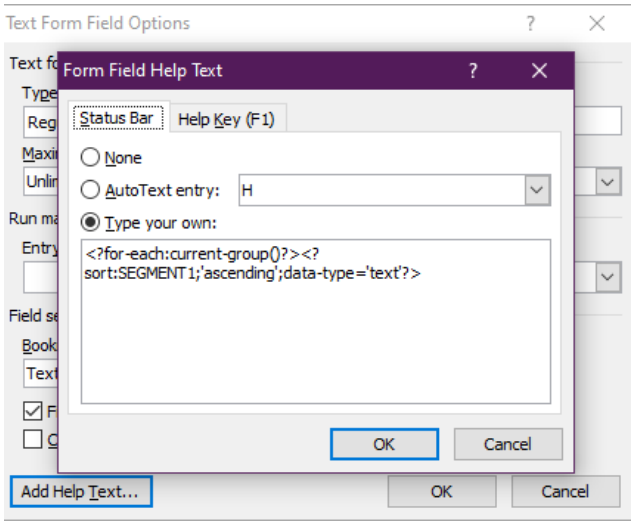
Below are the images each and every column- names tag changing.

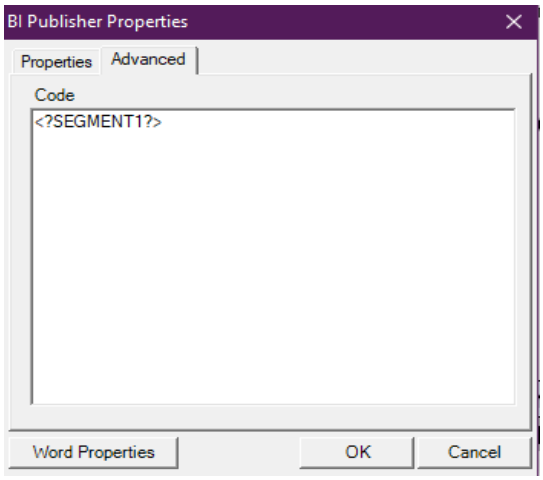


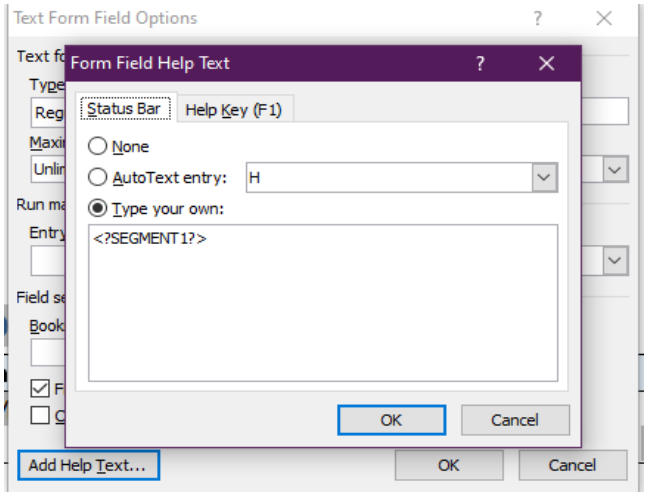

BI Publisher Properties
Properties
Advanced
Code
<?TYPE_LOOKUP_CODE?>
Word Properties
OK
Cancel

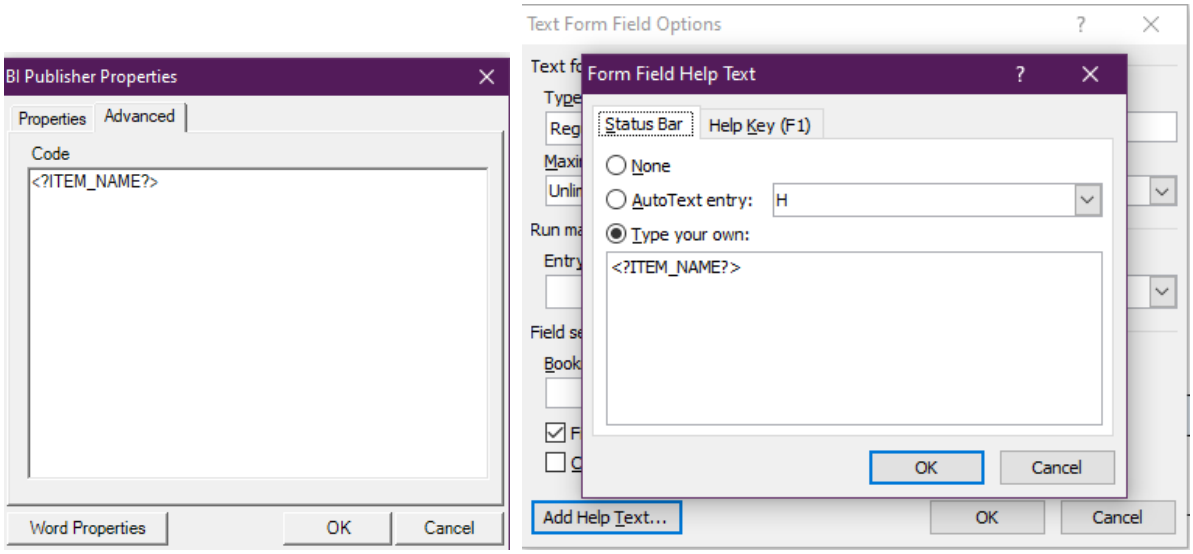

Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?TYPE_LOOKUP_CODE?>
OK
Cancel
Add Help Text...
OK
Cancel

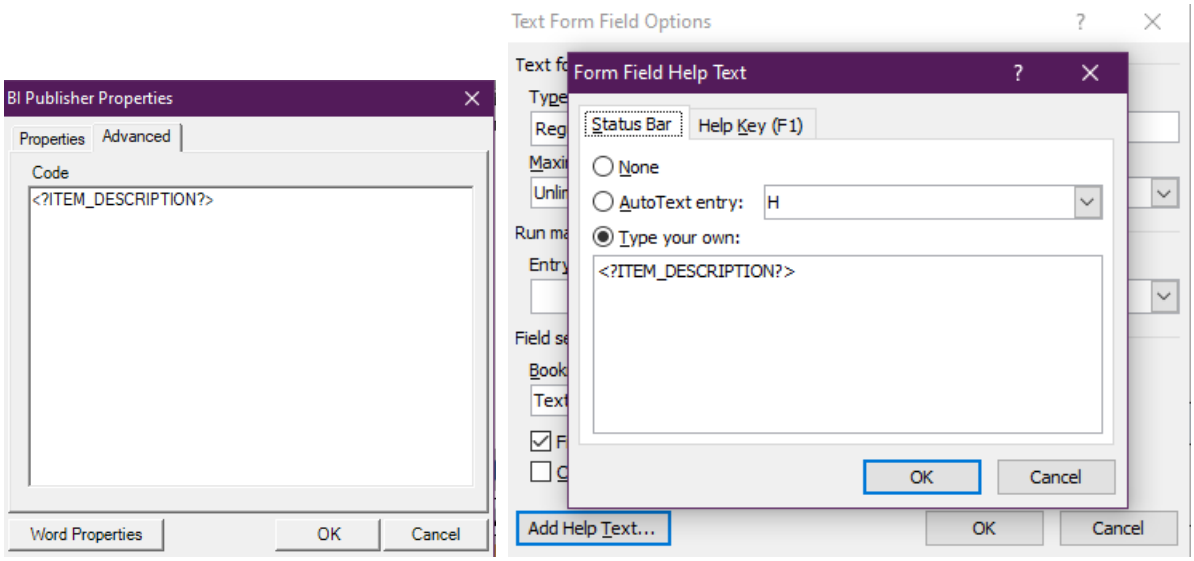

BI Publisher Properties
Properties
Sorting
Advanced
Code
<?for-each:current-group()?><?sort:SEGMENT1;'ascending';data-type='text'?>
Text to display
FOR LOOP
Word Properties
OK
Cancel

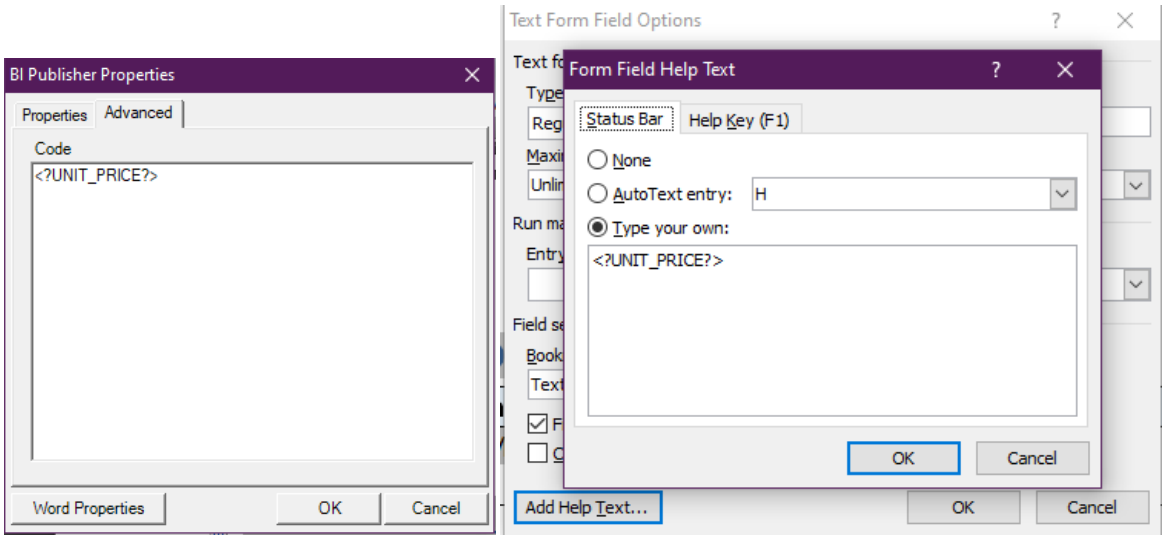

Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?for-each:current-group()?><?
sort:SEGMENT1;'ascending';data-type='text'?>
OK
Cancel
Add Help Text...
OK
Cancel

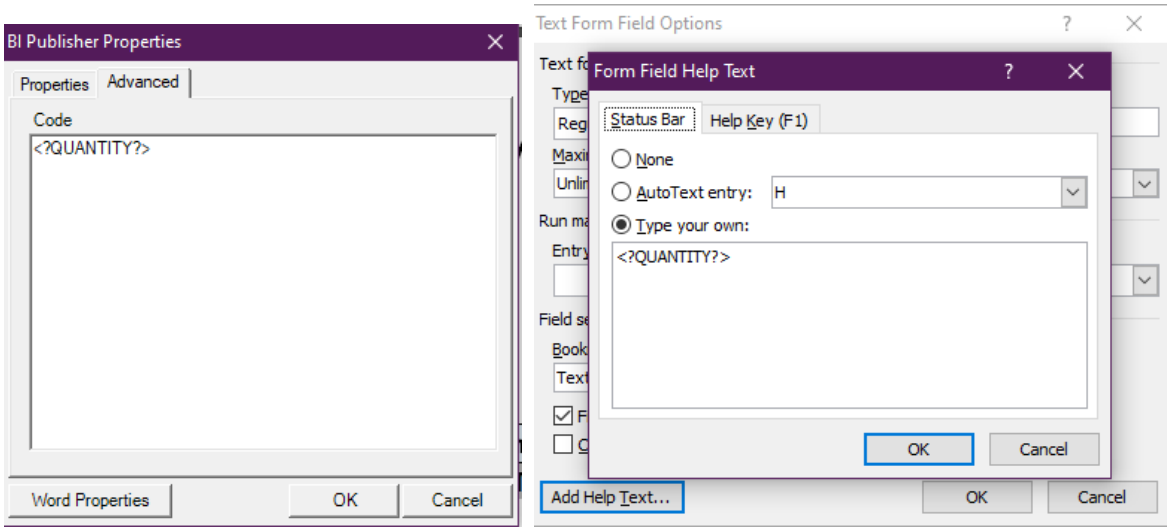

BI Publisher Properties
Properties
Advanced
Code
<?SEGMENT1?>
Word Properties
OK
Cancel

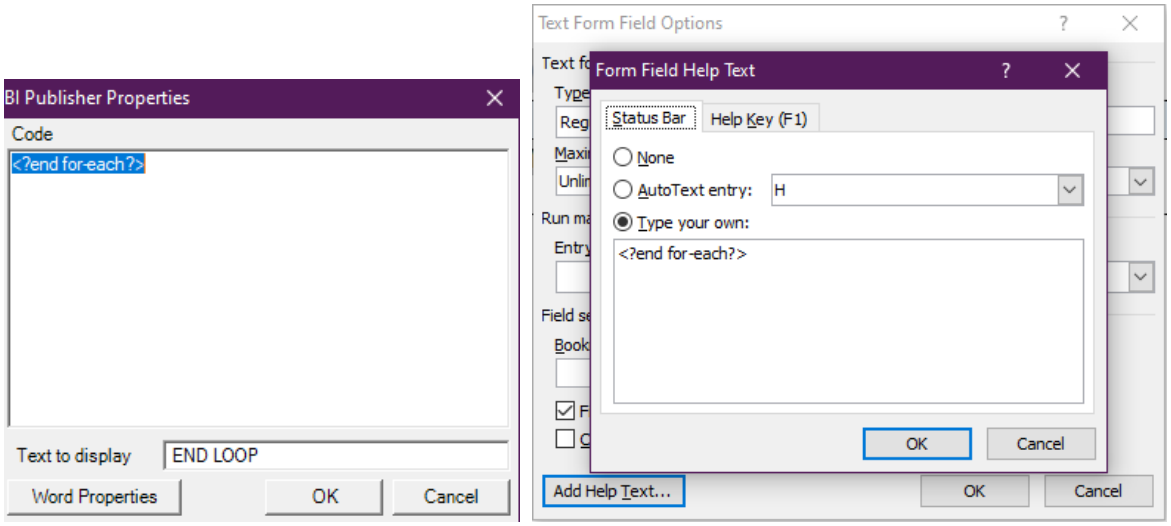

Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?SEGMENT1?>
OK
Cancel
Add Help Text...
OK
Cancel

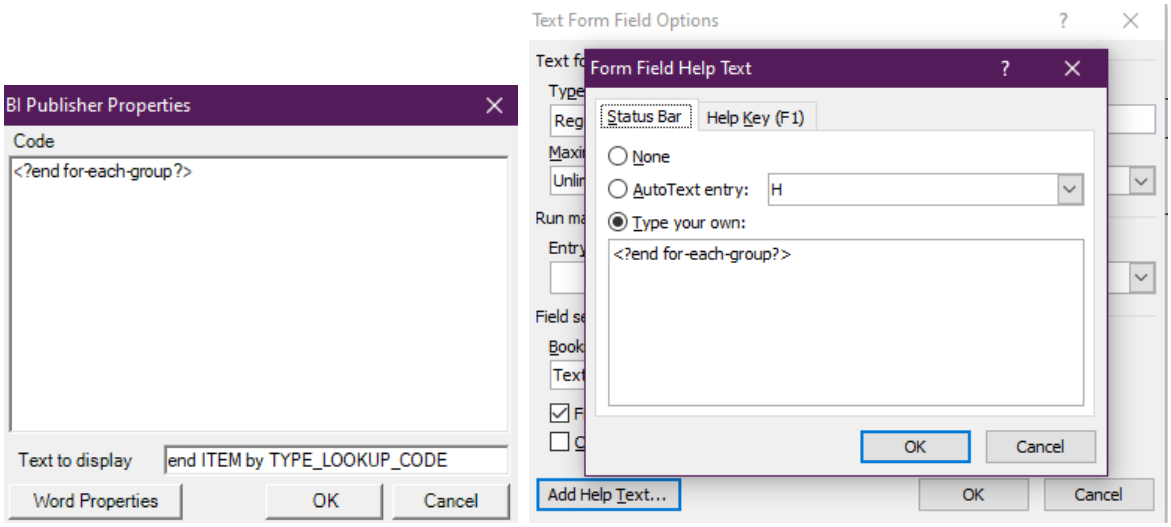

BI Publisher Properties
Properties
Advanced
Code
<?ITEM_NAME?>
Word Properties
OK
Cancel
Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?ITEM_NAME?>
OK
Cancel
Add Help Text...
OK
Cancel


BI Publisher Properties
Properties
Advanced
Code
<?ITEM_DESCRIPTION?>
Word Properties
OK
Cancel
Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?ITEM_DESCRIPTION?>
OK
Cancel
Add Help Text...
OK
Cancel

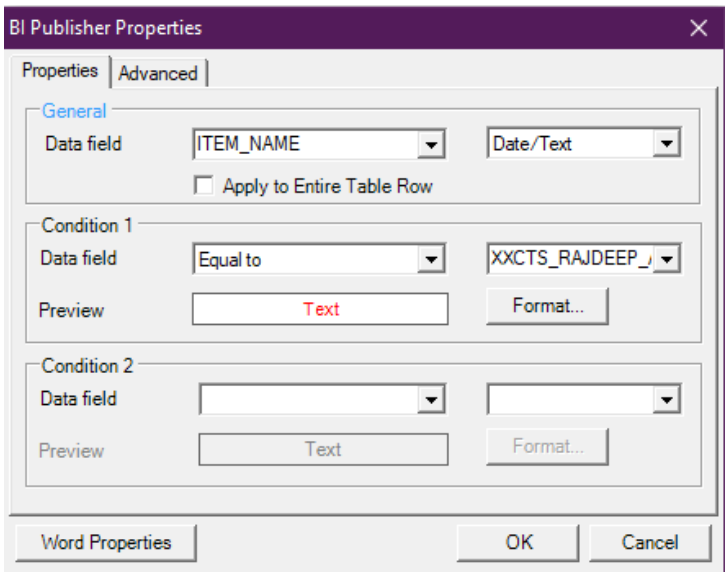

BI Publisher Properties
Properties
Advanced
Code
<?UNIT_PRICE?>
Word Properties
OK
Cancel
Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?UNIT_PRICE?>
OK
Cancel
Add Help Text...
OK
Cancel


BI Publisher Properties
Properties
Advanced
Code
<?QUANTITY?>
Word Properties
OK
Cancel
Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?QUANTITY?>
OK
Cancel
Add Help Text...
OK
Cancel

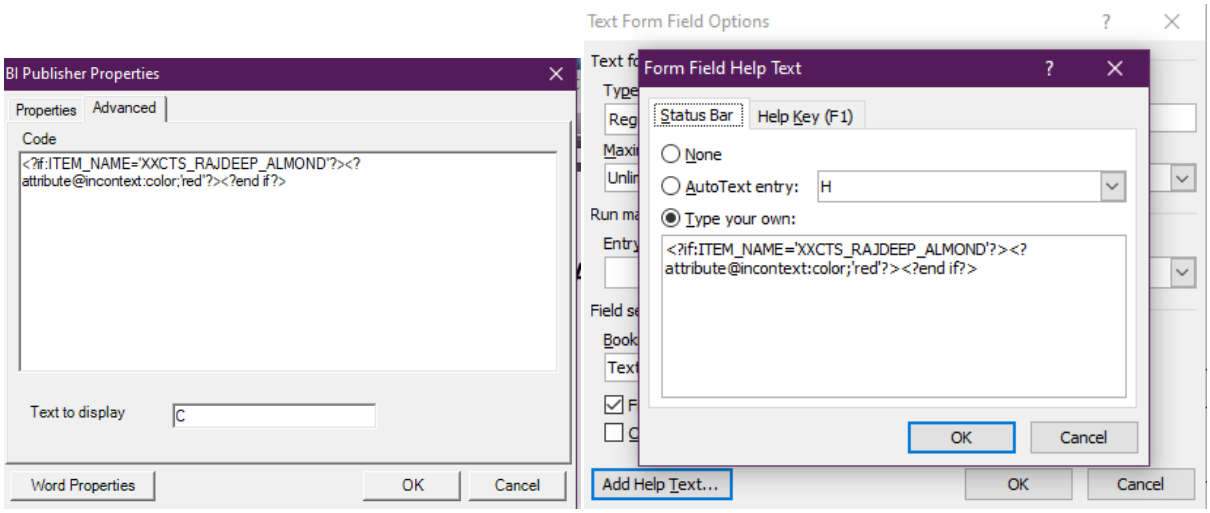

BI Publisher Properties
Code
<?end for-each?>
Text to display
END LOOP
Word Properties
OK
Cancel
Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?end for-each?>
OK
Cancel
Add Help Text...
OK
Cancel


BI Publisher Properties
Code
<?end for-each-group?>
Text to display
end ITEM by TYPE_LOOKUP_CODE
Word Properties
OK
Cancel
Text Form Field Options
Form Field Help Text
Status Bar
Help Key (F1)
None
AutoText entry:
H
Type your own:
<?end for-each-group?>
OK
Cancel
Add Help Text...
OK
Cancel

Step 14: Applied Conditional Format on ITEM_NAME.



Also I have changed the tag for Conditional Format.



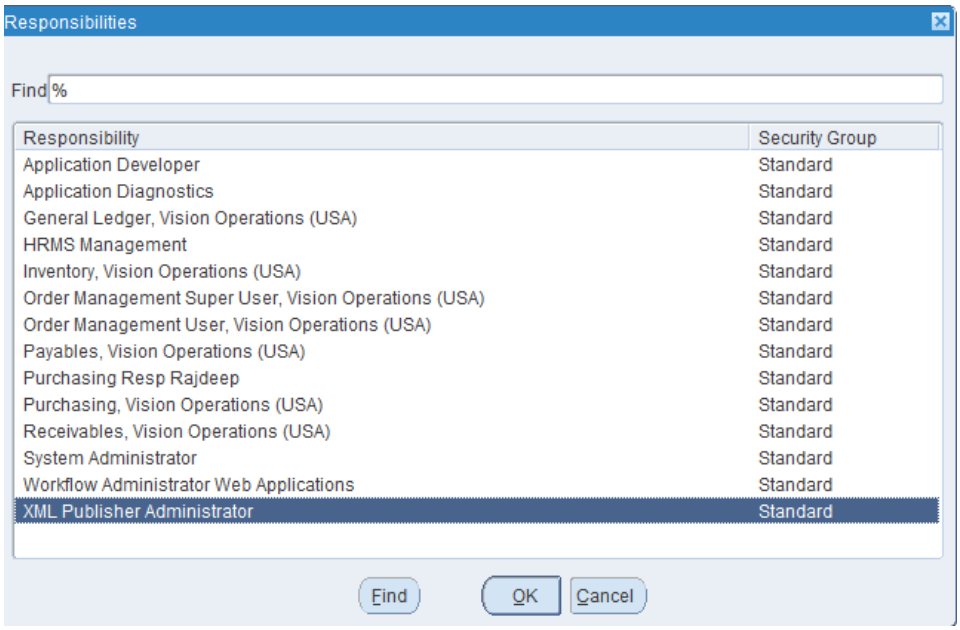
I have chosen XXCTS_RAJDEEP_ALMOND as Data Field Condition.

Now repeating Step 13 each and every column name in the table.

Step 15: Saved the file as 'MASTER_DETAILS _PROJECT_REPORT.rtf'

Step 16: Clicked on PDF on the BI Publisher tab to check if the pdf id generating or no.

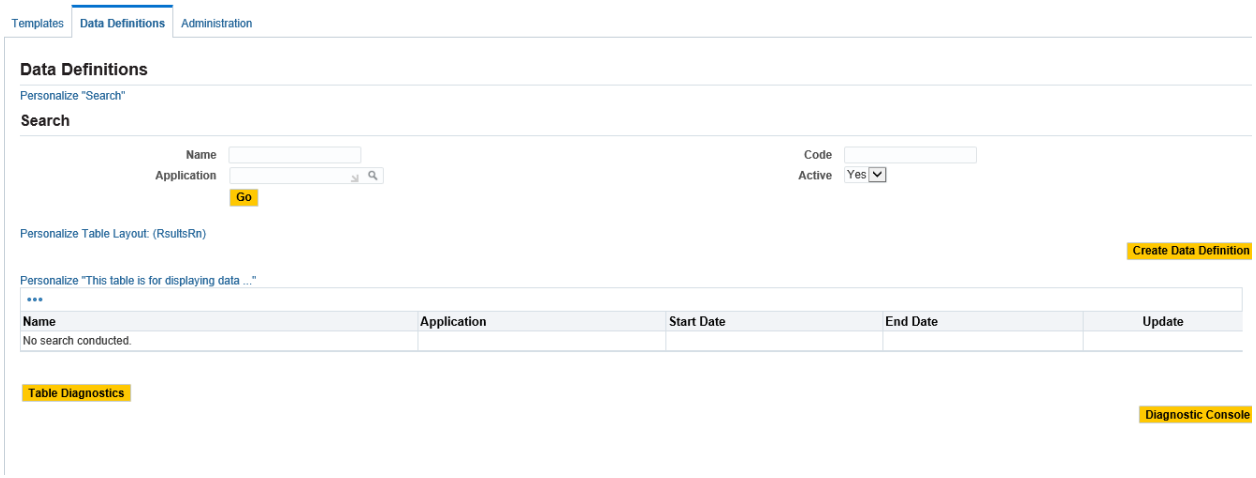
Step 17: Now switch to XML Publisher Administrator responsibility and click OK.



Step 18: Clicked on the Home →Data Definition option.

Step 19: A New tab will be opened on the Internet Explorer.

Step 20: Clicked on the 'Create Data Definition' in the right side.

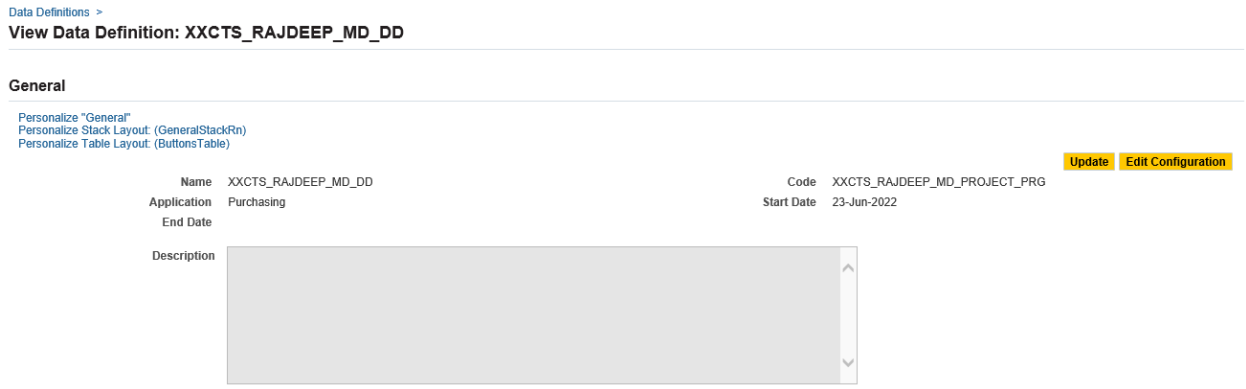


Step 21: Given the Data Definition name as 'XXCTS_RAJDEEP_MD_DD'.

Step 22: Chosen the Application as 'Purchasing' option.

Step 23: In the Code, given the Short Concurrent name of the process as XXCTS_RAJDEEP_MD_PROJECT_PRG.

Step 24: Create Apply.

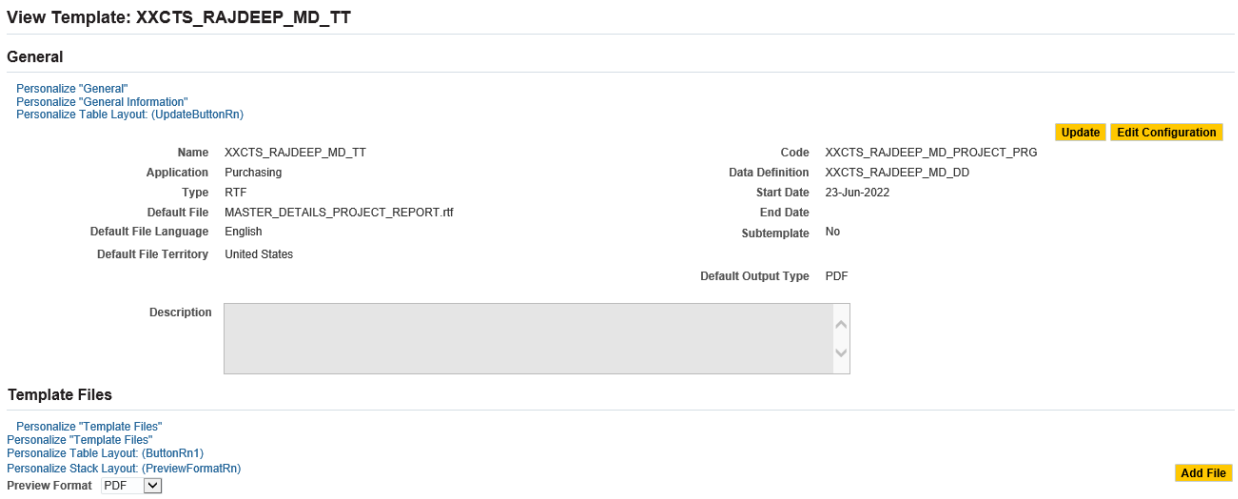


Step 25: Go to the Templates tab above and click on Create Template in the right hand side.

Step 26: Given the template name as 'XXCTS_RAJDEEP_MD_TT'. Application as Purchasing, Code name is the same Concurrent name of the Procedure i.e. XXCTS_RAJDEEP_MD_PROJECT_PRG.

Step 27: Choose the file type as 'RTF'. Default Output Type as PDF.

Step 28: Under the Template File section, browse the path of the recently made MASTER_DETAILS_PROJECT_REPORT.rtf file. Choose the Language as English. Choose the Territory as United States. Click on Apply button.

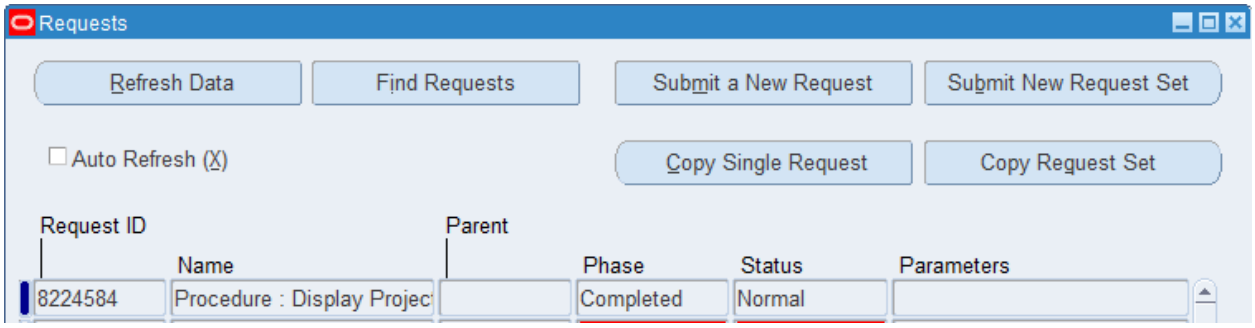


Step 29: Gone to the Oracle EBS Java Application and switch to the Responsibility. Purchasing, Vision Operations (USA).

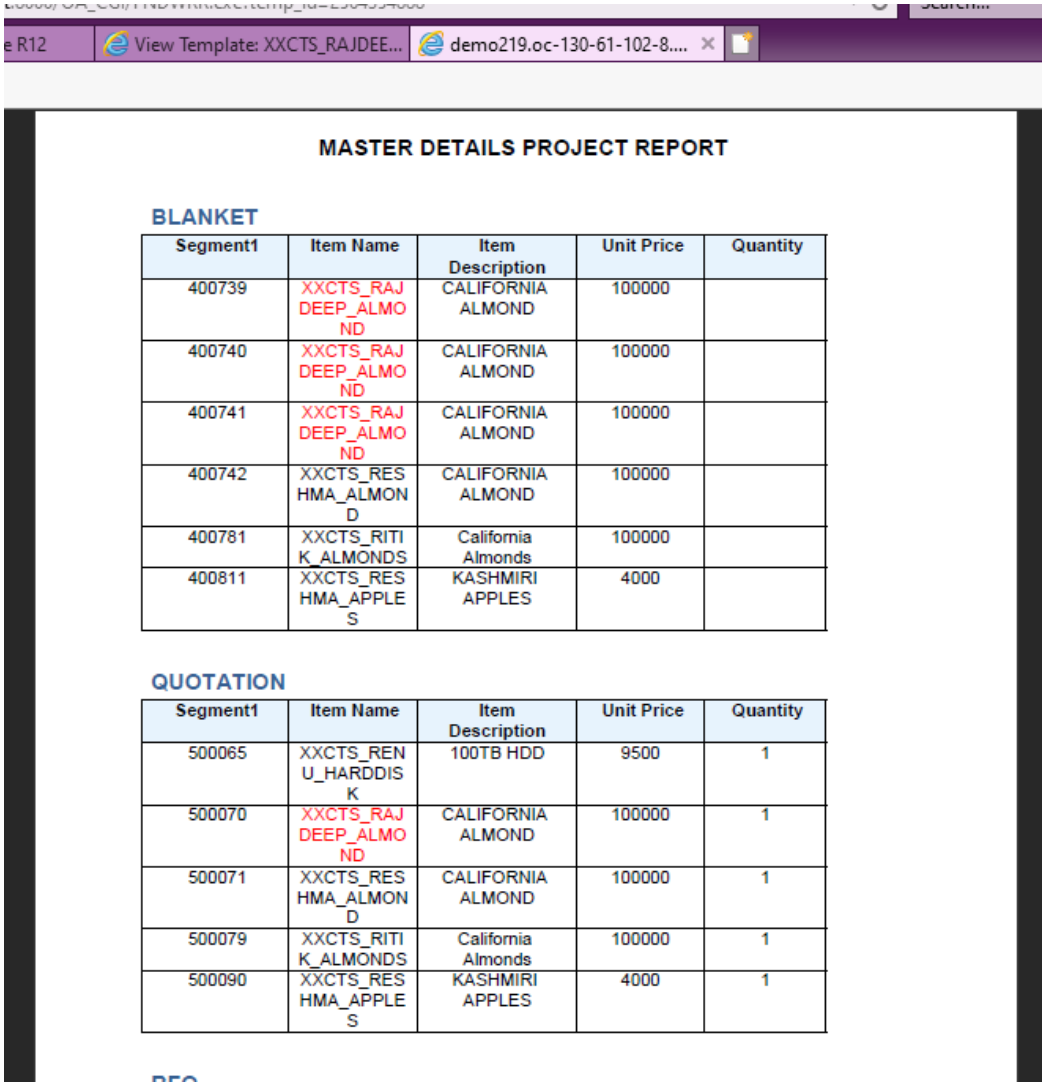
Step 30: Click to the View option in the menu bar and then click on Requests option there in the dropdown.

Step 31: Click on the Submit a New Request. Then I have mentioned the Concurrent Name of the Procedure, i.e. Procedure: Display Project MD Program : Rajdeep.

Step 32: Clicked on Submit button and then clicked on No. The procedure got compiled with Normal Status. Clicked on “View Output”.



Finally I have received the .rtf in .pdf format on the Internet Explorer successfully.


e R12
View Template: XXCTS_RAJDEE...
demo219.oc-130-61-102-8....
MASTER DETAILS PROJECT REPORT
BLANKET
Segment1 | Item Name | Item Description | Unit Price | Quantity
400739 | XXCTS_RAJDEEP_ALMOND | CALIFORNIA ALMOND | 100000 |
400740 | XXCTS_RAJDEEP_ALMOND | CALIFORNIA ALMOND | 100000 |
400741 | XXCTS_RAJDEEP_ALMOND | CALIFORNIA ALMOND | 100000 |
400742 | XXCTS_RESHMA_ALMOND | CALIFORNIA ALMOND | 100000 |
400781 | XXCTS_RITIK_ALMONDS | California Almonds | 100000 |
400811 | XXCTS_RESHMA_APPLES | KASHMIRI APPLES | 4000 |
QUOTATION
Segment1 | Item Name | Item Description | Unit Price | Quantity
500065 | XXCTS_RENU_HARDDISK | 100TB HDD | 9500 | 1
500070 | XXCTS_RAJDEEP_ALMOND | CALIFORNIA ALMOND | 100000 | 1
500071 | XXCTS_RESHMA_ALMOND | CALIFORNIA ALMOND | 100000 | 1
500079 | XXCTS_RITIK_ALMONDS | California Almonds | 100000 | 1
500090 | XXCTS_RESHMA_APPLES | KASHMIRI APPLES | 4000 | 1

Q2. Create a table XXCTS_Items with columns as item_id and item_name. Populate the table with 5 rows and demonstrate Outbound Interface.

Steps:

Step 1: Creating the table

At first I have created the table XXCTS_RAJDEEP_PRJ_ITEM, taking two columns ITEM_ID and ITEM_NAME.

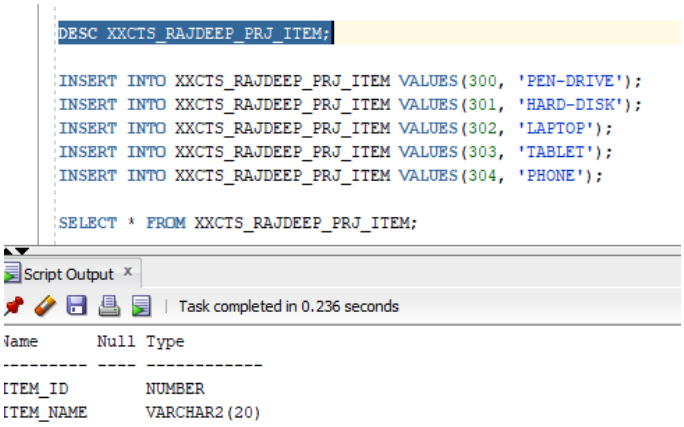
CREATE TABLE XXCTS_RAJDEEP_PRJ_ITEM

(ITEM_ID NUMBER,

ITEM_NAME VARCHAR2(20));

-----The table is created successfully

DESC XXCTS_RAJDEEP_PRJ_ITEM;



------ *Now inserting data into XXCTS_RAJDEEP_PRJ_ITEM*

Step 2: Inserting data into the table.

INSERT INTO XXCTS_RAJDEEP_PRJ_ITEM VALUES(300, 'PEN-DRIVE');

INSERT INTO XXCTS_RAJDEEP_PRJ_ITEM VALUES(301, 'HARD-DISK');

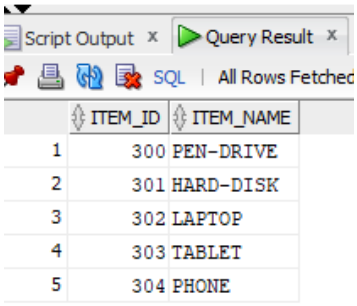
INSERT INTO XXCTS_RAJDEEP_PRJ_ITEM VALUES(302, 'LAPTOP');

INSERT INTO XXCTS_RAJDEEP_PRJ_ITEM VALUES(303, 'TABLET');

INSERT INTO XXCTS_RAJDEEP_PRJ_ITEM VALUES(304, 'PHONE');

---- All rows are inserted with specified data.

*SELECT * FROM XXCTS_RAJDEEP_PRJ_ITEM;*


Script Output | Query Result | SQL | All Rows Fetched

	ITEM_ID	ITEM_NAME
1	300	PEN-DRIVE
2	301	HARD-DISK
3	302	LAPTOP
4	303	TABLET
5	304	PHONE

Step 3: 'Write' Procedure

----- Now creating 'WRITE'

CREATE OR REPLACE DIRECTORY RD_PRJ_RAJDEEP AS '/usr/tmp';

GRANT WRITE ON DIRECTORY RD_PRJ_RAJDEEP to public;

----- Creating directory and granting write access to RD_PRJ_RAJDEEP

CREATE OR REPLACE PROCEDURE XXCTS_RD_PRJ_OUTBOUND_WRITE

IS

CURSOR ITEM_RAJ_CUR

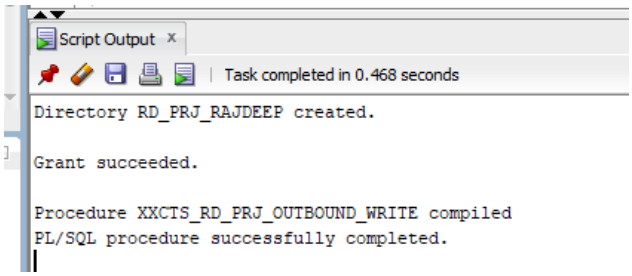
IS

SELECT ITEM_ID, ITEM_NAME FROM XXCTS_RAJDEEP_PRJ_ITEM;

V_FILE UTL_FILE.FILE_TYPE; -- File Handle

V_DIR VARCHAR2(25);

```
 v_filename VARCHAR2(25);
BEGIN
 V_DIR     := 'RD_PRJ_RAJDEEP';
 V_FILENAME := 'ITEM_RAJDEEP_DATA.csv';
 V_FILE    := UTL_FILE.FOPEN(V_DIR,V_FILENAME,'w');
 UTL_FILE.PUT_LINE(V_FILE,'ITEM_ID ' || 'ITEM_NAME');
 FOR ITEM_RAJ_REC IN ITEM_RAJ_CUR
 LOOP
  UTL_FILE.PUT(V_FILE,ITEM_RAJ_REC.ITEM_ID);
  UTL_FILE.PUT_LINE(V_FILE, ' ' || ITEM_RAJ_REC.ITEM_NAME);
 END LOOP;
 UTL_FILE.FCLOSE(V_FILE);
END;
EXEC XXCTS_RD_PRJ_OUTBOUND_WRITE;
----- Executing the Write Procedure.
```



Step 3: 'Read' Procedure

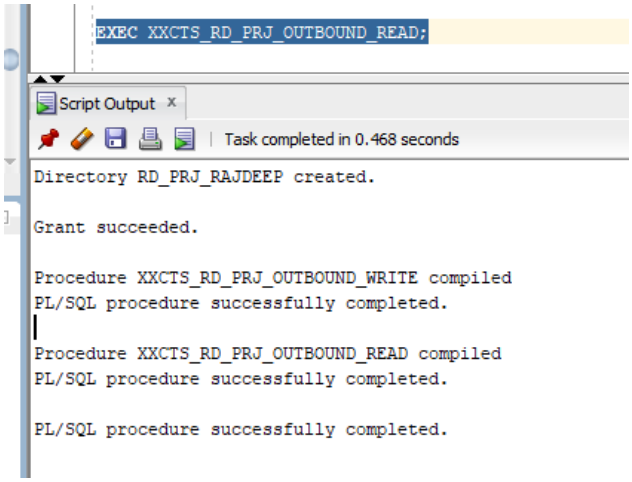
```
----- Now creating Read Procedure below
CREATE OR REPLACE PROCEDURE XXCTS_RD_PRJ_OUTBOUND_READ
```

```
IS
  V_FILE UTL_FILE.FILE_TYPE; -- File Handle
  V_DIR      VARCHAR2(25);
  V_FILENAME VARCHAR2(25);
  V_OUT      VARCHAR2(50);
BEGIN
  V_DIR      := 'RD_PRJ_RAJDEEP';
  V_FILENAME := 'ITEM_RAJDEEP_DATA.csv';
  V_FILE     := UTL_FILE.FOPEN(V_DIR,V_FILENAME,'r');
  LOOP
    BEGIN
      UTL_FILE.GET_LINE(V_FILE,V_OUT);
      DBMS_OUTPUT.PUT_LINE(V_OUT);
    EXCEPTION
    WHEN No_Data_Found THEN
      EXIT;
    END;
  END LOOP;
  UTL_FILE.FCLOSE(V_FILE);
END;
```

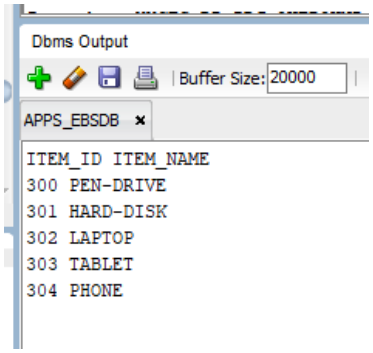
EXEC XXCTS_RD_PRJ_OUTBOUND_READ;

----- Executing the Read Procedure.

Outputs below:



In the DBMS Output window I have received this following output.



Outbound completed.

Q3. Interface for Vendor Details

Step 1: I have one base table i.e. AP_SUPPLIERS.

Step 2: I have taken two Interface Tables. One is AP_SUPPLIERS_INT and another is AP_SUPPLIER_SITES_INT.

Step 3: I have created two staging table based on the Interface tables.

1. **Staging table 1: MP_RAJDEEP_PRJ_INT1 TABLE**
   CREATE TABLE MP_RAJDEEP_PRJ_INT1

```
(
  VENDOR_INTERFACE_ID NUMBER,
  VENDOR_NAME        VARCHAR2(100),
  SEGMENT1           VARCHAR2(100),
  SUMMARY_FLAG       VARCHAR2(100),
  ENABLED_FLAG       VARCHAR2(100)
);
```

	COLUMN_NAME	DATA_TYPE
1	VENDOR_INTERFACE_ID	NUMBER
2	VENDOR_NAME	VARCHAR2(100 BYTE)
3	SEGMENT1	VARCHAR2(100 BYTE)
4	SUMMARY_FLAG	VARCHAR2(100 BYTE)
5	ENABLED_FLAG	VARCHAR2(100 BYTE)

2. **Staging table 2: MP_RAJDEEP_PRJ_INT2 TABLE**

```
CREATE TABLE MP_RAJDEEP_PRJ_INT2
(
  VENDOR_INTERFACE_ID NUMBER,
  VENDOR_SITE_CODE    NUMBER,
  VENDOR_ID           NUMBER,
  ZIP                 VARCHAR2(60),
  STATE               VARCHAR2(100),
  VENDOR_SITE_INTERFACE_ID NUMBER
);
```

	COLUMN_NAME	DATA_TYPE
1	VENDOR_INTERFACE_ID	NUMBER
2	VENDOR_SITE_CODE	NUMBER
3	VENDOR_ID	NUMBER
4	ZIP	VARCHAR2(60 BYTE)
5	STATE	VARCHAR2(100 BYTE)
6	VENDOR_SITE_INTERFACE_ID	NUMBER

Step 4: Here are my two .ctl and .dat file which I have created:

→.ctl files:

1. MP_RAJDEEP_PRJ_INT1.ctl

```
LOAD DATA
INFILE'D:\Cognizant\Oracle EBS\EBS Project\MP_RAJDEEP_PRJ_INT1'
INSERT INTO TABLE MP_RAJDEEP_PRJ_INT1
Fields terminated by ","
Optionally enclosed by ' "" '
(
VENDOR_INTERFACE_ID,
VENDOR_NAME,
SEGMENT1,
SUMMARY_FLAG,
ENABLED_FLAG
)
```

2. MP_RAJDEEP_PRJ_INT2.ctl

```
LOAD DATA
INFILE'D:\Cognizant\Oracle EBS\EBS Project\MP_RAJDEEP_PRJ_INT2'
INSERT INTO TABLE MP_RAJDEEP_PRJ_INT2
Fields terminated by ","
Optionally enclosed by ' "" '
(
VENDOR_INTERFACE_ID,
VENDOR_SITE_CODE,
VENDOR_ID,
ZIP,
STATE,
VENDOR_SITE_INTERFACE_ID
)
```

→.dat files:

1. MP_RAJDEEP_PRJ_INT1.dat

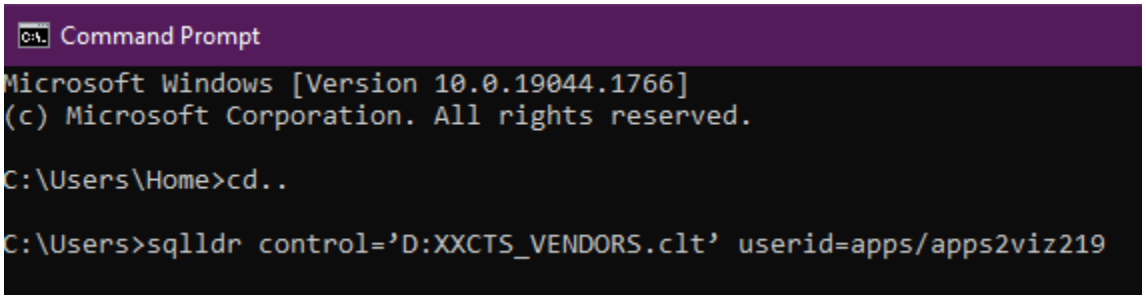
```
54,"RAJDEEP DAS","RD123","Y","Y"
55,"HARRY POTTER","RD456","Y","Y"
56,"JOSHEF MICHEAL","RD789","Y","Y"
57,"BRIAN HUMPRIES","RD111","Y","Y"
```

2. MP_RAJDEEP_PRJ_INT2.dat

```
100,500,700,"IN","WEST BENGAL",3001
101,501,701,"IN","BIHAR",3002
102,502,702,"IN","JHARKHAND",3003
```

Step 5: Now open the command prompt and run the following:

sqlldr control='D:XXCTS_VENDORS.clt' userid=apps/apps2viz219



The data will be inserted from .dat files to their corresponding staging tables via .ctl files respectively.

Note: In case of sqlldr not working, forcefully insert the data into their staging tables respectively.

MP_RAJDEEP_PRJ_INT1 table data inserted

	VENDOR_INTERFACE_ID	VENDOR_NAME	SEGMENT1	SUMMARY_FLAG	ENABLED_FLAG
1	54	RAJDEEP DAS	RD123	Y	Y
2	55	HARRY POTTER	RD456	Y	Y
3	56	JOSHEF MICHEAL	RD789	Y	Y
4	57	BRIAN HUMPRIES	RD111	Y	Y

MP_RAJDEEP_PRJ_INT2 table data also inserted.

	VENDOR_INTERFACE_ID	VENDOR_SITE_CODE	VENDOR_ID	ZIP	STATE	VENDOR_SITE_INTERFACE_ID
1	100	500	700	IN	WEST BENGAL	3001
2	101	501	701	IN	BIHAR	3002
3	102	502	702	IN	JHARKHAND	3003
4	103	503	703	IN	UP	3004

Step 7: Now add a column Process Flag for validation in each of the staging table.

ALTER TABLE MP_RAJDEEP_PRJ_INT1 ADD PROCESS_FLAG VARCHAR2(1);

	COLUMN_NAME	DATA_TYPE
1	VENDOR_INTERFACE_ID	NUMBER
2	VENDOR_NAME	VARCHAR2(100 BYTE)
3	SEGMENT1	VARCHAR2(100 BYTE)
4	SUMMARY_FLAG	VARCHAR2(100 BYTE)
5	ENABLED_FLAG	VARCHAR2(100 BYTE)
6	PROCESS_FLAG	VARCHAR2(1 BYTE)

Same will applied to MP_RAJDEEP_PRJ_INT2

ALTER TABLE MP_RAJDEEP_PRJ_INT2 ADD PROCESS_FLAG VARCHAR2(1);

	COLUMN_NAME	DATA_TYPE
1	VENDOR_INTERFACE_ID	NUMBER
2	VENDOR_SITE_CODE	NUMBER
3	VENDOR_ID	NUMBER
4	ZIP	VARCHAR2(60 BYTE)
5	STATE	VARCHAR2(100 BYTE)
6	VENDOR_SITE_INTERFACE_ID	NUMBER
7	PROCESS_FLAG	VARCHAR2(1 BYTE)

Step 6: Now creating package:

```
CREATE OR REPLACE PACKAGE XXCTS_RAJDEEP_PRJ_INTERFACE
IS
  PROCEDURE VALIDATION_RAJDEEP_V1;
  PROCEDURE VALIDATION_RAJDEEP_V2;
  PROCEDURE INSERT_RAJDEEP_PROC;
END XXCTS_RAJDEEP_PRJ_INTERFACE;
```

```
CREATE OR REPLACE PACKAGE BODY XXCTS_RAJDEEP_PRJ_INTERFACE
IS
```

```
PROCEDURE VALIDATION_RAJDEEP_V1
IS
 CURSOR ITEM_CUR
 IS
  SELECT * FROM MP_RAJDEEP_PRJ_INT1;
 V_NAME_COUNT NUMBER;
 V_SEG_COUNT  NUMBER;
 V_TYPE PO_LOOKUP_CODES_OLD.LOOKUP_CODE%TYPE;
 V_ERROR_FLAG BOOLEAN:=FALSE;
 ----------------------------------------------------------------------
BEGIN
 FOR ITEM_REC IN ITEM_CUR
 LOOP
  BEGIN
   SELECT COUNT(VENDOR_NAME)
   INTO V_NAME_COUNT
   FROM AP_SUPPLIERS
   WHERE VENDOR_NAME=ITEM_REC.VENDOR_NAME;
   IF V_NAME_COUNT  >0 THEN
    FND_FILE.PUT_LINE(FND_FILE.LOG,'DUPLICATE NAME EXISTS');
    DBMS_OUTPUT.PUT_LINE('DUPLICATE NAME EXISTS');
    V_ERROR_FLAG:=TRUE;
```

```
  END IF;
END;
--------------------------------------------------------------------------
BEGIN
  SELECT COUNT(SEGMENT1)
  INTO V_SEG_COUNT
  FROM AP_SUPPLIERS
  WHERE SEGMENT1=ITEM_REC.SEGMENT1;
  IF V_SEG_COUNT>0 THEN
    FND_FILE.PUT_LINE(FND_FILE.LOG,'DUPLICATE SEGMENT EXISTS');
    DBMS_OUTPUT.PUT_LINE('DUPLICATE SEGMENT HAS BEEN FOUND');
    V_ERROR_FLAG:=TRUE;
  END IF;
END;
-----------------------------------------------------------------------------
IF V_ERROR_FLAG=TRUE THEN
  UPDATE MP_RAJDEEP_PRJ_INT1
  SET PROCESS_FLAG='N'
  WHERE SEGMENT1  = ITEM_REC.SEGMENT1;
ELSE
  UPDATE MP_RAJDEEP_PRJ_INT1
  SET PROCESS_FLAG='Y'
```

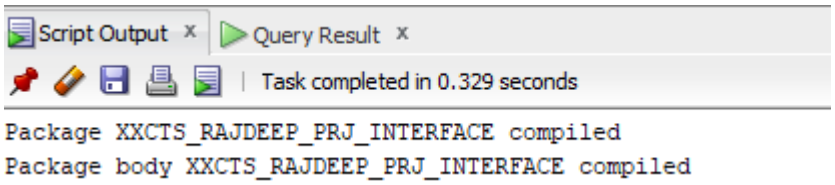
```
    WHERE SEGMENT1  = ITEM_REC.SEGMENT1;
   END IF;
  END LOOP;
END VALIDATION_RAJDEEP_V1;
PROCEDURE VALIDATION_RAJDEEP_V2
IS
  CURSOR VEN_CUR2
  IS
   SELECT * FROM MP_RAJDEEP_PRJ_INT2;
  V_CODE FND_TERRITORIES_TL.TERRITORY_CODE%TYPE;
  V_ERROR_FLAG BOOLEAN:=FALSE;
BEGIN
  FOR VEN_REC2 IN VEN_CUR2
  LOOP
   BEGIN
    SELECT TERRITORY_CODE
    INTO V_CODE
    FROM FND_TERRITORIES_TL
    WHERE TERRITORY_CODE=VEN_REC2.ZIP
    AND LANGUAGE       ='US';
    IF V_CODE         IS NULL THEN
      FND_FILE.PUT_LINE(FND_FILE.LOG,'ZIP CODE NOT FOUND');
```

```
      V_ERROR_FLAG:=TRUE;
    END IF;
  END;
  IF V_ERROR_FLAG=TRUE THEN
    UPDATE MP_RAJDEEP_PRJ_INT2
    SET PROCESS_FLAG        ='N'
    WHERE VENDOR_INTERFACE_ID = VEN_REC2.VENDOR_INTERFACE_ID;
  ELSE
    UPDATE MP_RAJDEEP_PRJ_INT2
    SET PROCESS_FLAG        ='Y'
    WHERE VENDOR_INTERFACE_ID = VEN_REC2.VENDOR_INTERFACE_ID;
  END IF;
 END LOOP;
END VALIDATION_RAJDEEP_V2;
PROCEDURE INSERT_RAJDEEP_PROC
IS
 CURSOR INSERT_CUR1
 IS
  SELECT * FROM MP_RAJDEEP_PRJ_INT1;
 CURSOR INSERT_CUR2
 IS
  SELECT * FROM MP_RAJDEEP_PRJ_INT2;
```

```
BEGIN
 DELETE FROM AP_SUPPLIERS_INT;
 DELETE FROM AP_SUPPLIER_SITES_INT;
 COMMIT;
 FOR INSERT_REC1 IN INSERT_CUR1
 LOOP
  INSERT
  INTO AP_SUPPLIERS_INT
   (
    VENDOR_INTERFACE_ID,
    VENDOR_NAME,
    SUMMARY_FLAG,
    ENABLED_FLAG,
    SEGMENT1
   )
   VALUES
   (
    INSERT_REC1.VENDOR_INTERFACE_ID,
    INSERT_REC1.VENDOR_NAME,
    INSERT_REC1.SUMMARY_FLAG,
    INSERT_REC1.ENABLED_FLAG,
    INSERT_REC1.SEGMENT1
```

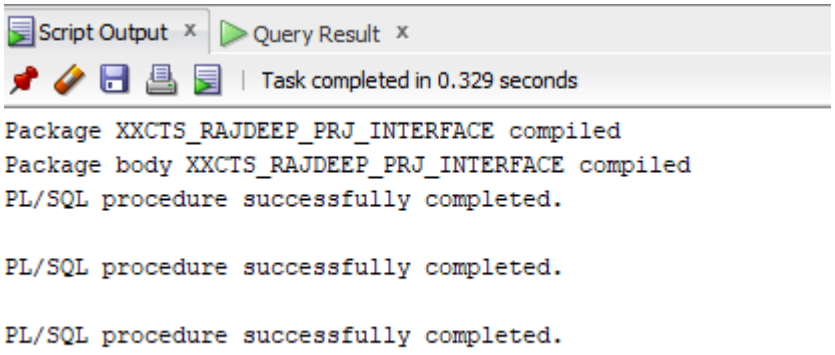
```
  );
 COMMIT;
END LOOP;
FOR INSERT_REC2 IN INSERT_CUR2
LOOP
 INSERT
 INTO AP_SUPPLIER_SITES_INT
  (
   VENDOR_INTERFACE_ID,
   VENDOR_SITE_CODE,
   VENDOR_ID,
   ZIP,
   STATE,
   VENDOR_SITE_INTERFACE_ID
  )
  VALUES
  (
   INSERT_REC2.VENDOR_INTERFACE_ID,
   INSERT_REC2.VENDOR_SITE_CODE,
   INSERT_REC2.VENDOR_ID,
   INSERT_REC2.ZIP,
   INSERT_REC2.STATE,
```

```
        INSERT_REC2.VENDOR_SITE_INTERFACE_ID
      );
    COMMIT;
  END LOOP;
 END INSERT_RAJDEEP_PROC;
END XXCTS_RAJDEEP_PRJ_INTERFACE;
```



Step 8: Executing Validation and Insert Procedure.

```
exec XXCTS_RAJDEEP_PRJ_INTERFACE.VALIDATION_RAJDEEP_V1;

exec XXCTS_RAJDEEP_PRJ_INTERFACE.VALIDATION_RAJDEEP_V2;

exec XXCTS_RAJDEEP_PRJ_INTERFACE.INSERT_RAJDEEP_PROC;
```



Step 9: Checking if the Validation is proceed successfully or not

For Staging table 1:

```
SELECT * FROM MP_RAJDEEP_PRJ_INT1;
```

```
SELECT * FROM MP_RAJDEEP_PRJ_INT1;
```

Script Output | Query Result

SQL | All Rows Fetched: 4 in 0.215 seconds

	VENDOR_INTERFACE_ID	VENDOR_NAME	SEGMENT1	SUMMARY_FLAG	ENABLED_FLAG	PROCESS_FLAG
1	54	RAJDEEP DAS	RD123	Y	Y	Y
2	55	HARRY POTTER	RD456	Y	Y	Y
3	56	JOSHEF MICHEAL	RD789	Y	Y	Y
4	57	BRIAN HUMPRIES	RD111	Y	Y	Y

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	VENDOR_INTERFACE_ID	VENDOR_NAME	SEGMENT1	SUMMARY_FLAG	ENABLED_FLAG	PROCESS_FLAG
1	54	RAJDEEP DAS	RD123	Y	Y	Y
2	55	HARRY POTTER	RD456	Y	Y	Y
3	56	JOSHEF MICHEAL	RD789	Y	Y	Y
4	57	BRIAN HUMPRIES	RD111	Y	Y	Y

For Staging table 2:

SELECT * FROM MP_RAJDEEP_PRJ_INT2;

```
SELECT *
FROM MP_RAJDEEP_PRJ_INT2;
```

Query Result

SQL | All Rows Fetched: 4 in 0.234 seconds

	VENDOR_INTERFACE_ID	VENDOR_SITE_CODE	VENDOR_ID	ZIP	STATE	VENDOR_SITE_INTERFACE_ID	PROCESS_FLAG
1	100	500	700	IN	WEST BENGAL	3001	Y
2	101	501	701	IN	BIHAR	3002	Y
3	102	502	702	IN	JHARKHAND	3003	Y
4	103	503	703	IN	UP	3004	Y

	VENDOR_INTERFACE_ID	VENDOR_SITE_CODE	VENDOR_ID	ZIP	STATE	VENDOR_SITE_INTERFACE_ID	PROCESS_FLAG
1	100	500	700	IN	WEST BENGAL	3001	Y
2	101	501	701	IN	BIHAR	3002	Y
3	102	502	702	IN	JHARKHAND	3003	Y
4	103	503	703	IN	UP	3004	Y

Step 10: Checking the Interface table if the data has been inserted or not:

Checking Interface Table 1 is AP_SUPPLIERS_INT,

SELECT * FROM AP_SUPPLIERS_INT;

```
SELECT * FROM AP_SUPPLIERS_INT;
SELECT * FROM AP_SUPPLIER_SITES_INT;

--STAGING TABLE 1
```

Script Output | Query Result | SQL | All Rows Fetched: 4 in 0.343 seconds

	VENDOR_INTERFACE_ID	LAST_UPDATE_DATE	LAST_UPDATED_BY	VENDOR_NAME	VENDOR_NAME_ALT	SEGMENT1	SUMMARY_FLAG	ENABLED_FLAG	LAST_UPDATE_LOGIN	CREATION_DA
1	54	(null)	(null)	RAJDEEP DAS	(null)	RD123	Y	Y	(null)	(null)
2	55	(null)	(null)	HARRY POTTER	(null)	RD456	Y	Y	(null)	(null)
3	56	(null)	(null)	JOSHEF MICHEAL	(null)	RD789	Y	Y	(null)	(null)
4	57	(null)	(null)	BRIAN HUMPRIES	(null)	RD111	Y	Y	(null)	(null)

So the data has been inserted to interface table 1.

Checking Interface Table 2 is AP_SUPPLIERS_SITES_INT:

SELECT * FROM AP_SUPPLIER_SITES_INT;

```
SELECT * FROM AP_SUPPLIER_SITES_INT;

--STAGING TABLE 1
```

Script Output | Query Result | SQL | All Rows Fetched: 4 in 0.316 seconds

	VENDOR_INTERFACE_ID	LAST_UPDATE_DATE	LAST_UPDATED_BY	VENDOR_ID	VENDOR_SITE_CODE	VENDOR_SITE_CODE_ALT	LAST_UPDATE_LOGIN	CREATION_DATE	CREATED_BY	PURC
1	100	(null)	(null)	700	500	(null)	(null)	(null)	(null)	(null)
2	101	(null)	(null)	701	501	(null)	(null)	(null)	(null)	(null)
3	102	(null)	(null)	702	502	(null)	(null)	(null)	(null)	(null)
4	103	(null)	(null)	703	503	(null)	(null)	(null)	(null)	(null)

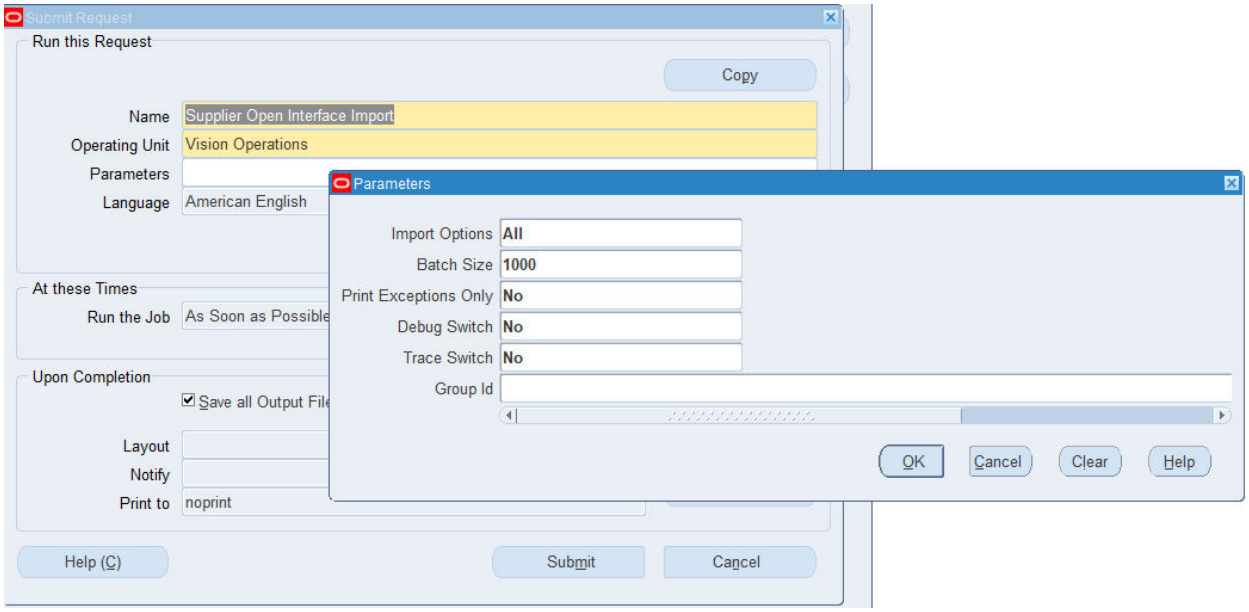
So the data has been inserted to interface table 2.

Step 11: Now open Oracle EBS Application. Switch to Payables responsibility and go to View → Requests.

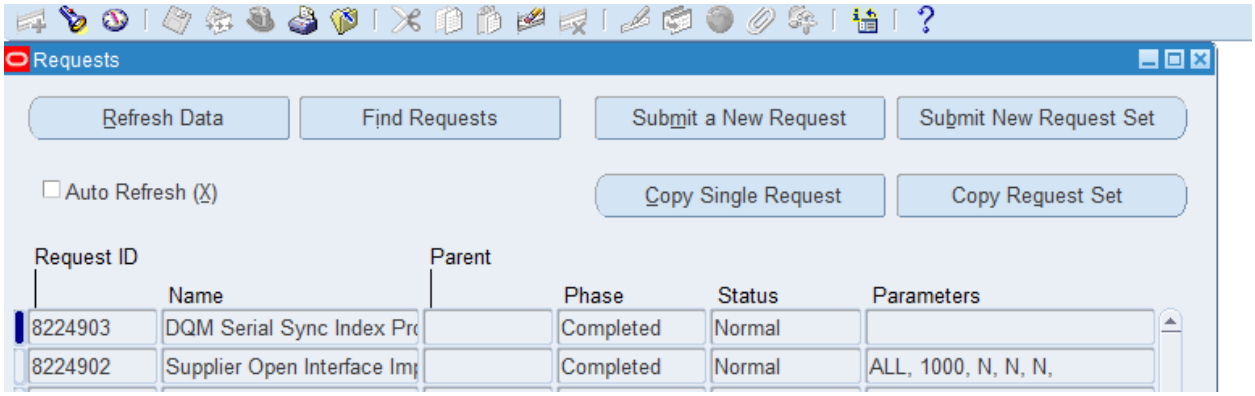
We have three Standard Programs in total for our base table i.e. AP_SUPPLIERS.

1. Supplier Open Interface Import.
2. Supplier Sites Open Interface Import
3. Supplier Site Contacts Open Interface Import.

I have taken Supplier Open Interface Import to run request.



Click OK. Click Submit. Click No. Click Find.

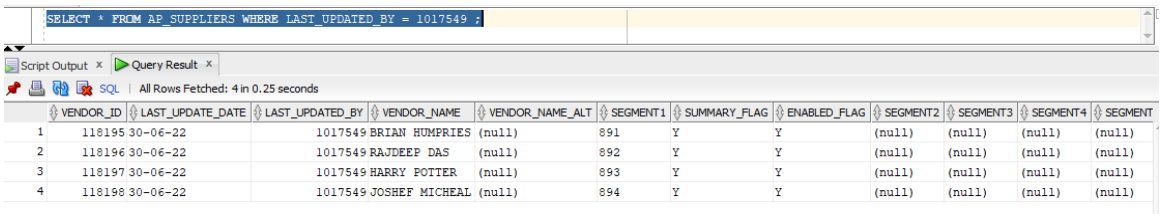


The Supplier Open Interface Import program successfully completed request.

Step 12: In SQL Developer, checking if the data is inserted into the base table or not.

SELECT * FROM AP_SUPPLIERS WHERE LAST_UPDATED_BY = 1017549 ;

--1017549 is my user-id in the EBS.



Data is successfully inserted into to the base table.

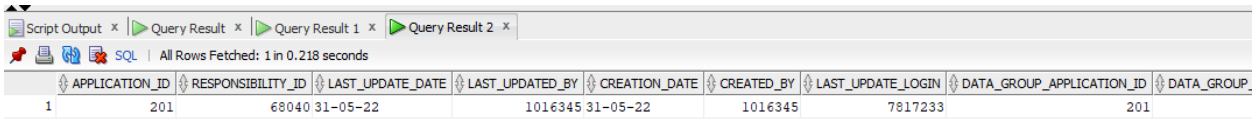
Q4. API to create Responsibility

Steps:

Steps 1: I have used FND_RESPONSIBILITY table to get required details.

SELECT * FROM FND_RESPONSIBILITY

WHERE RESPONSIBILITY_KEY = 'XXCTS_RAJDEEP_RESP_MAY22';



-------Now I have created procedure XXCTS_RAJDEEP_PRJ_RESP_API

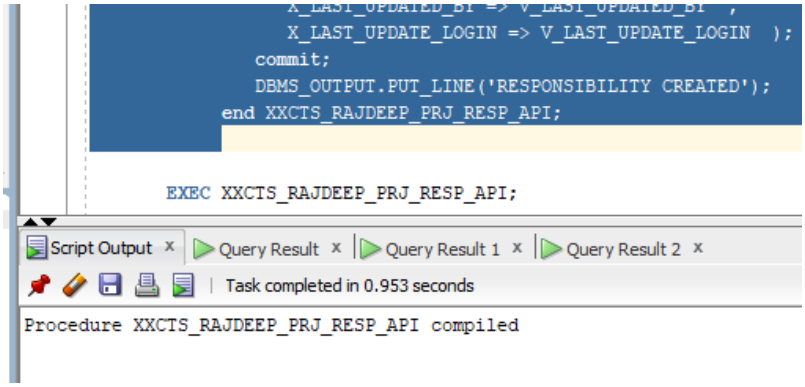
-------Taking *RESPONSIBILITY_KEY as 'XXCTS_RAJDEEP_MINI' and RESPONSIBILITY_NAME as 'RAJDEEP RESP MINI'*

```
CREATE OR REPLACE PROCEDURE XXCTS_RAJDEEP_PRJ_RESP_API
 IS
    V_ROWID                         VARCHAR2(200);
    V_RESPONSIBILITY_ID         NUMBER;
    V_APPLICATION_ID    NUMBER;
    V_WEB_HOST_NAME                 VARCHAR2(200);
    V_WEB_AGENT_NAME                        VARCHAR2(200);
    V_DATA_GROUP_APPLICATION_ID         NUMBER;
    V_DATA_GROUP_ID             NUMBER;
    V_MENU_ID     NUMBER;
```

```
V_START_DATE                    DATE;
V_END_DATE          DATE;
V_GROUP_APPLICATION_ID          NUMBER;
V_REQUEST_GROUP_ID              NUMBER;
V_VERSION           VARCHAR2(200);
V_RESPONSIBILITY_KEY            VARCHAR2(200);
V_RESPONSIBILITY_NAME VARCHAR2(200);
V_DESCRIPTION       VARCHAR2(200);
V_CREATION_DATE         DATE;
V_CREATED_BY                    NUMBER;
V_LAST_UPDATE_DATE              DATE;
V_LAST_UPDATED_BY       NUMBER;
V_LAST_UPDATE_LOGIN             NUMBER;

BEGIN
        V_ROWID                 := V_ROWID;
        SELECT FND_RESPONSIBILITY_S.NEXTVAL INTO V_RESPONSIBILITY_ID
        FROM DUAL;
        --V_RESPONSIBILITY_ID       := FND_RESPONSIBILITY_S.NEXTVAL;
        V_APPLICATION_ID        := 201;
        V_WEB_HOST_NAME           := null;
        V_WEB_AGENT_NAME          := null;
```

```
V_DATA_GROUP_APPLICATION_ID   := 201;
V_DATA_GROUP_ID            := 0;
V_MENU_ID                  := 68071;
V_START_DATE               := '27-06-22';
V_END_DATE                 := null;
V_GROUP_APPLICATION_ID     := 201;
V_REQUEST_GROUP_ID         := 112;
V_VERSION                  := 4;
V_RESPONSIBILITY_KEY       := 'XXCTS_RAJDEEP_MINI';
V_RESPONSIBILITY_NAME      := 'RAJDEEP RESP MINI';
V_DESCRIPTION              := 'Responsibility for API';
V_CREATION_DATE            := '27-06-22';
V_CREATED_BY               := 1016345;
V_LAST_UPDATE_DATE         := '27-06-22';
V_LAST_UPDATED_BY          := 1016345;
V_LAST_UPDATE_LOGIN        := 7817233;
--INVOKING THE API
--FNDGLOBAL.APPSINITIALISE
 FND_RESPONSIBILITY_PKG.INSERT_ROW(
X_ROWID         =>     V_ROWID    ,
X_RESPONSIBILITY_ID =>     V_RESPONSIBILITY_ID     ,
X_APPLICATION_ID =>V_APPLICATION_ID                ,
```

```
            X_WEB_HOST_NAME => V_WEB_HOST_NAME,
            X_WEB_AGENT_NAME =>   V_WEB_AGENT_NAME   ,
            X_DATA_GROUP_APPLICATION_ID =>
V_DATA_GROUP_APPLICATION_ID    ,
            X_DATA_GROUP_ID => V_DATA_GROUP_ID          ,
            X_MENU_ID => V_MENU_ID                      ,
            X_START_DATE => V_START_DATE                ,
            X_END_DATE   => V_END_DATE          ,
            X_GROUP_APPLICATION_ID => V_GROUP_APPLICATION_ID,
            X_REQUEST_GROUP_ID => V_REQUEST_GROUP_ID        ,
            X_VERSION => V_VERSION          ,
            X_RESPONSIBILITY_KEY => V_RESPONSIBILITY_KEY    ,
            X_RESPONSIBILITY_NAME => V_RESPONSIBILITY_NAME     ,
            X_DESCRIPTION => V_DESCRIPTION              ,
            X_CREATION_DATE => V_CREATION_DATE      ,
            X_CREATED_BY => V_CREATED_BY                ,
            X_LAST_UPDATE_DATE =>  V_LAST_UPDATE_DATE   ,
            X_LAST_UPDATED_BY => V_LAST_UPDATED_BY      ,
            X_LAST_UPDATE_LOGIN     => V_LAST_UPDATE_LOGIN      );
          commit;
          DBMS_OUTPUT.PUT_LINE('RESPONSIBILITY CREATED');
        end XXCTS_RAJDEEP_PRJ_RESP_API;
```



------Now executing XXCTS_RAJDEEP_PRJ_RESP_API

Executing…
EXEC XXCTS_RAJDEEP_PRJ_RESP_API;

Q5. Create a workflow to allow the user to work on Weekends between 9 am to 6 pm only.

Code:

--CREATING THE PROCEDURE WITHIN THE PACKAGE

CREATE OR REPLACE PACKAGE XXCTS_RAJDEEP_PRJ_WF-- PACKAGE NAME IS XXCTS_WORKFLOW_PACK_WEEKEND

IS

PROCEDURE CHECK_DAY_TIME (ITEMTYPE IN VARCHAR2,--PROCEDURE NAME IS CHECK_DAY_TIME

itemkey in varchar2,

actid in number,

FUNCMODE IN VARCHAR2,

RESULT IN OUT VARCHAR2);

END XXCTS_RAJDEEP_PRJ_WF;

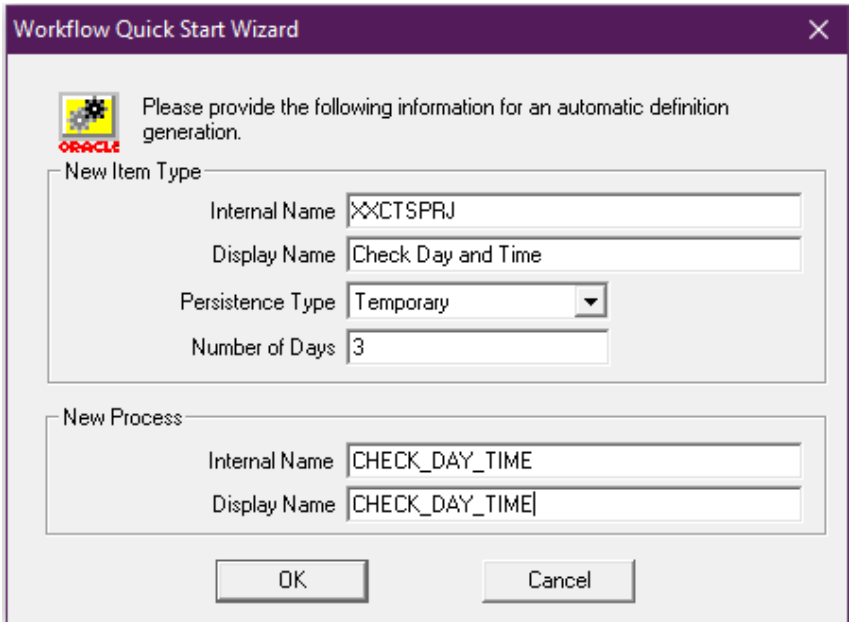
/

```
CREATE OR REPLACE PACKAGE BODY XXCTS_RAJDEEP_PRJ_WF
IS
  PROCEDURE CHECK_DAY_TIME(
     ITEMTYPE IN VARCHAR2,
     itemkey  IN VARCHAR2,
     actid    IN NUMBER,
     FUNCMODE IN VARCHAR2,
     RESULT   IN OUT VARCHAR2)
  IS
    V_DAY1  VARCHAR2(3);
    V_DAY2  VARCHAR2(3);
    V_TIME1 NUMBER;
    V_TIME2 NUMBER;
  BEGIN
    SELECT TO_CHAR(SYSDATE,'DY') INTO V_DAY1 FROM DUAL;
    SELECT TO_CHAR(SYSDATE,'HH24') INTO V_TIME1 FROM DUAL;
    WF_ENGINE.SETITEMATTRTEXT(ITEMTYPE=>ITEMTYPE, ITEMKEY =>ITEMKEY,
ANAME=>'DAY_OF_THE_WEEK',AVALUE=>V_DAY1 );
    V_DAY2 := WF_ENGINE.GETITEMATTRTEXT ( ITEMTYPE => ITEMTYPE, ITEMKEY
=> ITEMKEY,ANAME => 'DAY_OF_THE_WEEK');
    WF_ENGINE.SETITEMATTRTEXT(ITEMTYPE=>ITEMTYPE, ITEMKEY =>ITEMKEY,
ANAME=>'TIME_OF_THE_DAY',AVALUE=>V_TIME1 );
```

```
   V_TIME2 := WF_ENGINE.GETITEMATTRTEXT ( ITEMTYPE => ITEMTYPE, ITEMKEY
=> ITEMKEY,ANAME => 'TIME_OF_THE_DAY');

   IF V_DAY2 IN ('SAT','SUN') AND V_TIME2 BETWEEN 9 AND 18 THEN

     RESULT := 'COMPLETE:Y';

   ELSE

     RESULT := 'COMPLETE:N';

   END IF;

 END CHECK_DAY_TIME;

END XXCTS_RAJDEEP_PRJ_WF;

/
```

STEPS:

Step 1: Open Oracle Workflow Builder (Run as Administrator).
Step 2: Go to File in the menu bar→Quick Start Wizard.
Step 3: Fill the dialogue box as following and click on OK:

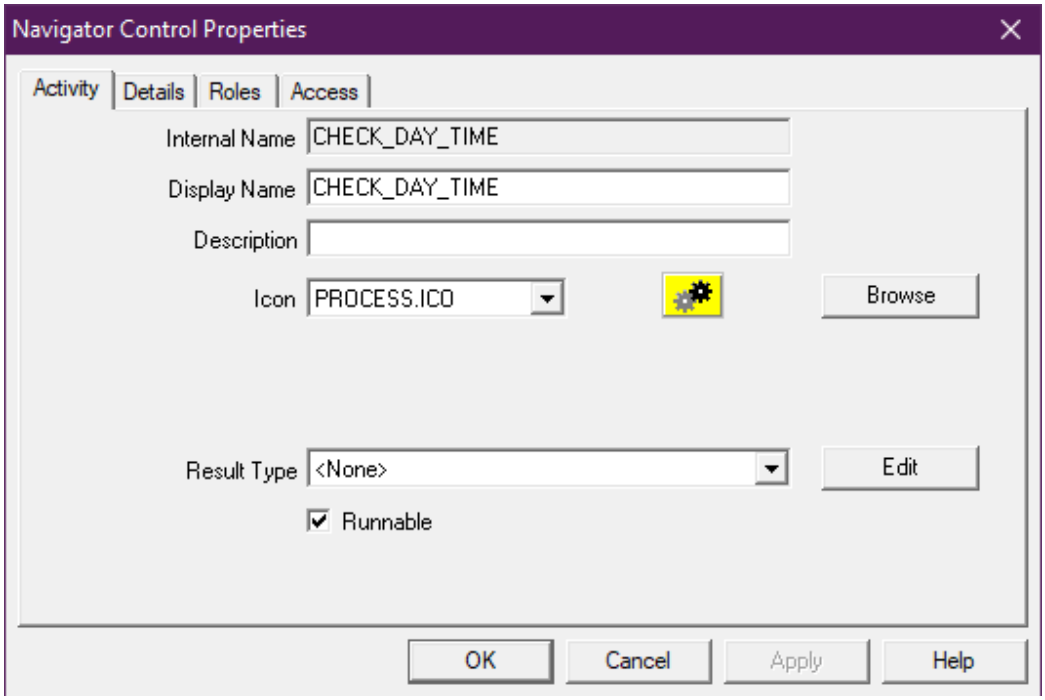


Here I have given the Internal Name as XXCTSPRJ under Item Type.

I have named the Internal Name of the Process as CHECK_DAY_TIME.

Number of Days will be 3 and the Persistence Type is Temporary.

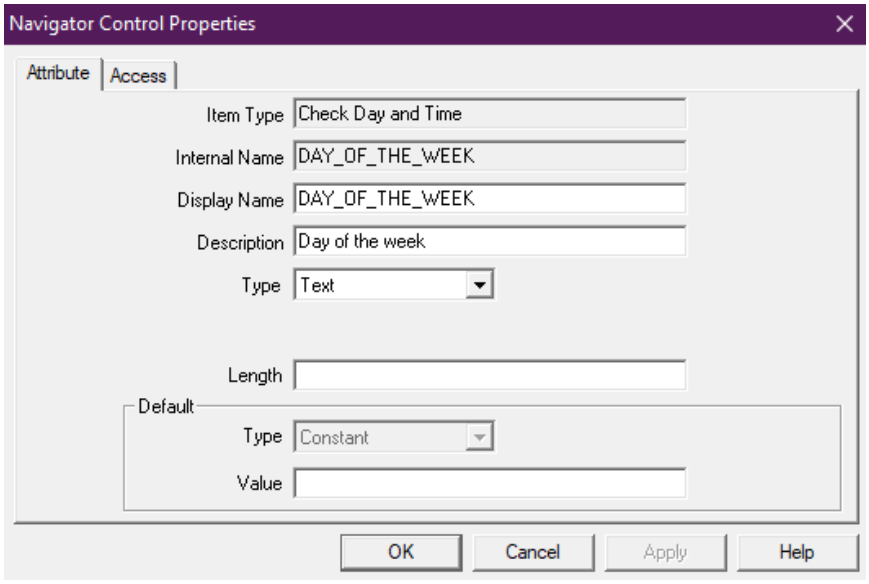
After creating the Process, the CHECK_DAY_TIME Process will be like this:



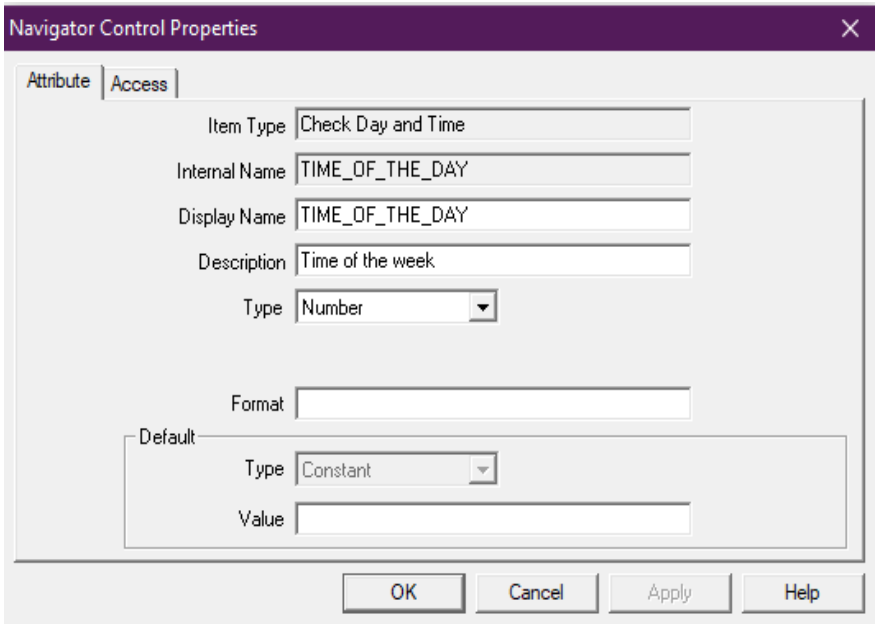
Step 4: After the workflow has been created. In the Navigator, right click on the Attribute → Click on the 'New Attribute'.
Here I have created two attributes.
One will be as DAY_OF_THE_WEEK and another will be TIME_OF_THE_DAY.
In the DAY_OF_THE_WEEK, I have taken Type as Text. In the TIME_OF_THE_DAY, I have taken the Type as number.

In the following images the first one is of DAY_OF_THE_WEEK.

The second image is of TIME_OF_THE_DAY.



Click on Apply, click on OK.

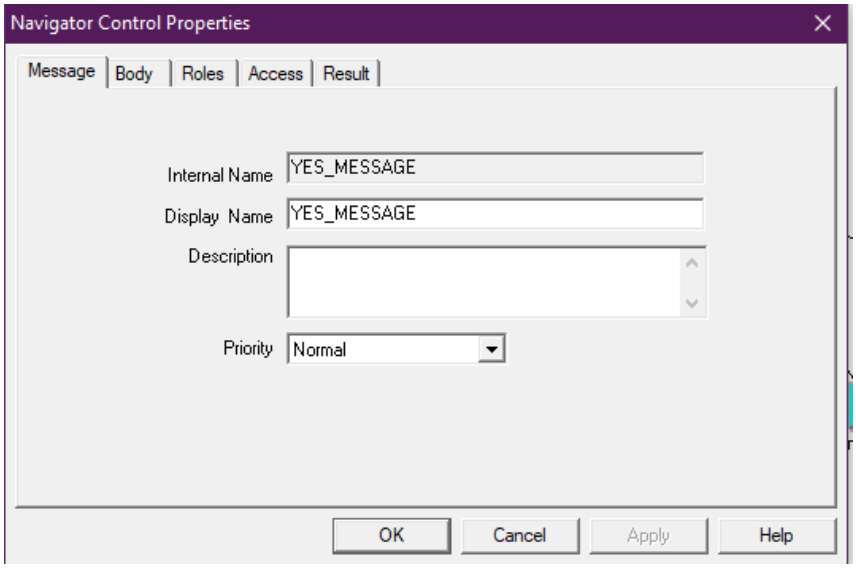


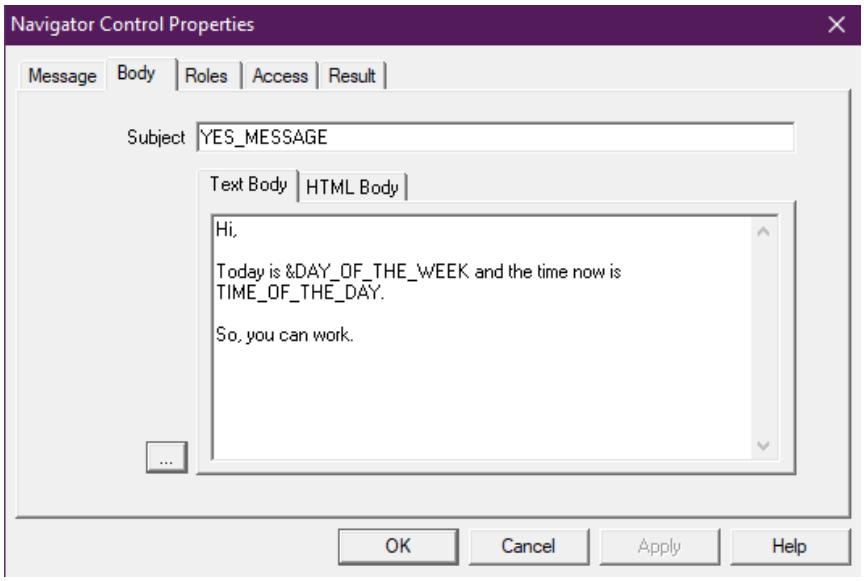
Click on Apply, click on OK.

Step 5: Now I have created two types of messages. One is for Yes and another is for No.

So in the navigation bar, just right click on the Messages→ Click on New Message. Under the Message tab in the dialogue box fill up as it is shown:

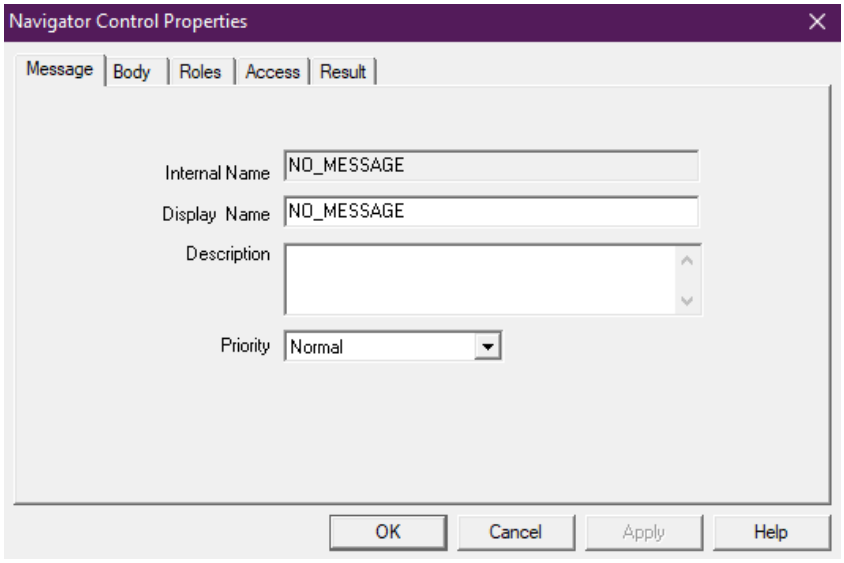
A. FOR YES MESSAGE AS FOLLOWS:

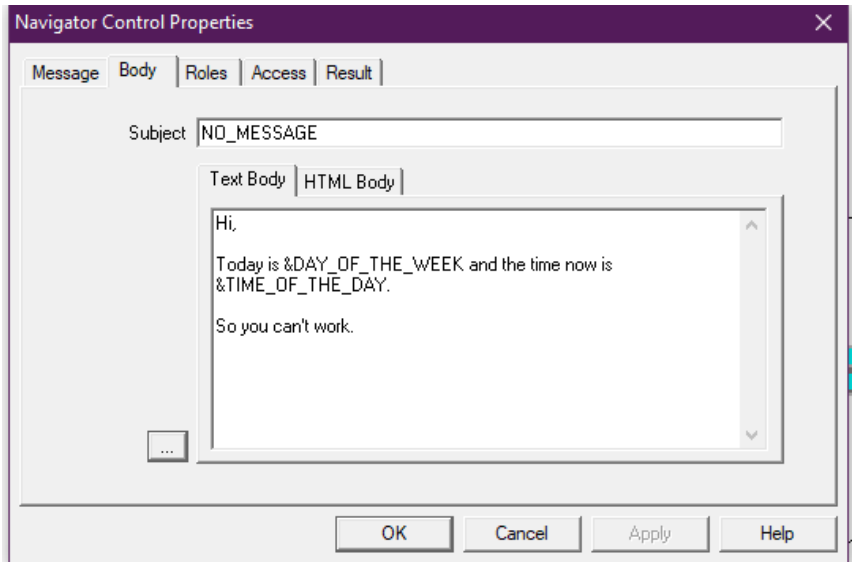




Click on Apply, Click OK.

B. FOR NO MESSAGE, AS FOLLOWS:



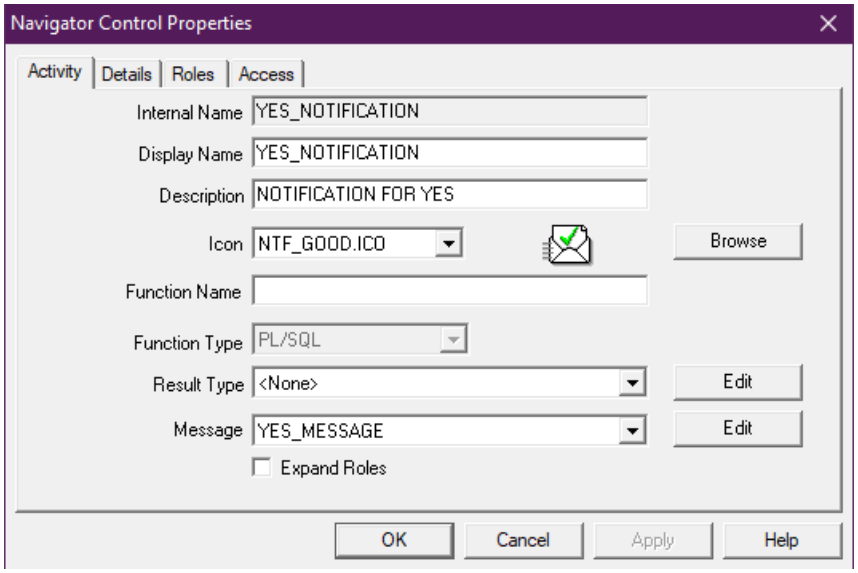


Click Apply, Click OK.

Step 6: Now I have created two types of Notifications. ONE is for YES, another is for NO.

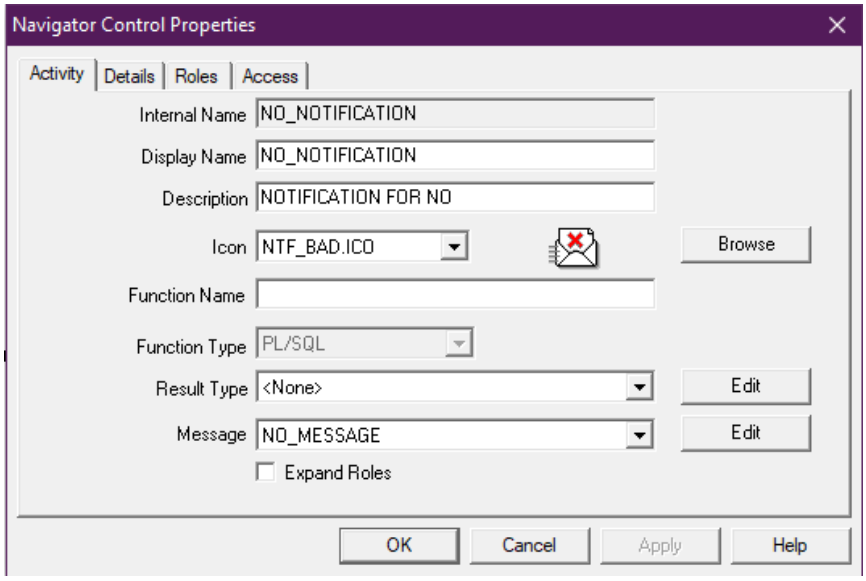
So in the navigation bar, Right click on Notifications →Click on New Notification. Fill the dialogue box as it is shown under the Activity tab as it is shown below:

A. For 'Yes' Notification:



Click on Apply, click on OK.

B. For 'No' Notification



Step 7: In the navigation bar, right Click on Functions→ Click on New Function. Fill the dialogue box as following:

Function name will be as: apps.(Package name).(Procedure name)

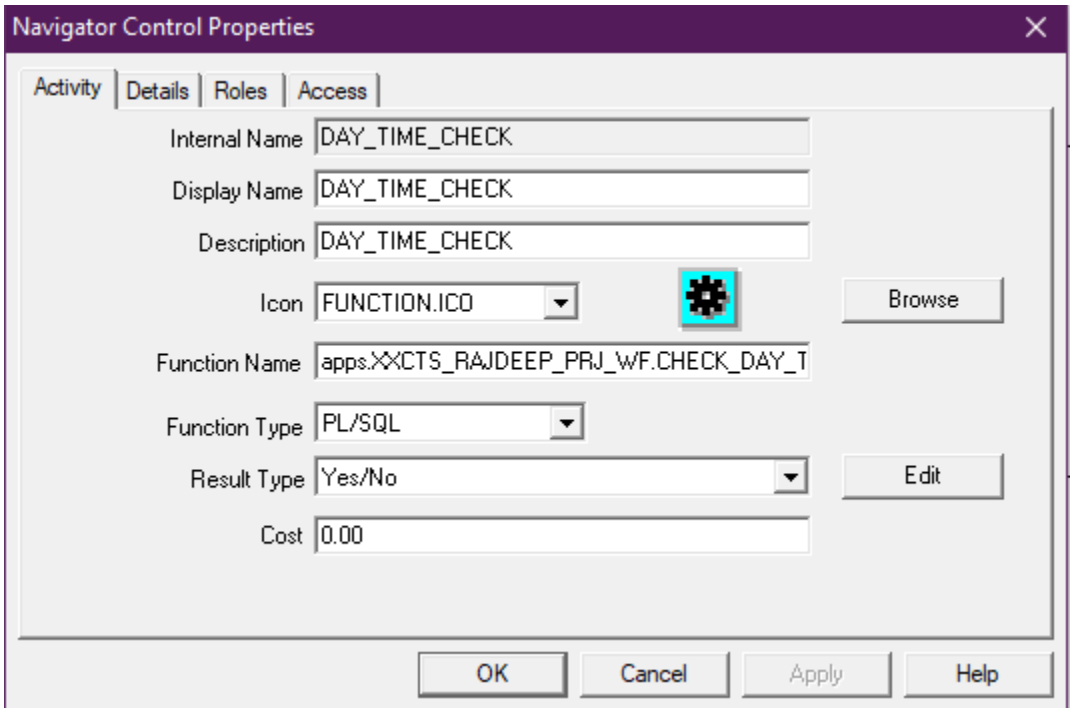
Here it is **apps.XXCTS_RAJDEEP_PRJ_WF.CHECK_DAY_TIME**.

Internal Name as DAY_TIME_CHECK.

Function Type: PL/SQL

Result Type: Yes/No

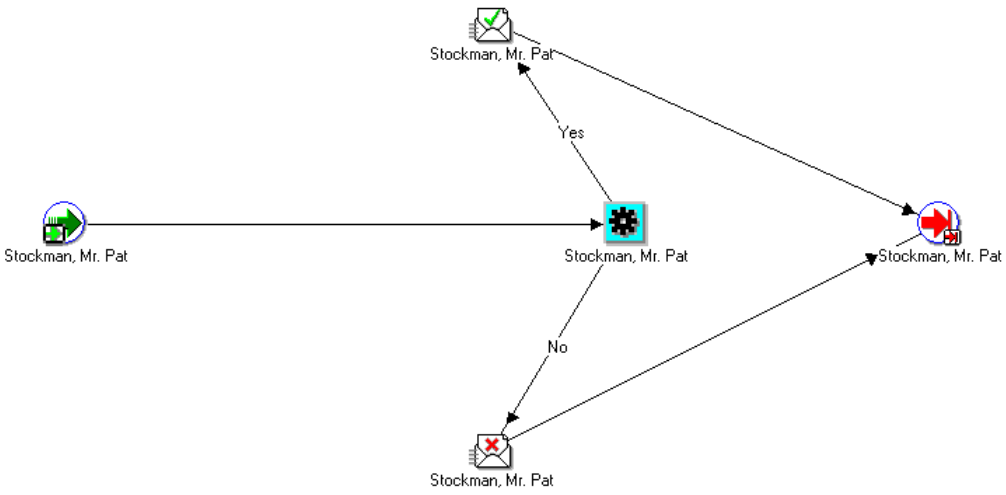
Click Apply. Click OK.



Step 8: Drag the DAY_TIME_CHECK function to the workflow diagram box between the START & END. Now drag the YES_NOTIFICATION & NO_NOTIFICATION in opposite direction to each other (Upside and Downside position of CHECKTIME respectively as shown in the next image)

Step 9: Now connect the nodes in the following order:

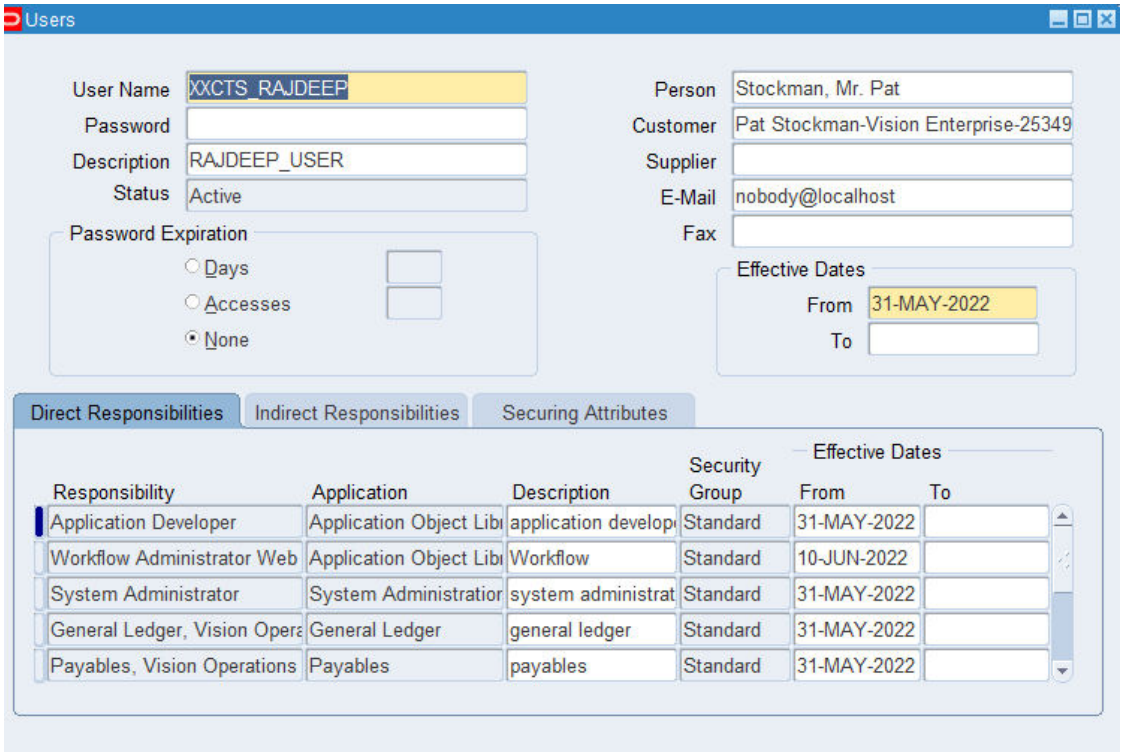
From START→ create a connection to DAY_TIME_CHECK function → From DAY_TIME_CHECK function, one connection goes to YES_NOTIFICATION above (It will ask to choose yes/no. Choose Yes). Another connection goes to the downward which is NO_NOTIFICATION (It will ask to choose yes/no. Choose No)→ Now both from the YES_NOTIFICATION & NO_NOTIFICATION, two separate connections will go to the END.



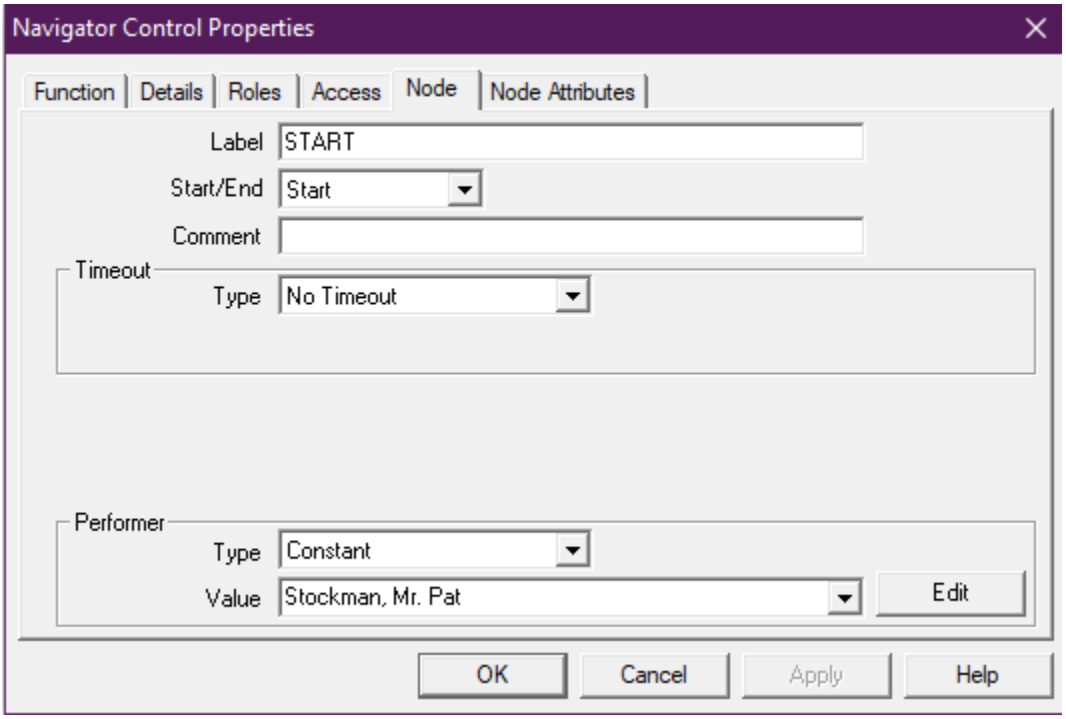
The connection will be as exactly like this above one.

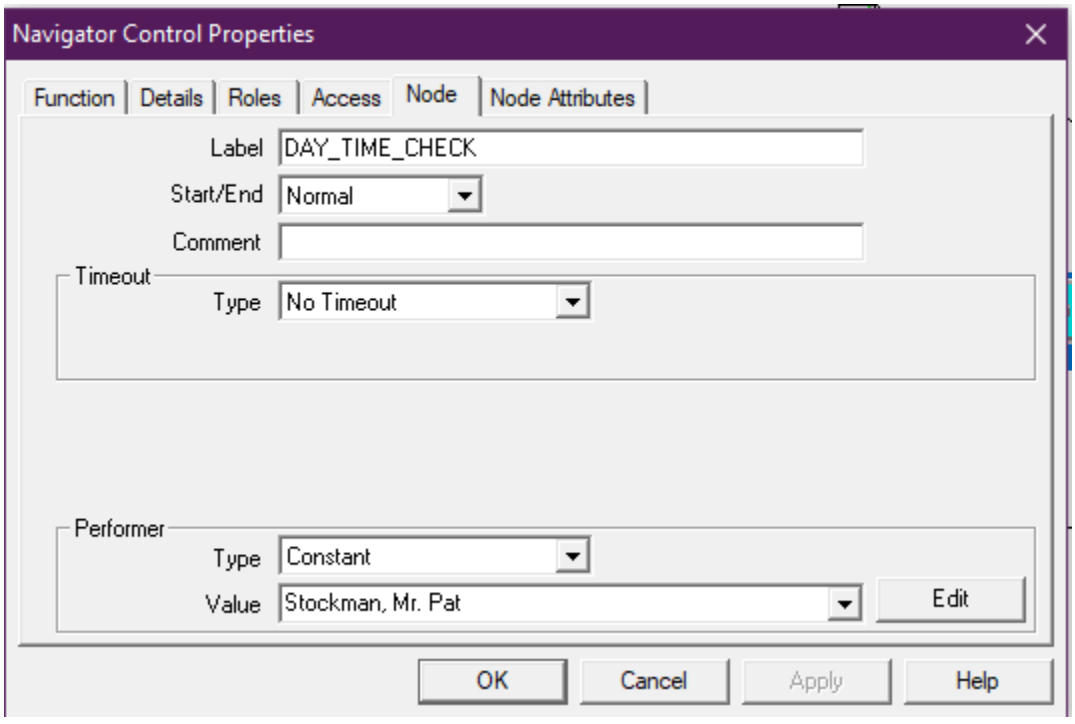
Step 10: Now open Oracle EBS Java Application. Go to System Administrator responsibility→ Security→ User → Define. Press 'fn+f11' to start query. Type User Name 'XXCTS_RAJDEEP%' and press 'ctrl + fn+ F11'. (This user is me).

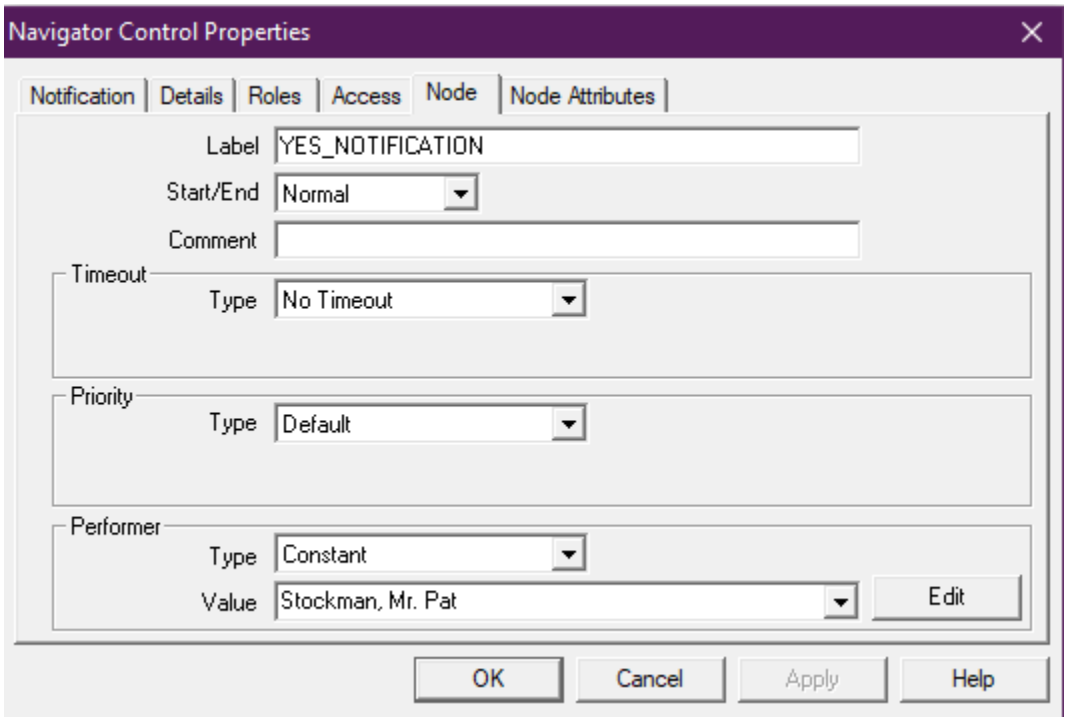
Step 11: Copy the Stockman, Mr. Pat in the Person input box. This name will be considered as the Performer's name.

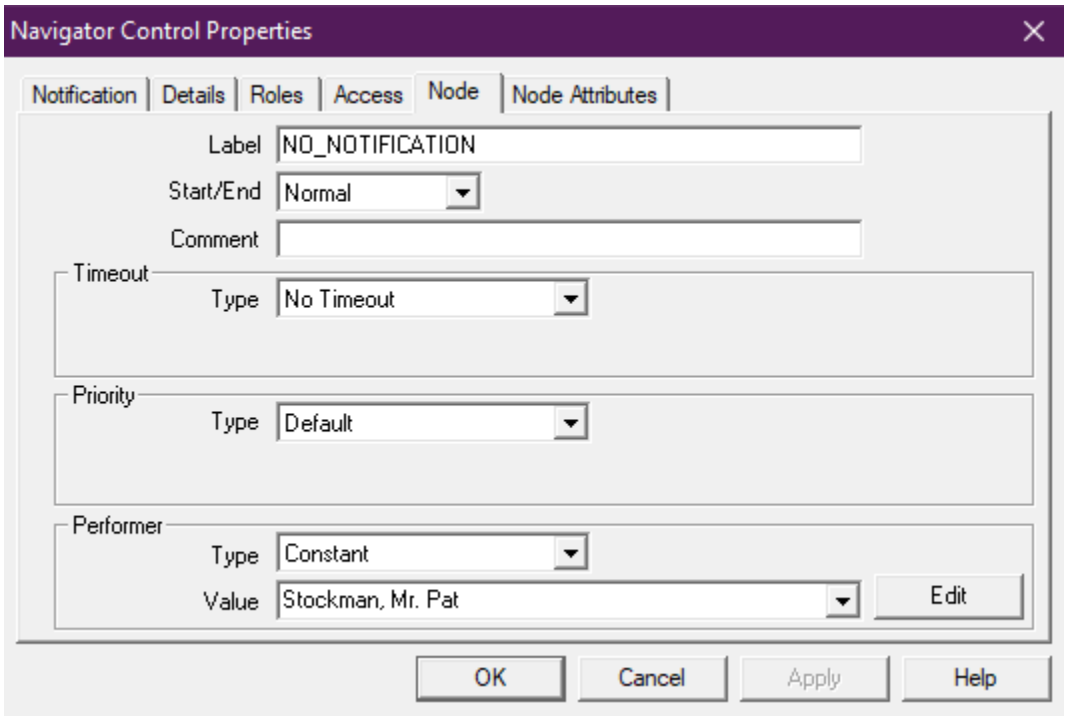


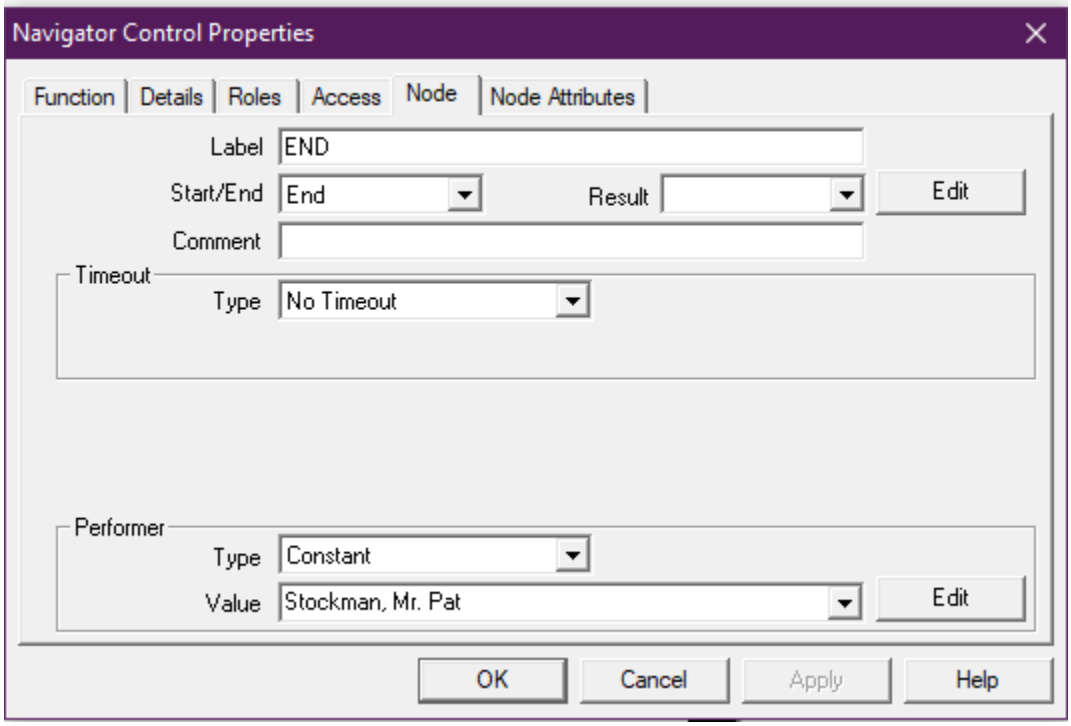
Step 12: Go back to Oracle Workflow Builder. Right click on each of the nodes in the workflow diagram i.e. START, DAY_TIME_CHECK, YES_NOTIFICATION, NO_NOTIFICATION, END and then click on each of these nodes' Properties option. In the coming dialogue box, go the Node tab, go to the Performer section below in the dialogue box and paste the name Stockman, Mr. Pat.



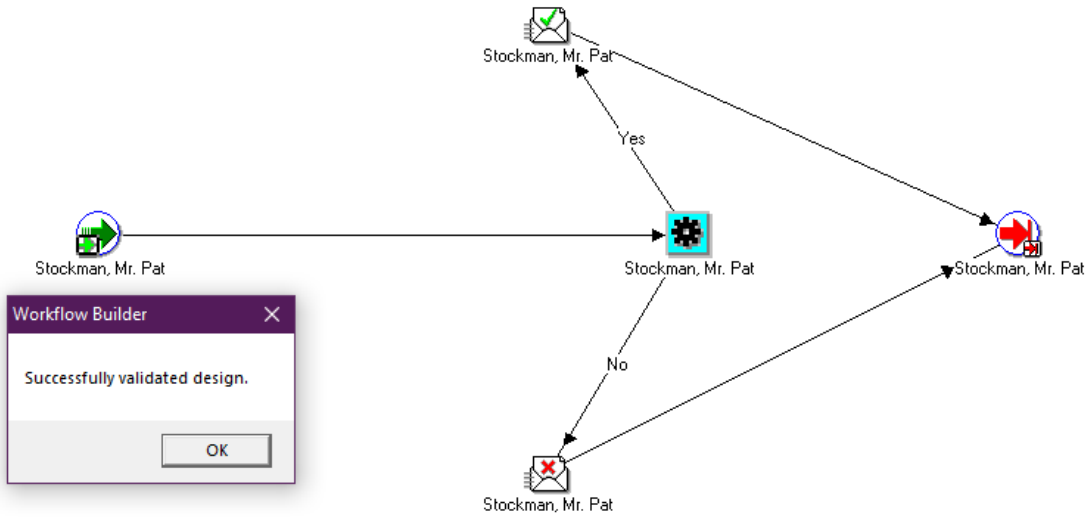



Navigator Control Properties
Notification
Details
Roles
Access
Node
Node Attributes
Label
YES_NOTIFICATION
Start/End
Normal
Comment
Timeout
Type
No Timeout
Priority
Type
Default
Performer
Type
Constant
Value
Stockman, Mr. Pat
Edit
OK
Cancel
Apply
Help


Navigator Control Properties
Notification
Details
Roles
Access
Node
Node Attributes
Label
NO_NOTIFICATION
Start/End
Normal
Comment
Timeout
Type
No Timeout
Priority
Type
Default
Performer
Type
Constant
Value
Stockman, Mr. Pat
Edit
OK
Cancel
Apply
Help

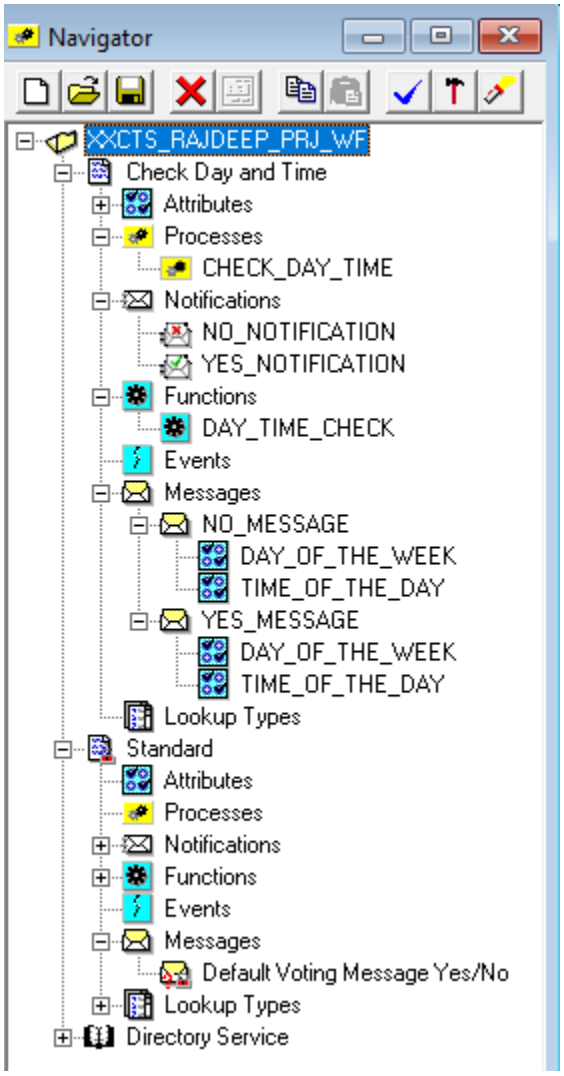


Step 13: Now I have validated the workflow design in the Navigator tool bar by clicking on the 'blue tick' sign. The 'Successfully validated design comes'.

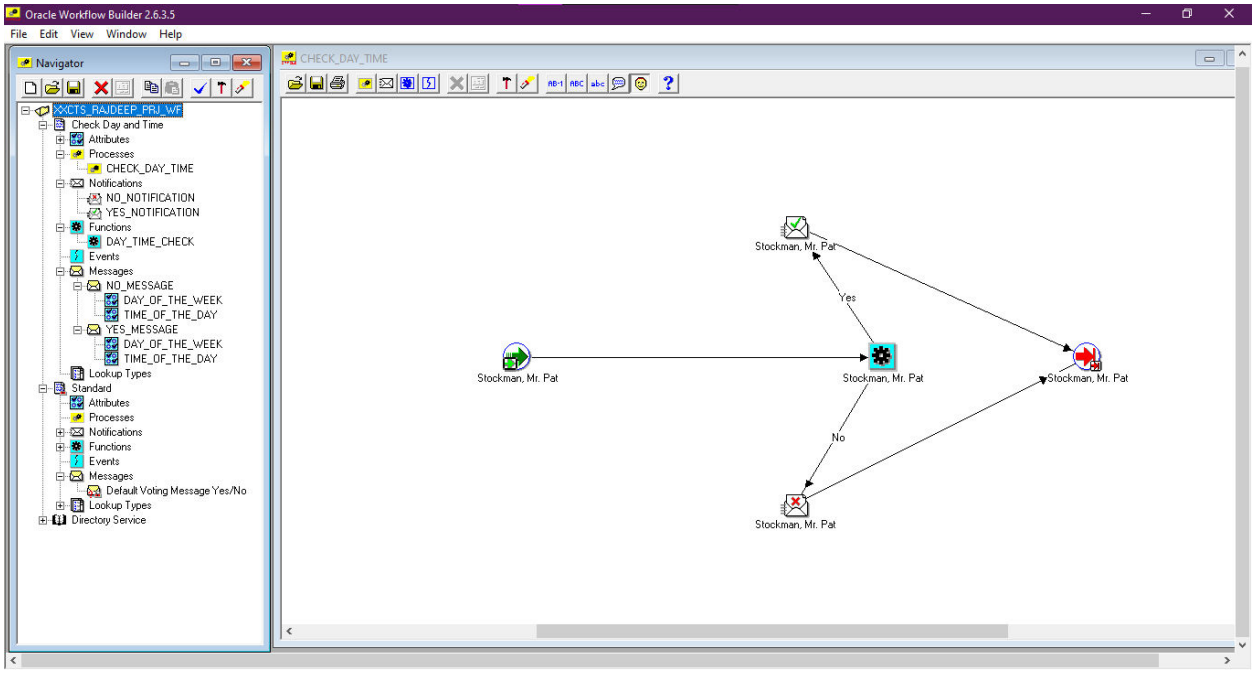


Step 14: Click on save button. Choose Database option. Given the username, password and database name and saved it. Or we can save it in the local system.

Below image is of the Navigator panel.

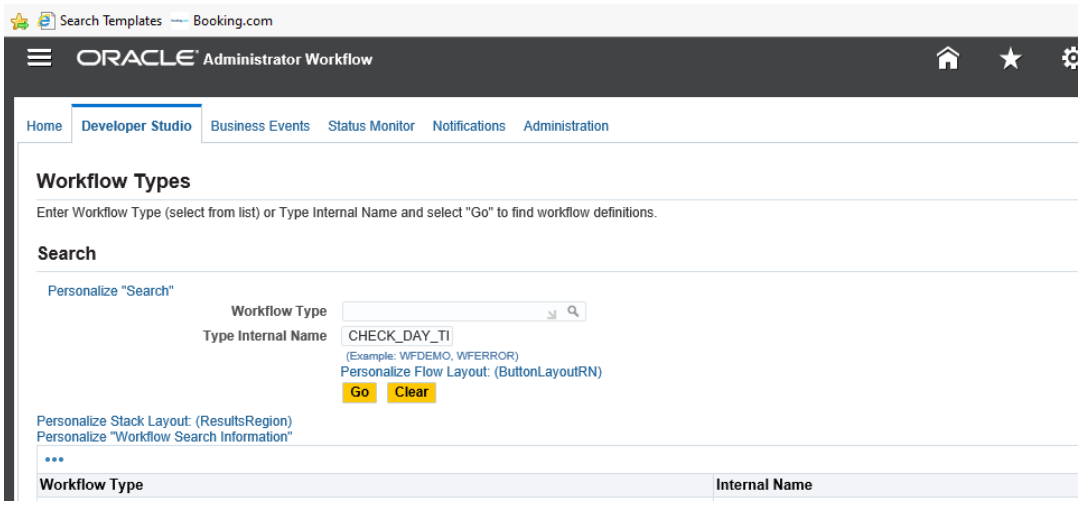


Now the below image is the full diagram of Workflow diagram and the navigator in the Workflow Builder window.



Step 15: Now I have to execute the workflow: Login to EBS Application. Switch to Workflow Administrator Web Applications → Administrator Workflow→ Developer Studio >

Step 16: Put the Internal name of the WorkFlow i.e. CHECK_DAY_TIME and Click on “GO”.



Step 17: Work Flow result appears. Click on RUN then Put Item Key, User Key and User Name. Click on Submit.

Step 18: Click on Status Monitor→ Put Internal Name CHECK_DAY_TIME→ Click on GO

Step 19: Once it shows status completed → Click on Status Diagram to view the Work Flow.

Conclusion:

It has been a great pleasure for us to work on this exciting and challenging project. This project proved good to us as it provided me with practical knowledge of not only programming in ORACLE ERP (E-Business Suit) stack but also gave us knowledge on how online business works.

Reference & Bibliography

1. **Inbound :**Oracle Applications: Supplier Interface in Oracle Apps R12 (oracleapps88.blogspot.com)
2. **API:** How to Create Responsibility API - Doyensys Blog

----------END----------