**PO Interface**

**What is Interface?**

Interfaces are used in Oracle Applications to integrate external systems and Data Conversion.

The interfaces are mainly used to either transfer data from Oracle Applications to a **flat file** or data from legacy system to **Oracle Applications**.

Interfaces in oracle apps r12 are ways to exchange data between Oracle Applications and external systems.

**The main components in Interfaces: -**

* Transfer Program (The data Transferred from Staging table to Interface table.)
* Interface Table
* Import Program (The data Transferred from Interface table to Base table.)

1. **Transfer Program:**

Sometimes, we need to move data from one system to another. Oracle Applications is a system that has many modules, such as accounting, inventory, sales, etc. Each module has its own data. If we want to move data from one module to another in Oracle Applications, we can use Transfer Programs that are already built-in. But if we want to move data from another system that is not Oracle Applications, we have to make our own Transfer Programs. Transfer Programs are usually written in PL/SQL, JAVA or SQL Loader, which are languages that computers understand.

1. **Interface Tables:**

The Interface tables basically have 4 types of columns.

* Mandatory Columns.
* Conditionally Required Columns.
* Optional Columns.
* Internal Processing Columns.

**Mandatory Columns: -**

These are the primary columns of the destination tables (oracle application module tables) that must be present. Only the Import Program can transform the records from source to destination with the help of required columns.

**Conditionally Required Columns: -**

These columns' values are determined by taking the values of the mandatory columns. For example, you must include conditionally required columns such as Currency conversion rate, Conversion Time, and Conversion Date if you are converting foreign currency transactions to Indian rupees.

**Optional Columns: -**

When a customer needs to move some extra data from the source to the destination, they are used. These are based on client’s requirement.

**Internal Processing Columns: -**

Internal Processing Columns are those that contain the Status and Error Messages. These are unique to Interface Table alone. If a record fails validation during import from the Interface Table to the Destination Table, the Import Program will use these columns to update the status and error message.

1. **Import Program: -**

The Oracle Application Package will offer import programs for all Interface Tables. The destination modules are typically registered with these. The languages used to create these import programs are PL/SQL, Java, C, C++, etc.

**Types of Interfaces: -**

There are 2 types of Interfaces:

1. **Inbound Interface.**
2. **Outbound Interface.**

**Inbound Interface:** These interfaces are used to transfer data from external systems to Oracle Applications.

**Inbound Process: -**

1. Get data from legacy system Called Flat file.
2. Load the flat file data into temporary table called staging table using SQL loader
3. (control files are used .ctl extensions)
4. Validate the data in staging table using PL/SQL procedures to define rules of validation.
5. Successful Validation takes up the data to the interface tables.
6. These Validate data can be directed to base tables using predefined Seeded programs.

**Outbound Interface:** These interfaces are used to transfer data from Oracle Applications to External systems.

**Outbound Process: -**

1. Develop the PL/SQL Program (Either Procedure or Package)
2. Write the Cursor to retrieve the data from database tables.
3. Create file or Open the File by using UTL\_File.fopen().
4. Open the Cursor
5. If any validations are there write the validations
6. Transfer the Data into File by using UTL\_File.Put\_Line().
7. Close the Cursor.
8. Close the File by using UTL\_File.fclose();
9. Register the Procedure or Package as Concurrent Program and submit from SRS Window.

**We will use following three functions to generate the file.**

1. Utl\_File.fopen = To open (or) Create the file
2. Utl\_File.Put\_line = To Transfer the data into the File.
3. Utl\_File.fclose = To close the File after Data transfer.

**Interface vs Application Program Interface:**

Interface are used to transfer data from legacy system to oracle application System where as API is used convert the data from one form to another form with in the Oracle Application Module.

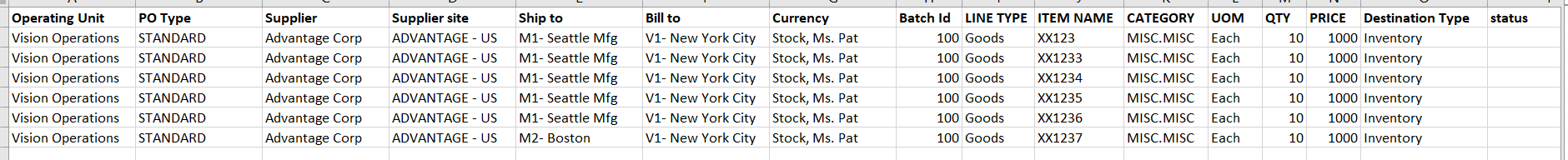
**Interfaces and Conversions Process:**

Data can be feeded in 3 ways:

1. **The data can be entered using the Application Screens.**
2. **The data can be entered using Oracle’s Open System Interface.**
3. **The data can be stored in the database table directly.**

**Example for Inbound Process: -**

According to the flat file data we create a table and control file.



**Step 1:** Create a Staging table to insert data from flat file.

CREATE TABLE **xxbb\_po\_hed\_line\_dist\_stg** (

operating\_unit VARCHAR2(240),

document\_type\_code VARCHAR2(25),

vendor\_name VARCHAR2(240),

vendor\_site\_code VARCHAR2(15),

ship\_to\_location VARCHAR2(60),

bill\_to\_location VARCHAR2(60),

buyer VARCHAR2(80),

batch\_id NUMBER,

line\_type VARCHAR2(25),

item VARCHAR2(1000),

category VARCHAR2(2000),

unit\_of\_measure VARCHAR2(25),

quantity NUMBER,

unit\_price NUMBER,

destination\_type VARCHAR2(25),

status CHAR (1),

creation\_date DATE,

created\_by NUMBER,

last\_update\_date DATE,

last\_updated\_by NUMBER,

last\_update\_login NUMBER

);

**Step 2:** Create a control file to load the flat file data into the staging table.

**options(SKIP=1)**

**LOAD DATA**

**infile \***

**TRUNCATE**

**INTO TABLE XXBB\_PO\_HED\_LINE\_DIST\_STG**

**FIELDS TERMINATED BY ','**

**OPTIONALLY ENCLOSED BY '"'**

**TRAILING NULLCOLS**

(OPERATING\_UNIT "TRIM(:OPERATING\_UNIT)"

,DOCUMENT\_TYPE\_CODE "TRIM(:DOCUMENT\_TYPE\_CODE)"

,VENDOR\_NAME "TRIM(:VENDOR\_NAME)"

,VENDOR\_SITE\_CODE "TRIM(:VENDOR\_SITE\_CODE)"

,SHIP\_TO\_LOCATION "TRIM(:SHIP\_TO\_LOCATION)"

,BILL\_TO\_LOCATION "TRIM(:BILL\_TO\_LOCATION)"

,Buyer "TRIM(:Buyer)"

,BATCH\_ID "TRIM(:BATCH\_ID)"

,LINE\_TYPE "TRIM(:LINE\_TYPE)"

,ITEM "TRIM(:ITEM)"

,CATEGORY "TRIM(:CATEGORY)"

,UNIT\_OF\_MEASURE "TRIM(:UNIT\_OF\_MEASURE)"

,QUANTITY "TRIM(:QUANTITY)"

,UNIT\_PRICE "TRIM(:UNIT\_PRICE)"

,DESTINATION\_TYPE "TRIM(:DESTINATION\_TYPE)"

,STATUS "nvl(:STATUS,'N')"

,CREATION\_DATE SYSDATE

,CREATED\_BY "fnd\_global.user\_id"

,LAST\_UPDATE\_DATE SYSDATE

,LAST\_UPDATED\_BY "fnd\_global.user\_id"

,LAST\_UPDATE\_LOGIN "fnd\_global.LOGIN\_ID")

**Step 3:** After creating the creating file, we need to place the control file and flat file in respective module bin folder

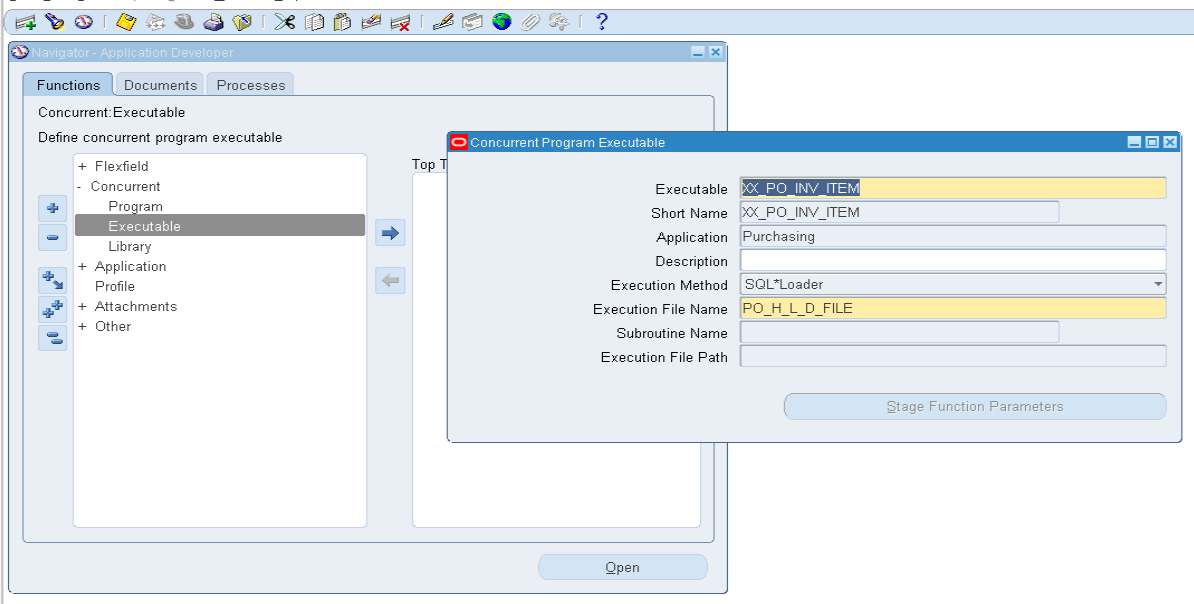
**Ex-Path:** D:\oracle\VIS\apps\apps\_st\appl\po\12.0.0\bin

**Step 4:** Create a concurrent program for inserting records from flat file to stagging table.

* First, we create an executable.

**Navigation:** Application Developer 🡪 Concurrent 🡪 Executables

* Executable Name : ‘XX\_PO\_INV\_ITEM’
* Short Name : ‘XX\_PO\_INV\_ITEM’
* Application : ‘Purchasing’
* Execution Metho : ‘SQL \*Loader’
* Execution Method Name : <ctl file name> ‘PO\_H\_L\_D\_FILE’

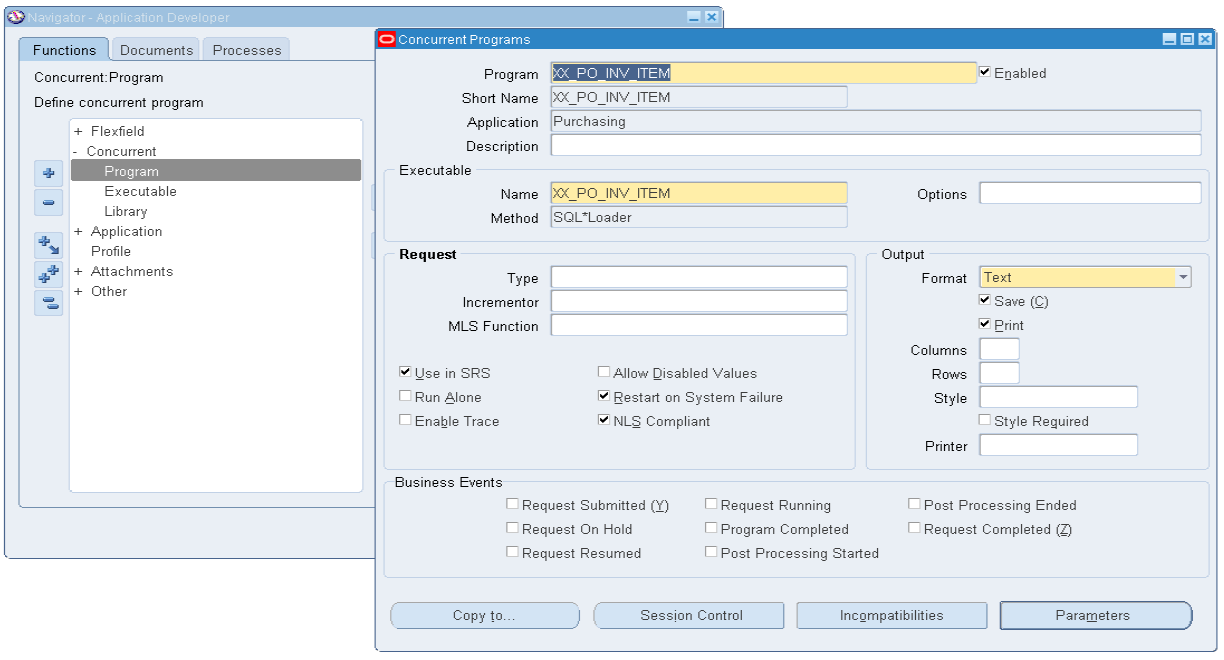


Click on **SAVE** Button.

* Create a Concurrent program for the above executable

Navigation: Application Developer 🡪 Concurrent 🡪 program

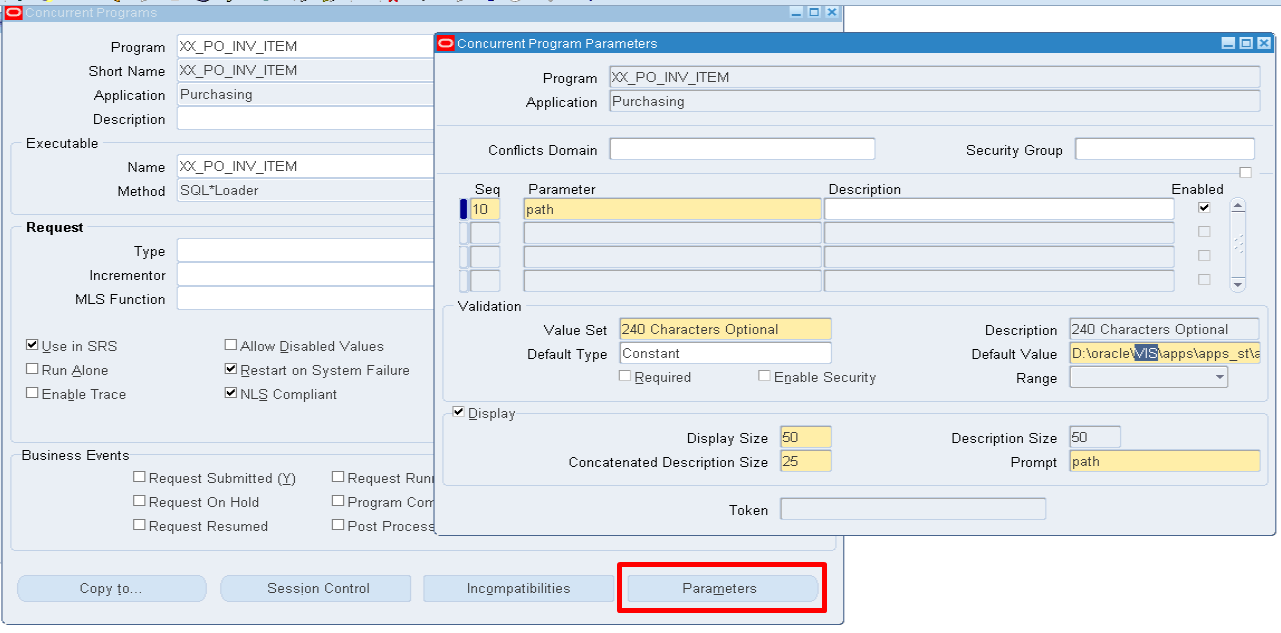
* + Program : ‘XX\_PO\_INV\_ITEM’
  + Short Name : ‘XX\_PO\_INV\_ITEM’
  + Application : ‘Purchasing’
  + Executable Name : ‘XX\_PO\_INV\_ITEM’



Then click on **Parameters**.

Give the parameter name as **PATH**

* Value Set :’240 Characters Optional’
* Default Type : ‘Constant’



Click on **SAVE** Button.

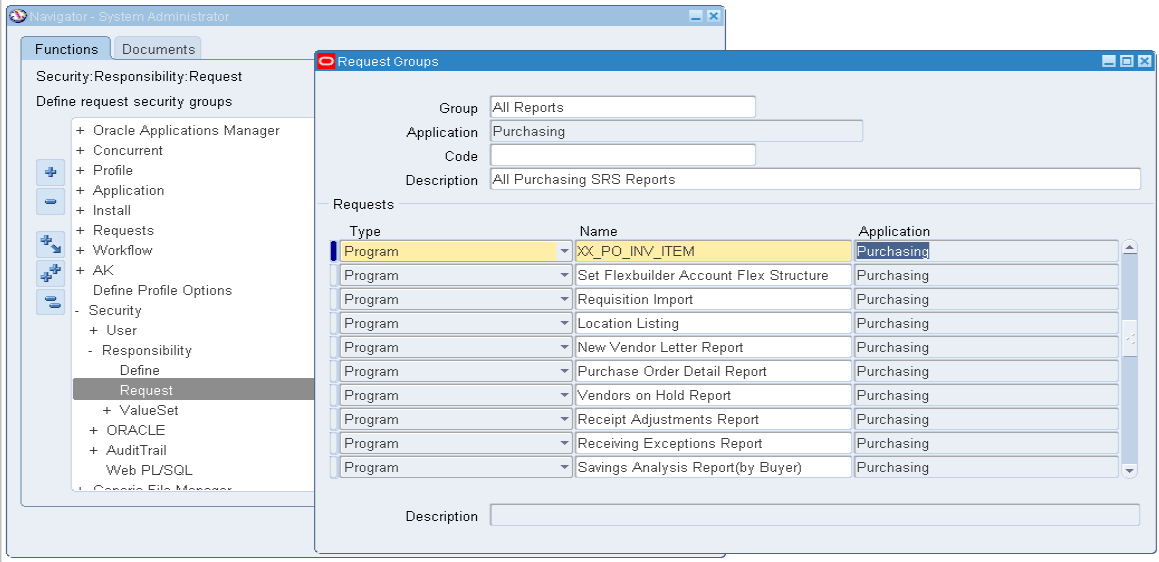
* Assign a Concurrent Program to the Respective responsibility

Example : (**Purchasing, Vision Operation (USA)**)

**Navigation:** System Administrator 🡪 Security 🡪 Responsibility 🡪 Request

* Group : ‘All Reports’
* Application : ‘Purchasing’

Click on **type** and add a concurrent program <XX\_PO\_INV\_ITEM>



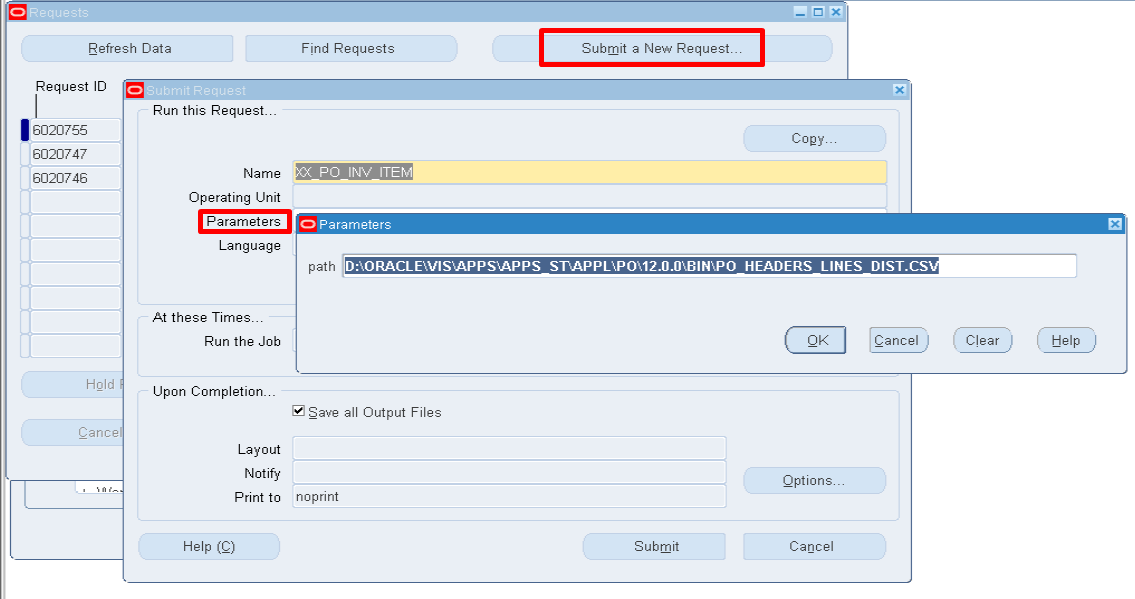
Click on **SAVE** Button.

* Run the Program

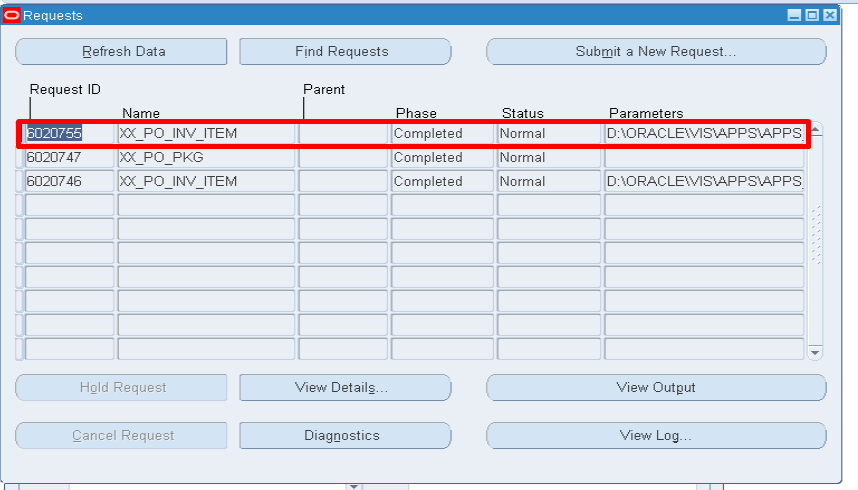
**Navigation:** Purchasing, Vision Operation (USA)🡪 top menu🡪 view 🡪 request 🡪

Submit a new Request

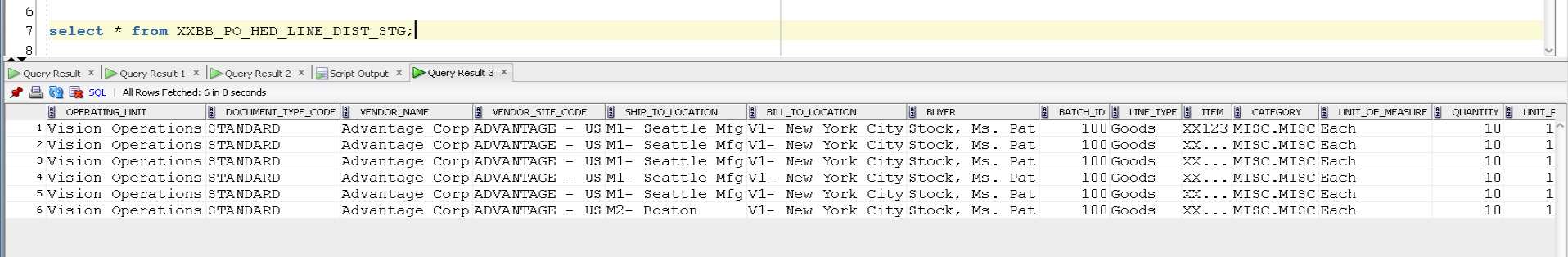
* Name: ‘XX\_PO\_INV\_ITEM’

****

Click on **OK,** then Click on **SUBMIT** Button.



The Concurrent program Successfully completed… then the data in flat file is inserted into a stagging table.



**Step 5:** Create a concurrent program to transfer the data from stagging table to interface table.

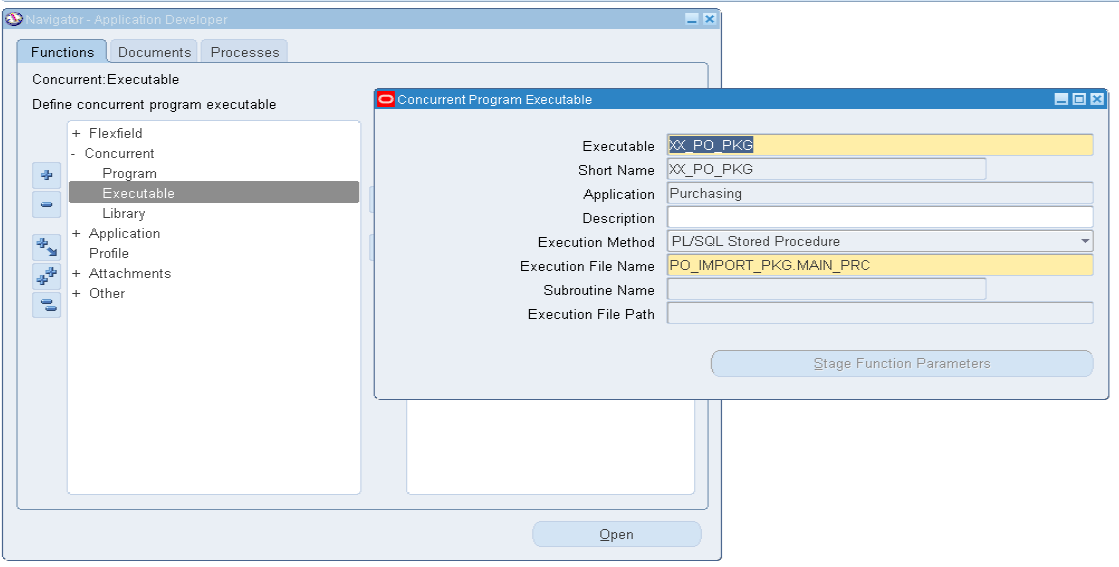
* Firstly, we create an Executable.

**Navigation:** Application Developer 🡪 Concurrent 🡪 Executables

* Executable Name : ‘XX\_PO\_PKG’
* Short Name : ‘XX\_PO\_PKG’
* Application : ‘Purchasing’
* Execution Metho : ‘PL/SQL Stored Procedure’
* Execution Method Name : <Package.Procedure name> Example: - ‘PO\_IMPORT\_PKG.MAIN\_PRC’

To see the package Spec and body click on below file. 

The file talk about the validations required to insert a data from stagging table to interface table of Purchase Order.

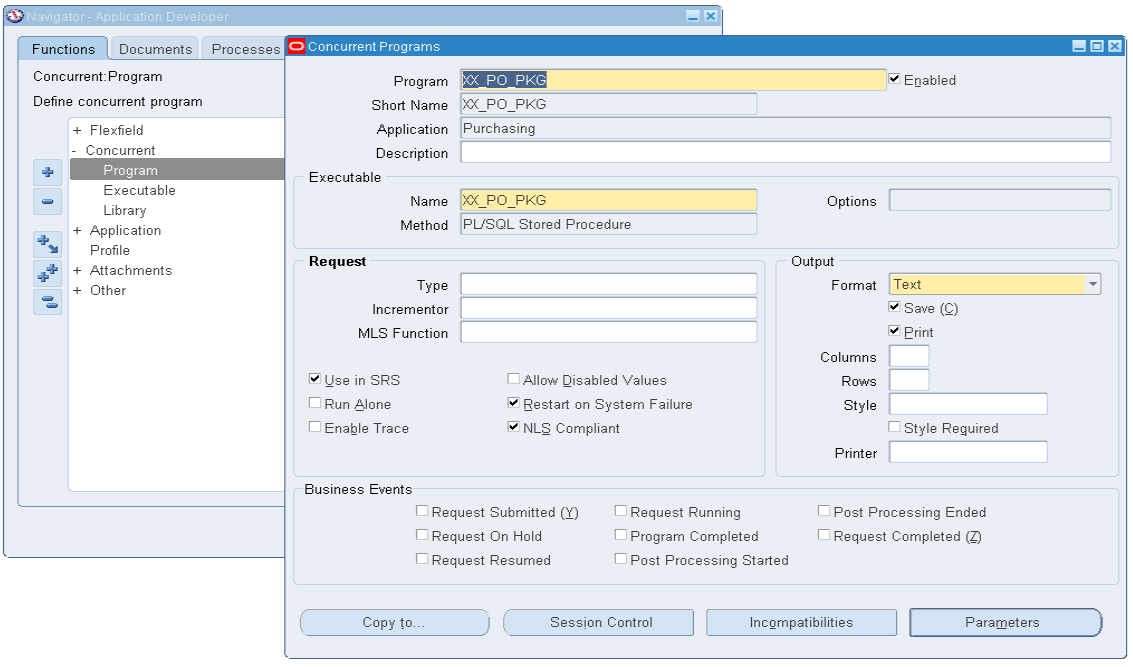


Then click on **SAVE** Buton.

* Create a Concurrent program for the above executable

Navigation: Application Developer 🡪 Concurrent 🡪 program

* + Program : ‘XX\_PO\_PKG’
  + Short Name : ‘XX\_PO\_PKG’
  + Application : ‘Purchasing’
  + Executable Name : ‘XX\_PO\_PKG’



Note: For this Executable method file, there is no parameters. So, no need to create parameters.

Click on **SAVE** Button.

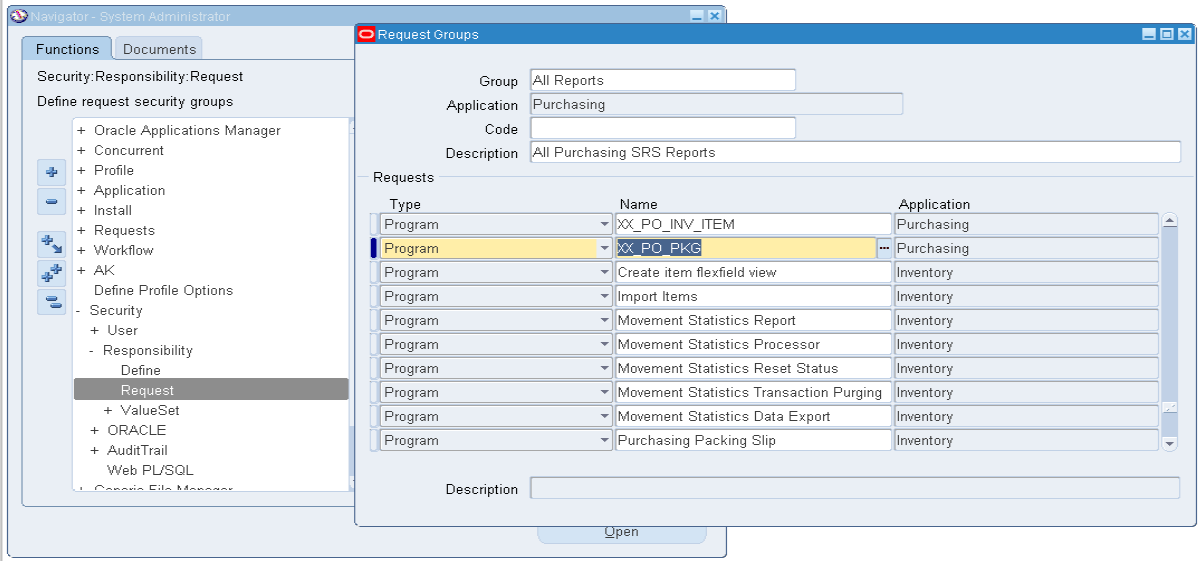
* Assign a Concurrent Program to the Respective responsibility

Example: (**Purchasing, Vision Operation (USA)**)

**Navigation:** System Administrator 🡪 Security 🡪 Responsibility 🡪 Request

* Group : ‘All Reports’
* Application : ‘Purchasing’

Click on **type** and add a concurrent program <XX\_PO\_PKG>



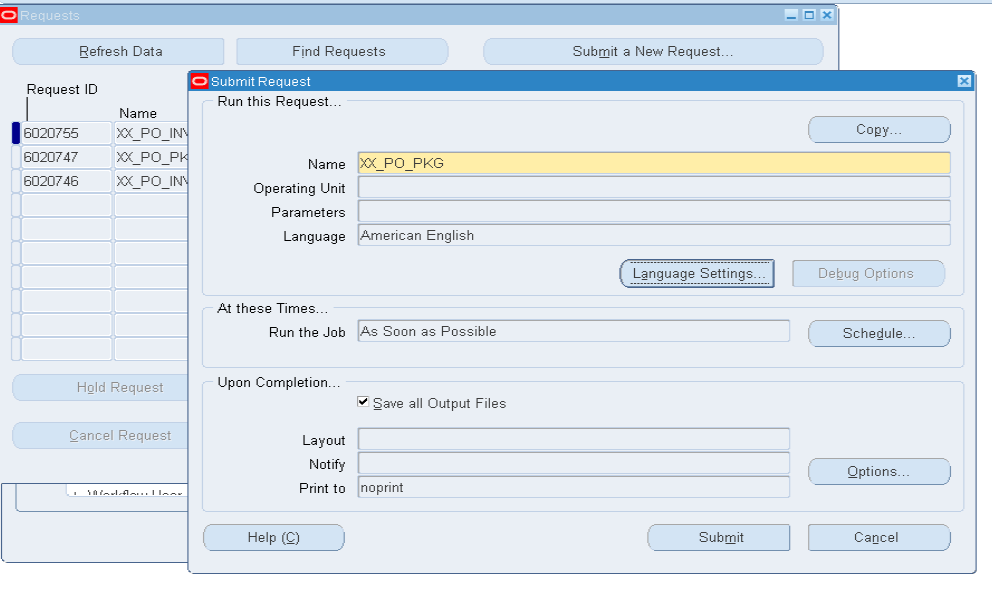
Click on **SAVE** Button.

* Run the Program

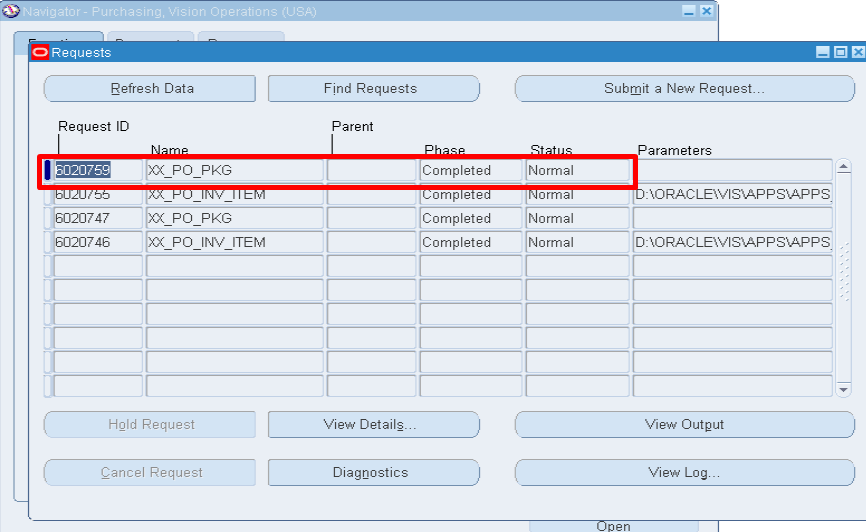
**Navigation:** Purchasing, Vision Operation (USA)🡪 top menu🡪 view 🡪 request 🡪

Submit a new Request

* Name: ‘XX\_PO\_PKG’

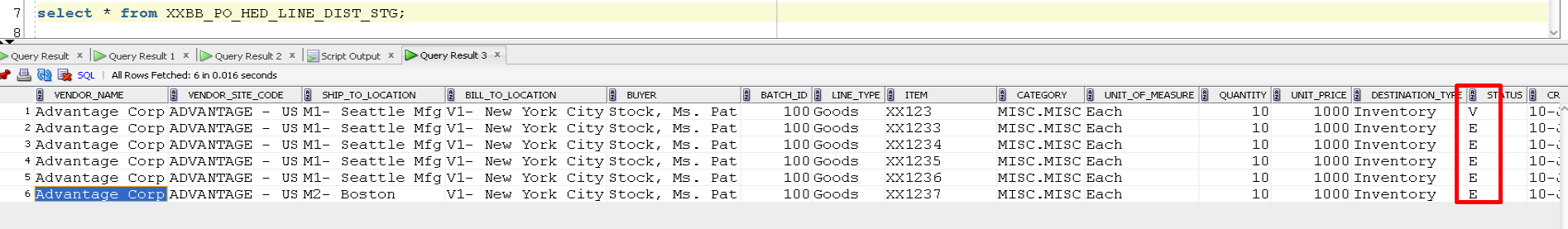


Click on **SUBMIT** Button.



The Concurrent program Successfully completed… then the data in stagging table is inserted into a respective interface table.

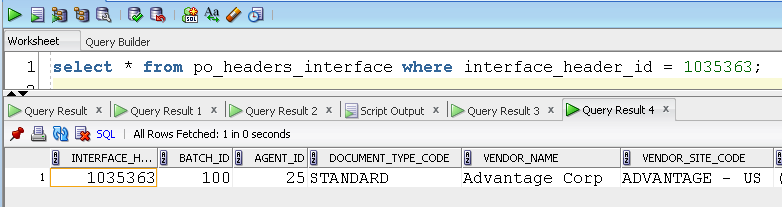
In a staging table, the status column is updated only when all validations pass in the preceding concurrent program.



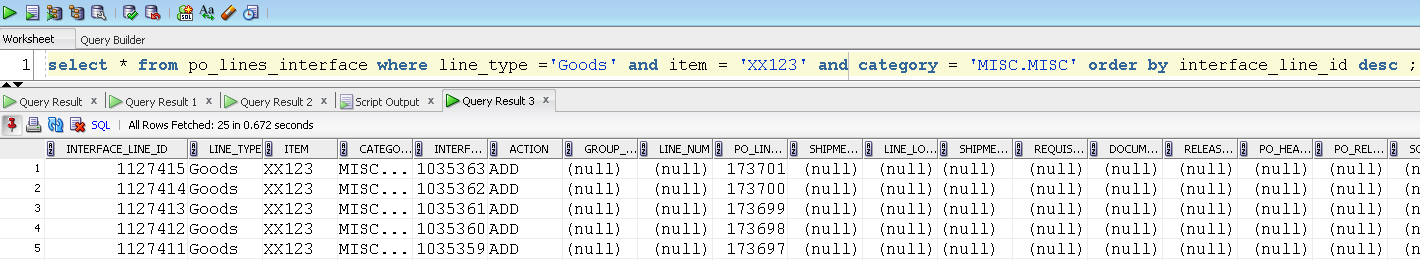
After Step 4, **Status** of the stagging table is by default ‘E’,

but after Step 5 **Status** of the stagging table is ‘V’.

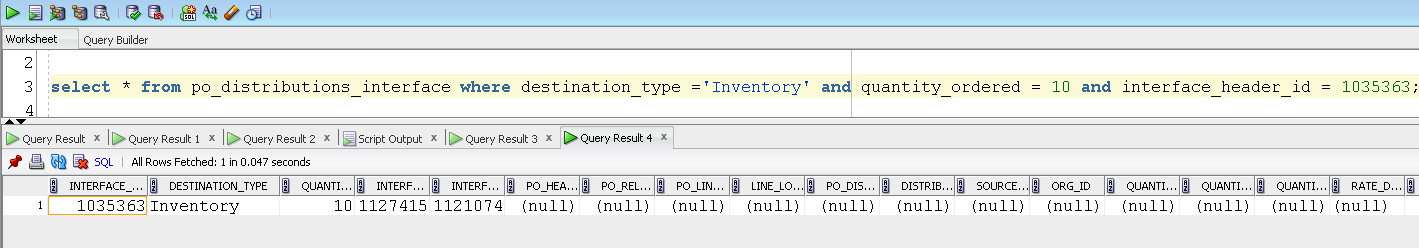
* Data inserted in **PO\_HEADERS\_INTERFACE**



* Data Inserted in **PO\_LINES\_INTERFACE**



* Data inserted in **PO\_DISTRIBUTIONS\_INTERFACE**

****

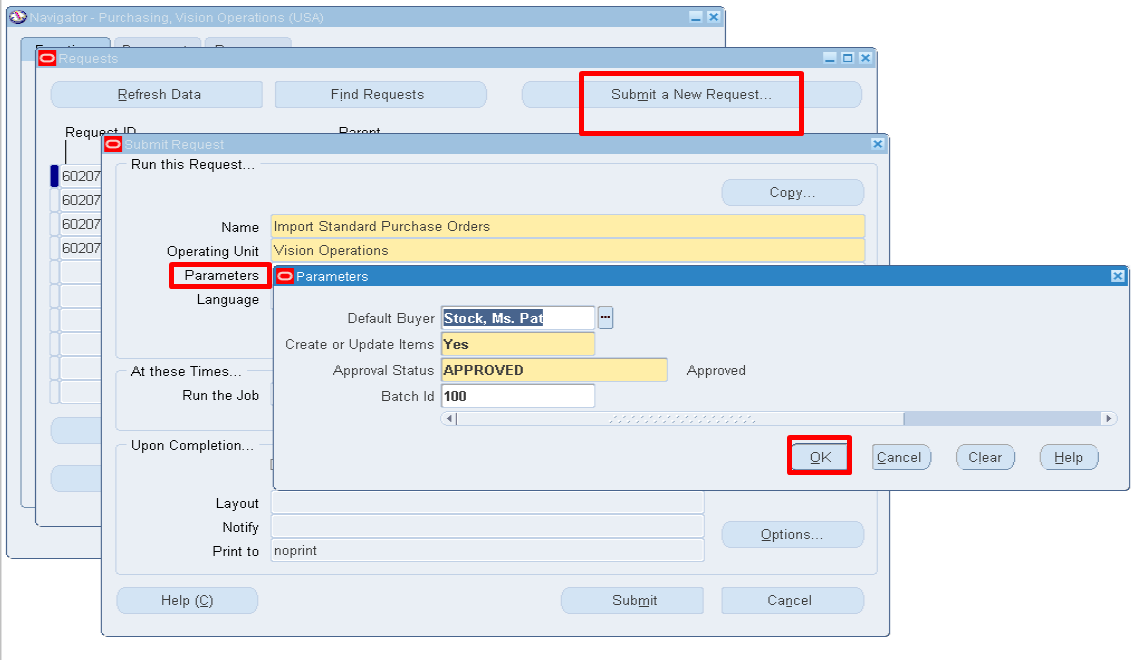
**Step 6:** Now we need to transfer data from **Interface Table** to **Base table,** for that we need to run one seeded Concurrent Program (Import Standard Purchase Orders)

**Navigation:** Purchasing, Vision Operation (USA)🡪 top menu🡪 view 🡪 request 🡪

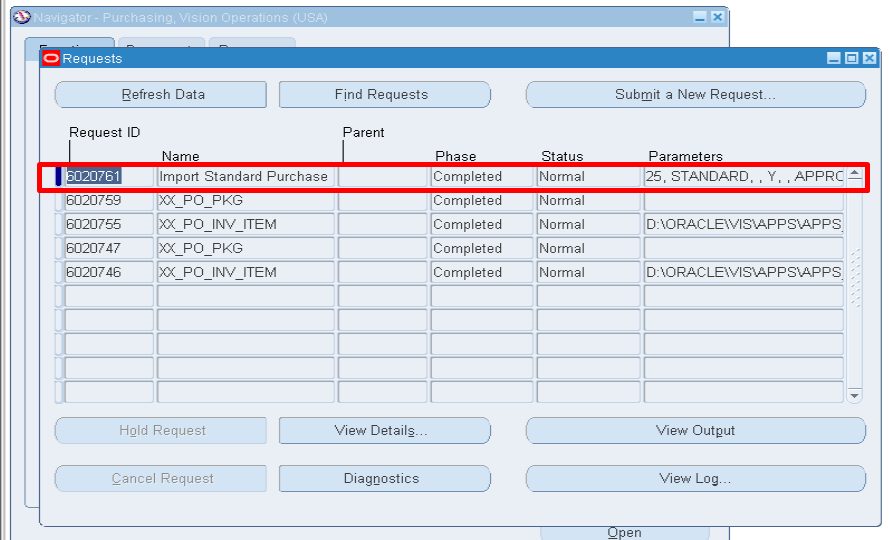
Submit a new Request

* Name : ‘XX\_PO\_PKG’
* Parameters 🡪

1. Default Buyer : ‘Stock. Mr. Pat’
2. Create or Update Items : ‘Yes’
3. Approved Status : ‘Approved’
4. Batch Id : ‘100’

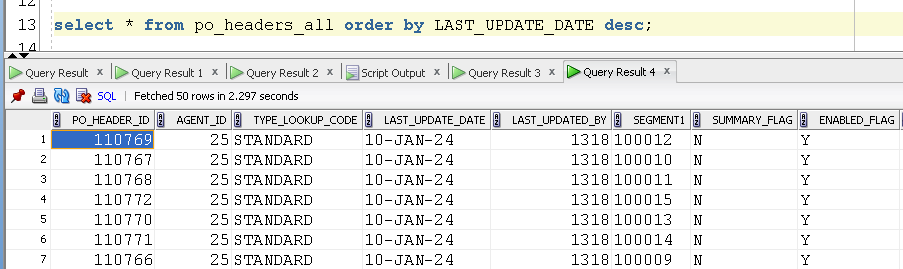


Click on **OK**, then Click on **SUBMIT** Button.

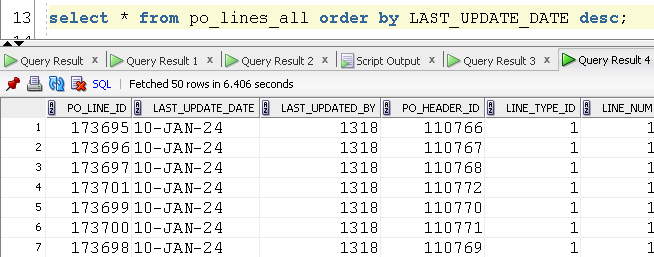


The Concurrent program Successfully completed… then the data in **Interface table** is inserted into a respective **Base table**.

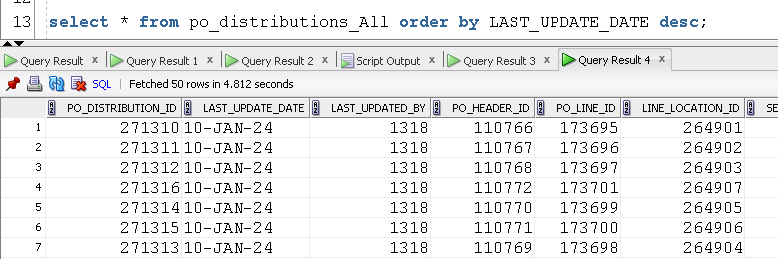
* Data inserted in **PO\_HEADERS\_ALL**



* Data inserted in **PO\_LINES\_ALL**



* Data inserted on **PO\_DISTRIBUTIONS\_ALL**



**PO INTERFACE TABLES: - PO BASE TABLES: -**

PO\_DISTRIBUTIONS\_INTERFACE TABLE PO\_DISTRIBUTIONS\_ALL

PO\_HEADERS\_INTERFACE TABLE PO\_HEADERS\_ALL

PO\_LINES\_INTERFACE TABLE PO\_LINES\_ALL

**Outbound Interface Example Using UTL FILE: -**

Outbound Interface will be used to extract the data from oracle Database tables into the flat files. While developing the outbound Interface we will use **UTL\_File** to Extract the data.

**Step 1:** First, we need to create our own directory in oracle

Syntax: **CREATE OR REPLACE DIRECTORY** Directory\_name **AS** ‘Path for directory’.

Example: **CREATE OR REPLACE DIRECTORY** utl\_file AS **'E:'**;

**Step 2:** After creating directory, we need to give necessary grants to directory to read, write and execute.

Syntax: **GRANT READ, WRITE ON DIRECTORY** directory\_name **TO** username;

Example: **GRANT READ, WRITE ON DIRECTORY** utl\_file TO **APEX**;

**Here is an example or Outbound Interface:**



Check the above file for Outbound Interface.

**We will use following three functions to generate the file.**

1. Utl\_File.fopen = To open (or) Create the file
2. Utl\_File.Put\_line = To Transfer the data into the File.
3. Utl\_File.fclose = To close the File after Data transfer.

**OUTPUT FILE: -**

****

* Open the above file to see the output.