

R12 Interfaces and API's - IBM Graduate Program

Student Guide

D81888GC20

Edition 2.0

August 2013

ORACLE®

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Trademark Notice

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

Contents

- Lesson 1: Overview of Interface**
- Lesson 2: Writing Interface Program**
- Lesson 3: Types of Interfaces / Integration**
- Lesson 4: Open Interfaces**
- Lesson 5: Interfaces Deployment**
- Lesson 6: Debugging Interface Program Errors**
- Lesson 7: List of Interface Tables**
- Case Study**
- Appendix A**
- Appendix B**
- Appendix C**

1

Overview of Interface

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

After completing this lesson, you should be able to understand

- What is an Interface
- When do we use Interface Table
- Understanding Interface Tables
- Business Benefit
- ERDs and Applications Technology
- Basic Interface Process Flow



Overview of Interface

Interface:

Interface is a communication channel that allows transfer of data between the source system and the target system.

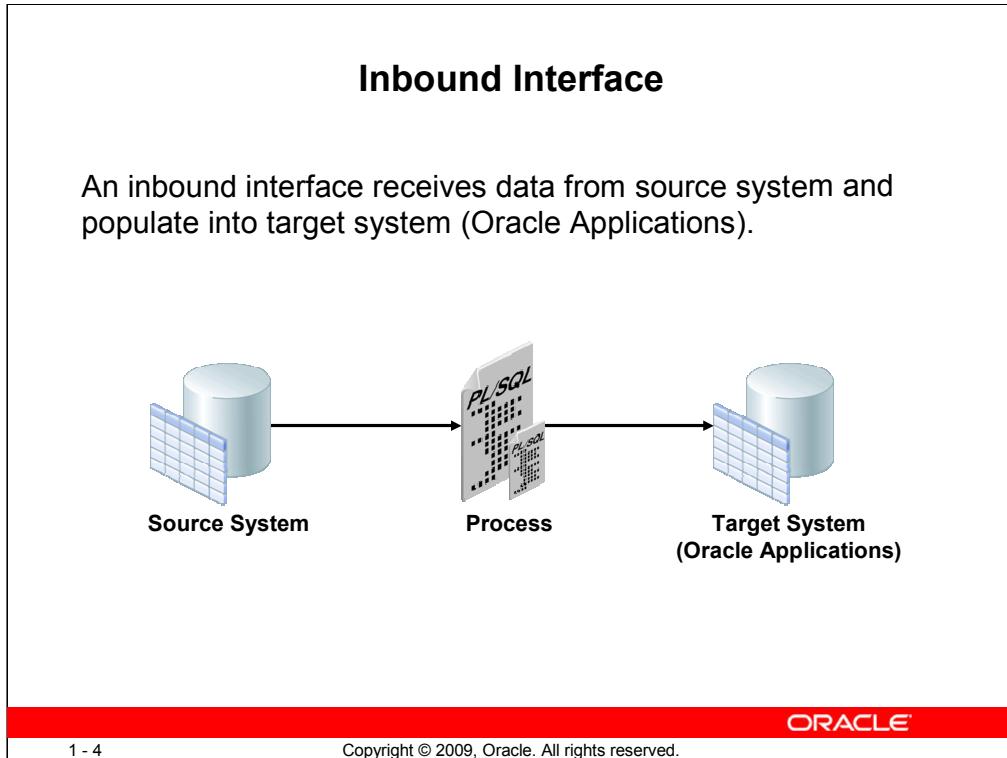
Types of Interface:

- Inbound Interface
- Outbound Interface

ORACLE®

1 - 3

Copyright © 2009, Oracle. All rights reserved.

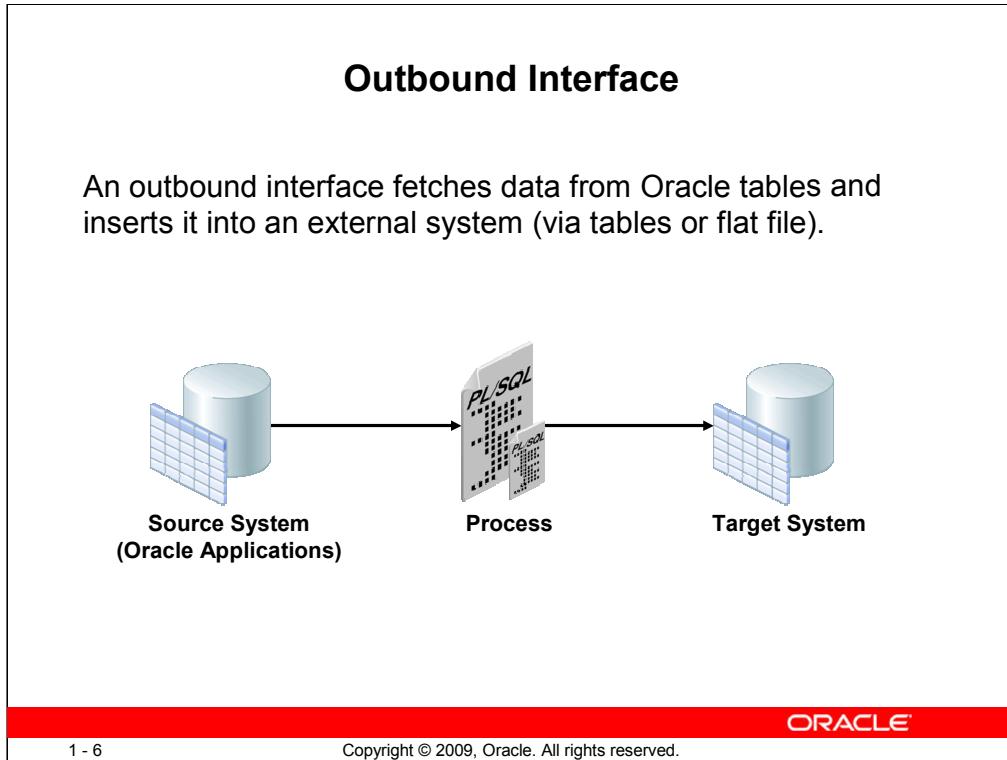


Inbound Interface: Steps

1. Extract the data from source system into a flat file (csv file, text file).
2. Use SQL*Loader or equivalent tool to upload data present in the flat file into the staging (Intermediate) tables.
3. Write a PL/SQL program to process and validate the data present in the staging tables and populate them into the respective open interface tables.
4. Submit the corresponding standard oracle interface import program to transform the records present in the interface tables into Oracle base tables.

To understand the inbound interface steps, let us take the example of Payables interfaces:

1. Extract AP Invoices data from the source system into a csv file
2. a) Create custom staging tables XX_AP_INVOICES_INTERFACE, XX_AP_INVOICE_LINES_INTERFACE which has the same table structure as the interface tables
AP_INVOICES_INTERFACE and
AP_INVOICE_LINES_INTERFACE
b) Load the data from the csv file into the staging tables using SQL*Loader
3. Using a PL/SQL program, validate the records in the staging tables and populate them into the interface tables AP_INVOICES_INTERFACE and AP_INVOICE_LINES_INTERFACE
4. Run the Payables Open Interface Import Program to convert the data in the interface tables into the base tables AP_INVOICES_ALL, AP_INVOICE_DISTRIBUTIONS_ALL, AP_PAYMENT_SCHEDULES_ALL.



Outbound Interface: Steps

1. Write a PL/SQL program to extract data from Oracle base tables into a flat file.
2. The UTL_FILE utility can be used for writing operating system (OS) flat files in the target system.

What is an Interface Table

- Interface table is used as a medium for transferring data from source systems to target system (Oracle Applications).
- There are a set of open interface tables available for data upload in Oracle Applications products such as GL, AP, AR, PO, etc.

ORACLE®

1 - 7

Copyright © 2009, Oracle. All rights reserved.

For example:-

Important interface tables provided by Oracle Purchasing (PO):

- PO_DISTRIBUTIONS_INTERFACE
- PO_HEADERS_INTERFACE
- PO_INTERFACE_ERRORS
- PO_LINES_INTERFACE

Important interface tables provided by Oracle General Ledger (GL):

- GL_DAILY_RATES_INTERFACE
- GL_BUDGET_INTERFACE
- GL_IEA_INTERFACE
- GL_INTERFACE

When do we use Interface Table

- To load the data from source system into Oracle Applications system.
- To perform validation and error reporting of the record present in Interface Table.
- To correct and reload the records which are not processed successfully by the standard open interface program.

ORACLE®

1 - 8

Copyright © 2009, Oracle. All rights reserved.

Let us take the example of PO Requisition Interface

- Records present in the interface table PO_REQ_INTERFACE_ALL are processed by Requisition Import concurrent program and the data is populated in the base tables PO_REQ_HEADERS_ALL, PO_REQ_LINES_ALL and PO_REQ_DISTRIBUTIONS_ALL
- The Requisition Import generates an error message for every failed record and creates a row in the PO_INTERFACE_ERRORS table with detailed information about each error.
- In case of any validation error, the PROCESS_FLAG column in the PO_REQ_INTERFACE_ALL table is updated as ERROR
- The errored records can be checked by using the following query

```
SELECT PIE.*  
FROM   PO_REQ_INTERFACE_ALL PRI  
       ,PO_INTERFACE_ERRORS PIE  
WHERE  PRI.transaction_id = PIE.interface_transaction_id  
AND    PRI.request_id      = <concurrent program request id>
```

Understanding Interface Table

- Interface table is a Oracle standard table to migrate data from an external system to various Oracle Applications products.
- Oracle Applications products have their own interface tables and standard import programs to process the source data.

ORACLE®

1 - 9

Copyright © 2009, Oracle. All rights reserved.

- Oracle Applications use interface tables to insert and update data in Oracle Applications. For example, sales order records from source systems can be generated into Oracle Applications sales orders automatically using Interface tables. Data is never populated into Oracle Applications base tables directly. Instead, data is first populated into the interface tables and then using Oracle standard concurrent programs, data is transferred from interface tables to base tables. This ensures that all business logic and processing is handled using Oracle components.
- Some of the Oracle Applications products have their own interface tables. For example:
 - AR - RA_INTERFACE_LINES_ALL
 - AP - AP_INVOICES_INTERFACE
 - GL - GL_INTERFACE
 - PO - PO_HEADER_INTERFACE
 - OM - OE_HEADERS_IFACE_ALL

Understanding Interface Table

The following points should be considered while populating records into the interface tables

1. Mandatory data must be passed while loading data into interface table.
2. Status column of the interface table must be passed with appropriate value for processing (Ex. "NEW", "N").
3. If there are any validation error, the interface error table must be updated with proper error message.
4. Check Oracle eTRM for column description of interface table.

ORACLE®

1 - 10

Copyright © 2009, Oracle. All rights reserved.

- Standard Who columns like CREATED_BY, CREATION_DATE, LAST_UPDATED_BY, LAST_UPDATE_DATE, LAST_UPDATE_LOGIN must be passed along with the other mandatory columns in the interface table.
- For processing interface records through standard import programs, the status column must be updated with the proper value. For example, in MTL_TRANSACTION_SYSTEMS interface table, the PROCESS_FLAG column must be updated with "1".
- If validation error occurred while processing the records in interface table, the standard open interface import program updates the error message in the interface table for errors with proper error message. For example, in case of PO Interface Table the PO_INTERFACE_ERRORS table gets populated if any record fails while processing.
- Follow the steps for using eTRM (Electronic Technical Reference Manual)
 - a) Log into metalink.oracle.com
 - b) Click on tab Knowledge Management
 - c) Click on eTRM
 - d) Enter your table name

Business Benefits

- Frequency
- Batch
- Volume
- Technology
- Security
- Easy to use Interface and API's

ORACLE®

1 - 11

Copyright © 2009, Oracle. All rights reserved.

- **Frequency:** Frequency can be defined as the number of times we use interfaces for loading data into Oracle Applications. Most of the interfaces are run during month end processing while few are run on a nightly basis.
- **Batch:** Multiple interfaces can be run as a batch process.
- **Volume:** The volume of data that needs to be passed in interface depends on the technology.
- **Technology:** Oracle provides various technology to upload the source data and create the records in the base tables. Few of them are Open Interface Tables, API's, BPEL, EDI and EAI.
- **Security:** Following data security issues are taken care by Interface Tables.
 - One way sync (Read Only)
 - vs. bi-directional (Read Write)
 - vs. access in place
- **Easy to use interface and API's:** Oracle Applications provide various open interface tables and API's for loading source data into Oracle applications. These interface table and API's are easy to use while implementing business logic.

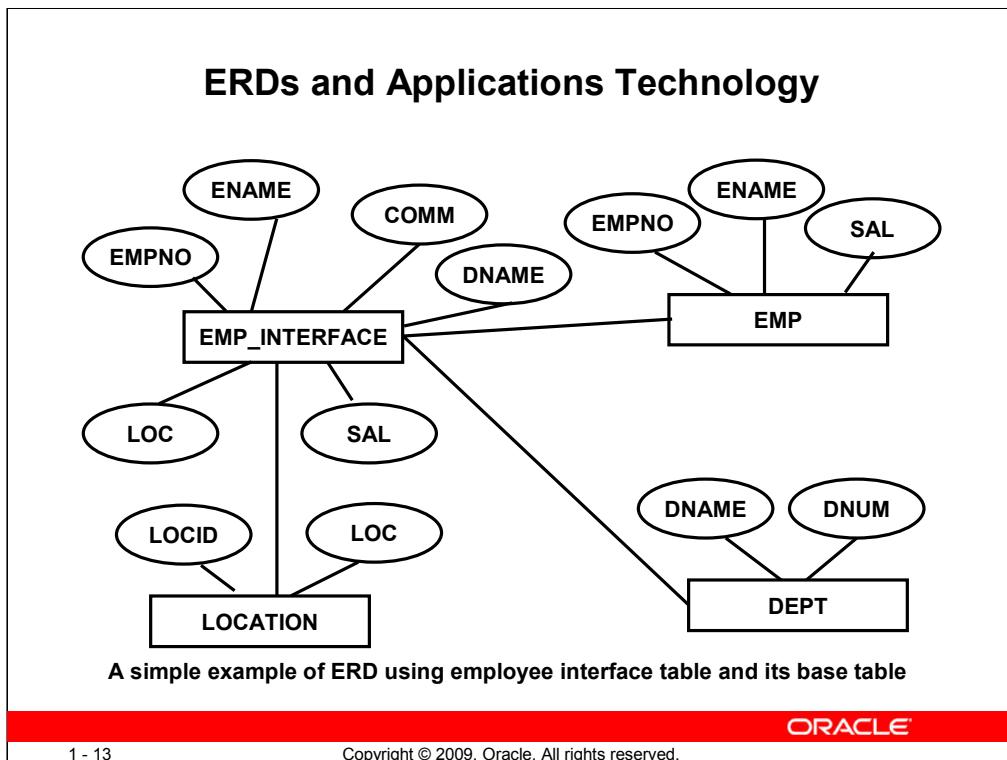
ERDs and Applications Technology

- Entity relationship diagram (ERD) can be used to show the relationship between the open interface tables with the corresponding Oracle Applications base tables.

ORACLE®

1 - 12

Copyright © 2009, Oracle. All rights reserved.



- Consider the ERD diagram for the classical EMP and DEPT table.
- In the employee interface table the employee source data is loaded.
- When the data from the interface table is processed for creating records in base tables, the validation program will refer the EMP and DEPT records for the existence of the employee and the associated department.
- If data validation completes successfully, the employee records are created successfully in the EMP Table.

Basic Interface process flow

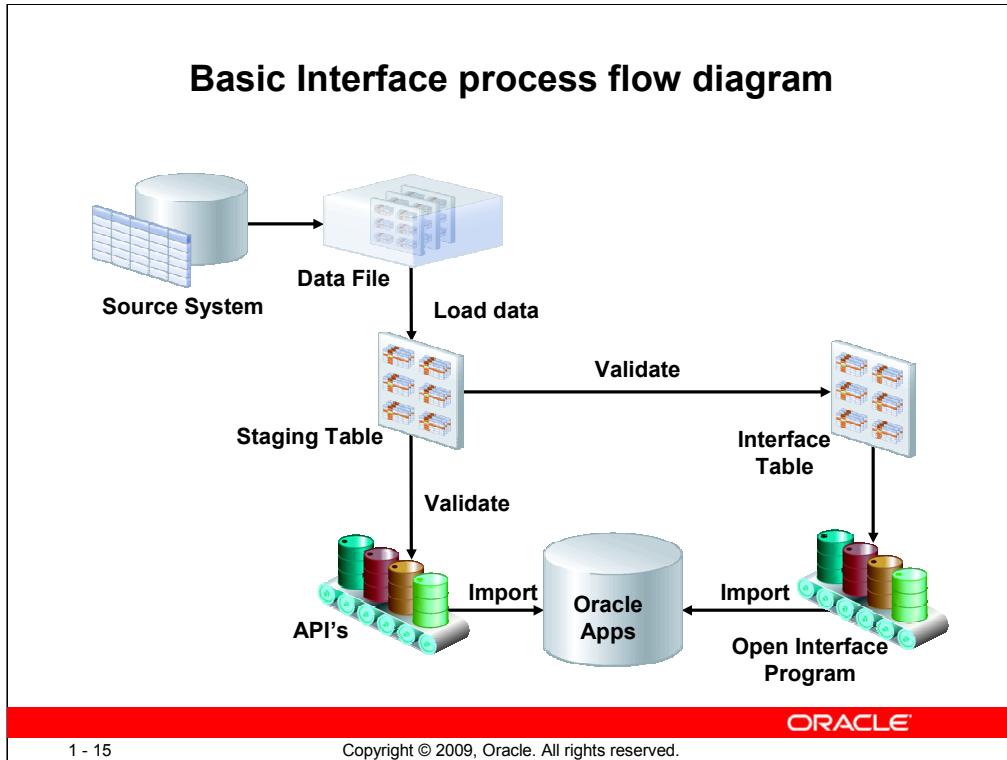
- Requirement Definition.
- A control file is created based on the flat file.
- The control file loads the data into staging (custom) table.
- Through PL/SQL programs the data is mapped, validated and then populated into the interface tables.
- The Oracle standard programs populates the data into the base tables from the interface tables.

ORACLE®

1 - 14

Copyright © 2009, Oracle. All rights reserved.

- A technical design document (MD.070) is prepared by analyzing the requirements specified in functional design document (MD.050)
Note: MD: Module Design and Build
- Control file is used for loading the source data to the Staging / Interface using either loader program or SQL* Loader utility.
- A PL/SQL stored procedure is created for processing and validating the records present in staging table and loading the validated records in the interface table.
- Once the validated records are loaded successfully in interface table, the respective standard import program is submitted which will process the records present in the interface table and will create records in the Oracle Application base tables.
- In case of validation errors, the erroneous records are either populated in the corresponding interface error table or the error_code and error_description columns of the same interface table are populated.



- Data file is extracted from the source system.
- Create a control file and call the loader program to load source data to staging table using the control file.
- When all the records are loaded successfully, execute PL/SQL program to validate the records present in staging table and populate them in the interface table.
- Call the standard open interface program or the corresponding API to import the data into the Oracle Applications base tables.

Summary

You should now be able to understand :

- What is a Interface Table
- When do we use Interface Table
- Understanding Interface Tables
- Business Benefit
- ERDs and Applications Technology
- Basic Interface Process Flow



Writing Interface Program

2

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

After completing this lesson, you should be able to understand the following:

- Writing Interface Programs
 - Using PI/SQL
 - Using JAVA
 - Using BPEL/SOA

ORACLE

2 - 2

Copyright © 2009, Oracle. All rights reserved.

PL/SQL Code

```
CREATE OR REPLACE PACKAGE XX_INTERFACE_PKG AS
PROCEDURE main (x_errbuf      OUT NOCOPY VARCHAR2
                ,x_retcode    OUT NOCOPY NUMBER
);
END XX_INTERFACE_PKG;
```

ORACLE

2 - 3

Copyright © 2009, Oracle. All rights reserved.

This is an example of GL Interface using PL/SQL

Package Specification

- main procedure
 - This procedure is used for calling validate/import/print_error private procedures
 - This procedure will be called from the Apps concurrent program which will in turn call the validate, import and print_error private procedures.
 - There are two out parameters which are mandatory to define.
 - x_errbuf - This parameter will hold the error message if concurrent program completes with error or successful message if it completes with success.
 - x_retcode - There are 3 values which x_retcode will hold. 0 for success, 1 for warning and 2 for error.

PL/SQL Code

```

CREATE OR REPLACE PACKAGE BODY XX_INTERFACE_PKG AS
PROCEDURE validate (x_errbuf OUT NOCOPY VARCHAR2
                    ,x_retcode OUT NOCOPY NUMBER
                   );
PROCEDURE import (x_errbuf OUT NOCOPY VARCHAR2
                   ,x_retcode OUT NOCOPY NUMBER
                  );
PROCEDURE print_error (x_errbuf      OUT NOCOPY VARCHAR2
                       ,x_retcode    OUT NOCOPY NUMBER
                      );
END XX_INTERFACE_PKG;

```

ORACLE

2 - 4

Copyright © 2009, Oracle. All rights reserved.

Package Body

- validate procedure
 - This procedure is used for business rule validation which will validate the records available in the interface table.
 - If records are validated successfully, it updates the status column ‘Validated’
 - If there are some error while validating the records, it updates the status column as ‘Error’ with proper error message in error message column.
- import procedure
 - This procedure is used for transferring data into application which are successfully validated in validate routine.
 - We call standard concurrent program using FND_REQUEST API which will run the standard import program for creating records in the base tables using interface data.

- `print_error` procedure
 - This procedure will be called if there are validation errors while validating the interface records.
 - It will show the records with error message.

Using Java

```
import java.sql.*;
import oracle.jdbc.driver.OracleCallableStatement;
import oracle.apps.fnd.cp.request.* ;
import java.io.*;
import oracle.apps.fnd.util.*;

public class xxInterfaceFromModule implements
    JavaConcurrentProgram
{
    \\ Write your program logic here
}

private void registerControlRecord()
{
    \\ Write your program logic here
}
```

ORACLE

2 - 6

Copyright © 2009, Oracle. All rights reserved.

- **Importing classes**
 - For compilation of java file we require standard class file.
 - The above mentioned classes in import area are required to compile the java code.
- **xxInterfaceFromModule**
 - This function will be used to define global variables, calling private function for business logic etc.
- **registerControlRecord**
 - This function is used to register the database control like opening and closing the database connections.

Using Java

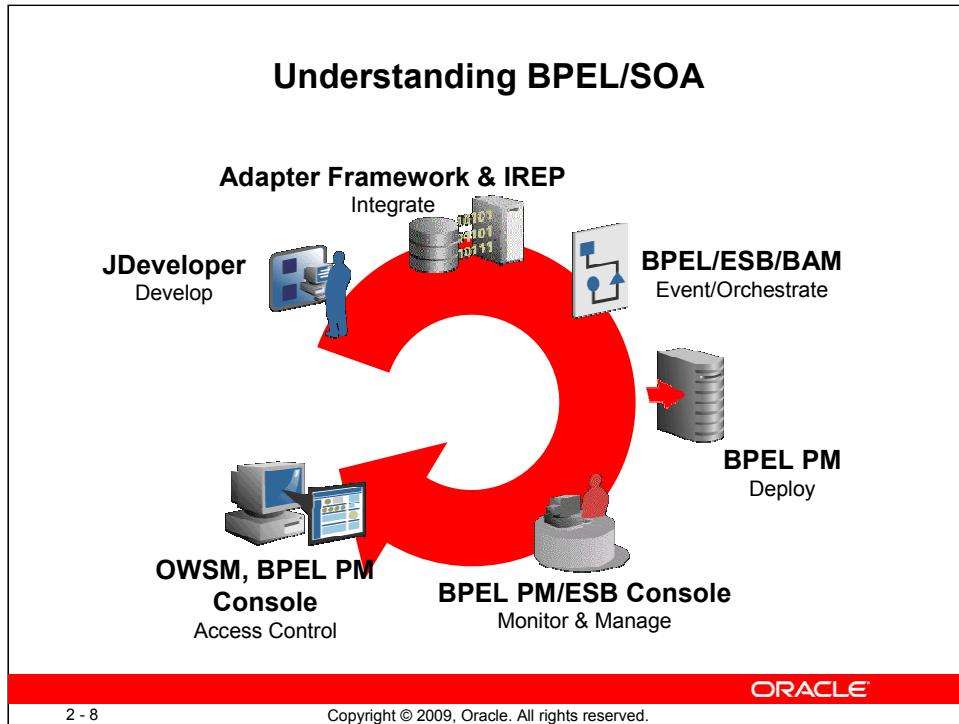
```
private void logMessage(String str, String lDebugMode)
{
    \\ Write you program logic here
}
private String getProfileValue(String profile)
{
    \\ Write you program logic here
}
public void openConection()
{
    \\ Write you program logic here
}
public void runProgram(CpContext pCpContext)
{
    \\ Write you program logic here
}
```

ORACLE

2 - 7

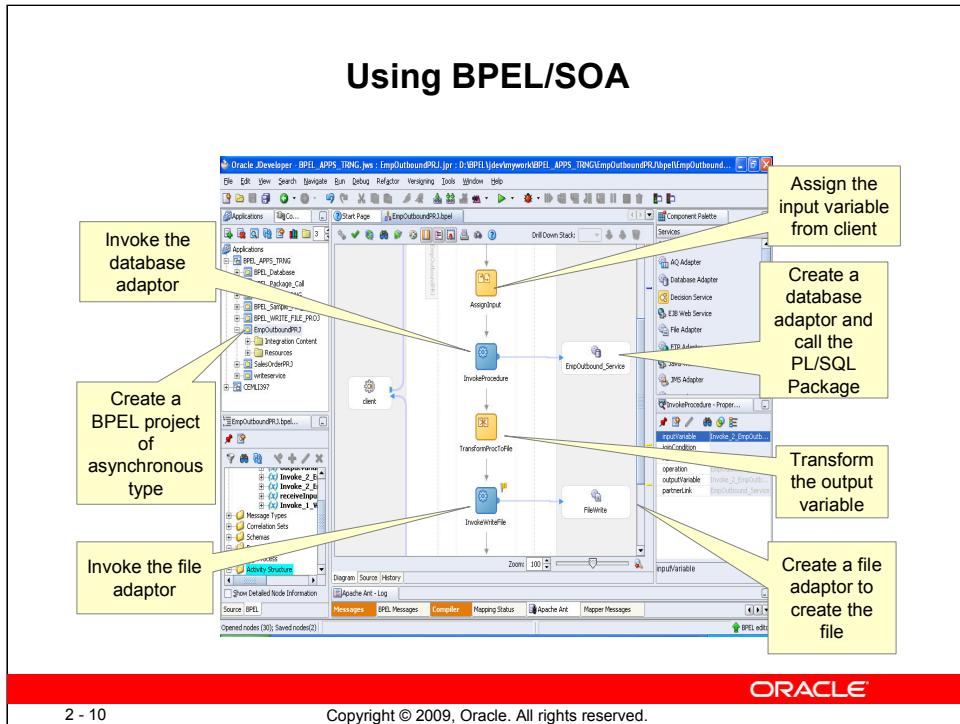
Copyright © 2009, Oracle. All rights reserved.

- **logMessage**
 - This function will be used to logged the error message or debug messages.
- **getProfileValue**
 - This function will be used when we required to retrieve profile options values.
- **openConection**
 - This function is used to open the database connections.
- **runProgram**
 - This function is used to open the run the concurrent program for loading the source data into staging tables.



- **JDeveloper:** JDeveloper provide the facilities to invoke the BPEL/SOA process and integration with others RDBMS system.
- **Adapter Framework & IREP:** It provides Bi-directional connectivity and Broad range of connectivity with other RDBMS systems.
- **BPEL/ESB/BAM:** A standards-based product that connects existing IT systems and business partners as a set of services, Delivers real-time insight into business operations
- **BPEL PM:**
 - Oracle BPEL Process Manager enables business processes to be modeled, automated, and monitored.
 - BPEL Process Manager includes a native BPEL (Business Process Execution Language) engine that executes the processes.
 - Oracle BPEL Process Manager provides a comprehensive, standards-based and easy to use solution for creating, deploying and managing cross-application business processes like interface and conversions.

- **BPEL PM/ESB Console:**
 - **BPEL Process Manager(BPEL PM):**
 - Monitoring statistics and performance of processes
 - Managing and administering processes
 - Enabling state of long running process to be automatically maintained in database
 - Fault & Exception handling management
 - Debugging BPEL Processes
 - Managing and creating domains
 - **ESB Console**
 - It consumes Business services from back-end apps
 - Eliminate point to point coupling using SOA
 - Monitoring and reporting of services
- **OWSM, BPEL PM Console:** BPEL Process Manager is used for access control while integrating the system which may be of different RDBMS.



- Creating a BPEL process for outbound interface

Summary

After completing this lesson, you should be able to understand

- Writing Interfaces
 - Using PI/SQL
 - Using JAVA
 - Using BPEL/SOA

ORACLE

2 - 11

Copyright © 2009, Oracle. All rights reserved.

Types of Interfaces / Integration

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

After completing this lesson, you should be able to understand the following

- Type of Interfaces
- Type of Integration
- Technology used to develop interfaces and integration

ORACLE

3 - 2

Copyright © 2009, Oracle. All rights reserved.

Interfaces / Integration

- PL/SQL APIs
- Open Interface Tables
- Concurrent Programs
- XML Gateway
- eCommerce Gateway

ORACLE

3 - 3

Copyright © 2009, Oracle. All rights reserved.

- **API's:** An application program interface (API) is a set of exposed data structures and functions that an application can use to invoke services on a component. Oracle Application APIs are used to process the input data and create\update\delete the records present in the Oracle base tables. Oracle Applications provides more than 2650 public API's for Interface\Conversion
- **Open Interface Table:** Open Interface Tables are the intermediate staging tables when the data is inserted / updated for source data. The data from these tables is populated into the standard Oracle Applications tables using the standard import concurrent programs. more than 147 Open interface tables are available for migrating the data into Oracle Apps.
- **Concurrent program:** A program with the associated parameters used to perform the business logic and import the data into base tables and generating reports. Execution file can be an operating system file or database stored procedure which contains your application logic (Example: SQL* Stored procedure, Java). Oracle Applications provide more than 214 standard import concurrent programs for importing data into base tables.

- **XML Gateway:** XML Gateway provides a generic framework to send information as an outbound transaction or to receive the information as an inbound transaction between different RDBMS from / into the Oracle e-Business Suite, There are more than 170 XML Gateway provided by Oracle Applications.
- **eCommerce Gateway:** eCommerce Gateway is used to expose internal business objects in a standardized format for the trading partners to integrate with the external business objects, to send/receive information to/from Oracle EBusiness Suite. There are more than 24 eCommerce Gateway provided by Oracle Applications.

Types of Interfaces / Integration

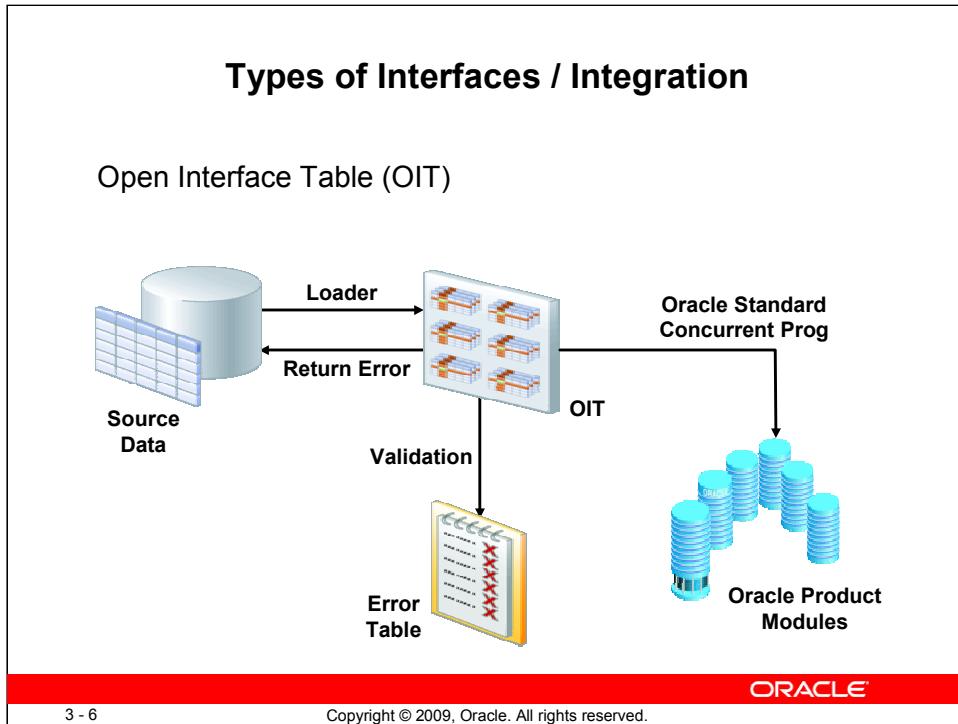
- Open Interface Table (OIT)
- API's
- BPEL/SOA

ORACLE®

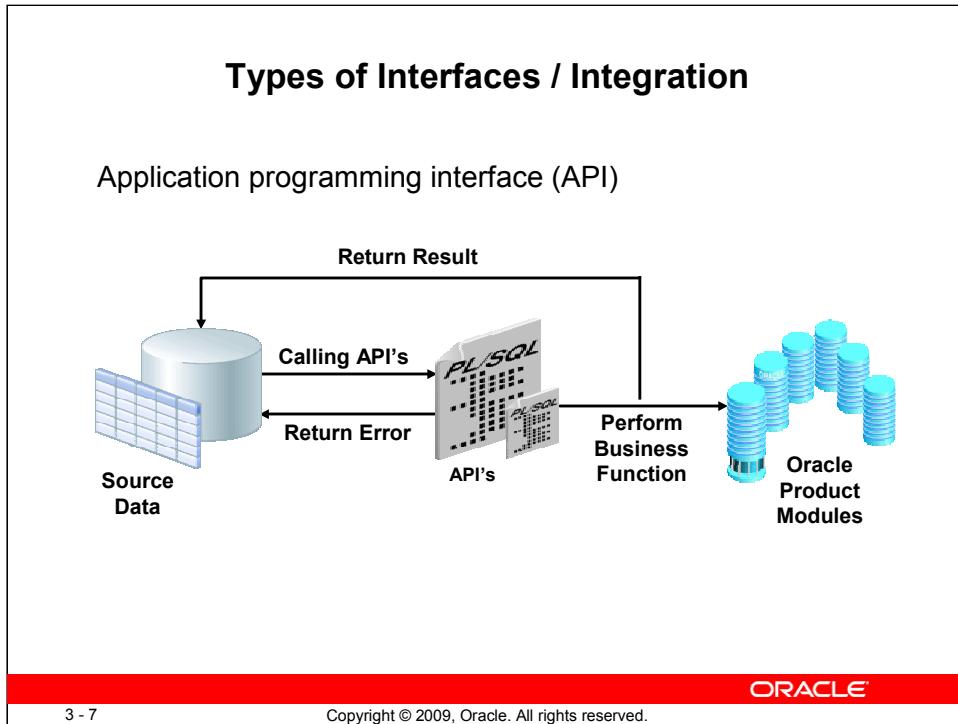
3 - 5

Copyright © 2009, Oracle. All rights reserved.

- Interfaces or Integration are technology which are used to migrate the source data either into the Oracle Apps product or from the Oracle Apps product to external system.



- Open Interface Table (OIT) are widely used for loading the data from source system to the target system.



- The public API's provided by the Oracle Applications can also be used for loading the records present in the source system to the target system.
- A major benefit associated with the public APIs is they directly create\update\delete the records in the Oracle Applications base tables and no concurrent program is called for this purpose which improves the efficiency of the program

- The public APIs provided by Oracle Applications will have the following important parameters :-

1. P_API_VERSION =>
PUBLIC_API_NAME.API_VERSION

API version is defined in specification part of public API, we use the same value for parameter while calling the public API

2. P_COMMIT => FND_API.G_TRUE

If P_COMMIT parameter is set as FND_API.G_TRUE then after completion of each process by API it will commit the transaction

3. P_RECORD_TYPE_VAL => RECORD_TYPE_VAL

Either Record Type or Table Type variables are used to pass the interface data to the public API's for processing.

4. X_RETURN_STATUS => lc_return_status

5. X_MSG_COUNT => ln_msg_count

6. X_MSG_DATA => lc_error_message

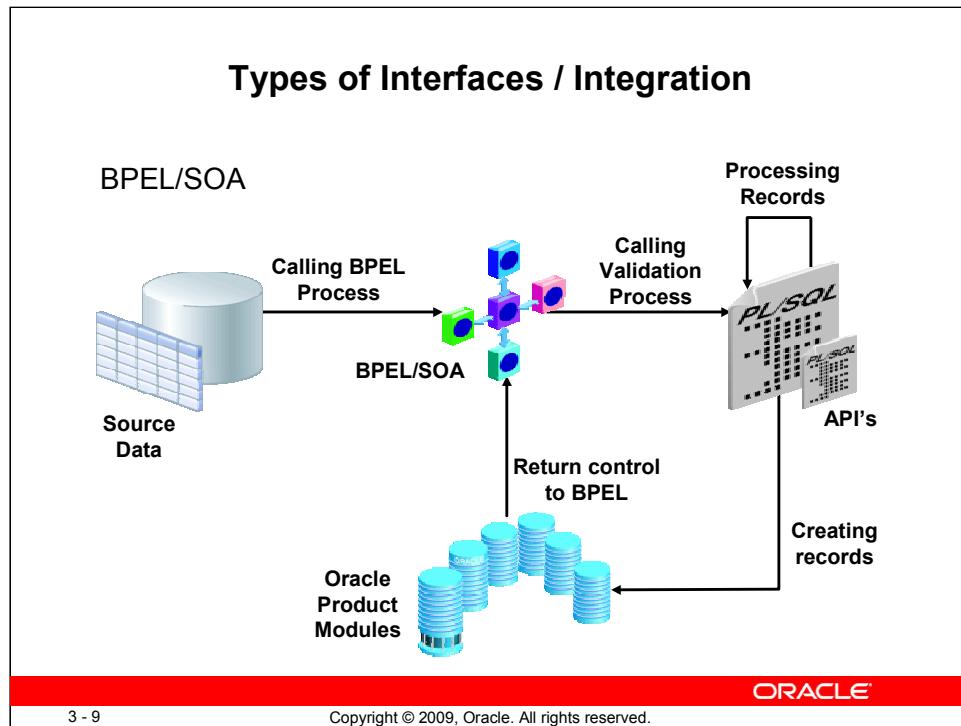
There should be at least 3 out parameters for each public API to return the status (success, error), number of error message and error message if any.

- Use FND_MSG_PUB.INITIALIZE to initialize the message buffer before calling the public API's
- If API's return status is E (Error) then use the below code to retrieve the error message.

```

IF lc_return_status <> 'S' THEN
  IF ln_msg_count <= 1 THEN
    lc_errbuf := lc_error_message;
  ELSE
    FOR ln_i IN 1 .. ln_msg_count LOOP
      lc_errbuf := lc_errbuf || FND_MSG_PUB.GET (
        ln_i, 'F');
    END LOOP;
  END IF;

```



- BPEL stands for Business Process Execution Language and SOA stands for Service Oriented Architecture.
- BPEL/SOA can be used for integration with the external system.
- Follows industry standards, web services, and business process design to orchestrate integration development.
- Similar to EAI.
- Provides dashboard for server maintenance and monitoring.

Summary

After completing this lesson, you should be able to understand

- Type of Interfaces
- Type of Integration
- Technology used to develop interfaces and integration
 - Open Interface Table (OIT)
 - Application Programming Interfaces (APIs)
 - Service Oriented Architecture (SOA)



Open Interfaces



ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

After completing this lesson, you should understand the following

- Various Interface Options
- Open Interface Model
- API Diagram
- Purchasing Open Interfaces
- Payables Open Interfaces
- Order Management Open Interfaces
- Receivables Open Interfaces
- Designing, Developing, and Performing Conversion Process
- Development standards.

ORACLE

4 - 2

Copyright © 2009, Oracle. All rights reserved.

Interface Options

1. Open Interface Table (OIT)
2. Application Programming Interface (API)
3. Enterprise Application Integration Tools (EAII) / Middleware Tools
4. Electronic Data Interchange (EDI)
5. BPEL Process Manager
6. Other Oracle Technology
 - PL/SQL
 - Stored Procedures
 - Oracle Advanced Queuing (AQ)

ORACLE

4 - 3

Copyright © 2009, Oracle. All rights reserved.

- **Open Interface Table (OIT):** Open Interface Table (OIT) are widely used for loading the data from source system to the target system. Oracle Applications provides open interface tables with each business module which are used for loading , validation and error reporting of the source data.
- **Application Programming Interface (API's):** The public API's provided by the Oracle Applications can also be used for loading the records present in the source system to the target system.
- **Enterprise Application Integration Tools (EAII) / Middleware Tools:** When there is no standard Oracle process (no interface tables) as well no API's to import data.In that case, you have to write a program that inserts data directly into the Oracle Base Tables.Though this is not recommended by oracle as it any changes from oracle standard functionality may malfunction during patch or upgrade version. So there is another options where we can use some EAI or middle ware tool. EAI/Middle ware tools typically works as adaptor or connector which then take care of any insert /update in oracle base table without any risk.There are number of adapter available to integrate with oracle application. These tools normally provide robust visual development GUI as well as have strong functionality to provide integration monitoring and alerting tools.

- **Electronic Data Interchange (EDI):** EDI (Electronic Data Interchange) uses industry standard data definitions (US/ANSI/X.12) for transmission of documents such as PO's, Invoices, Sales Order etc. Oracle provides some EDI transactions through EDI Gateway.
- **BPEL Process Manager:** BPEL uses industry standards, web services, and business process design to orchestrate integration development. It is typically as similar to EAI. It provides Dashboard for server maintenance and monitoring and moreover this is exposes Oracle APIs as web services.
- **Other Oracle Technology:** By means of making a either PL/SQL or stored procedure or hybrid process.
 - **PL / SQL**
 - Oracle Corporation's proprietary server-based procedural extension to the SQL database language
 - Relatively sophisticated procedural programming language
 - Compiled to DB; allows for syntax checking and high-performance DB integration
 - Myriad of purposes in Oracle applications: application logic, integration logic, exception handling
 - Exposes an API layer to Oracle application logic
 - **Stored Procedures**
 - Can be implemented as server-side Java
 - Capable of calling PL/SQL using JDBC; capable of being called by PL/SQL
 - Higher computational performance than PL/SQL
 - Allows for use of an industry standard technology instead of proprietary PL/SQL
 - Often coupled with PL/SQL as a hybrid approach to Oracle application integration
 - **Oracle Advanced Queuing (AQ)**
 - Database Integrated Messaging
 - Integrated data, message, and message history
 - Recoverability(any point in time recovery)
 - High Performance
 - Integrated with MQSeries via Oracle Messaging Gateway

Open Interface Overview

- Open Interfaces are used to load the source data into the Oracle Applications using the associated standard import programs.
- The Import programs have in built business logic to check the correctness of the data present in the open interface tables

ORACLE

4 - 5

Copyright © 2009, Oracle. All rights reserved.

For example:-

- You can import accounting data from the feeder systems.
- You can also import historical data from your previous accounting or management systems into Oracle Application systems using Open Interface table and the associated Import program.

Open Interface Overview

Some of the open interfaces provided by Oracle Applications:-

Oracle General Ledger

- Budget Upload
- Importing Journals
- Loading Daily Rates

Oracle Payables

- Credit Card Transaction Interface Table
- Invoice Import Interface
- Payables Open Interface
- Purchase Order Matching

ORACLE®

4 - 6

Copyright © 2009, Oracle. All rights reserved.

Open Interface Overview

Oracle Purchasing

- Requisitions Open Interface
- Purchasing Documents Open Interface
- Receiving Open Interface

Oracle Receivables

- AutoInvoice
- AutoLockbox
- Customer Interface
- Sales Tax Rate Interface
- Tax Vendor Extension

ORACLE®

4 - 7

Copyright © 2009, Oracle. All rights reserved.

Advantages of Open Interfaces

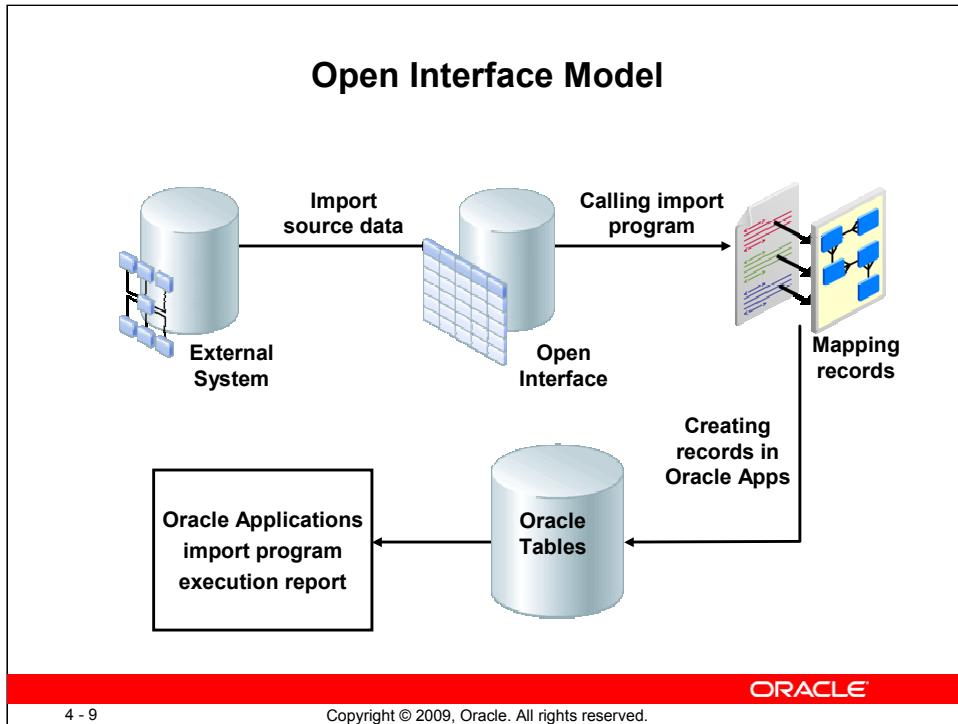
- Automatically convert significant data
- Import from wide range of environments
- Import historical data to keep the records up-to-date
- Review the imported data
- Correct the invalid data
- Validate the reliability of the data
- Archive the source data

ORACLE®

4 - 8

Copyright © 2009, Oracle. All rights reserved.

- Import information in the easiest way possible and automatically convert the valid data you import into data that is meaningful to you and your organization.
- Import information from a variety of environments, including your own and other accounting systems.
- Import historical data from your previous accounting, sales order or other management systems to keep your records consistent and up-to-date.
- Review the results of your import run. Identify which data has been successfully imported, and any errors which may have occurred during the import process.
- Correct invalid data online which was reported by the Standard Import program.
- Validate the integrity of any data before introducing it into your Oracle application.
- Choose to optionally archive your source data each time you run an import program.



- The goal of import program is to convert data from open interface into a standard data format that Oracle applications can read and then convert for further modification or processing in Oracle application.
- The type of environment from which you want to import data determines the type of open interface program you need to write. For example, you can use SQL*Loader to write an import program to feed data from a non-Oracle system. Or, you can write a open interface program to import historical data from your previous accounting system. Regardless of the type of open interface program you write, the output should be in standard data format that an Oracle Applications import program can use to convert your import data into your Oracle Applications system.
- You need to choose a tool for writing a open interface program to extract data from existing application system's printed reports, flat file, relational database, or other repository of application information. Using a open interface program, you populate an Oracle Applications import table with the information you want to introduce to your Oracle Financials system.
- SQL*Loader is a powerful and easy-to-use tool, to write a feeder program. SQL*Loader lets you map elements of a regularly formatted file, such as a listing or flat file, and specify which columns of which tables to populate.

Open Interface Processing

- Oracle Applications provide both inbound and outbound interfaces.

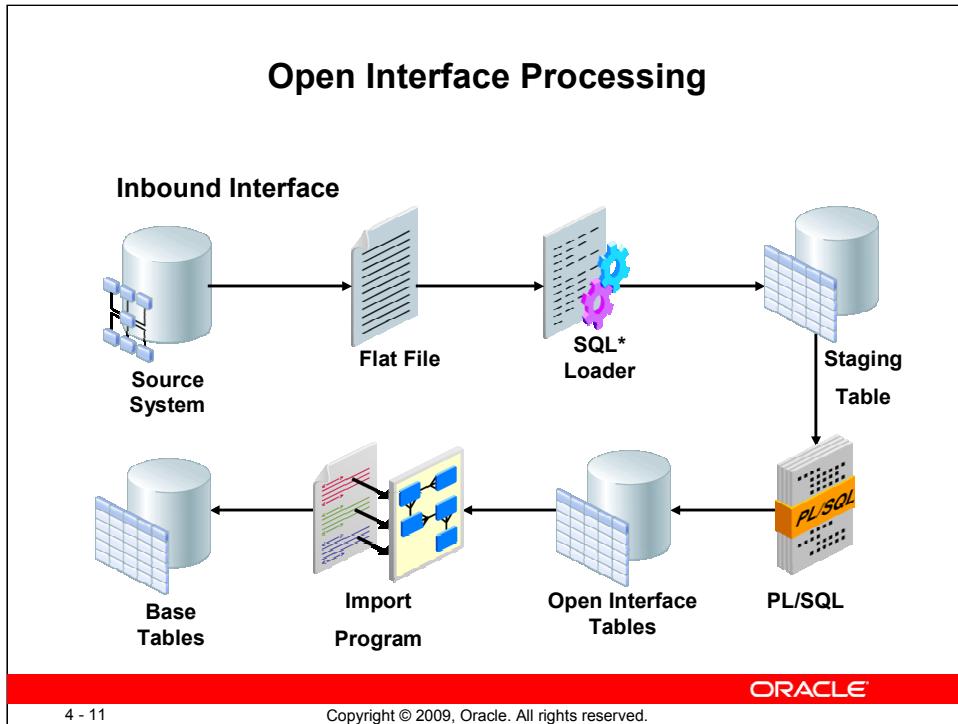
Outbound Interface

The diagram illustrates the Outbound Interface process. It starts with a **Source System** (represented by a cylinder with a grid icon) which feeds into a series of **PL/SQL Processes** (represented by a gear icon). These processes then output to a **Data File** (represented by a cylinder with a grid icon). The **Data File** is then transferred via an **FTP Utility** (represented by a computer monitor icon) to a **Target System** (represented by a cylinder with a grid icon). The data file contains binary data: 010010110010, 010010110011, and 010010110010.

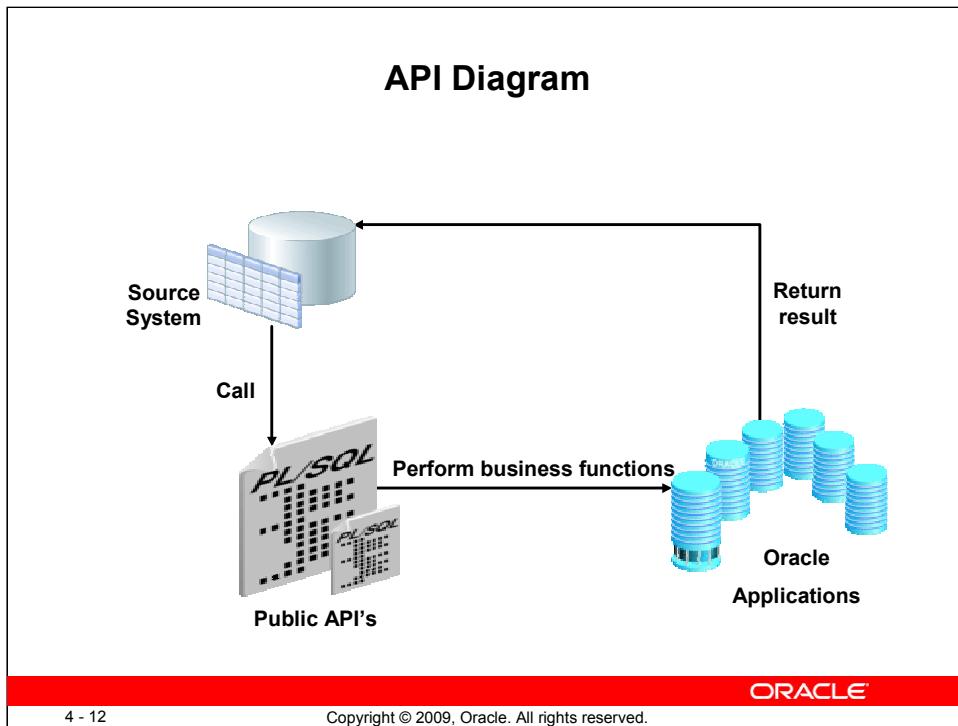
ORACLE

4 - 10 Copyright © 2009, Oracle. All rights reserved.

1. The PL/SQL program will extract and process the data from source system and will generate a data file in the UTL_FILE_DIR path or generate the data as concurrent program output or log file.
2. Based on the client requirement transfer the file in the target system using any FTP utility.



- Data file is extracted from the source system into a flat file.
- Using SQL* Loader load the data from flat file into the staging table.
- When all the records are loaded successfully, execute PL/SQL program to validate the records present in staging table and populate them in the interface table.
- Call the standard open interface program or the corresponding API to import the data into the Oracle Applications base tables.



- **Source System:** Provides the data for processing to the Public APIs
- **Public API's:** Used for populating the records in base tables, while processing the public APIs perform following types of validations :-
- **Control Columns:** Control columns track the status (status column for new or processed records) of each row in the interface table as it is inserted, validated, errored, processed, and ultimately deleted. Additional control columns identify who last updated the row and the last update date.

- **Data Columns:** Data columns store the specific attributes that the source application is sending to the Oracle Applications product. For example, transaction quantity is one attribute of a move transaction.
 - **Required Columns:** Required columns store the minimum information needed by the Oracle Applications product to successfully process the interface row. For example, organization code is required for all transactions in Oracle Apps.
 - **Derived Columns:** Derived columns are created by the destination product from information in the required columns.
 - **Optional Columns:** Optional columns are not necessarily required by Oracle Applications products but can be used for additional value-added functionality.
- **Oracle Application:** The processed records are populated in base tables.
- Most of the Oracle Applications public API's have common parameters to be passed, they are
 1. P_API_VERSION => PUBLIC_API_NAME.PROCESS_NAME.API_VERSION
API version is defined in specification part of public API, we use the same value for parameter while calling the public API
 2. P_COMMIT => FND_API.G_TRUE
If P_COMMIT parameter is set as FND_API.G_TRUE then after completion of each process by API it will commit the transaction
 3. P_RECORD_TYPE_VAL => RECORD_TYPE_VAL
Either Record Type or Table Type variables are used to pass the interface data to the public API's for processing.
 4. X_RETURN_STATUS => lc_return_status
 5. X_MSG_COUNT => ln_msg_count
 6. X_MSG_DATA => lc_error_message

There should be at least 3 out parameters for each public API to return the status (success, error), number of error message and error message if any.

- Use FND_MSG_PUB.INITIALIZE before calling the public API's
- If API's return status is E (Error) then use the below code to retrieve the error message.

```
IF lc_return_status <> 'S' THEN
    IF ln_msg_count <= 1 THEN
        lc_errbuf := lc_msg_data;
    ELSE
        FOR ln_i IN 1 .. ln_msg_count LOOP
            lc_errbuf := lc_errbuf || FND_MSG_PUB.GET
            (ln_i, 'F');
        END LOOP;
    END IF;
```

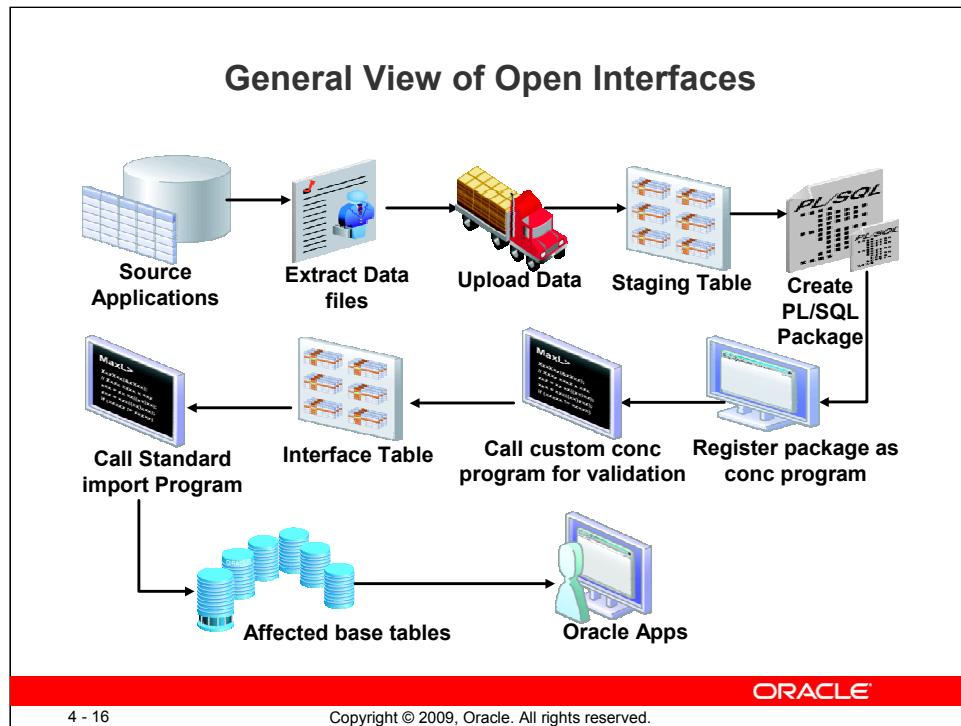
Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- Purchasing Open Interfaces
- Order Management Open Interfaces
- Designing the Conversion Process
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 15

Copyright © 2009, Oracle. All rights reserved.



This open interface example will show how to migrate the transactions data from source system to Oracle Apps module.

- **Interface Approach / Reference Documents**
 - Interface Tables / Interfaces User Guide
- **Responsibility / Navigation / Module**
 - Interfaces Responsibility, System Administrator.
- **Application Setup Requirements**
 - Register the concurrent program for Loader and Package
 - Add the concurrent program in specified responsibility
- **Extract data file in comma separated value(CSV) format from source system.**
- **Create a loader program for loading the data file into staging(custom) table**
 - Create a control file which will have command and mapping column for loading the data from the source data file to staging table.
 - Register the control file in Oracle Applications as concurrent program as per the Application setup requirement.
 - Call the custom concurrent program to load the data file from specified location to staging table.

- **Custom concurrent program for validation**
 - Call custom concurrent program for validation of data in staging table
 - When all the records are validated successfully then load the data into standard interface table
 - Call the standard import program to process the records available in standard interface table
- **Assumptions / Prerequisites / Validations**
 - Assumption: All Setup, GL code combinations should be created.
 - Required Columns validations through standard import program
- **Base Tables Affected**
 - Oracle Applications base tables
- **Interface Tables / APIs**
 - Open Interface Tables for All Modules

Agenda

- General View of Open Interfaces
- **Payables Open Interfaces**
- Receivables Open Interfaces
- Purchasing Open Interfaces
- Order Management Open Interfaces
- Designing the Conversion Process
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 18

Copyright © 2009, Oracle. All rights reserved.

Payables Open Interfaces

Using Payables Open Interface you can do the following

- Import standard Invoices, Credit Memo, Expense Report etc.
- Replace or Update standard Invoices, Credit Memo, Expense Report .

ORACLE

4 - 19

Copyright © 2009, Oracle. All rights reserved.

Payables Invoice open interface process:

1. Launch the Payables Open Interface Import program to validate the records present in the Payables open interface table.
2. Check AP_INTERFACE_REJECTIONS table for any validation error in the Payables Open Interface tables
3. After successful validation, the processed records are populated into the Payables base tables.

Payables Open Interface Table

Tables: AP_INVOICES_INTERFACE

COLUMN NAME	DESCRIPTION
INVOICE_ID	Interface header unique identifier
INVOICE_NUM	AP vendor's unique invoice number
INVOICE_TYPE_LOOKUP_CODE	Invoice type category
TERM_ID	uniquely identify term name from setup
CURRENCY_CODE	Transaction currency code
VENDOR_ID	Supplier unique identifier
VENDOR_SITE_ID	Supplier site unique identifier
INVOICE_AMOUNT	Vendor invoice amount

ORACLE

4 - 20

Copyright © 2009, Oracle. All rights reserved.

AP_INVOICES_INTERFACE is the interface table that holds the header information for the Payables Open Interface Import program to create standard, Credit Memo, Debit Memo etc.

The Payables Open Interface Import program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the AP_INTERFACE_REJECTIONS table.

Payables Open Interface Table

Tables: AP_INVOICE_LINES_INTERFACE

COLUMN NAME	DESCRIPTION
INVOICE_ID	Interface transaction source
INVOICE_LINE_ID	Requisition destination type
LINE_NUMBER	Preparer unique identifier
LINE_TYPE	Quantity ordered
AMOUNT	Item unit price
DESCRIPTION	Inventory Item unique identifier
CODE_COMBINATION_ID	Line Type unique identifier

ORACLE

4 - 21

Copyright © 2009, Oracle. All rights reserved.

AP_INVOICE_LINES_INTERFACE is the interface table that holds the lines information for the Payables Open Interface Import program to create lines for invoices like standard, Credit Memo, Debit Memo etc.

The Payables Open Interface Import program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the AP_INTERFACE_REJECTIONS table.

- **Custom concurrent program for validation**
 - Call custom concurrent program for validation of data in staging table
 - When all the records are validated successfully then load the data into standard interface table
 - Call the standard import program to process the records available in standard interface table
- **Assumptions / Prerequisites / Validations**
 - Assumption: Vendors Setup, GL code combinations, Purchase Order should be created.
 - Required Columns validations through standard import program
 - Organization id (org_id)
 - Vendor name (vendor_name)
 - Vendor site code (vendor_site_code)
 - An terms (terms_name)
 - AWT group name
 - Invoice number (invoice_num)
 - Invoice type lookup code
 - Accts pay code combination (accts_pay_code_combination_id)
 - Payment method lookup code
 - Code combination id
 - Item number
 - Invoice id
 - Line type lookup code
 - Tax code
 - PO Distributions
- **Base Tables Affected**
 - AP_INVOICES_ALL
 - AP_INVOICES_DISTRIBUTIONS_ALL
 - Error Table: AP_INVOICE_REJECTIONS
- **Interface Tables / APIs**
 - AP_INVOICES_INTERFACE
 - AP_INVOICE_LINES_INTERFACE
 - AP_INVOICE_DISTRIBUTIONS_INTERFACE

- **Navigation of Import Program & Parameters**

Others => Request => Run => ‘Payables Open Interface Import’

Parameters

1 => Source

2 => Batch Name

- **Import Errors**

- For finding the errored records, we use

AP_INTERFACE_REJECTIONS table and the column we will use to join with AP_INVOICE_LINES_INTERFACE is INVOICE_ID

Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- **Receivables Open Interfaces**
- Purchasing Open Interfaces
- Order Management Open Interfaces
- Designing the Conversion Process
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 24

Copyright © 2009, Oracle. All rights reserved.

Receivables Open Interfaces

Using Receivables Open Interface you can do the following

- Import Standard Invoices, Credit Memo, Debit Memo.
- Replace or Update Standard Invoices, Credit Memo, Debit Memo.

ORACLE

4 - 25

Copyright © 2009, Oracle. All rights reserved.

Purchasing document open interface process:

1. Launch the AutoInvoice Import program to validate the records present in the Receivables open interface table.
2. If there are any validation errors, AR_TRX_ERROR_GT table is populated.
3. After successful validation, the processed records are populated into the Receivables base tables.

Receivables Open Interface Table

Tables: RA_INTERFACE_LINES_ALL

COLUMN NAME	DESCRIPTION
BATCH_SOURCE_NAME	Batch source name identified the records in interface table
LINE_TYPE	Transactions line type
DESCRIPTION	Inventory item descriptions
CURRENCY_CODE	Currency code of Transactions
CONVERSION_TYPE	Currency conversion type
TERM_ID	Payment term ID for the Transactions
CUST_TRX_TYPE_ID	Transactions type ID
ORIG_SYSTEM_BILL_CUSTOMER_ID	Customer Account ID
ORIG_SYSTEM_BILL_ADDRESS_ID	Customer Account bill to site ID

ORACLE

4 - 26

Copyright © 2009, Oracle. All rights reserved.

RA_INTERFACE_LINES_ALL is the interface table that holds the header and lines information for the AutoInvoice Import program to create AR Transactions.

The AutoInvoice Import program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the AR_TRX_ERROR_GT table.

- **Custom concurrent program for validation**
 - Call custom concurrent program for validation of data in staging table
 - When all the records are validated successfully then load the data into standard interface table
 - Call the standard import program to process the records available in standard interface table
- **Assumptions / Prerequisites / Validations**
 - Assumption: Customer Setup, GL code combinations, Sales Order should be created.
 - Required Columns validations through standard import program
 - Organization id (org_id)
 - Customer name (customer_name)
 - Customer site code (customer_site_code)
 - AR terms (terms_name)
 - Tax name
 - Transaction number (trx_num)
 - Invoice type
 - Receivables code combination (accts_pay_code_combination_id)
 - Payment Term
 - Code combination id
 - Inventory Item code
 - AR Trx id
 - Line type
 - Tax code
 - Sales Order Number
- **Base Tables Affected**
 - RA_CUSTOMER_TRX_ALL
- **Interface Tables / APIs**
 - RA_INTERFACE_LINES_ALL

- **Navigation of Import Program & Parameters**

Others => Request => Run => ‘Autoinvoice Master Program’

Parameters

1 => Number of Instances

2 => Organization

3 => Invoice Source

4 => Default Date

5 => Base Due Date on Trx Date

- **Import Errors**

- For finding the errored records, we use

AR_TRX_ERRORS_GT table and the column we will use to
join with RA_INTERFACE_LINES_ALL is
TRX_HEADER_ID

Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- **Purchasing Open Interfaces**
- Order Management Open Interfaces
- Designing the Conversion Process
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 29

Copyright © 2009, Oracle. All rights reserved.

Purchasing Open Interfaces

Using Purchasing Open Interface you can do the following

- Import standard purchase orders, blanket purchase agreements or catalog quotations.
- Replace or Update Blanket purchase agreements and quotations. Standard purchase orders can only be imported as new documents.

ORACLE

4 - 30

Copyright © 2009, Oracle. All rights reserved.

Purchasing document open interface process:

1. Launch the Purchasing Documents Open Interface program, to validate the records present in the Purchasing open interface table.
2. If there are any validation errors, PO_INTERFACE_ERRORS table is populated.
3. After successful validation, the processed records are populated in the purchasing base tables.

Additional Information:

1. Launch Purge Purchasing Documents Open Interface Processed Data program to purge PO interface data. This program removes data that has been accepted or rejected, not data that is still pending.
2. Launch Purchasing Interface Errors Report to view errors. If you want to purge data from the Purchasing Interface Errors table, set the Purge Data field to Yes.
3. Refer metalink note [Doc ID: 252499.1](#) for mandatory column information on Purchasing Interfaces.

Purchasing Open Interface Table

Tables: PO_HEADERS_INTERFACE

COLUMN NAME	DESCRIPTION
INTERFACE_HEADER_ID	Interface header unique identifier
PROCESS_CODE	Interface record status
ACTION	Interface action: ADD, NEW
DOCUMENT_TYPE_CODE	Document type to be created: PO or RFQ
BATCH_ID	Batch unique identifier
VENDOR_ID	Supplier unique identifier
VENDOR_SITE_ID	Supplier site unique identifier
SHIP_TO_LOCATION_ID	Ship-to location unique identifier
BILL_TO_LOCATION_ID	Bill-to location unique identifier

ORACLE

4 - 31

Copyright © 2009, Oracle. All rights reserved.

PO_HEADERS_INTERFACE is the interface table that holds the header information for the PDOI program to create standard/blanket purchase orders and catalog quotations.

The PDOI program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the PO_INTERFACE_ERRORS table.

Purchasing Open Interface Table

Tables: PO_LINES_INTERFACE

COLUMN NAME	DESCRIPTION
INTERFACE_LINE_ID	Interface transaction source
INTERFACE_HEADER_ID	Requisition destination type
ITEM_ID	Preparer unique identifier
QUANTITY	Quantity ordered
UNIT_PRICE	Item unit price
ITEM_ID	Inventory Item unique identifier
LINE_TYPE	Line Type unique identifier
NEED_BY_DATE	Item need by date
UNIT_OF_MEASURE	Unit of Measure

ORACLE

4 - 32

Copyright © 2009, Oracle. All rights reserved.

PO_LINES_INTERFACE is the interface table that holds the lines information for the PDOI program to create standard/blanket purchase orders and catalog quotations.

The PDOI program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the PO_INTERFACE_ERRORS table.

Purchasing Open Interface Table

Tables: PO_DISTRIBUTIONS_INTERFACE

COLUMN NAME	DESCRIPTION
INTERFACE_HEADER_ID	Interface header unique identifier
INTERFACE_LINE_ID	Interface line unique identifier
INTERFACE_DISTRIBUTION_ID	Interface distribution unique identifier
SET_OF_BOOKS_ID	Set of Books unique identifier
CHARGE_ACCOUNT_ID	Unique identifier for the General Ledger charge account
BUDGET_ACCOUNT_ID	Unique identifier for the General Ledger budget account
ACCRUAL_ACCOUNT_ID	Unique identifier for the General Ledger accrual account
VARIANCE_ACCOUNT_ID	Unique identifier for the General Ledger variance account

ORACLE

4 - 33

Copyright © 2009, Oracle. All rights reserved.

PO_DISTRIBUTIONS_INTERFACE is the interface table that holds the distribution information for the PDOI program to create standard purchase order.

The PDOI program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the PO_INTERFACE_ERRORS table.

- **Custom concurrent program for validation**
 - Call custom concurrent program for validation of data in staging table
 - When all the records are validated successfully then load the data into standard interface table
 - Call the standard import program to process the records available in standard interface table
- **Assumptions / Prerequisites / Validations**
 - Assumption: Customer Setup, GL code combinations, Internal Requisitions should be created.
 - Required Columns validations through standard import program
 - Interface source code
 - Destination type code
 - Authorization status
 - Preparer id or preparer name
 - Quantity
 - Charge account id
 - Destination organization id or destination organization code
 - Deliver to location id or deliver to location code
 - Deliver to requestor id or deliver to requestor name
- **Base Tables Affected**
 - PO_REQ_DISTRIBUTIONS_ALL
 - PO_REQ_HEADERS_ALL
 - PO_REQ_LINES_ALL
- **Interface Tables / APIs**
 - PO_REQ_INTERFACE_ALL
- **Navigation of Import Program & Parameters**
 - Others => Request => Run => ‘Requisition Import’
 - Parameters
 - 1 => Group By
 - 2 => Multiple Distributions
 - 3 => Initiate Approval after ReqImport

- **Import Errors**

- For finding the errored records, we use PO_INTERFACE_ERRORS table and the column we will use to join with PO_REQUSITIONS_INTERFACE_ALL is INTERFACE_TRANSACTION_ID
- The PROCESS_FLAG column in the PO_REQUSITIONS_INTERFACE_ALL table is updated as ERROR

Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- Purchasing Open Interfaces
- **Order Management Open Interfaces**
- Designing the Conversion Process
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 36

Copyright © 2009, Oracle. All rights reserved.

Order Management Open Interfaces

Using Order Management Open Interface you can do the following

- Import standard sales order, internal sales order, sales returns,etc.
- Replace or Update standard sales order, internal sales order and sales returns

ORACLE

4 - 37

Copyright © 2009, Oracle. All rights reserved.

Order Management open interface process:

1. Launch the Order Import program to validate the records present in the Order Management open interface tables.
2. Check OE_UPGRADE_ERRORS table for any validation error in the Order Management open interface tables
3. After successful validation, the processed records are populated into the Order Management base tables.

Order Management Open Interface Table

Tables: OE_HEADERS_IFACE_ALL

COLUMN NAME	DESCRIPTION
HEADER_ID	Sales order header unique identifier
ORDER_NUMBER	Sales order number
PRICE_LIST_ID	Price list
ORDER_TYPE_ID	Transaction type
TRANSACTIONAL_CURR_CODE	Transaction currency code
PAYMENT_TERM_ID	Payment term
SALESREP_ID	Salesperson unique identifier

ORACLE

4 - 38

Copyright © 2009, Oracle. All rights reserved.

OE_HEADERS_IFACE_ALL is the interface table that holds the header information for the Order Import program to create standard, Credit Memo, Debit Memo etc.

The Order Import program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the OE_UPGRADE_ERRORS table.

Order Management Open Interface Table

Tables: OE_LINES_IFACE_ALL

COLUMN NAME	DESCRIPTION
HEADER_ID	Sales order header unique identifier
LINE_ID	Sales order line unique identifier
LINE_NUMBER	Line Number
LINE_TYPE_ID	Transaction Type
INVENTORY_ITEM_ID	Inventory Item unique identifier
ORDERED_QUANTITY	Quantity ordered
ORDER_QUANTITY_UOM	Unit of measure of the item ordered

ORACLE

4 - 39

Copyright © 2009, Oracle. All rights reserved.

OE_LINES_IFACE_ALL is the interface table that holds the lines information for the Order Import program to create lines for invoices like standard, Credit Memo, Debit Memo etc.

The Order Import program validates your data, derives or defaults additional information and writes an error message for every validation that fails into the OE_UPGRADE_ERRORS table.

- **Custom concurrent program for validation**
 - Call custom concurrent program for validation of data in staging table
 - When all the records are validated successfully then load the data into standard interface table
 - Call the standard import program to process the records available in standard interface table
- **Assumptions / Prerequisites / Validations**
 - Assumption: Customer Setup, GL code combinations, Sales Orders should be created.
 - Required Columns validations through standard import program
 - Transaction type
 - Sold to Org ID
 - Site use code
 - Amount
 - Unit Price
 - Pricelist
 - Ordered Quantity
 - UOM (Unit of Measure)
 - Inventory Item ID
- **Base Tables Affected**
 - OE_ORDER_HEADERS_ALL
 - OE_LINES_ALL
 - OE_LOT_SERIAL_NUMBERS
 - OE_SALES_CREDITS
- **Interface Tables / APIs**
 - OE_HEADERS_IFACE_ALL
 - OE_LINES_IFACE_ALL
 - OE_LOTSERIALS_IFACE_ALL
 - OE_CREDITS_IFACE_ALL
 - OE_ACTIONS_IFACE_ALL
 - OE_PRICE_ADJS_IFACE_ALL
 - OE_RESERVTNS_IFACE_ALL

- **Navigation of Import Program & Parameters**

Others => Request => Run => ‘Order Import’

Parameters

1 => Validate Only

2 => Validate Descriptive Flexfield

- **Import Errors**

- For finding the errored records, we use OE_UPGRADE_ERRORS table and the column we will use to join with OE_HEADERS_IFACE_ALL is HEADER_ID

- The ERROR_FLAG column in the OE_HEADERS_IFACE_ALL table is updated as Y

- **Ex:** For creating a sales order through open interface table mainly two interface tables are require

- OE_HEADERS_IFACE_ALL
 - OE_LINES_IFACE_ALL

Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- Purchasing Open Interfaces
- Order Management Open Interfaces
- **Designing the Conversion Process**
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 42

Copyright © 2009, Oracle. All rights reserved.

What is Conversion

- It is a process where existing data from the source system is extracted, cleansed, formatted and installed into a new system.
- Conversions can be manual or automated.
- It is a one-time process that requires extensive loading and preparation.
- Must be executed before a Oracle Applications goes into production.

ORACLE®

4 - 43

Copyright © 2009, Oracle. All rights reserved.

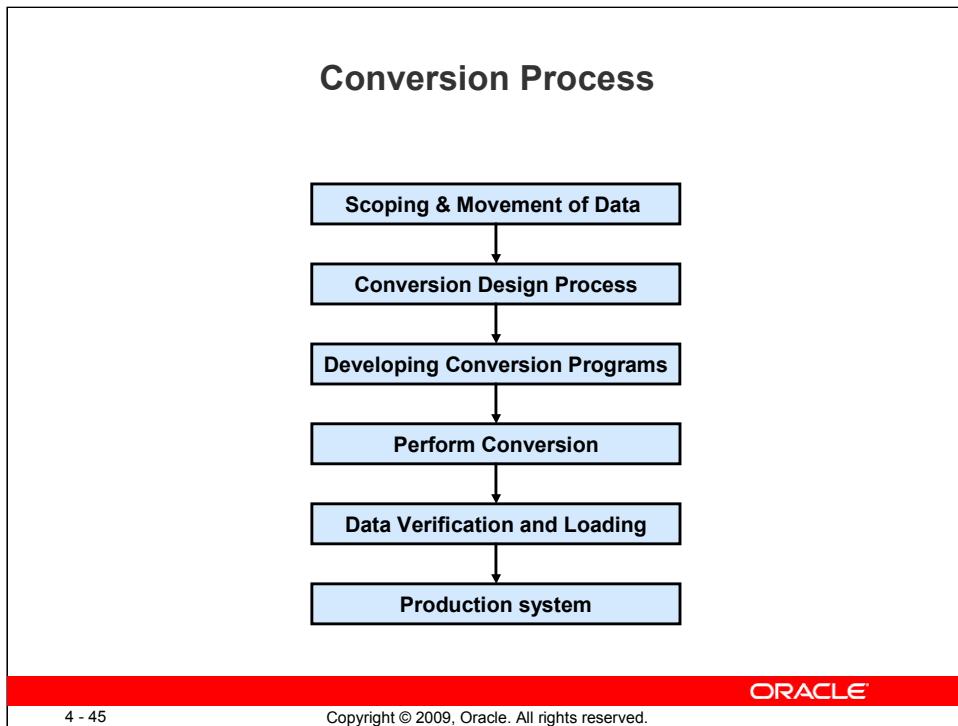
Interfaces & Conversion

	Interfaces	Conversions
Frequency	Periodical process	One-time process
Occurrence	Executed during production	Executed before production
Manner of execution	Batch or real time	Batch
Maintenance	Cost intensive	Cost effective

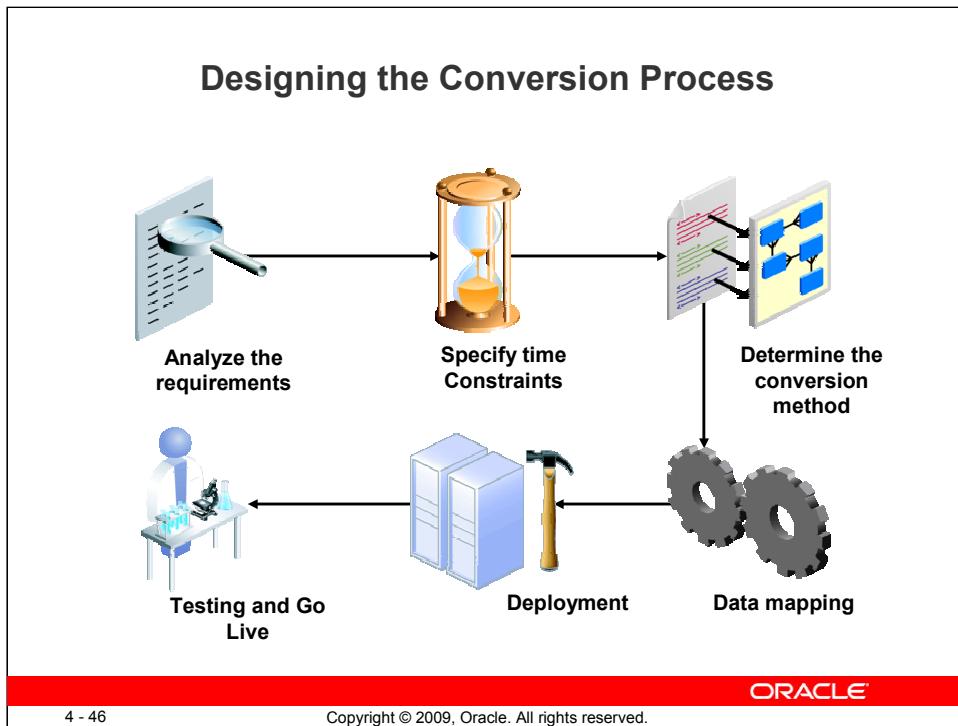
ORACLE®

4 - 44

Copyright © 2009, Oracle. All rights reserved.



1. Based on the requirements, identify the source data.
2. Design the conversion process by identifying the target tables and strategy of validating and loading the data.
3. Develop the conversion programs, for example
 1. Using SQL* Loader, develop a loader program.
 2. Develop a validation program to process the records present in the staging table and populate in the interface tables.
4. Run the conversion programs.
5. Verify the data in target tables or run the verification reports provided by Oracle Applications.
6. After successful conversion, the production system is ready to go live.



This is where you have to decide what need to be converted.

- Examine the business objectives and requirements to determine the data to be converted.
- Specify the time constraints for the conversion, especially for transaction data.
- Determine the appropriate conversion method. It is not recommended to go for manual entry. If data volume is low, try to find alternate product.
- Perform data mapping.
- Install all the hardware and software required for the conversion process.
- Determine the testing requirements. Identify testing method if available in Oracle else design a custom query to compare the result.

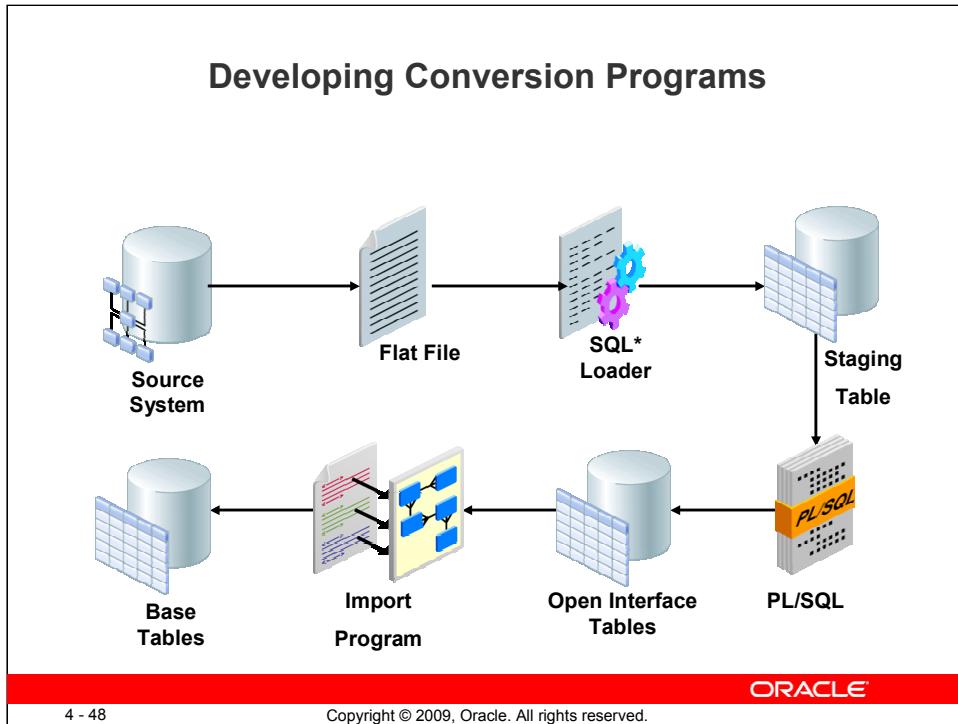
Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- Purchasing Open Interfaces
- Order Management Open Interfaces
- Designing the Conversion Process
- **Developing Conversion Programs**
- Performing the Conversion
- Development Standards

ORACLE®

4 - 47

Copyright © 2009, Oracle. All rights reserved.



- Data file is extracted from the source system into a flat file.
- Using SQL* Loader tool upload the data from flat file to staging table.
- When all the records are loaded successfully, execute PL/SQL program to validate the records present in staging table and populate them in the interface table.
- Call the standard open interface program or the corresponding API to import the data into the Oracle Applications base tables.

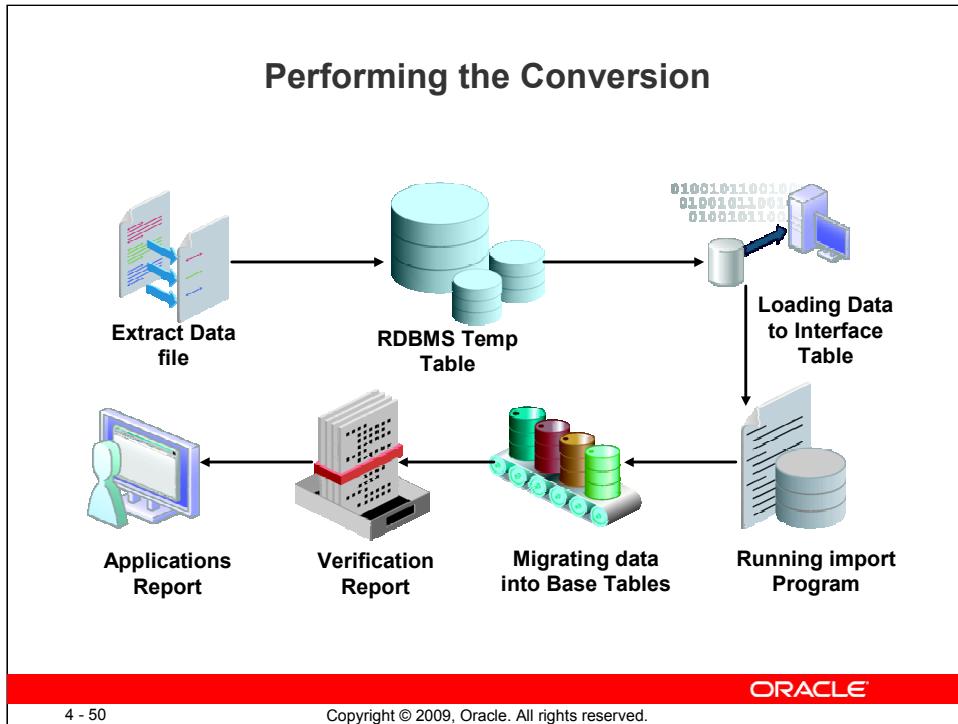
Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- Purchasing Open Interfaces
- Order Management Open Interfaces
- Designing the Conversion Process
- Developing Conversion Programs
- **Performing the Conversion**
- Development Standards

ORACLE®

4 - 49

Copyright © 2009, Oracle. All rights reserved.



This is process in which the major activity is performed.

- Extract and format data.
- Create temporary interface tables.
- Populate data into interface tables.
- Run import programs, which will migrate data into production tables.
- Verify the output of Import Program.
- Run application reports to verify converted data.

Agenda

- General View of Open Interfaces
- Payables Open Interfaces
- Receivables Open Interfaces
- Purchasing Open Interfaces
- Order Management Open Interfaces
- Designing the Conversion Process
- Developing Conversion Programs
- Performing the Conversion
- Development Standards

ORACLE®

4 - 51

Copyright © 2009, Oracle. All rights reserved.

Development Standards

- Coding standards to be followed for SQL and PL/SQL during implementation / customization / upgradation activities

ORACLE®

4 - 52

Copyright © 2009, Oracle. All rights reserved.

Development Standards

- Standard File Headers

```
-- +-----+
-- |          <Company Name>      |
-- |          <Location>        |
-- +-----+
-- | $Id:$           |
-- |Description:    |
-- |               |
-- |Change Record: |
-- |=====|
-- |Version  Date      Author Remarks   |
-- |=====  ====== ====== ======|
-- |DRAFT 1A DD-MON-YYYY  Initial draft version |
-- |DRAFT 1B DD-MON-YYYY  Changes as per RCL ID   |
-- |1.0     DD-MON-YYYY  Baseline after testing  |
-- +-----+
```

ORACLE®

4 - 53

Copyright © 2009, Oracle. All rights reserved.

- For any new package, procedures, SQL scripts, installation files etc we should include the standard header for reference of the scripts
- This header is used to track the change history.
- Draft 1A is the initial draft version which is the first release of the code.
- Draft 1B is released after changes are made in the code after the review.
- 1.0 is the final release pack after baselined testing.

Development Standards

- Naming standards
 - Prefix the database objects like custom table, package, synonym, index, view, procedure, sequences, triggers, concurrent program, profile option name etc with XX
 - Suffix the database objects with object type like for package “_PKG”, staging table “_STG”, synonym “_SYN” etc.
 - Use database script file extension with respects to the object type like for package file, file extension might be “.PKB”, staging table, file extension might be “.tab” etc.

ORACLE®

4 - 54

Copyright © 2009, Oracle. All rights reserved.

Naming Standards for Files

File Type	File Extensions
Package Specification	.PKS
Package Body	.PKB
Table Script	.TAB
Synonym Script	.SYN
Grant Script	.GRT
Sequence Script	.SEQ
Trigger Script	.TRG
Procedure Script	.PRC
Shell Script	.PROG
Installation File Script	.INSTALL
Layout Definition Template	.LDT

Development Standards

- SQL Standards
 - Alignment/Structuring
 - UPPER and lower case in SQL Statements
 - Alignment within SQL Statements
 - Alignment of Function calls in SQL Statements
 - Positioning of tables in FROM Clause
 - Positioning of Conditions in WHERE/HAVING Clause

ORACLE

4 - 55

Copyright © 2009, Oracle. All rights reserved.

- **Alignment/Structuring:** Whenever we write any SQL query or PL/SQL script, the script should be align vertically and horizontally.
- **UPPER and lower case in SQL Statements:**
 - All standard keywords, standard or custom package/function names, Table aliases should be in upper case and all others (column names, column aliases etc.) should be in lower case.
 - **Examples:** -
 - SQL Statements: SELECT, UPDATE, INSERT, ALTER, CREATE, COMMIT, ROLLBACK
 - Clauses: INTO, FROM, WHERE, GROUP BY, ORDER BY, SET
 - Functions: TO_CHAR, TO_DATE, DECODE, NULL, NVL, RTRIM
 - Operators: AND, OR, BETWEEN, EXISTS, IN, NOT
 - Table Aliases: RCT

- **Alignment within SQL Statements:**

- The SQL Statements should be aligned and indented as follows:
- Each ‘Clause’ of SQL statement should start a new line.
- Only one column/expression, table or selectivity condition should appear on one line.
- Comma (,) separating the columns or tables should come at the beginning of lines and not at the end.
- All major clauses of a SQL statement should be aligned to one another.
- SELECT clause of sub-query should start on a new line. The ‘opening’ bracket enclosing the sub-query should be at the end of previous line.
- The column aliases and table aliases should be aligned to one another.

- **Examples: -**

- **SELECT Statement**

```

SELECT RCT trx_number invoice_number
      , RCT trx_date     invoice_date
      , SUM(APS amount_due_original)
      invoice_amount
FROM   ra_customer_trx      RCT
      , ar_payment_schedules  APS
WHERE RCT.customer_trx_id    =
      APS.customer_trx_id
      AND APS class           = 'INV'
      AND RCT completed_flag  = 'Y'
      AND RCT.customer_trx_id IN (SELECT
      DISTINCT ARA.customer_trx_id
      FROM ar_receivable_applications ARA
      WHERE apply_date BETWEEN
      ADD_MONTHS(SYSDATE,-1) AND
      SYSDATE)
      GROUP BY RCT trx_number
      , RCT trx_date
      ORDER BY RCT trx_number;

```

- **Alignment of Function calls in SQL Statements**
 - No more than 3 parameters of a function (standard or custom) can occur per line.
 - If parameter is an expression or nested function, then each expression or nested function should appear on new lines.
 - Comma (,) separating parameters should appear at the beginning of lines.
 - Expressions/Nested function calls should be properly indented to come inside the calling function calls.
 - In the case of DECODE/REPLACE, each combination of ‘search’ and ‘replace’ string should appear on a new line.

- **Examples**

```
TO_DATE('10102002','DDMMYYYY')
DECODE(APS.class
      , 'INV', 'Invoice'
      , 'CM', 'Credit Memo')
```

- **Positioning of tables in FROM Clause**

- The tables in the FROM clause of SELECT statement should be placed in the order in which system would access them, based on join and other selectivity conditions are provided.

- **Example: -**

```
SELECT /*+ ORDERED */
       FROM ra_customer_trx      RCT
         -- (Note- Base Table)
             , ra_customer_trx_lines  RCTL
             , ar_payment_schedules  APS
             , ra_customers          RCB
             , ra_site_uses           RSUB
             , ra_customers          RCS
             , ra_site_uses           RSUS
             , ar_customer_profiles   ACP
         -- (Note- Lookup type of tables)
             , ar_terms               AT
             , ra_cust_trx_types      RCTT
```

- **Positioning of Conditions in WHERE/HAVING Clause**
 - Only one join condition should appear in one line.
 - Keywords: AND or OR should appear at the beginning of line (and not at the end).
 - Nested Join conditions and OR conditions should be enclosed within brackets ('()').

Development Standards

- PL/SQL Variables / Parameters Declaration
 - Variables Declaration
 - Parameters Declaration

ORACLE

4 - 59

Copyright © 2009, Oracle. All rights reserved.

- **Variables Declaration:**
 - The variables should be declared at only one place per anonymous block, procedure, function or trigger.
 - The variables are declared in the following sequence
 - CONSTANT variables
 - EXCEPTION Declarations
 - Scalar Variables
 - Defined in alphabetical order
 - Record and Table Datatype definitions along with associated variables
 - Cursors along with associated record variables
 - The declaration of various kind of variables should be done as per the following structure.

- **Examples**

- DECLARE (or AS for package/procedure/function/triggers)
- e_exception EXCEPTION;
- l_variable1 datatype[(length)]
[:= value];
- l_variable2 datatype[(length)]
[:= value];
- l_variable3 <table.column>%TYPE;
- lr_variable4 user Datatype; --
RECORD or TABLE or REF Cursor

- **Parameter Declaration:**

- **Examples**

- **Procedure**

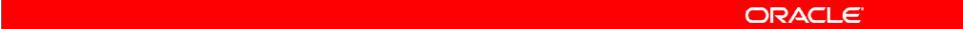
```
PROCEDURE/FUNCTION name(
    p_parameter1 datatype [DEFAULT value]
    , p_parameter2 datatype [DEFAULT value]
)
```

- **Cursor**

```
CURSOR c_cursor_name(
    p_parameter1 Datatype
    , p_parameter2 Datatype
)
```

Development Standards

- Oracle Applications Standards
 - New Tables, Indexes and Sequences should be created in a custom schema, and the database object of custom schema should be grants and synonyms should be created so that these are available in the APPS schema
 - Tablespace and Storage clauses should explicitly specify in every Table and Index creation command (i.e. CREATE TABLE|INDEX command).
 - Refer to ‘Oracle Applications Developer’s Guide’ on instructions to use Oracle Applications related standards.

ORACLE

4 - 61

Copyright © 2009, Oracle. All rights reserved.

Summary

After completing this lesson, you should be able to understand:

- Various Interface Options
- Open Interface Model
- API Diagram
- Purchasing Open Interfaces
- Payables Open Interfaces
- Order Management Open Interfaces
- Receivables Open Interfaces
- Designing, Developing, and Performing Conversion Process
- Development standards for code

ORACLE®

4 - 62

Copyright © 2009, Oracle. All rights reserved.

Interfaces Deployment

Environment (Source and Target)

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

After completing this lesson, you should be able to do the following:

- Steps involved in Inbound Interface Deployment
- Steps involved in Outbound Interface Deployment
- Prepare Installation Pack



Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®

5 - 3

Copyright © 2009, Oracle. All rights reserved.

Data file sample

Column	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	
1	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	39.68	0	920952004	J	MURPHY	1091564AC - BUSINESS MILES						
2	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	34.55	0	920952004	JL	MURPHY	1091564AC - LUMP SUM NO NI						
3	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	34.17	0	920952004	JL	MURPHY	1091564AC - ESS SUMM SUM						
4	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	2	0	920952004	JL	MURPHY	1091564AC - TELEPHONE CALLS							
5	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	548.00	0	920952004	JL	MURPHY	1091564AC - BASIC HRLY RATE 1073.30HR						
6	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	548.46	0	920952004	JL	MURPHY	1091564AC - OVERTIME x 3	1024.29HR					
7	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	170	0	0	568.08	0	920952004	JL	MURPHY	1091564AC - ERG						
8	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	170	0	0	568.08	0	920952004	JL	MURPHY	1091564AC - SUPER ERG						
9	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	227.86	0	920952004	J	AITKEN	1111414AB - BUSINESS MILES						
10	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	49.13	0	920952004	J	AITKEN	1111414AC - LUMP SUM NO NI						
11	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	77.00	0	920952004	J	AITKEN	1111414AC - BUSINESS MILES						
12	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	3	0	920952004	J	AITKEN	1111414AC - TELEPHONE CALLS						
13	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	1201.2	0	920952004	J	AITKEN	111140207 - BASIC HRLY RATE 105.60HR						
14	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	1201.2	0	920952004	J	AITKEN	111140207 - BASIC HRLY RATE 101.20HR						
15	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	498.46	0	920952004	J	AITKEN	111140206 - OVERTIME x 3	1020.06HR					
16	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	170	0	0	163.76	0	920952004	J	AITKEN	111140204JN1 - ERG						
17	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	163.76	0	920952004	J	AITKEN	111140204JN1 - SUPER ERG						
18	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	199.38	0	920952004	S	BALAGUE	1091564AC - BUSINESS MILES						
19	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	34.36	0	920952004	S	BALAGUE	1091564AC - LUMP SUM NO NI						
20	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	22.47	0	920952004	S	BALAGUE	1091564AC - BUSINESS MILES						
21	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	5.7	0	920952004	S	BALAGUE	1091564AC - TELEPHONE CALLS						
22	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	9999	8804	0	0	0	0	67.91	920952004	S	BALAGUE	1091564CV7 - PROFESSIONAL ADV						
23	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	149.22	0	920952004	S	BALAGUE	1091564CV7 - BASIC HRLY RATE 1176.52HR						
24	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	298.03	0	920952004	S	BALAGUE	1091564CV7 - BASIC HRLY RATE 1024.04HR						
25	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	170	0	0	173.45	0	920952004	S	BALAGUE	1091564CV7 - OVERTIME x 3	1020.06HR					
26	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	170	0	0	139.81	0	920952004	S	BALAGUE	1091564CV7 - ERG						
27	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	173.45	0	920952004	S	BROOKER	1090564AC - BUSINESS MILES						
28	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	23.15	0	920952004	S	BROOKER	1090564AC - LUMP SUM NO NI						
29	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	34.68	0	920952004	S	BROOKER	1090564AC - ESS SUMM SUM						
30	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	13.33	0	920952004	S	BROOKER	1090564AC - TELEPHONE CALLS						
31	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	9999	8804	0	0	0	0	6.00	920952004	S	BROOKER	1090564AC - BUSINESS MILES						
32	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	736.65	0	920952004	S	BROOKER	1090564DZ - BASIC HRLY RATE 1097.00HR						
33	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	732.00	0	920952004	S	BROOKER	1090564DZ - BASIC HRLY RATE 1020.00HR						
34	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	77.31	0	920952004	S	BROOKER	1090564DZ - OVERTIME x 3	1003.12HR					
35	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	170	0	0	53.62	0	920952004	S	BROOKER	1090564DZ - UNISON						
36	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	102.00	0	920952004	S	BROOKER	1090564DZ - SUPER ERG						
37	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	104.34	0	920952004	S	BROOKER	1090564DZ - BUSINESS MILES						
38	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	23.49	0	920952004	S	BROOKER	1090564DZ - LUMP SUM NO NI						
39	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	54.24	0	920952004	S	BROOKER	1090564DZ - ESS SUMM SUM						
40	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	9999	8804	0	0	0	7.24	920952004	D	LENNON	10912E02 - BUSINESS MILES							
41	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	465.37	0	920952004	D	LENNON	10912E02 - LUMP SUM NO NI						
42	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	150	0	0	37.00	0	920952004	D	LENNON	10912E02 - ESS SUMM SUM						
43	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	90	160	0	0	37.22	0	920952004	D	LENNON	10912E02 - TELEPHONE CALLS						
44	LIBERATA PAYROLL	92/6/5/2004	GBP	A	1	53991	461	0	0	0	274.91	0	920952004	LJ	TURFREY	10912E02 - BUSINESS MILES						

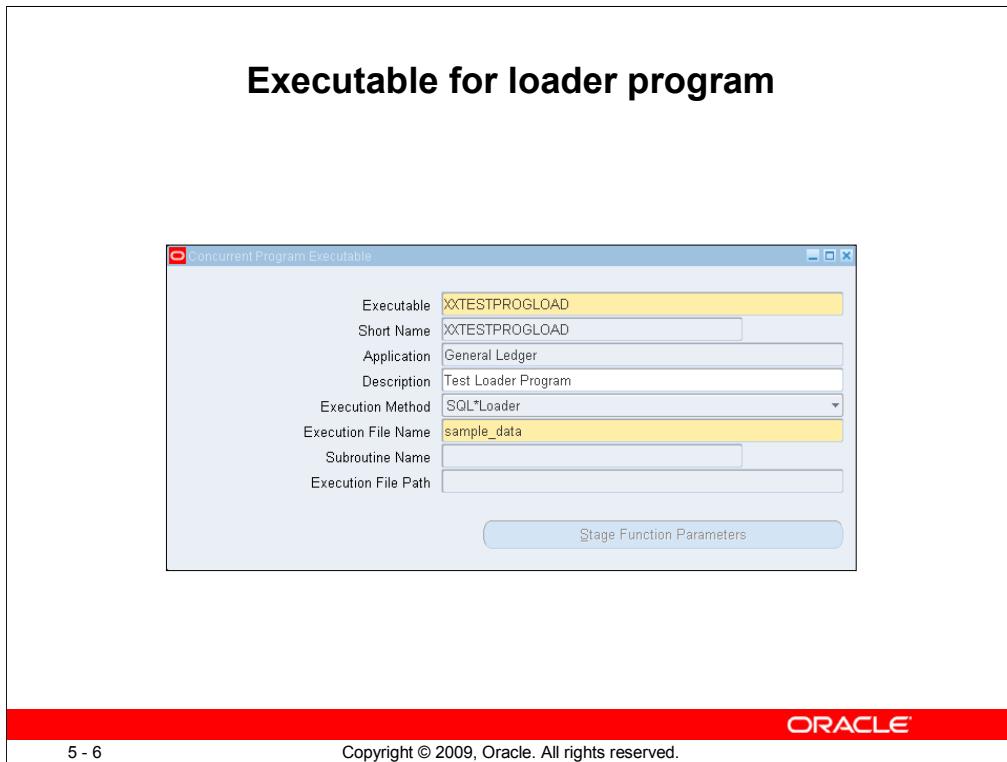
Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®

5 - 5

Copyright © 2009, Oracle. All rights reserved.



5 - 6

Copyright © 2009, Oracle. All rights reserved.

ORACLE

Navigation:

- Select System Administrator Responsibility
- Click on Concurrent -> Program -> Executable

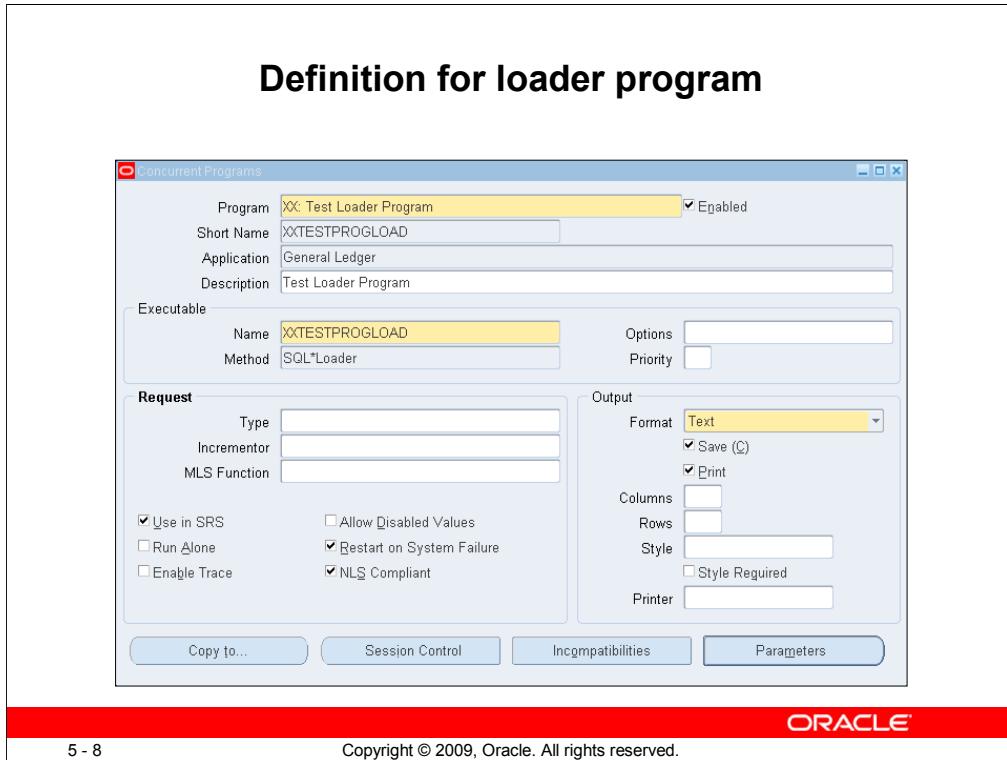
Notes:

- Enter the executable name without any spaces and prefixed with XX.
- Enter the short name as same as the executable name.
- Identify the application where the program needs to be registered (Generally the DBA creates a custom application specific to the project).
- Give a meaningful description of the executable.
- Select SQL*Loader for the execution method.
- Enter the name of the csv file in the execution file name.
- Save the executable by clicking File -> Save.

Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- **Create definition for loader program**
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®



Navigation:

- Select System Administrator Responsibility
- Click on Concurrent -> Program -> Define

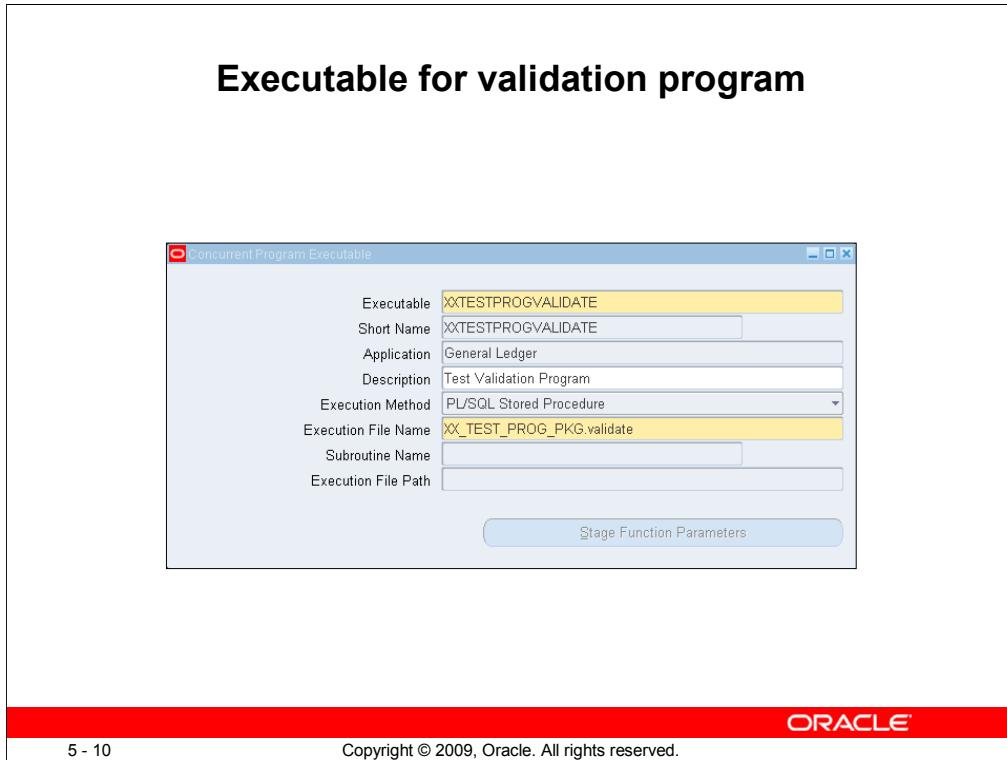
Notes:

- Enter the program name prefixed with XX, followed by a colon(:) and a meaningful description.
- Enter the short name as same as the executable name.
- Identify the application where the program needs to be registered. This should be same as the application of the executable.
- Give a meaningful description of the definition.
- Save the definition by clicking File -> Save.

Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- **Create executable for validation program**
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®



5 - 10

Copyright © 2009, Oracle. All rights reserved.

ORACLE

Navigation:

- Select System Administrator Responsibility
- Click on Concurrent -> Program -> Executable

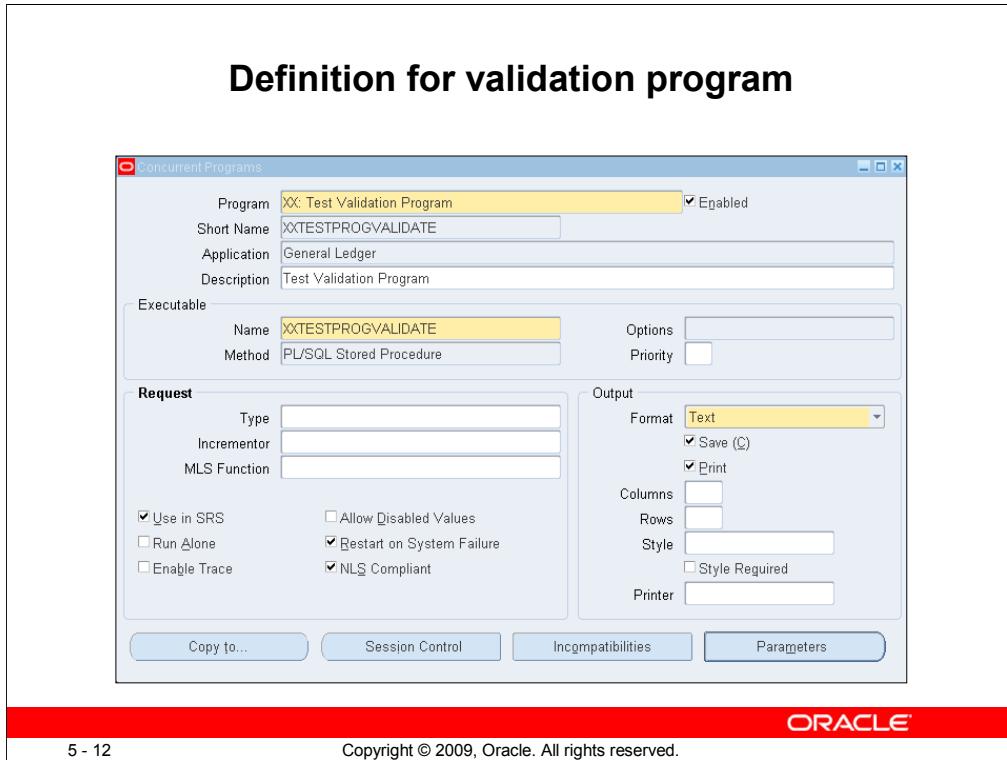
Notes:

- Enter the executable name without any spaces and prefixed with XX.
- Enter the short name as same as the executable name.
- Identify the application where the program needs to be registered.
- Give a meaningful description of the executable.
- Select PL/SQL Stored Procedure for the execution method.
- Enter the name of the validation procedure in the execution file name.
- Save the executable by clicking File -> Save.

Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- **Create definition for validation program**
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®



Navigation:

- Select System Administrator Responsibility
- Click on Concurrent -> Program -> Define

Notes:

- Enter the program name prefixed with XX, followed by a colon(:) and a meaningful description.
- Enter the short name as same as the executable name.
- Identify the application where the program needs to be registered. This should be same as the application of the executable.
- Give a meaningful description of the definition.
- Save the definition by clicking File -> Save.

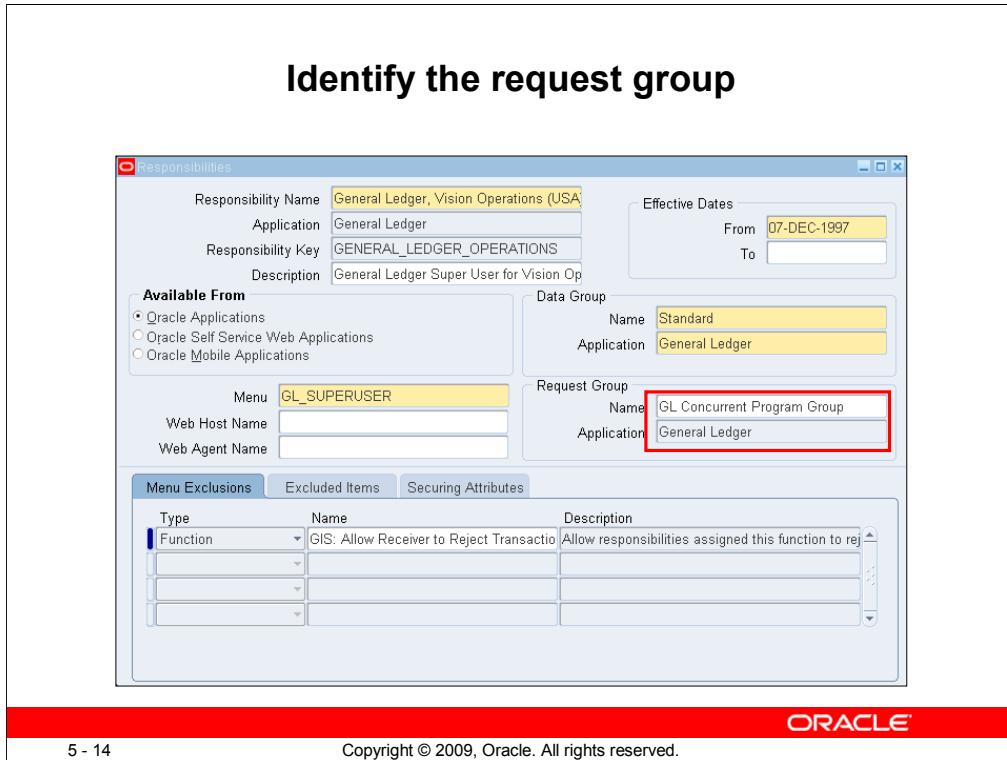
Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®

5 - 13

Copyright © 2009, Oracle. All rights reserved.



Navigation:

- Select System Administrator Responsibility
- Click on Security -> Responsibility -> Define

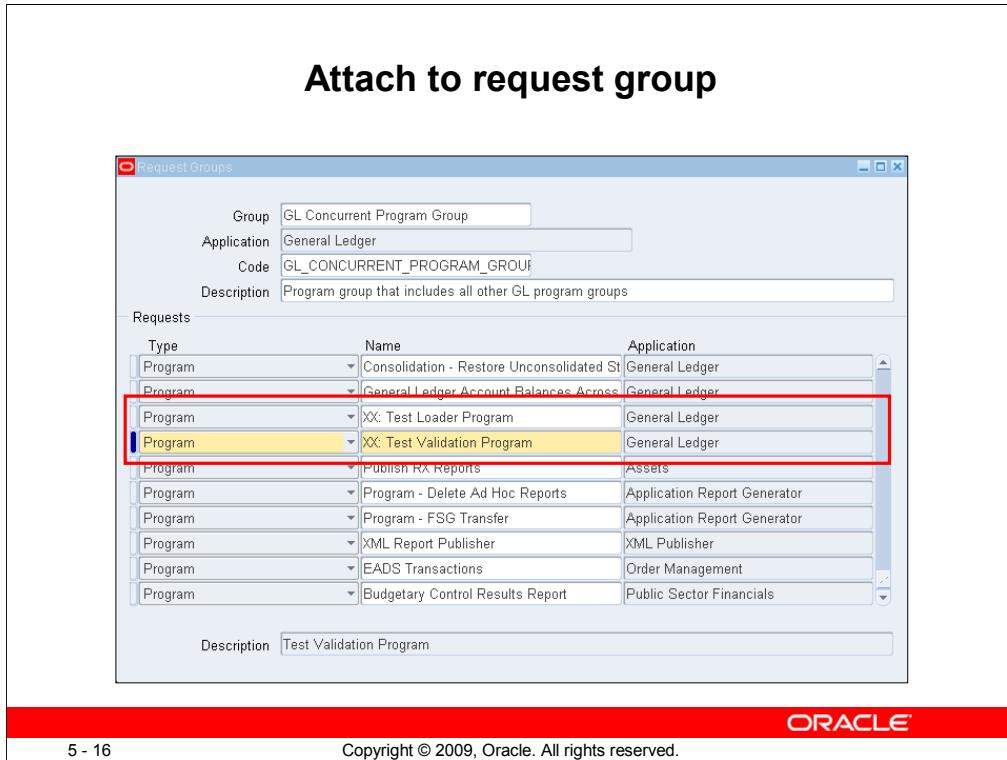
Notes:

- Press F11 to search for the responsibility. For example, Enter ‘General Ledger%’ to search for General Ledger related responsibilities.
- Press Ctrl + F11 after the search query is entered.
- Note down the request group name. In this case, the request group name is ‘GL Concurrent Program Group’

Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- **Attach the programs to the request group**
- Download the concurrent programs
- Create Installation pack

ORACLE®

**Navigation:**

- Select System Administrator Responsibility
- Click on Security -> Responsibility -> Request

Notes:

- Press F11 to search for the request group. For example, Enter 'GL Concurrent Program Group' to search for General Ledger request group.
- Press Ctrl + F11 after the search query is entered.
- Click on the Requests panel and create a new row by clicking File -> New.
- Enter the names of the concurrent program definitions that are to be added to the request group.
- Save the request group by clicking File -> Save.

Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- **Download the concurrent programs**
- Create Installation pack

ORACLE®

Download concurrent programs

- Run the following command from the Unix box of the source system to download the concurrent programs.
- FNDLOAD apps/<apps_password> 0 Y DOWNLOAD
\$FND_TOP/patch/115/import/afcpprog.lct
xx_concurrent_program_file_name.ldt PROGRAM
CONCURRENT_PROGRAM_NAME="XX_NAME"
APPLICATION_SHORT_NAME="XX_APPLICATION"

ORACLE®

5 - 18

Copyright © 2009, Oracle. All rights reserved.

- The Generic Loader (FNDLOAD) is a concurrent program that can transfer Oracle Application entity data between database and text file.
 - xx_concurrent_program_file_name.ldt - File name of the concurrent program definition
 - XX_NAME - Short name of the concurrent program executable
 - XX_APPLICATION - Short name of the application where the concurrent program is registered
- In this case, the following command is used for downloading the loader concurrent program
 - FNDLOAD apps/apps 0 Y DOWNLOAD
\$FND_TOP/patch/115/import/afcpprog.lct xxtestprogload.ldt
PROGRAM
CONCURRENT_PROGRAM_NAME="XXTESTPROGLOAD"
APPLICATION_SHORT_NAME="GL"

Note:

Please ensure that the environment file is initialized to access \$FND_TOP. The environment file is present in the home directory of the unix box with an extension of .env. For example, run ./apps.env to initialize the environment file for the instance.

Inbound Interface deployment steps

- Request the client to provide the data file (csv file)
- Create executable for loader program
- Create definition for loader program
- Create executable for validation program
- Create definition for validation program
- Identify the request group associated with the responsibility
- Attach the programs to the request group
- Download the concurrent programs
- Create Installation pack

ORACLE®

Installation Pack

- Transfer the files which are part of the installation pack into a new folder in the unix box.
- Create a tar file of the contents using the following command
 - `tar -cvf XXTESTPROG.tar .`
- Compress the tar file using the command
 - `compress XXTESTPROG.tar`

ORACLE®

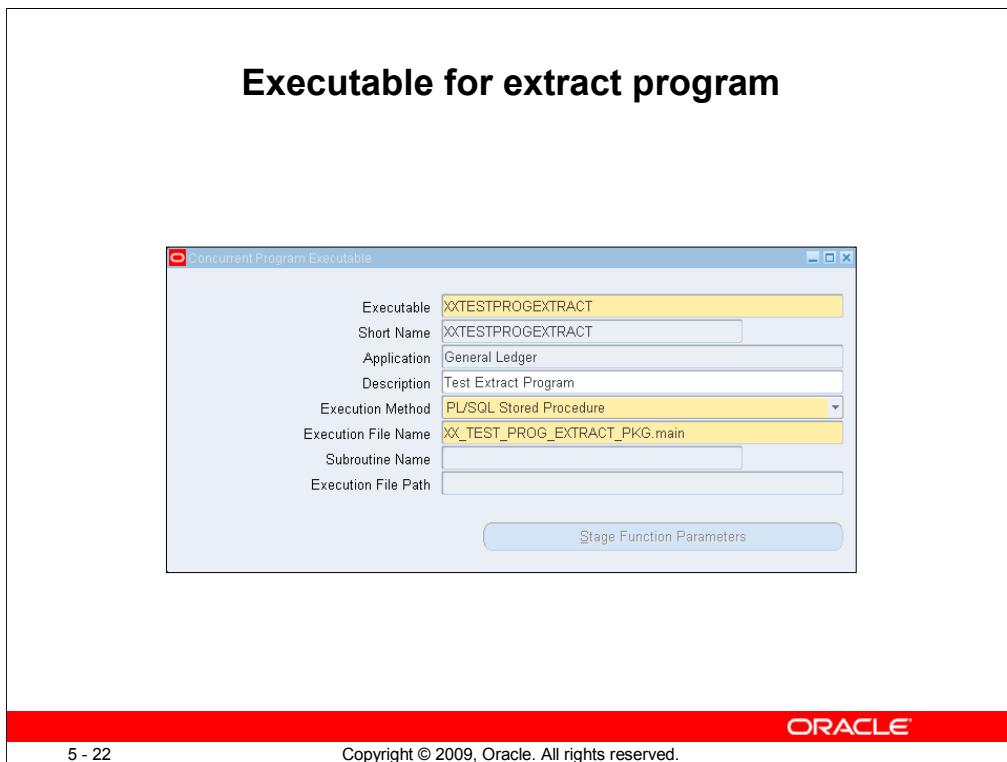
Following are the common files which are part of the installation pack of an inbound interface

- Staging table creation scripts
- Grant scripts for staging tables
- Synonym creation scripts
- Package specification
- Package body
- Loader program shell script
- Concurrent program LDT files
- Installation scripts

Outbound Interface deployment steps

- Create executable for extract program
- Create definition for extract program
- Identify the request group associated with the responsibility
- Attach the program to the request group
- Download the concurrent program
- Create the installation pack

ORACLE®



Navigation:

- Select System Administrator Responsibility
- Click on Concurrent -> Program -> Executable

Notes:

- Enter the executable name without any spaces and prefixed with XX.
- Enter the short name as same as the executable name.
- Identify the application where the program needs to be registered.
- Give a meaningful description of the executable.
- Select PL/SQL Stored Procedure for the execution method.
- Specify the file creation path in the package body as mentioned in the functional design document
- Enter the name of the extract procedure in the execution file name.
- Save the executable by clicking File -> Save.

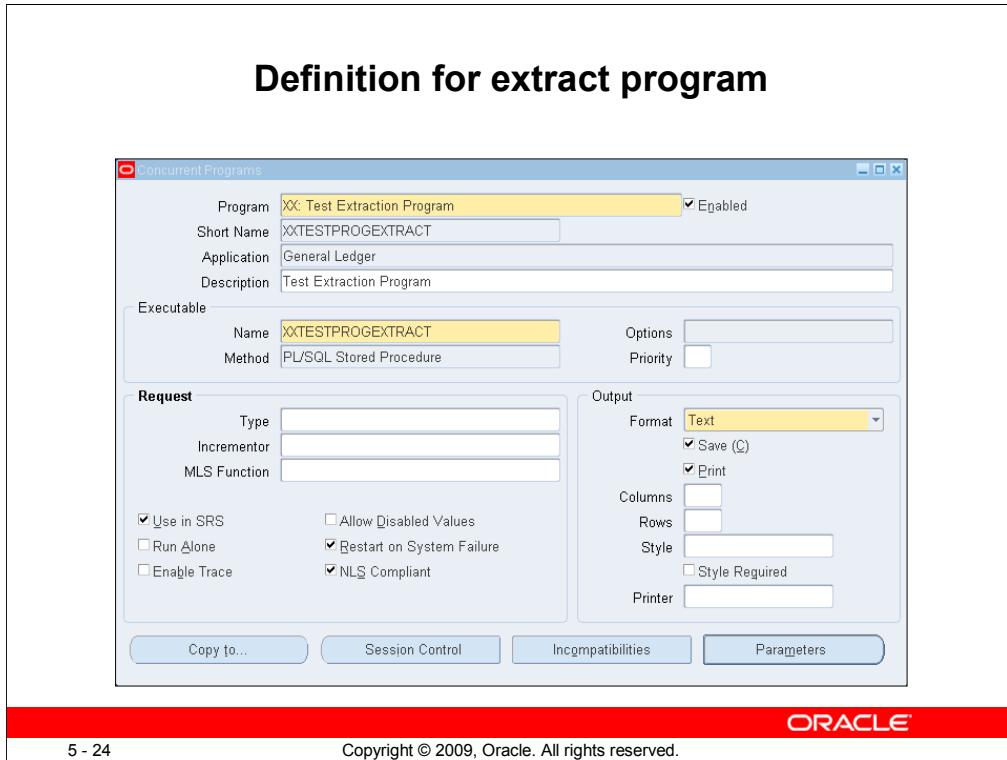
Outbound Interface deployment steps

- Create executable for extract program
- Create definition for extract program
- Identify the request group associated with the responsibility
- Attach the program to the request group
- Download the concurrent program
- Create the installation pack

ORACLE®

5 - 23

Copyright © 2009, Oracle. All rights reserved.



Navigation:

- Select System Administrator Responsibility
- Click on Concurrent -> Program -> Define

Notes:

- Enter the program name prefixed with XX, followed by a colon(:) and a meaningful description.
- Enter the short name as same as the executable name.
- Identify the application where the program needs to be registered. This should be same as the application of the executable.
- Give a meaningful description of the definition.
- Save the definition by clicking File -> Save.

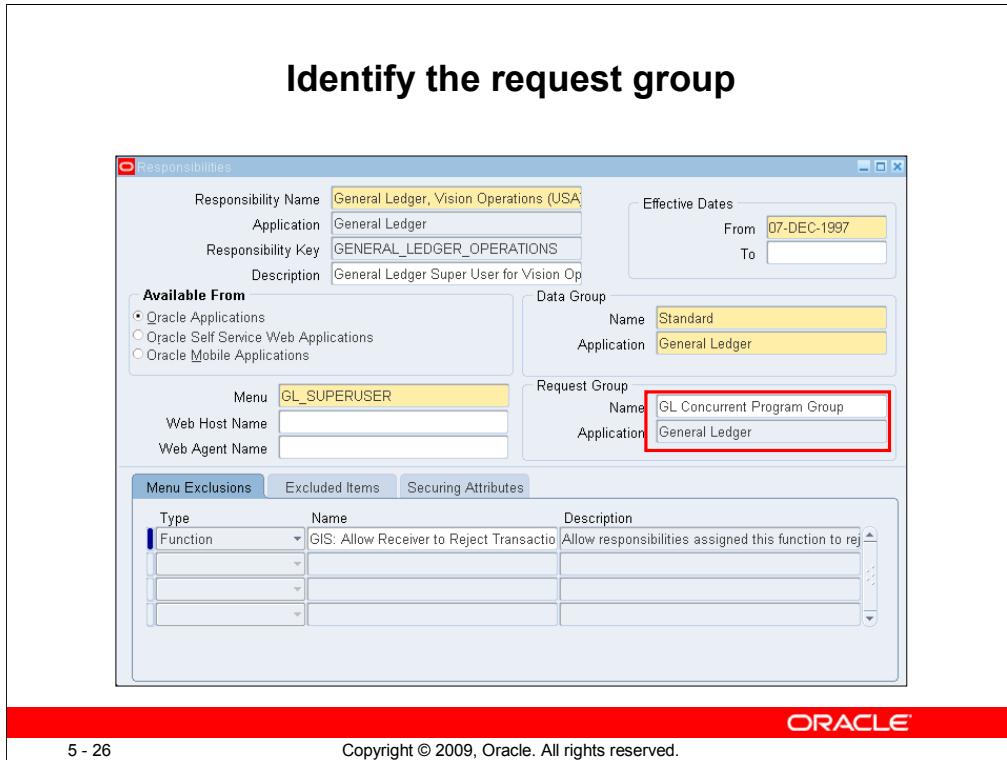
Outbound Interface deployment steps

- Create executable for extract program
- Create definition for extract program
- Identify the request group associated with the responsibility
- Attach the program to the request group
- Download the concurrent program
- Create the installation pack

ORACLE®

5 - 25

Copyright © 2009, Oracle. All rights reserved.



Navigation:

- Select System Administrator Responsibility
- Click on Security -> Responsibility -> Define

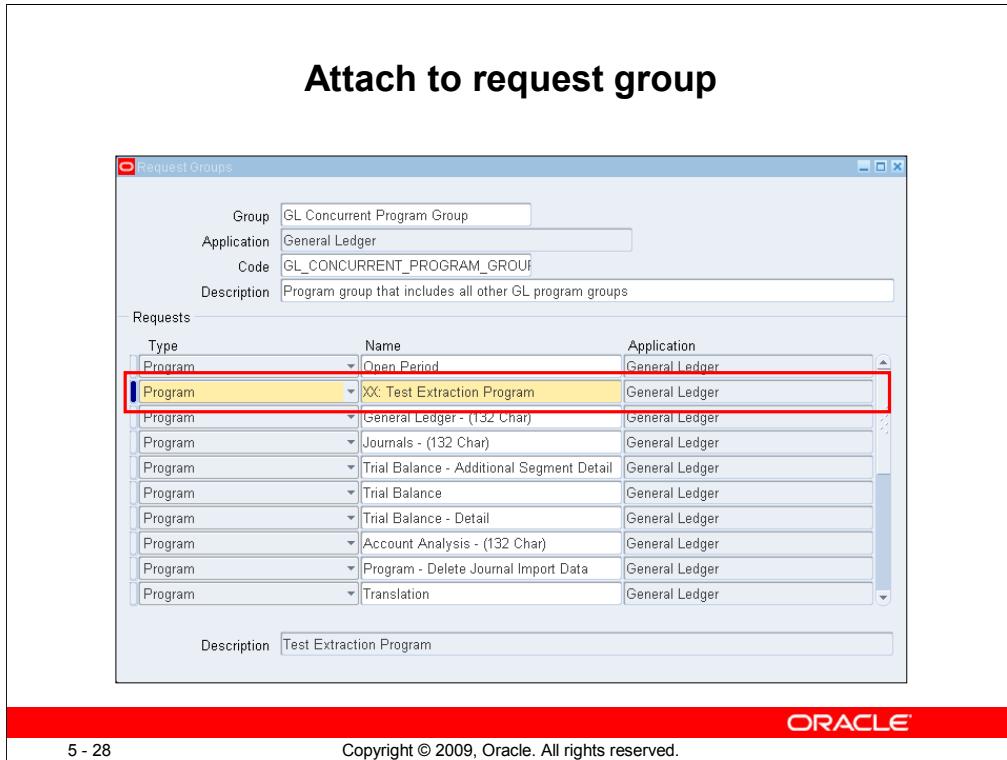
Notes:

- Press F11 to search for the responsibility. For example, Enter ‘General Ledger%’ to search for General Ledger related responsibilities.
- Press Ctrl + F11 after the search query is entered.
- Note down the request group name. In this case, the request group name is ‘GL Concurrent Program Group’

Outbound Interface deployment steps

- Create executable for extract program
- Create definition for extract program
- Identify the request group associated with the responsibility
- **Attach the program to the request group**
- Download the concurrent program
- Create the installation pack

ORACLE®

**Navigation:**

- Select System Administrator Responsibility
- Click on Security -> Responsibility -> Request

Notes:

- Press F11 to search for the request group. For example, Enter 'GL Concurrent Program Group' to search for General Ledger request group.
- Press Ctrl + F11 after the search query is entered.
- Click on the Requests panel and create a new row by clicking File -> New.
- Enter the name of the concurrent program definition that needs to be added to the request group.
- Save the request group by clicking File -> Save.

Outbound Interface deployment steps

- Create executable for extract program
- Create definition for extract program
- Identify the request group associated with the responsibility
- Attach the program to the request group
- Download the concurrent program
- Create the installation pack

ORACLE®

5 - 29

Copyright © 2009, Oracle. All rights reserved.

Download concurrent programs

- Run the following command from the Unix box of the source system to download the concurrent program.
- FNDLOAD apps/<apps_password> 0 Y DOWNLOAD
\$FND_TOP/patch/115/import/afcpprog.lct
xx_concurrent_program_file_name.ldt PROGRAM
CONCURRENT_PROGRAM_NAME="XX_NAME"
APPLICATION_SHORT_NAME="XX_APPLICATION"

ORACLE®

5 - 30

Copyright © 2009, Oracle. All rights reserved.

- The Generic Loader (FNDLOAD) is a concurrent program that can transfer Oracle Application entity data between database and text file.
 - xx_concurrent_program_file_name.ldt - File name of the concurrent program definition
 - XX_NAME - Short name of the concurrent program executable
 - XX_APPLICATION - Short name of the application where the concurrent program is registered
- In this case, the following command is used for downloading the extract concurrent program
 - FNDLOAD apps/apps 0 Y DOWNLOAD
\$FND_TOP/patch/115/import/afcpprog.lct xxtestprogextract.ldt
PROGRAM
CONCURRENT_PROGRAM_NAME="XXTESTPROGEXTRA
CT" APPLICATION_SHORT_NAME="GL"

Note:

Please ensure that the environment file is initialized to access \$FND_TOP. The environment file is present in the home directory of the unix box with an extension of .env. For example, run ./apps.env to initialize the environment file for the instance.

Outbound Interface deployment steps

- Create executable for extract program
- Create definition for extract program
- Identify the request group associated with the responsibility
- Attach the program to the request group
- Download the concurrent program
- Create the installation pack

ORACLE®

Installation Pack

- Transfer the files which are part of the installation pack into a new folder in the unix box.
- Create a tar file of the contents using the following command
 - `tar -cvf XXTESTPROGEXTRACT.tar .`
- Compress the tar file using the command
 - `compress XXTESTPROGEXTRACT.tar`

ORACLE®

Following are the common files which are part of the installation pack of an outbound interface

- Package specification
- Package body
- Extract program shell script
- Concurrent program LDT file
- Installation scripts

Summary

In this lesson, you should have learned to do the following:

- Steps involved in Inbound Interface Deployment
- Steps involved in Outbound Interface Deployment
- Prepare Installation Pack



Debugging Interface Program Errors



Supplemental Student Practices

Purchasing (PO and RCV)

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

After completing this lesson, you should be able to do the following using Oracle Applications:

- Create a Supplier
- Enter a Purchase Requisition
- Auto Create a Purchase Order
- Receive Purchased Material



Agenda

- Practice - PO: Create a Supplier
- Solution - PO: Create a Supplier
- Practice - PO: Enter a Purchase Requisition
- Solution - PO: Enter a Purchase Requisition
- Practice - PO: AutoCreate a Purchase Order
- Solution - PO: AutoCreate a Purchase Order
- Practice - RCV: Receive Purchased Material
- Solution - RCV: Receive Purchased Material



Practice - PO: Create a Supplier

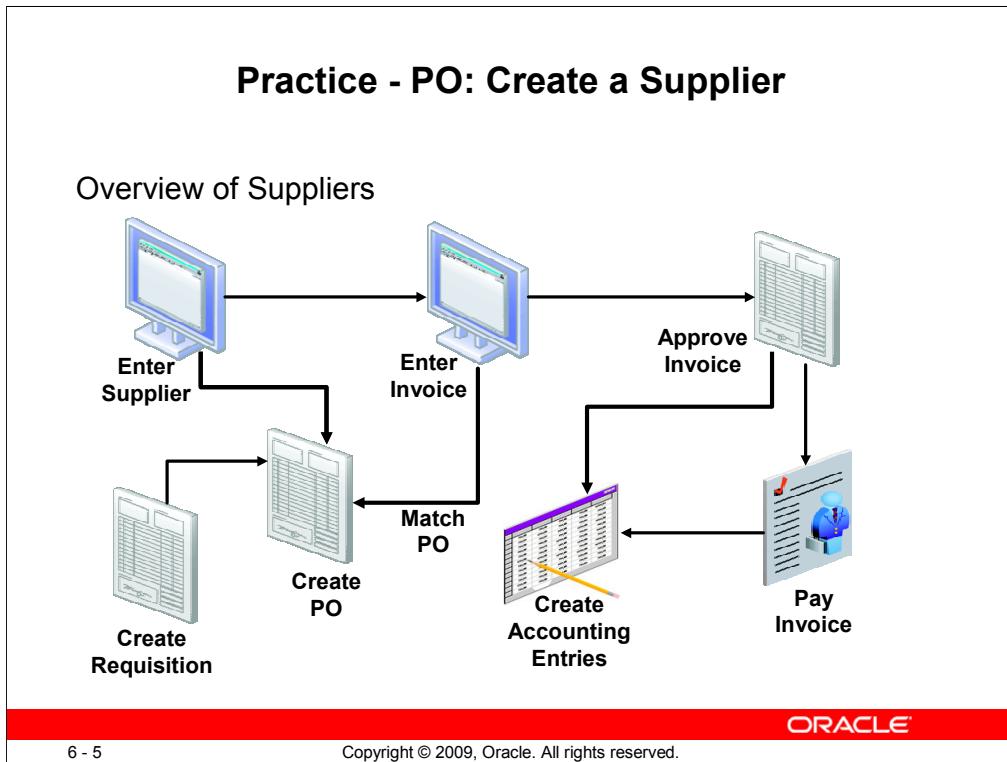
Solution - PO: Create a Supplier

- Overview of Suppliers
- Define suppliers
- Define supplier sites

ORACLE®

6 - 4

Copyright © 2009, Oracle. All rights reserved.

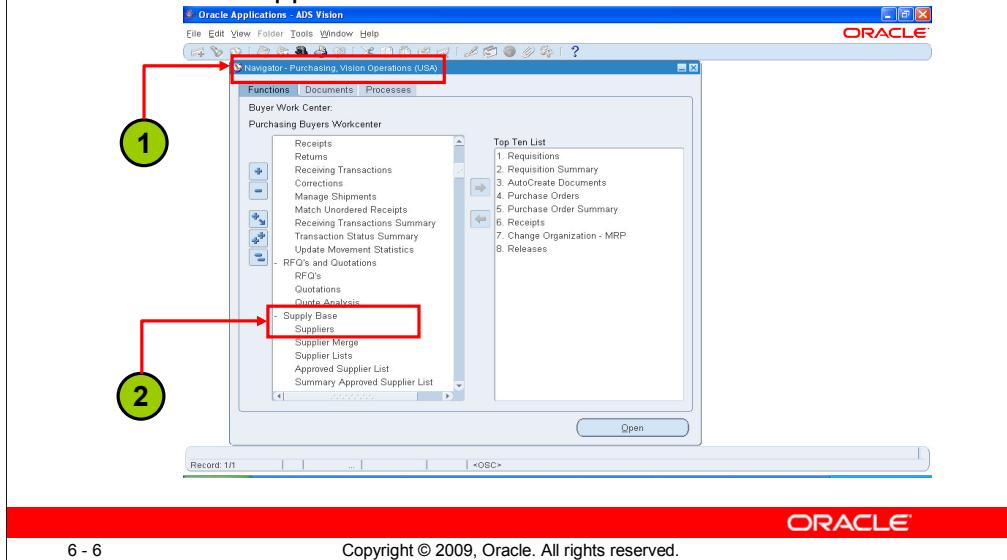


Overview of Suppliers

- Set up suppliers in the Suppliers window to record information about companies and individuals from whom you purchase goods and services. When you enter a supplier that does business from multiple locations, you store supplier header information only once, and you enter supplier sites for each location. Most supplier information defaults to supplier sites. However, you can override the values that default if necessary. After you define suppliers, you can use them when you enter invoices and create purchasing documents.

Practice - PO: Create a Supplier

- Define Suppliers

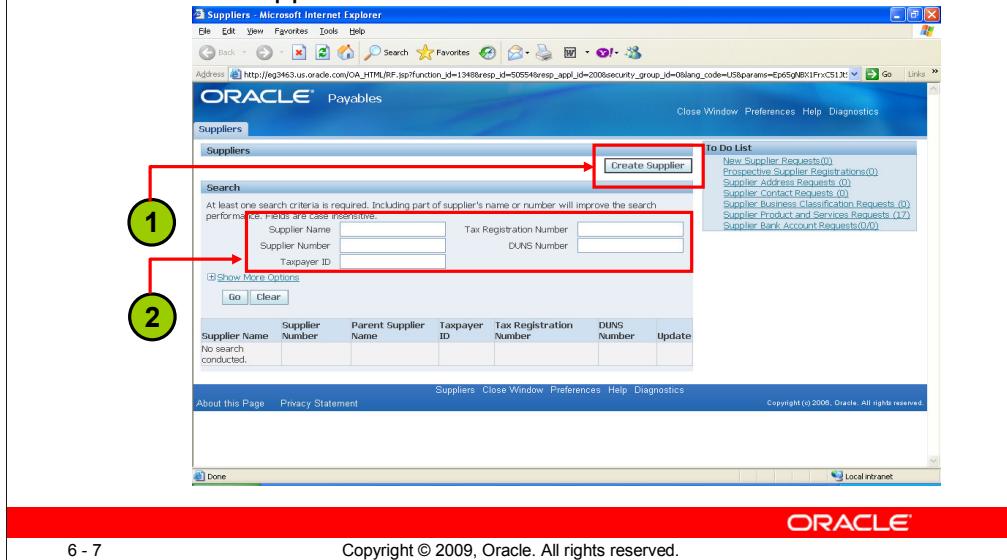


Define supplier

1. Navigate to the Purchasing Responsibility
2. Select Supply Base->Suppliers to create a new supplier

Practice - PO: Create a Supplier

- Define Suppliers



Define supplier

1. For creating a new supplier click on Create Supplier Button
2. For searching an existing supplier, enter the search keyword and click on Go Button.

Practice - PO: Create a Supplier

- Define Suppliers Headers

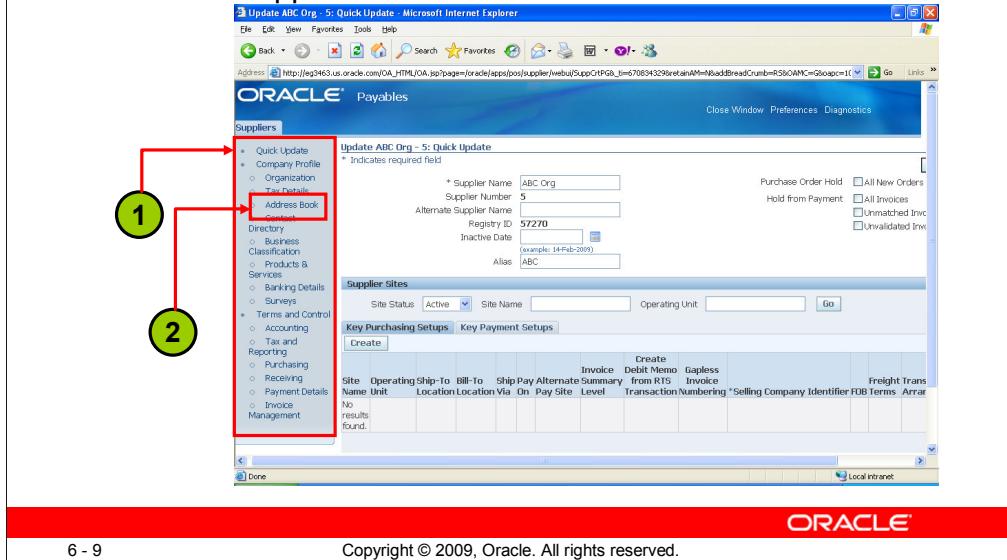
The screenshot shows the 'Create Supplier' page in Oracle Payables. The 'Supplier Type' is set to 'Standard supplier'. The 'Organization Name' field contains 'ABC Org'. The 'Content Value' dropdown is set to 'Must include: http://'. The 'Apply' button is highlighted. The page includes standard Oracle navigation links like Back, Forward, Stop, Refresh, and Home.

Define supplier

1. Select the Standard template for creating the supplier
2. Enter the Supplier information
3. Click on Apply button to create supplier header information
4. When the Apply button is pressed it will create the supplier number automatically

Practice - PO: Create a Supplier

- Define Suppliers Sites

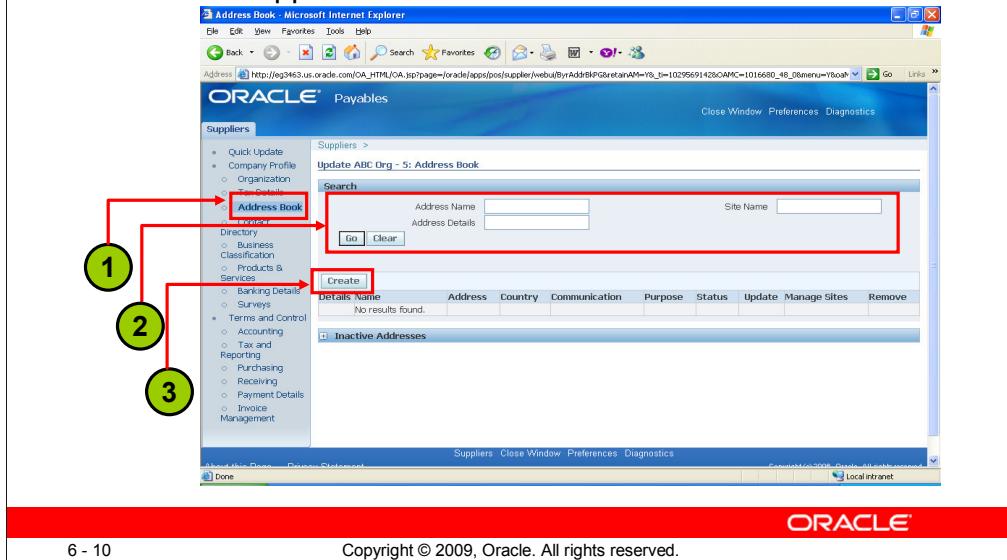


Define supplier

- After creating Supplier Header information, enter the other information for Supplier Sites
- For creating Supplier Site first create the site address, click on Address Book for creating the address

Practice - PO: Create a Supplier

- Define Suppliers Sites



Define supplier

1. Click on Address Book for creating the address for Supplier Site
2. If address exist then either use the existing address or create a new address for a different site.
3. For creating a new site click on Create button.

Practice - PO: Create a Supplier

- Define Suppliers Sites

The screenshot shows the 'Create/Update Address' page in Microsoft Internet Explorer. A red box highlights the main address input area, and three green circles numbered 1, 2, and 3 point to specific fields within this box:

- 1**: Points to the 'Country' dropdown set to 'United States'.
- 2**: Points to the 'Address Line 1' field containing 'Street #5, Roland'.
- 3**: Points to the 'Address Name' field containing 'LA'.

On the right side of the page, there is a 'Contact Details and Purpose' section with several input fields and checkboxes. A red box highlights the 'Address Purpose' section, which contains three checkboxes: 'Purchasing' (checked), 'Payment' (checked), and 'RFQ Only' (unchecked). The 'Continue' button at the top right is also highlighted with a red box.

6 - 11

Copyright © 2009, Oracle. All rights reserved.

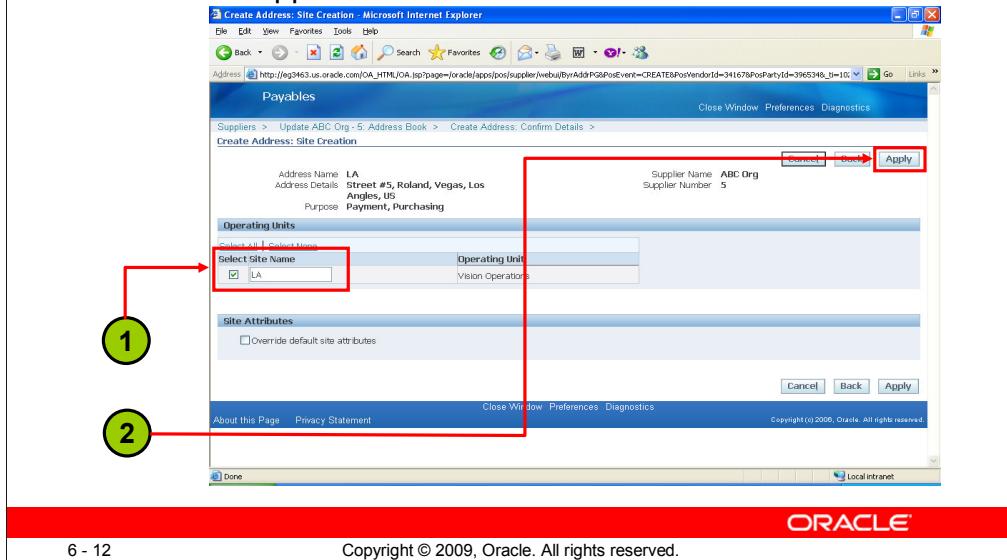
ORACLE

Define supplier

1. Enter the mandatory Address Information
2. Click on the check boxes for Purchasing and Payables
3. Click on Continue Button

Practice - PO: Create a Supplier

- Define Suppliers Sites



Define supplier

1. Click the Check Box for Site Name.
2. Click on Apply Button.

Practice - PO: Create a Supplier

- Define Suppliers Sites

The screenshot shows the Oracle Invoice Management interface. On the left, a navigation menu is visible with various categories like Quick Update, Company Profile, Organization, Tax Details, Address Book, Contact, Details, Business Classification, Products & Services, Banking Details, Surveys, Terms and Control, Accounting, Tax and Reporting, Purchasing, Receiving, Payment Details, and Invoice Management. The 'Invoice Management' option is selected.

The main area displays the 'Update ABC_Org - 5: Invoice Management' screen. It includes sections for 'Invoice Payment Terms' (with fields for Invoice Currency, Amount Limit, Match Option, Payment Currency, Priority, Terms, and Dates) and 'Supplier Sites' (with columns for Site Name, Status, Operating Unit, Invoice Amount Limit, Tolerance, Match Option, Currency, and checkboxes for Hold from Payment, Pay Group, and Unmatched Invoices). A red box highlights the 'Invoice Payment Terms' section, and another red box highlights the 'Supplier Sites' table. A green circle labeled '1' points to the 'Invoice Payment Terms' section, and a green circle labeled '2' points to the 'Supplier Sites' table.

At the bottom, there is a red footer bar with the text 'Copyright © 2009, Oracle. All rights reserved.' and the 'ORACLE' logo.

Define supplier

1. Enter other mandatory information for Supplier Sites
2. When all the information for Supplier Sites are entered, click on Save Button to apply the changes for Supplier Sites.

Agenda

- Practice - PO: Create a Supplier
- Solution - PO: Create a Supplier
- Practice - PO: Enter a Purchase Requisition
- Solution - PO: Enter a Purchase Requisition
- Practice - PO: AutoCreate a Purchase Order
- Solution - PO: AutoCreate a Purchase Order
- Practice - RCV: Receive Purchased Material
- Solution - RCV: Receive Purchased Material

ORACLE®

Practice - PO: Enter a Purchase Requisition

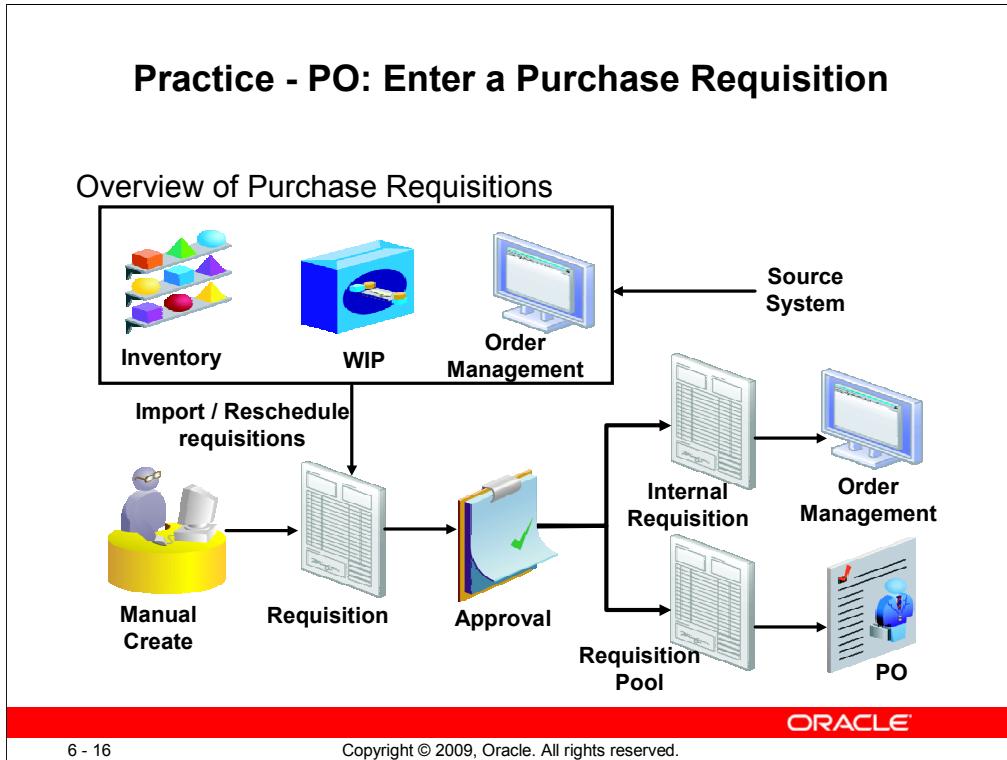
Solution - PO: Enter a Purchase Requisition

- Overview of Purchase Requisitions
- Create Purchase Requisitions Header
- Create Purchase Requisitions Lines

ORACLE®

6 - 15

Copyright © 2009, Oracle. All rights reserved.

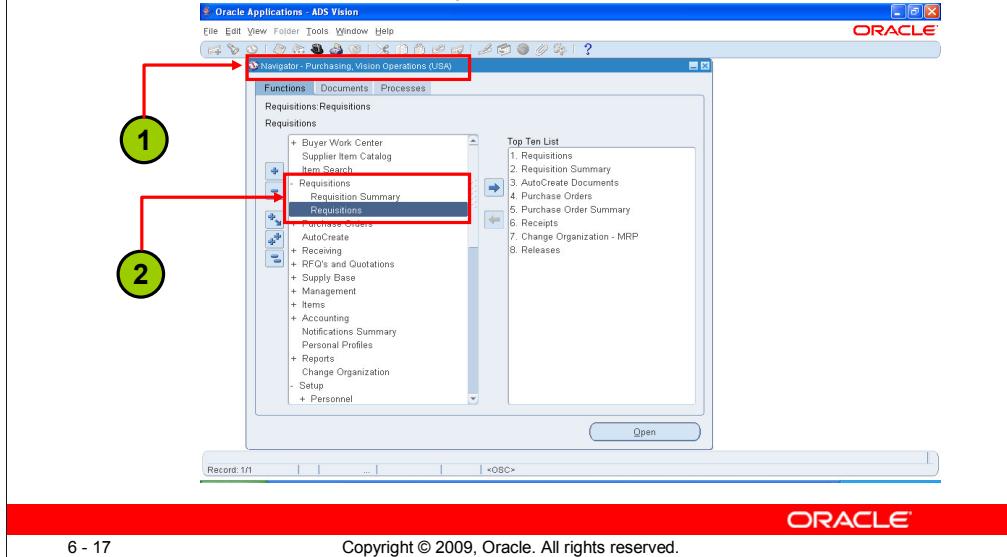


Overview of Suppliers

- Requisitions represent demand for goods and services. Demand can come from a variety of sources.
- Using these variety of sources or by manually we can create Purchase Requisitions
- When Purchase Requisitions got created it goes for supervisor Approval.
- When Purchase Requisition get Approved by Supervisor either we can create Sales Order or Purchase Order from this.

Practice - PO: Enter a Purchase Requisition

- Create a Purchase Requisition

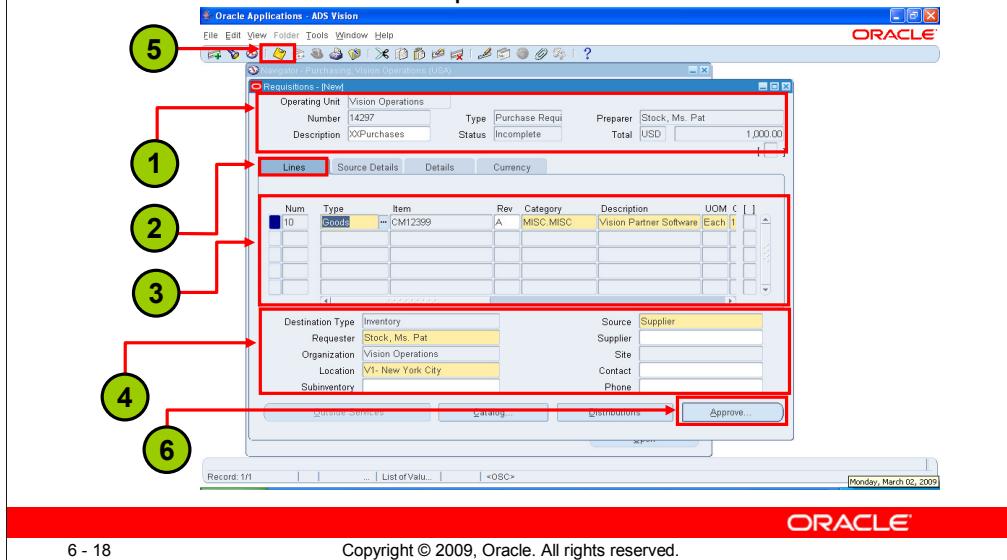


Create a Purchase Requisitions

1. Navigate to the Purchasing Responsibility
2. Select Requisitions-> Requisitions to enter a Purchase Requisition

Practice - PO: Enter a Purchase Requisition

- Create a Purchase Requisition



6 - 18

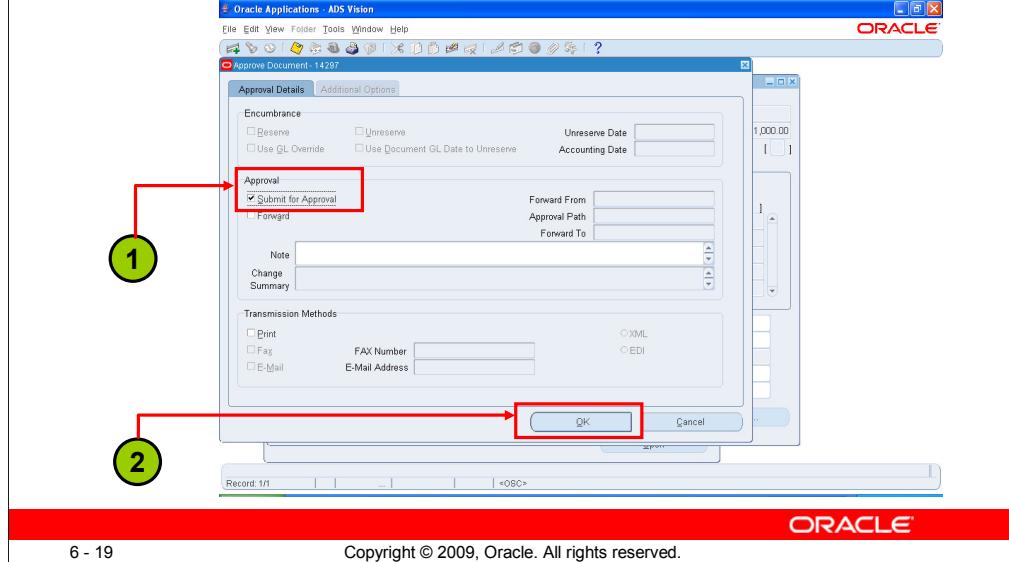
Copyright © 2009, Oracle. All rights reserved.

Create a Purchase Requisitions

- Enter Header Description = XXPurchases
- Select Lines Tab
- Enter Num = 10, Type = Goods, Item = CM12399, Rev = A, Quantity = 10, Price = 100, Need by = Today's Date
- Enter Destination Type = Inventory, Requester = Requester Name, Source = Supplier
- Click on Save (ctrl+s).
- Click on Approve Button to send a notification to supervisor for approval

Practice - PO: Enter a Purchase Requisition

- Create a Purchase Requisition



Create a Purchase Requisitions

1. Click the check box to Submit for Approval
2. Click OK Button to send the notifications to Supervisor for Approval

Note:

To see the status of the Purchase Requisition, Navigate to Purchasing Responsibility-> Requisitions-> Requisition Summary, Now Query the Purchase Requisition Number which is generated while creating the Requisitions.

Agenda

- Practice - PO: Create a Supplier
- Solution - PO: Create a Supplier
- Practice - PO: Enter a Purchase Requisition
- Solution - PO: Enter a Purchase Requisition
- **Practice - PO: AutoCreate a Purchase Order**
- **Solution - PO: AutoCreate a Purchase Order**
- Practice - RCV: Receive Purchased Material
- Solution - RCV: Receive Purchased Material

ORACLE®

6 - 20

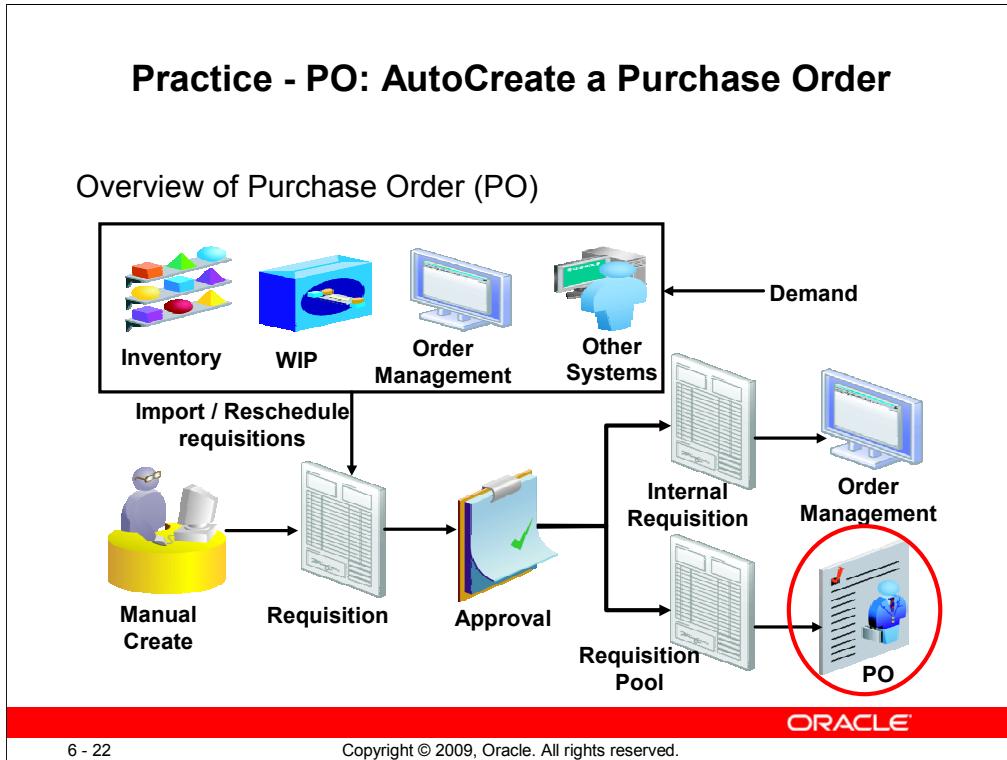
Copyright © 2009, Oracle. All rights reserved.

Practice - PO: AutoCreate a Purchase Order

Solution - PO: AutoCreate a Purchase Order

- Overview of Purchase Orders
- Create Purchase Orders Header
- Create Purchase Order Lines

ORACLE®



Purchase Orders

Demand for Goods or Services

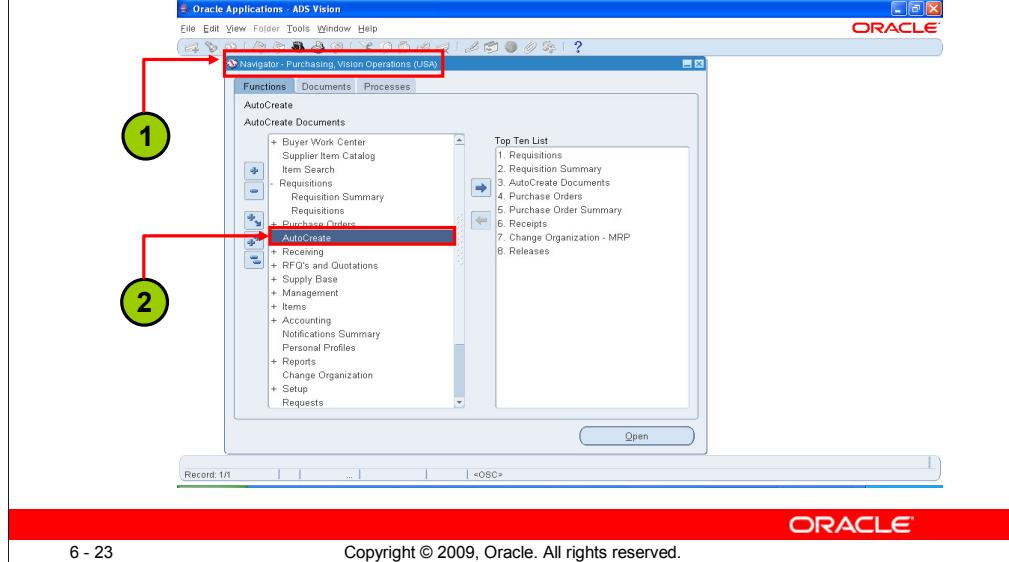
- Import requisitions from demand generated in Demand using Inventory, Work-in-Process (WIP), Order Management or other non-Oracle systems.
- Demand can also be generated manually through detail entry, with the assistance of requisition templates, or through the iProcurement module.

Sourcing

- Goods or services can be sourced from external suppliers through the creation of purchase orders or from within the organization through the creation of internal sales orders.

Practice - PO: AutoCreate a Purchase Order

- AutoCreate Purchase Order Navigation



Create a Purchase Requisitions

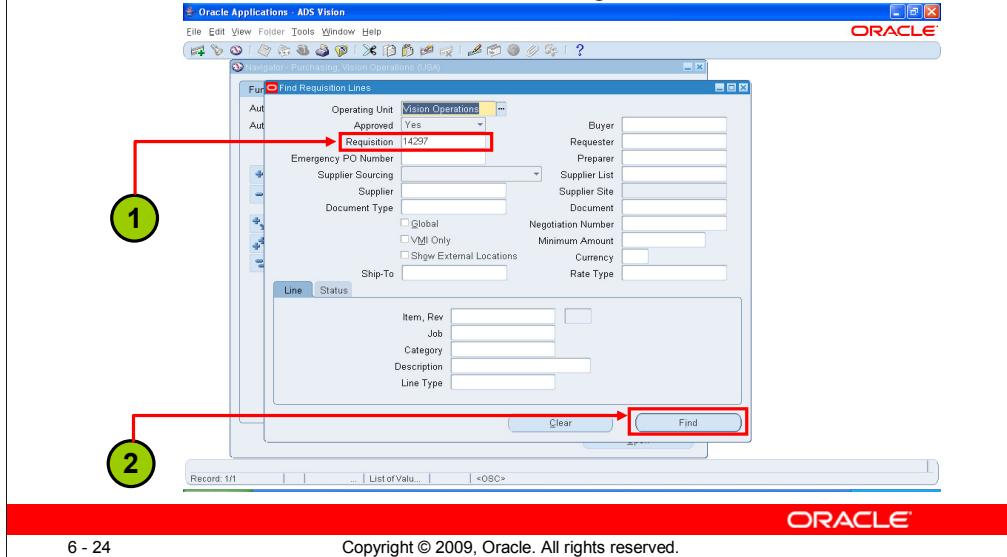
1. Navigate to Purchasing Responsibility
2. Select AutoCreate

Note:

In the previous practice for Creating a Purchase Requisitions, we have created a Purchase Requisition Number 14297. Use this Purchase Requisition Number for AutoCreate Purchase Order.

Practice - PO: AutoCreate a Purchase Order

- AutoCreate Purchase Order Navigation



Create a Purchase Requisitions

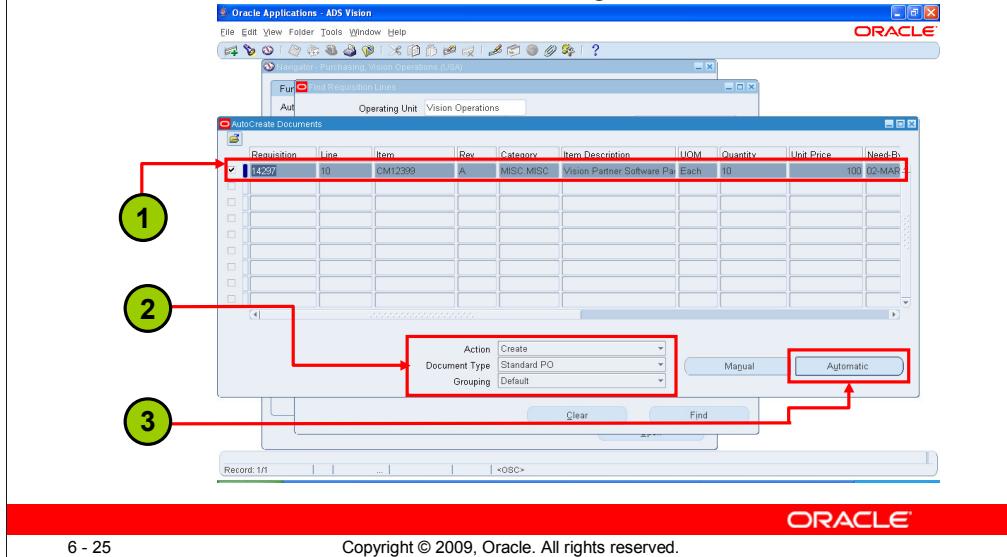
- Enter the Purchase Requisition Number = 14297
- Click on Find Button to query the Purchase Requisitions for AutoCreate Purchase Order.

Note:

In the previous practice for Creating a Purchase Requisitions, we have created a Purchase Requisition Number 14297. Use this Purchase Requisition Number for AutoCreate Purchase Order.

Practice - PO: AutoCreate a Purchase Order

- AutoCreate Purchase Order Navigation



Create a Purchase Requisitions

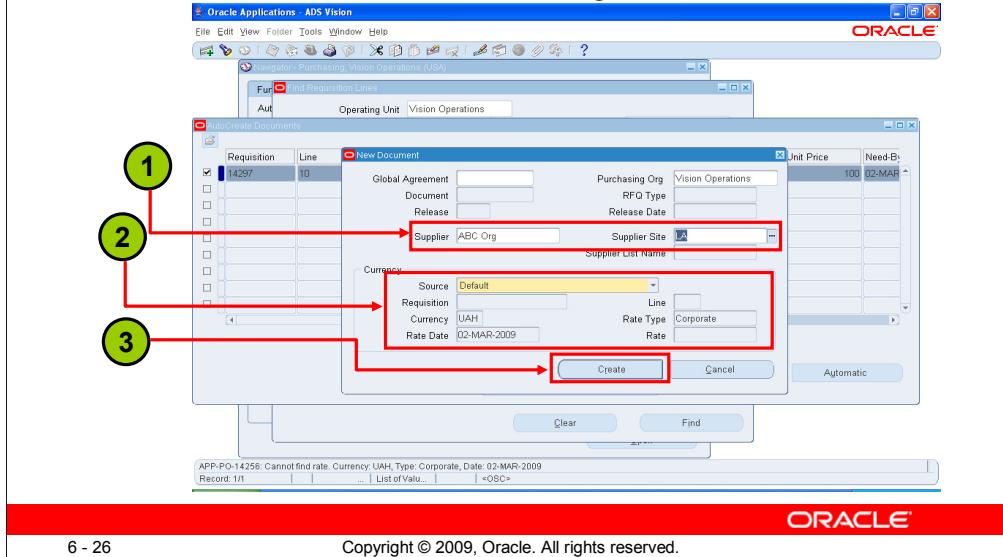
- Select the Purchase Requisitions.
- Select Action, Document Type, Grouping from the Options
- Click on Automatic Button to create the Purchase Order automatically based on Purchase Requisitions

Note:

In the previous practice for Creating a Purchase Requisitions, we have created a Purchase Requisition Number 14297. Use this Purchase Requisition Number for AutoCreate Purchase Order.

Practice - PO: AutoCreate a Purchase Order

- AutoCreate Purchase Order Navigation



Create a Purchase Requisitions

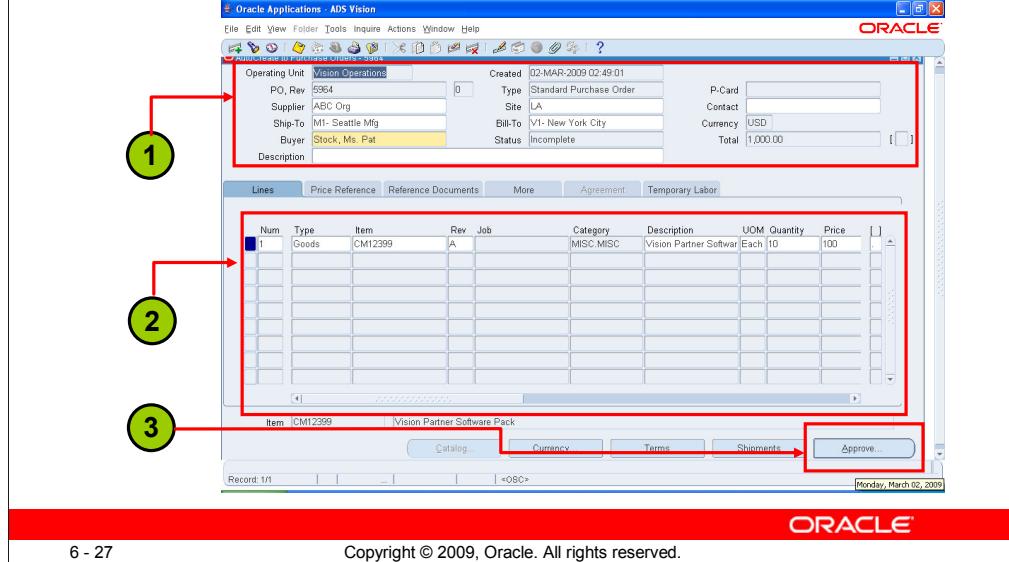
- Enter the Supplier Name and Supplier Site Information
- Enter Currency Information
- Click on Create Button to create the Purchase Order automatically based on Purchase Requisitions

Note:

In the previous practice for Creating a Purchase Requisitions, we have created a Purchase Requisition Number 14297. Use this Purchase Requisition Number for AutoCreate Purchase Order.

Practice - PO: AutoCreate a Purchase Order

- AutoCreate Purchase Order Navigation



6 - 27

Copyright © 2009, Oracle. All rights reserved.

ORACLE

Create a Purchase Requisitions

- Header Information of Purchase Order will be created automatically for Purchase Requisitions
- Lines Information of Purchase Order will be created Automatically for Purchase Requisitions
- Click on Approve Button to send a notification for Approval to the Supervisor.

Note:

In the previous practice for Creating a Purchase Requisitions, we have created a Purchase Requisition Number 14297. Use this Purchase Requisition Number for AutoCreate Purchase Order.

Agenda

- Practice - PO: Create a Supplier
- Solution - PO: Create a Supplier
- Practice - PO: Enter a Purchase Requisition
- Solution - PO: Enter a Purchase Requisition
- Practice - PO: AutoCreate a Purchase Order
- Solution - PO: AutoCreate a Purchase Order
- Practice - RCV: Receive Purchased Material
- Solution - RCV: Receive Purchased Material

ORACLE®

Practice - RCV: Receive Purchased Material

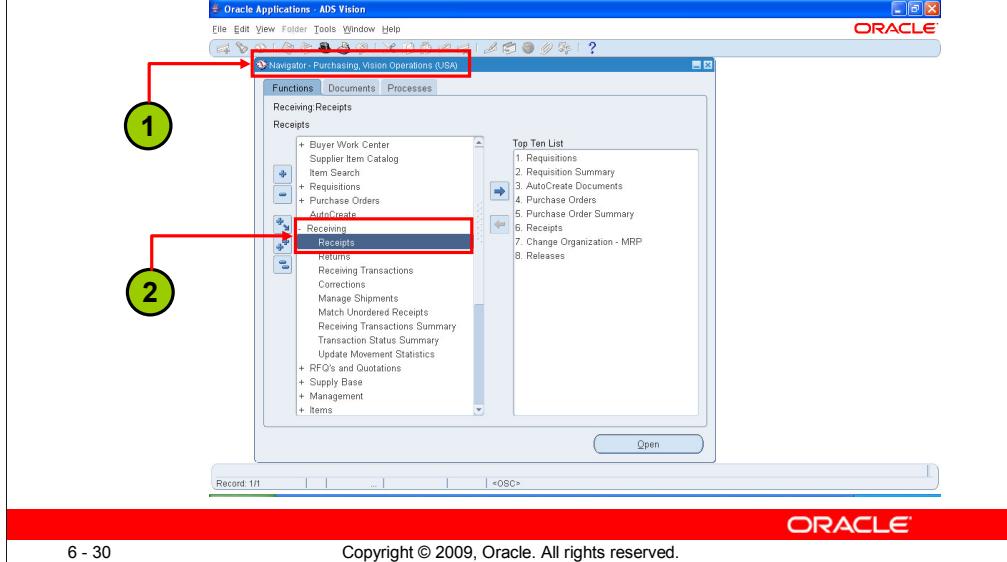
Solution - RCV: Receive Purchased Material

- Selecting Organization for Receiving Purchasing Order Materials
- Receiving Purchase Order Materials

ORACLE®

Practice - RCV: Receive Purchased Material

- Receive Purchase Order Material



Create a Purchase Requisitions

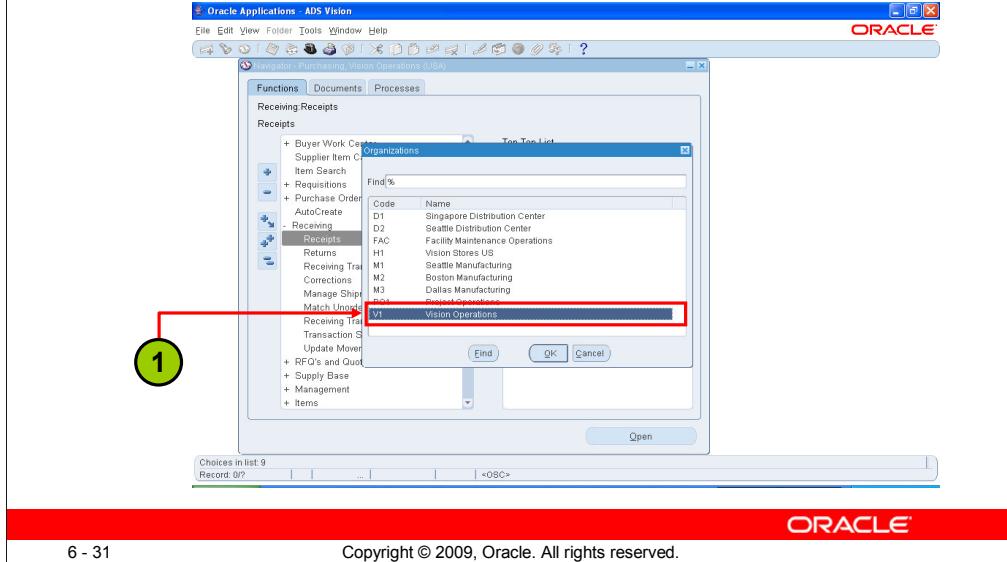
1. Navigate to Purchasing Responsibility
2. Select Receiving -> Receipt

Note:

In the previous practice for AutoCreate a Purchase Order, we have created a Purchase Order Number 5964, Use this Purchase Order Number to receive the Items from Supplier

Practice - RCV: Receive Purchased Material

- Receive Purchase Order Material



Create a Purchase Requisitions

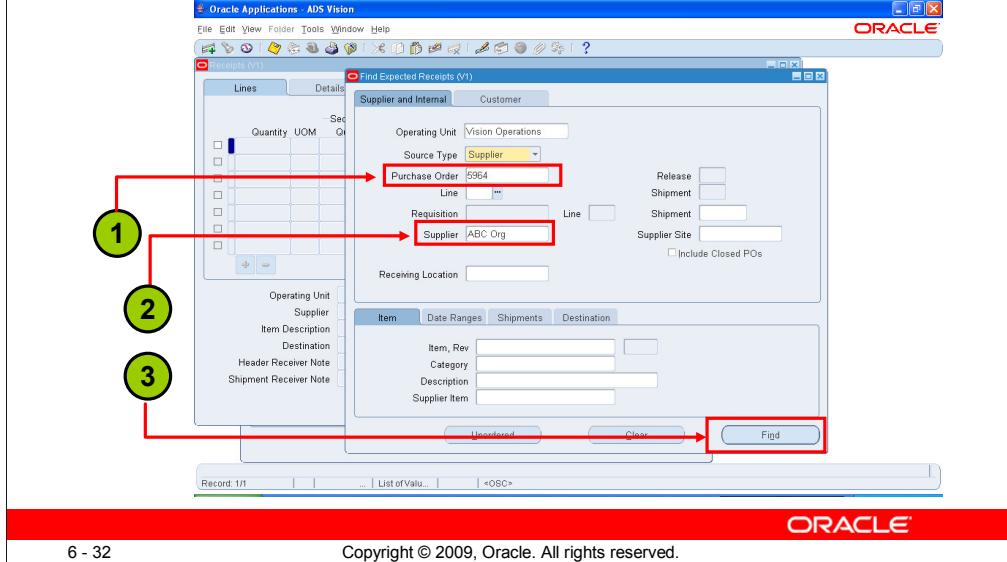
1. Select the Organization for receiving the Purchase Order Items

Note:

When you first access the Receipts window, you will be forced to choose an inventory organization from the list of values if you can process receipts for more than one. You can change the organization manually if you have access to the Change Organization window. Although it is not always the case. When you create purchase orders, for example, you specify goods to be shipped to specific inventory organizations. You will only be able to view expected receipts (scheduled shipments for a purchase order line item) for the inventory organization you choose. Use the Organization Access form to limit access to the list of inventory organizations for a specific responsibility.

Practice - RCV: Receive Purchased Material

- Receive Purchase Order Material



Create a Purchase Requisitions

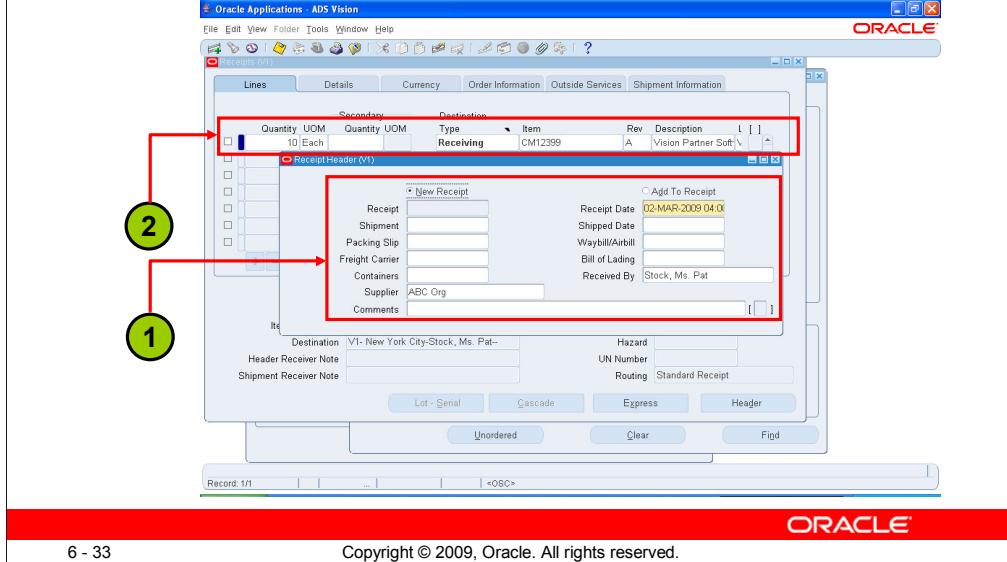
1. Query the Purchase Order Number
2. Enter the Supplier Name
3. Click on Find Button to search the Purchase Order for Receiving

Note:

In the previous practice for AutoCreate a Purchase Order, we have created a Purchase Order Number 5964, Use this Purchase Order Number to receive the Items from Supplier

Practice - RCV: Receive Purchased Material

- Receive Purchase Order Material



Create a Purchase Requisitions

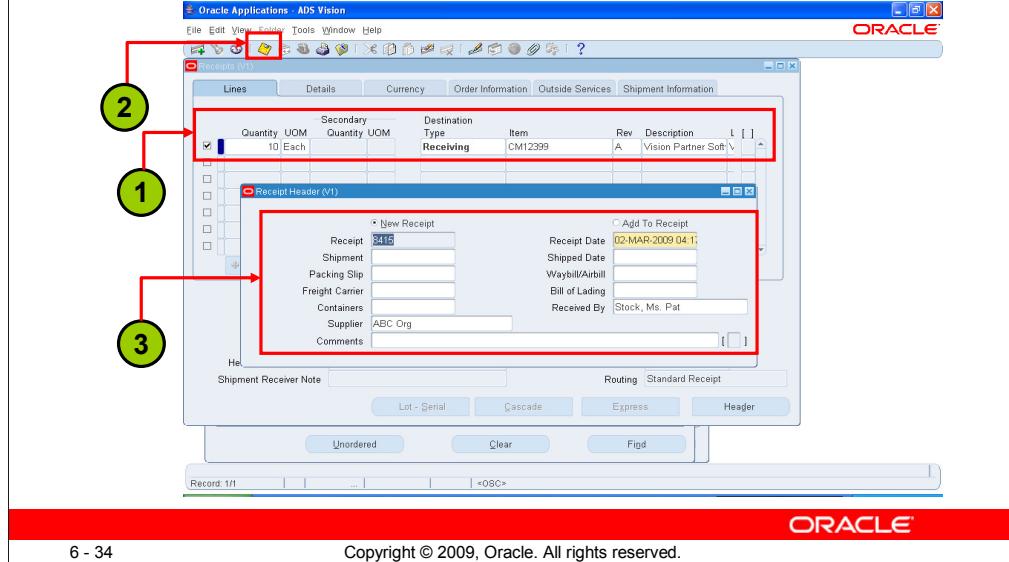
1. Receipt Header Information
2. Receipt Lines Information

Note:

In the previous practice for AutoCreate a Purchase Order, we have created a Purchase Order Number 5964, Use this Purchase Order Number to receive the Items from Supplier

Practice - RCV: Receive Purchased Material

- Receive Purchase Order Material



Create a Purchase Requisitions

1. Click the Check Box for Line Items (By default it will show Line Quantity from Purchase Order, If we are receiving less than the Line Quantity, We can change the value for Quantity)
2. Click on Save to generate the Receipt Number in Header Part
3. Receipt Number is generated automatically.

Note:

In the previous practice for AutoCreate a Purchase Order, we have created a Purchase Order Number 5964, Use this Purchase Order Number to receive the Items from Supplier

Summary

After completing this lesson, you should now be able to do the following:

- Create a Supplier
- Enter a Purchase Requisition
- Auto Create a Purchase Order
- Receive Purchased Material



List of Interface Tables

Use of eTRM

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

Objectives

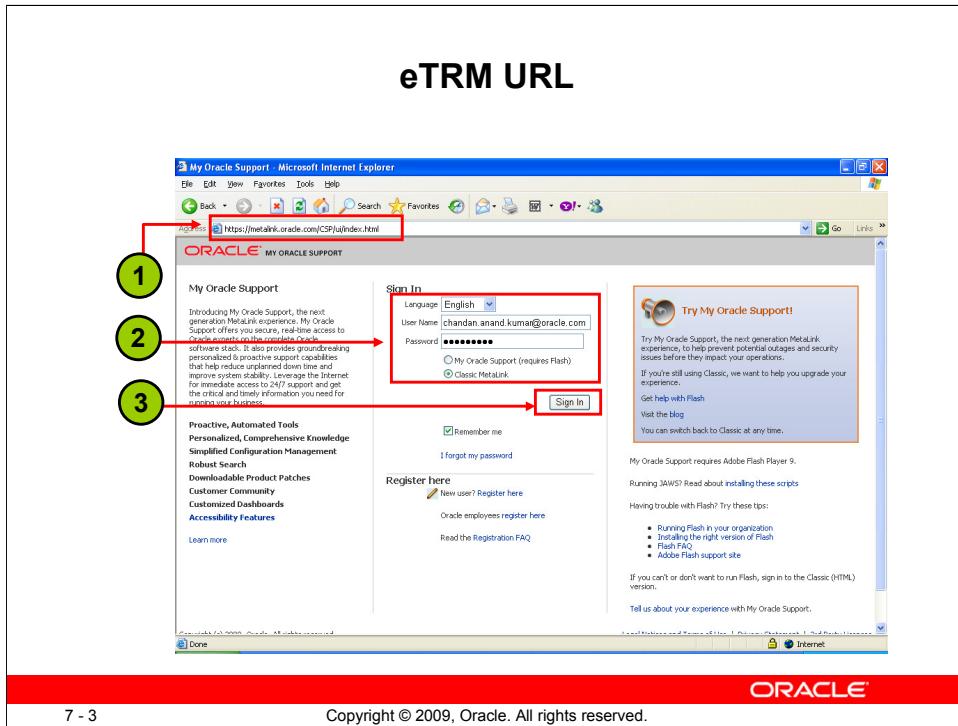
After completing this lesson, you should be able to understand

- How to use eTRM
- Searching tables and API's using eTRM

ORACLE®

7 - 2

Copyright © 2009, Oracle. All rights reserved.



Open eTRM using web browser

1. Enter “<https://metalink.oracle.com>” in URL area of web browser
2. Enter the login credential
3. Click on Sign In button to login into the Metalink

Knowledge Tab

The screenshot shows the Oracle Knowledge Browser interface. At the top, there's a navigation bar with links like 'My Oracle Support (the new MetaLink)', 'Bookmarks', 'Admin', 'Profile', 'Feedback', 'Sign Out', and 'Help'. Below the navigation bar, the main content area has tabs for 'Knowledge' (which is highlighted with a red box), 'Service Request', 'Collector', 'Patches & Updates', 'Community', and 'Certify'. The 'Knowledge' tab is currently active, showing a search bar with 'Quick Find' and 'Knowledge Base' options, and a dropdown menu for 'Alphabetical List of Categories'.

Product Categories

- Application Integration Architecture
- CollabSuite and Beehive
- Commerce Apps
- Databases
- Developer Suite
- E-Business Suite
- Enterprise Manager
- Fusion Middleware
- Insurance Applications
- Unbreakable Linux Software
- Networking
- Oracle VM
- Platform SysAdmin
- Taxation Management Apps
- Utilities Apps

In the Knowledge

Getting your Grid Control Environment Ready for the Upcoming Daylight Saving Time (DST) - Spring 2009 - February 23, 2009

In order to avoid any issue related to the Daylight Saving Time (DST) changes in Spring 2009, you need to proactively check to ensure your Grid Control components installed and running are DST compliant. If you have Grid Control OMS and/or Agents located in the affected countries, please review [Note 73981.8.1 - Is your Grid Control ready for the next Daylight Saving Time \(DST\) - spring 2009?](#) for details.

The countries affected by DST are as follows:

United States Of America, Canada, Australia, New Zealand, Brazil, Venezuela, Argentina, Egypt, Chile, Pakistan, Morocco, Chobalan.

Japanese Knowledge Base (Disk)

Japanese Knowledge Base Available via OSC. Oracle Internet Support Center (OISC) requires a separate account. Access is restricted for customers who have a current support contract with Oracle Japan.

Customer Knowledge Exchange

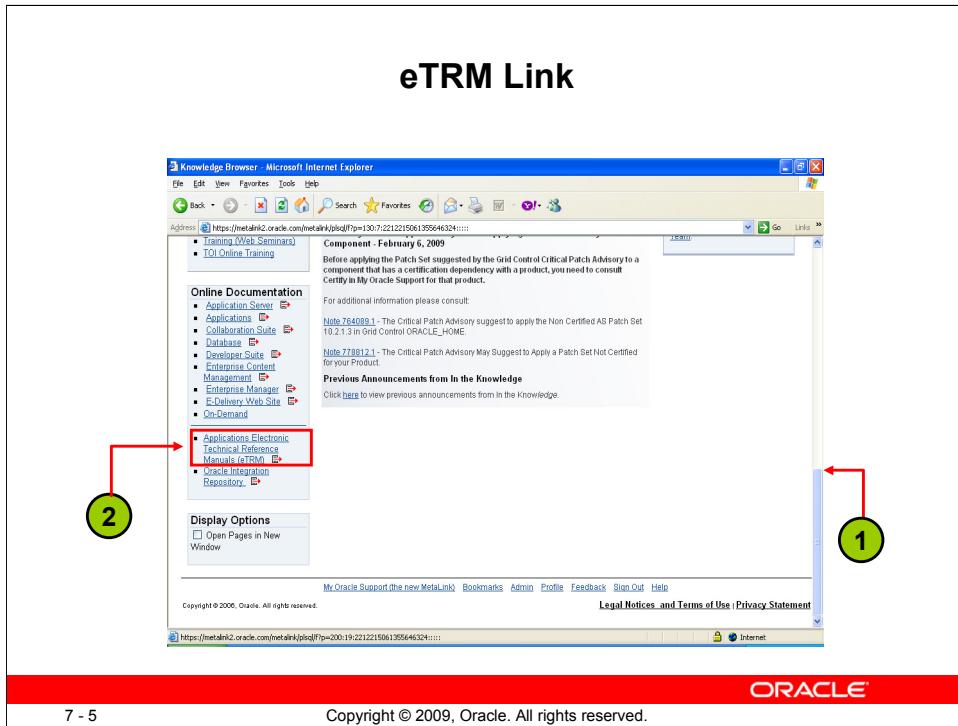
The Customer Knowledge Exchange provides a channel for customers to showcase their expertise and to share their knowledge about Oracle products and services with others to the [Customer Knowledge Exchange](#).

ORACLE

7 - 4 Copyright © 2009, Oracle. All rights reserved.

Open eTRM using web browser

1. Click on Knowledge tab



Open eTRM using web browser

1. Scroll down the web browser
2. Click on Applications Electronic Technical Reference Manuals (eTRM)
3. After click on eTRM a new web page will open for eTRM

eTRM Page

Oracle eTRM - Microsoft Internet Explorer

Address: https://etrm.oracle.com/pls/etrmktest/etrm_search.search

ORACLE®
eTRM Technical Reference

eTRM Version 12.0.0

Oracle eBusiness Suite Electronic Technical Reference Manual - eTRM

Search results for AP_INVOICES_INTERFACE

AP_INVOICES_INTERFACE

AP_INVOICES_INTERFACE
TABLE in the AP schema.

SYNONYM in the APPS schema.

AP_INVOICE_LINES_INTERFACE
Information used to create one or more invoice distributions

ECE_AP_INVOICES_INTERFACE_X
Extension table for ECE_AP_INVOICES_INTERFACE

MAP_INTERFACE_REJECTIONS
Information about data that could not be loaded by Payables Open Interface Import

AP_INVOICE_LINES_INTERFACE
Information used to create one or more invoice distributions

ECE_AP_INVOICES_INTERFACE_X
Extension table for ECE_AP_INVOICES_INTERFACE

MAP_INTERFACE_REJECTIONS
Information about data that could not be loaded by Payables Open Interface Import

1
2
3

7 - 6 Copyright © 2009, Oracle. All rights reserved.

Open eTRM using web browser

1. Select the Application version.
2. Enter the table or API name in the search field and then click on Search Button
3. All the database object related to the table \ API name will be displayed.

eTRM Search Result

The screenshot shows a Microsoft Internet Explorer window displaying the DBA Data page for the table AP_AP_INVOICES_INTERFACE. The page includes object details, a description of the table, and storage details. A red box highlights the table name 'TABLE: AP_AP_INVOICES_INTERFACE'. A green circle with the number '1' points to this red box. Another red box highlights the descriptive text about the table's purpose and data sources. A green circle with the number '2' points to this red box.

Object Details

- Object Name: AP_INVOICES_INTERFACE
- Object Type: TABLE
- Owner: AP
- FND Design Date: 10/10/2008
- Subobject Name:
- Status: VALID

Description

AP_INVOICES_INTERFACE stores header information about invoices that you create or load for import. Invoice data comes from sources including EDI invoices from your suppliers that you load through Oracle e-Commerce Gateway, supplier invoices that you transfer through the Oracle XML Gateway, invoices that you load using Oracle SQL*Loader, lease invoices from Oracle Property Manager, lease payments from Oracle Assets, credit card transaction data that you load using the Credit Card Invoice Interface Summary, and invoices that you enter through the Invoice Gateway. There is one row for each invoice you import. Your Oracle Payables application uses this information to create invoice header information when you submit the Payables Open Interface program.

Storage Details

Done ORACLE Copyright © 2009, Oracle. All rights reserved.

Open eTRM using web browser

Search result showing

1. Table name
2. Description of the table

eTRM Search Result (Cont..)

Indexes

Name	Type	Uniqueness	Tablespace	Column
AP_INVOICES_INTERFACE_U1	NORMAL UNIQUE	APP_TS_INTERFACE	INVOICE_ID	
AP_INVOICES_INTERFACE_N1	NORMAL NONUNIQUE	APP_TS_INTERFACE	STATUS	
AP_INVOICES_INTERFACE_N2	NORMAL NONUNIQUE	APP_TS_INTERFACE	SOURCE	
			GROUP_ID	

Columns

Name	Datatype	Length	Mandatory	Comments
INVOICE_ID	NUMBER	(15)	Yes	Invoice identifier
INVOICE_NUM	VARCHAR2	(50)		Invoice number
INVOICE_TYPE_LOOKUP_CODE	VARCHAR2	(25)		Type of invoice (can be STANDARD or CREDIT)
INVOICE_DATE	DATE			Invoice date
PO_NUMBER	VARCHAR2	(20)		Purchase order number
VENDOR_ID	NUMBER	(15)		Supplier identifier. Validated against PO_VENDORS.VENDOR_ID
VENDOR_NUM	VARCHAR2	(30)		Supplier number
VENDOR_NAME	VARCHAR2	(240)		Supplier name
VENDOR_SITE_ID	NUMBER	(15)		Supplier site identifier. Validated against PO_VENDOR_SITES_ALL.VENDOR_SITE_ID
INVOICE_SITE_CODE	VARCHAR2	(15)		Supplier site code
INVOICE_AMOUNT	NUMBER			Currency of invoice. Validated against FND_CURRENCIES.CURRENCY_CODE
INVOICE_CURRENCY_CODE	VARCHAR2	(15)		Exchange rate for foreign currency invoices
EXCHANGE_RATE	NUMBER			Exchange rate type for foreign currency invoices. Validated against GL_DAILY_CONVERSION_TYPES.CONVERSION_TYPE
EXCHANGE_RATE_TYPE	VARCHAR2	(30)		Date exchange rate is effective, usually accounting date of a transaction
EXCHANGE_DATE	DATE			Payment terms identifier. Validated against AP_TERMS_LT.TERM_ID
TERMS_NAME	VARCHAR2	(50)		Payment terms name
TERMS_CODE	VARCHAR2	(50)		Invoice description
DESCRIPTION	VARCHAR2	(240)		Withholding tax group identifier. Validated against AP_AWT_GROUPS.AWT_GROUP_ID
AWT_GROUP_ID	NUMBER	(15)		Withholding tax group name
AWT_GROUP_NAME	VARCHAR2	(25)		Standard Who column - date when a user last updated this row
LAST_UPDATE_DATE	DATE			

ORACLE®

7 - 8 Copyright © 2009, Oracle. All rights reserved.

Open eTRM using web browser

1. Table Indexes
2. Column descriptions.

Summary

After completing this lesson, you should be able to understand

- How to use eTRM
- Searching Tables and API's using eTRM

ORACLE®

7 - 9

Copyright © 2009, Oracle. All rights reserved.

R12 Interfaces and API's - IBM Graduate Program

Case Study

D81888GC20

Edition 2.0

August 2013

ORACLE®

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Trademark Notice

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

Migration of AP Invoices using Open Interface

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Data Dependencies.....	3
Business Rules	3
Approach.....	4
Module List.....	5
Registering Concurrent Program	9
Downloading Concurrent Program Script	18
Deployment.....	19
Extract File Layout	20
Extract File Mapping to Staging Table Columns	21
Calling Custom Concurrent Program.....	22
Calling Standard Import Concurrent Program.....	23
Query interface records.....	24
Error Handling	25
Open and Closed Issues for this Deliverable	26
Open Issues	26
Closed Issues.....	26

Introduction

The AP Invoice has been designed to assist the accounts payable department in the organization of vendor invoices for approval and payment. It can be of Standard, Credit Memo, and Debit Memo, PrePayment, Expense Report etc.

Use the Payables Invoice window to enter the standard invoices, debit memos, credit memos, Expense Report etc. You can also query and update supplier invoices in this window.

To enter or query an AP Invoices from front end navigate to Payables Responsibility, Select **Invoice -> Entry -> Invoices**.

This document demonstrates the use of open interface table for AP Invocies to migrate AP Invoices from source system to Oracle Applications.

Purpose

This document describes the:

- This case study is intended to demonstrate the creation of Standard Invoices Using Oracle Payables Interface table.
- Detailed data mapping from the source system(s) to the Oracle Applications E-Business Suite (EBS) **Payables** (AP) Transactions records
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during implementation of oracle Payables.

Scope and Application

The following boundaries are specific to the Interfaces of AP Invoices using **AP_INVOICES_INTERFACE** and **AP_INVOICE_LINES_INTERFACE** open interface table

- All Invoices, Credit & Debit Memos would be migrated to Oracle Applications

- The ‘XX<EMPID>: AP Invoice Interface Program’ uploads AP Invoices data from data file into staging table “[XXAR_INVOICE_IFACE_STG](#)” and will perform validation and transfer valid records into Oracle Payables interface tables. The standard import program ‘[Payables Open Interface Import](#)’ is then run to perform the validation on data in the interface table and load valid data into base tables.
 1. The AP Invoices data will be provided through data files. This data file will be loaded into staging table(s) on the Oracle applications database using loader program.
 2. The loaded data will be validated from staging tables and then moved into the standard transaction interface tables.
 3. The standard [Payables Open Interface Import](#) program will be called to import the invoices from Interface table to Oracle payables base tables.

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.

Source system

The implementation team studies the source system and understands the data required to be migrated to Oracle Applications. This data is then extracted and cleansed from the source system (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team. The data mapping section describes the data format of source system to Oracle Applications.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Payable Manager
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.
- Users have access to Oracle Applications database ([apps schema](#))

- All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.
- The extract-file layout must be in the format as specified in the extract file layout section

Pre-Requisites

Prerequisite set-ups are required for the interface. These setups include, but are not limited to:

1. Payables System Options
2. Payment Terms
3. Invoice Lookup Types
4. Invoice Sources
5. Distribution Set Rules (Must be setup a distribution set for PUB)
6. Currency
7. System Profile Options for Oracle Payables
8. Tax Codes and Rates

Data Dependencies

AP Invoices

In order to migrate AP Invoices the following entities must have been migrated successfully:

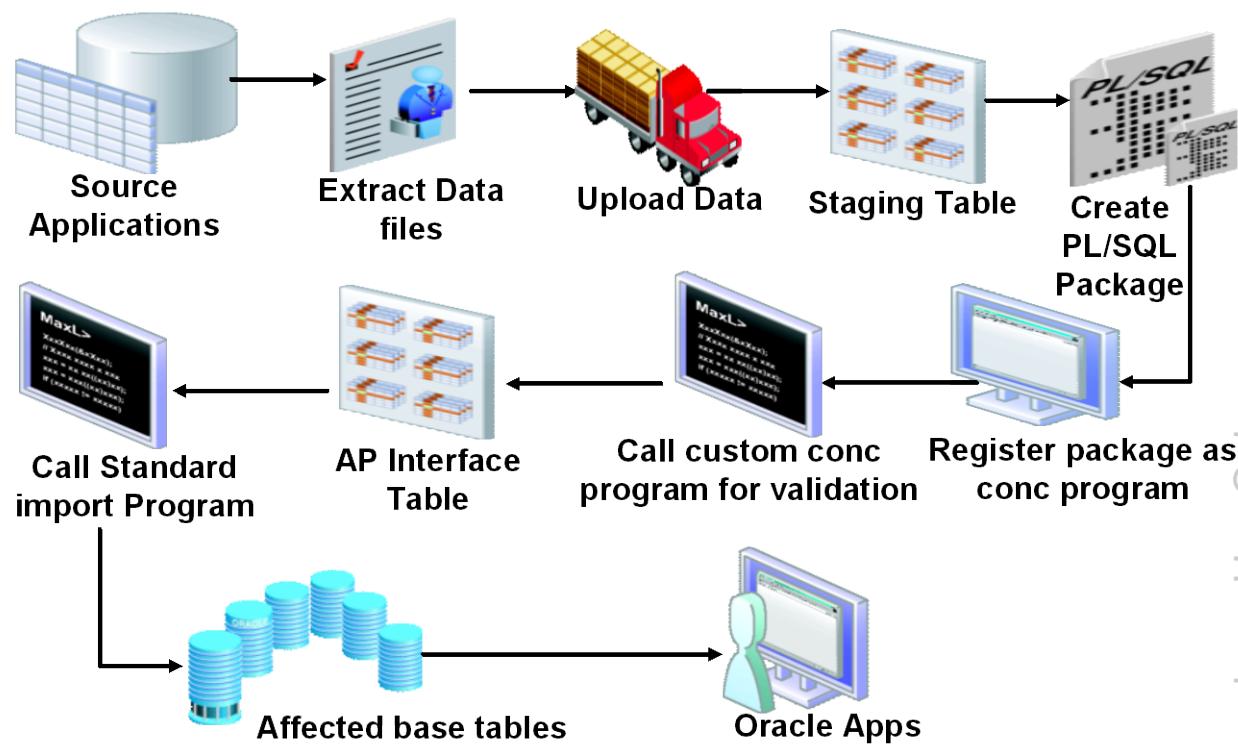
- Suppliers

Business Rules

Following business rules are applicable for the **AP Invoices** conversion process.

- N/A

Approach



The Approach for migrating **AP Invoices** data from source system to Payables open interface table

1. Extract the data file from source system
2. Load the data file into staging table
3. Using PL/SQL write a package
4. Register the PL/SQL package as concurrent program in Oracle Applications
5. Call the custom concurrent program for validation and loading the records available in staging table to open interface table
6. Call the standard import program, which will create the records in AP_INVOICES_ALL, AP_INVOICE_LINES_ALL and AP_INVOICE_DISTRIBUTIONS_ALL tables based on open interface records.
7. Query the records, which got created for interface table and see the result in Oracle Applications front end.

Module List

Concurrent Programs

‘XX<EMPID>: AP Invoices Interface Program’ includes the following concurrent programs:

‘XX<EMPID>: AP Invoices Interface Program’

The ‘XX<EMPID>: AP Invoices Interface Program’ concurrent program is based on the stored procedure **XXINT_AP_INVOICE_PKG.main** that is registered as a PL/SQL executable.

- The above program calls loader program (**XX<EMPID>: AP Invoices Interface Loader Program**) to load data into staging tables, performs validations on the staging table data and uploads data into the open interface tables (AP_INVOICES_INTERFACE and AP_INVOICE_LINES_INTERFACE).

‘XX<EMPID>: AP Invoices Interface Loader Program’

The ‘XX<EMPID>: AP Invoices Interface Loader Program’ concurrent program is based on the SQL* Loader control file, that is registered as a Loader executable.

- The above program loads data from the path specified as parameter value in loader concurrent program to the staging table.

NOTE:

1. Change the <EMPID> with your Employee ID,
2. Follow the steps for registering concurrent program from Registering Concurrent Program section.

Stored Procedures

XXINT_AP_INVOICE_PKG package consist the following stored procedures:

XXINT_AP_INVOICE_PKG.main

This is the main procedure that calls the load, validate, import, print_error and record_summary procedures.

XXINT_AP_INVOICE_PKG.validate

This procedure performs the necessary validations on the staging tables.

XXINT_AP_INVOICE_PKG.import

This procedure loads all the valid data from the staging table to Oracle Invoices interface tables AP_INVOICES_INTERFACE and AP_INVOICE_LINES_INTERFACE.

XXINT_AP_INVOICE_PKG.print_error

This procedure reports the errored records in the log file.

XXINT_AP_INVOICE_PKG.record_summary

This procedure will display the status of staging table records

NOTE: Please refer the attached documents for Package Specifications and Package body. Kindly open the attached script in notepad.



XX_AP_INVOICE_PK
G.pks

1. Package Specification:



XX_AP_INVOICE_PK
G.pkb

2. Package Body:

Staging Table

XXAP_INVOICE_IFACE_STG table created in database to store records from source system data file.

NOTE: Please refer the attached script staging table creation. Kindly open the attached script in notepad.



XX_AP_INVOICE_TA
B.tab

Table Creation Script:

Grant Script

If table and package are created in custom schema then grant the table and package to APPS schema

NOTE: Please refer the attached script for table and package granting. Kindly open the attached script in notepad.



XX_AP_INVOICE_PK
G.grt



XX_AP_INVOICE_TA
B.grt

1. Package Grant Script:

2. Table Grant Script:

Synonym Script

If table and package are created in custom schema then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



XX_AP_INVOICE_PK
G.syn

1. Package synonym script:



XX_AP_INVOICE_TA
B.syn

2. Table synonym script:

SQL* Loader Script

SQL*Loader is controlled by its own data definition language, which is kept in the control file. The control file describes the data to be loaded, the destination tables of the data and describes the interdependency between the data and the columns within the tables. Which in-turn used to register as a concurrent program in Oracle Applications to load the data file into staging table.

NOTE: Please refer the attached SQL* Loader script for loading data file into staging table. Kindly open the attached script in notepad.



XX_AP_INVOICE_LO
AD.ctl

1. SQL* Loader Script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script
3. Staging Table Script
4. SQL* Loader Script

NOTE: Please refer the attached Install script for deployment of AP Invoices object on Oracle Applications Server. Kindly open the attached script in notepad.

1. Install Script:



Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

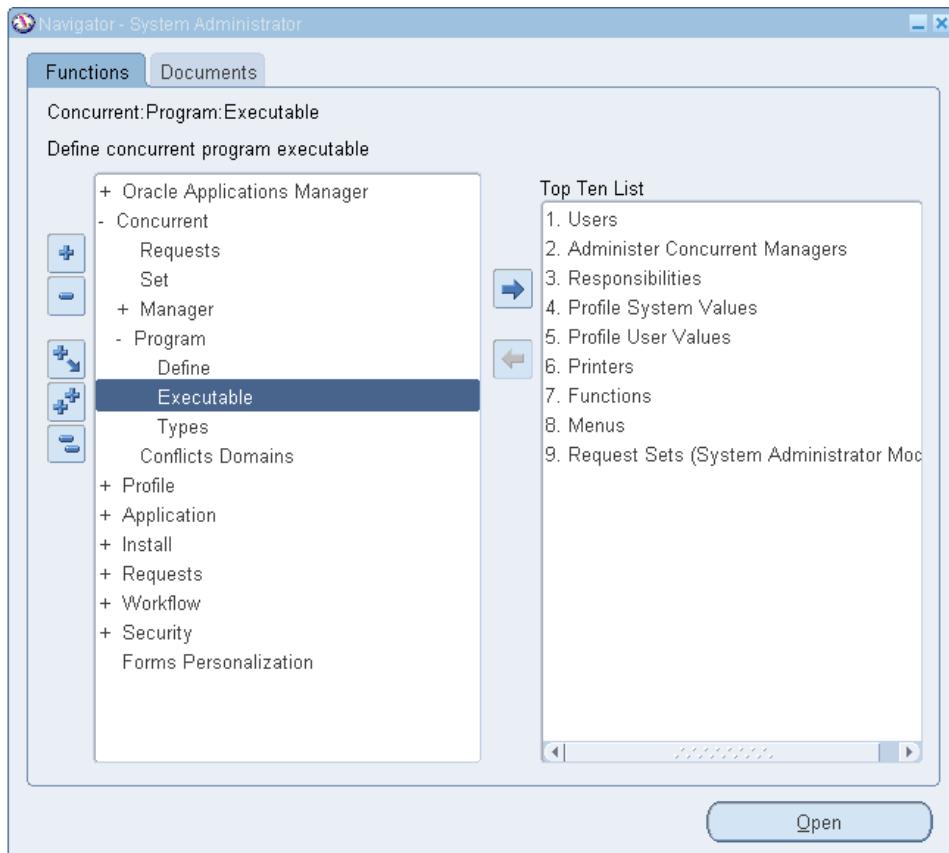
- A.) Package Name
- B.) Table Name
- C.) Synonym Name
- D.) Concurrent Program Executable.
- E.) Concurrent Program Name.

Registering Concurrent Program

Registering Interface Loader Concurrent Program

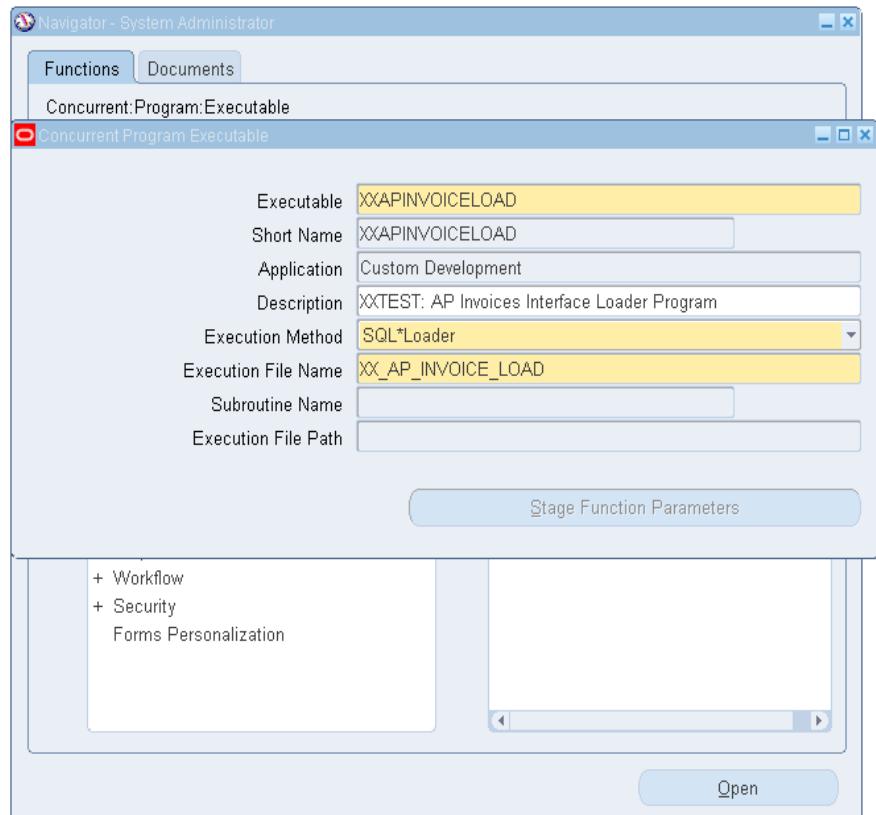
To register a concurrent program for '[XX<EMPID>: AP Invoices Interface Loader Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



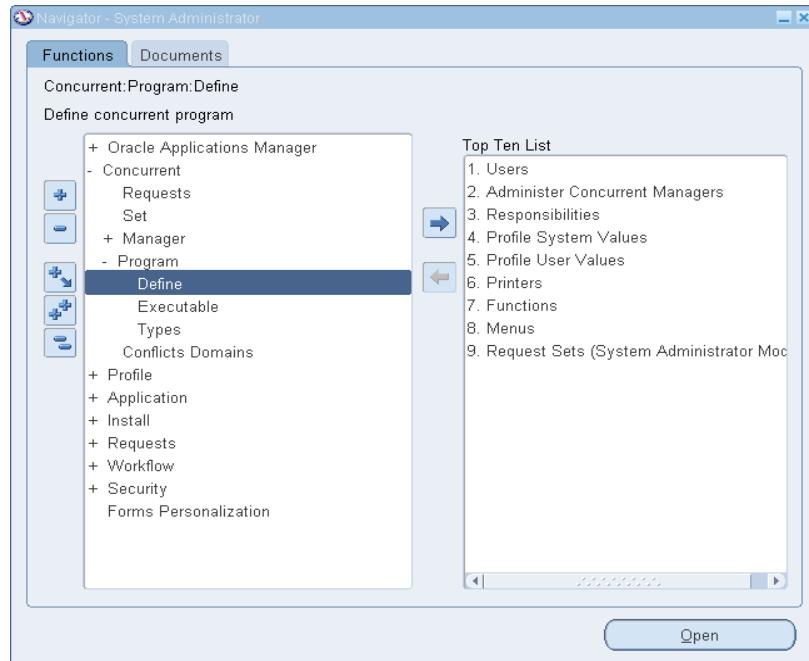
2. Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for AP Invoices so the Application Name should be [Custom Application](#).
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be SQL*Loader, because we are registering a concurrent program for loading the data to staging table.
- Execution File Name:** Here we would be passing the file name of SQL*Loader control file. As discussed in Module List the control file name would be [XX_AP_INVOICE_LOAD](#)



Executable Registering Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name: Enter “XX<EMPID>: AP Invoices Interface Loader Program”
- Short Name: Enter a unique concurrent program short name, for example: **XXAPINVOICELOAD**
- Application: Select the value from LOV, as we would be executing this concurrent program from Payables Applications so select Payables.
- Description: Enter the description of this concurrent program.
- Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXAPINVOICELOAD** from LOV.
- Executable Method: Executable method will be populated automatically.
- Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A). Data File

The screenshot shows the 'Concurrent Programs' dialog box with the following settings:

- Program:** XXTEST: AP Invoices Interface Loader Program (Enabled)
- Short Name:** XXAPINVOICELOAD
- Application:** Custom Development
- Executable:**
 - Name: XXAPINVOICELOAD
 - Method: SQL*Loader
 - Options: Priority (unchecked)
- Request:**
 - Type: (empty)
 - Incrementor: (empty)
 - MLS Function: (empty)
 - Checkboxes:
 - Use in SRS (checked)
 - Run Alone (unchecked)
 - Enable Trace (unchecked)
 - Allow Disabled Values (unchecked)
 - Restart on System Failure (checked)
 - NLS Compliant (checked)
- Output:**
 - Format: Text (selected)
 - Checkboxes:
 - Save (C) (checked)
 - Print (checked)
 - Style Required (unchecked)
 - Columns: (empty)
 - Rows: (empty)
 - Style: (empty)
 - Printer: (empty)

Buttons at the bottom: Copy to..., Session Control, Incompatibilities, Parameters.

Concurrent Program Form

Concurrent Program Parameters

Program: XXTEST: AP Invoices Interface Loader Program
Application: Custom Development

Conflicts Domain: [] Security Group: []

Seq	Parameter	Description	Enabled
10	Data_File	Data file name with path	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>

Validation:

- Value Set: 100 Characters
- Default Type: [] Description: 100 Characters
- Required
- Enable Security
- Range: []

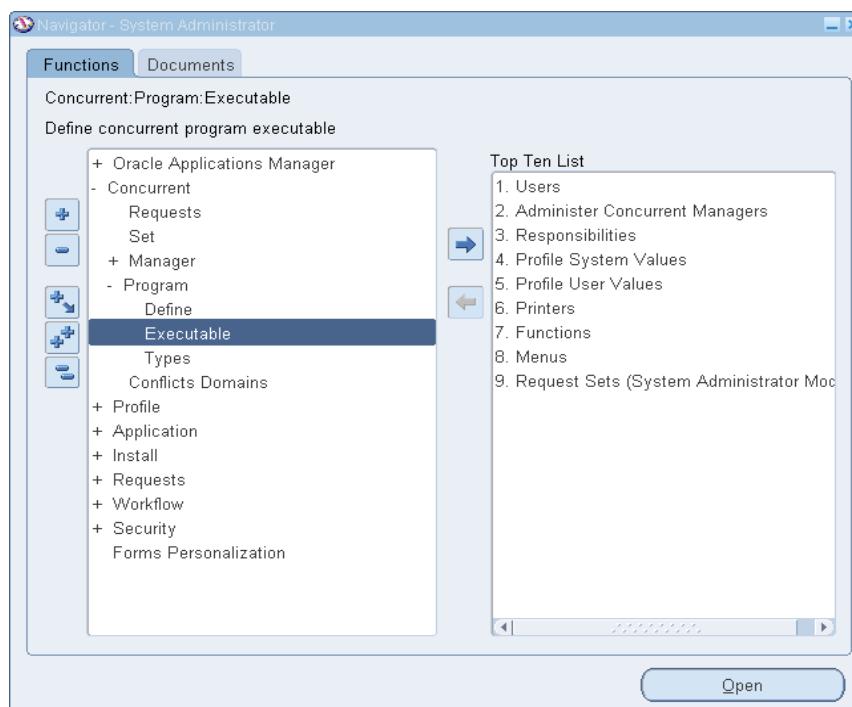
Display:

- Display Size: 50
- Concatenated Description Size: 25
- Description Size: 50
- Prompt: Data File

Token: []

Concurrent Program Parameters Form**Registering Interface Concurrent Program**

To register a concurrent program for '[XX<EMPID>: AP Invoices Interface Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable

2 Enter The Following in the Concurrent Program Executable Form

- a. **Executable Name:** Enter unique name for executable.
- b. **Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- c. **Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for AP Invoices so the Application Name should be **Custom Application**.
- d. **Description:** Enter the description of Executable
- e. **Execution Method:** Execution method should be PL/SQL Stored Procedure.
- f. **Execution File Name:** Here we would be passing the package name with one public procedure. As discussed in Module List the package name would be **XXINT_AP_INVOICE_PKG** and the public procedure defined in this package is **main**.

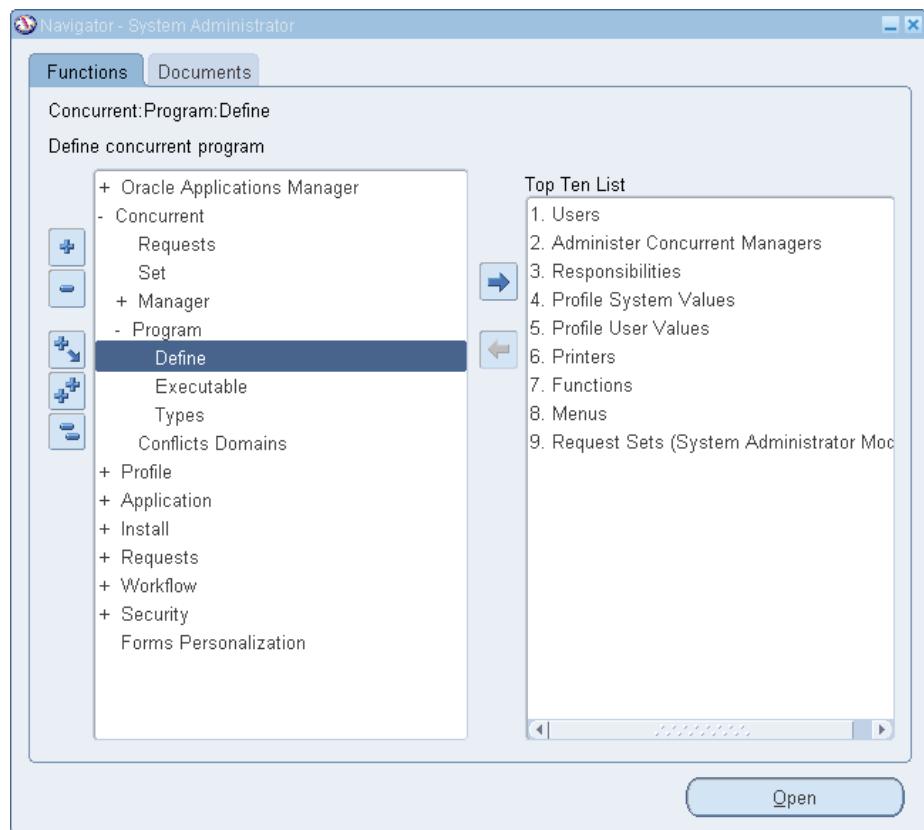
The screenshot shows the Oracle Applications Manager interface for defining a concurrent program executable. The window title is "Navigator - System Administrator". The top menu bar has "Functions" and "Documents" tabs, with "Functions" selected. The main area is titled "Concurrent: Program: Executable" and contains the sub-section "Define concurrent program executable". Below this, there is a toolbar with "Create Applications Manager" and "Top Ten List" buttons. The main configuration area has the following fields:

Executable	XXAPINVOICEMAIN
Short Name	XXAPINVOICEMAIN
Application	Custom Development
Description	(empty)
Execution Method	PL/SQL Stored Procedure
Execution File Name	XXINT_AP_INVOICE_PKG.main
Subroutine Name	(empty)
Execution File Path	(empty)

At the bottom right of the configuration area is a "Stage Function Parameters" button. At the very bottom of the window are "Forms Personalization" and "Open" buttons.

Concurrent Program Executable Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- a. Program Name: Enter “XX<EMPID>: AP Invoices Interface Program”
- b. Short Name: Enter a unique concurrent program short name, for example: **XXAPINVOICEMAIN**
- c. Application: Select the value from LOV, as we would be executing this concurrent program from Payables Applications so select Custom Development and add this concurrent program to the Payables Request Group.
- d. Description: Enter the description of this concurrent program.
- e. Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXAPINVOICEMAIN** from LOV.
- f. Executable Method: Executable method will be populated automatically.
- g. Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A) Data File Path B) Data File Name

The screenshot shows the 'Concurrent Programs' window with the following details:

- Program:** XXTEST: AP Invoices Interface Program
- Short Name:** XXAPINVOICEMAIN
- Application:** Custom Development
- Description:** (empty)
- Executable:**
 - Name:** XXAPINVOICEMAIN
 - Method:** PL/SQL Stored Procedure
 - Options:** (empty)
 - Priority:** (empty)
- Request:**
 - Type:** (empty)
 - Incrementor:** (empty)
 - MLS Function:** (empty)
 - Checkboxes:**
 - Use in SRS
 - Run Alone
 - Enable Trace
 - Allow Disabled Values
 - Restart on System Failure
 - NLS Compliant
- Output:**
 - Format:** Text
 - Save (C)
 - Print
 - Columns:** (empty)
 - Rows:** (empty)
 - Style:** (empty)
 - Style Required
 - Printer:** (empty)

Buttons at the bottom: Copy to..., Session Control, Incompatibilities, Parameters.

Concurrent Program Form

The screenshot shows the 'Concurrent Program Parameters' window with the following details:

- Program:** XXTEST: AP Invoices Interface Program
- Application:** Custom Development
- Conflicts Domain:** (empty)
- Security Group:** (empty)
- Parameter List:**

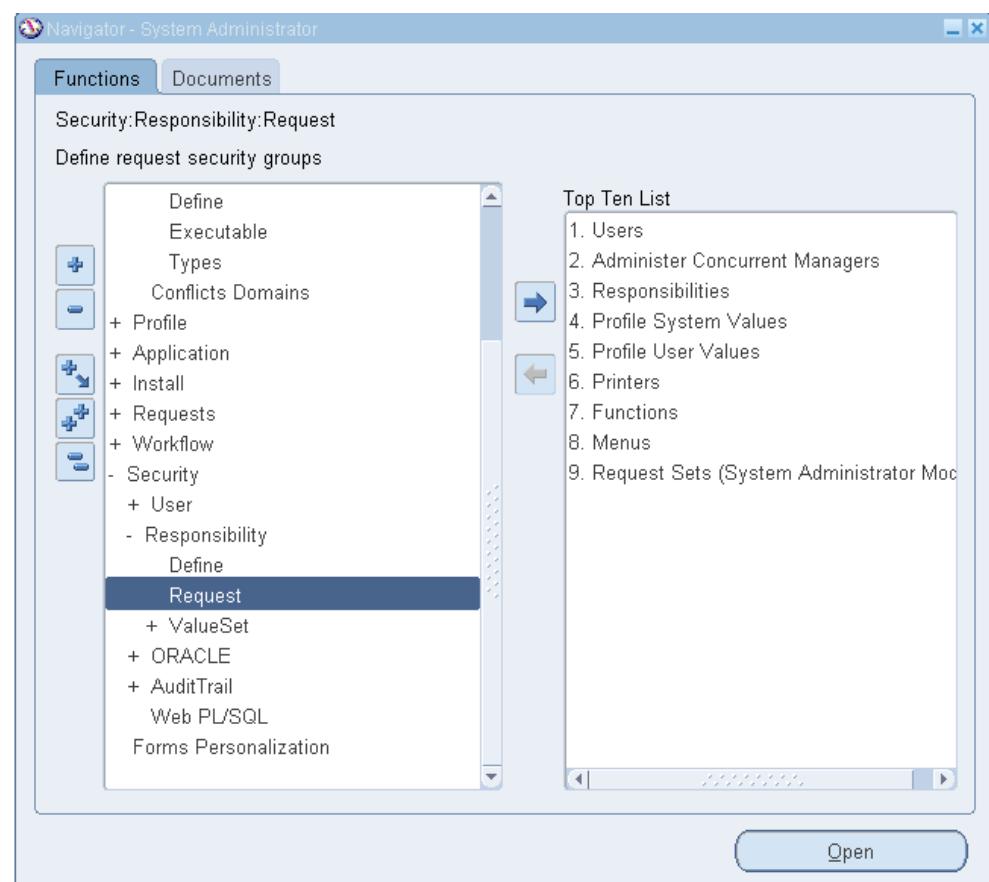
Seq	Parameter	Description	Enabled
10	Data_File_Path	Data File Path	<input checked="" type="checkbox"/>
20	Data_File_Name	Data file name	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
- Validation:**
 - Value Set:** 100 Characters
 - Default Type:** (empty)
 - Required:**
 - Enable Security:**
 - Description:** 100 Characters
 - Default Value:** (empty)
 - Range:** (empty)
- Display:**
 - Display Size:** 50
 - Concatenated Description Size:** 25
 - Description Size:** 50
 - Prompt:** Data File Path
 - Token:** (empty)

Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: AP Invoices Interface Program’ in Request Group, Navigate to [System Administrator Responsibility](#)

1. Select Security -> Responsibility-> Request



2. Query the Request Group – All Reports and Applications as Payables

The screenshot shows the 'Request Groups' dialog box. At the top, there are four input fields: 'Group' (set to 'All Reports'), 'Application' (set to 'Payables'), 'Code' (empty), and 'Description' (set to 'All Payables SRS reports (including update programs)'). Below these fields is a section titled 'Requests' with a table. The table has three columns: 'Type', 'Name', and 'Application'. The 'Type' column contains values like 'Application', 'Program', 'Program', etc. The 'Name' column lists various reports and programs such as 'Payables', 'Workflow Background Process', 'ADS Financials', etc. The 'Application' column shows the application they belong to, such as 'Payables', 'Application Object Library', 'General Ledger', etc. A scroll bar is visible on the right side of the table.

Type	Name	Application
Application	Payables	Payables
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	Mass Additions Create Report	Assets
Program	Publish RX Reports	Assets
Program	Invoice Register	Payables
Program	Invoice on Hold Report	Payables
Program	Cash Requirement Report	Payables
Program	Payment Register	Payables
Program	Missing Document Numbers Report	Payables

Request Group Form

3. Add an entry for Concurrent Program in the Request Group and save the work.

The screenshot shows the 'Request Groups' dialog box with the same initial settings as the previous screenshot: 'Group' set to 'All Reports', 'Application' set to 'Payables', 'Code' empty, and 'Description' set to 'All Payables SRS reports (including update programs)'. The 'Requests' table now includes a new row where the 'Type' is 'Program', the 'Name' is 'XXTEST: AP Invoices Interface Program', and the 'Application' is 'Custom Development'. This new row is highlighted with a yellow background.

Type	Name	Application
Application	Payables	Payables
Program	Workflow Background Process	Application Object Library
Program	XXTEST: AP Invoices Interface Program	Custom Development
Program	ADS Financials	General Ledger
Program	Mass Additions Create Report	Assets
Program	Publish RX Reports	Assets
Program	Invoice Register	Payables
Program	Invoice on Hold Report	Payables
Program	Cash Requirement Report	Payables
Program	Payment Register	Payables

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and further it can be used to install in another Application Server.

Before downloading the script of Concurrent Program, connect your local system to Applications Server

Run the Following Command to download the script

```
FNDLOAD <Apps User/Apps Pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<Concurrent_Program_Short_Name>
```

For Example:

- To Download the “[XX<EMPID>: AP Invoices Interface Loader Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct XXAPINVOICELOAD.ldt
PROGRAM CONCURRENT_PROGRAM_NAME =
XXAPINVOICELOAD
```

- To Download the “[XX<EMPID>: AP Invoices Interface Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct XXAPINVOICEMAIN.ldt
PROGRAM CONCURRENT_PROGRAM_NAME =
XXAPINVOICEMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



1. Save the attached zip file in local system
2. Transfer the zip to the Oracle Applications Server
3. Create a Staging Directory [E.g. XXAPINTOIT] in temporary area on Oracle Application Server.
4. `mkdir -p XXAPINTOIT`
5. Change the Permissions on the temp directory.
6. `chmod 755 XXAPINTOIT`
7. FTP the **XXAPINTOIT.tar.gz** file in BINARY mode to directory **XXAPINTOIT**
8. Change directory to **XXAPINTOIT** as given below
9. `cd XXAPINTOIT`
10. Uncompress **XXAPINTOIT.tar.gz** as given below
11. `gunzip XXAPINTOIT.tar.gz`
12. Untar the file **XXAPINTOIT.tar** using
13. `tar -xvf XXAPINTOIT.tar.gz`
14. Grant the execute permission on the install script using
15. `chmod 755 XX_AP_INVOICE.install`
16. Run `XX_AP_INVOICE.install`
17. `sh XX_AP_INVOICE.install`
18. Enter the apps schema user name password (apps/ <apps_pwd>), when asked for.

Extract File Layout

Please find the attached extract file for AP Invoices.



Note: Please remove the header row before putting the same file on Oracle Application Server for loading purpose.

Below is the example of extract file for AP Invoices

	A	B	C	D	E	F	G	H	I	J	K	L
1	Invoice_type	Invoice_num	Curr_co	Vendor_num	Vendor_site	Payment	Line_number	Description	Header_a	Line_amnt	Source	Distribution_set
2	STANDARD	12345	USD	2007	DANVILLE-CTR	Net 30	1	TEST	600	200	Import	PUB
3	STANDARD	12345	USD	2007	DANVILLE-CTR	Net 31	2	TEST	600	200	Import	PUB
4	STANDARD	12345	USD	2007	DANVILLE-CTR	Net 32	3	TEST	600	200	Import	PUB
5	STANDARD	12346	USD	20061	INTERNAL-OPS	Net 33	1	TEST	1200	300	Import	PUB
6	STANDARD	12346	USD	20061	INTERNAL-OPS	Net 34	2	TEST	1200	300	Import	PUB
7	STANDARD	12346	USD	20061	INTERNAL-OPS	Net 35	3	TEST	1200	300	Import	PUB
8	STANDARD	12346	USD	20061	INTERNAL-OPS	Net 36	4	TEST	1200	300	Import	PUB
9	STANDARD	12347	USD	20144	VO TRI	Net 37	1	TEST	1500	500	Import	PUB
10	STANDARD	12347	USD	20144	VO TRI	Net 38	3	TEST	1500	500	Import	PUB
11	STANDARD	12347	USD	20144	VO TRI	Net 39	4	TEST	1500	500	Import	PUB
12												
13												
14												
15												
16												
17												
18												
19												
20												
21												
22												
23												
24												
25												
26												
27												
28												
29												
30												
31												

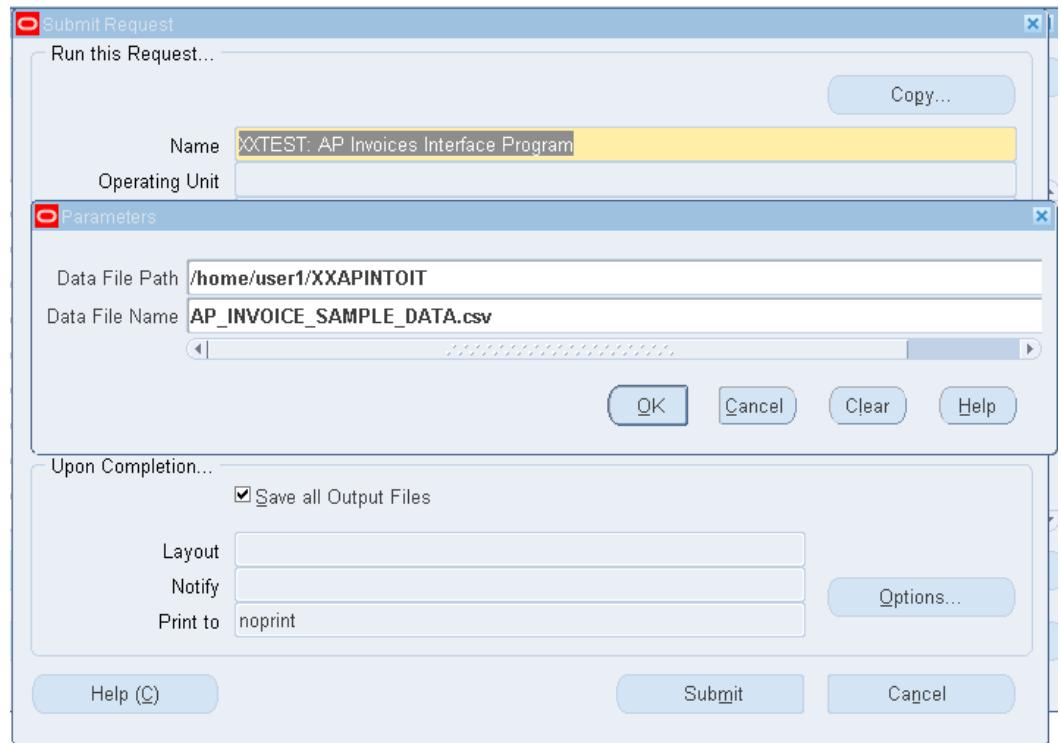
Extract File Mapping to Staging Table Columns

Data File Columns	Datatype	Size	Staging Table's Columns
Invoice_type	Varchar2	50	Invoice_type
Invoice_num	Varchar2	20	Invoice_num
Curr_code	Varchar2	20	Curr_code
Vendor_number	Varchar2	30	Vendor_number
Vendor_site	Varchar2	30	Vendor_site
Payment_term	Varchar2	10	Payment_term
Line_number	Varchar2	10	Line_number
Description	Varchar2	250	Description
Header_amount	Number		Header_amount
Line_amount	Number		Line_amount
Source	Varchar2	100	Source
Distribution_set_name	Varchar2	100	Distribution_set_name

Calling Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “**XX<EMPID>: AP Invoices Interface Program**”. To run the concurrent program navigate to [Payables Responsibility](#).

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Calling Standard Import Concurrent Program

When “[XX<EMPID>: AP Invoices Interface Program](#)” completed successfully, run the standard import program to populate the data from AP Transaction Interface Table (RA_INTERFACE_LINES_ALL) to Payables base table.

The standard Import “[Payables Open Interface Import](#)” program will fetch the data from interface table and populate it into the Payables base tables.

Call the “[Payables Open Interface Import](#)” Program with the following Parameters

- a. Transaction Source: Test
- b. Default Date: Sysdate
- c. Base Due Date on Trx Date: No

Query interface records

When “[Payables Open Interface Import](#)” standard import program completes successfully navigates to **Payables Responsibility**, **Select Invoices -> Entry -> Invoices** and queries the records of interface table.

Operating Unit	Customer Taxpayer ID	Type	PO Number	Trading Pa	Supplier Num	Supplier Site	Invoice Date	Invoice Num	Invoice I
Vision Operations		Standard		Office Supp	1008	OFFICESUP	05-MAR-199	19879-781	USD
Vision Operations		Standard		Star Gate L	1004	STAR GATE	04-JAN-199	1715	USD
Vision Operations		Standard		GE Plastic	1000	GE PLASTIC	04-JAN-199	6181-1	USD
Vision Operations		Standard		Advanced I	1013	SANTA CLA	04-JAN-199	1617	USD

AP Transaction Form

Error Handling

Following error can occur while processing the AP Transaction data file?

1. During loading data into staging table.

- When error occurred while loading the data file using SQL*Loader concurrent program it generates two files they are
 - **SQL*Loader Bad File:** The bad file contains records that weren't loaded into the staging table. These records could have been rejected by SQL*Loader due to an invalid format. Also they could have been rejected by the Oracle database if they violate an integrity constraint or had an invalid data type for the staging table.
 - **SQL*Loader Log File:** Detailed information about the load is stored in the log file. Any errors found during parsing of the control file are stored in the log file. The log file also identifies the number of records successfully loaded. The log file must be available during the entire run of the SQL*Loader. When loading data with SQL*Loader, nothing should be assumed without reviewing the log files

2. Validating the records in staging table

- While validating the data in staging table it can complete with error and the concurrent program will complete with warning, in this case look into the output report of concurrent program and correct the data file and re-load it again.

3. While calling the Standard AutoInvoice Import Program

- After populating the records in interface table it again validated by the standard import program and in few cases it completes with error or warning, in this case see the output report of Standard Import Program and correct the data in data file and re-load it, than re-run the custom concurrent program.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Migration of AR Transactions using Standard API

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Data Dependencies.....	3
Business Rules	4
Approach.....	4
Module List.....	5
Registering Concurrent Program	9
Downloading Concurrent Program Script	18
Deployment.....	19
Extract File Layout	20
Extract File Mapping to Staging Table Columns	21
Calling Custom Concurrent Program.....	22
Query interface records.....	23
Error Handling	24
Open and Closed Issues for this Deliverable	25
Open Issues	25
Closed Issues.....	25

Introduction

AR Transactions are the invoices, which has to be sent to customer for the services provided. It can be of Standard Invoice, Credit Memo, and Debit Memo etc.

Use the Transaction window to enter the invoices, debit memos, credit memos and commitments. You can also query and update your transactions in this window and review your transactions and chargeback's in the Transactions Summary window.

To enter or query an AR Transaction from front end navigate to Receivables Responsibility, Select **Transactions -> Transactions**.

This document demonstrates the use of Standard API for AR transactions to migrate AR Invoices from source system to Oracle Applications.

Purpose

This document describes the:

- This case study is intended to demonstrate the creation of Standard Invoice, Credit Memo and Debit Memo.
- Detailed data mapping from the source system(s) to the Oracle Applications E-Business Suite (EBS) Receivables (AR) Transactions records
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during implementation of oracle receivables.

Scope and Application

The following boundaries are specific to the Interface of AR Transactions using [AR_INVOICE_API_PUB.create_single_invoice](#) standard API.

- All Invoices, Credit & Debit Memos would be migrated to Oracle Applications

- The ‘[XX<EMPID>: AR Transactions Interface API Program](#)’ uploads AR Transactions data from data file into staging table “XXAR_INVOICE_API_STG,” and will perform validation and transfer valid records into Oracle Receivables interface tables. There is no need to run standard import program ‘AutoInvoice Import’ to perform the migration AR Transactions data from staging table to Receivables base tables because the API itself will migrate the data from staging table to Receivables base tables.
 1. The AR Transactions data will be provided through data files. This data file will be loaded into staging table(s) on the Oracle applications database using loader program.
 2. The loaded data will be validated from staging tables and then passed to the API as parameters.
 3. As the API itself will populate the data from staging table to Receivables base tables so there is no need of running Auto Invoice Import program.

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.

Source system

The implementation team studies the source system and understands the data required to be migrated to Oracle Applications. This data is then extracted and cleansed from the source system (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team. The data mapping section describes the data format of source system to Oracle Applications.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Receivables Manager
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.

- Users have access to Oracle Applications database (**apps schema**)
 - All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.
 - The extract-file layout must be in the format as specified in the extract file layout section
-

Pre-Requisites

Prerequisite set-ups are required for the interface. These setups include, but are not limited to:

1. Receivables System Options
2. Payment Terms
3. Transaction Types
4. Invoice Sources
5. Invoicing and Accounting Rules
6. Standard Memo Lines
7. Customer Profile Classes
8. System Profile Options for Oracle Receivables
9. Tax Codes and Rates
10. Batch Sources (Setup a batch source with name Test)

Data Dependencies

AR Invoices

In order to migrate AR Transactions the following entities must have been migrated successfully:

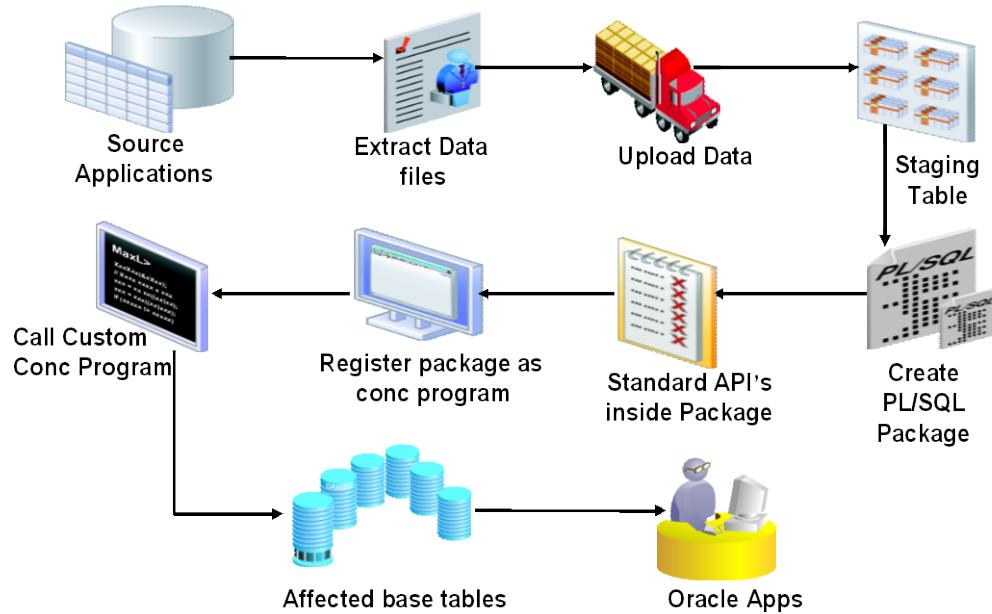
- Customers
- Items

Business Rules

Following business rules are applicable for the AR Transactions conversion process.

- N/A

Approach



The Approach for migrating AR Transactions data from source system to Receivables open interface table

1. Extract the data file from source system
2. Load the data file into staging table
3. Using PL/SQL write a package
4. Register the PL/SQL package as concurrent program in Oracle Applications
5. Call the custom concurrent program for validation and using standard API populating the records available in staging table to Receivables base tables.
6. Query the records, which got created for interface table and see the result in Oracle Applications front end.

Module List

Concurrent Programs

'[XX-<EMPID>: AR Transactions Interface API Program](#)' includes the following concurrent programs:

['XX<EMPID>: AR Transactions Interface Program'](#)

The '[XX<EMPID>: AR Transactions Interface API Program](#)' concurrent program is based on the stored procedure **XXAPI_AR_INVOICE_PKG.main** that is registered as a PL/SQL executable.

- The above program calls loader program ([XX<EMPID>: AR Transactions Interface API Loader Program](#)) to load data into staging tables, performs validations on the staging table data calling the standard API to populate the staging table records in Receivables base tables.

['XX<EMPID>: AR Transactions Interface API Loader Program'](#)

The '[XX<EMPID>: AR Transactions Interface API Loader Program](#)' concurrent program is based on the SQL* Loader control file, that is registered as a Loader executable.

- The above program loads data from the path specified as parameter value in loader concurrent program to the staging table.

NOTE:

1. Change the <EMPID> with your Employee ID,
2. Follow the steps for registering concurrent program from Registering Concurrent Program section.

Stored Procedures

[XXAPI_AR_INVOICE_PKG](#) package consist the following stored procedures:

[XXAPI_AR_INVOICE_PKG.main](#)

This is the main procedure that calls the load, validate, import, print_error and record_summary procedures.

[XXAPI_AR_INVOICE_PKG.validate](#)

This procedure performs the necessary validations on the staging tables.

[XXAPI_AR_INVOICE_PKG.import](#)

This procedure will call the standard API “AR_INVOICE_API_PUB.create_single_invoice”, which will create the records in Receivables base table for AR Transactions

[XXAPI_AR_INVOICE_PKG.print_error](#)

This procedure reports the errored records in the log file.

[XXAPI_AR_INVOICE_PKG.record_summary](#)

This procedure will display the status of staging table records

NOTE: Please refer the attached documents for Package Specifications and Package body. Kindly open the attached script in notepad.



1. Package Specification:



2. Package Body:

Staging Table

XXAR_INVOICE_API_STG table created in database to store records from source system data file.

NOTE: Please refer the attached script staging table creation. Kindly open the attached script in notepad.



Table Creation Script:

Grant Script

If table and package are created in custom schema then grant the table and package to APPS schema

NOTE: Please refer the attached script for table and package granting. Kindly open the attached script in notepad.



XXAPI_AR_INVOICE
_PKG.grt

1. Package Grant Script:



XXAPI_AR_INVOICE
_TAB.grt

2. Table Grant Script:

Synonym Script

If table and package are created in custom schema then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



XXAPI_AR_INVOICE
_PKG.syn

1. Package synonym script:



XXAPI_AR_INVOICE
_TAB.syn

2. Table synonym script:

SQL* Loader Script

SQL*Loader is controlled by its own data definition language, which is kept in the control file. The control file describes the data to be loaded, the destination tables of the data and describes the interdependency between the data and the columns within the tables. Which in-turn used to register as a concurrent program in Oracle Applications to load the data file into staging table.

NOTE: Please refer the attached SQL* Loader script for loading data file into staging table. Kindly open the attached script in notepad.



XXAPI_AR_INVOICE
_LOAD.ctl

1. SQL* Loader Script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script
3. Staging Table Script
4. SQL* Loader Script

NOTE: Please refer the attached Install script for deployment of AR Transactions object on Oracle Applications Server. Kindly open the attached script in notepad.



1. Install Script:

Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

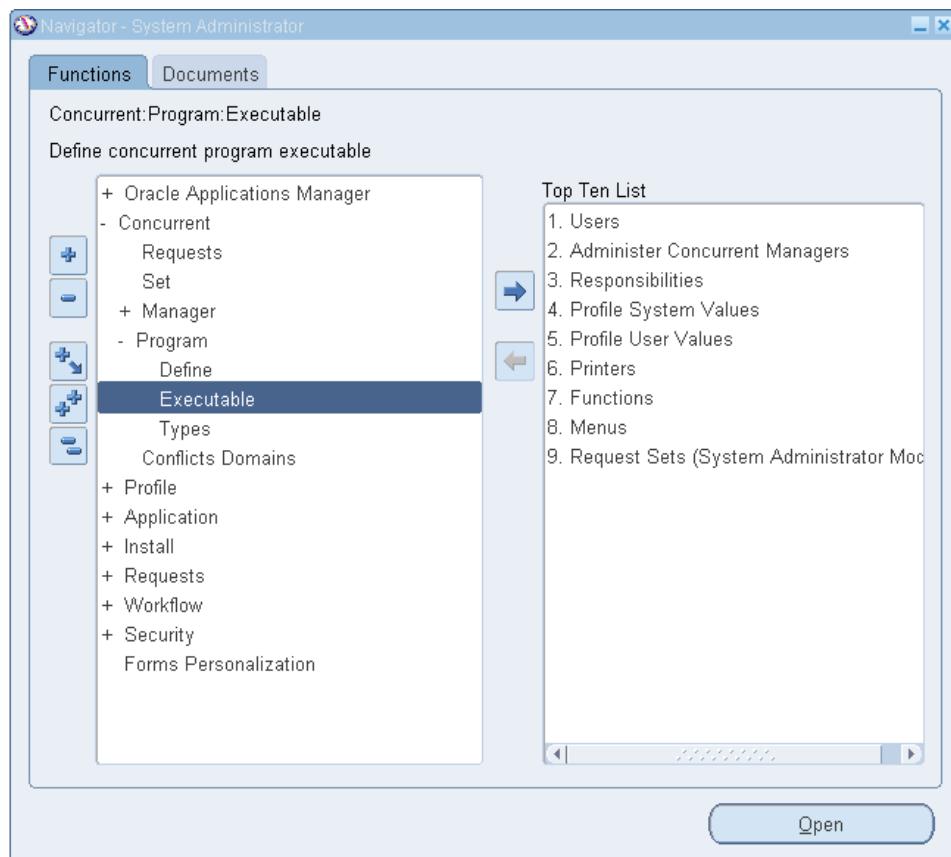
- A.) Package Name
- B.) Table Name
- C.) Synonym Name
- D.) Concurrent Program Executable.
- E.) Concurrent Program Name.

Registering Concurrent Program

Registering Interface Loader Concurrent Program

To register a concurrent program for '[XX<EMPID>: AR Transactions Interface API Loader Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

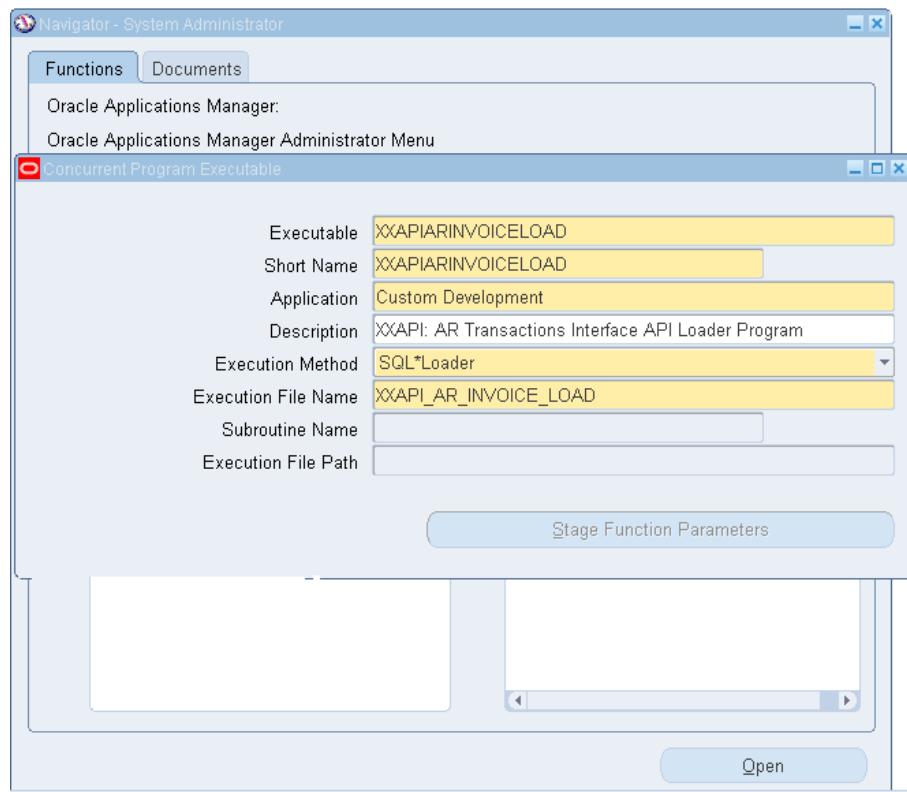
1. Select Concurrent -> Program -> Executable



2. Enter The Following in the Concurrent Program Executable Form

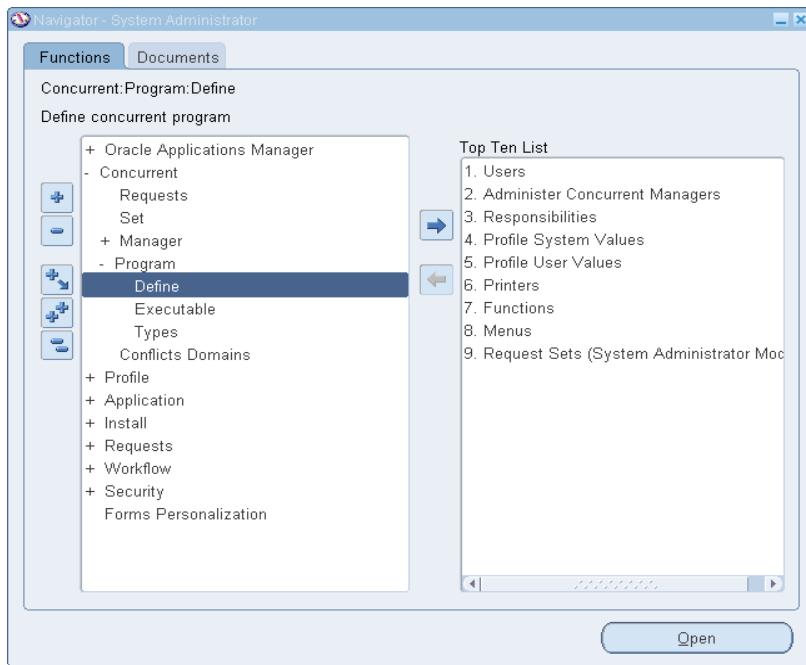
- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for AR Transaction so the Application Name should be [Custom Applications](#).
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be SQL*Loader, because we are registering a concurrent program for loading the data to staging table.

- f. **Execution File Name:** Here we would be passing the file name of SQL*Loader control file. As discussed in Module List the control file name would be **XXAPI_AR_INVOICE_LOAD**



Executable Registering Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name: Enter “XX<EMPID>: AR Transactions Interface API Loader Program”
- Short Name: Enter a unique concurrent program short name, for example: **XXAPIARINVOICELOAD**
- Application: Select the value from LOV, as we would be executing this concurrent program from Receivables Applications so select Receivables.
- Description: Enter the description of this concurrent program.
- Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXAPIARINVOICELOAD** from LOV.
- Executable Method: Executable method will be populated automatically.
- Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A). Data File

The screenshot shows the 'Concurrent Programs' dialog box. The 'Program' field is set to 'XXAPI: AR Transactions Interface API Loader Program' and has the 'Enabled' checkbox checked. The 'Short Name' field is set to 'XXAPIARINVOICELOAD'. The 'Application' field is set to 'Custom Development'. The 'Executable' section shows 'Name' as 'XXAPIARINVOICELOAD' and 'Method' as 'SQL*Loader'. The 'Request' section includes fields for 'Type', 'Incrementor', and 'MLS Function', all of which are currently empty. Under 'Request' settings, there are checkboxes for 'Use in SRS' (checked), 'Run Alone' (unchecked), 'Enable Trace' (unchecked), 'Allow Disabled Values' (unchecked), 'Restart on System Failure' (checked), and 'NLS Compliant' (checked). The 'Output' section specifies 'Format' as 'Text', with 'Save' and 'Print' checkboxes checked. It also includes fields for 'Columns', 'Rows', 'Style' (with 'Style Required' unchecked), and 'Printer'. At the bottom of the dialog are buttons for 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters'.

Concurrent Program Form

Concurrent Program Parameters

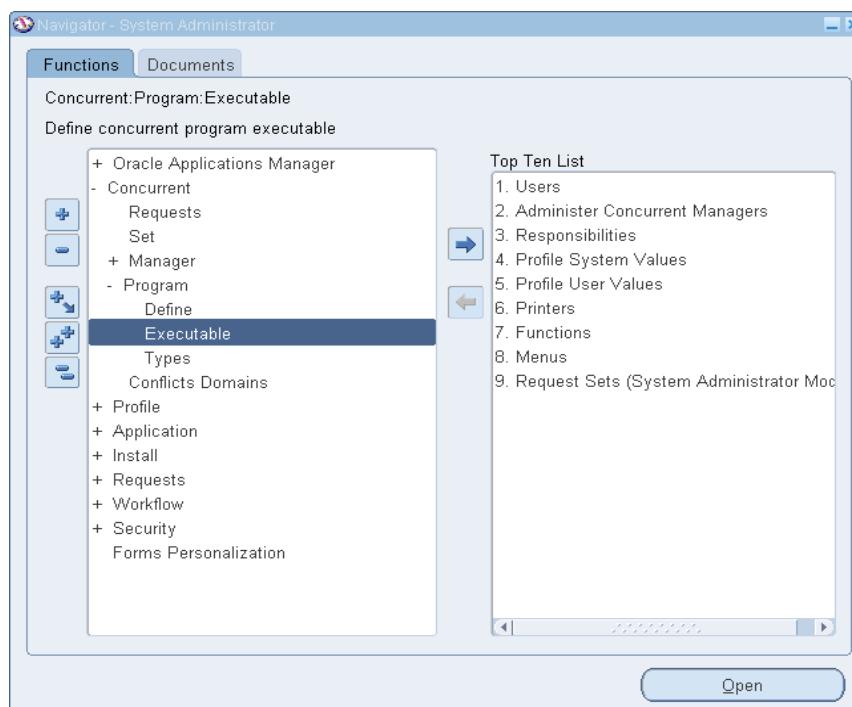
Program	XXAPI: AR Transactions Interface API Loader Program		
Application	Custom Development		
Conflicts Domain			
Security Group			
Seq	Parameter	Description	Enabled
10	Data_file	Data file name with path	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
Validation			
Value Set		100 Characters	Description
Default Type			Default Value
<input type="checkbox"/> Required		<input type="checkbox"/> Enable Security	Range
<input checked="" type="checkbox"/> Display			
Display Size		50	Description Size
Concatenated Description Size		25	Prompt
Token			

Concurrent Program Parameters Form

Registering Interface Concurrent Program

To register a concurrent program for '[XX<EMPID>: AR Transactions Interface API Program](#)' in Oracle Applications Navigate to System Administrator Responsibility

1. Select Concurrent -> Program -> Executable



2 Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for AR Transaction so the Application Name should be **Custom Applications** and add this custom concurrent program in Receivables Request Group.
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be PL/SQL Stored Procedure.
- Execution File Name:** Here we would be passing the package name with one public procedure. As discussed in Module List the package name would be **XXAPI_AR_INVOICE_PKG** and the public procedure defined in this package is **main**.

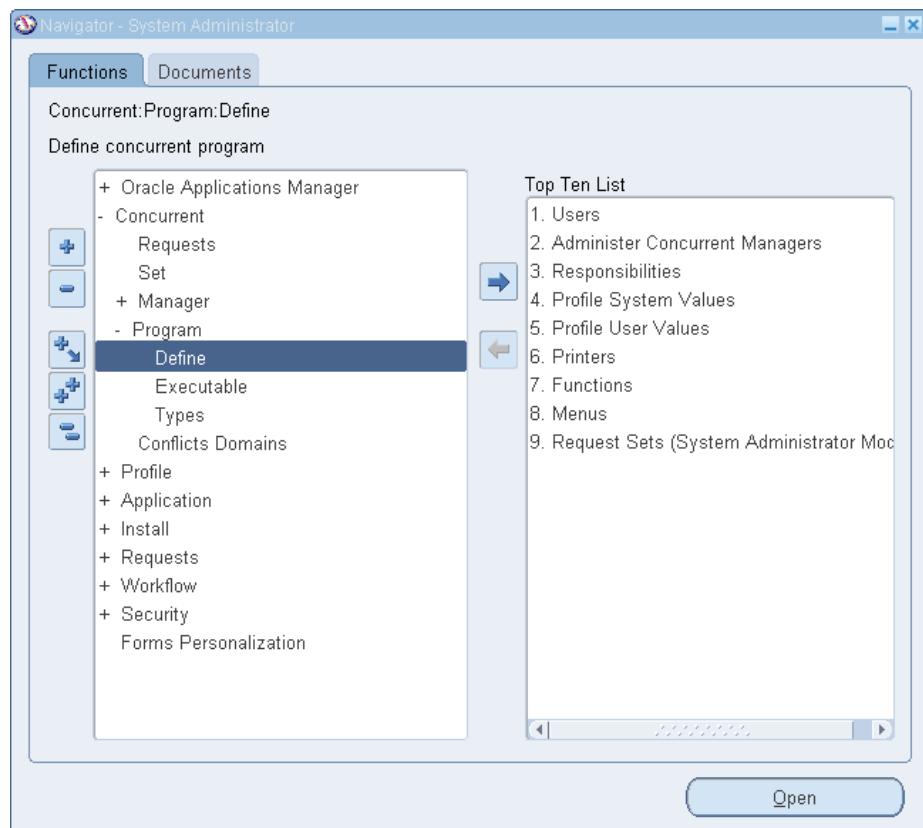
The screenshot shows the Oracle Applications Manager interface for defining a concurrent program executable. The window title is "Navigator - System Administrator". Inside, there's a sub-menu bar with "Functions" and "Documents". The main area is titled "Concurrent: Program: Executable" with the sub-instruction "Define concurrent program executable". Below this, there's a toolbar with "+ Oracle Applications Manager" and "Top Ten List". The main content area is titled "Concurrent Program Executable" and contains the following fields:

Executable	XXAPIARINVOICEMAIN
Short Name	XXAPIARINVOICEMAIN
Application	Custom Development
Description	XXAPI: AR Transactions Interface API Program
Execution Method	PL/SQL Stored Procedure
Execution File Name	XXAPI_AR_INVOICE_PKG.main
Subroutine Name	
Execution File Path	

At the bottom right of the form is a "Stage Function Parameters" button. Below the form is a "Forms Personalization" section with a scroll bar and an "Open" button at the bottom right.

Concurrent Program Executable Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name: Enter “XX<EMPID>: AR Transactions Interface API Program”
- Short Name: Enter a unique concurrent program short name, for example: **XXAPIARINVOICEMAIN**
- Application: Select the value from LOV, as we would be executing this concurrent program from Receivables Applications so select Receivables.
- Description: Enter the description of this concurrent program.
- Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXAPIARINVOICEMAIN** from LOV.
- Executable Method: Executable method will be populated automatically.
- Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A) Data File Path B) Data File Name

Program: XXAPI: AR Transactions Interface API Program

Short Name: XXAPIARINVOICEMAIN

Application: Custom Development

Description:

Executable:

- Name:** XXAPIARINVOICEMAIN
- Method:** PL/SQL Stored Procedure
- Options:**
- Priority:**

Request:

- Type:**
- Incrementor:**
- MLS Function:**
- Checkboxes:**
 - Use in SRS
 - Run Alone
 - Enable Trace
 - Allow Disabled Values
 - Restart on System Failure
 - NLS Compliant

Output:

- Format:** Text
- Checkmarks:**
 - Save (C)
 - Print
- Columns:**
- Rows:**
- Style:**
- Checkboxes:**
 - Style Required
- Printer:**

Buttons: Copy to..., Session Control, Incompatibilities, Parameters

Concurrent Program Form

Program: XXAPI: AR Transactions Interface API Program

Application: Custom Development

Conflicts Domain:

Security Group:

Seq	Parameter	Description	Enabled
10	Data_File_Path	Data file path	<input checked="" type="checkbox"/>
20	Data_File_Name	Data file name	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>

Validation:

- Value Set:** 100 Characters
- Default Type:**
- Checkboxes:**
 - Required
 - Enable Security
- Description:** 100 Characters
- Default Value:**
- Range:**

Display:

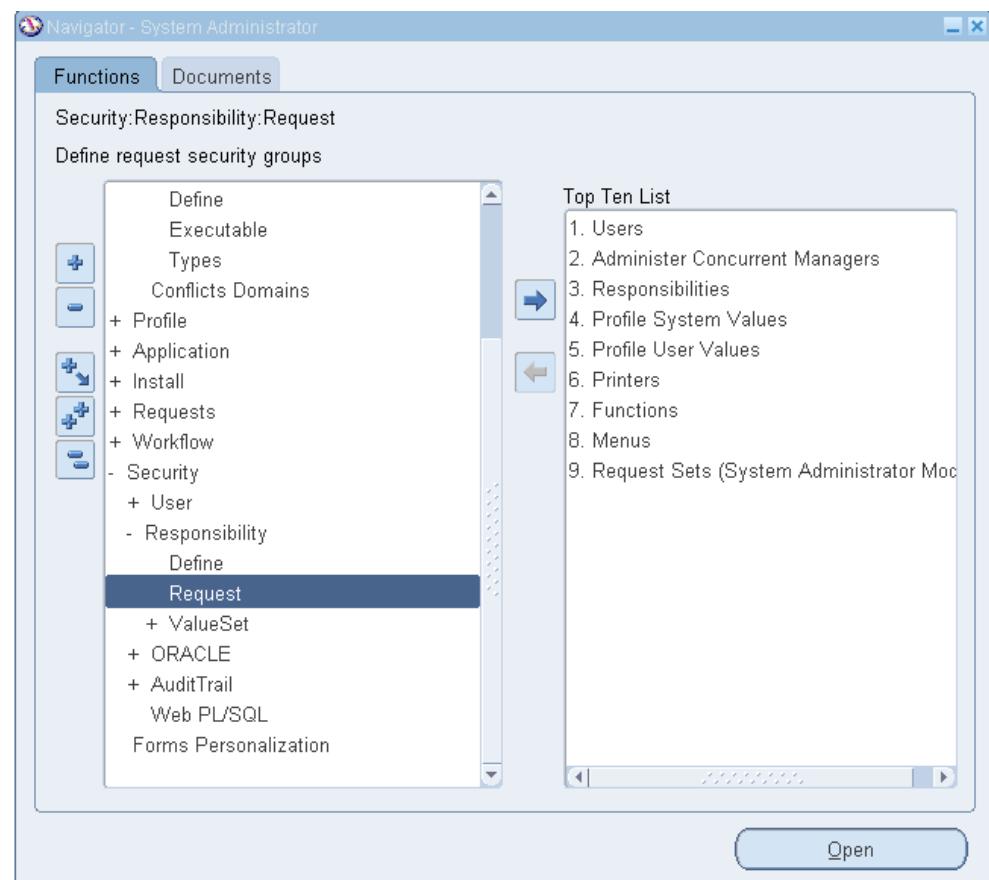
- Display Size:** 50
- Concatenated Description Size:** 25
- Description Size:** 50
- Prompt:** Data File Name
- Token:**

Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: AR Transactions Interface API Program’ in Request Group, Navigate to System Administrator Responsibility

1. Select Security -> Responsibility-> Request



2. Query the Request Group – Receivables All and Applications – Receivables

The screenshot shows the 'Request Groups' dialog box. The 'Group' field is set to 'Receivables All'. The 'Application' field is set to 'Receivables'. The 'Description' field contains 'All reports and programs'. The 'Requests' section lists various programs under the 'Receivables' application, including 'Workflow Background Process', 'ADS Financials', 'Publish RX Reports', etc. A new program entry, 'XXAPI: AR Transactions Interface API Prog', has been added to the list.

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	Publish RX Reports	Assets
Program	Collection Effectiveness Indicators	Receivables
Program	Autoinvoice Import Program	Receivables
Program	Aging - 7 Buckets - By Amount Report (obsolete)	Receivables
Program	Aging - 4 Buckets Report	Receivables
Program	Aging - 7 Buckets - By Salesperson/Agent	Receivables
Program	Aging - 7 Buckets Report (obsolete)	Receivables
Program	Billing and Receipt History	Receivables

Description: Workflow background process for deferred and timeout activities

Request Group Form

3. Add an entry for Concurrent Program in the Request Group and save the work.

The screenshot shows the 'Request Groups' dialog box with the same settings as the previous one. The 'Requests' section now includes the newly added concurrent program 'XXAPI: AR Transactions Interface API Prog' under the 'Custom Development' application.

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	XXAPI: AR Transactions Interface API Prog	Custom Development
Program	ADS Financials	General Ledger
Program	Publish RX Reports	Assets
Program	Collection Effectiveness Indicators	Receivables
Program	Autoinvoice Import Program	Receivables
Program	Aging - 7 Buckets - By Amount Report (obsolete)	Receivables
Program	Aging - 4 Buckets Report	Receivables
Program	Aging - 7 Buckets - By Salesperson/Agent	Receivables
Program	Aging - 7 Buckets Report (obsolete)	Receivables

Description:

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and further it can be used to install in another Application Server.

Before downloading the script of Concurrent Program, connect your local system to Applications Server

Run the Following Command to download the script

```
FNDLOAD <Apps User/Apps Pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<Concurrent_Program_Short_Name>
```

For Example:

- To Download the “[XX<EMPID>: AR Transactions Interface Loader Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXAPIARINVOICELOAD.ldt PROGRAM
CONCURRENT_PROGRAM_NAME = XXAPIARINVOICELOAD
```

- To Download the “[XX<EMPID>: AR Transactions Interface API Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXAPIARINVOICEMAIN.ldt PROGRAM
CONCURRENT_PROGRAM_NAME = XXAPIARINVOICEMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



1. Save the attached zip file in local system
2. Transfer the zip to the Oracle Applications Server
3. Create a Staging Directory [E.g. XXAPIARINVOICE] in temporary area on Oracle Application Server.
4. `mkdir -p XXAPIARINVOICE`
5. Change the Permissions on the temp directory.
6. `chmod 755 XXAPIARINVOICE`
7. FTP the `XXAPIARINVOICE.tar.gz` file in BINARY mode to directory `XXAPIARINVOICE`
8. Change directory to `XXAPIARINVOICE` as given below
9. `cd XXAPIARINVOICE`
10. Uncompress `XXAPIARINVOICE.tar.gz` as given below
11. `gunzip XXAPIARINVOICE.tar.gz`
12. Untar the file `XXAPIARINVOICE.tar` using
13. `tar -xvf XXAPIARINVOICE.tar.gz`
14. Grant the execute permission on the install script using
15. `chmod 755 XXAPI_AR_INVOICE.install`
16. Run `XXAPI_AR_INVOICE.install`
17. `sh XXAPI_AR_INVOICE.install`
18. Enter the apps schema user name password (apps/ <apps_pwd>), when asked for.

Extract File Layout

Please find the attached extract file for AR Transactions.



Case_Study_AR_TRX_Extract_Data.csv

Note: Please remove the header row before putting the same file on Oracle Application Server for loading purpose.

Below is the example of extract file for AR Transactions

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Inv Class	Trx Type	Trx Num	Curr	Cust Num	Loc	Pay Term	Inv Item	Line Nu	Desc	UOM	Amt	Price	Quantity
2	INVOICE	Interest Invoice	1000180277	EUR	1001	New York	30 NET	0001-0120H	1	AR Transaction Interface Test	Ea	30	10	3
3	INVOICE	Interest Invoice	1000180277	EUR	1001	New York	31 NET	0001-0120H	2	AR Transaction Interface Test	Ea	30	10	3
4	INVOICE	Interest Invoice	1000180277	EUR	1001	New York	32 NET	0001-0120H	3	AR Transaction Interface Test	Ea	30	10	3
5	Credit Memo	BR Credit Memo	3005376112	EUR	1002	Oakdale	33 NET	0001-0120H	1	AR Transaction Interface Test	Ea	80	20	4
6	Credit Memo	BR Credit Memo	3005376112	EUR	1002	Oakdale	34 NET	0001-0120H	2	AR Transaction Interface Test	Ea	80	20	4
7	Credit Memo	BR Credit Memo	3005376112	EUR	1002	Oakdale	35 NET	0001-0120H	3	AR Transaction Interface Test	Ea	80	20	4
8	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	36 NET	0001-0120H	1	AR Transaction Interface Test	Ea	150	30	5
9	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	37 NET	0001-0120H	2	AR Transaction Interface Test	Ea	150	30	5
10	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	38 NET	0001-0120H	3	AR Transaction Interface Test	Ea	150	30	5
11	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	39 NET	0001-0120H	4	AR Transaction Interface Test	Ea	150	30	5
12														
13														
14														
15														
16														
17														
18														
19														
20														
21														
22														
23														
24														
25														
26														
27														
28														
29														
30														
31														

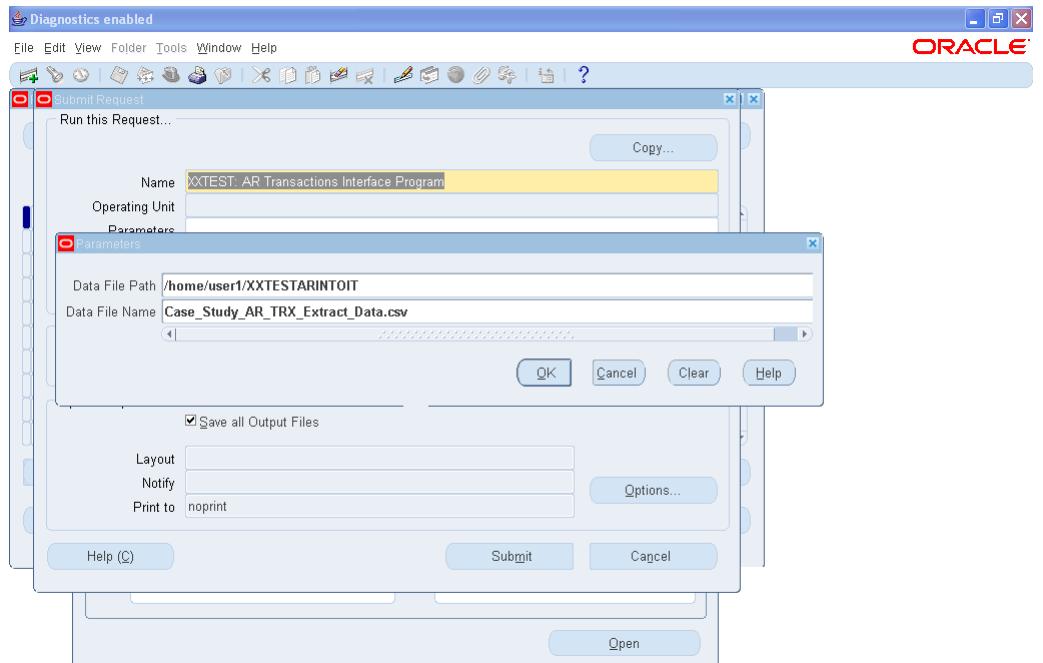
Extract File Mapping to Staging Table Columns

Data File Columns	Datatype	Size	Staging Table's Columns
Invoice_class	Varchar2	50	Invoice_class
Trx_type	Varchar2	20	Trx_type
Trx_num	Varchar2	20	Trx_num
Curr_code	Varchar2	10	Curr_code
Customer_number	Varchar2	30	Customer_number
Location	Varchar2	10	Location
Payment_term	Varchar2	10	Payment_term
Inventory_item	Varchar2	40	Inventory_item
Line_number	Number		Line_number
Description	Varchar2	250	Description
Uom	Varchar2	10	Uom
Amount	Number		Amount
Price	Number		Price
Quantity	Number		Quantity

Calling Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “[XX<EMPID>: AR Transactions Interface API Program](#)”. To run the concurrent program navigate to [Receivables Responsibility](#).

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



SRS Form

1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Query interface records

When custom concurrent program “[XX<EMPID>: AR Transactions Interface API Program](#)” completes successfully navigates to **Receivables Responsibility**, Select **Transactions -> Transactions** and queries the records of interface table.

The screenshot shows the Oracle Applications - ADS Vision interface with the AR Transaction Form open. The form is divided into several sections:

- Transaction:** Includes fields for Source (OKS_CONTRACTS), Number (125), Date (20-JUN-2001), GL Date (20-JUN-2001), Currency (USD), and Balance Due (Line: 0.00, Tax: 0.00, Freight: 0.00, Charges: 0.00, Total: 0.00).
- Ship To:** Fields for Name, Number, Location, Address, and Contact.
- Bill To:** Fields for Name, Number, Location, and Address, showing "World of Business" as the recipient.
- Sold To:** Fields for Name, Number, and Location, showing "World of Business" as the customer.
- Paying Customer:** Fields for Name, Number, and Location.
- Payment Details:** Fields for Receipt Method, Payment Method, and Instrument Number, with a "Select Instrument" button.
- Commitment:** Fields for Commitment, Payment Term (Immediate), Invoicing Rule (In Advance), and Due Date (20-JUN-2001).
- Buttons:** Main, More, Notes, Commitment, Reference Information, Details, Refresh, Line Items, Tax, Freight, Distributions, Sales Credits, and Incomplete.
- Status Bar:** Record: 1/2 <OSC>

AR Transaction Form

Error Handling

Following error can occur while processing the AR Transaction data file?

1. During loading data into staging table.

- When error occurred while loading the data file using SQL*Loader concurrent program it generates two files they are
 - **SQL*Loader Bad File:** The bad file contains records that weren't loaded into the staging table. These records could have been rejected by SQL*Loader due to an invalid format. Also they could have been rejected by the Oracle database if they violate an integrity constraint or had an invalid data type for the staging table.
 - **SQL*Loader Log File:** Detailed information about the load is stored in the log file. Any errors found during parsing of the control file are stored in the log file. The log file also identifies the number of records successfully loaded. The log file must be available during the entire run of the SQL*Loader. When loading data with SQL*Loader, nothing should be assumed without reviewing the log files

2. Validating the records in staging table

- While validating the data in staging table it can complete with error and the concurrent program will complete with warning, in this case look into the output report of concurrent program and correct the data file and re-load it again.

3. While calling the Standard API

While calling standard API it may complete with error and it returns the error message in AR_TRX_ERRORS_GT table, this table is a temporary table and store the error message for the current session only. As API error message displayed in output report of custom concurrent program verify the report and do the correction in data, after correcting the data file re-load it and run the custom current program again.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Migration of AR Transactions using Open Interface

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Data Dependencies.....	3
Business Rules	4
Approach.....	4
Module List.....	5
Registering Concurrent Program	9
Downloading Concurrent Program Script	18
Deployment.....	19
Extract File Layout	20
Extract File Mapping to Staging Table Columns	21
Calling Custom Concurrent Program.....	22
Calling Standard Import Concurrent Program.....	23
Query interface records.....	24
Error Handling	25
Open and Closed Issues for this Deliverable	26
Open Issues	26
Closed Issues.....	26

Introduction

AR Transactions are the invoices, which has to be sent to customer for the services provided. It can be of Standard Invoice, Credit Memo, and Debit Memo etc.

Use the Transaction window to enter the invoices, debit memos, credit memos and commitments. You can also query and update your transactions in this window and review your transactions and chargebacks in the Transactions Summary window.

To enter or query an AR Transaction from front end navigate to Receivables Responsibility, Select **Transactions -> Transactions**.

This document demonstrates the use of open interface table for AR transactions to migrate AR Invoices from source system to Oracle Applications.

Purpose

This document describes the:

- This case study is intended to demonstrate the creation of Standard Invoice, Credit Memo and Debit Memo.
- Detailed data mapping from the source system(s) to the Oracle Applications E-Business Suite (EBS) Receivables (AR) Transactions records
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during implementation of oracle receivables.

Scope and Application

The following boundaries are specific to the Interface of AR Transactions using RA_INTERFACE_LINES_ALL open interface table

- All Invoices, Credit & Debit Memos would be migrated to Oracle Applications

- The ‘XX<EMPID>: AR Transactions Interface Program’ uploads AR Transactions data from data file into staging table “XXAR_INVOICE_IFACE_STG,” and will perform validation and transfer valid records into Oracle Receivables interface tables. The standard import program ‘Autoinvoice Import’ is then run to perform the validation on data in the interface table and load valid data into base tables.
 1. The AR Transactions data will be provided through data files. This data file will be loaded into staging table(s) on the Oracle applications database using loader program.
 2. The loaded data will be validated from staging tables and then moved into the standard transaction interface tables.
 3. The standard AutoInvoice Import program will be called to import the transactions from Interface table to Oracle receivables base tables.

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.

Source system

The implementation team studies the source system and understands the data required to be migrated to Oracle Applications. This data is then extracted and cleansed from the source system (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team. The data mapping section describes the data format of source system to Oracle Applications.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Receivables Manager
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.
- Users have access to Oracle Applications database (**apps schema**)

- All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.
- The extract-file layout must be in the format as specified in the extract file layout section

Pre-Requisites

Prerequisite set-ups are required for the interface. These setups include, but are not limited to:

1. Receivables System Options
2. Payment Terms
3. Transaction Types
4. Invoice Sources
5. Invoicing and Accounting Rules
6. Standard Memo Lines
7. Customer Profile Classes
8. System Profile Options for Oracle Receivables
9. Tax Codes and Rates
10. Batch Sources (Setup a batch source with name Test)

Data Dependencies

AR Invoices

In order to migrate AR Transactions the following entities must have been migrated successfully:

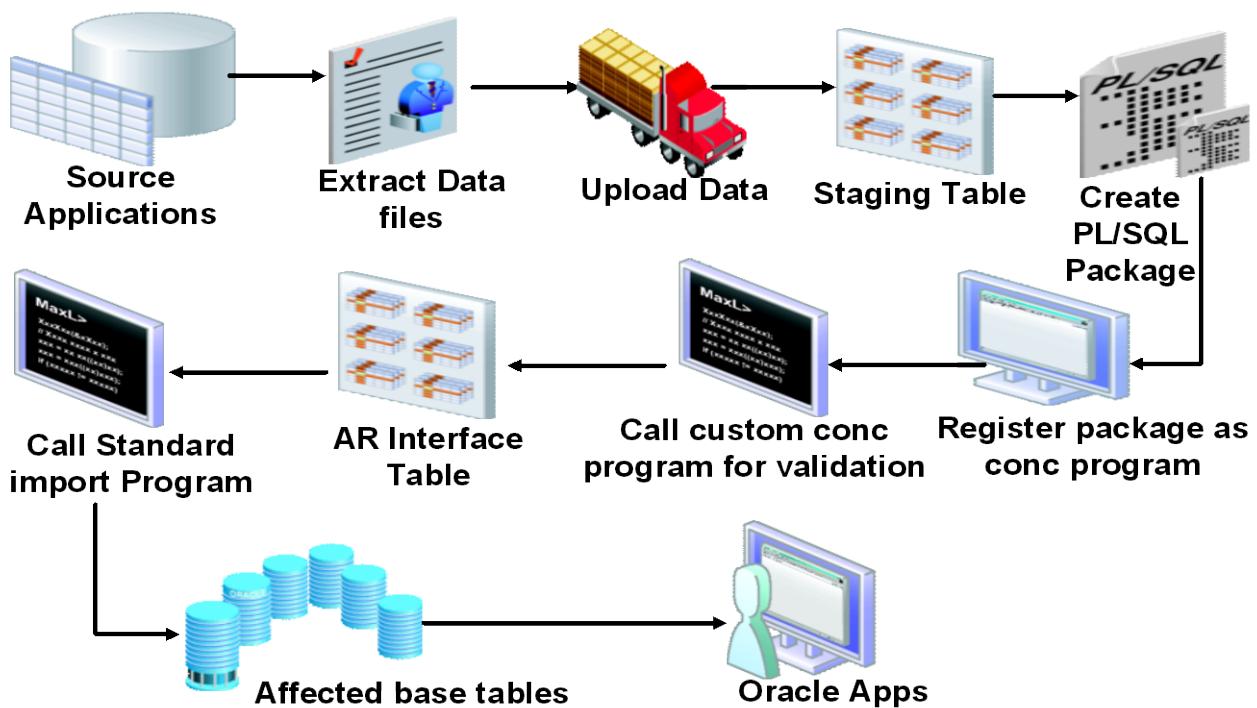
- Customers
- Items

Business Rules

Following business rules are applicable for the AR Transactions conversion process.

- N/A

Approach



The Approach for migrating AR Transactions data from source system to Receivables open interface table

1. Extract the data file from source system
2. Load the data file into staging table
3. Using PL/SQL write a package
4. Register the PL/SQL package as concurrent program in Oracle Applications
5. Call the custom concurrent program for validation and loading the records available in staging table to open interface table
6. Call the standard import program, which will create the records in RA_CUSTOMER_TRX_ALL table based on open interface records.
7. Query the records, which got created for interface table and see the result in Oracle Applications front end.

Module List

Concurrent Programs

‘XX<EMPID>: AR Transactions Interface Program’ includes the following concurrent programs:

‘XX<EMPID>: AR Transactions Interface Program’

The ‘XX<EMPID>: AR Transactions Interface Program’ concurrent program is based on the stored procedure **XXTEST_AR_INVOICE_PKG.main** that is registered as a PL/SQL executable.

- The above program calls loader program (**XX<EMPID>: AR Transactions Interface Loader Program**) to load data into staging tables, performs validations on the staging table data and uploads data into the open interface tables (RA_INTERFACE_LINES_ALL).

‘XX<EMPID>: AR Transactions Interface Loader Program’

The ‘XX<EMPID>: AR Transactions Interface Loader Program’ concurrent program is based on the SQL* Loader control file, that is registered as a Loader executable.

- The above program loads data from the path specified as parameter value in loader concurrent program to the staging table.

NOTE:

1. Change the <EMPID> with your Employee ID,
2. Follow the steps for registering concurrent program from Registering Concurrent Program section.

Stored Procedures

XXTEST_AR_INVOICE_PKG package consist the following stored procedures:

XXTEST_AR_INVOICE_PKG.main

This is the main procedure that calls the load, validate, import, print_error and record_summary procedures.

XXTEST_AR_INVOICE_PKG.validate

This procedure performs the necessary validations on the staging tables.

XXTEST_AR_INVOICE_PKG.import

This procedure loads all the valid data from the staging table to Oracle Receivables interface tables RA_INTERFACE_LINES_ALL

XXTEST_AR_INVOICE_PKG.print_error

This procedure reports the errored records in the log file.

XXTEST_AR_INVOICE_PKG.record_summary

This procedure will display the status of staging table records

NOTE: Please refer the attached documents for Package Specifications and Package body. Kindly open the attached script in notepad.



XX_AR_INVOICE_PK
G.pks

1. Package Specification:



XX_AR_INVOICE_PK
G.pkb

2. Package Body:

Staging Table

XXAR_INVOICE_IFACE_STG table created in database to store records from source system data file.

NOTE: Please refer the attached script staging table creation. Kindly open the attached script in notepad.



XX_AR_INVOICE_TA
B.tab

Table Creation Script:

Grant Script

If table and package are created in custom schema then grant the table and package to APPS schema

NOTE: Please refer the attached script for table and package granting. Kindly open the attached script in notepad.



1. Package Grant Script:



2. Table Grant Script:

Synonym Script

If table and package are created in custom schema then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



1. Package synonym script:



2. Table synonym script:

SQL* Loader Script

SQL*Loader is controlled by its own data definition language, which is kept in the control file. The control file describes the data to be loaded, the destination tables of the data and describes the interdependency between the data and the columns within the tables. Which in-turn used to register as a concurrent program in Oracle Applications to load the data file into staging table.

NOTE: Please refer the attached SQL* Loader script for loading data file into staging table. Kindly open the attached script in notepad.



1. SQL* Loader Script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script
3. Staging Table Script
4. SQL* Loader Script

NOTE: Please refer the attached Install script for deployment of AR Transactions object on Oracle Applications Server. Kindly open the attached script in notepad.

1. Install Script:



Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

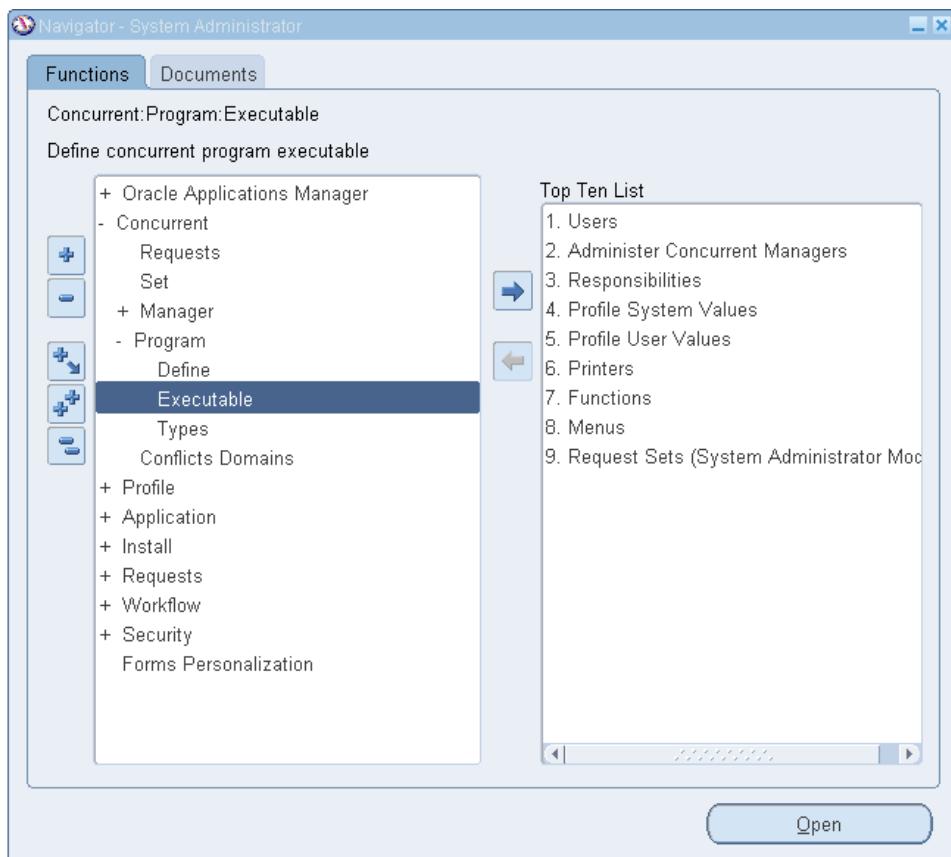
- A.) Package Name
- B.) Table Name
- C.) Synonym Name
- D.) Concurrent Program Executable.
- E.) Concurrent Program Name.

Registering Concurrent Program

Registering Interface Loader Concurrent Program

To register a concurrent program for '[XX<EMPID>: AR Transactions Interface Loader Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

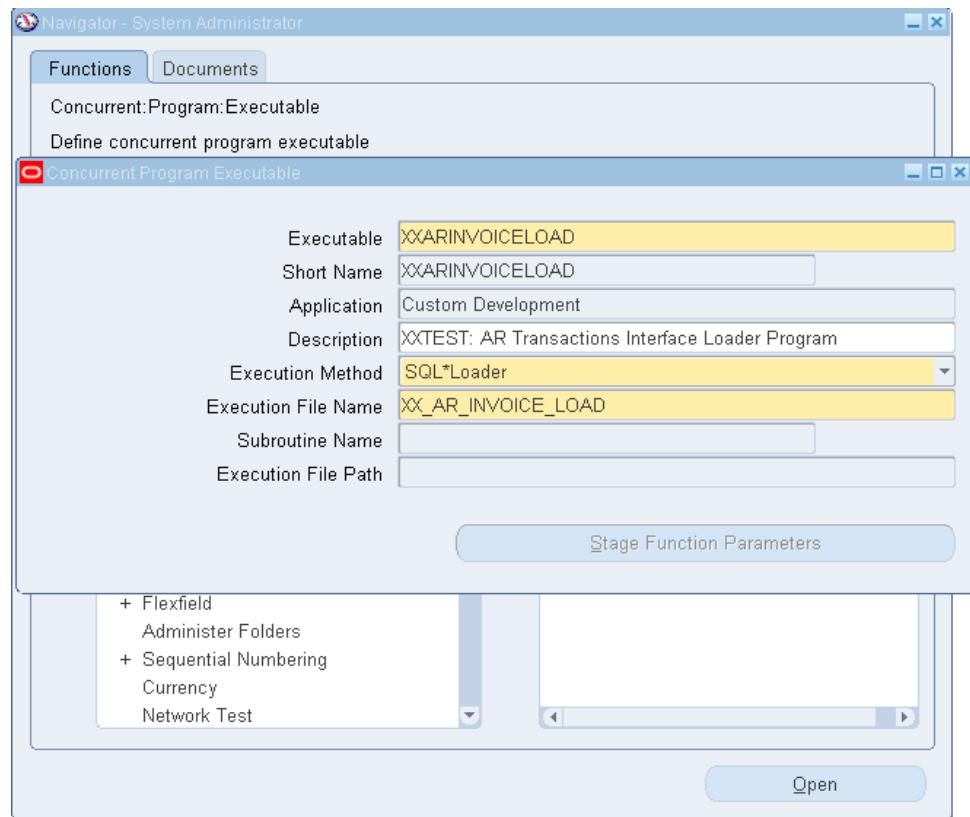
1. Select Concurrent -> Program -> Executable



2. Enter The Following in the Concurrent Program Executable Form

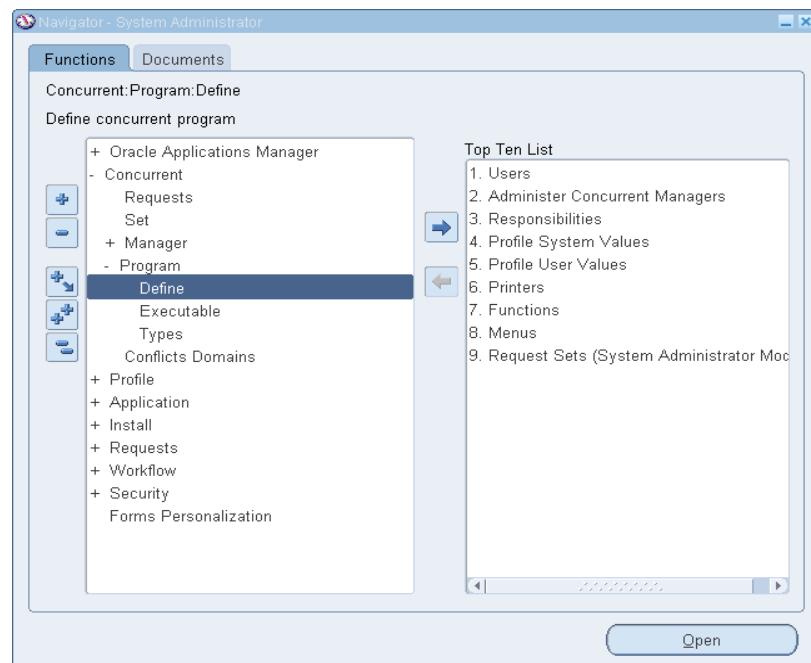
- a. **Executable Name:** Enter unique name for executable.
- b. **Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- c. **Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for AR Transaction so the Application Name should be [Custom Application](#).
- d. **Description:** Enter the description of Executable
- e. **Execution Method:** Execution method should be SQL*Loader, because we are registering a concurrent program for loading the data to staging table.

- f. **Execution File Name:** Here we would be passing the file name of SQL*Loader control file. As discussed in Module List the control file name would be **XX_AR_INVOICE_LOAD**



Executable Registering Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name: Enter “XX<EMPID>: AR Transactions Interface Loader Program”
- Short Name: Enter a unique concurrent program short name, for example: **XXARINVOICELOAD**
- Application: Select the value from LOV, as we would be executing this concurrent program from Receivables Applications so select Receivables.
- Description: Enter the description of this concurrent program.
- Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXARINVOICELOAD** from LOV.
- Executable Method: Executable method will be populated automatically.
- Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A). Data File

The screenshot shows the 'Concurrent Programs' dialog box. The 'Program' field contains 'XXTEST: AR Transactions Interface Loader Program' with the 'Enabled' checkbox checked. The 'Short Name' field is set to 'XXARINVOICELOAD'. Under the 'Executable' section, the 'Name' is 'XXARINVOICELOAD' and the 'Method' is 'SQL*Loader'. The 'Request' section includes fields for 'Type', 'Incrementor', and 'MLS Function', along with checkboxes for 'Use in SRS', 'Run Alone', 'Enable Trace', 'Allow Disabled Values', 'Restart on System Failure', and 'NLS Compliant'. The 'Output' section specifies 'Format' as 'Text', with options for 'Save (C)' and 'Print'. It also includes fields for 'Columns', 'Rows', 'Style', and 'Printer'. At the bottom are buttons for 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters'.

Concurrent Program Form

Concurrent Program Parameters

Program: XXTEST: AR Transactions Interface Loader Program
Application: Custom Development

Conflicts Domain: [] Security Group: []

Seq	Parameter	Description	Enabled
10	Data File	Data file name with file path	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>

Validation:

Value Set: 100 Characters Description: 100 Characters
Default Type: Required Default Value: []
 Enable Security Range: []

Display:

Display
Display Size: 50 Description Size: 50
Concatenated Description Size: 25 Prompt: Data File

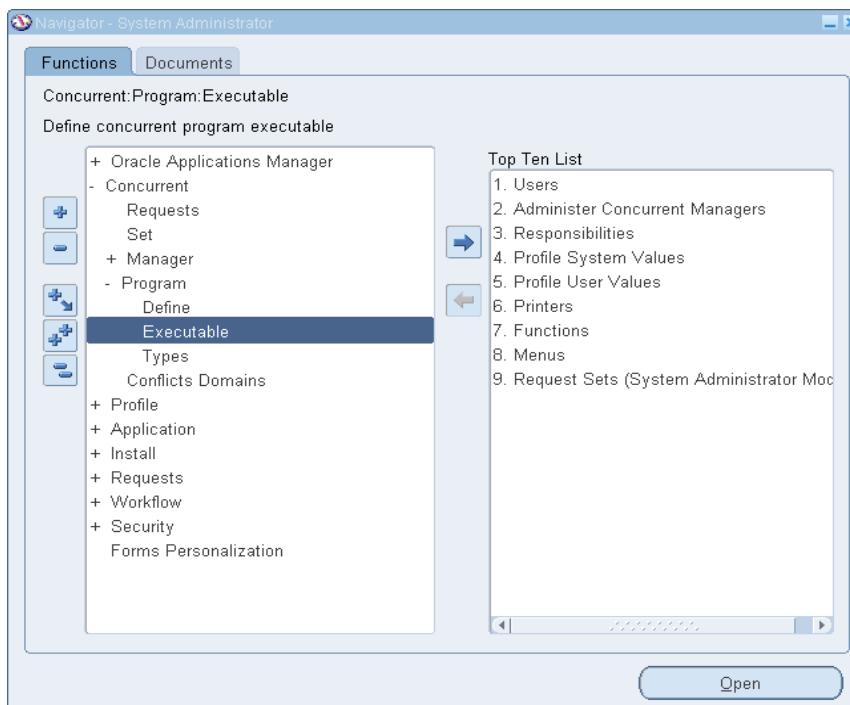
Token: []

Concurrent Program Parameters Form

Registering Interface Concurrent Program

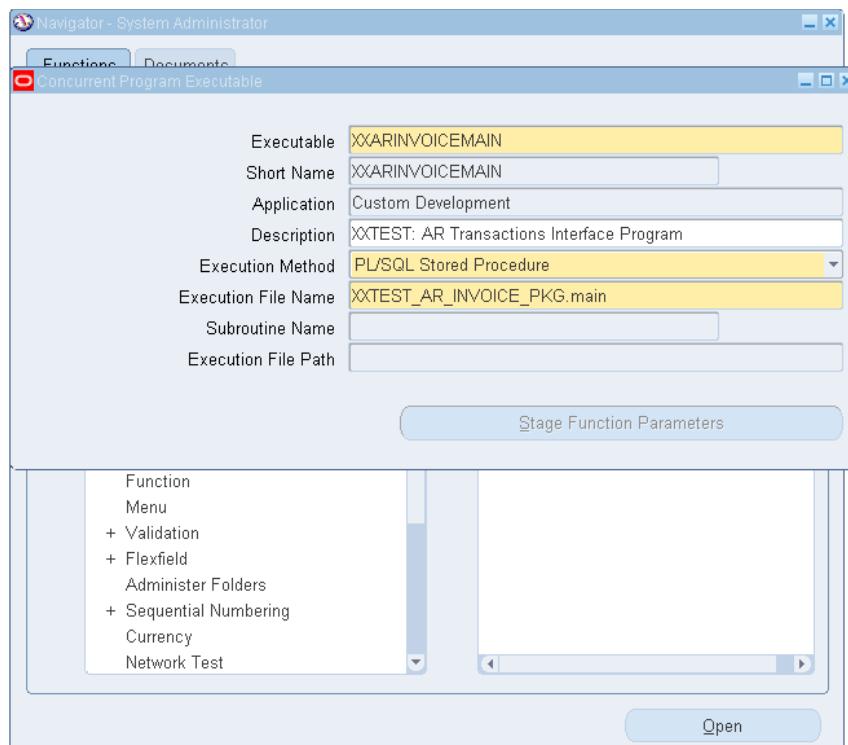
To register a concurrent program for '[XX<EMPID>: AR Transactions Interface Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



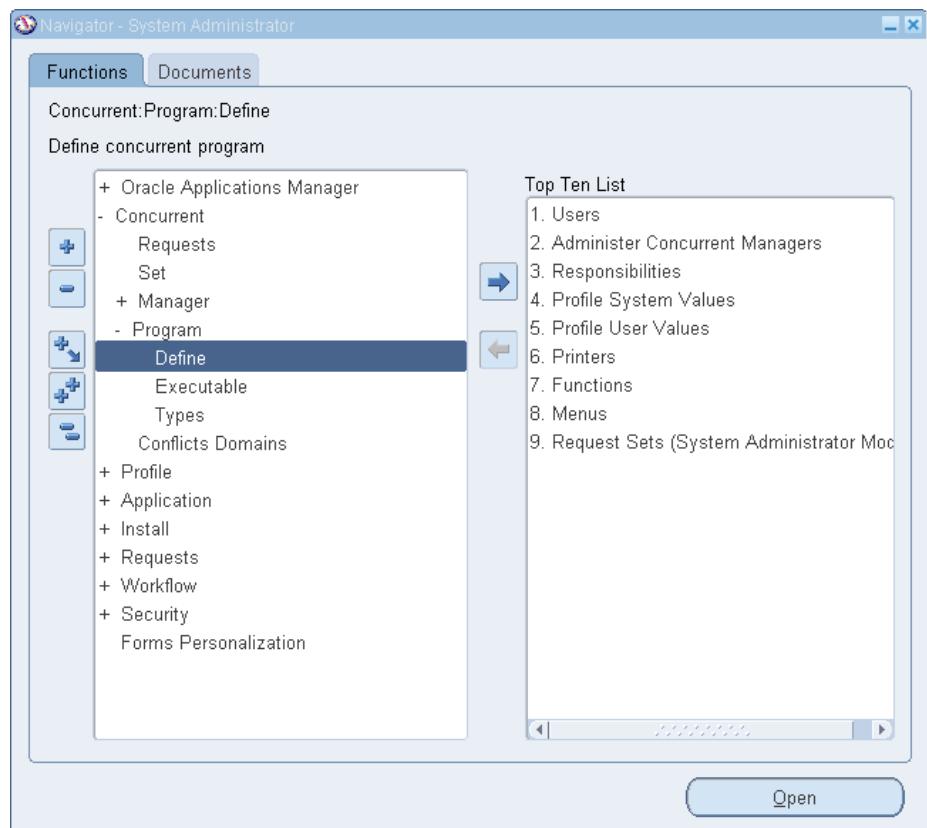
2 Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for AR Transaction so the Application Name should be **Custom Application** and add this concurrent program in Receivables Request Group.
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be PL/SQL Stored Procedure.
- Execution File Name:** Here we would be passing the package name with one public procedure. As discussed in Module List the package name would be **XXTEST_AR_INVOICE_PKG** and the public procedure defined in this package is **main**.



Concurrent Program Executable Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- a. Program Name: Enter "**XX<EMPID>: AR Transactions Interface Program**"
- b. Short Name: Enter a unique concurrent program short name, for example: **XXARINVOICEMAIN**
- c. Application: Select the value from LOV, as we would be executing this concurrent program from Receivables Applications so select Receivables.
- d. Description: Enter the description of this concurrent program.
- e. Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXARINVOICEMAIN** from LOV.
- f. Executable Method: Executable method will be populated automatically.
- g. Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A) Data File Path B) Data File Name

The screenshot shows the 'Concurrent Programs' window with the following details:

- Program:** XXTEST: AR Transactions Interface Program
- Short Name:** XXARINVOICEMAIN
- Application:** Custom Development
- Description:** (empty)
- Executable:**
 - Name:** XXARINVOICEMAIN
 - Method:** PL/SQL Stored Procedure
 - Options:** (empty)
 - Priority:** (empty)
- Request:**
 - Type:** (empty)
 - Incrementor:** (empty)
 - MLS Function:** (empty)
 - Checkboxes:**
 - Use in SRS
 - Allow Disabled Values
 - Run Alone
 - Restart on System Failure
 - Enable Trace
 - NLS Compliant
- Output:**
 - Format:** Text
 - Save (Q)
 - Print
 - Columns:** (empty)
 - Rows:** (empty)
 - Style:** (empty)
 - Style Required
 - Printer:** (empty)

Buttons at the bottom: Copy to..., Session Control, Incompatibilities, Parameters.

Concurrent Program Form

The screenshot shows the 'Concurrent Program Parameters' window with the following details:

- Program:** XXTEST: AR Transactions Interface Program
- Application:** Custom Development
- Conflicts Domain:** (empty)
- Security Group:** (empty)
- Parameter List:**

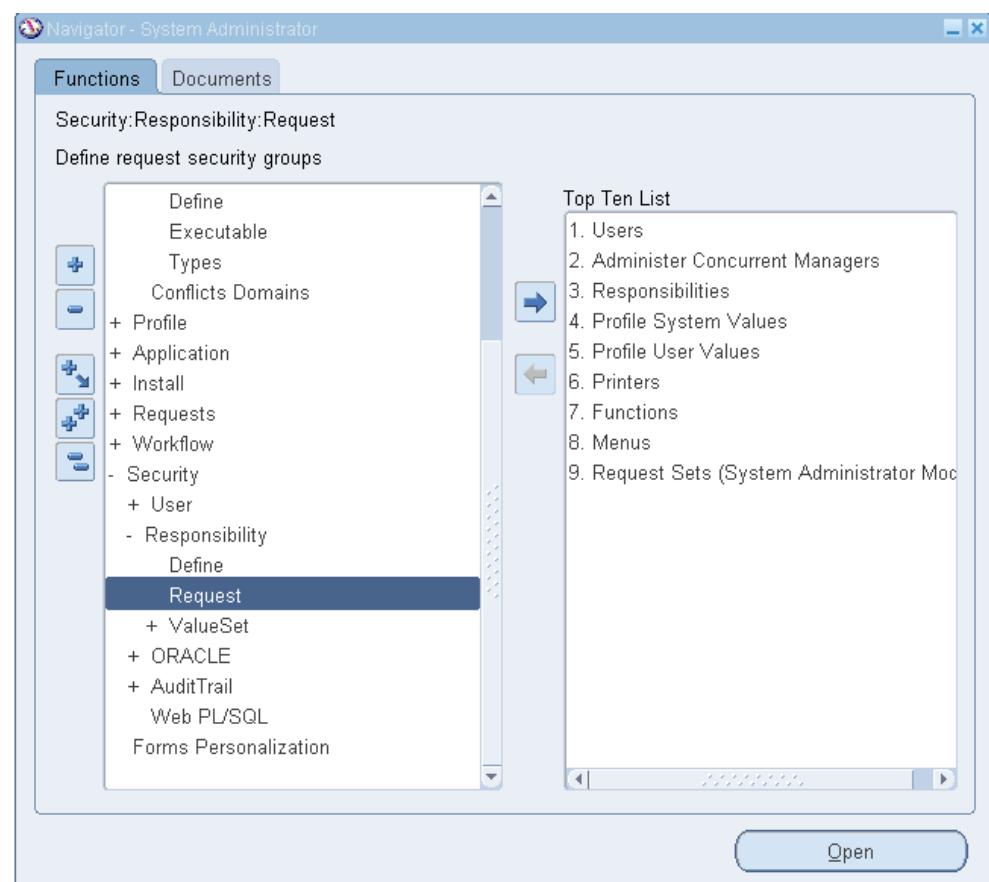
Seq	Parameter	Description	Enabled
10	Data_File_Path	Data file path	<input checked="" type="checkbox"/>
20	Data_File_Name	Data file name	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
- Validation:**
 - Value Set:** 100 Characters
 - Default Type:** (empty)
 - Required:**
 - Enable Security:**
 - Description:** 100 Characters
 - Default Value:** (empty)
 - Range:** (empty)
- Display:**
 - Display:**
 - Display Size:** 50
 - Concatenated Description Size:** 25
 - Description Size:** 50
 - Prompt:** Data File Path
 - Token:** (empty)

Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: AR Transactions Interface Program’ in Request Group, Navigate to [System Administrator Responsibility](#)

1. Select Security -> Responsibility-> Request



2. Query the Request Group – Receivables All and Applications – Receivables

The screenshot shows the 'Request Groups' dialog box with the following details:

- Group:** Receivables All
- Application:** Receivables
- Code:** (empty)
- Description:** All reports and programs
- Requests:** A table listing various programs under the 'Receivables' application.

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	Publish RX Reports	Assets
Program	Collection Effectiveness Indicators	Receivables
Program	Autoinvoice Import Program	Receivables
Program	Aging - 7 Buckets - By Amount Report (obsolete)	Receivables
Program	Aging - 4 Buckets Report	Receivables
Program	Aging - 7 Buckets - By Salesperson/Agent	Receivables
Program	Aging - 7 Buckets Report (obsolete)	Receivables
Program	Billing and Receipt History	Receivables

Description: Workflow background process for deferred and timeout activities

Request Group Form

3. Add an entry for Concurrent Program in the Request Group and save the work.

The screenshot shows the 'Request Groups' dialog box with the following details:

- Group:** Receivables All
- Application:** Receivables
- Code:** (empty)
- Description:** All reports and programs
- Requests:** A table listing various programs under the 'Receivables' application. A new program row has been added and is highlighted.

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	XXTEST: AR Transactions Interface Program	Receivables
Program	Publish RX Reports	Assets
Program	Collection Effectiveness Indicators	Receivables
Program	Autoinvoice Import Program	Receivables
Program	Aging - 7 Buckets - By Amount Report (obsolete)	Receivables
Program	Aging - 4 Buckets Report	Receivables
Program	Aging - 7 Buckets - By Salesperson/Agent	Receivables
Program	Aging - 7 Buckets Report (obsolete)	Receivables

Description: Main program for AR Transaction Interface Program

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and further it can be used to install in another Application Server.

Before downloading the script of Concurrent Program, connect your local system to Applications Server

Run the Following Command to download the script

```
FNDLOAD <Apps User/Apps Pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<Concurrent_Program_Short_Name>
```

For Example:

- To Download the “[XX<EMPID>: AR Transactions Interface Loader Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct XXARINVOICELOAD.ldt
PROGRAM CONCURRENT_PROGRAM_NAME =
XXARINVOICELOAD
```

- To Download the “[XX<EMPID>: AR Transactions Interface Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct XXARINVOICEMAIN.ldt
PROGRAM CONCURRENT_PROGRAM_NAME =
XXARINVOICEMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



1. Save the attached zip file in local system
2. Transfer the zip to the Oracle Applications Server
3. Create a Staging Directory [E.g. XXTESTARINTOIT] in temporary area on Oracle Application Server.
4. `mkdir -p XXTESTARINTOIT`
5. Change the Permissions on the temp directory.
6. `chmod 755 XXTESTARINTOIT`
7. FTP the **XXTESTARINTOIT.tar.gz** file in BINARY mode to directory **XXTESTARINTOIT**
8. Change directory to XXTESTARINTOIT as given below
9. `cd XXTESTARINTOIT`
10. Uncompress **XXTESTARINTOIT.tar.gz** as given below
11. `gunzip XXTESTARINTOIT.tar.gz`
12. Untar the file **XXTESTARINTOIT.tar** using
13. `tar -xvf XXTESTARINTOIT.tar.gz`
14. Grant the execute permission on the install script using
15. `chmod 755 XX_AR_INVOICE.install`
16. Run `XX_AR_INVOICE.install`
17. `sh XX_AR_INVOICE.install`
18. Enter the apps schema user name password (apps/<apps_pwd>), when asked for.

Extract File Layout

Please find the attached extract file for AR Transactions.



Case_Study_AR_TRX_Extract_Data.csv

Note: Please remove the header row before putting the same file on Oracle Application Server for loading purpose.

Below is the example of extract file for AR Transactions

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Inv Class	Trx Type	Trx Num	Curr	Cust Num	Loc	Pay Term	Inv Item	Line Nu	Desc	UOM	Amt	Price	Quantity
2	INVOICE	Interest Invoice	1000180277	EUR	1001	New York	30 NET	0001-0120H	1	AR Transaction Interface Test	Ea	30	10	3
3	INVOICE	Interest Invoice	1000180277	EUR	1001	New York	31 NET	0001-0120H	2	AR Transaction Interface Test	Ea	30	10	3
4	INVOICE	Interest Invoice	1000180277	EUR	1001	New York	32 NET	0001-0120H	3	AR Transaction Interface Test	Ea	30	10	3
5	Credit Memo	BR Credit Memo	3005376112	EUR	1002	Oakdale	33 NET	0001-0120H	1	AR Transaction Interface Test	Ea	80	20	4
6	Credit Memo	BR Credit Memo	3005376112	EUR	1002	Oakdale	34 NET	0001-0120H	2	AR Transaction Interface Test	Ea	80	20	4
7	Credit Memo	BR Credit Memo	3005376112	EUR	1002	Oakdale	35 NET	0001-0120H	3	AR Transaction Interface Test	Ea	80	20	4
8	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	36 NET	0001-0120H	1	AR Transaction Interface Test	Ea	150	30	5
9	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	37 NET	0001-0120H	2	AR Transaction Interface Test	Ea	150	30	5
10	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	38 NET	0001-0120H	3	AR Transaction Interface Test	Ea	150	30	5
11	Debit Memo	BR Debit Memo	3005381141	EUR	1003	Atlanta	39 NET	0001-0120H	4	AR Transaction Interface Test	Ea	150	30	5
12														
13														
14														
15														
16														
17														
18														
19														
20														
21														
22														
23														
24														
25														
26														
27														
28														
29														
30														
31														

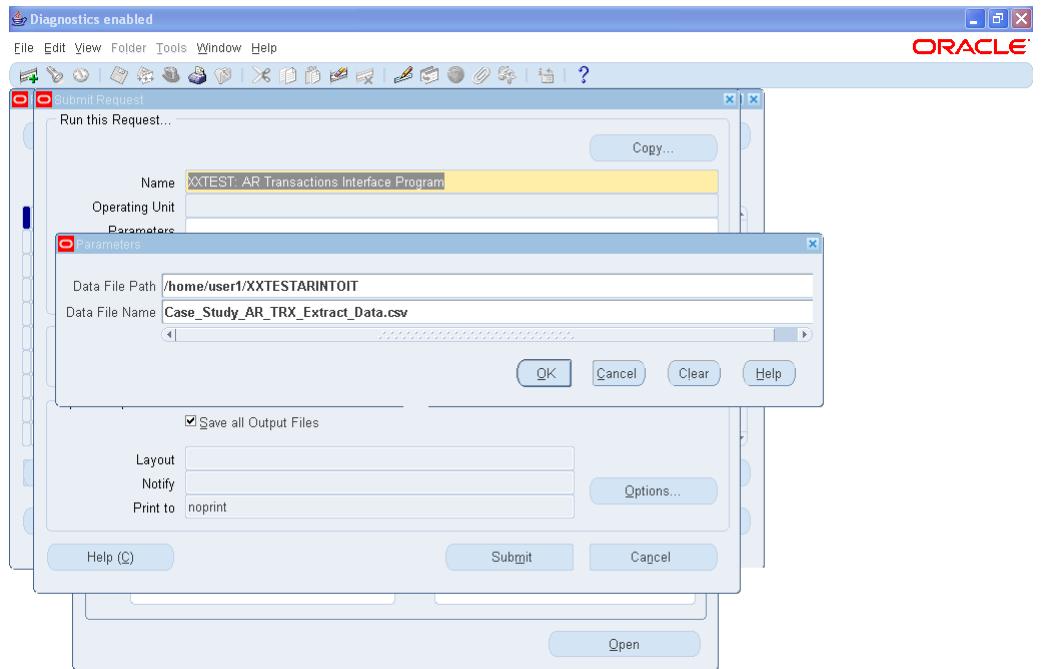
Extract File Mapping to Staging Table Columns

Data File Columns	Datatype	Size	Staging Table's Columns
Invoice_class	Varchar2	50	Invoice_class
Trx_type	Varchar2	20	Trx_type
Trx_num	Varchar2	20	Trx_num
Curr_code	Varchar2	10	Curr_code
Customer_number	Varchar2	30	Customer_number
Location	Varchar2	10	Location
Payment_term	Varchar2	10	Payment_term
Inventory_item	Varchar2	40	Inventory_item
Line_number	Number		Line_number
Description	Varchar2	250	Description
Uom	Varchar2	10	Uom
Amount	Number		Amount
Price	Number		Price
Quantity	Number		Quantity

Calling Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “[XX<EMPID>: AR Transactions Interface Program](#)”. To run the concurrent program navigate to [Receivables Responsibility](#).

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



SRS Form

1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Calling Standard Import Concurrent Program

When “[XX<EMPID>: AR Transactions Interface Program](#)” completed successfully, run the standard import program to populate the data from AR Transaction Interface Table (RA_INTERFACE_LINES_ALL) to Receivables base table.

The standard Import “[Autoinvoice Import](#)” program will fetch the data from interface table and populate it into the receivables base tables.

Call the “[AutoInvoice Import](#)” Program with the following Parameters

- a. Transaction Source: Test
- b. Default Date: Sysdate
- c. Base Due Date on Trx Date: No

Query interface records

When “AutoInvoice Import” standard import program completes successfully navigates to **Receivables Responsibility**, Select **Transactions** -> **Transactions** and queries the records of interface table.

The screenshot displays the Oracle Applications - ADS Vision AR Transaction Form. The main window title is "Transactions (Vision Operations - USD)". The form is divided into several sections:

- Transaction:** Includes fields for Source (OKS_CONTRACTS), Number (125), Class (Invoice), Type (Invoice-OKS), Reference (15543), Legal Entity (Vision Operations), Date (20-JUN-2001), GL Date (20-JUN-2001), Currency (USD), Document Num, Transaction, and a checkbox for Complete.
- Balance Due:** Displays Line 0.00, Tax 0.00, Freight 0.00, Charges 0.00, and Total 0.00.
- Main Tab:** Contains fields for Name, Number, Location, Address, Contact, Commitment, Payment Term (Immediate), Invoicing Rule (In Advance), and Due Date (20-JUN-2001).
- Ship To:** Fields for Name, Number, Location, Address, and Contact.
- Bill To:** Fields for Name, Number, Location, and Address, showing "World of Business" at "1000 San Jose (OPS) 2391 L Street San Jose, CA 95106 United States".
- Sold To:** Fields for Name (World of Business) and Number (1000).
- Paying Customer:** Fields for Name, Number, and Location.
- Payment Details:** Fields for Receipt Method, Payment Method, Instrument Number, and a "Select Instrument" button.
- Buttons:** Line Items, Tax, Freight, Distributions, Sales Credits, and Incomplete.
- Status Bar:** Shows Record: 1/2 and <OSC>.

AR Transaction Form

Error Handling

Following error can occur while processing the AR Transaction data file?

1. During loading data into staging table.

- When error occurred while loading the data file using SQL*Loader concurrent program it generates two files they are
 - **SQL*Loader Bad File:** The bad file contains records that weren't loaded into the staging table. These records could have been rejected by SQL*Loader due to an invalid format. Also they could have been rejected by the Oracle database if they violate an integrity constraint or had an invalid data type for the staging table.
 - **SQL*Loader Log File:** Detailed information about the load is stored in the log file. Any errors found during parsing of the control file are stored in the log file. The log file also identifies the number of records successfully loaded. The log file must be available during the entire run of the SQL*Loader. When loading data with SQL*Loader, nothing should be assumed without reviewing the log files

2. Validating the records in staging table

- While validating the data in staging table it can complete with error and the concurrent program will complete with warning, in this case look into the output report of concurrent program and correct the data file and re-load it again.

3. While calling the Standard AutoInvoice Import Program

- After populating the records in interface table it again validated by the standard import program and in few cases it completes with error or warning, in this case see the output report of Standard Import Program and correct the data in data file and re-load it, than re-run the custom concurrent program.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Sales Order Extract Outbound

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Business Rules	3
Approach.....	3
Module List.....	4
Registering Concurrent Program	7
Downloading Concurrent Program Script	13
Deployment.....	14
Extract File Layout	15
Calling Extract Custom Concurrent Program	16
Open and Closed Issues for this Deliverable	17
Open Issues	17
Closed Issues.....	17

Introduction

A sales order is an internal document of the company, which should record the customer's originating purchase order.

You can enter, view and update sales orders using the Sales Orders window. You can order standard items, both shippable and non-shippable and configurations using this window.

To enter or query a Sales Order from front end, navigate to Order Management Responsibility and select **Orders -> Returns -> Sales Orders**.

This document demonstrates the extract of sales order data for the previous year.

Purpose

This document describes the:

- This case study is intended to demonstrate the extraction of data from Oracle Applications Sales Order for legacy system.
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during the implementation of Oracle Order Management.

Scope and Application

The following boundaries are specific to the extractions of Sales Order data using [OE_ORDER_HEADERS_ALL](#), [OE_ORDER_LINES_ALL](#), [MTL_SYSTEM_ITEMS](#), [RA_SALESREPS_ALL](#) and [OE_TRANSACTION_TYPES_ALL](#) standards table

- All standard sales orders are available in Oracle Applications
 1. The '[XX<EMPID>: Sales Orders Extract Program](#)' creates a file which will contain Sales Order data which are approved and received.
 2. The '[XX<EMPID>: Sales Orders Extract Program](#)' has two parameters.

- **Data File Path:** Enter the path of Oracle Applications Server where this custom concurrent program will create the file for outbound.
 - **Data File Name:** Enter the file name, which will be used for naming the file while extracting the data from Oracle Applications.
-

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.
-

Source system

The implementation team studies the source system and understands the data required to be migrated from Oracle Applications to legacy system. This data is then extracted and cleansed from the Oracle Applications (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Order Management Super User
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.
- Users have access to Oracle Applications database (apps schema)
- All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.

Pre-Requisites

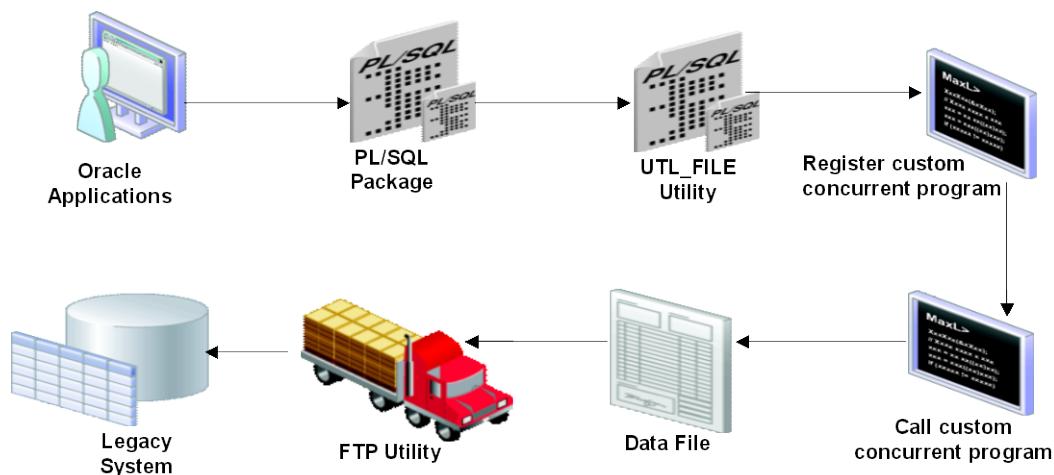
Sales order records should be available in Oracle Applications

Business Rules

Following business rules are applicable for the Sales Order extract process.

- Extract all the purchase order records for the previous year
- Please refer Extract file layout for the columns to be extracted from the sales order.

Approach



The Approach for extracting the Sales Order data from Oracle Applications to a data file is

1. Login to the Oracle Applications
2. Using PL/SQL, write a package
3. Use UTL_FILE utility in PL/SQL package to create the data file
4. Data file would be created by custom concurrent program
5. Use any FTP utility to transfer the extract data file into legacy system.

Module List

Concurrent Programs

['XX<EMPID>: Sales Order Extract Program'](#)

The '[XX<EMPID>: Sales Order Extract Program](#)' concurrent program is based on the stored procedure **XXOM_ORDER_OUT_PKG.main** that is registered as a PL/SQL executable.

- The above program calls the stored procedures to create the data file (Data file name and path passed as parameter values)

NOTE:

1. Change <EMPID> with your Employee ID,
2. Follow the steps provided in 'Registering Concurrent Program' section for registering the concurrent program.

Stored Procedures

[XX_OM_ORDER_PKG](#) package consists of the following stored procedures:

[XXOM_ORDER_OUT_PKG.main](#)

This is the main procedure that calls the `create_headers` and `create_lines` procedures.

[XXOM_ORDER_OUT_PKG.create_headers](#)

This procedure is used to create the header for the extract file.

[XXOM_ORDER_OUT_PKG.create_lines](#)

This procedure is used to create the lines for the extract file.

NOTE: Please refer the attached documents for package specification and package body. Kindly open the attached script in notepad.



XXOM_ORDER_OUT_
PKG.pks

1. Package Specification:



XXOM_ORDER_OUT_
PKG.pkb

2. Package Body:

Grant Script

If the table and package are created in custom schema, then grant the table and package to APPS schema

NOTE: Please refer to the attached script for table and package granting.
Kindly open the attached script in notepad.



XXOM_ORDER_OUT_
PKG.grt

Package Grant Script:

Synonym Script

If table and package are created in custom schema, then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym.
Kindly open the attached script in notepad.



XXOM_ORDER_OUT_
PKG.syn

Package synonym script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script

NOTE: Please refer the attached Install script for deployment of OM Sales Orders object on Oracle Applications Server. Kindly open the attached script in notepad.



XXOM_ORDER_OUT.i
nstall

Install Script:

Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

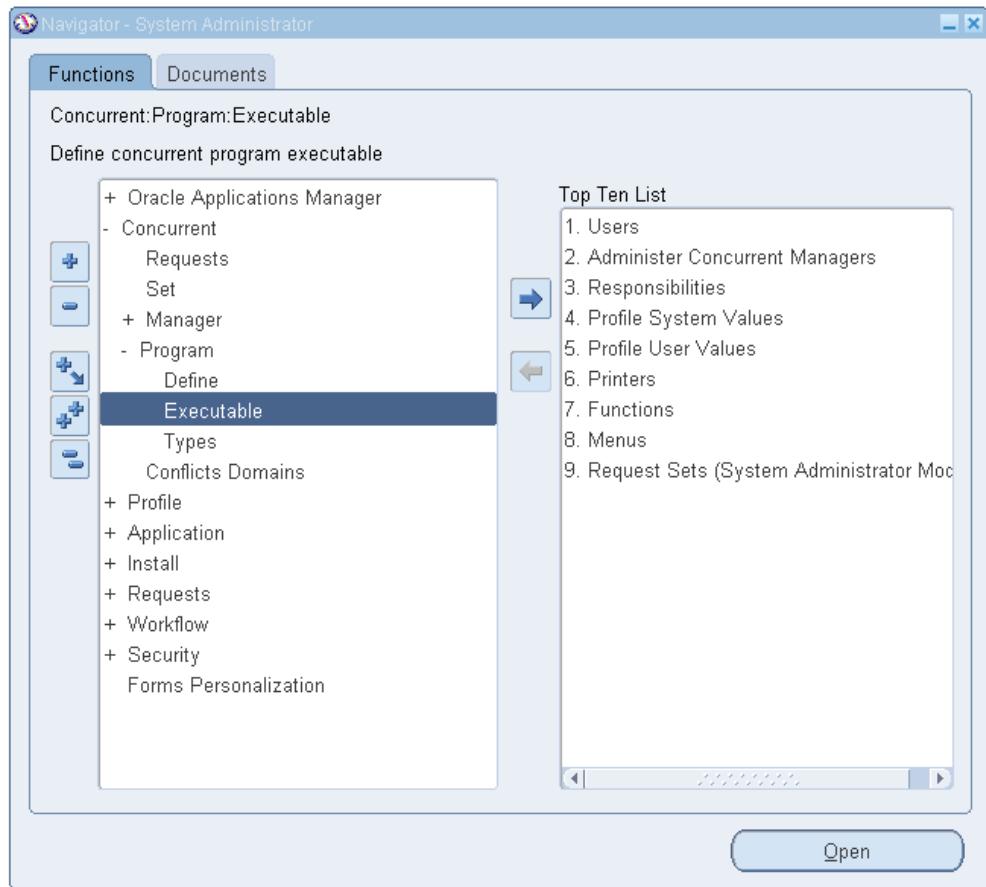
- A. **Package Name**
- B. **Synonym Name**
- C. **Concurrent Program Executable**
- D. **Concurrent Program Name**

Registering Concurrent Program

Registering Extract Concurrent Program

To register a concurrent program for ‘XX<EMPID>: Sales Order Extract Program’ in Oracle Applications Navigate to [System Administrator Responsibility](#)

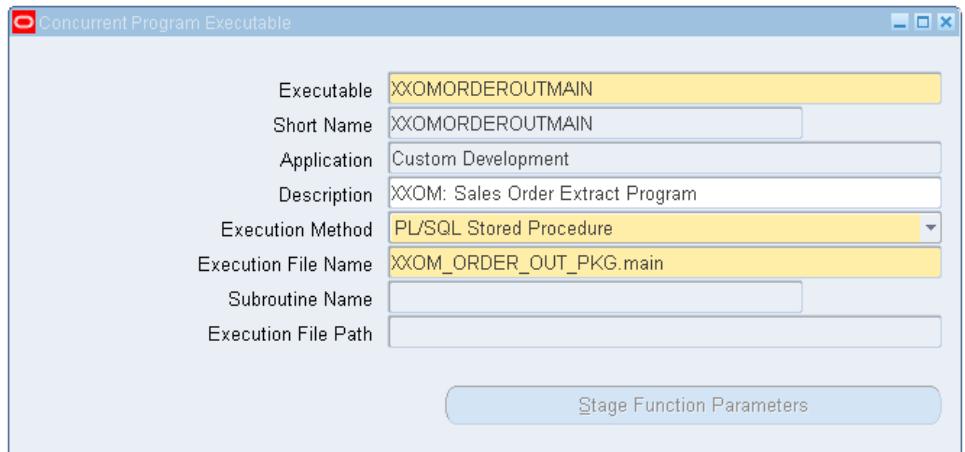
1. Select Concurrent -> Program -> Executable



2. Enter The following in the Concurrent Program Executable Form

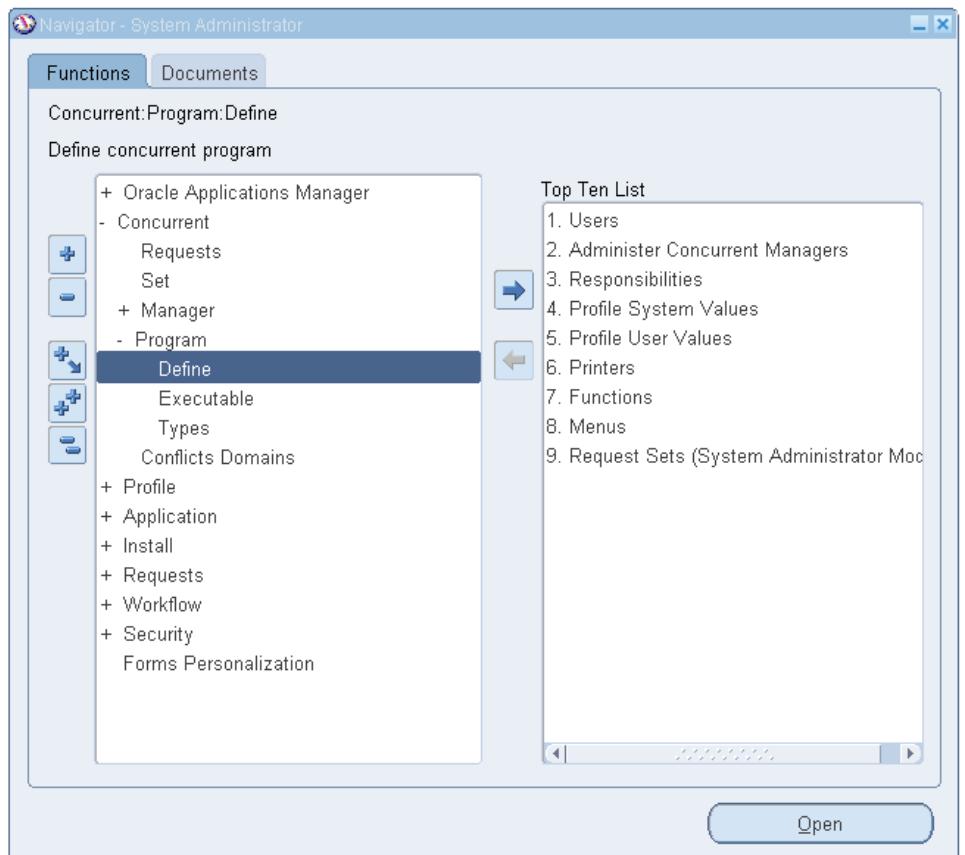
- a. **Executable Name:** Enter a unique name for executable.
- b. **Short Name:** Enter a unique short name and this short name later will be used for defining Concurrent Program.
- c. **Application:** Enter the Application Name from the List of Values (LOV). As we are developing an interface program for OM Sales Orders, the Application Name should be [Custom Development](#).
- d. **Description:** Enter the description of Executable
- e. **Execution Method:** Execution method should be PL/SQL Stored Procedure.

- f. **Execution File Name:** Here we would be passing the package name with one public procedure. As discussed in Module List the package name would be **XXOM_ORDER_OUT_PKG** and the public procedure define in this package is **main**.



Concurrent Program Executable Form

3. Select Concurrent -> Program -> Define



4. Enter the following in the Concurrent Program Form

- g. **Program Name:** Enter “XX<EMPID>: Sales Order Extract Program”
- h. **Short Name:** Enter a unique concurrent program short name, for example: **XXOMORDEROUTMAIN**
- i. **Application:** Select the value from LOV. As we would be executing this concurrent program from Order Management Applications so select **Custom Development**.
- j. **Description:** Enter the description of this concurrent program.
- k. **Executable Name:** Select the executable name from LOV. As we have already registered an executable for the loader program, select **XXOMORDEROUTMAIN** from LOV.
- l. **Executable Method:** Executable method will be populated automatically.
- m. **Parameters (if any):** Select the Parameters Button to specify the parameter list. In this case, two values should be passed
 - i) Data File Path
 - ii) Data File Name

The screenshot shows the 'Concurrent Programs' window with the following details:

- Program:** XXOM: Sales Order Extract Program
- Short Name:** XXOMORDEROUTMAIN
- Enabled:** Checked
- Application:** Custom Development
- Description:** XXOM: Sales Order Extract Program
- Executable** section:
 - Name:** XXOMORDEROUTMAIN
 - Method:** PL/SQL Stored Procedure
 - Options:** (empty)
 - Priority:** (empty)
- Request** section:
 - Type:** (empty)
 - Incrementor:** (empty)
 - MLS Function:** (empty)
 - Checkboxes:**
 - Use in SRS
 - Allow Disabled Values
 - Run Alone
 - Restart on System Failure
 - Enable Trace
 - NLS Compliant
- Output** section:
 - Format:** Text
 - Save (C)
 - Print
 - Columns:** (empty)
 - Rows:** (empty)
 - Style:** (empty)
 - Style Required
 - Printer:** (empty)
- Buttons:** Copy to..., Session Control, Incompatibilities, Parameters

Concurrent Program Definition Form

Seq	Parameter	Description	Enabled
10	p_data_file_path	Data File Path	<input checked="" type="checkbox"/>
20	p_data_file_name	Data File Name	<input checked="" type="checkbox"/>

Validation

Value Set	100 Characters	Description	100 Characters
Default Type		Default Value	
<input type="checkbox"/> Required	<input type="checkbox"/> Enable Security	Range	

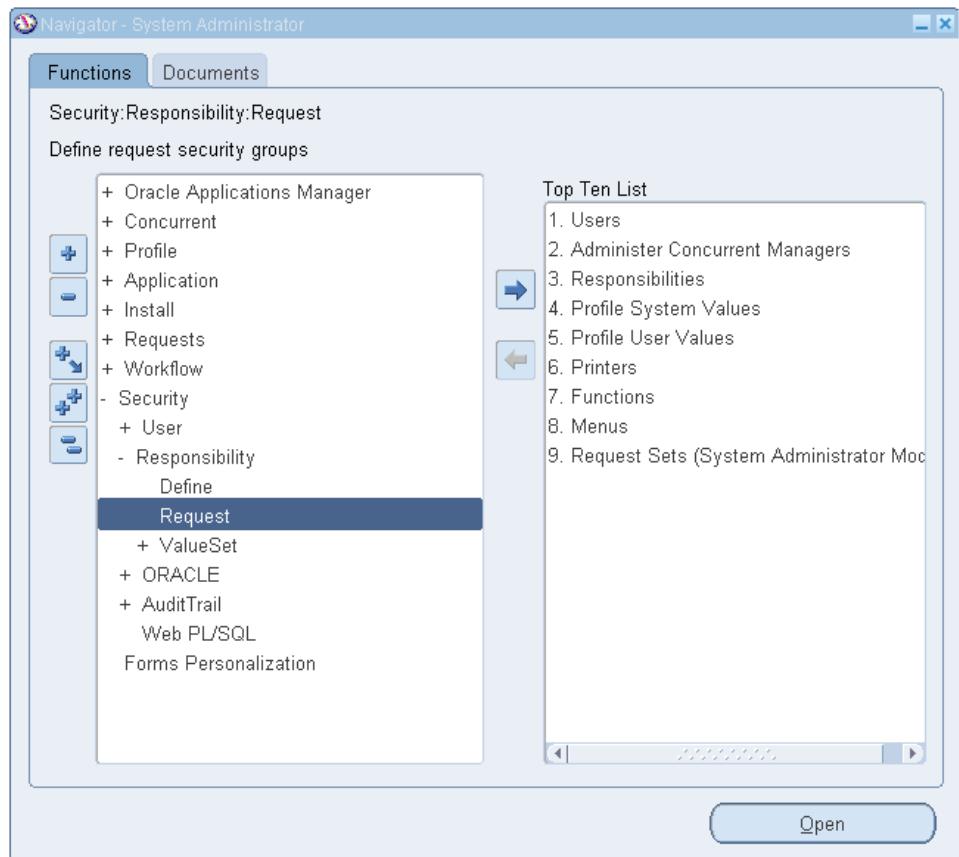
Display

Display Size	50	Description Size	50
Concatenated Description Size	25	Prompt	Data File Name

Concurrent Program Parameters Form**Registering Interface Concurrent Program in Request Group**

To register a concurrent program ‘XX<EMPID>: Sales Order Extract Program’ in Request Group, Navigate to [System Administrator Responsibility](#)

1. Select Security -> Responsibility-> Request



2. Query the Request Group (OM Concurrent Programs) and Applications (Order Management)

The screenshot shows the Request Groups form with the following details:

Group: OM_Concurrent_Programs

Application: Order Management

Code: OM_CONC_PROGRAMS

Description: Order Management Concurrent Programs

Requests:

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	Expense Report Import (Obsolete)	Payables
Program	Payables Open Interface Import	Payables
Program	Requisition Import	Purchasing
Program	Requisition Import Exceptions Report	Purchasing
Program	Create Internal Orders	Purchasing
Program	Revenue Contingency Analyzer	Receivables
Program	Defaulting Generator	Order Management
Program	Order Import	Order Management
Program	Schedule Orders	Order Management

Description: Workflow background process for deferred and timeout activities

Request Group Form

3. Add an entry for the concurrent program in the Request Group and save the work.

The screenshot shows the 'Request Groups' window with the following details:

- Group:** OM Concurrent Programs
- Application:** Order Management
- Code:** OM_CONC_PROGRAMS
- Description:** Order Management Concurrent Programs
- Requests:** A grid of concurrent programs:

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	XXOM: Sales Order Extract Program	Custom Development
Program	Expense Report Import (Obsolete)	Payables
Program	Payables Open Interface Import	Payables
Program	Requisition Import	Purchasing
Program	Requisition Import Exceptions Report	Purchasing
Program	Create Internal Orders	Purchasing
Program	Revenue Contingency Analyzer	Receivables
Program	Defaulting Generator	Order Management
Program	Order Import	Order Management
- Description:** XXOM: Sales Order Extract Program

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and it can be further used to install in another Application Server.

Before downloading the script of the concurrent Program, connect to your local system to Applications Server

Run the following command to download the script

```
FNDLOAD <apps_user/apps_pwd> 0 Y DOWNLOAD  
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM  
CONCURRENT_PROGRAM_NAME=<concurrent_program_short_name>
```

For Example:

- To download the “XX<EMPID>: Sales Order Extract Program” script, enter the following command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD  
$FND_TOP/patch/115/import/afcpprog.lct  
XXOMORDEROUTMAIN.ldt PROGRAM  
CONCURRENT_PROGRAM_NAME=XXOMORDEROUTMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



- Save the attached zip file in local system
- Transfer the zip to the Oracle Applications Server
- Create a Staging Directory [Eg. XXOMORDEROUT] in temporary area on Oracle Application Server.

```
mkdir -p XXOMORDEROUT
```

- Change the Permissions on the temp directory.

```
chmod 755 XXOMORDEROUT
```

- FTP the **XXOMORDEROUT.tar.gz** file in BINARY mode to directory **XXOMORDEROUT**
- Change directory to XXOMORDEROUT as given below

```
cd XXOMORDEROUT
```

- Uncompress **XXOMORDEROUT.tar.gz** as given below

```
gunzip XXOMORDEROUT.tar.gz
```

- Untar the file **XXOMORDEROUT.tar** using

```
tar -xvf XXOMORDEROUT.tar.gz
```

- Grant the execute permission on the install script using

```
chmod 755 XXOM_ORDER_OUT.install
```

- Run **XXOM_ORDER_OUT.install**

```
sh XXOM_ORDER_OUT.install
```

- Enter the apps schema user name password (apps/ <apps_pwd>) when asked for.

Extract File Layout

Data File Columns	Datatype	Size	Table Name
ORDER_NUMBER	NUMBER		OE_ORDER_HEADERS_ALL
LINE_NUMBER	NUMBER		OE_ORDER_LINES_ALL
CURRENCY	VARCHAR2	15	FND_CURRENCIES
CUSTOMER	VARCHAR2	360	HZ_PARTIES
LOCATION	VARCHAR2	240	HZ_CUST_SITEUSES_ALL
ORDER_TYPE	VARCHAR2	40	OE_TRANSACTION_TYPES
ORDER_CATEGORY	VARCHAR2	240	OE_TRANSACTION_TYPES_ALL
ITEM	VARCHAR2	40	MTL_SYSTEM_ITEMS
DESCRIPTION	VARCHAR2	240	MTL_SYSTEM_ITEMS
PAYMENT_TERM	VARCHAR2	15	RA_TERMS
QUANTITY	NUMBER		OE_ORDER_LINES_ALL
UNIT_OF_MEASURE	VARCHAR2	3	OE_ORDER_LINES_ALL
PRICE_LIST	VARCHAR2	2000	QP_LIST_HEADERS
SALESREP	VARCHAR2	240	RA_SALESREPS_ALL
STATUS	VARCHAR2	30	OE_ORDER_LINES_ALL

Please find the attached extract file for Sales Order, which has been extracted from Oracle Applications.



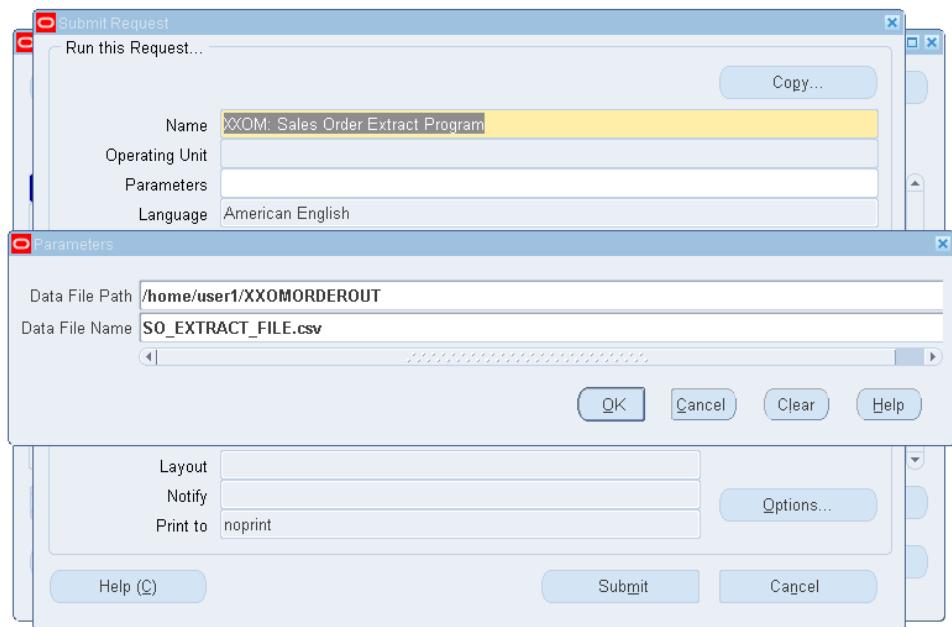
Calling Extract Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “XX<EMPID>: Sales Order Extract Program”. To run the concurrent program navigate to [Order Management Responsibility](#).

Use the following query to find out the path to create the extract file and use one of the path from value as parameter value for Data File path.

```
SELECT VALUE
FROM V$PARAMETER
WHERE NAME LIKE '%utl_file_dir%';
```

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



SRS Form

1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Migration of OM Sales Order using Standard API

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Data Dependencies.....	3
Business Rules	4
Approach.....	4
Module List.....	5
Registering Concurrent Program	9
Downloading Concurrent Program Script	18
Deployment.....	19
Extract File Layout	20
Extract File Mapping to Staging Table Columns	21
Calling Custom Concurrent Program.....	22
Query Interface Records	23
Error Handling	24
Open and Closed Issues for this Deliverable	25
Open Issues	25
Closed Issues.....	25

Introduction

A sales order is an internal document of the company, which should record the customer's originating purchase order.

You can enter, view and update sales orders using the Sales Orders window. You can order standard items, both shippable and non-shippable and configurations using this window.

To enter or query a Sales Order from front end, navigate to Order Management Responsibility and select **Orders -> Returns -> Sales Orders**.

This document demonstrates the use of standard API to migrate OM Sales Order from source system to Oracle Applications.

Purpose

This document describes the:

- This case study is intended to demonstrate the creation of Standard Sales Order.
- Detailed data mapping from the source system(s) to the Oracle Applications E-Business Suite (EBS) Order Management (OM) Sales Order records
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during the implementation of Oracle Order Management.

Scope and Application

The following boundaries are specific to the interface of OM Sales Order using [OE_ORDER_PUB.process_order](#) standard API.

- All Sales Orders and Returns would be migrated to Oracle Applications

- The '[XX<EMPID>: OM Sales Order Interface API Program](#)' uploads OM Sales Order data from data file into staging table "XXOM_ORDER_API_STG," performs validation and transfers the valid records into Oracle Order Management interface tables. There is no need to run standard import program 'Order Import' to perform the migration OM Sales Orders data from staging table to Order Management base tables because the API itself will migrate the data from staging table to Order Management base tables.
 1. The OM Sales Order data will be provided through data file. This data file will be loaded into staging table(s) on the Oracle applications database using loader program.
 2. The loaded data will be validated from staging tables and then passed to the API as parameters.
 3. As the API itself will populate the data from staging table to the Order Management base tables, there is no need of running Order Import program.

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.

Source system

The implementation team studies the source system and understands the data required to be migrated to Oracle Applications. This data is then extracted and cleansed from the source system (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team. The data mapping section describes the data format of source system to Oracle Applications.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Order Management Super User
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.

- Users have access to Oracle Applications database (apps schema)
 - All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.
 - The extract-file layout must be in the format as specified in the extract file layout section
-

Pre-Requisites

Prerequisite set-ups are required for the interface. These setups include, but are not limited to:

1. Transaction Types
2. Customers
3. Price Lists
4. Items assigned to Price Lists
5. Discounts assigned to Price Lists
6. Salespersons
7. Tax Codes and Tax Rates
8. Invoicing and Accounting Rules
9. Freight Carriers, Freight Terms, and FOB options
10. Shipping Warehouses

Data Dependencies

OM Sales Order

In order to migrate Sales Orders, the following entities must have been migrated successfully:

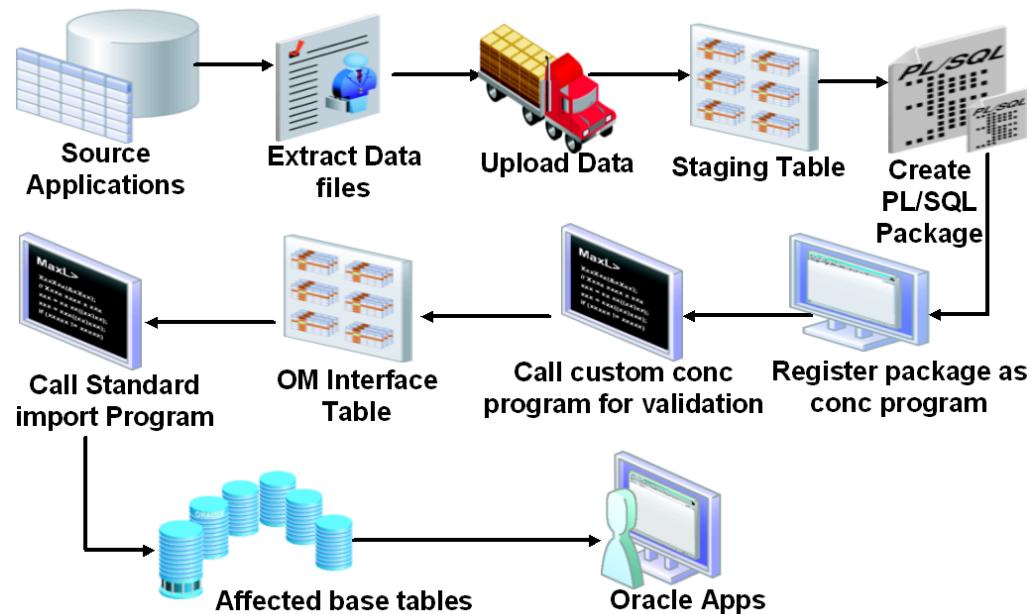
- Customers
- Price Lists
- Items

Business Rules

Following business rules are applicable for the OM Sales Order conversion process.

- N/A

Approach



The Approach for migrating OM Sales Order data from source system to Order Management open interface table

1. Extract the data file from source system
2. Load the data file into staging table
3. Using PL/SQL write a package
4. Register the PL/SQL package as concurrent program in Oracle Applications
5. Call the custom concurrent program for validation and using standard API populate the records available in the staging table to Order Management base tables.
6. Query the records that are created from the staging table and see the result in Oracle Applications front end.

Module List

Concurrent Programs

‘XX<EMPID>: OM Sales Order Interface API Program’ includes the following concurrent programs:

‘XX<EMPID>: OM Sales Order Interface API Program’

The ‘XX<EMPID>: OM Sales Order Interface API Program’ concurrent program is based on the stored procedure **XX_OM_ORDER_PKG.main** that is registered as a PL/SQL executable.

- The above program calls the loader program (**XX<EMPID>: OM Sales Order Interface API Loader Program**) which loads the data into staging tables, performs validations on the staging table data and calls the standard API to populate the staging table records in Order Management base tables.

‘XX<EMPID>: OM Sales Order Interface API Loader Program’

The ‘XX<EMPID>: OM Sales Order Interface API Loader Program’ concurrent program is based on the SQL*Loader control file, that is registered as a Loader executable.

- The above program loads data from the path specified as parameter value in loader concurrent program to the staging table.

NOTE:

1. Change <EMPID> with your Employee ID,
2. Follow the steps provided in ‘Registering Concurrent Program’ section for registering the concurrent program.

Stored Procedures

XX_OM_ORDER_PKG package consist the following stored procedures:

XX_OM_ORDER_PKG.main

This is the main procedure that calls the load, validate, import, print_error and record_summary procedures.

XX_OM_ORDER_PKG.validate

This procedure performs the necessary validations on the staging tables.

XX_OM_ORDER_PKG.import

This procedure will call the standard API 'OE_ORDER_PUB.process_order' which creates the records in Order Management base table for OM Sales Orders.

XX_OM_ORDER_PKG.print_error

This procedure reports the errored records in the log file.

XX_OM_ORDER_PKG.record_summary

This procedure displays the status of the staging table records

NOTE: Please refer the attached documents for Package specifications and Package body. Kindly open the attached script in notepad.



XX_OM_ORDER_API
_PKG.pks

1. Package Specification:



XX_OM_ORDER_API
_PKG.pkb

2. Package Body:

Staging Table

XXOM_ORDER_API_STG table is created in the database to store records from source system data file.

NOTE: Please refer to the attached script for staging table creation. Kindly open the attached script in notepad.



XX_OM_ORDER_API
_TAB.tab

Table Creation Script:

Grant Script

If the table and package are created in custom schema, then grant the table and package to APPS schema

NOTE: Please refer to the attached script for table and package granting. Kindly open the attached script in notepad.



XX_OM_ORDER_API
_PKG.grt

1. Package Grant Script:



XX_OM_ORDER_API
_TAB.grt

2. Table Grant Script:

Synonym Script

If table and package are created in custom schema, then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



XX_OM_ORDER_API
_PKG.syn

1. Package synonym script:



XX_OM_ORDER_API
_TAB.syn

2. Table synonym script:

SQL*Loader Script

SQL*Loader is controlled by its own data definition language, which is kept in the control file. The control file describes the data to be loaded, the destination tables of the data and describes the interdependency between the data and the columns within the tables.

NOTE: Please refer the attached SQL* Loader script for loading data file into staging table. Kindly open the attached script in notepad.



XX_OM_ORDER_API
_LOAD.ctl

SQL* Loader Script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script
3. Staging Table Script
4. SQL*Loader Script

NOTE: Please refer the attached Install script for deployment of OM Sales Orders object on Oracle Applications Server. Kindly open the attached script in notepad.



XX_OM_ORDER_API.
install

Install Script:

Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

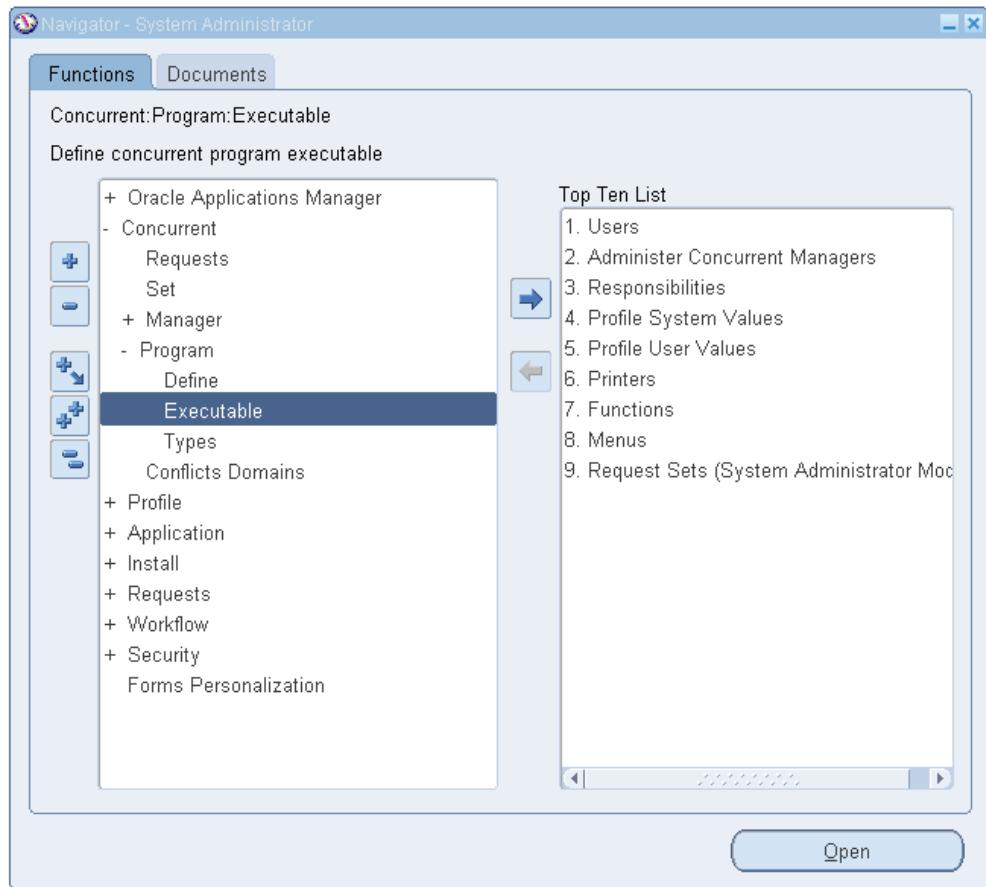
- A. **Package Name**
- B. **Table Name**
- C. **Synonym Name**
- D. **Concurrent Program Executable**
- E. **Concurrent Program Name**

Registering Concurrent Program

Registering Interface Loader Concurrent Program

To register a concurrent program for ‘XX<EMPID>: OM Sales Order Interface API Loader Program’ in Oracle Applications Navigate to [System Administrator Responsibility](#)

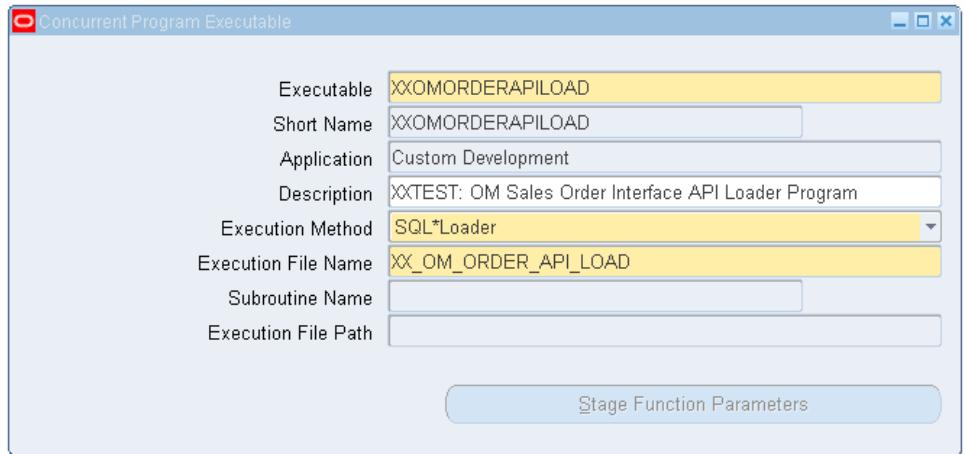
1. Select Concurrent -> Program -> Executable



3. Enter The Following in the Concurrent Program Executable Form

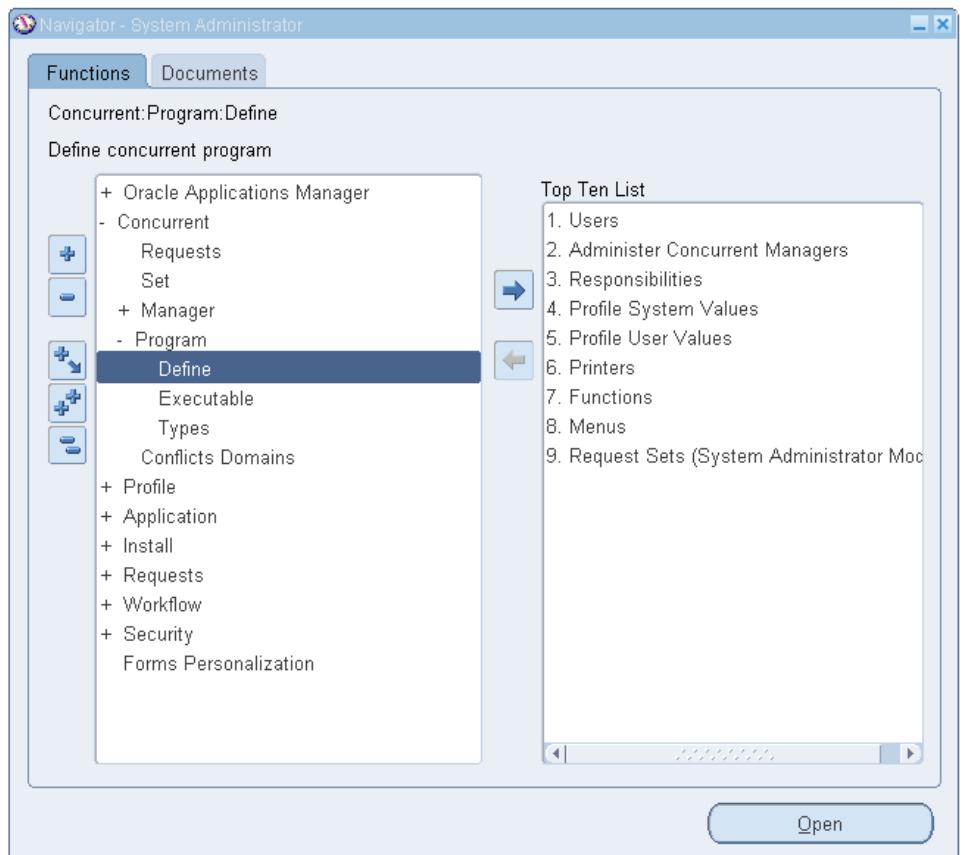
- Executable Name:** Enter a unique name for executable.
- Short Name:** Enter a unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from the List of Values (LOV). As we are developing an interface program for OM Sales Orders, the Application Name should be [Custom Development](#).
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be SQL*Loader, because we are registering a concurrent program for loading the data to staging table.

- f. **Execution File Name:** Here we would be passing the file name of SQL*Loader control file. As discussed in Module List the control file name would be **XX_OM_ORDER_API_LOAD**



Concurrent Program Executable Form

4. Select Concurrent -> Program -> Define



5. Enter the following in the Concurrent Program Form

- a. **Program Name:** Enter “**XX<EMPID>: OM Sales Order Interface API Loader Program**”

- b. **Short Name:** Enter a unique concurrent program short name, for example: **XXOMORDERAPILOAD**
- c. **Application:** Select the value from LOV. As we would be executing this concurrent program from Order Management Applications so select **Custom Development**.
- d. **Description:** Enter the description of this concurrent program.
- e. **Executable Name:** Select the executable name from LOV. As we have already registered an executable for the loader program, select **XXOMORDERAPILOAD** from LOV.
- f. **Executable Method:** Executable method will be populated automatically.
- g. **Parameters (if any):** Select the Parameters Button to specify the parameter list. In this case, two values should be passed in a single parameter - Data File Name and Data File Path

The screenshot shows the Oracle Concurrent Programs window. The 'Program' field contains 'XXTEST: OM Sales Order Interface API Loader Program'. The 'Short Name' field is set to 'XXOMORDERAPILOAD'. Under the 'Application' section, 'Custom Development' is selected. The 'Description' field contains the same text as the program name. In the 'Executable' section, the 'Name' field is 'XXOMORDERAPILOAD' and the 'Method' field is 'SQL*Loader'. The 'Request' section includes fields for 'Type', 'Incrementor', and 'MLS Function'. Under 'Request' settings, 'Use in SRS' is checked, while 'Run Alone' and 'Enable Trace' are unchecked. 'Allow Disabled Values', 'Restart on System Failure', and 'NLS Compliant' are also listed. The 'Output' section specifies 'Format' as 'Text', with 'Save' and 'Print' options checked. 'Columns', 'Rows', 'Style', and 'Printer' fields are present but empty. At the bottom, buttons for 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters' are visible.

Concurrent Program Definition Form

Concurrent Program Parameters Form

Seq	Parameter	Description	Enabled
10	p_data_file	Data file name with file path	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>

Validation

Value Set	100 Characters	Description	100 Characters
Default Type		Default Value	
<input type="checkbox"/> Required	<input type="checkbox"/> Enable Security	Range	

Display

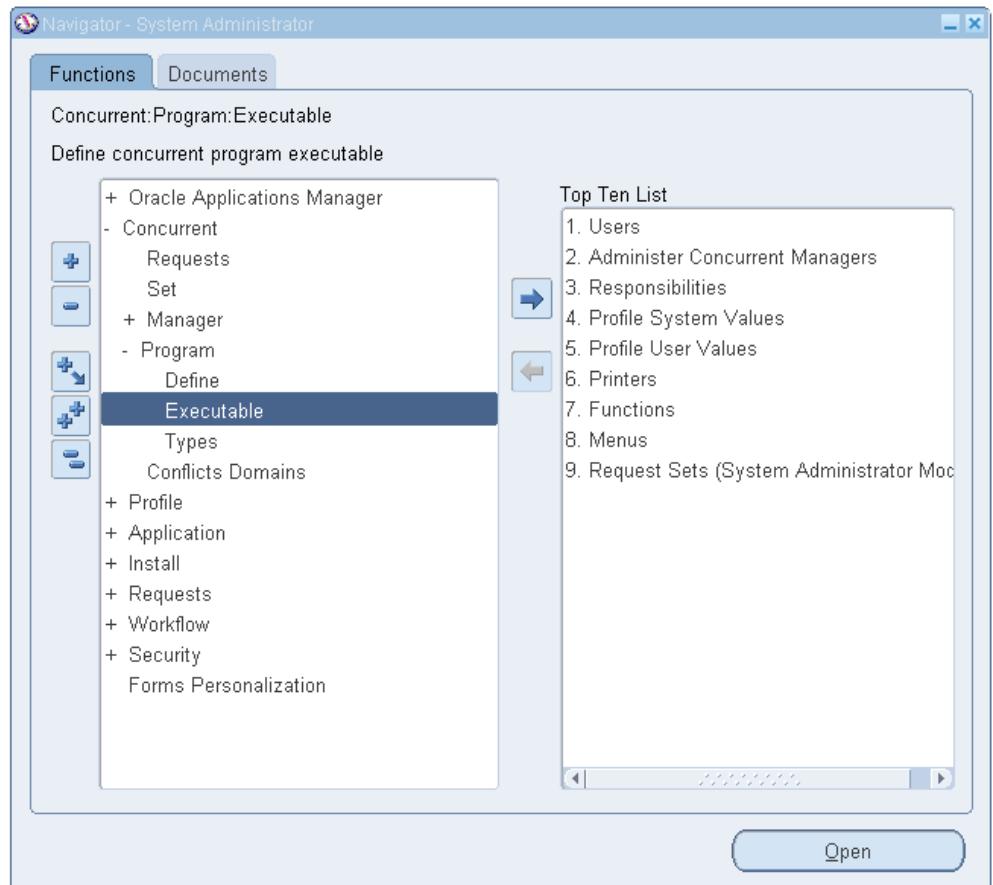
Display Size	50	Description Size	50
Concatenated Description Size	25	Prompt	Data File

Token: []

Registering Interface Concurrent Program

To register a concurrent program for '[XX<EMPID>: OM Sales Order Interface API Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



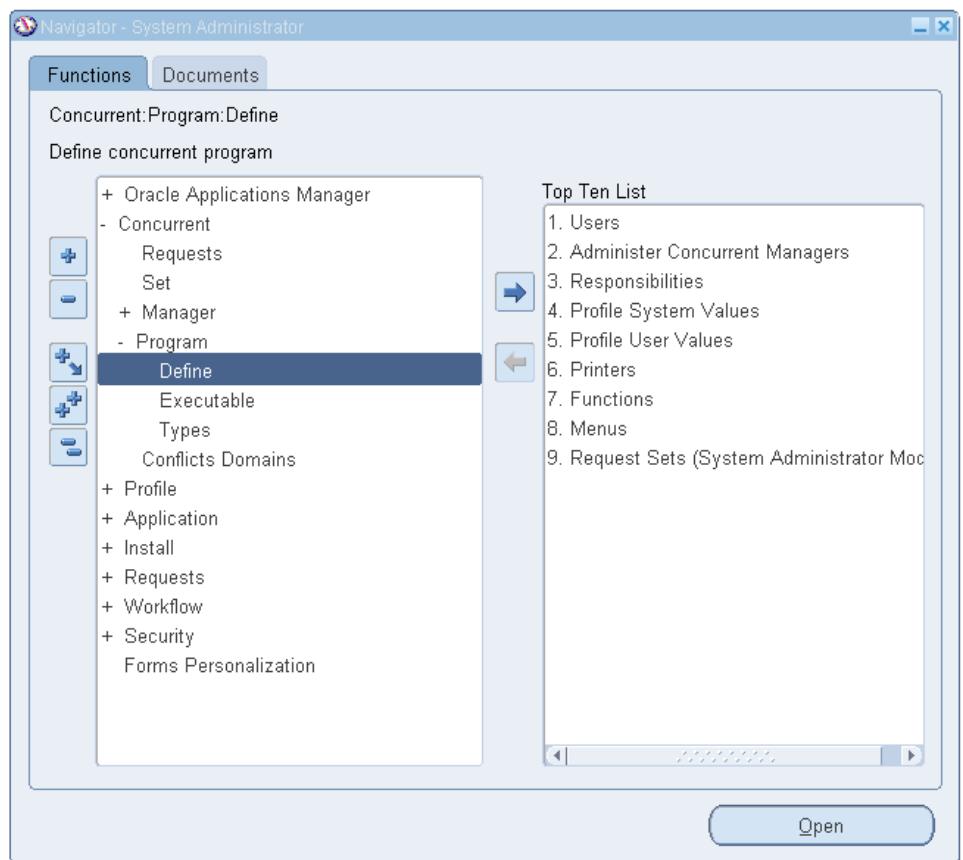
2. Enter the following in the Concurrent Program Executable Form

- Executable Name:** Enter a unique name for executable.
- Short Name:** Enter a unique short name, which will be used for defining Concurrent Program.
- Application:** Enter the Application Name from the List of Values (LOV). As we are developing an interface program for OM Sales Order, the Application Name should be **Custom Development**.
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be PL/SQL Stored Procedure.
- Execution File Name:** Here we pass the package name with one public procedure. As discussed in Module List the package name would be **XX_OM_ORDER_PKG** and the public procedure defined in this package is **main**.

Concurrent Program Executable Form

Executable	XXOMORDERAPIMAIN
Short Name	XXOMORDERAPIMAIN
Application	Custom Development
Description	XXTEST: OM Sales Order Interface API Program
Execution Method	PL/SQL Stored Procedure
Execution File Name	XX_OM_ORDER_API_PKG.main
Subroutine Name	
Execution File Path	
Stage Function Parameters	

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name:** Enter “XX<EMPID>: OM Sales Order Interface API Program”
- Short Name:** Enter a unique concurrent program short name, for example: **XXOMORDERAPIMAIN**
- Application:** Select the value from LOV. As we would be executing this concurrent program from Order Management Applications, select **Custom Development**.

- d. **Description:** Enter the description of this concurrent program.
- e. **Executable Name:** Select the executable name from LOV. As we have already registered an executable for loader program, select **XXOMORDERAPIMAIN** from LOV.
- f. **Executable Method:** Executable method will be populated automatically.
- g. **Parameters (if any):** Select the Parameters Button to specify the parameter list. In this case, two parameters are passed
 - i. Data File Path
 - ii. Data File Name

The screenshot shows the 'Concurrent Programs' window. The 'Program' field is set to 'XXTEST: OM Sales Order Interface API Program'. The 'Executable' section shows 'Name' as 'XXOMORDERAPIMAIN' and 'Method' as 'PL/SQL Stored Procedure'. The 'Request' section contains fields for 'Type', 'Incrementor', and 'MLS Function', along with checkboxes for 'Use in SRS', 'Run Alone', 'Enable Trace', 'Allow Disabled Values', 'Restart on System Failure', and 'NLS Compliant'. The 'Output' section specifies 'Format' as 'Text', with options for 'Save' and 'Print'. Buttons at the bottom include 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters'.

Concurrent Program Definition Form

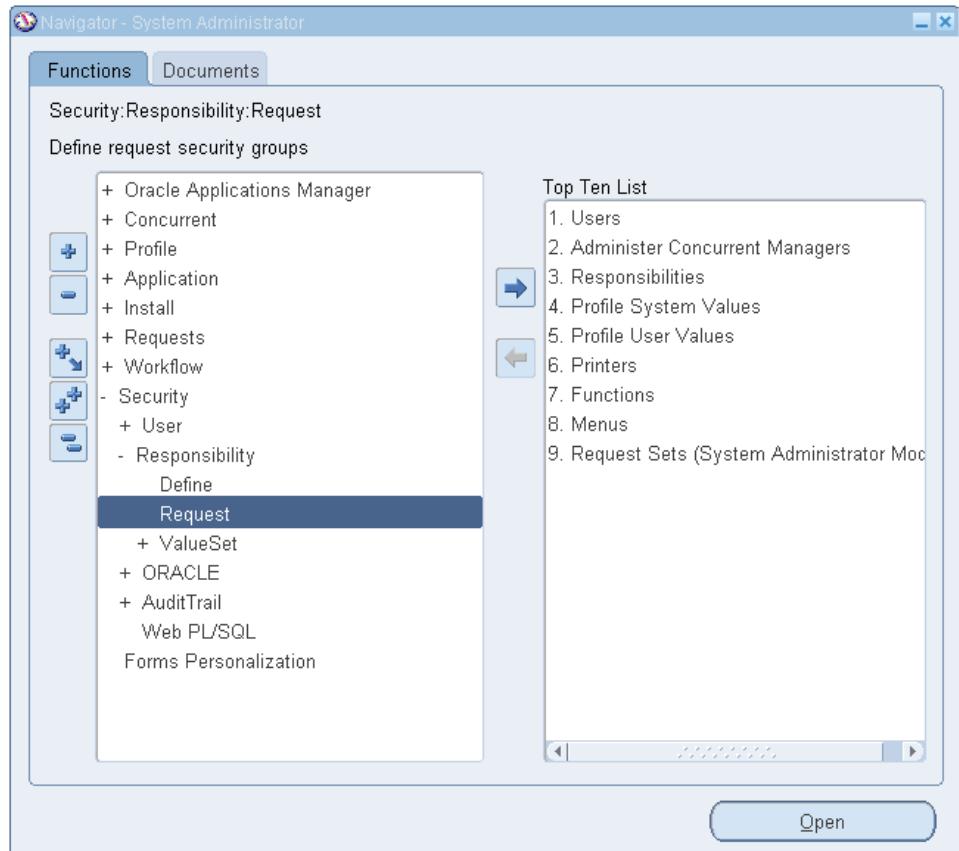
The screenshot shows the 'Concurrent Program Parameters' window. The 'Program' field is set to 'XXTEST: OM Sales Order Interface API Program'. The 'Parameters' table lists two parameters: 'p_data_file_path' (Seq 10) and 'p_data_file_name' (Seq 20). Both parameters are described as 'Data File Path' and 'Data File Name' respectively. The 'Validation' section shows a value set of '100 Characters' and a default value of '100 Characters'. The 'Display' section shows a display size of 50, a concatenated description size of 25, and a prompt of 'Data File Name'. A 'Token' field is also present.

Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: OM Sales Order Interface API Program’ in Request Group, Navigate to System Administrator Responsibility

1. Select Security -> Responsibility-> Request



2. Query the Request Group (OM Concurrent Programs) and Applications (Order Management)

The screenshot shows the Oracle Request Groups dialog box. At the top, there are four input fields: Group (OM_Concurrent_Programs), Application (Order Management), Code (OM_CONC_PROGRAMS), and Description (Order Management Concurrent Programs). Below these fields is a section titled "Requests" which contains a table. The table has three columns: Type, Name, and Application. The rows show various concurrent programs: Workflow Background Process (Application Object Library), Expense Report Import (Obsolete) (Payables), Payables Open Interface Import (Payables), Requisition Import (Purchasing), Requisition Import Exceptions Report (Purchasing), Create Internal Orders (Purchasing), Revenue Contingency Analyzer (Receivables), Defaulting Generator (Order Management), Order Import (Order Management), and Schedule Orders (Order Management). A note at the bottom states: "Description Workflow background process for deferred and timeout activities".

Request Group Form

3. Add an entry for the concurrent programs in the Request Group and save the work.

The screenshot shows the Oracle Request Groups dialog box. The Group, Application, Code, and Description fields are identical to the previous screenshot. In the Requests table, a new row has been added for "XXTEST: OM Sales Order Interface API Program". This row is highlighted with a yellow background. The table structure remains the same as in the previous screenshot, with columns for Type, Name, and Application. The note at the bottom is also present.

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and it can be further used to install in another Application Server.

Before downloading the script of Concurrent Program, connect to your local system to Applications Server

Run the following command to download the script

```
FNDLOAD <apps_user/apps_pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<concurrent_program_short_name>
```

For Example:

- To download the “XX<EMPID>: OM Sales Order Interface API Loader Program” script, enter the following command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXOMORDERAPILOAD.ldt PROGRAM
CONCURRENT_PROGRAM_NAME=XXOMORDERAPILOAD
```

- To download the “XX<EMPID>: OM Sales Order Interface API Program” script, enter the following command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXOMORDERAPIMAIN.ldt PROGRAM
CONCURRENT_PROGRAM_NAME=XXOMORDERAPIMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



- Save the attached zip file in local system
- Transfer the zip to the Oracle Applications Server
- Create a Staging Directory [Eg. XXTESTOMINTAPI] in temporary area on Oracle Application Server.

```
mkdir -p XXTESTOMINTAPI
```

- Change the Permissions on the temp directory.

```
chmod 755 XXTESTOMINTAPI
```

- FTP the **XXTESTOMINTAPI.tar.gz** file in BINARY mode to directory **XXTESTOMINTAPI**
- Change directory to XXTESTOMINTAPI as given below

```
cd XXTESTOMINTAPI
```

- Uncompress **XXTESTOMINTAPI.tar.gz** as given below

```
gunzip XXTESTOMINTAPI.tar.gz
```

- Untar the file **XXTESTOMINTAPI.tar** using

```
tar -xvf XXTESTOMINTAPI.tar.gz
```

- Grant the execute permission on the install script using

```
chmod 755 XX_OM_ORDER_API.install
```

- Run **XX_OM_ORDER_API.install**

```
sh XX_OM_ORDER_API.install
```

- Enter the apps schema user name password (apps/ <apps_pwd>) when asked for.

Extract File Layout

Please find the attached extract file for OM Sales Orders.



OM_Sales_Order_Extract_Data.csv

Note: Please remove the header row before putting the same file on Oracle Application Server for loading purpose.

Below is the example of extract file for OM Sales Orders

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	version	order_type	customer	price_list	transaction	payment_term	organization	location	booker	salesrep	line_nu	line_type	order_qu	ordered_qty	inventory_itm	shipment_subinvento		
2	1	Mixed	American Telephone & Teleg	Corporate	USD	30 NET	Vision Operations	New York (OPS N	Sell, Mr. Thomas		1	Standard (Line Invoicing)	Ea	100	KB10759	1	Stores	
3	1	Mixed	Peter Quance	Corporate	USD	N30	Vision Operations	6550 N	No Sales Credit		1	Standard (Line Invoicing)	Ea	435	MK-T007007	1		
4	1	Mixed	Hilman and Associates	Corporate	USD	30 NET	Vision Operations	Tulsa (OPS) N	Clary, Mr. Timothy		1	Standard (Line Invoicing)	Ea	18	XPR0003	1	FGI	
5	1	Mixed	Computer.Com	Corporate	USD	30 NET	Vision Operations	San Jose (OPS N	Sprague, Mr. Howard		1	Standard (Line Invoicing)	Ea	10	A554888	1	FGI	
6	1	Mixed	Janet MacMullen	Corporate	EUR	2/10, Net 30	Vision Operations	8228 N	Taylor, Mr. Phillip Charles		1	Standard (Line Invoicing)	Ea	1	MK-T007007	1	FGI	
7																		
8																		
9																		
10																		
11																		
12																		
13																		
14																		
15																		
16																		
17																		
18																		
19																		
20																		
21																		
22																		
23																		
24																		
25																		
26																		
27																		
28																		
29																		
30																		
31																		
32																		
33																		
34																		
35																		
36																		
37																		
38																		
39																		
40																		
41																		
42																		
43																		
44																		

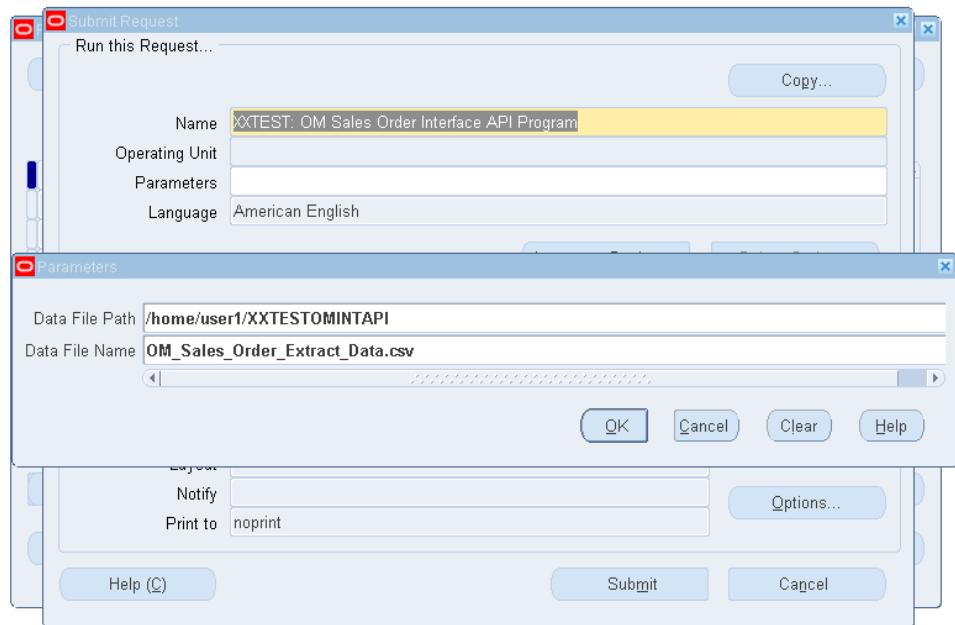
Extract File Mapping to Staging Table Columns

Data File Columns	Datatype	Size	Staging Table's Columns
version_number	NUMBER		version_number
order_type	VARCHAR2	40	order_type
ship_to_customer	VARCHAR2	360	ship_to_customer
price_list	VARCHAR2	240	price_list
transactional_curr_code	VARCHAR2	15	transactional_curr_code
payment_term	VARCHAR2	30	payment_term
organization	VARCHAR2	240	organization
location	VARCHAR2	240	location
booked_flag	VARCHAR2	1	booked_flag
salesrep	VARCHAR2	240	salesrep
line_number	NUMBER		line_number
line_type	VARCHAR2	30	line_type
order_quantity_uom	VARCHAR2	3	order_quantity_uom
ordered_quantity	NUMBER		ordered_quantity
inventory_item	VARCHAR2	2000	inventory_item
shipment_number	NUMBER		shipment_number
subinventory	VARCHAR2	10	subinventory

Calling Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “XX<EMPID>: OM Sales Order Interface API Program”. To run the concurrent program navigate to [Order Management Responsibility](#).

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



SRS Form

1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Query Interface Records

When custom concurrent program “[XX<EMPID>: OM Sales Order Interface API Program](#)” completes successfully, navigate to the **Order Management Responsibility**, select **Orders -> Returns -> Sales Orders** and query the records of interface table.

Customer	American Telephone & Tele	Order Number	66171
Customer Number	1001	Order Type	Mixed
Customer PO		Date Ordered	04-MAR-2009 01:31:37
Customer Contact		Price List	Corporate
Blanket Number		Salesperson	Sell, Thomas
Ship To Location	New York (OPS)	Status	Entered
	32 Ave of the Americas	Currency	USD
	New York, NY, 10013, US	Subtotal	2,849.00
Bill To Location	New York (OPS)	Tax	242.16
	32 Ave of the Americas	Charges	0.00
	New York, NY, 10013, US	Total	3,091.16

OM Sales Order Form

Error Handling

Following errors can occur while processing the OM Sales Order data file:

1. During loading data into staging table.

- When error occurs while loading the data file using SQL*Loader concurrent program, it generates two files.
 - **SQL*Loader Bad File:** The bad file contains records that were not loaded into the staging table. These records could have been rejected by SQL*Loader due to an invalid format. They could also have been rejected by the Oracle database if they violate an integrity constraint or had an invalid data type for the staging table.
 - **SQL*Loader Log File:** Detailed information about the load is stored in the log file. The errors found during parsing of the control file are stored in the log file. The log file also identifies the number of records successfully loaded. The log file must be available during the entire run of the SQL*Loader. When loading data with SQL*Loader, nothing should be assumed without reviewing the log files

2. Validating the records in staging table

- While validating the data in staging table, it may result in error and the concurrent program could complete in warning. In such cases, look into the output report of concurrent program and correct the data file and re-load it again.

3. While calling the Standard API

- While calling standard API it may complete with error and it returns the error message in OE_UPGRADE_ERRORS table. As the API error message is displayed in output report of custom concurrent program, verify the report and do the correction in data. After correcting the data file, re-load it and run the custom current program again.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Migration of OM Sales Order using Open Interface

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Data Dependencies.....	3
Business Rules	3
Approach.....	4
Module List.....	5
Registering Concurrent Program	9
Downloading Concurrent Program Script	18
Deployment.....	19
Extract File Layout	20
Extract File Mapping to Staging Table Columns	21
Calling Custom Concurrent Program.....	22
Calling Standard Import Concurrent Program.....	23
Query Interface Records	24
Error Handling	25
Open and Closed Issues for this Deliverable	26
Open Issues	26
Closed Issues.....	26

Introduction

A sales order is an internal document of the company, which should record the customer's originating purchase order.

You can enter, view, and update sales orders using the Sales Orders window. You can order standard items, both shippable and non-shippable and configurations using this window.

To enter or query a Sales Order from front end, navigate to Order Management Responsibility and select **Orders -> Returns -> Sales Orders**.

This document demonstrates the use of open interface table to migrate OM Sales Order from source system to Oracle Applications.

Purpose

This document describes the:

- Detailed data mapping from the source system(s) to the Oracle Applications E-Business Suite (EBS) Order Management (OM) Sales Order records
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during implementation of Oracle Order Management.

Scope and Application

The following boundaries are specific to the interface of OM Sales Order using OE_HEADERS_IFACE_ALL and OE_LINES_IFACE_ALL open interface tables

- All Sales Orders and Returns would be migrated to Oracle Applications
- The '[XX<EMPID>: OM Sales Order Interface Program](#)' uploads OM Sales Order data from data file into staging table "XXOM_ORDER_IFACE_STG," performs validation and transfers the valid records into Oracle Order Management interface tables. The standard import program 'Order Import' is then run to perform the validation on the data in the interface tables and load the valid data into base tables.

1. The OM Sales Order data will be provided through data file. This data file will be loaded into staging table(s) on the Oracle applications database using loader program.
2. The loaded data will be validated from staging tables and then moved into the standard sales order interface tables.
3. The standard Order Import program will be called to import the sales orders from interface tables to Oracle Order Management base tables.

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.

Source system

The implementation team studies the source system and understands the data required to be migrated to Oracle Applications. This data is then extracted and cleansed from the source system (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team. The data mapping section describes the data format of source system to Oracle Applications.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Order Management Super User
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.
- Users have access to Oracle Applications database (apps schema)
- All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.
- The extract-file layout must be in the format as specified in the extract file layout section

Pre-Requisites

Prerequisite set-ups are required for the interface. These setups include, but are not limited to:

1. Transaction Types
2. Customers
3. Price Lists
4. Items assigned to Price Lists
5. Discounts assigned to Price Lists
6. Salespersons
7. Tax Codes and Tax Rates
8. Invoicing and Accounting Rules
9. Freight Carriers, Freight Terms, and FOB options
10. Shipping Warehouses

Data Dependencies

OM Sales Order

In order to migrate Sales Orders, the following entities must have been migrated successfully:

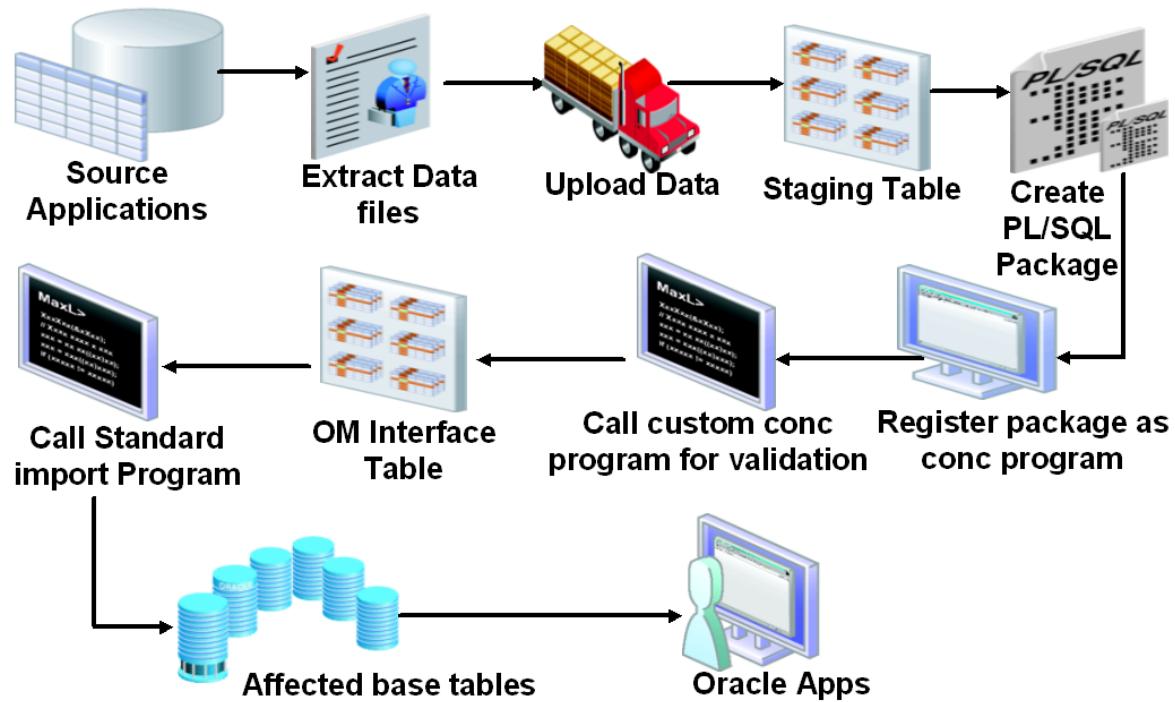
- Customers
- Price Lists
- Items

Business Rules

Following business rules are applicable for the OM Sales Order conversion process.

- N/A

Approach



The Approach for migrating OM Sales Order data from source system to Order Management open interface table

1. Extract the data file from source system
2. Load the data file into staging table
3. Using PL/SQL write a package
4. Register the PL/SQL package as concurrent program in Oracle Applications
5. Call the custom concurrent program to validate and load the records available in staging table into open interface table
6. Call the standard import program, which creates the records in OE_ORDER_HEADERS_ALL and OE_ORDER_LINES_ALL tables based on the open interface records.
7. Query the records that are created from interface table and see the result in Oracle Applications front end.

Module List

Concurrent Programs

‘[XX<EMPID>: OM Sales Order Interface Program](#)’ includes the following concurrent programs:

‘[XX<EMPID>: OM Sales Order Interface Program](#)’

The ‘[XX<EMPID>: OM Sales Order Interface Program](#)’ concurrent program is based on the stored procedure **XX_OM_ORDER_PKG.main** that is registered as a PL/SQL executable.

- The above program calls the loader program ([XX<EMPID>: OM Sales Order Interface Loader Program](#)) which loads the data into staging tables, performs validations on the staging table data and uploads data into the open interface tables (OE_HEADERS_IFACE_ALL and OE_LINES_IFACE_ALL).

‘[XX<EMPID>: OM Sales Order Interface Loader Program](#)’

The ‘[XX<EMPID>: OM Sales Order Interface Loader Program](#)’ concurrent program is based on the SQL*Loader control file, that is registered as a Loader executable.

- The above program loads data from the path specified as parameter value in loader concurrent program to the staging table.

NOTE:

1. Change <EMPID> with your Employee ID,
2. Follow the steps provided in ‘Registering Concurrent Program’ section for registering the concurrent program.

Stored Procedures

[XX_OM_ORDER_PKG](#) package consist the following stored procedures:

[XX_OM_ORDER_PKG.main](#)

This is the main procedure that calls the load, validate, import, print_error and record_summary procedures.

[XX_OM_ORDER_PKG.validate](#)

This procedure performs the necessary validations on the staging tables.

[**XX_OM_ORDER_PKG.import**](#)

This procedure loads all the valid data from the staging table to Oracle Order Management interface tables OE_HEADERS_IFACE_ALL and OE_LINES_IFACE_ALL

[**XX_OM_ORDER_PKG.print_error**](#)

This procedure reports the errored records in the log file.

[**XX_OM_ORDER_PKG.record_summary**](#)

This procedure displays the status of the staging table records

NOTE: Please refer the attached documents for Package specifications and Package body. Kindly open the attached script in notepad.



1. Package Specification:



2. Package Body:

Staging Table

XXOM_ORDER_IFACE_STG table is created in the database to store records from source system data file.

NOTE: Please refer to the attached script for staging table creation. Kindly open the attached script in notepad.



Table Creation Script:

Grant Script

If the table and package are created in custom schema, then grant the table and package to APPS schema

NOTE: Please refer to the attached script for table and package granting. Kindly open the attached script in notepad.



XX_OM_ORDER_PKG
.grt

1. Package Grant Script:



XX_OM_ORDER_TAB
.grt

2. Table Grant Script:

Synonym Script

If table and package are created in custom schema, then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



XX_OM_ORDER_PKG
.syn

1. Package synonym script:



XX_OM_ORDER_TAB
.syn

2. Table synonym script:

SQL*Loader Script

SQL*Loader is controlled by its own data definition language, which is kept in the control file. The control file describes the data to be loaded, the destination tables of the data and describes the interdependency between the data and the columns within the tables.

NOTE: Please refer the attached SQL* Loader script for loading data file into staging table. Kindly open the attached script in notepad.



XX_OM_ORDER_LOA
D.ctl

SQL* Loader Script:

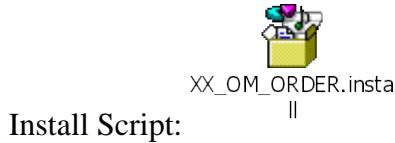
Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script
3. Staging Table Script
4. SQL*Loader Script

NOTE: Please refer the attached Install script for deployment of OM Sales Orders object on Oracle Applications Server. Kindly open the attached script in notepad.



Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

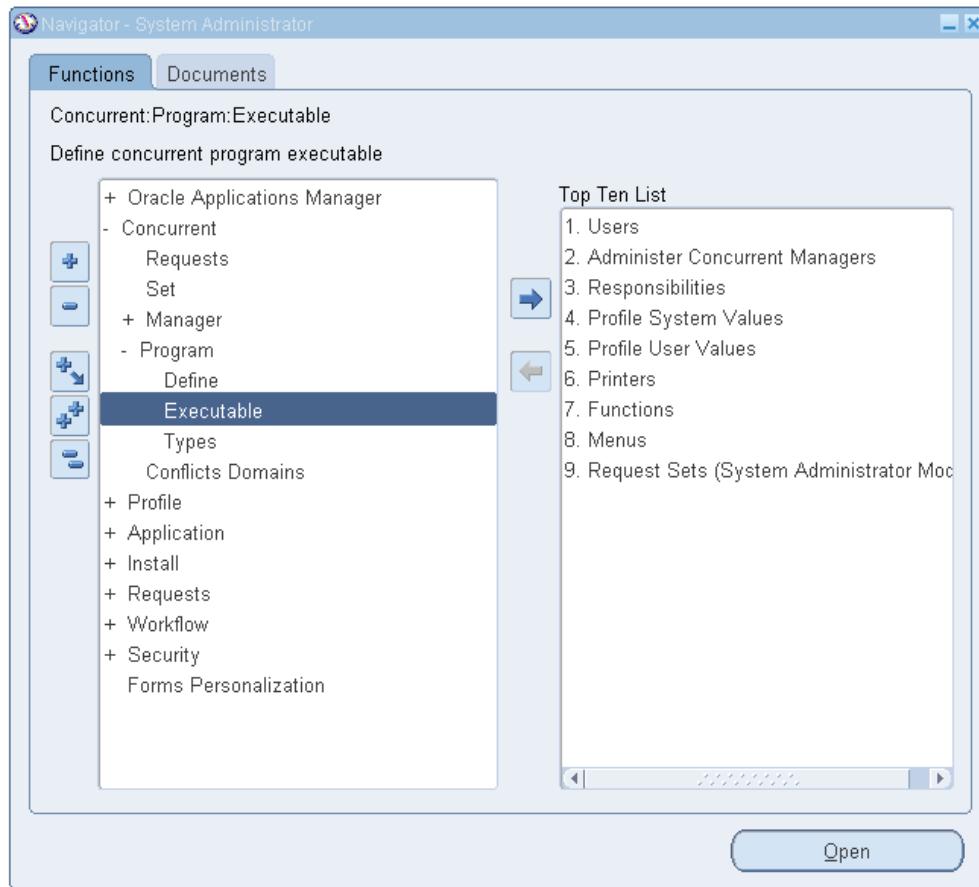
- A. **Package Name**
- B. **Table Name**
- C. **Synonym Name**
- D. **Concurrent Program Executable**
- E. **Concurrent Program Name**

Registering Concurrent Program

Registering Interface Loader Concurrent Program

To register a concurrent program for '[XX<EMPID>: OM Sales Order Interface Loader Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



2. Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter a unique name for executable.
- Short Name:** Enter a unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from the List of Values (LOV). As we are developing an interface program for OM Sales Orders, the Application Name should be [Custom Development](#).
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be SQL*Loader, because we are registering a concurrent program for loading the data to staging table.

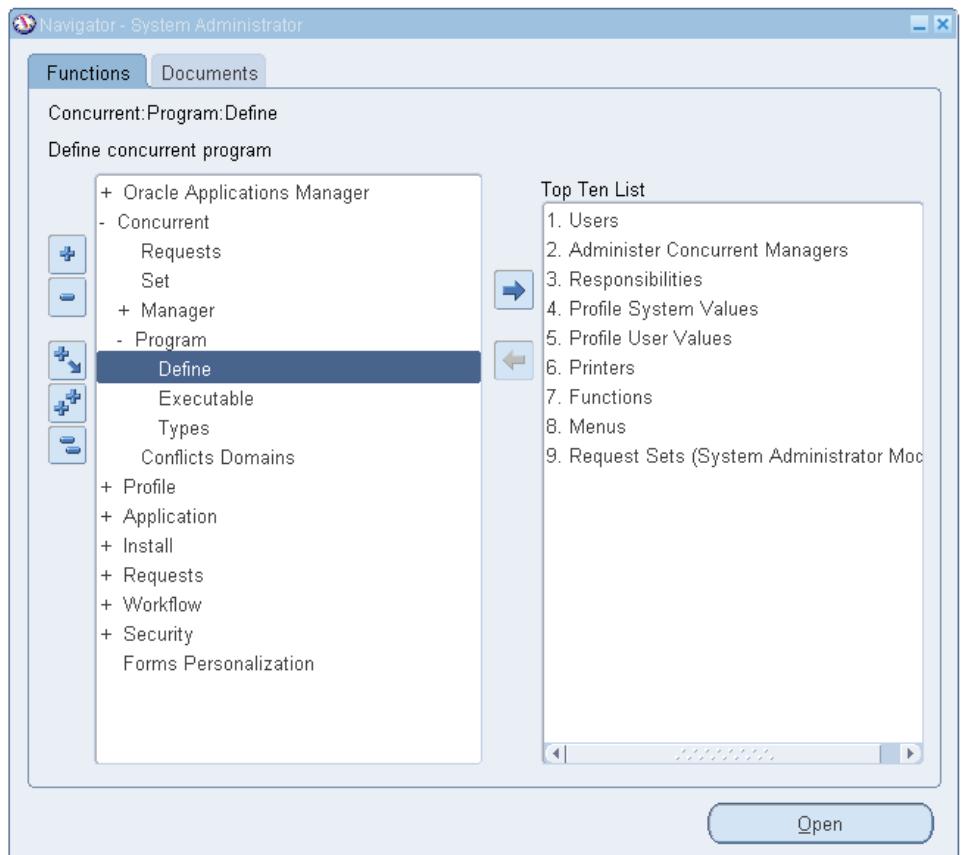
- f. **Execution File Name:** Here we would be passing the file name of SQL*Loader control file. As discussed in Module List the control file name would be **XX_OM_ORDER_LOAD**

Executable	XXOMORDERLOAD
Short Name	XXOMORDERLOAD
Application	Custom Development
Description	XXTEST: OM Sales Order Interface Loader Program
Execution Method	SQL*Loader
Execution File Name	XX_OM_ORDER_LOAD
Subroutine Name	
Execution File Path	

Stage Function Parameters

Concurrent Program Executable Form

3. Select Concurrent -> Program -> Define



4. Enter the following in the Concurrent Program Form

- a. **Program Name:** Enter “XX<EMPID>: OM Sales Order Interface Loader Program”

- b. **Short Name:** Enter a unique concurrent program short name, for example: **XXOMORDERLOAD**
- c. **Application:** Select the value from LOV. As we would be executing this concurrent program from Order Management Applications so select **Custom Development**.
- d. **Description:** Enter the description of this concurrent program.
- e. **Executable Name:** Select the executable name from LOV. As we have already registered an executable for the loader program, select **XXOMORDERLOAD** from LOV.
- f. **Executable Method:** Executable method will be populated automatically.
- g. **Parameters (if any):** Select the Parameters Button to specify the parameter list. In this case, two values should be passed in a single parameter - Data File Name and Data File Path

Concurrent Programs

Program	XXTEST: OM Sales Order Interface Loader Program	<input checked="" type="checkbox"/> Enabled
Short Name	XXOMORDERLOAD	
Application	Custom Development	
Description	XXTEST: OM Sales Order Interface Loader Program	
Executable		
Name	XXOMORDERLOAD	Options
Method	SQL*Loader	Priority
Request		
Type		
Incrementor		
MLS Function		
<input checked="" type="checkbox"/> Use in SRS <input type="checkbox"/> Run Alone <input type="checkbox"/> Enable Trace		<input type="checkbox"/> Allow Disabled Values <input checked="" type="checkbox"/> Restart on System Failure <input checked="" type="checkbox"/> NLS Compliant
Output Format: Text <input checked="" type="checkbox"/> Save (C) <input checked="" type="checkbox"/> Print Columns: Rows: Style: <input type="checkbox"/> Style Required Printer:		
<input type="button" value="Copy to..."/> <input type="button" value="Session Control"/> <input type="button" value="Incompatibilities"/> <input type="button" value="Parameters"/>		

Concurrent Program Definition Form

Concurrent Program Parameters Form

Seq	Parameter	Description	Enabled
10	p_data_file	Data File name with file path	<input checked="" type="checkbox"/>

Validation

Value Set	100 Characters	Description	100 Characters
Default Type		Default Value	
<input type="checkbox"/> Required	<input type="checkbox"/> Enable Security	Range	

Display

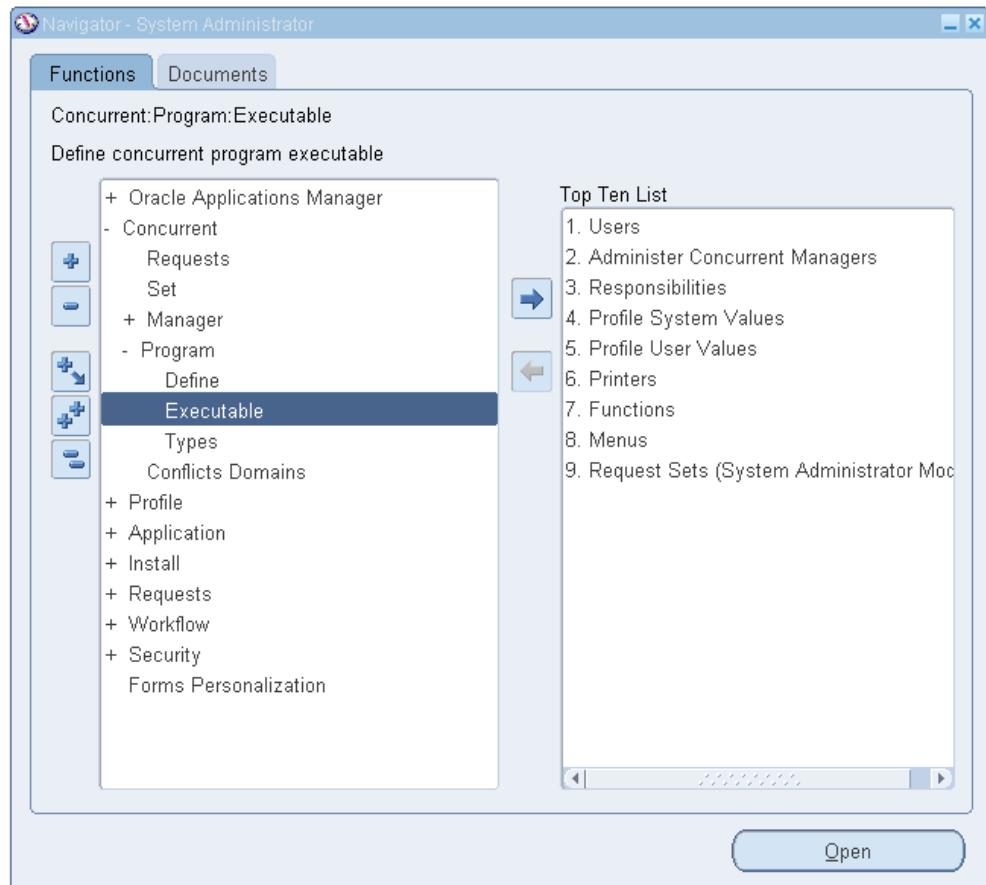
Display Size	50	Description Size	50
Concatenated Description Size	25	Prompt	Data File

Token:

Registering Interface Concurrent Program

To register a concurrent program for '[XX<EMPID>: OM Sales Order Interface Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



2. Enter the following in the Concurrent Program Executable Form

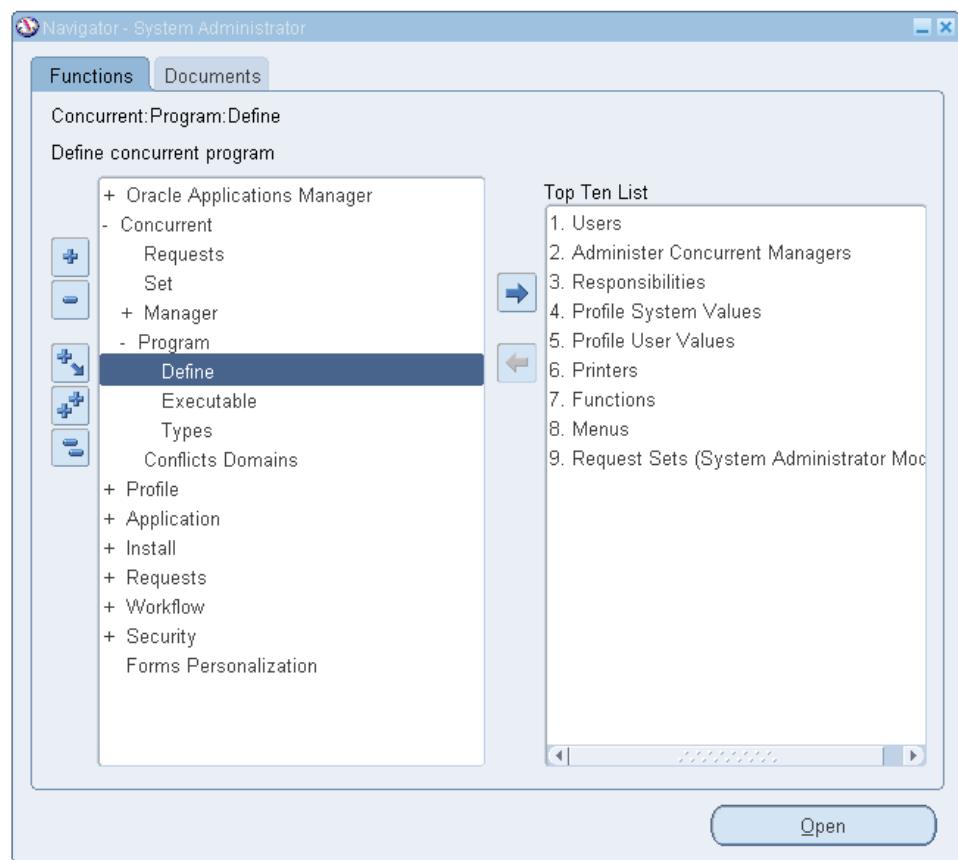
- Executable Name:** Enter a unique name for executable.
- Short Name:** Enter a unique short name, which will be used for defining Concurrent Program.
- Application:** Enter the Application Name from the List of Values (LOV). As we are developing an interface program for OM Sales Order, the Application Name should be **Custom Development**.
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be PL/SQL Stored Procedure.
- Execution File Name:** Here we pass the package name with one public procedure. As discussed in Module List the package name would be **XX_OM_ORDER_PKG** and the public procedure defined in this package is **main**.

Concurrent Program Executable Form

Executable	XXOMORDERMAIN
Short Name	XXOMORDERMAIN
Application	Custom Development
Description	XXTEST: OM Sales Order Interface Program
Execution Method	PL/SQL Stored Procedure
Execution File Name	XX_OM_ORDER_PKG.main
Subroutine Name	
Execution File Path	

Stage Function Parameters

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name:** Enter “XX<EMPID>: OM Sales Order Interface Program”
- Short Name:** Enter a unique concurrent program short name, for example: **XXOMORDERMAIN**
- Application:** Select the value from LOV. As we would be executing this concurrent program from Order Management Applications, select **Custom Development**.

- d. **Description:** Enter the description of this concurrent program.
- e. **Executable Name:** Select the executable name from LOV. As we have already registered an executable for loader program, select **XXOMORDERMAIN** from LOV.
- f. **Executable Method:** Executable method will be populated automatically.
- g. **Parameters (if any):** Select the Parameters Button to specify the parameter list. In this case, two parameters are passed
 - i. Data File Path
 - ii. Data File Name

The screenshot shows the Oracle Concurrent Programs window. The 'Program' field is set to 'XXTEST: OM Sales Order Interface Program'. The 'Short Name' is 'XXOMORDERMAIN', 'Application' is 'Custom Development', and 'Description' is 'XXTEST: OM Sales Order Interface Program'. Under the 'Executable' section, 'Name' is 'XXOMORDERMAIN' and 'Method' is 'PL/SQL Stored Procedure'. The 'Request' section includes fields for 'Type', 'Incrementor', and 'MLS Function'. The 'Output' section specifies 'Format' as 'Text', with options for 'Save' and 'Print'. Buttons at the bottom include 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters'.

Concurrent Program Definition Form

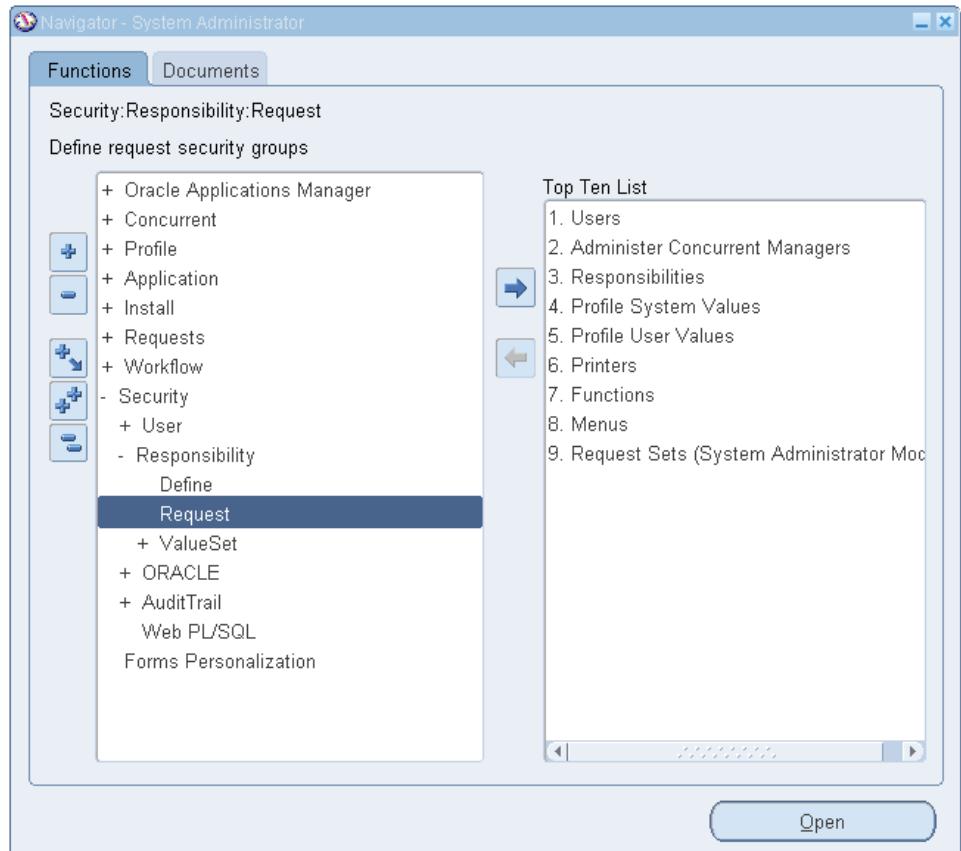
The screenshot shows the Concurrent Program Parameters window. The 'Program' is 'XXTEST: OM Sales Order Interface Program' and 'Application' is 'Custom Development'. The 'Conflicts Domain' and 'Security Group' fields are empty. The 'Parameter' section lists two parameters: 'p_data_file_path' (Seq 10) with description 'Data File Path' and 'Enabled' checked; and 'p_data_file_name' (Seq 20) with description 'Data File Name' and 'Enabled' checked. The 'Validation' section includes a 'Value Set' of '100 Characters', 'Default Type' (empty), and checkboxes for 'Required' and 'Enable Security'. The 'Display' section shows 'Display Size' as 50, 'Concatenated Description Size' as 25, 'Description Size' as 50, and 'Prompt' as 'Data File Name'. A 'Token' field is also present.

C Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: OM Sales Order Interface Program’ in Request Group, Navigate to [System Administrator Responsibility](#)

1. Select Security -> Responsibility-> Request



2. Query the Request Group (OM Concurrent Programs) and Applications (Order Management)

The screenshot shows the 'Request Groups' window with the following details:

- Group:** OM Concurrent Programs
- Application:** Order Management
- Code:** OM_CONC_PROGRAMS
- Description:** Order Management Concurrent Programs

Requests

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	Expense Report Import (Obsolete)	Payables
Program	Payables Open Interface Import	Payables
Program	Requisition Import	Purchasing
Program	Requisition Import Exceptions Report	Purchasing
Program	Create Internal Orders	Purchasing
Program	Revenue Contingency Analyzer	Receivables
Program	Defaulting Generator	Order Management
Program	Order Import	Order Management
Program	Schedule Orders	Order Management

Description: Workflow background process for deferred and timeout activities

Request Group Form

3. Add an entry for the concurrent programs in the Request Group and save the work.

The screenshot shows the 'Request Groups' window with the following details:

- Group:** OM Concurrent Programs
- Application:** Order Management
- Code:** OM_CONC_PROGRAMS
- Description:** Order Management Concurrent Programs

Requests

Type	Name	Application
Program	Workflow Background Process	Application Object Library
Program	Expense Report Import (Obsolete)	Payables
Program	XXTEST: OM Sales Order Interface Loader	Custom Development
Program	XXTEST: OM Sales Order Interface Program	Custom Development
Program	Payables Open Interface Import	Payables
Program	Requisition Import	Purchasing
Program	Requisition Import Exceptions Report	Purchasing
Program	Create Internal Orders	Purchasing
Program	Revenue Contingency Analyzer	Receivables
Program	Defaulting Generator	Order Management

Description: XXTEST: OM Sales Order Interface Program

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and it can be further used to install in another Application Server.

Before downloading the script of Concurrent Program, connect to your local system to Applications Server

Run the Following command to download the script

```
FNDLOAD <apps_user/apps_pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<concurrent_program_short_name>
```

For Example:

- To download the “XX<EMPID>: OM Sales Order Interface Loader Program” script, enter the following command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct XXOMORDERLOAD.ldt
PROGRAM
CONCURRENT_PROGRAM_NAME=XXOMORDERLOAD
```

- To download the “XX<EMPID>: OM Sales Order Interface Program” script, enter the following command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct XXOMORDERMAIN.ldt
PROGRAM
CONCURRENT_PROGRAM_NAME=XXOMORDERMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



- Save the attached zip file in local system
- Transfer the zip to the Oracle Applications Server
- Create a Staging Directory [Eg. XXTESTOMINTOIT] in temporary area on Oracle Application Server.

```
mkdir -p XXTESTOMINTOIT
```

- Change the Permissions on the temp directory.

```
chmod 755 XXTESTOMINTOIT
```

- FTP the **XXTESTARINTOIT.tar.gz** file in BINARY mode to directory **XXTESTOMINTOIT**
- Change directory to XXTESTARINTOIT as given below

```
cd XXTESTOMINTOIT
```

- Uncompress **XXTESTARINTOIT.tar.gz** as given below

```
gunzip XXTESTOMINTOIT.tar.gz
```

- Untar the file **XXTESTOMINTOIT.tar** using

```
tar -xvf XXTESTOMINTOIT.tar.gz
```

- Grant the execute permission on the install script using

```
chmod 755 XX_OM_ORDER.install
```

- Run **XX_OM_ORDER.install**

```
sh XX_OM_ORDER.install
```

- Enter the apps schema user name password (apps/ <apps_pwd>), when asked for.

Extract File Layout

Please find the attached extract file for OM Sales Orders.



OM_Sales_Order_Extract_Data.csv

Note: Please remove the header row before putting the same file on Oracle Application Server for loading purpose.

Below is the example of extract file for OM Sales Orders

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	
1	version_num	order_type	customer	price_list	transactional_payment	ter_organization	location	booked_flag	salesrep	line_number	line_type	order_qty	ordered_qty	in_inventory	itshipment	isubinventory
2	1	Mixed	American Tel Corporate	USD	30 NET	Vision Operat New York	(OFN)	Sell, Mr. Thor	1 Standard (Line)			100	KB18759			1 Stores
3	1	SE Web	Computer Ser Swedish	SEK	30 NET	Vision Swede	Göteborg	N	Stig Linder	1 SE Standard	Ea		1 AS54000			1 A1
4	1	DE RETURN	Computer Cor EURO	EUR	2/10, Net 30	Vision Germn	Bremen	N	Anne Golting	1 DE Retourn	Ea	395	620000			1
5	1	NL Web	Computer Cor EURO	EUR	30 NET	Vision Nether	2200	N	Jan Bruinsma	1 NL Standard	Ea	18	AS52688			1 FG1
6	1	UK Web	Worldwide Co United Kingd	GBP	30 NET	Vision Industr	Birmingham	N	Agate, Mr. Le	1 UK Standard	Ea	10	CM94532			1 FG1
7																
8																
9																
10																
11																
12																
13																
14																
15																
16																
17																
18																
19																
20																
21																
22																
23																
24																
25																
26																
27																
28																
29																
30																
31																
32																
33																
34																
35																
36																
37																
38																
39																
40																
41																
42																
43																
44																

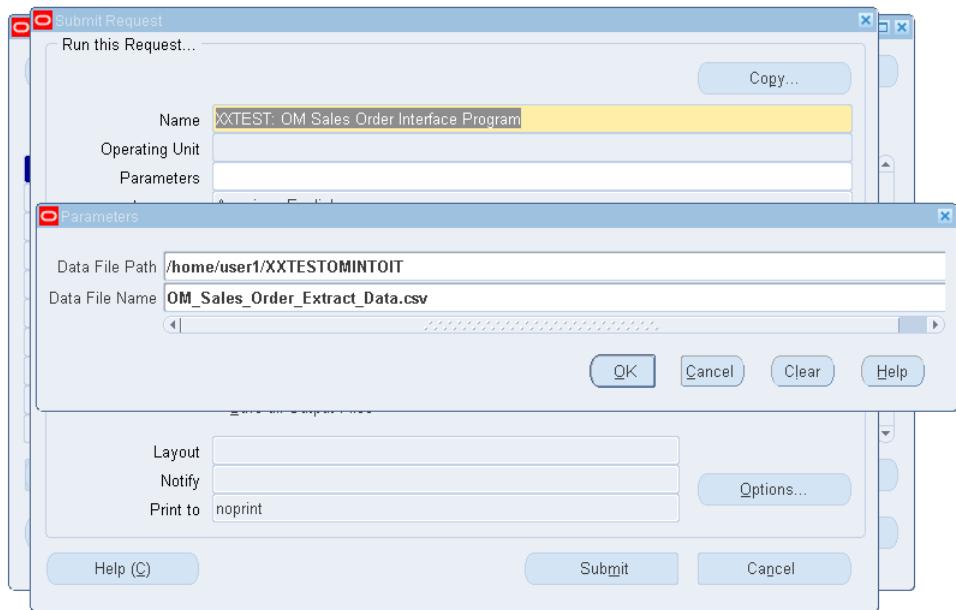
Extract File Mapping to Staging Table Columns

Data File Columns	Datatype	Size	Staging Table's Columns
version_number	NUMBER		version_number
order_type	VARCHAR2	40	order_type
ship_to_customer	VARCHAR2	360	ship_to_customer
price_list	VARCHAR2	240	price_list
transactional_curr_code	VARCHAR2	15	transactional_curr_code
payment_term	VARCHAR2	30	payment_term
organization	VARCHAR2	240	organization
location	VARCHAR2	240	location
booked_flag	VARCHAR2	1	booked_flag
salesrep	VARCHAR2	240	salesrep
line_number	NUMBER		line_number
line_type	VARCHAR2	30	line_type
order_quantity_uom	VARCHAR2	3	order_quantity_uom
ordered_quantity	NUMBER		ordered_quantity
inventory_item	VARCHAR2	2000	inventory_item
shipment_number	NUMBER		shipment_number
subinventory	VARCHAR2	10	subinventory

Calling Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “XX<EMPID>: OM Sales Order Interface Program”. To run the concurrent program navigate to [Order Management Responsibility](#).

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



SRS Form

1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Calling Standard Import Concurrent Program

When “[XX<EMPID>: OM Sales Order Interface Program](#)” completes successfully, run the standard import program to populate the data from OM Sales Order Interface Tables (OE_HEADERS_IFACE_ALL and OE_LINES_IFACE_ALL) to Order Management base tables.

The standard Import “[Order Import](#)” program will fetch the data from interface table and populate it into the Order Management base tables.

Call the “[Order Import](#)” Program with the following Parameters

- a. **Validate Only:** No
- b. **Validate Descriptive Flexfield:** Yes

Query Interface Records

When “[Order Import](#)” standard import program completes successfully, navigates to the **Order Management Responsibility**, selects **Orders -> Returns -> Sales Orders** and query the records of interface table.

Customer	American Telephone & Tele	Order Number	66171
Customer Number	1001	Order Type	Mixed
Customer PO		Date Ordered	04-MAR-2009 01:31:37
Customer Contact		Price List	Corporate
Blanket Number		Salesperson	Sell, Thomas
Ship To Location	New York (OPS) 32 Ave of the Americas New York, NY, 10013, US	Status	Entered
Bill To Location	New York (OPS) 32 Ave of the Americas New York, NY, 10013, US	Currency	USD
		Subtotal	2,849.00
		Tax	242.16
		Charges	0.00
		Total	3,091.16

OM Sales Order Form

Error Handling

Following errors can occur while processing the OM Sales Order data file:

1. During loading data into staging table.

- When error occurs while loading the data file using SQL*Loader concurrent program, it generates two files.
 - **SQL*Loader Bad File:** The bad file contains records that were not loaded into the staging table. These records could have been rejected by SQL*Loader due to an invalid format. They could also have been rejected by the Oracle database if they violate an integrity constraint or had an invalid data type for the staging table.
 - **SQL*Loader Log File:** Detailed information about the load is stored in the log file. The errors found during parsing of the control file are stored in the log file. The log file also identifies the number of records successfully loaded. The log file must be available during the entire run of the SQL*Loader. When loading data with SQL*Loader, nothing should be assumed without reviewing the log files

2. Validating the records in staging table

- While validating the data in staging table, it may result in error and the concurrent program could complete in warning. In such cases, look into the output report of concurrent program and correct the data file and re-load it again.

3. While calling the Standard Order Import Program

- After populating the records in interface tables, they are validated once more by the standard import program and in few cases it completes with error or warning. In such cases, see the output report of Order Import Program and correct the data in data file. Then re-load the data file and re-run the custom concurrent program.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Migration of Purchase Order data using Open Interface

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Data Dependencies.....	3
Business Rules	4
Approach.....	5
Module List.....	6
Registering Concurrent Program	10
Downloading Concurrent Program Script	19
Deployment.....	20
Extract File Layout	21
Extract File Mapping to Staging Table Columns	22
Calling Custom Concurrent Program.....	23
Calling Standard Import Concurrent Program.....	24
Query interface records.....	25
Error Handling	26
Open and Closed Issues for this Deliverable	27
Open Issues	27
Closed Issues.....	27

Introduction

A purchase order (PO) is a commercial document issued by a buyer to a supplier, indicating the type, quantities and agreed prices for products or services the seller will provide to the buyer. It will specify payment terms, delivery dates, item identification, quantities, shipping terms and all other obligations and conditions.

Purchase orders are generally preprinted, numbered documents generated by the retailer's financial management system, which shows that purchase details have been recorded and payment will be made.

Use the Purchase Order window to enter the standard purchase order, blanket purchase order, planed purchase order, purchase requisitions etc. You can also query and update purchase order in this window.

To enter or query a **Purchase Order** from front end navigate to Purchasing Responsibility, Select **Purchase Orders -> Purchase Orders**.

This document demonstrates the use of open interface table for PO to migrate **Purchase Order** data from source system to Oracle Applications.

Purpose

This document describes the:

- This case study is intended to demonstrate the creation of Standard Invoices Using Oracle Payables Interface table.
- Detailed data mapping from the source system(s) to the Oracle Applications E-Business Suite (EBS) **Purchase Order (PO)** Transactions records
- File layout to be used for interface.
- Approach and technical design for the interface

Background

This case study document is designed to demonstrate a practical scenario that occurs during implementation of oracle Purchase Orders.

Scope and Application

The following boundaries are specific to the Interfaces of **Purchase Order** using **PO_HEADERS_INTERFACE** and **PO_LINES_INTERFACE** open interface table

- All standard purchase orders would be migrated to Oracle Applications
- The ‘XX<EMPID>: Purchase Orders Interface Program’ uploads Purchase Order data from data file into staging table “XXPO_PURCHASE_ORDER_IFACE_STG” and will perform validation and transfer valid records into Oracle Purchase Orders interface tables. The standard import program ‘Import Standard Purchase Orders’ is then run to perform the validation on data in the interface table and load valid data into base tables.
 1. The **Purchase Order** data will be provided through data files. This data file will be loaded into staging table(s) on the Oracle applications database using loader program.
 2. The loaded data will be validated from staging tables and then moved into the standard purchase order open interface tables.
 3. The standard **Import Standard Purchase Orders** program will be called to import the standard purchase orders from Interface table to Oracle Purchase Orders base tables.

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.

Source system

The implementation team studies the source system and understands the data required to be migrated to Oracle Applications. This data is then extracted and cleansed from the source system (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team. The data mapping section describes the data format of source system to Oracle Applications.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Purchasing Super User
 - System Administrator

- Users have access to Oracle Applications server access with write privilege.
 - Users have access to Oracle Applications database (**apps schema**)
 - All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.
 - The extract-file layout must be in the format as specified in the extract file layout section
-

Pre-Requisites

Prerequisite set-ups are required for the interface. These setups include, but are not limited to:

1. Purchasing Options
2. Receiving Options
3. Financial Options
4. Document Type Lookup
5. Approval Setup
6. Currency
7. System Profile Options for Oracle Payables
8. Tax Codes and Rates

Data Dependencies

Purchase Order

In order to migrate **Purchase Order** the following entities must have been migrated successfully:

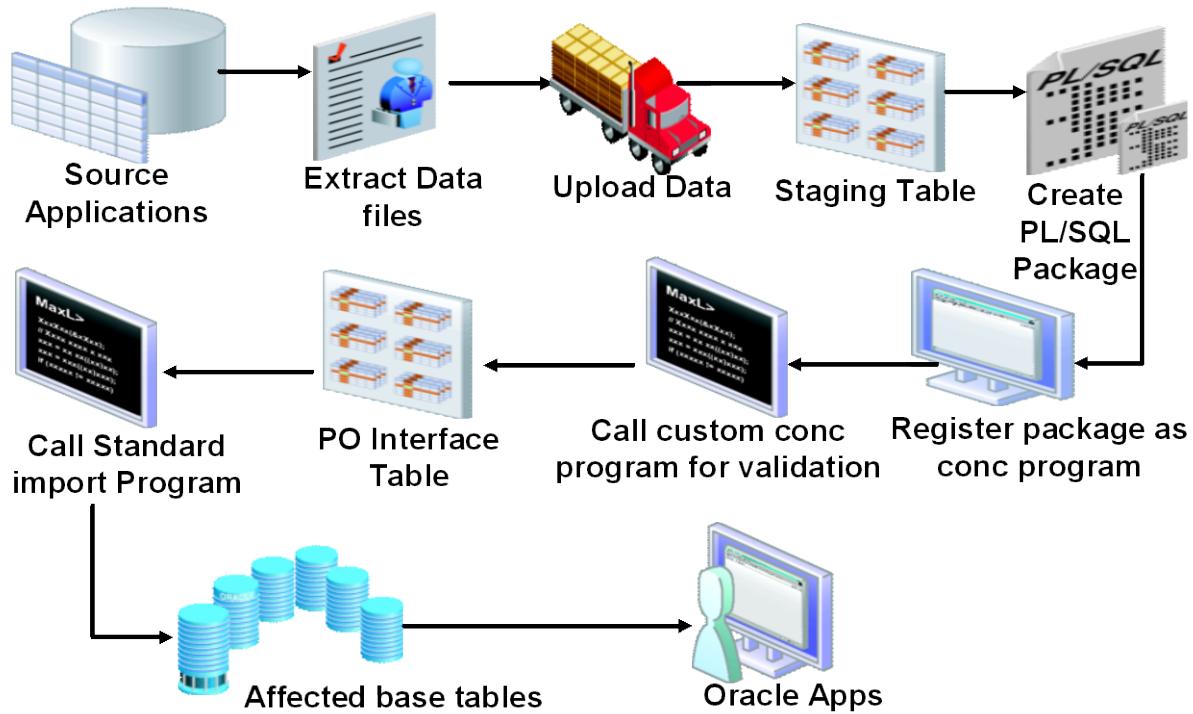
- Suppliers
- Inventory Items

Business Rules

Following business rules are applicable for the [Purchase Order](#) conversion process.

- N/A

Approach



The Approach for migrating **Purchase Order** data from source system to Purchase Orders open interface table

1. Extract the data file from source system
2. Load the data file into staging table
3. Using PL/SQL write a package
4. Register the PL/SQL package as concurrent program in Oracle Applications
5. Call the custom concurrent program for validation and loading the records available in staging table to open interface table
6. Call the standard import program, which will create the records in PO_HEADERS_ALL and PO_LINES_ALL tables based on open interface records.
7. Query the records, which got created for interface table and see the result in Oracle Applications front end.

Module List

Concurrent Programs

‘XX<EMPID>: Purchase Order Interface Program’ includes the following concurrent programs:

‘XX<EMPID>: Purchase Order Interface Program’

The ‘XX<EMPID>: Purchase Order Interface Program’ concurrent program is based on the stored procedure **XXPO_PURCHASE_ORDER_PKG.main** that is registered as a PL/SQL executable.

- The above program calls loader program (**XX<EMPID>: Purchase Order Interface Loader Program**) to load data into staging tables, performs validations on the staging table data and uploads data into the open interface tables (**PO_HEADERS_INTERFACE** and **PO_LINES_INTERFACE**).

‘XX<EMPID>: Purchase Order Interface Loader Program’

The ‘XX<EMPID>: Purchase Order Interface Loader Program’ concurrent program is based on the SQL* Loader control file, that is registered as a Loader executable.

- The above program loads data from the path specified as parameter value in loader concurrent program to the staging table.

NOTE:

1. Change the <EMPID> with your Employee ID,
2. Follow the steps for registering concurrent program from Registering Concurrent Program section.

Stored Procedures

XXPO_PURCHASE_ORDER_PKG package consist the following stored procedures:

XXPO_PURCHASE_ORDER_PKG.main

This is the main procedure that calls the load, validate, import, print_error and record_summary procedures.

XXPO_PURCHASE_ORDER_PKG.validate

This procedure performs the necessary validations on the staging tables.

XXPO_PURCHASE_ORDER_PKG.import

This procedure loads all the valid data from the staging table to Oracle Invoices interface tables PO_HEADERS_INTERFACE and PO_LINES_INTERFACE .

XXPO_PURCHASE_ORDER_PKG.print_error

This procedure reports the errored records in the log file.

XXPO_PURCHASE_ORDER_PKG.record_summary

This procedure will display the status of staging table records

NOTE: Please refer the attached documents for Package Specifications and Package body. Kindly open the attached script in notepad.



XXPO_PURCHASE_O
RDER_PKG.pks

1. Package Specification:



XXPO_PURCHASE_O
RDER_PKG.pkb

2. Package Body:

Staging Table

XXPO_PURCHASE_ORDER_IFACE_STG table created in database to store records from source system data file.

NOTE: Please refer the attached script staging table creation. Kindly open the attached script in notepad.



XXPO_PURCHASE_O
RDER_TAB.tab

Table Creation Script:

Grant Script

If table and package are created in custom schema then grant the table and package to APPS schema

NOTE: Please refer the attached script for table and package granting. Kindly open the attached script in notepad.



XXPO_PURCHASE_O
RDER_PKG.grt

1. Package Grant Script:



XXPO_PURCHASE_O
RDER_TAB.grt

2. Table Grant Script:

Synonym Script

If table and package are created in custom schema then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



XXPO_PURCHASE_O
RDER_PKG.syn

1. Package synonym script:



XXPO_PURCHASE_O
RDER_TAB.syn

2. Table synonym script:

SQL* Loader Script

SQL*Loader is controlled by its own data definition language, which is kept in the control file. The control file describes the data to be loaded, the destination tables of the data and describes the interdependency between the data and the columns within the tables. Which in-turn used to register as a concurrent program in Oracle Applications to load the data file into staging table.

NOTE: Please refer the attached SQL* Loader script for loading data file into staging table. Kindly open the attached script in notepad.



XXPO_PURCHASE_O
RDER_LOAD.ctl

1. SQL* Loader Script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script
3. Staging Table Script
4. SQL* Loader Script

NOTE: Please refer the attached Install script for deployment of Purchase Order object on Oracle Applications Server. Kindly open the attached script in notepad.



1. Install Script:

Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

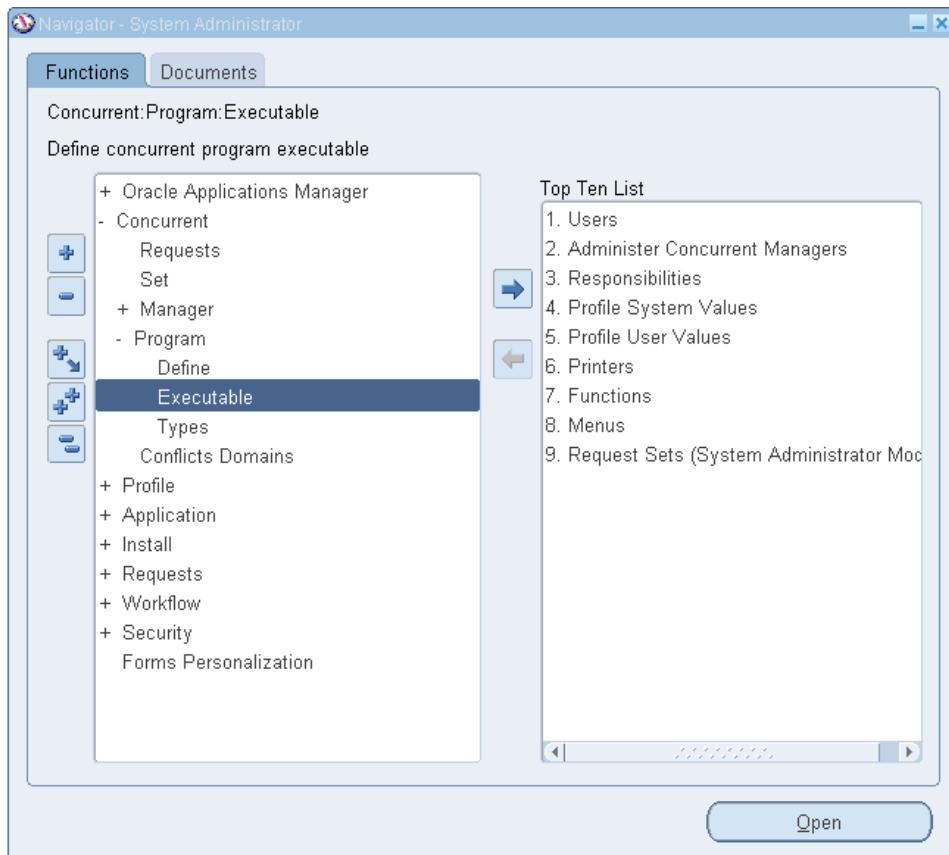
- A.) Package Name
- B.) Table Name
- C.) Synonym Name
- D.) Concurrent Program Executable.
- E.) Concurrent Program Name.

Registering Concurrent Program

Registering Interface Loader Concurrent Program

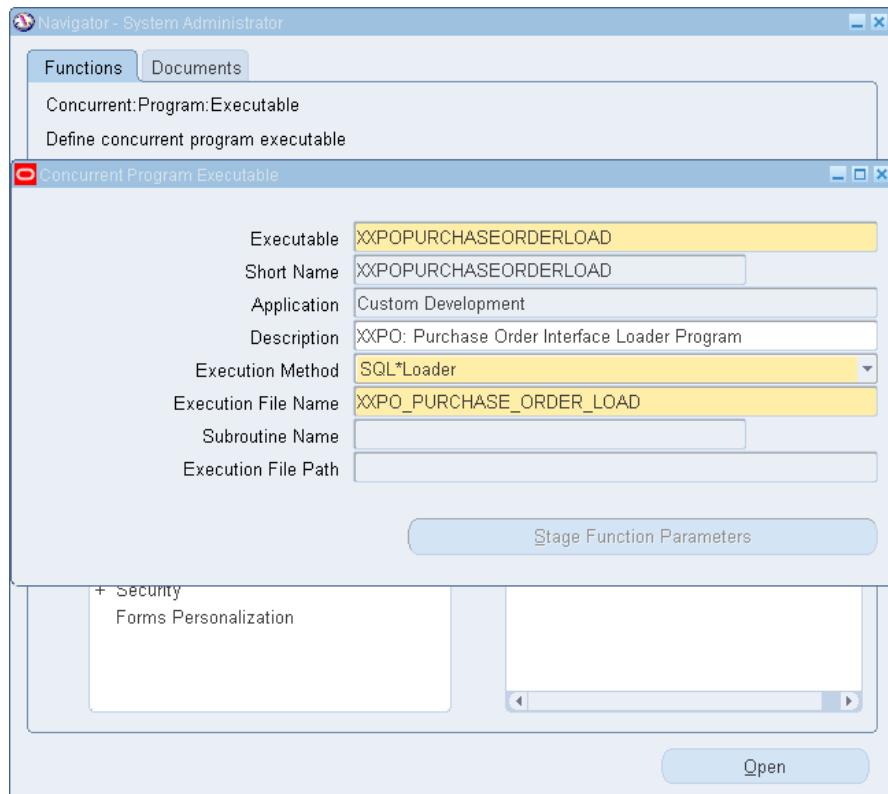
To register a concurrent program for '[XX<EMPID>: Purchase Order Interface Loader Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



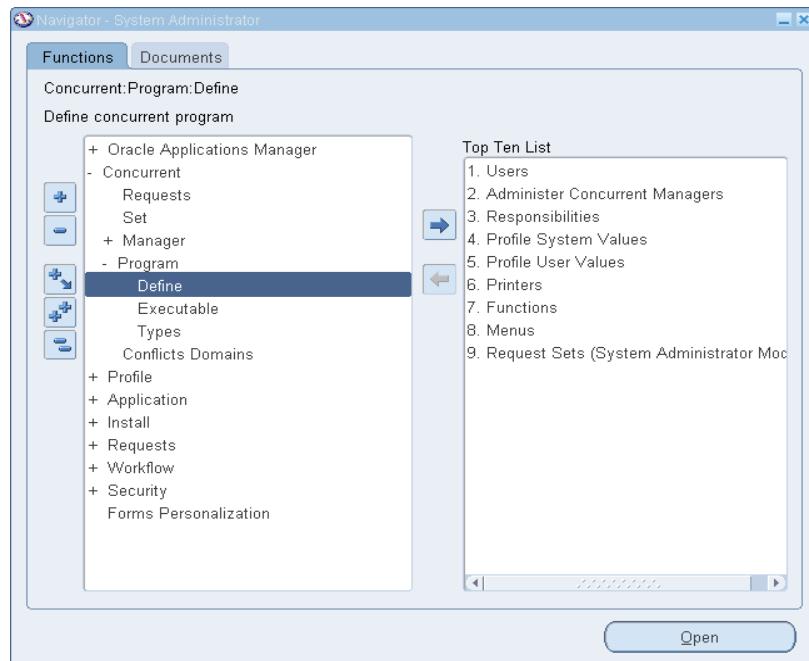
2. Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for Purchase Order so the Application Name should be [Custom Application](#).
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be SQL*Loader, because we are registering a concurrent program for loading the data to staging table.
- Execution File Name:** Here we would be passing the file name of SQL*Loader control file. As discussed in Module List the control file name would be [XXPO_PURCHASE_ORDER_LOAD](#)



Executable Registering Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name: Enter “XX<EMPID>: Purchase Order Interface Loader Program”
- Short Name: Enter a unique concurrent program short name, for example: **XXPOPURCHASEORDERLOAD**
- Application: Select the value from LOV, as we would be executing this concurrent program from Payables Applications so select Custom Development and add this concurrent program in Purchase Order Request Group.
- Description: Enter the description of this concurrent program.
- Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXPOPURCHASEORDERLOAD** from LOV.
- Executable Method: Executable method will be populated automatically.
- Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A). Data File

The screenshot shows the 'Concurrent Programs' dialog box. The 'Program' field is set to 'XXPO: Purchase Order Interface Loader Program'. The 'Enabled' checkbox is checked. The 'Short Name' is 'XXPOPURCHASEORDERLOAD'. Under 'Application', 'Custom Development' is selected. The 'Description' field is empty. In the 'Executable' section, the 'Name' is 'XXPOPURCHASEORDERLOAD' and the 'Method' is 'SQL*Loader'. In the 'Request' section, 'Type' is empty, 'Incrementor' is empty, and 'MLS Function' is empty. Under 'Request' checkboxes, 'Use in SRS' is checked, while 'Run Alone', 'Enable Trace', 'Allow Disabled Values', 'Restart on System Failure', and 'NLS Compliant' are unchecked. In the 'Output' section, 'Format' is set to 'Text', with 'Save' and 'Print' checkboxes checked. 'Columns', 'Rows', 'Style', 'Style Required', and 'Printer' fields are empty. At the bottom are buttons for 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters'.

Concurrent Program Form

Concurrent Program Parameters form

Seq	Parameter	Description	Enabled
10	Data_File	Data file name with path	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>

Validation

Value Set	100 Characters	Description	100 Characters
Default Type		Default Value	
<input type="checkbox"/> Required	<input type="checkbox"/> Enable Security	Range	

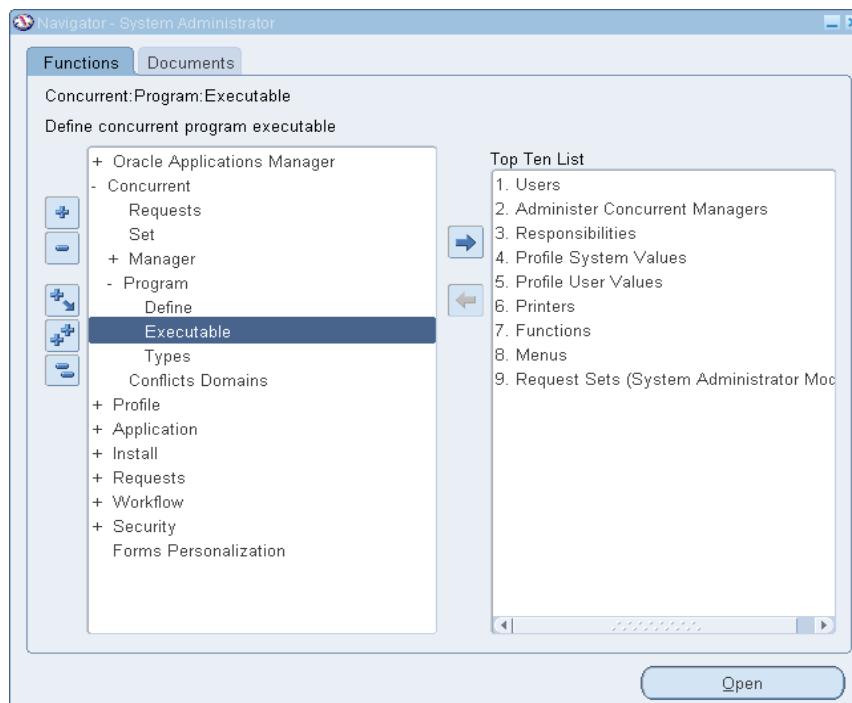
Display

Display Size	50	Description Size	50
Concatenated Description Size	25	Prompt	Data File
Token			

Registering Interface Concurrent Program

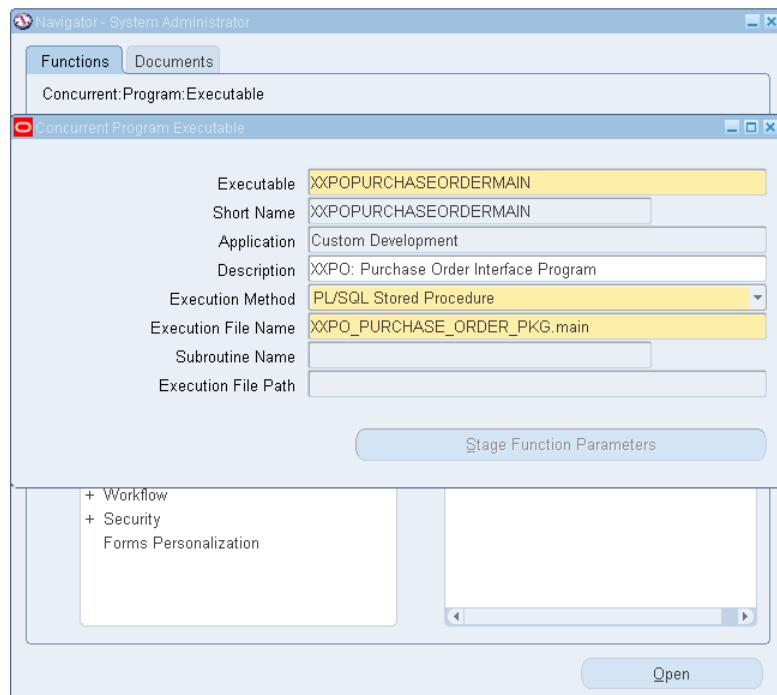
To register a concurrent program for '[XX<EMPID>: Purchase Order Interface Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable



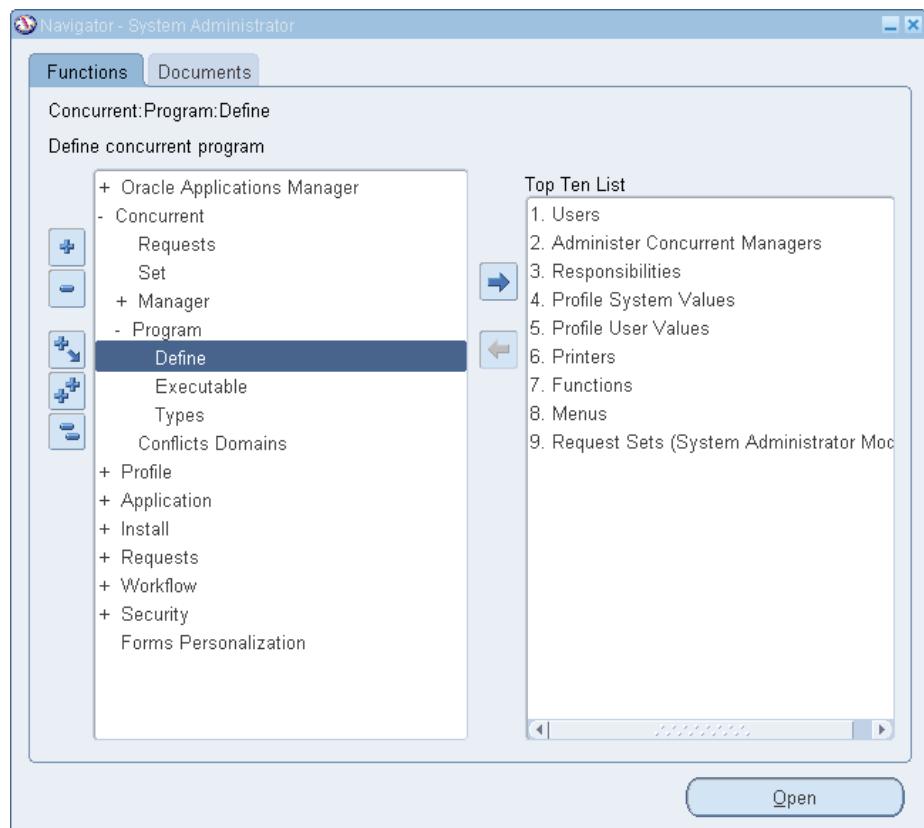
2 Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for Purchase Order so the Application Name should be **Custom Application**.
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be PL/SQL Stored Procedure.
- Execution File Name:** Here we would be passing the package name with one public procedure. As discussed in Module List the package name would be **XXPO_PURCHASE_ORDER_PKG** and the public procedure defined in this package is **main**.



Concurrent Program Executable Form

3. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- a. Program Name: Enter “XX<EMPID>: Purchase Order Interface Program”
- b. Short Name: Enter a unique concurrent program short name, for example: **XXPOPURCHASEORDERMAIN**
- c. Application: Select the value from LOV, as we would be executing this concurrent program from Payables Applications so select Custom Development and add this concurrent program in Purchase Order Request Group.
- d. Description: Enter the description of this concurrent program.
- e. Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXPOPURCHASEORDERMAIN** from LOV.
- f. Executable Method: Executable method will be populated automatically.
- g. Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A) Data File Path B) Data File Name

The screenshot shows the 'Concurrent Programs' window. The 'Program' field is set to 'XXPO: Purchase Order Interface Program'. The 'Enabled' checkbox is checked. The 'Short Name' is 'XXPOPURCHASEORDERMAIN', 'Application' is 'Custom Development', and 'Description' is empty. Under the 'Executable' section, 'Name' is 'XXPOPURCHASEORDERMAIN' and 'Method' is 'PL/SQL Stored Procedure'. In the 'Request' section, there are fields for 'Type', 'Incrementor', and 'MLS Function'. Checkboxes include 'Use in SRS', 'Run Alone', 'Enable Trace', 'Allow Disabled Values', 'Restart on System Failure', and 'NLS Compliant'. The 'Output' section specifies 'Format' as 'Text', with options for 'Save' and 'Print'. Buttons at the bottom include 'Copy to...', 'Session Control', 'Incompatibilities', and 'Parameters'.

Concurrent Program Form

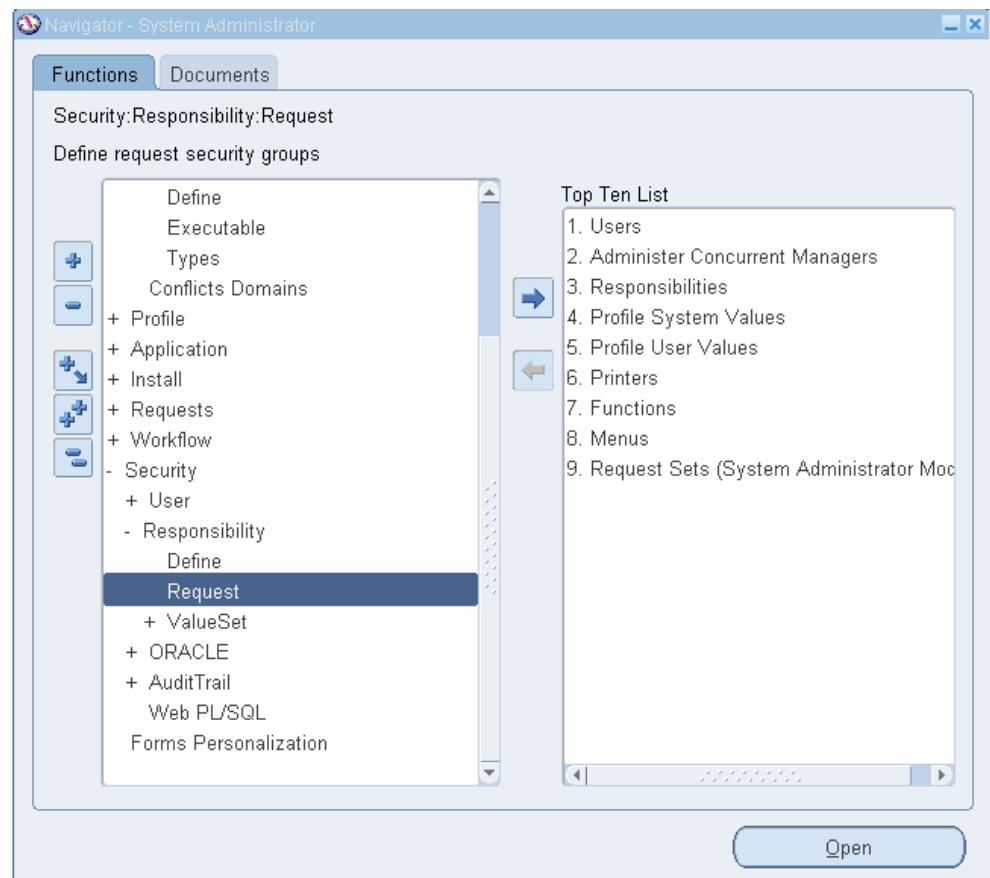
The screenshot shows the 'Concurrent Program Parameters' window. The 'Program' is 'XXPO: Purchase Order Interface Program' and the 'Application' is 'Custom Development'. The 'Conflicts Domain' and 'Security Group' fields are empty. A table lists parameters: Row 10 has 'Parameter' 'Data_File_Path' and 'Description' 'Data file path', with 'Enabled' checked. Row 20 has 'Parameter' 'Data_File_Name' and 'Description' 'Data file name', also with 'Enabled' checked. Below the table, 'Validation' settings include 'Value Set' '100 Characters', 'Default Type' empty, and checkboxes for 'Required' and 'Enable Security'. Under 'Display', 'Display Size' is 50, 'Concatenated Description Size' is 25, 'Description Size' is 50, and 'Prompt' is 'Data File Path'. A 'Token' field is at the bottom.

Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: Purchase Order Interface Program’ in Request Group, Navigate to [System Administrator Responsibility](#)

1. Select Security -> Responsibility-> Request



2. Query the Request Group – All Reports and Applications – Purchasing

The screenshot shows the 'Request Groups' window with the following details:

- Group:** All Reports
- Application:** Purchasing
- Code:** (empty)
- Description:** All Purchasing SRS Reports
- Requests:** A table listing concurrent programs:

Type	Name	Application
Application	Subledger Accounting	Subledger Accounting
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	Catalog Index Create - Intermedia	Oracle iProcurement
Program	Rebuild Catalog interMedia Index	Oracle iProcurement
Program	Purge Catalog interMedia Index	Oracle iProcurement
Program	Catalog Child Data Exceptions Report	Oracle iProcurement
Program	Supplier Mailing Labels	Payables
Program	Supplier Audit Report	Payables
Program	Suppliers Report	Payables
- Description:** (empty)

Request Group Form

3. Add an entry for Concurrent Program in the Request Group and save the work.

The screenshot shows the 'Request Groups' window with the following details, after adding a new entry:

- Group:** All Reports
- Application:** Purchasing
- Code:** (empty)
- Description:** All Purchasing SRS Reports
- Requests:** A table listing concurrent programs:

Type	Name	Application
Application	Subledger Accounting	Subledger Accounting
Program	XXPO: Purchase Order Interface Program	Custom Development
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	Catalog Index Create - Intermedia	Oracle iProcurement
Program	Rebuild Catalog interMedia Index	Oracle iProcurement
Program	Purge Catalog interMedia Index	Oracle iProcurement
Program	Catalog Child Data Exceptions Report	Oracle iProcurement
Program	Supplier Mailing Labels	Payables
Program	Supplier Audit Report	Payables
- Description:** (empty)

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and further it can be used to install in another Application Server.

Before downloading the script of Concurrent Program, connect your local system to Applications Server

Run the Following Command to download the script

```
FNDLOAD <Apps User/Apps Pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<Concurrent_Program_Short_Name>
```

For Example:

- To Download the “[XX<EMPID>: Purchase Order Interface Loader Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXPURCHASEORDERLOAD.ldt PROGRAM
CONCURRENT_PROGRAM_NAME =
XXPURCHASEORDERLOAD
```

- To Download the “[XX<EMPID>: Purchase Order Interface Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXPURCHASEORDERMAIN.ldt PROGRAM
CONCURRENT_PROGRAM_NAME =
XXPURCHASEORDERMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



1. Save the attached zip file in local system
2. Transfer the zip to the Oracle Applications Server
3. Create a Staging Directory [E.g. XXPOINTOIT] in temporary area on Oracle Application Server.
4. `mkdir -p XXPOINTOIT`
5. Change the Permissions on the temp directory.
6. `chmod 755 XXPOINTOIT`
7. FTP the **XXPOINTOIT.tar.gz** file in BINARY mode to directory **XXPOINTOIT**
8. Change directory to **XXPOINTOIT** as given below
9. `cd XXPOINTOIT`
10. Uncompress **XXPOINTOIT.tar.gz** as given below
11. `gunzip XXPOINTOIT.tar.gz`
12. Untar the file **XXPOINTOIT.tar** using
13. `tar -xvf XXPOINTOIT.tar.gz`
14. Grant the execute permission on the install script using
15. `chmod 755 XXPO_PURCHASE_ORDER.install`
16. Run `XXPO_PURCHASE_ORDER.install`
17. `sh XXPO_PURCHASE_ORDER.install`
18. Enter the apps schema user name password (apps/ <apps_pwd>), when asked for.

Extract File Layout

Please find the attached extract file for Purchase Order.



PURCHASING_SAME
PLE_DATA_FILE.csv

Note: Please remove the header row before putting the same file on Oracle Application Server for loading purpose.

Below is the example of extract file for Purchase Order

	A	B	C	D	E	F	G	H	I	J	K
1	vendor_number	doc_type_cc	agent_number	bill_to_location	ship_to_location	curr_inventory_item	unit_of_measur	unit_price	quantity	line_number	
2	1013	STANDARD		30 FRESNO	FRESNO	USD f10000	Ea	10	10	1	
3	1013	STANDARD		30 FRESNO	FRESNO	USD f10000	Ea	20	20	2	
4	1013	STANDARD		30 SANTA CLARA-ERS	SANTA CLARA-ERS	USD f10000	Ea	20	20	1	
5	1013	STANDARD		30 SANTA CLARA-ERS	SANTA CLARA-ERS	USD f10000	Ea	20	20	2	
6	1013	STANDARD		30 SANTA CLARA-ERS	SANTA CLARA-ERS	USD f10000	Ea	20	20	3	
7											
8											
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											
26											
27											
28											
29											
30											
31											

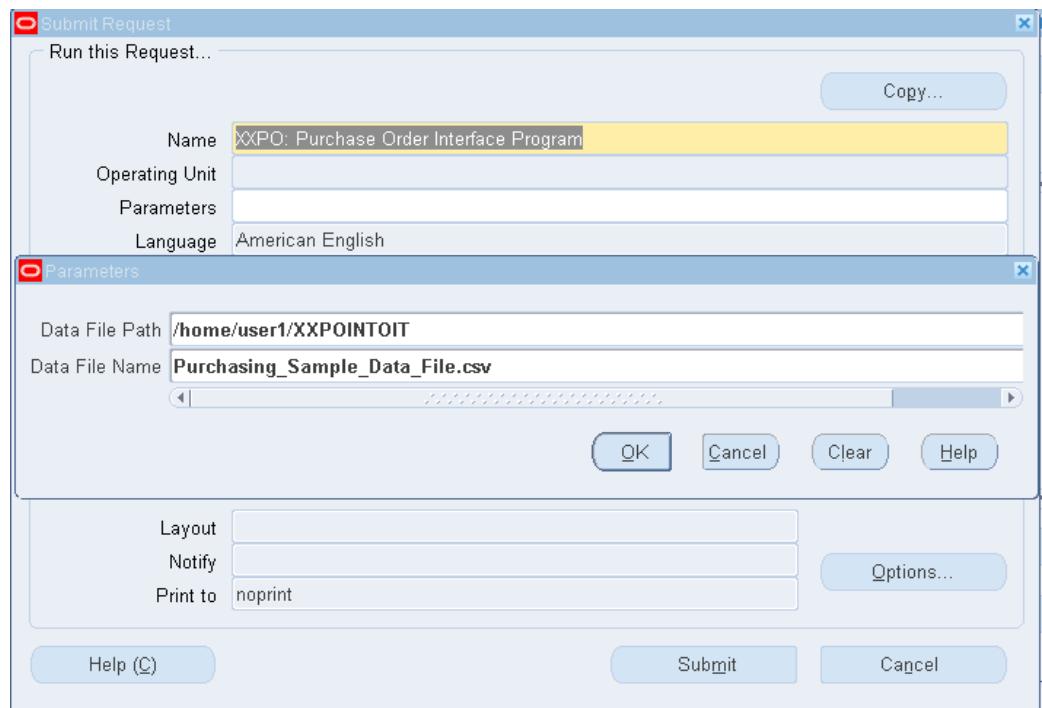
Extract File Mapping to Staging Table Columns

Data File Columns	Datatype	Size	Staging Table's Columns
Vendor_number	Varchar2	30	Vendor_number
Document_type_code	Varchar2	10	Document_type_code
Agent_number	Varchar2	240	Agent_number
Bill_to_location	Varchar2	30	Bill_to_location
Ship_to_location	Varchar2	30	Ship_to_location
Curr_code	Varchar2	10	Curr_code
Inventory_item	Varchar2	30	Inventory_item
Unit_of_measure	Varchar2	15	Unit_of_measure
Unit_price	Number		Unit_price
Quantity	Number		Quantity
Need_by_date	DATE		Need_by_date
Line_number	NUMBER		Line_number

Calling Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “XX<EMPID>: Purchase Order Interface Program”. To run the concurrent program navigate to [Purchasing Responsibility](#).

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



SRS Form

1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

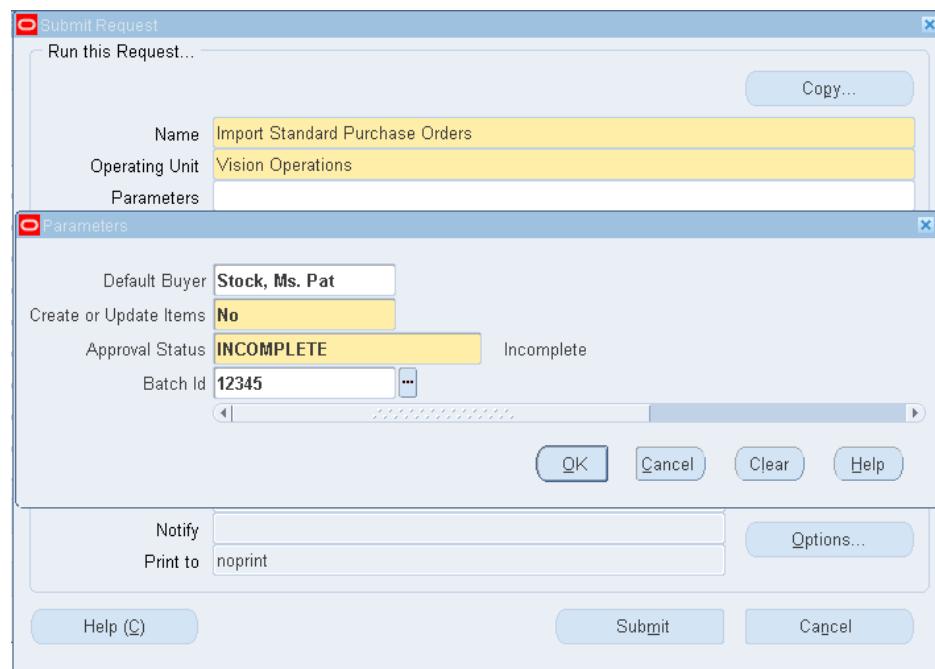
Calling Standard Import Concurrent Program

When “[XX<EMPID>: Purchase Order Interface Program](#)” completed successfully, run the standard import program to populate the data from AP Transaction Interface Table (RA_INTERFACE_LINES_ALL) to Payables base table.

The standard Import “[Import Standard Purchase Orders](#)” program will fetch the data from interface table and populate it into the Purchase Orders base tables.

Call the “[Import Standard Purchase Orders](#)” Program with the following Parameters

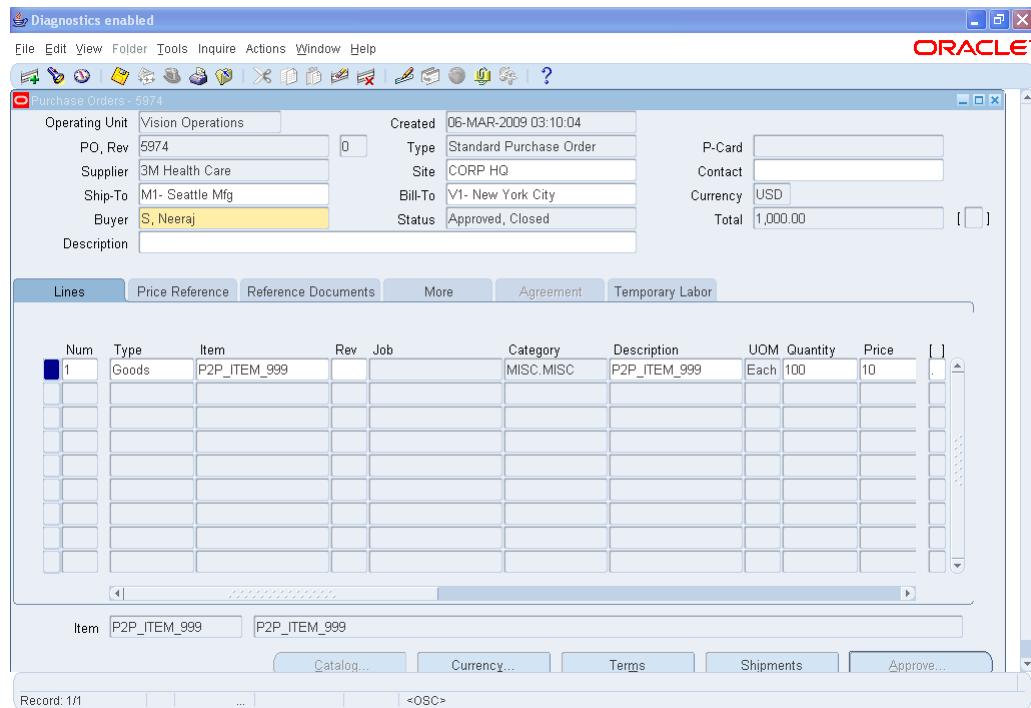
- a. Default Buyer: Stock, Ms. Pat (For Operations User)
- b. Create or Update Items: No
- c. Approval Status: INCOMPLETE
- d. Batch Id: Query the staging table for last run and pass it here



Submission of Standard Import Program

Query interface records

When “[Import Standard Purchase Orders](#)” standard import program completes successfully navigates to **Purchasing Responsibility**, **Select Purchase Orders -> Purchase Orders** and queries the records of interface table.



Purchase Orders Form

Error Handling

Following error can occur while processing the AP Transaction data file?

1. During loading data into staging table.

- When error occurred while loading the data file using SQL*Loader concurrent program it generates two files they are
 - **SQL*Loader Bad File:** The bad file contains records that weren't loaded into the staging table. These records could have been rejected by SQL*Loader due to an invalid format. Also they could have been rejected by the Oracle database if they violate an integrity constraint or had an invalid data type for the staging table.
 - **SQL*Loader Log File:** Detailed information about the load is stored in the log file. Any errors found during parsing of the control file are stored in the log file. The log file also identifies the number of records successfully loaded. The log file must be available during the entire run of the SQL*Loader. When loading data with SQL*Loader, nothing should be assumed without reviewing the log files

2. Validating the records in staging table

- While validating the data in staging table it can complete with error and the concurrent program will complete with warning, in this case look into the output report of concurrent program and correct the data file and re-load it again.

3. While calling the Standard AutoInvoice Import Program

- After populating the records in interface table it again validated by the standard import program and in few cases it completes with error or warning, in this case see the output report of Standard Import Program and correct the data in data file and re-load it, than re-run the custom concurrent program.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Purchase Order Extract Outbound

Case Study

Contents

Document Control.....	ii
Introduction.....	1
Purpose.....	1
Background	1
Scope and Application	1
Audience	2
Source system	2
Assumptions.....	2
Pre-Requisites	3
Business Rules	3
Approach.....	4
Module List.....	5
Registering Concurrent Program	8
Downloading Concurrent Program Script	14
Deployment.....	15
Extract File Layout	16
Calling Extract Custom Concurrent Program	17
Open and Closed Issues for this Deliverable	18
Open Issues	18
Closed Issues.....	18

Introduction

A purchase order (PO) is a commercial document issued by a buyer to a supplier, indicating the type, quantities and agreed prices for products or services the seller will provide to the buyer. It will specify payment terms, delivery dates, item identification, quantities, shipping terms and all other obligations and conditions.

Purchase orders are generally preprinted, numbered documents generated by the retailer's financial management system, which shows that purchase details have been recorded and payment will be made.

Use the Purchase Order window to enter the standard purchase order, blanket purchase order, planed purchase order, purchase requisitions etc. You can also query and update purchase order in this window.

To enter or query a [Purchase Order](#) from front end navigate to Purchasing Responsibility, Select [Purchase Orders -> Purchase Orders](#).

This document demonstrates the extract of purchase order data, which is approved and received.

Purpose

This document describes the:

- This case study is intended to demonstrate the extraction of data from Oracle Applications Purchase Order for legacy system.
- File layout to be used for interface.
- Approach and technical design for extractions.

Background

This case study document is designed to demonstrate a practical scenario that occurs during implementation of oracle Purchase Orders.

Scope and Application

The following boundaries are specific to the extractions of [Purchase Order](#) data using [PO_HEADERS_ALL](#), [PO_LINES_ALL](#), [PO_LINE_LOCATIONS_ALL](#), [RCV_SHIPMENT_HEADERS](#) and [RCV_SHIPMENT_LINES](#) standards table

- All standard purchase orders are available in Oracle Applications

1. The ‘XX<EMPID>: Purchase Orders Extract Program’ create a file which will contain Purchase Order data which are approved and received.
 2. The ‘XX<EMPID>: Purchase Orders Extract Program’ have two parameters.
 - **Data File Path:** Enter the path of Oracle Applications Server, where this custom concurrent program will create the file for outbound.
 - **Data File Name:** Enter the file name, which will be used for naming the file while extracting the data from Oracle Applications.
-

Audience

This document is intended for the following individuals:

- This document should be used for demonstration purposes only.
-

Source system

The implementation team studies the source system and understands the data required to be migrated from Oracle Applications to legacy system. This data is then extracted and cleansed from the Oracle Applications (this is done by IT team of source system) in the format as agreed by the client, source system IT team and the implementation team.

Assumptions

This case study document is based on the following assumptions:

- Users have access to Oracle Applications for the following responsibility.
 - Purchasing Super User
 - System Administrator
- Users have access to Oracle Applications server access with write privilege.
- Users have access to Oracle Applications database (**apps schema**)
- All the functional setups and master data (as listed in pre-requisites) are already completed in Oracle Applications.

- The extract-file layout must be in the format as specified in the extract file layout section

Pre-Requisites

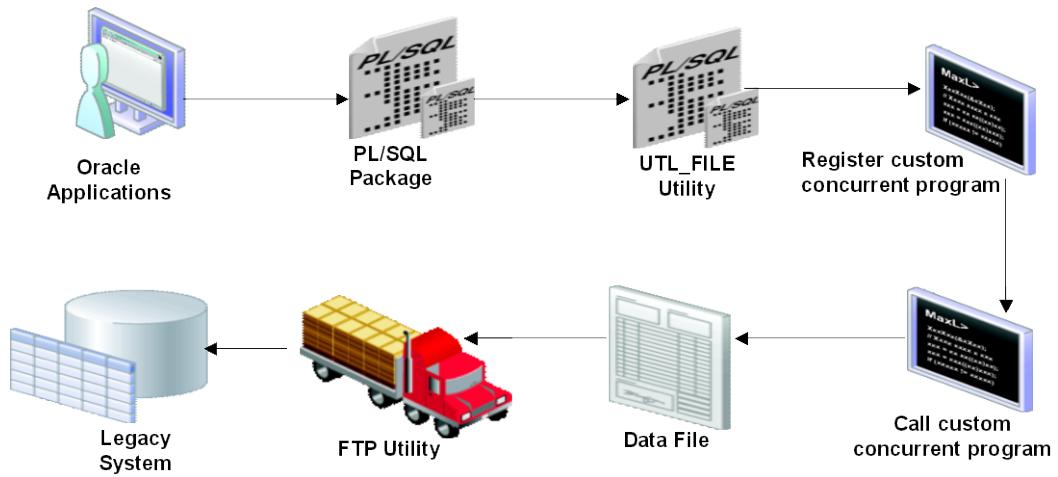
Approved and Received Purchase order records should be available in Oracle Applications

Business Rules

Following business rules are applicable for the [Purchase Order](#) extract process.

- Extract all the purchase order records which are approved and received
- Please refer Extract file layout for the columns to be extract for the purchase order.

Approach



The Approach for extracting **Purchase Order** data from Oracle Applications to a data file is

1. Login to the Oracle Applications
2. Using PL/SQL write a package
3. Use UTL_FILE utility in PL/SQL package to create the data file
4. Data file would be created by custom concurrent program
5. Use any FTP utility to transfer the extract data file into legacy system.

Module List

Concurrent Programs

'XX<EMPID>: Purchase Order Extract Program'

The '**'XX<EMPID>: Purchase Order Extract Program'**' concurrent program is based on the stored procedure **XXPO_PURCHASE_OUTBOUND_PKG.main** that is registered as a PL/SQL executable.

- The above program calls stored procedures to create the data file (Data file name and path passed as parameter values)

NOTE:

1. Change the <EMPID> with your Employee ID,
2. Follow the steps for registering concurrent program from Registering Concurrent Program section.

Stored Procedures

XXPO_PURCHASE_OUTBOUND_PKG package consist the following stored procedures:

XXPO_PURCHASE_OUTBOUND_PKG.main

This is the main procedure that calls the `create_headers` and `create_lines` procedures.

XXPO_PURCHASE_OUTBOUND_PKG.create_headers

This procedure is used to create the header for the extract file.

XXPO_PURCHASE_OUTBOUND_PKG.create_lines

This procedure is used to create the lines for the extract file.

NOTE: Please refer the attached documents for Package Specifications and Package body. Kindly open the attached script in notepad.



XXPO_PURCHASE_OUTBOUND_PKG.pks

1. Package Specification:



XXPO_PURCHASE_O

UTBOUND_PKG.pkb

2. Package Body:

Grant Script

If package is created in custom schema then grant the package to APPS schema

NOTE: Please refer the attached script for table and package granting. Kindly open the attached script in notepad.



XXPO_PURCHASE_O

UTBOUND_PKG.grt

1. Package Grant Script:

Synonym Script

If table and package are created in custom schema then grant the package and create synonym for table and package in APPS schema

NOTE: Please refer the attached script for table and package synonym. Kindly open the attached script in notepad.



XXPO_PURCHASE_O

UTBOUND_PKG.syn

1. Package synonym script:

Install Script

Install script contains UNIX Shell Script and will be used to deploy the objects attached to this document on Oracle Applications Server.

Following are the object to be deployed on server.

1. Package Specification Script
2. Package Body Script

NOTE: Please refer the attached Install script for deployment of Purchase Order object on Oracle Applications Server. Kindly open the attached script in notepad.



XXPO_PURCHASE_O

UTBOUND.install

1. Install Script:

Note: Please prefix your employee ID for each object you create and deploy in Application Server and Database.

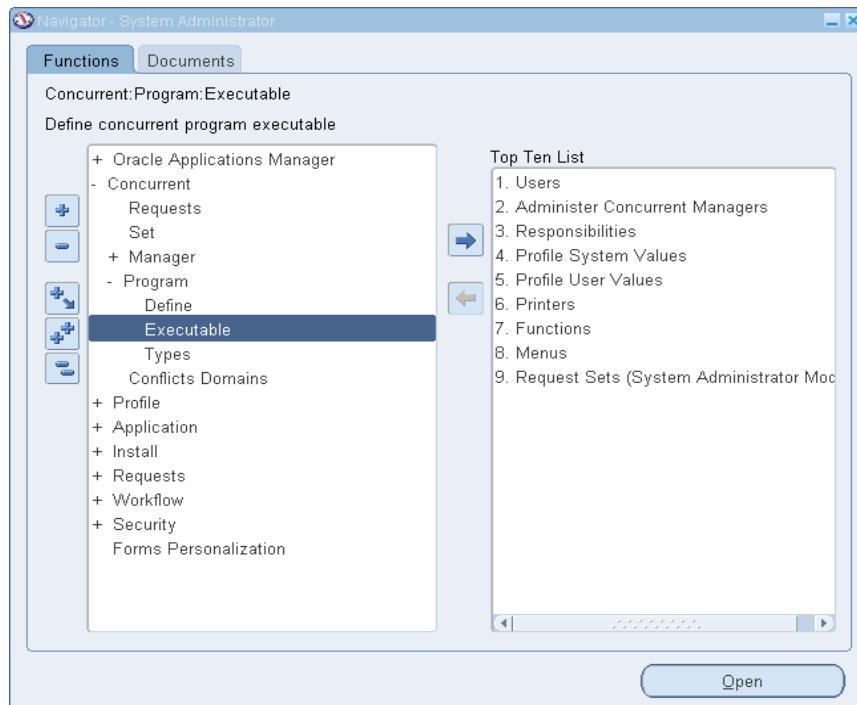
- A.) **Package Name**
- B.) **Table Name**
- C.) **Synonym Name**
- D.) **Concurrent Program Executable.**
- E.) **Concurrent Program Name.**

Registering Concurrent Program

Registering Extract Concurrent Program

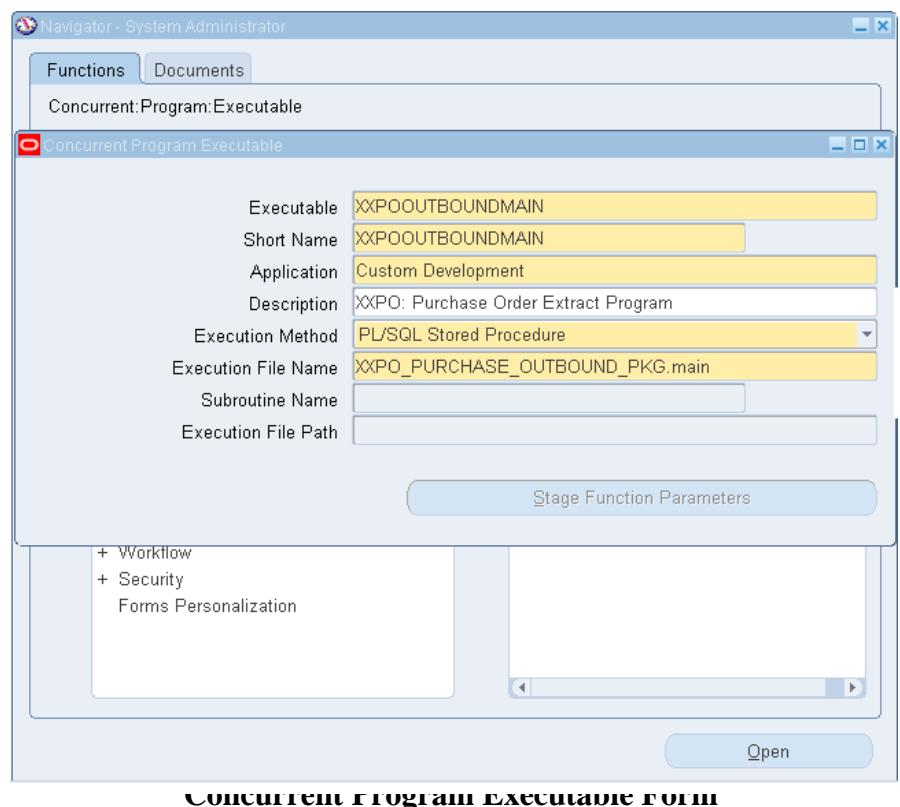
To register a concurrent program for '[XX<EMPID>: Purchase Order Extract Program](#)' in Oracle Applications Navigate to [System Administrator Responsibility](#)

1. Select Concurrent -> Program -> Executable

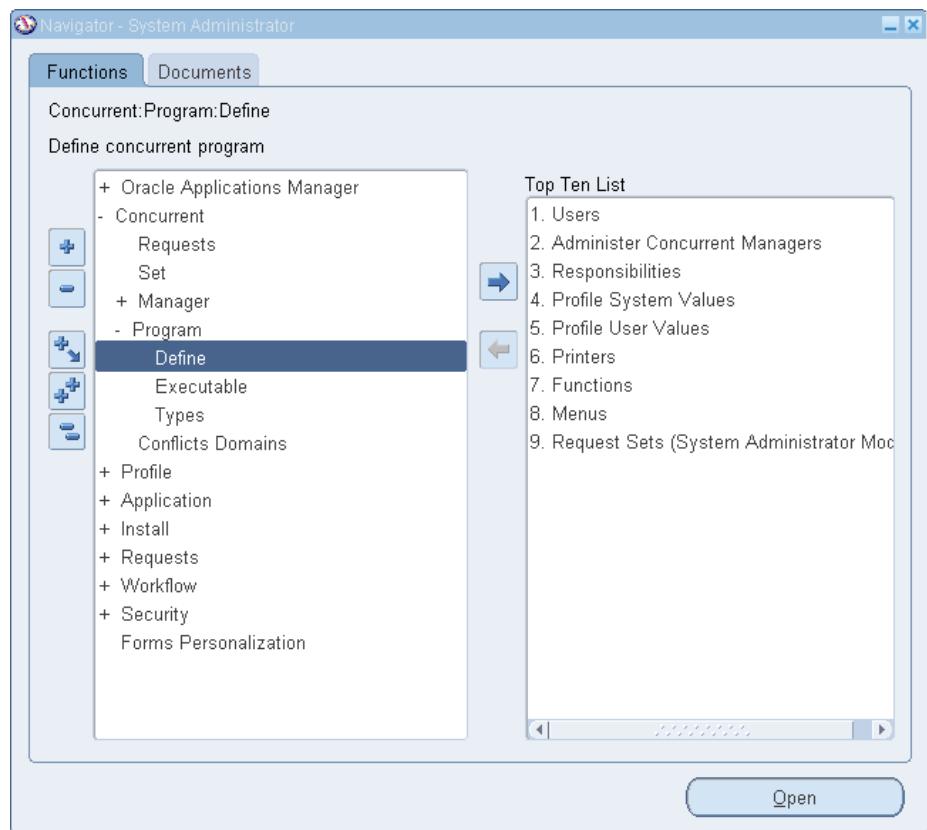


2 Enter The Following in the Concurrent Program Executable Form

- Executable Name:** Enter unique name for executable.
- Short Name:** Enter unique short name and this short name later will be used for defining Concurrent Program.
- Application:** Enter the Application Name from List of Values (LOV), as we are developing an interface program for Purchase Order so the Application Name should be **Custom Application**.
- Description:** Enter the description of Executable
- Execution Method:** Execution method should be PL/SQL Stored Procedure.
- Execution File Name:** Here we would be passing the package name with one public procedure. As discussed in Module List the package name would be **XXPO_PURCHASE_OUTBOUND_PKG** and the public procedure defined in this package is **main**.



2. Select Concurrent -> Program -> Define



4. Enter The Following in the Concurrent Program Form

- Program Name: Enter “XX<EMPID>: Purchase Order Extract Program”
- Short Name: Enter a unique concurrent program short name, for example: **XXPOOUTBOUNDMAIN**
- Application: Select the value from LOV, as we would be executing this concurrent program from Payables Applications so select Custom Development and add this concurrent program in Purchase Order Request Group.
- Description: Enter the description of this concurrent program.
- Executable Name: Select the executable name from LOV, as we have already registered an executable for loader program so select **XXPOOUTBOUNDMAIN** from LOV.
- Executable Method: Executable method will be populated automatically.
- Parameters if any: Select the Parameter Button to specify the parameter list. In our case there should be two parameters we would be passing. A) Data File Path B) Data File Name

The screenshot shows the 'Concurrent Programs' window with the following details:

- Program:** XXPO: Purchase Order Extract Program
- Short Name:** XXPOOUTBOUNDMAIN
- Application:** Custom Development
- Description:** (empty)
- Executable:**
 - Name:** XXPOOUTBOUNDMAIN
 - Method:** PL/SQL Stored Procedure
 - Options:** (empty)
 - Priority:** (empty)
- Request:**
 - Type:** (empty)
 - Incrementor:** (empty)
 - MLS Function:** (empty)
 - Checkboxes:**
 - Use in SRS
 - Run Alone
 - Enable Trace
 - Allow Disabled Values
 - Restart on System Failure
 - NLS Compliant
- Output:**
 - Format:** Text
 - Save (S)
 - Print
 - Columns:** (empty)
 - Rows:** (empty)
 - Style:** (empty)
 - Style Required
 - Printer:** (empty)

Buttons at the bottom: Copy to..., Session Control, Incompatibilities, Parameters.

Concurrent Program Form

The screenshot shows the 'Concurrent Program Parameters' window with the following details:

- Program:** XXPO: Purchase Order Extract Program
- Application:** Custom Development
- Conflicts Domain:** (empty)
- Security Group:** (empty)
- Parameter List:**

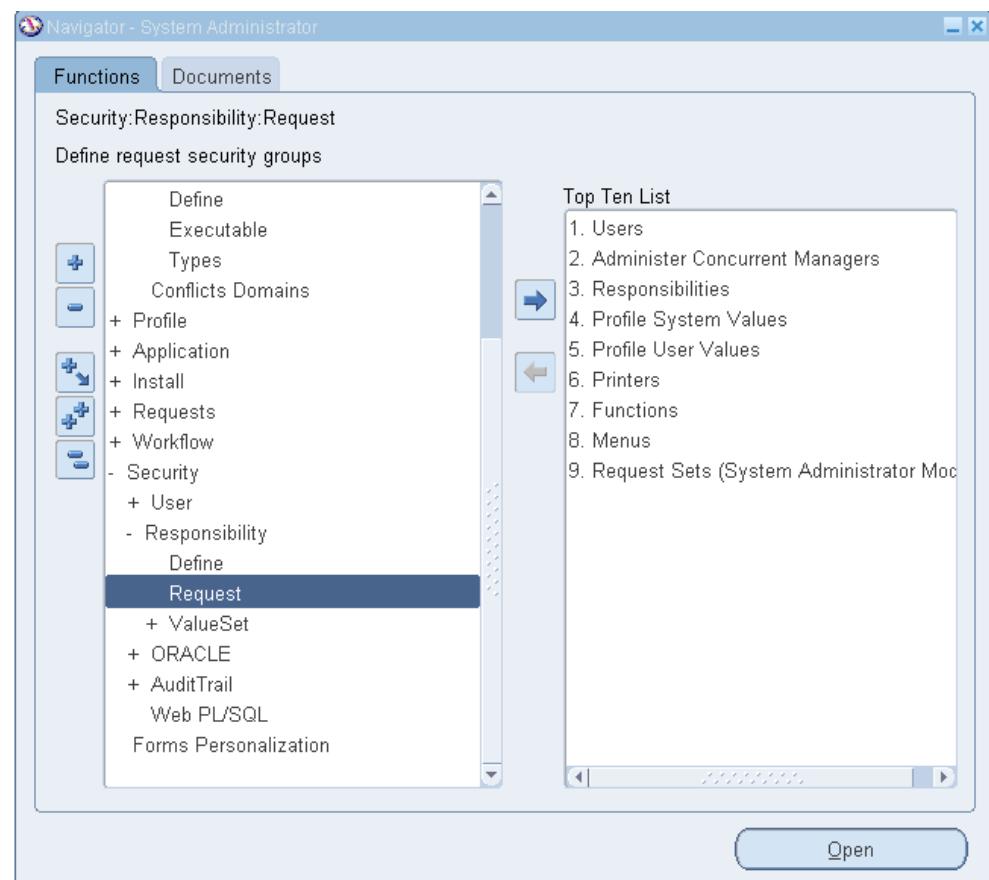
Seq	Parameter	Description	Enabled
10	Data_File_Path	Data file path	<input checked="" type="checkbox"/>
20	Data_File_Name	Data file name	<input checked="" type="checkbox"/>
			<input type="checkbox"/>
			<input type="checkbox"/>
- Validation:**
 - Value Set:** 100 Characters
 - Default Type:** (empty)
 - Required:**
 - Enable Security:**
 - Description:** 100 Characters
 - Default Value:** (empty)
 - Range:** (empty)
- Display:**
 - Display:**
 - Display Size:** 30
 - Concatenated Description Size:** 25
 - Description Size:** 50
 - Prompt:** Data File Name
 - Token:** (empty)

Concurrent Program Parameters Form

Registering Interface Concurrent Program in Request Group

To register a concurrent program ‘XX<EMPID>: Purchase Order Extract Program’ in Request Group, Navigate to [System Administrator Responsibility](#)

1. Select Security -> Responsibility-> Request



2. Query the Request Group – All Reports and Applications – Purchasing

The screenshot shows the 'Request Groups' window with the following details:

- Group:** All Reports
- Application:** Purchasing
- Code:** (empty)
- Description:** All Purchasing SRS Reports
- Requests:** A table listing concurrent programs:

Type	Name	Application
Application	Subledger Accounting	Subledger Accounting
Program	Workflow Background Process	Application Object Library
Program	ADS Financials	General Ledger
Program	Catalog Index Create - Intermedia	Oracle iProcurement
Program	Rebuild Catalog interMedia Index	Oracle iProcurement
Program	Purge Catalog interMedia Index	Oracle iProcurement
Program	Catalog Child Data Exceptions Report	Oracle iProcurement
Program	Supplier Mailing Labels	Payables
Program	Supplier Audit Report	Payables
Program	Suppliers Report	Payables
- Description:** (empty)

Request Group Form

3. Add an entry for Concurrent Program in the Request Group and save the work.

The screenshot shows the 'Request Groups' window with the following details, after adding a new program:

- Group:** All Reports
- Application:** Purchasing
- Code:** (empty)
- Description:** All Purchasing SRS Reports
- Requests:** A table listing concurrent programs, including the newly added one:

Type	Name	Application
Application	Subledger Accounting	Subledger Accounting
Program	Workflow Background Process	Application Object Library
Program	XXPO: Purchase Order Extract Program	Custom Development
Program	ADS Financials	General Ledger
Program	Catalog Index Create - Intermedia	Oracle iProcurement
Program	Rebuild Catalog interMedia Index	Oracle iProcurement
Program	Purge Catalog interMedia Index	Oracle iProcurement
Program	Catalog Child Data Exceptions Report	Oracle iProcurement
Program	Supplier Mailing Labels	Payables
Program	Supplier Audit Report	Payables
- Description:** (empty)

Request Group Form

Downloading Concurrent Program Script

After completion of Registering Concurrent program in System Administrator responsibility, we can download the script for concurrent program and further it can be used to install in another Application Server.

Before downloading the script of Concurrent Program, connect your local system to Applications Server

Run the Following Command to download the script

```
FNDLOAD <Apps User/Apps Pwd> 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct <File_Name.ldt> PROGRAM
CONCURRENT_PROGRAM_NAME=<Concurrent_Program_Short_Name>
```

For Example:

- To Download the “[XX<EMPID>: Purchase Order Extract Program](#)” script please enter the below command at unix/linux prompt.

```
FNDLOAD apps/apps 0 Y DOWNLOAD
$FND_TOP/patch/115/import/afcpprog.lct
XXPURCHASEORDERLOAD.ldt PROGRAM
CONCURRENT_PROGRAM_NAME = XXPOOUTBOUNDMAIN
```

Deployment

- To deploy the attached zip file which contains package, table scripts, grant script, synonym script, concurrent program script and install script, please follow the below steps



1. Save the attached zip file in local system
2. Transfer the zip to the Oracle Applications Server
3. Create a Staging Directory [E.g. XXPOOUTBOUND] in temporary area on Oracle Application Server.
4. `mkdir -p XXPOOUTBOUND`
5. Change the Permissions on the temp directory.
6. `chmod 755 XXPOOUTBOUND`
7. FTP the **XXPOOUTBOUND.tar.gz** file in BINARY mode to directory **XXPOOUTBOUND**
8. Change directory to XXPOOUTBOUND as given below
9. `cd XXPOOUTBOUND`
10. Uncompress **XXPOOUTBOUND.tar.gz** as given below
11. `gunzip XXPOOUTBOUND.tar.gz`
12. Untar the file **XXPOOUTBOUND.tar** using
13. `tar -xvf XXPOOUTBOUND.tar.gz`
14. Grant the execute permission on the install script using
15. `chmod 755 XXPO_PURCHASE_OUTBOUND.install`
16. Run `XXPO_PURCHASE_OUTBOUND.install`
17. `sh XXPO_PURCHASE_OUTBOUND.install`
18. Enter the apps schema user name password (apps/ <apps_pwd>), when asked for.

Extract File Layout

Data File Columns	Datatype	Size	Table Name
PO_NUMBER	VARCHAR2	30	PO_HEADERS_ALL
PO_TYPE	VARCHAR2	25	PO_HEADERS_ALL
AGENT_NAME	VARCHAR2	240	PER_ALL_PEOPLE_F
VENDOR_NAME	VARCHAR2	240	AP_SUPPLIERS
VENDOR_SITE	VARCHAR2	15	AP_SUPPLIER_SITES_ALL
PAYMENT_TERM	VARCHAR2	30	AP_TERMS_TL
CURR_CODE	VARCHAR2	30	PO_HEADERS_ALL
AUTH_STATUS	VARCHAR2	25	PO_HEADERS_ALL
LINE_NUM	NUMBER	1	PO_LINES_ALL
INVENTORY_ITEM	VARCHAR2	30	MTL_SYSTEM_ITEMS
ITEM_DESCRIPTION	VARCHAR2	250	MTL_SYSTEM_ITEMS
UOM	VARCHAR2	25	MTL_SYSTEM_ITEMS
UNIT_PRICE	NUMBER		PO_LINES_ALL
LINE_QUANTITY	NUMBER		PO_LINES_ALL
RECEIPT_NUM	VARCHAR2	30	RCV_SHIPMENT_HEADERS
QUANTITY_RECEIVED	NUMBER		RCV_SHIPMENT_LINES

Please find the attached extract file for Purchase Order, which has been extracted, from Oracle Applications.



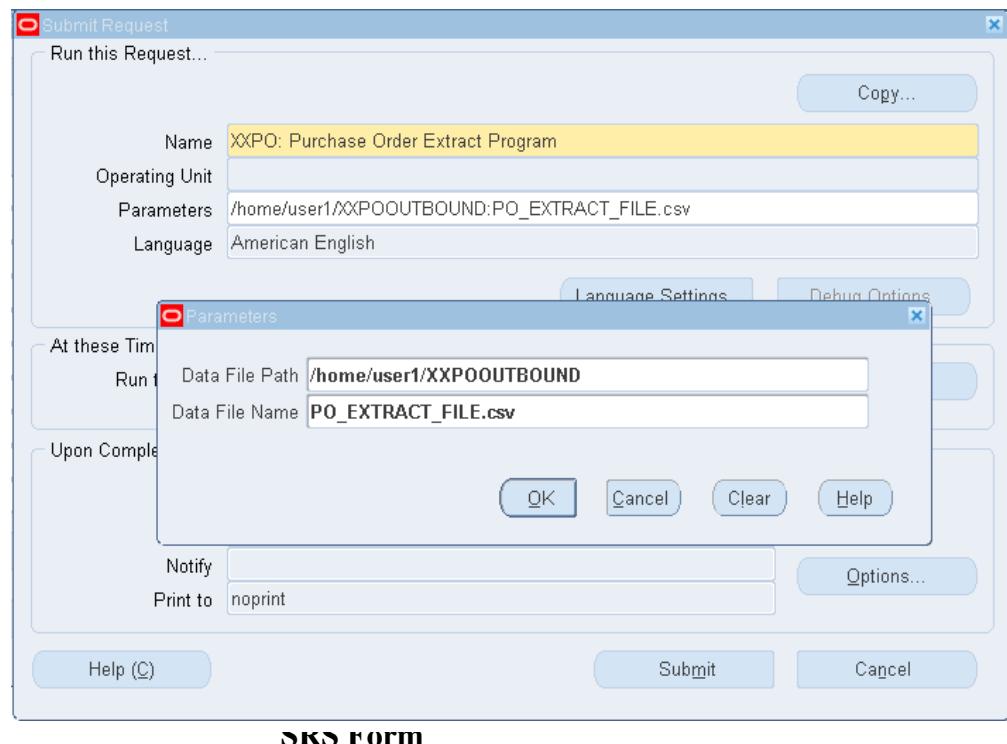
Calling Extract Custom Concurrent Program

After installation of all the script attached with this document run the concurrent program “XX<EMPID>: Purchase Order Extract Program”. To run the concurrent program navigate to [Purchasing Responsibility](#).

Use the below query to find out the path to create the extract file and use one of the path from value to pass as parameter value for Data File path.

```
SELECT VALUE
FROM V$PARAMETER
WHERE NAME LIKE '%utl_file_dir%';
```

Select View Menu -> Request, Submit a new request, and click on single request and than click OK Button.



1. Select the Concurrent Program name from LOV
2. Enter the parameter value for Data File Path and Data File Name.
3. Click on Submit Button to run the request.

Open and Closed Issues for this Deliverable

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

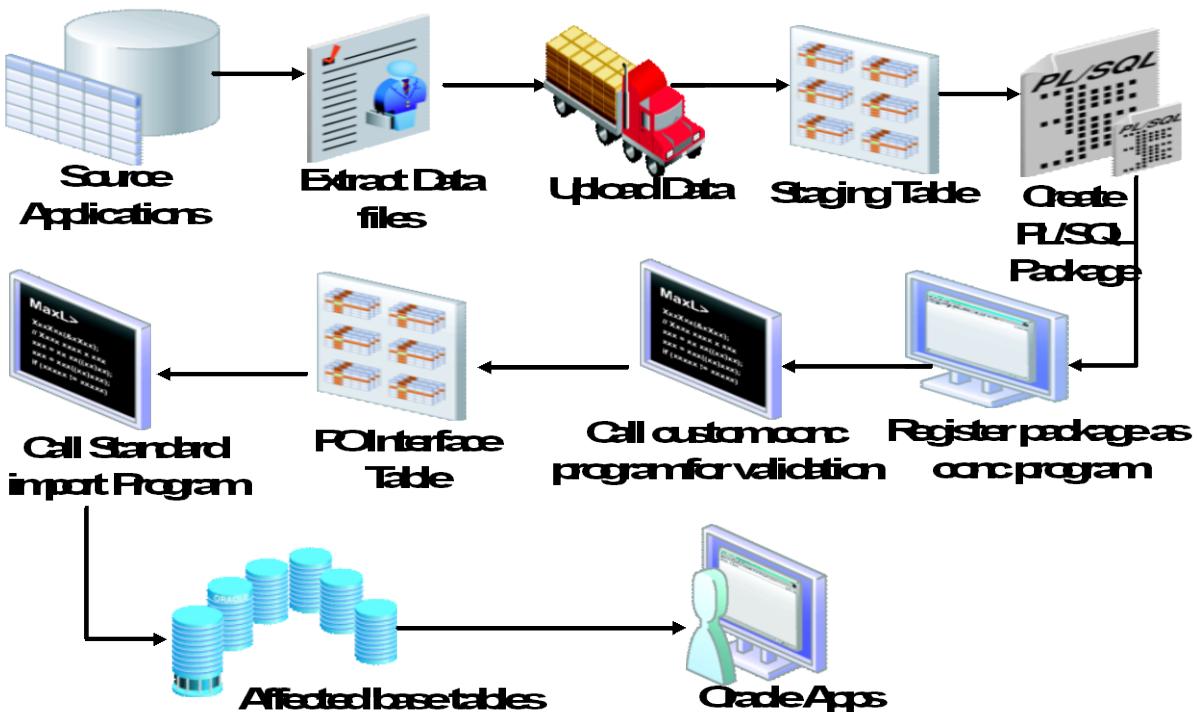
ID	Issue	Resolution	Responsibility	Target Date	Impact Date

CASE STUDY

1. Open Interface.....Duration 5 days.

The workshop – Interface design case study (Inbound and outbound) takes the student to evaluate the knowledge on

- Overall understanding of building open interface techniques in Oracle EBS
- Concurrent program & registration process.
- SQL / PL SQL understanding.
- Oracle Applications base table related to interfaces.
- Errors & error correction techniques



2. Business Scenario / Functional Doc:

Vision Corporation, Inc US has implemented Oracle EBS towards streamlining their business process. Apart from manual entry of Purchase orders by Purchase department staff in US, the Purchase orders are received from ABC Corporation Ltd (subsidiary company) based in China which are entered at their legacy system in China. ABC Corporation has worked out a strategy to automate the PO received from their subsidiary through available standard open interface in EBS.

The Purchase orders are sent through FTP in a prescribed flat file format ABC corporation server.

As a technical developer you are expected design for the inbound process which describes the approach, create & identify the appropriate table, and develop scripts which use the appropriate standard open interface process of purchase order, register in Oracle EBS environment and test the solution.

3. Problem statement / Technical doc :(Purchase Order Interface)

- Understand of Purchase Order Flat file from Source System and column mapping of Flat File to identify the column requirement for staging table and create the staging table.
- Create a Technical Specification of process involved.
- Understand SQL*Loader (Control File) to Load data from flat file to staging table.
- Identify the Open Interface table to Migrate Purchase Order from Source System and their mapping with staging table.
- Write suitable scripts to load into staging table and use PL/SQL procedure to perform pre validate of records and update flag of staging table in case of error record.
- Write Validation Before inserting data into Base table.
- Use concurrent program to move data into Base application tables.

Hints: Problem statement / Technical doc :(Purchase Order Interface)

- Understand Steps to Create Purchase Order in EBS (R12).
 - Create Standard and Blanket Purchase Order
- Understand of Purchase Order Flat file from Source System and column mapping of Flat File to identify the column requirement for staging table and create the staging table.
 - Structure of Flat File



	A	B	C	D	E	F	G	H	I	J	K
	vendor_number	doc_type_code	agent_number	bill_to_location	ship_to_location	curr_code	inventory_item	unit_of_measure	unit_price	quantity	line_number
1	1013	STANDARD	30	V1- New York City	V1- New York City	USD	f10000	Ea	10	10	1
2	1013	STANDARD	30	V1- New York City	V1- New York City	USD	f10000	Ea	20	20	2
3	1013	STANDARD	30	HR- San Jose	HR- San Jose	USD	f10000	Ea	20	20	1
4	1013	STANDARD	30	HR- San Jose	HR- San Jose	USD	f10000	Ea	20	20	2
5	1013	STANDARD	30	HR- San Jose	HR- San Jose	USD	f10000	Ea	20	20	3
6	1013	STANDARD	30	HR- San Jose	HR- San Jose	USD	f10000	Ea	20	20	

(Screen Shot of Flat File: Sample data)

Flat File mapping with staging table:

Data File Columns	Datatype	Size	Staging Table's Columns
Vendor_number	Varchar2	30	Vendor_number
Document_type_code	Varchar2	10	Document_type_code
Agent_number	Varchar2	240	Agent_number
Bill_to_location	Varchar2	30	Bill_to_location
Ship_to_location	Varchar2	30	Ship_to_location
Curr_code	Varchar2	10	Curr_code

Data File Columns	Datatype	Size	Staging Table's Columns
Inventory_item	Varchar2	30	Inventory_item
Unit_of_measure	Varchar2	15	Unit_of_measure
Unit_price	Number		Unit_price
Quantity	Number		Quantity
Need_by_date	DATE		Need_by_date
Line_number	NUMBER		Line_number

Hints : While we creating staging table please create few extra column i.e.

- Request_id : To capture Request_id of concurrent Program.
- Records_status: to update staging table for the records successfully inserted into base table after validation, initial value will be 'NEW'.
- error_message : to update error message in staging table.
- Who Columns : for History Purpose
 - created_by
 - creation_date
 - last_updated_by
 - last_updated_date
 - last_updated_login
- Create a Technical Specification of process involved.
- Understand SQL*Loader (Control File) to Load data from flat file to staging table.
- Identify the Open Interface table to Migrate Purchase Order from Source System and their mapping with staging table.

- Hints:
 - PO_HEADERS_INTERFACE
 - For Populating Header Information.
 - PO_LINES_INTERFACE
 - For Populating Lines Information.
 - PO_DISTRIBUTION_INTERFACE
 - For accounts at distribution level. (Note: Not required, from this case study point of view).
- Validation on the performed on input transaction before inserting data into interface table
 - Supplier Validations reference table AP_SUPPLIERS
 - Supplier Site Validations reference table AP_SUPPLIER_SITES_ALL
 - Operating Unit Validation reference table HR_OPERATING_UNITS or take the value from Profile FND_PROFILE.VALUE('ORG_ID')
 - Validate Ship to Organization reference table MTL_PARAMETERS
 - Validate items reference table MTL_SYSTEM_ITEMS_B
 - Validate units of measure reference table MTL_UNITS_OF_MEASURE.
 - Validate currency reference table FND_CURRENCIES
 - Validate Buyer reference table PER_ALL_PEOPLE_F

- Write suitable scripts to load into staging table and use PL/SQL procedure to perform pre validate of records and update flag of staging table in case of error record.

Hints for Error Handling: Sample Code

-- Validating Agent Name

```
BEGIN

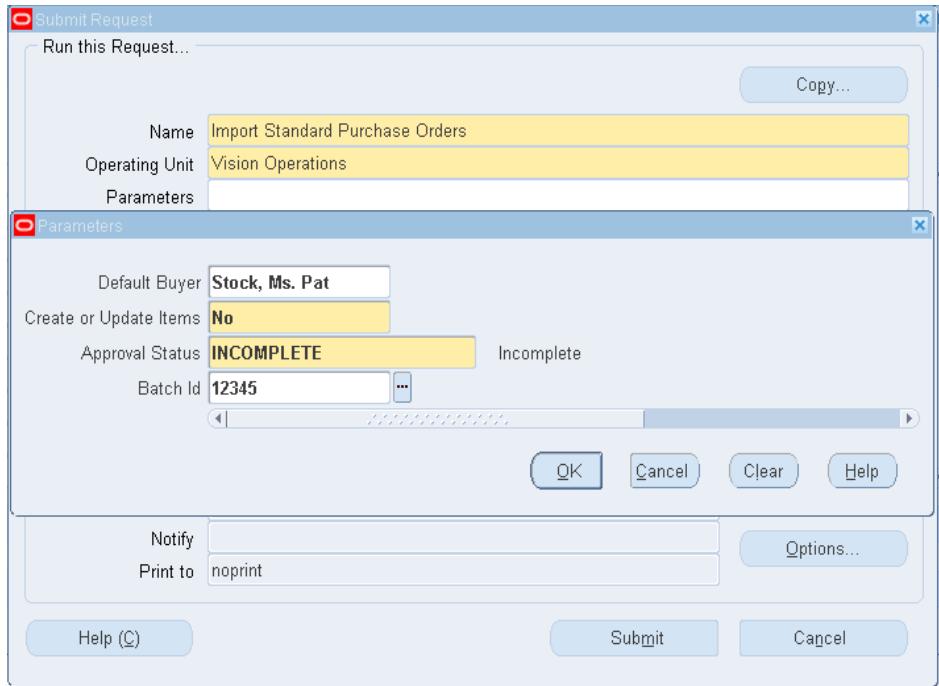
    SELECT PAPF.person_id
    INTO  l_agent_id
    FROM  per_all_people_f PAPF
    WHERE employee_number = <Value from Cursor>
    AND   SYSDATE BETWEEN EFFECTIVE_START_DATE AND
    EFFECTIVE_END_DATE
    AND   PAPF.BUSINESS_GROUP_ID = (select
    BUSINESS_GROUP_ID
    from hr_operating_units
    where organization_id = gn_org_id);
    EXCEPTION WHEN NO_DATA_FOUND THEN
    FND_FILE.PUT_LINE(FND_FILE.LOG,'Agent Number does
    not exist'||<from Cursor>);
    l_error_flag :=1;
```

Note: Exception Records update l_error_flag to “1” or insert invalids records in error table.

- Hint : Use concurrent program to move data into Base application tables.

Call the “[Import Standard Purchase Orders](#)” Program with the following Parameters

- a. Default Buyer: Stock, Ms. Pat (For Operations User)
- b. Create or Update Items: No
- c. Approval Status: INCOMPLETE
- d. Batch Id: Query the staging table for last run and pass it here



4. Unit test:

Appropriate unit testing of has to be performed. The technical specification section with has to be updated.

Program	Condition	Condition Description	Expected /ActualResults
Sql Loader test			
XX<EMPID>: Purchase Order Interface Loader Program	Run With Parameter: Data File Name	1. Data File Name : <Name of Flat File> 2. Data File Should be placed in \$CUSTOM_TOP/bin 3. Control File should be placed in \$CUSTOM_TOP/bin	<All records from flat file should be inserted into staging table>
XX<EMPID>: Purchase Order Interface Loader Program	No Paramaeter	Login TO EBS 1. Navigate to Purchasing Responsibility 2. Navigate to Requests 3. Select Request " XX<EMPID>: Purchase Order Interface Loader Program " 4. Parameters : NA 5. Submit 6. View Output	<Records should be inserted into Open Interface PO_HEADERS_INTERFACE and PO_LINES_INTERFACE. And see the View ouput for any validation failure>

Program	Condition	Condition Description	Expected /ActualResults
Standard import test			
Import Standard Purchase Orders	1	Login TO EBS 1. Navigate to Purchasing Responsibility 2. Navigate to Requests 3. Select Request " Import Standard Purchase Order " 4. Parameters : <ul style="list-style-type: none"> • Default Buyer : Default Value • Create or Update Items : No • Approval Status : Incomplete • Batch ID : <Select from LOV> 5. Submit 6. View Output	<Records should be transferred from interface table to base table PO_Headers_all,PO_LINES_ALL,PO-LINE_LOCATIONS_ALL and PO_DISTRIBUTIONS_ALL. In case of error see open Interface error table "PO_INTERFACE_ERRORS". Correct those records and rerun the standard Purchase Order Programs.>
Modify transaction in flat file to test above		Re-submit above program	The developed program should log and provided appropriate output for error correction.

5. Schedule:

This case study is expected to be performed in five days. The estimated schedule is as follows:

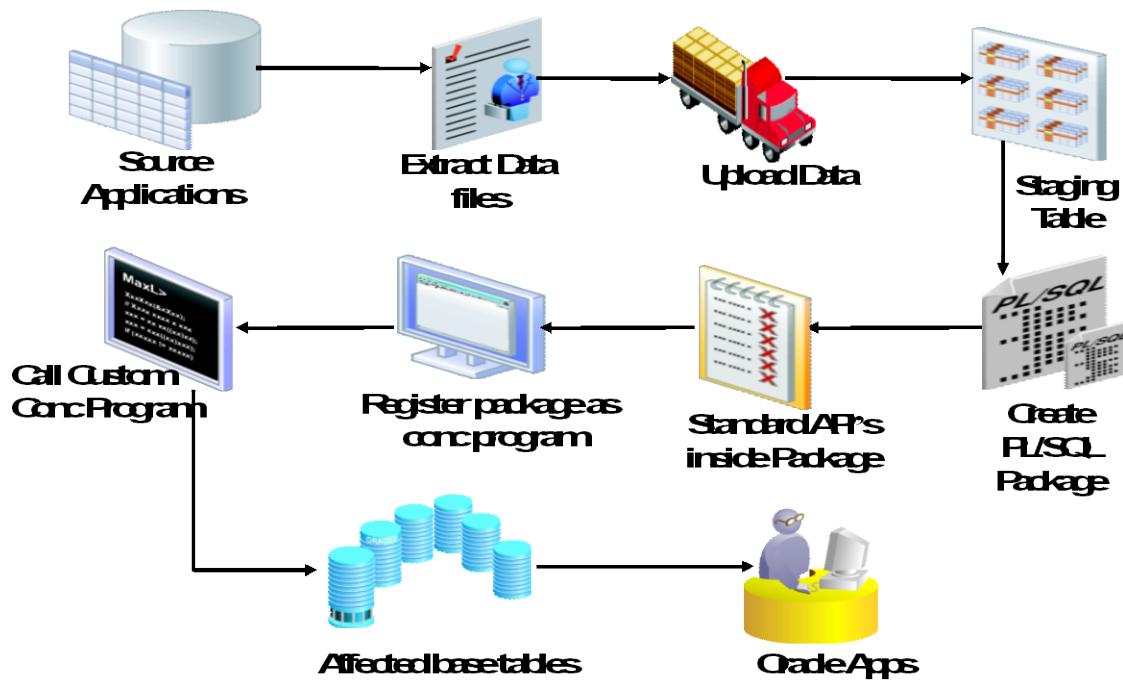
- **Design , develop & update of Technical Specification for the Interface Purchase Order(Inbound Interface) – Duration 3 days**
 - Conversion of Purchase Order from Flat File to Oracle Apps.
- **Unit test – Duration 1 day.**
- **Evaluation – Duration 1 day.**

CASE STUDY

1. R12 Application Programming Interface.....Duration 5 days.

The Workshop – The application programming interface (API) takes the student to evaluate the knowledge on

- Understanding the usage of Oracle EBS Public API and it's implementation.
- Identification and understanding the interface process using API.
- Understanding the process of using SQL loader technique.
- Understanding the validation technique towards inbound API transactions.
- Understanding the implementation of file processing.
- Understanding the API debugging process and generation of error log file.
- Analysing the error log and implement the required correction technique.



1. Business Scenario:

Vision Corporation Australia is planning to rollout Oracle EBS at its headquarters.

Vision Corporation has branches at various locations across the country. The branch offices have their own legacy application of capturing sales order. As branch offices are located even at remote location without proper internet bandwidth, the branch office team is unable to access Oracle EBS for entry of sales order details on daily basis.

Hence, it was decided by senior management that branch office will ftp a flat file to headquarters which would then be interfaced through API to Oracle EBS. The file containing the sales orders information is sent to Headquarters at end of the day.

It is assumed Items and customer information are set up manually at the headquarters and only the sales order along with relevant details are sent to Headquarters towards interfacing the sales order to Oracle EBS.

As a technical developer you are expected design a solution which describes the approach, create & identify the appropriate table, and develop scripts which use the appropriate API, register in Oracle EBS environment and test the solution.

2. Problem Statement (Order Import API):

- Creation of technical specification.
- Understand & create the flat file requirement towards API interface.
- Create relevant staging custom stage table.
- Understand & create SQL loader program.
- Create scripts / plsql packages to validate record and call relevant API for integration.
- Understand and register concurrent request /set towards integration.
- Create appropriate message output / logging and error correction procedures.
- Create appropriate scripts, packages and registration process to test the solution

Hint : Problem Statement/ Technical Document (Order Import API):

- Creation of technical specification.
- Identify the technical objects and design the information flow.
- Understand & create the flat file requirement towards API interface.
- Create the flat file to load data for order header and line tables. The detail file structures have given in flat file section.
- Create relevant staging custom stage table.
- There will be at least two custom staging tables one for order header and another for order line.
- The structure of order header table will be like oe_order_pub.header_rec_type.
- XXCUSTOM_ORDER_HEADERS_STG
For Populating Header Information



XXCUSTOM_ORDER_HEADERS_STG.tbl

- The structure of order line table will be like oe_order_pub.line_tbl_type.
XXCUSTOM_ORDER_LINES_STG
For Populating Lines Information
-
- XXCUSTOM_ORDER_LINES_STG.tbl

 - Understand & create SQL loader program.
 - Write sql loader program .ctl file to load order header and line records from flat file to staging tables.
 - Create scripts / PL/SQL packages to validate record and call relevant API for integration.
 - a. Prepare the PL/SQL store program with pre validation for input records. The following pre validation can be considered.
 - The customer, ship to and bill to address validation with reference of HZ_LOCATIONS, HZ_PARTY_SITES, HZ_CUST_ACCT_SITES_ALL, HZ_CUST_SITESUSES_ALL tables.
 - The operation unit validation (org_id) with reference of HR_OPERATIONNG_UNITS.
 - The ship to, invoice to organization supplied with reference of MTL_PARAMETERS.
 - The order type and line type validation with reference of mtl_parameters table.

- Validate the item with respect to master and shipping Org with reference with MTL_SYSTEM_ITEMS_B and MTL_PARAMETERS tables.
- The oe_order_pub.process_order API can be used for this purpose.



oe_order_pub.process_order.API

- Understand and register concurrent request /set towards integration.
- Create appropriate message output / logging and error correction procedures.
- In case if any error occurred, go through the log details and check the error message. If it reflects validation error search the same in pl/sql package to find out exact error location verify the sql query to find out the root cause of error.
- Once cause of error derived reprocess the flat file after deleting the error data from staging table.

2.1 Flat File mapping with staging table:

Order header flat file:

DataFileColumn Name	Column Description	Data type	Staging Table Columns
ORIG_SYS_DOCUMENT_REF	Original system document reference	VARCHAR2(50)	ORIG_SYS_DOCUMENT_REF
ORDER_CATEGORY	Order category	VARCHAR2(30)	ORDER_CATEGORY
ORDER_TYPE	Order type	VARCHAR2(40)	ORDER_TYPE
SOLD_TO_ORG_ID	Sold to org id represents the end customer.	Number	SOLD_TO_ORG_ID
SHIP_TO_ORG_ID	Ship to org id represents the ship to .	Number	SHIP_TO_ORG_ID
INVOICE_TO_ORG_ID	Invoice to org id represents the Bill to.	Number	INVOICE_TO_ORG_ID
SHIP_FROM_ORG_ID	Ship from org id represents the warehouse	Number	SHIP_FROM_ORG_ID
ORG_ID	Org id represents the operating unit.	Number	ORG_ID
ORDERED_DATE	Ordered date	Date	ORDERED_DATE

The sample data file is attached here.



Double click to open in notepad.

Order line flat file:

Column Name	Column Description	Data type	Column Name
ORIG_SYS_DOCUMENT_REF	Original system document reference.	VARCHAR2(50)	ORIG_SYS_DOCUMENT_REF
ORDERED_ITEM	Inventory Item	VARCHAR2(2000)	ORDERED_ITEM
LINE_CATEGORY_CODE	Category code for order line	VARCHAR2(30)	LINE_CATEGORY_CODE
ORDERED_QUANTITY	Quantity of item ordered.	Number	ORDERED_QUANTITY
ORDER_QUANTITY_UOM	Unit of measure of ordered item.	Varchar2(3)	ORDER_QUANTITY_UOM
LINE_TYPE	Line type.	Varchar2(240)	LINE_TYPE

The sample data file is attached here.



orderlineimp.dat

Double click to open in notepad.

3 Unit Test cases.

(Note - Perform unit test and provide the details in technical specification documents)

Unit Test Steps	Condition	Condition Description	Expected/Actual result
1. Design the control files to load data into sales order header and line tables.		1Data File Name: <Flat File Name> 1. Data File Should be placed in \$CUSTOM_TOP/bin 2. Control File should be placed in \$CUSTOM_TOP/bin	<All record from flat file should be inserted into corresponding tables>
2. Run the concurrent Program request Set XXCustom Order Import Process from OM responsibility and verify the status.		1. Login to ebs 2. Navigate to Order Management Responsibility. 3. Select Request set and submit the requestset XXCustomOrder Import program 4. Parameters: NA 5. Submit 6. View output	If concurrent program request set completed successfully verify the order number from log and query the same in order management module.
3. Modify the data file with some wrong value in data field and run the concurrent program request set XXCustom Order Import Process. Verify the log for validation error.		Re-submit concurrent program.	The developed program should log and provide appropriate output for error correction.

4 Schedule:

This case study is expected to be completed perform in 5 days. The details about the estimate have given below.

- **Design, develop and update technical specification – Duration 3 days.**
 - Create associated sales order data similar to flat file load into staging table.
 - Create / develop program and process the imported records through OM API to import it into OM base table.
 - Create technical specification.
- **Testing and error fix – Duration 1 day.**
 - Create associated item transaction similar to flat file and load into staging table.
 - Create / develop programs to process the Items records through Item API and import to EBS inventory table.
- **Evaluation – Duration 1 Day.**



IBM GH Program

Technical Specification For Extension

<Some Name>

Version 1.0

Creation Date

Last Update Date

Last Print Date



Contents

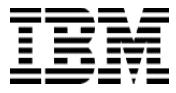
Document Control	3
Change Record	3
Object Information	4
Technical Overview.....	5
Overview	5
Description	5
Approach	5
Assumptions	5
Module list	5
Document References	6
Technical Specification.....	7
Program Flow	7
PL/SQL Program Design	8
Program Elements	8
Program Flow Diagram	9
Pseudo-Code	10
Error Handling	10
Table and View Usage	10
Application Set up	11
New/Updated Seed Data	11
Descriptive Flexfield	11
Value Sets	11
Concurrent Program Executable	11
Concurrent Program Definition	11
Database Design	13
Unit Test Conditions	14
Unit Test Plan	14
Pre-Requisites	14
Submission/Launching	14
Post-Requisites	14
Performance Considerations.....	15
Volume Consideration	15
Installation Requirements.....	16
Issue Log.....	17
Open Issues	17
Closed Issues	17



Document Control

Change Record

Date	Author	Version	Change Reference



Object Information

Project Name	IBM GH Induction Training
Build Name	<Some Meaningful Name>
Functional Spec Reference	<Reference to the Case Study >
Primary RICE Group	Extension
Functional Area	< As per Requirement >
Technology Area	< As per Requirement >

Technical Overview

Overview

< Give a brief overview of the functional requirement >

Description

Approach

< The Case Study (Case Study name)> will cover the following:

- Step 1
- Step 2
- ..
- Step n

Assumptions

The following are the <Case Study Name > assumptions:

- < Assumptions if any >

Module list

Table

1. List All Custom Table names here

PL/SQL

1. List all PL/SQL package/procedure names here

Document References

< List the name of the FS provided >

Document	Document Link / Reference	Version

Technical Specification

Program Flow

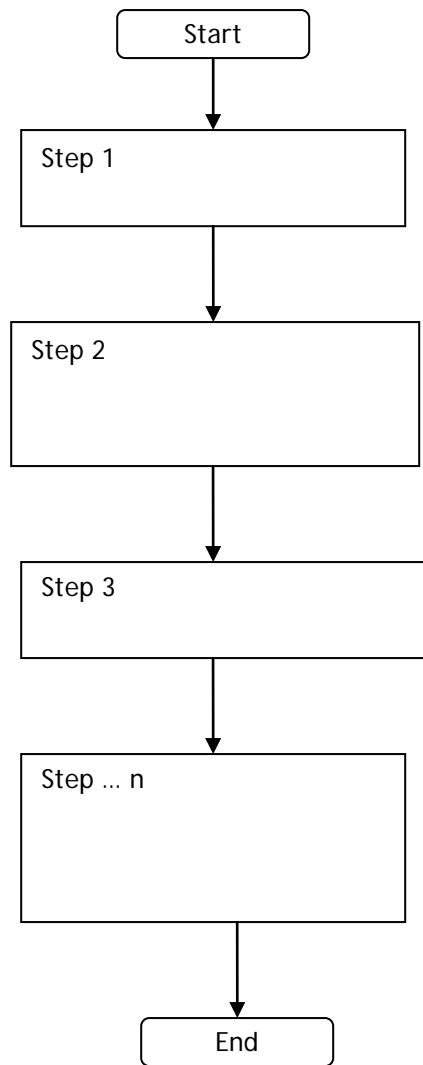
< Provide a BLOCK DIAGRAM of the proposed solution >

PL/SQL Program Design

Program Elements

Script Name / Package / Procedure / Function Name		Description
Name	Type	
< Name of Package/ Procedure >	Type (package/Procedure/Function etc)	Brief Description of what this will do

Program Flow Diagram



Pseudo-Code

Example:

```
Begin Program;  
Read Data file;  
Insert Into Staging Table;
```

```
..  
..  
..
```

```
Begin Validation Program  
    < Menation Validation Steps >  
End Validation;  
Begin Insert Routine  
    < Logic to Insert validated data to Interface Table >  
End Insert Routine  
Call Interface Program  
    < Insert logic to call the API/ Open Interface Program>  
End Routine
```

Error Handling

< Enter Appropriate error handling logic here >

Table and View Usage

Name	Type	Is New?	Functional Area	SELECT	INSERT	UPDATE	DELETE

Application Set up

New/Updated Seed Data

Lookup Type	Application	Code	Meaning	Access Level

Descriptive Flexfield

N/A

Application			Title				
Form Name			Base Table				
Segment Name	Window Prompt	Column	Value Set	Displayed	Enabled		

Value Sets

Existing seeded value sets are used.

Value Set Name	Description	List Type	Security Type	Format Type	Max Size	Precision	Numbers Only	Uppercase Only	Right Justify	Min Value	Max Value	Validation Type

Concurrent Program Executable

Executable	Short Name	Application	Description	Execution Method	Execution File Name

Concurrent Program Definition

Program Definition

Program	Short Name	Application	Description	Executable Name	Method	Output Format



Program	Short Name	Application	Description	Executable Name	Method	Output Format

Incompatibilities

Application	Name	Scope	Type

Parameters – Display

Seq	Parameter	Description	Enabled	Display Size	Concatenated Description	Prompt	Token

Parameters – Validation

Seq	Value Set	Default Type	Default Value	Range	Req.	Enable Security

Concurrent Request Group

Group	Application	Code	Description	Request Type	Name	Application

Database Design

<Put the Staging table design here >

Unit Test Conditions

Unit Test Plan

Program	Condition	Condition Description	Expected Results

Pre-Requisites

FYI

Provide information on any pre requisite functional or technical setups in this section.

Submission/Launching

FYI

Provide information on different submission methods possible for this object. (eg: SRS, Workflow, form, any other object reference etc.)

Post-Requisites

FYI

Provide information on any objects integrated with this build, which needs to run after this build.

Performance Considerations

For the time being cost very low, no any full table scan are there.

Volume Consideration

TBD

Installation Requirements

FYI	Provide information regarding the installation procedure in this section. The section should detail out the installation steps if any, to be performed, any special procedures to be performed or provide reference to promotion packs/installation documents etc.
-----	--

Issue Log

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date