**Table of Contents**

[1 Overview 2](#_Toc515545094)

[1.1 Introduction 2](#_Toc515545095)

[1.2 Functional Overview 2](#_Toc515545096)

[1.3 Approach 2](#_Toc515545097)

[1.4 Prerequisites and Set-up Dependencies 3](#_Toc515545098)

[1.5 Assumptions 3](#_Toc515545099)

[1.6 Design Summary 3](#_Toc515545100)

[1.7 Process Flow 4](#_Toc515545101)

[2 Report Details 4](#_Toc515545102)

[2.1 Output Type and Destination 4](#_Toc515545103)

[2.2 Initiation Mode and Schedule 4](#_Toc515545104)

[2.3 Multi-Language Support 5](#_Toc515545105)

[2.4 Table and View Usage 6](#_Toc515545106)

[2.5 Field Definition / Mapping 7](#_Toc515545107)

[2.6 Subtotal by/ Total by/ Sort by 10](#_Toc515545108)

[2.7 Conditions / Where Clause 10](#_Toc515545109)

[2.8 Report Layout 10](#_Toc515545110)

[2.9 Program Logic (pseudo code) 10](#_Toc515545111)

[2.10 SQL Statements 13](#_Toc515545112)

[2.11 Additional Database Objects 61](#_Toc515545114)

[2.12 Log Output 61](#_Toc515545115)

[2.13 Installation Steps 61](#_Toc515545116)

[2.14 Registration 62](#_Toc515545117)

[2.16 Incompatibilities 148](#_Toc515545125)

[2.17 Performance Considerations 148](#_Toc515545126)

[2.18 Restart/Recovery Procedures 148](#_Toc515545127)

[3 BI Publisher Approach 192](#_Toc515545141)

[4 Unit /Link Test Scenario 193](#_Toc515545142)

[5 Open/Closed Issues 201](#_Toc515545143)

[5.1 Open Issues 201](#_Toc515545144)

[5.2 Closed Issues 201](#_Toc515545145)

**Note:** Embedded objects will not work within the PDF version of this document. Please contact IT Document Coordination for a Word version of this document.

# Overview

## Introduction

This document provides the technical specifications required to implement the Gilead Commercial Invoice report. It has been built to the intention of the technical resources that are involved in the development of this report.

This Technical Design document complements the Functional Design document for the Gilead Commercial Invoice report and this set should be considered being the complete detailed design.

## Functional Overview

This report will produce valid documents that will reflect accurately transactions made with customers, supporting multiple languages, in country specific formats, fulfilling legal, statutory and local requirements, and providing complete information about product and services shipped/billed to customers in order to facilitate collections in a timely manner.

It is also required to have the ability to reprint customer invoices

This document includes:

* The definition of specific terms that the reader must know to better understand the document
* The assumptions on which the technical design of this report is built
* Concurrent program definitions

## Approach

* Customize RAXINV.rdf(Print Selected invoices report) to fit Gilead commercial invoice print requirements. Rename RDF it to XXGILARCOMINV.rdf
* Create custom XML templates for each country for the report based on the Required Layout in Functional Document. The template names are <COUNTRY>.rtf
* There will be two concurrent programs. The “Gilead Print Commercial Invoices(Main)” is a parent program, this report accepts all necessary parameters including XML output template. The “Gilead Print Commercial Invoices” is child program kicked off from parent program to display actual XML output.
* Register concurrent executables with Executable method as ‘Oracle Reports’.
* Register concurrent programs with output format as XML.
* Create XML Data Definition from XML Publisher Administrator Responsibility.
* Create country specific “.rtf” XML Template and upload the RTF file to the Template Definition.
* Add the concurrent program to “Print Invoices” Request Group.
* Submit the Request " Gilead Print Commercial Invoices(Main)” from the Standard Print invoices submission form.
* The report will be available in Oracle Receivables and report group is Print Invoices. The report can be printed in PDF format only

Japan Changes

* Add Japan rollout changes to XXGILARCOMINV.rdf
* Add the concurrent program “GIL Print Commercial Invoices-Standard(Print)” to “GIL\_AR\_1\_AR INVOICES” Request Group.

Create new template definition for Japan “ Japan Invoice” and attach the Japan specific RTF File “Japan.rtf”

Started changes for v16.0

Following changes have been done as part of the R12.2.6 upgrade process:

* Any schema reference (other than APPS) on a table, sequence, view will be removed from the code.
* For any custom table residing in custom schema, if there were no Editioned view created, then ad\_zd\_table.upgrade function will be used to create Editioned view.
* In XXGIL\_EMAIL\_COMM\_INVOICE\_PKG.send\_mail\_with\_invoice procedure added language and application\_id joins to one of the queries, which fetches the email details to make the query to run quickly.
* Changes are made to the RTF file to fix the Page-break issues encountered in 12.2.6. To fix the same table structure of the RTF file was adjusted and font size of certain BIP XML code snippets (end body tag) was reduced to 1 point. Also the lines section bottom border and footer section top border are made visible to make the table look complete.

Ended changes for v16.0

## Prerequisites and Set-up Dependencies

1. XXGIL\_OTC\_TRX\_TYPE\_TRANSLATION has to be loaded. This lookup has country specific translated transaction type mappings in local languages.
2. Remit to bank information will be stored in DFF at Customer account level for each customer. The report derives this value and display on the report.

The HZ\_CUST\_ACCOUNTS.ATTRIBUTE2 is storing this value. For this purpose, BI team enables DFF on Customer account level, for validation uses XXGIL\_CM\_CUST\_REMIT\_TO\_BANK value set. This is mandatory setup.(As per CR)

Japan changes:

1. Lookup “XXGIL\_MLS\_PRODUCT\_DESCRIPTIONS” is used to store Product Descriptions in Local language.
2. Lookup “XXGIL\_OTC\_JDNET\_TRX\_TYPES” is used for transaction code derivation based on Invoice class, Order type and extended amount.
3. Lookup “XXGIL\_OTC\_COMM\_MANUAL\_ITEM” is used to derive items for Commercial Orders Manual Transactions.
4. Lookup “XXGIL\_MLS\_UOM\_CODE” is used to derive UOM code in local language.
5. The DFF “Site Use Information” is used to store JD NET code details at Site Use level.

Lookup “XXGIL\_OTC\_JP\_JDNET\_CODES” is used to derive the Maker code.

## Assumptions

Input Parameters: Following are input parameters.

## Design Summary

|  |  |
| --- | --- |
| **Document Type** | **Document Name** |
|  |  |
| Functional Design | ROTC85 AR Commercial Invoice 181109 v4.4.doc and ITD-2446 MD050 Commercial Invoice -SEPA |
| Technical Design | This Document. |
| Unit Test Plan | ROTC85 UTP AR Commercial Invoice.doc |

## Process Flow

START

Run **Gilead Print Commercial Invoices(Main)** report by entering parameters in Oracle Apps/EBS in specified responsibility

Above report attach xml template and internally calls Gilead Print Commercial Invoices. This report execute XXGILARCOMINV.rdf to generate output in XML format

Success

Error

Apply XXGILARCOMINV \_{Country}.rtf Country specific BI publisher template to xml output to display report output.

END

# Report Details

## Output Type and Destination

| **Attribute** | **Value** |
| --- | --- |
| Report Format | Landscape |
| Header Fields | Each page |
| Line Fields | Each page |
| Page Break | Happens at start of new transaction |
| Page Number should be reset after | For every new invoice number |

## Initiation Mode and Schedule

Run by user(s) on an as needed basis

## Multi-Language Support

| **Country** | **Headers** | **Data Language** |
| --- | --- | --- |
| **Denmark** |  |  |
| **Holland** |  |  |
| **Belgium** |  |  |
| **UK** | **English** | **English** |
| **Ireland** |  |  |
| **Australia/ New Zealand** | **English** | **English** |
| **USA** | **English** | **English** |
| **Canada** |  |  |
| **Finland** |  | **English** |
| **France** |  | **French** |
| **Switzerland** | **Local Language** | **German** |
| **Austria** | **Local Language** | **German** |
| **Germany** | **Local Language** | **German** |
| **Greece** | **Local Language** | **Greek** |
| **Iceland** |  | **English** |
| **Italy** | **Local Language** | **Italian** |
| **Norway** |  | **English** |
| **Portugal** | **Local Language (Independent Program. See section 2.8)** | **English** |
| **Spain** | **Local Language** | **Spanish** |
| **Sweden** |  | **English** |
| **Turkey** | **Local Language** | **Turkish** |
| **Japan** | Local Language | Japanese |

## Table and View Usage

| **Table Name** | **Select** | **Insert** | **Update** | **Delete** |
| --- | --- | --- | --- | --- |
| RA\_CUSTOMER\_TRX | X |  |  |  |
| PO\_CUSTOMER\_TRX\_LINES | X |  |  |  |
| RA\_CUST\_TRX\_LINE\_GL\_DIST | X |  |  |  |
| HZ\_CUST\_ACCOUNTS | X |  |  |  |
| RA\_TERMS | X |  |  |  |
| RA\_TERMS\_LINES | X |  |  |  |
| RA\_CUST\_TRX\_TYPES | X |  |  |  |
| AR\_LOOKUPS | X |  |  |  |
| HZ\_PARTIES | X |  |  |  |
| HZ\_CUST\_ACCT\_SITES | X |  |  |  |
| HZ\_PARTY\_SITES | X |  |  |  |
| HZ\_LOCATIONS | X |  |  |  |
| HZ\_CUST\_SITE\_USES | X |  |  |  |
| PA\_LINES\_ALL | X |  |  |  |
| AR\_ADJUSTMENTS | X |  |  |  |
| WSH\_NEW\_DELIVERIES | X |  |  |  |
| WSH\_DELIVERY\_ASSIGNMENTS | X |  |  |  |
| WSH\_DELIVERY\_DETAILS | X |  |  |  |
| MTL\_LOT\_NUMBERS | X |  |  |  |
| XLE\_REGISTRATIONS | X |  |  |  |
| XLE\_ENTITY\_PROFILES | X |  |  |  |
| HR\_LOCATIONS\_ALL | X |  |  |  |
| HZ\_ORGANIZATION\_PROFILES | X |  |  |  |
| MTL\_UNITS\_OF\_MEASURE | X |  |  |  |
| MTL\_SYSTEM\_ITEMS | X |  |  |  |
| ZX\_LINES\_V | X |  |  |  |
| For Japan : | | | | |
| MTL\_CROSS\_REFERENCES\_B | X | NA | NA | NA |
| MTL\_CROSS\_REFERENCES\_TL | X | NA | NA | NA |
| CR1936 – Kite Drop Ship Orders | | | | |
| OE\_DROP\_SHIP\_SOURCES | X |  |  |  |
| RCV\_TRANSACTIONS | X |  |  |  |
| RCV\_LOT\_TRANSACTIONS | X |  |  |  |

## Field Definition / Mapping

| **Field Name** | **Description / Remarks** | **Size** | **Std. Report** | **Oracle Table / Std. Report Field** | **Oracle Column** |
| --- | --- | --- | --- | --- | --- |
| For Japan: | | | | | |
| Order Date  受注日 | Sales Order Ordered Date  Or  Transaction Date | VARCHAR2(30) | OE\_ORDER\_HEADERS\_ALL/  RA\_CUSTOMER\_TRX\_ALL | ORDRED\_DATE/TRX\_DATE |  |
| TXN Code  取引区分 | Transaction code | VARCHAR2(240) | FND\_LOOKUP\_VALUES  “XXGIL\_OTC\_JDNET\_TRX\_TYPES” | ATTRIBUTE2 |  |
| Shikiri-Sho No.  仕切書 No. | Delivery Name/Transaction Number | VARCHAR2(100) | RA\_CUSTOMER\_TRX\_ALL | INTERFACE\_HEADER\_ATTRIBUTE3/TRX\_NUMBER |  |
| Wholesaler code  取引先コード | Wholesaler code for bill to site | VARCHAR2(50) | HZ\_CUST\_SITE\_USES\_ALL | ATTRIBUTE6 |  |
| Maker Code  メーカーコード | Maker code | VARCHAR2(100) | FND\_LOOKUP\_VALUES  “XXGIL\_OTC\_JP\_JDNET\_CODES ; DESCRIPTION : MAKERCODE ” | LOOKUP\_CODE |  |
| Consolidated Inv. No.  請求書 No. | Consolidated Invoice Number for the transaction | VARCHAR2(30) | AR\_CONS\_INV\_ALL | CONS\_BILLING\_NUMBER |  |
| Bill to Site Name | Party Site name for Bill to location | VARCHAR2(2000) | HZ\_PARTY\_SITES | PARTY\_SITE\_NAME |  |
| Ship to Site Name | Party Site name for Ship to location | VARCHAR2(2000) | HZ\_PARTY\_SITES | PARTY\_SITE\_NAME |  |
| Unified Item Code  製品番号 | Item cross reference derived for JDNET  OR  Item derivation for the transaction line for manual invoices | VARCHAR2(240) | MTL\_CROSS\_REFERENCES\_B  /  FND\_LOOKUP\_VALUES  “XXGIL\_OTC\_COMM\_MANUAL\_ITEM” | CROSS\_REFERENCE  /  DESCRIPTION |  |
| Lot Number  ロット番号 | Lot number derived for delivery line or the manual transaction line | VARCHAR2(100) | WSH\_DELIVERY\_DETAILS  /  RA\_CUSTOMER\_TRX\_LINES\_ALL | LOT\_NUMBER  /  ATTRIBUTE1 |  |
| Lot Expiration Date  ロット期限 | Lot expiration date for the lot number derived for delivery line or the manual transaction line | VARCHAR2(30) | MTL\_LOT\_NUMBERS | EXPIRATION\_DATE |  |
| Product Name  品名 | Product Name in Kanji – Japanese | VARCHAR2(2000) | FND\_LOOKUP\_VALUES  “XXGIL\_MLS\_PRODUCT\_DESCRIPTIONS” , “XXGIL\_OTC\_COMM\_MANUAL\_ITEM”  /  RA\_CUSTOMER\_TRX\_LINES\_ALL | ATTRIBUTE1  /  DESCRIPTION |  |
| UOM Code  単位 | UOM code for item derived from above for the delivery line/transaction line | VARCHAR2(240) | FND\_LOOKUP\_VALUES  “XXGIL\_MLS\_UOM\_CODE” | ATTRIBUTE1 |  |

## Subtotal by/ Total by/ Sort by

| **Operation** | **Field Name** | **Level** |
| --- | --- | --- |
| For Japan: | | |
| Order By | LINE\_NUMBER | ASC |
| Subtotal by | TAX\_AMOUNT | NA |
| Subtotal by | TOTAL\_AMOUNT | NA |

## Conditions / Where Clause

| **Table Name** | **Column Name** | **Relationship** | **Criteria** |
| --- | --- | --- | --- |
| NA |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Report Layout

** ******

## Program Logic (pseudo code)

**Program1: Gilead Print Commercial Invoices(Main)**

This program executes XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG.RUN package procedure

Begin

--Derive Language and Territory of Print template from XDO\_LOBS table

--Derive the current concurrent request printer name, responsibility and responsibility applications

--Initialize applications using responsibility and responsibility applications values

--Apply xml template and call **Gilead Print Commercial Invoices** concurrent program to execute and print the output.

lv\_conc\_request\_id := Fnd\_Request.SUBMIT\_REQUEST(

application => 'XXGIL',

program => 'XXGILARCOMINV',

description => NULL,

start\_time => TO\_CHAR(SYSDATE, 'DD-MON-YY HH:MI:SS'),

sub\_request => FALSE,

argument1 => P\_ORDER\_BY ,

argument2 => P\_choice ,

argument3 => p\_cust\_trx\_class,

argument4 => P\_cust\_trx\_type\_id,

argument5 => P\_trx\_number\_low,

argument6 => P\_trx\_number\_high,

argument7 => P\_dates\_low,

argument8 => P\_dates\_high,

argument9 => p\_sales\_order\_low ,

argument10 => p\_sales\_order\_high,

argument11 => P\_DELIVERY\_NUM\_LOW,

argument12 => P\_DELIVERY\_NUM\_HIGH ,

argument13 => P\_CUST\_PO\_NUM\_LOW,

argument14 => P\_CUST\_PO\_NUM\_HIGH,

argument15 => P\_customer\_class\_code ,

argument16 => P\_customer\_id,

argument17 => p\_customer\_num\_id,

-- argument18 => P\_installment\_number ,

argument18 => P\_open\_invoice,

argument19 => p\_check\_for\_taxyn,

argument20 => p\_tax\_registration\_number,

argument21 => p\_header\_pages,

argument22 => p\_debug\_flag,

argument23 => p\_message\_level--,

--argument25 => p\_random\_invoices\_flag ,

--argument26 => p\_invoice\_list\_string

);

--Wait for the request to complete

--After child request completed end the program.

End;

**Program2: Gilead Print Commercial Invoices**

BEGIN

Get Concurrent program parameters

In After Parameter Trigger

--

AfterParam\_Procs.Get\_Country\_Details;

AfterParam\_Procs.Switch\_On\_Debug;

AfterParam\_Procs.Get\_Trx\_Number\_Low;

AfterParam\_Procs.Get\_Trx\_Number\_High;

AfterParam\_Procs.Get\_Tax\_Option;

AfterParam\_Procs.Set\_lp\_trx\_date\_clause;

AfterParam\_Procs.Set\_lp\_doc\_seq\_clause;

Build\_Where\_Clause;

-- bug3847555 create query

Build\_Query ;

In Before Report Trigger

-- Build lexical parameters

BeforeReport\_Procs.Populate\_Printing\_Option;

BeforeReport\_Procs.Populate\_Tax\_Printing\_Option;

Get\_Message\_Details;

Get\_Org\_Profile;

BUILD\_WHERE\_CLAUSE;

--CALL

-- Assign input parameters to Report level place holder’s columns

-- Fetch Concurrent program name

Write Report Header

Select rows using input parameters (see SQL statement 1)

FOR EACH row returned

Select last receipt amount details for the PO’s selected in Query1 (see SQL statement 2)

Build output record(s) (see data sources)

Write row(s) to output;

ENDFOR

END

## SQL Statements

### Data Selection

a) **Q\_DELIVER\_TO\_CUSTOMER** query has been created.

SELECT party.party\_name DELIVER\_CUST\_NAME

, loc.ADDRESS1 DELIVER\_ADDRESS1

, loc.ADDRESS2 DELIVER\_ADDRESS2

, loc.ADDRESS3 DELIVER\_ADDRESS3

, loc.ADDRESS4 DELIVER\_ADDRESS4

, loc.CITY DELIVER\_CITY

, NVL(loc.STATE , loc.PROVINCE) DELIVER\_STATE

, loc.POSTAL\_CODE DELIVER\_POSTAL\_CODE

, loc.COUNTRY DELIVER\_COUNTRY

, U.SITE\_USE\_ID DELIVER\_SITE\_ID

, C.CUST\_account\_ID DELIVER\_CUSTOMER\_ID

, U.TAX\_REFERENCE DELIVER\_SITE\_TAX\_REFERENCE

, party.TAX\_REFERENCE DELIVER\_CUST\_TAX\_REFERENCE

, loc.STATE DELIVER\_TO\_STATE

, loc.PROVINCE DELIVER\_TO\_PROVINCE

FROM hz\_cust\_accounts C

, hz\_parties party

, hz\_cust\_acct\_sites A

, hz\_party\_sites party\_site

, hz\_locations loc

, hz\_cust\_SITE\_USES U

WHERE U.cust\_acct\_site\_ID = A.cust\_acct\_site\_ID

and a.party\_site\_id = party\_site.party\_site\_id

and loc.location\_id = party\_site.location\_id

AND U.SITE\_USE\_ID = :C\_DELIVER\_TO\_ORG\_ID

AND C.CUST\_account\_ID = A.CUST\_account\_ID

and c.party\_id = party.party\_id;

b) **Q\_LINES** query has been modified. Added new columns:

/\*Including Japan Changes\*/

SELECT \*

FROM ((SELECT

NULL delivery\_detail\_id /\* Added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; including delivery detail lines \*/

, c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, decode( c2.line\_number,

null, c.line\_number,

null) line\_number

, c.line\_type line\_type

, nvl(c.translated\_description,c.description) line\_item\_description

, nvl(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, nvl(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, NULL abs\_line\_qty\_inv /\* Added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; This field is only required for below Japan queries \*/

, u.unit\_of\_measure line\_uom

--gv added 01/03/10

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

--, c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id -- commented by Abhishek Gupta Incident 363639 052412

, DECODE(c.interface\_line\_context, 'ModelN', NULL, c.INTERFACE\_LINE\_ATTRIBUTE6) line\_so\_line\_id -- added by Abhishek Gupta Incident 363639 052412

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

/\*, NVL(c.unit\_standard\_price,0) \* nvl(c.quantity\_invoiced, c.quantity\_credited) line\_std\_extended\_amt

, ( NVL(c.unit\_standard\_price,0) - NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price)) line\_disc\_per\_unit\*/

, c.unit\_standard\_price line\_standard\_price

/\*end of gv\*/

, nvl (c.unit\_selling\_price, c.gross\_unit\_selling\_price) /\* Bug 2335596 \*/

line\_unit\_selling\_price

, nvl (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount /\* Bug 2335596 \*/

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, decode( to\_char(c2.line\_number),

null, 'N', 'Y') line\_is\_a\_child\_flag

, nvl(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, decode(c.line\_type,

'LINE', 0,

1) line\_child\_indicator,

nvl(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

/\* comment out for bug 745960 :

, decode(msi.item\_type,

'FRT', 'Z',

'A') line\_of\_type\_frt

\*/

, decode(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, decode(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 dummy

--gv added nvl 01/05/10

, nvl( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

/\* Begin of added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; changed position for order by to accomodate Union queries\*/

, decode( c2.line\_number,

null, decode( c.line\_type,

'LINE', c.line\_number\*10000+0,

'TAX', c.line\_number\*10000+8000,

100000000000),

decode( c2.line\_type,

'LINE', c2.line\_number\*10000+0,

'TAX', c2.line\_number\*10000+8000,

c2.line\_number\*10000+9000) +

decode( c.line\_type,

'LINE', 0,

'TAX', 8000,

'FREIGHT', 9000) +

c.line\_number ) order\_by\_field

/\* End of added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; changed position for order by to accomodate Union queries\*/

from ra\_customer\_trx\_lines c

, ra\_customer\_trx\_lines c2

, mtl\_units\_of\_measure u

, mtl\_system\_items msi

where c.customer\_trx\_id = :customer\_trx\_id

and c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

and c.uom\_code = u.uom\_code(+)

and c.inventory\_item\_id = msi.inventory\_item\_id (+)

and msi.organization\_id(+) = :so\_organization\_id

/\* Begin of added by Bharath Thampi on 28-Mar-2016 for Japanese reference ; for manual invoices previous query should be used \*/

AND :REPORT\_ORG\_COUNTRY <> 'JP'

)

/\* Begin of commented by Bharath Thampi on 07-Jun-2016 for Japanese reference ; changed position for order by to accomodate Union queries\*/

/\*order by

/\* comment out for bug 745960 :

decode(msi.item\_type,

'FRT', 'Z',

'A') ,

\*/

/\*decode( c2.line\_number,

null, decode( c.line\_type,

'LINE', c.line\_number\*10000+0,

'TAX', c.line\_number\*10000+8000,

100000000000),

decode( c2.line\_type,

'LINE', c2.line\_number\*10000+0,

'TAX', c2.line\_number\*10000+8000,

c2.line\_number\*10000+9000) +

decode( c.line\_type,

'LINE', 0,

'TAX', 8000,

'FREIGHT', 9000) +

c.line\_number )\*/

/\* End of commented by Bharath Thampi on 07-Jun-2016 for Japanese reference ; changed position for order by to accomodate Union queries\*/

/\* Begin of added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; Query to get manual invoices \*/

UNION

(SELECT NULL delivery\_detail\_id

, c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, decode( c2.line\_number,

null, c.line\_number,

null) line\_number

, c.line\_type line\_type

, nvl(c.translated\_description,c.description) line\_item\_description

, nvl(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, nvl(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, abs(nvl(c.quantity\_invoiced,c.quantity\_credited)) abs\_line\_qty\_inv

, u.unit\_of\_measure line\_uom

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

, DECODE(c.interface\_line\_context, 'ModelN', NULL, c.INTERFACE\_LINE\_ATTRIBUTE6) line\_so\_line\_id

--, DECODE(c.interface\_line\_context, 'INTERCOMPANY', c.INTERFACE\_LINE\_ATTRIBUTE7,-1) transaction\_id /\* Commented by Bharath Thampi on 07-Jun-2016 for Japanese reference ; This field is no longer required \*/

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

, c.unit\_standard\_price line\_standard\_price

, nvl (c.unit\_selling\_price, c.gross\_unit\_selling\_price)

line\_unit\_selling\_price

, nvl (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, decode( to\_char(c2.line\_number),

null, 'N', 'Y') line\_is\_a\_child\_flag

, nvl(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, decode(c.line\_type,

'LINE', 0,

1) line\_child\_indicator,

nvl(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

, decode(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, decode(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 dummy

, nvl( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

, decode( c2.line\_number,

null, decode( c.line\_type,

'LINE', c.line\_number\*10000+0,

'TAX', c.line\_number\*10000+8000,

100000000000),

decode( c2.line\_type,

'LINE', c2.line\_number\*10000+0,

'TAX', c2.line\_number\*10000+8000,

c2.line\_number\*10000+9000) +

decode( c.line\_type,

'LINE', 0,

'TAX', 8000,

'FREIGHT', 9000) +

c.line\_number )

from ra\_customer\_trx\_lines c

, ra\_customer\_trx\_lines c2

, mtl\_units\_of\_measure u

, (SELECT DISTINCT inventory\_item\_id,segment1,

description,item\_type

FROM mtl\_system\_items) msi

where c.customer\_trx\_id = :customer\_trx\_id

and c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

and c.uom\_code = u.uom\_code(+)

and c.inventory\_item\_id = msi.inventory\_item\_id (+)

--and c.interface\_line\_attribute10 = msi.organization\_id (+) /\*Commented by Bharath Thampi on 07-Jun-2016 item derivation\*/

AND :CF\_JP\_SHIKIRI\_SHO = :TRX\_NUMBER

AND :REPORT\_ORG\_COUNTRY = 'JP'

)

/\* End of added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; Query to get manual invoices \*/

UNION

(SELECT wdd.delivery\_detail\_id

, rctl.customer\_trx\_id line\_customer\_trx\_id

, rctl.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, ROW\_NUMBER() OVER (ORDER BY wdd.delivery\_detail\_id) line\_number

, 'LINE' line\_type

, msi.description line\_item\_description

, nvl(rctl.quantity\_ordered,

rctl.quantity\_invoiced) line\_qty\_ordered

, wdd.shipped\_quantity line\_qty\_invoiced

, ABS(wdd.shipped\_quantity) abs\_line\_qty\_inv

, u.unit\_of\_measure line\_uom

, :CF\_JP\_SHIKIRI\_SHO line\_delivery\_name

, rctl.interface\_line\_attribute6 line\_so\_line\_id

--, DECODE(rctl.interface\_line\_context, 'INTERCOMPANY', rctl.INTERFACE\_LINE\_ATTRIBUTE7,-1) transaction\_id /\* Commented by Bharath Thampi on 07-Jun-2016 for Japanese reference ; This field is no longer required \*/

, msi.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

, wdd.unit\_price line\_standard\_price

, oola.unit\_selling\_price line\_unit\_selling\_price

, (oola.unit\_selling\_price \* wdd.shipped\_quantity) line\_extended\_amount

, NULL line\_net\_amount

, TO\_CHAR(ooha.order\_number) line\_sales\_order

, ooha.ordered\_date line\_sales\_order\_date

, NULL line\_tax\_rate

, NULL line\_vat\_tax\_id

, NULL line\_tax\_exemption\_id

, NULL line\_location\_rate\_id

, NULL line\_tax\_precedence

, NULL line\_is\_a\_child\_flag

, NULL link\_to\_line

, NULL line\_child\_indicator,

NULL link\_to\_cust\_trx\_line\_id

, NULL line\_of\_type\_frt

, NULL order\_by1

, NULL dummy

, NULL line\_tax\_inclusive

, wdd.delivery\_detail\_id order\_by\_field

from mtl\_units\_of\_measure u

, mtl\_system\_items msi

, oe\_order\_lines oola

, oe\_order\_headers ooha

, ra\_customer\_trx\_lines rctl

, wsh\_new\_deliveries wnd

, wsh\_delivery\_details wdd

, wsh\_delivery\_assignments wda

where wnd.name = :CF\_JP\_SHIKIRI\_SHO

AND rctl.interface\_line\_attribute3 = :CF\_JP\_SHIKIRI\_SHO

AND rctl.customer\_trx\_id = :CUSTOMER\_TRX\_ID

AND rctl.interface\_line\_attribute3 = wnd.name

AND rctl.interface\_line\_attribute6 = TO\_CHAR(wdd.source\_line\_id)

AND wnd.delivery\_id = wda.delivery\_id

AND wdd.delivery\_detail\_id = wda.delivery\_detail\_id

AND wdd.source\_line\_id = oola.line\_id

AND ooha.header\_id = oola.header\_id

AND wdd.requested\_quantity\_uom = u.uom\_code

AND wdd.inventory\_item\_id = msi.inventory\_item\_id

AND msi.organization\_id = oola.ship\_from\_org\_id

AND :CF\_JP\_SHIKIRI\_SHO <> :TRX\_NUMBER

AND :REPORT\_ORG\_COUNTRY = 'JP')

/\* End of added by Bharath Thampi on 07-Jun-2016 for Japanese reference ; Query to get delivery detail lines \*/

) query

order by order\_by\_field

/\* End of modified by Bharath Thampi on 07-Jun-2016 for Japanese reference ; added new query for delivery detail lines \*/

**Package:** Following is XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG body.

create or replace PACKAGE BODY Xxgil\_Arp\_Trx\_Select\_Cont\_Pkg AS

/\* $Header: ARPLTSCB.pls 115.11 2002/11/15 02:43:54 anukumar ship $ \*/

/\*---------------------------------------------------------------------------+

| PUBLIC PROCEDURE |

| build\_where\_clause - Returns two where clauses depends on the |

| input parameters |

| |

| DESCRIPTION |

| See the algorithm follows |

| |

| REQUIRES |

| |

| EXCEPTIONS RAISED |

| ARP\_STANDARD.AR\_ERROR\_NUMBER ( in arp\_standard.fnd\_message ) |

| |

| KNOWN BUGS |

| |

| NOTES |

| |

| HISTORY |

| 12 May 93 Charles Huang Created |

| 19-FEB-99 Victoria Smith Added parameter p\_call\_from to identify |

| if procedure is called from MLS function |

| or invoice print program |

| 09-Nov-00 Debbie Jancis Modified for tca uptake. Removed all |

| references of ar/ra customer tables and |

| replaced with hz counterparts. |

| 03-JAN-10 Gopi Vemula Modified the package to include gilead requirements |

+---------------------------------------------------------------------------\*/

PROCEDURE build\_where\_clause (

P\_choice IN VARCHAR2,

P\_open\_invoice IN VARCHAR2,

P\_cust\_trx\_type\_id IN NUMBER,

p\_cust\_trx\_class IN VARCHAR2,

P\_installment\_number IN NUMBER,

P\_dates\_low IN DATE,

P\_dates\_high IN DATE,

P\_customer\_id IN NUMBER,

P\_customer\_class\_code IN VARCHAR2,

P\_trx\_number\_low IN VARCHAR2,

P\_trx\_number\_high IN VARCHAR2,

P\_batch\_id IN NUMBER,

P\_customer\_trx\_id IN NUMBER,

p\_adj\_number\_low IN VARCHAR2,

p\_adj\_number\_high IN VARCHAR2,

p\_adj\_dates\_low IN DATE,

p\_adj\_dates\_high IN DATE,

P\_where1 OUT NOCOPY VARCHAR2,

P\_where2 OUT NOCOPY VARCHAR2,

p\_table1 OUT NOCOPY VARCHAR2,

p\_table2 OUT NOCOPY VARCHAR2,

p\_call\_from IN VARCHAR2 DEFAULT 'INV'

) IS

/\*

/\*----------------------------------------------------------------------+

| Declare local variables |

+----------------------------------------------------------------------\*/

where1 VARCHAR2(8096);

where2 VARCHAR2(8096);

where3 VARCHAR2(8096);

table1 VARCHAR2(8096);

table2 VARCHAR2(8096);

cr CHAR(1);

dates\_low DATE;

dates\_high DATE;

adj\_dates\_low DATE;

adj\_dates\_high DATE;

adj\_number\_low VARCHAR2(80);

adj\_number\_high VARCHAR2(80);

trx\_number\_low VARCHAR2(80);

trx\_number\_high VARCHAR2(80);

sel\_new NUMBER := 1;

sel\_inv NUMBER := 2;

sel\_sel NUMBER := 3;

sel\_one NUMBER := 4;

sel\_batch NUMBER := 5;

sel\_adj NUMBER := 6;

choice NUMBER;

p\_userenv\_lang VARCHAR2(4);

language\_code VARCHAR2(4);

p\_base\_lang VARCHAR2(4);

BEGIN

/\*----------------------------------------------------------------------+

| initialization |

+----------------------------------------------------------------------\*/

where1 := '';

where2 := '';

cr := '

';

/\*----------------------------------------------------------------------+

| make dates\_low := P\_dates\_low 00:00:00 |

| make dates\_high := P\_dates\_high 23:59:59 |

+----------------------------------------------------------------------\*/

dates\_low := TRUNC(P\_dates\_low);

dates\_high := Arp\_Standard.CEIL(P\_dates\_high);

adj\_dates\_low := TRUNC( p\_adj\_dates\_low );

adj\_dates\_high := Arp\_Standard.CEIL( p\_adj\_dates\_high);

/\*----------------------------------------------------------------------+

| Protect every occurance of single quote in trx/adj number params|

+----------------------------------------------------------------------\*/

trx\_number\_low := REPLACE( p\_trx\_number\_low, '''', '''''') ;

trx\_number\_high := REPLACE( p\_trx\_number\_high, '''', '''''') ;

adj\_number\_low := REPLACE( p\_adj\_number\_low, '''', '''''') ;

adj\_number\_high := REPLACE( p\_adj\_number\_high, '''', '''''') ;

/\*----------------------------------------------------------------------+

| Check mandatory parameters |

+----------------------------------------------------------------------\*/

IF P\_choice IS NULL THEN

Arp\_Standard.Fnd\_Message( 'AR\_MAND\_PARAMETER\_NULL',

'PARAM', 'P\_choice' );

END IF;

IF P\_open\_invoice IS NULL THEN

Arp\_Standard.Fnd\_Message( 'AR\_MAND\_PARAMETER\_NULL',

'PARAM', 'P\_open\_invoice' );

END IF;

SELECT language\_code

INTO p\_base\_lang

FROM fnd\_languages

WHERE installed\_flag = 'B';

SELECT USERENV('LANG')

INTO p\_userenv\_lang

FROM dual;

IF p\_userenv\_lang IS NULL THEN

language\_code := p\_base\_lang;

ELSE

language\_code := p\_userenv\_lang;

END IF;

/\*----------------------------------------------------------------------+

| Define Tables and aliases |

+----------------------------------------------------------------------\*/

table1 :=

' AR\_ADJUSTMENTS COM\_ADJ, ' || cr ||

' AR\_PAYMENT\_SCHEDULES P, ' || cr ||

' RA\_CUST\_TRX\_LINE\_GL\_DIST REC, ' || cr ||

' RA\_CUSTOMER\_TRX A, ' || cr ||

' HZ\_CUST\_ACCOUNTS B, ' || cr ||

' RA\_TERMS T, ' || cr ||

' RA\_TERMS\_LINES TL, ' || cr ||

' RA\_CUST\_TRX\_TYPES TYPES, ' || cr ||

' AR\_LOOKUPS L\_TYPES, ' || cr ||

' HZ\_PARTIES PARTY, ' || cr || -- bug 1630907

' HZ\_CUST\_ACCT\_SITES A\_BILL, ' || cr ||

' HZ\_PARTY\_SITES PARTY\_SITE, ' || cr ||

' HZ\_LOCATIONS LOC, ' || cr ||

' HZ\_CUST\_SITE\_USES U\_BILL ' || cr;

table2 :=

' RA\_TERMS\_LINES TL, ' || cr ||

' RA\_CUST\_TRX\_TYPES TYPES, ' || cr ||

' AR\_LOOKUPS L\_TYPES, ' || cr ||

' HZ\_CUST\_ACCOUNTS B, ' || cr ||

' HZ\_PARTIES PARTY, ' || cr || --bug 1630907

' HZ\_CUST\_SITE\_USES U\_BILL, ' || cr ||

' HZ\_CUST\_ACCT\_SITES A\_BILL, ' || cr ||

' HZ\_PARTY\_SITES PARTY\_SITE, ' || cr ||

' HZ\_LOCATIONS LOC, ' || cr ||

' AR\_ADJUSTMENTS COM\_ADJ, ' || cr ||

' RA\_CUSTOMER\_TRX A, ' || cr ||

' AR\_PAYMENT\_SCHEDULES P, ' || cr ||

' RA\_TERMS T ' || cr;

/\*----------------------------------------------------------------------+

| Determine which clause to use |

+----------------------------------------------------------------------\*/

IF P\_choice = 'NEW' THEN

choice := sel\_new;

ELSIF p\_choice = 'ADJ' THEN

choice := sel\_adj;

ELSIF P\_choice = 'SEL' THEN

IF trx\_number\_low IS NOT NULL AND trx\_number\_high IS NOT NULL THEN

choice := sel\_inv;

ELSE

choice := sel\_sel;

END IF;

ELSIF P\_choice = 'ONE' THEN

BEGIN

IF P\_customer\_trx\_id IS NULL THEN

Arp\_Standard.Fnd\_Message( 'AR\_MAND\_PARAMETER\_NULL',

'PARAM', 'P\_customer\_trx\_id' );

END IF;

choice := sel\_one;

END;

ELSIF P\_choice = 'BATCH' THEN

BEGIN

IF P\_batch\_id IS NULL THEN

Arp\_Standard.Fnd\_Message( 'AR\_MAND\_PARAMETER\_NULL',

'PARAM', 'P\_batch\_id' );

END IF;

choice := sel\_batch;

END;

ELSE

Arp\_Standard.Fnd\_Message( 'AR\_RAXINV\_INVALID\_PARAMETERS',

'PARAM', 'P\_choice' );

END IF;

where1 := where1 || 'A.BILL\_TO\_CUSTOMER\_ID = B.CUST\_ACCOUNT\_ID';

/\* Join to receivable record \*/

where1 := where1 || cr || 'AND REC.CUSTOMER\_TRX\_ID = A.CUSTOMER\_TRX\_ID';

where1 := where1 || cr || 'AND REC.LATEST\_REC\_FLAG = ''Y'' ';

where1 := where1 || cr || 'AND REC.ACCOUNT\_CLASS = ''REC'' ';

/\* Join to ar\_adjustments for children of commitments. \*/

where1 := where1 || cr || 'AND P.PAYMENT\_SCHEDULE\_ID + DECODE(P.CLASS, ';

where1 := where1 || cr || ' ''INV'', 0,';

where1 := where1 || cr || ' '''')';

where1 := where1 || cr || ' = COM\_ADJ.PAYMENT\_SCHEDULE\_ID(+)';

where1 := where1 || cr || 'AND COM\_ADJ.SUBSEQUENT\_TRX\_ID IS NULL';

where1 := where1 || cr || 'AND ''C'' = COM\_ADJ.ADJUSTMENT\_TYPE(+)';

IF P\_Choice <> 'ADJ' THEN

where1 := where1 || cr || 'AND A.COMPLETE\_FLAG = ''Y''';

/\* bug 762450 :

if P\_Choice <> 'SEL' THEN

where1 := where1 || cr || 'AND A.PRINTING\_OPTION IN (''PRI'', ''REP'')';

end if;

\*/

END IF;

where1 := where1 || cr || 'AND A.CUST\_TRX\_TYPE\_ID = TYPES.CUST\_TRX\_TYPE\_ID';

where1 := where1 || cr || 'AND L\_TYPES.LOOKUP\_TYPE = ''INV/CM/ADJ''';

--

--gv added on 01/03/10

--

where1 := where1 || cr || 'AND L\_TYPES.LOOKUP\_CODE IN (''INV'',''DM'',''CM'')';

/\*where1 := where1 || cr || 'AND exists (select 1 from ra\_customer\_trx\_lines lines

where A.customer\_trx\_id = lines.customer\_trx\_id

)';\*/

--end gv

-- bug 762450 :

-- where1 := where1 || cr || 'AND TYPES.DEFAULT\_PRINTING\_OPTION = ''PRI''';

where1 := where1 || cr || 'AND A.PRINTING\_OPTION IN (''PRI'', ''REP'')';

IF p\_choice = 'ADJ'

THEN

where1 := where1 || cr || 'AND L\_TYPES.LOOKUP\_CODE = ''ADJ''';

ELSE

where1 := where1 || cr || 'AND L\_TYPES.LOOKUP\_CODE = ';

-- where1 := where1 || cr || 'DECODE( TYPES.TYPE,''DEP'',''INV'', TYPES.TYPE)';

where1 := where1 || cr || 'DECODE( TYPES.TYPE,''DEP'',''DEP'', TYPES.TYPE)'; --GV 0219

END IF;

where1 := where1 || cr || 'AND NVL(P.TERMS\_SEQUENCE\_NUMBER,nvl(TL.SEQUENCE\_NUM,0))=nvl(TL.SEQUENCE\_NUM,nvl(p.terms\_sequence\_number,0))';

where1 := where1 || cr || 'AND DECODE(P.PAYMENT\_SCHEDULE\_ID,'''',0, NVL(T.PRINTING\_LEAD\_DAYS,0))=0';

where1 := where1 || cr || 'AND A.BILL\_TO\_SITE\_USE\_ID = U\_BILL.SITE\_USE\_ID';

where1 := where1 || cr || 'AND U\_BILL.CUST\_ACCT\_SITE\_ID = A\_BILL.CUST\_ACCT\_SITE\_ID';

where1 := where1 || cr || 'AND A\_BILL.party\_site\_id = party\_site.party\_site\_id';

where1 := where1 || cr || 'AND B.PARTY\_ID = PARTY.PARTY\_ID'; -- bug 1630907

where1 := where1 || cr || 'AND loc.location\_id = party\_site.location\_id';

IF p\_call\_from = 'INV' THEN

where1 := where1 || cr || 'AND NVL(LOC.LANGUAGE,''' || p\_base\_lang || ''') = ''' || language\_code || '''';

END IF;

IF SUBSTR(UPPER(p\_open\_invoice),1,1) = 'Y'

THEN

-- Bug Fix 359960, where Credit memos are printed irrespective of whether they are open

-- or not is not what the documentation says. CM's like all other debit items should be

-- dependent on the p\_open\_invoice parameter. Hence reverting back the changes made for

-- Bug 359960. Changes made for bug 1639132.

where1 := where1 || cr || 'AND NVL(P.AMOUNT\_DUE\_REMAINING,0) <> 0';

END IF;

IF P\_customer\_id IS NOT NULL THEN

where1 := where1 || cr || 'AND B.CUST\_ACCOUNT\_ID = ' || P\_customer\_id;

END IF;

IF P\_cust\_trx\_type\_id IS NOT NULL THEN

where1 := where1 || cr || 'AND A.CUST\_TRX\_TYPE\_ID = ' || P\_cust\_trx\_type\_id;

END IF;

IF p\_cust\_trx\_class IS NOT NULL THEN

where1 := where1 || cr || 'AND TYPES.TYPE = ''' || p\_cust\_trx\_class

|| '''';

END IF;

where1 := where1 || cr || 'AND A.TERM\_ID = TL.TERM\_ID(+)';

where1 := where1 || cr || 'AND A.TERM\_ID = T.TERM\_ID(+)';

IF P\_installment\_number IS NOT NULL THEN

where1 := where1 || cr || 'AND A.CUSTOMER\_TRX\_ID = P.CUSTOMER\_TRX\_ID';

where1 := where1 || cr || 'AND NVL(TL.SEQUENCE\_NUM, 1) = '

|| P\_installment\_number;

ELSE

where1 := where1 || cr || 'AND A.CUSTOMER\_TRX\_ID = P.CUSTOMER\_TRX\_ID(+)';

END IF;

IF dates\_low IS NOT NULL AND dates\_high IS NOT NULL THEN

where1 := where1 || cr || 'AND A.TRX\_DATE BETWEEN TO\_DATE(''';

where1 := where1 || TO\_CHAR(dates\_low,'DD-MM-YYYY-HH24:MI:SS');

where1 := where1 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

where1 := where1 || cr || ' AND TO\_DATE(''';

where1 := where1 || TO\_CHAR(dates\_high,'DD-MM-YYYY-HH24:MI:SS');

where1 := where1 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

ELSIF dates\_low IS NOT NULL THEN

where1 := where1 || cr || 'AND A.TRX\_DATE >= TO\_DATE(''';

where1 := where1 || TO\_CHAR(dates\_low,'DD-MM-YYYY-HH24:MI:SS');

where1 := where1 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

ELSIF dates\_high IS NOT NULL THEN

where1 := where1 || cr || 'AND A.TRX\_DATE <= TO\_DATE(''';

where1 := where1 || TO\_CHAR(dates\_high,'DD-MM-YYYY-HH24:MI:SS');

where1 := where1 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

END IF;

IF P\_customer\_class\_code IS NOT NULL THEN

where1 := where1 || cr || 'AND B.CUSTOMER\_CLASS\_CODE = ''' || P\_customer\_class\_code || '''';

END IF;

where2 := where2 || 'A.BILL\_TO\_CUSTOMER\_ID = B.CUST\_ACCOUNT\_ID';

/\* Join to ar\_adjyustments for children of commitments. \*/

where2 := where2 || cr || 'AND P.PAYMENT\_SCHEDULE\_ID + DECODE(P.CLASS, ';

where2 := where2 || cr || ' ''INV'', 0,';

where2 := where2 || cr || ' '''')';

where2 := where2 || cr || ' = COM\_ADJ.PAYMENT\_SCHEDULE\_ID(+)';

where2 := where2 || cr || 'AND COM\_ADJ.SUBSEQUENT\_TRX\_ID IS NULL';

where2 := where2 || cr || 'AND ''C'' = COM\_ADJ.ADJUSTMENT\_TYPE(+)';

IF P\_Choice <> 'ADJ' THEN

where2 := where2 || cr || 'AND A.COMPLETE\_FLAG = ''Y''';

/\* BUG 762450

if P\_Choice <> 'SEL' THEN

where2 := where2 || cr || 'AND A.PRINTING\_OPTION IN (''PRI'', ''REP'')';

end if;

\*/

END IF;

where2 := where2 || cr || 'AND A.CUSTOMER\_TRX\_ID = P.CUSTOMER\_TRX\_ID';

where2 := where2 || cr || 'AND A.CUST\_TRX\_TYPE\_ID = TYPES.CUST\_TRX\_TYPE\_ID';

where2 := where2 || cr || 'AND L\_TYPES.LOOKUP\_TYPE = ''INV/CM/ADJ''';

--

--gv added on 01/03/10

--

where2 := where2 || cr || 'AND L\_TYPES.LOOKUP\_CODE IN (''INV'',''DM'',''CM'')';

/\*where2 := where2 || cr || 'AND exists (select 1 from ra\_customer\_trx\_lines lines

where A.customer\_trx\_id = lines.customer\_trx\_id

)';\*/

--end gv

-- bug 762450

-- where2 := where2 || cr || 'AND TYPES.DEFAULT\_PRINTING\_OPTION = ''PRI''';

where2 := where2 || cr || 'AND A.PRINTING\_OPTION IN (''PRI'', ''REP'')';

IF p\_choice = 'ADJ'

THEN

where2 := where2 || cr || 'AND L\_TYPES.LOOKUP\_CODE = ''ADJ''';

ELSE

where2 := where2 || cr || 'AND L\_TYPES.LOOKUP\_CODE = ';

--where2 := where2 || cr || 'DECODE( TYPES.TYPE,''DEP'',''INV'', TYPES.TYPE)';

where2 := where2 || cr || 'DECODE( TYPES.TYPE,''DEP'',''DEP'', TYPES.TYPE)'; --GV02/19

END IF;

where2 := where2 || cr || 'AND NVL(T.PRINTING\_LEAD\_DAYS,0) > 0';

where2 := where2 || cr || 'AND A.BILL\_TO\_SITE\_USE\_ID = U\_BILL.SITE\_USE\_ID';

where2 := where2 || cr || 'AND U\_BILL.CUST\_ACCT\_SITE\_ID = A\_BILL.CUST\_ACCT\_SITE\_ID';

where2 := where2 || cr || 'AND A\_BILL.PARTY\_SITE\_ID = PARTY\_SITE.PARTY\_SITE\_ID';

where2 := where2 || cr || 'AND B.PARTY\_ID = PARTY.PARTY\_ID'; -- bug 1630907

where2 := where2 || cr || 'AND LOC.LOCATION\_ID = PARTY\_SITE.LOCATION\_ID';

IF p\_call\_from = 'INV' THEN

where2 := where2 || cr || 'AND NVL(LOC.LANGUAGE,''' || p\_base\_lang || ''') = ''' || language\_code || '''';

END IF;

where2 := where2 || cr || 'AND NVL(P.TERMS\_SEQUENCE\_NUMBER,TL.SEQUENCE\_NUM)=TL.SEQUENCE\_NUM';

IF SUBSTR(UPPER(p\_open\_invoice),1,1) = 'Y'

THEN

-- Bug Fix 359960, where Credit memos are printed irrespective of whether they are open

-- or not is not what the documentation says. CM's like all other debit items should be

-- dependent on the p\_open\_invoice parameter. Hence reverting back the changes made for

-- Bug 359960. Changes made for bug 1639132.

where2 := where2 || cr || 'AND NVL(P.AMOUNT\_DUE\_REMAINING,0) <> 0';

END IF;

IF P\_customer\_id IS NOT NULL THEN

where2 := where2 || cr || 'AND B.CUST\_ACCOUNT\_ID = ' || P\_customer\_id;

END IF;

IF P\_cust\_trx\_type\_id IS NOT NULL THEN

where2 := where2 || cr || 'AND A.CUST\_TRX\_TYPE\_ID = ' || P\_cust\_trx\_type\_id;

END IF;

IF p\_cust\_trx\_class IS NOT NULL THEN

where2 := where2 || cr || 'AND TYPES.TYPE = ''' || p\_cust\_trx\_class || '''';

END IF;

IF P\_installment\_number IS NOT NULL THEN

where2 := where2 || cr || 'AND T.TERM\_ID = A.TERM\_ID' ;

where2 := where2 || cr || 'AND TL.TERM\_ID = T.TERM\_ID' ;

where2 := where2 || cr || 'AND P.TERMS\_SEQUENCE\_NUMBER = ' || P\_installment\_number ;

ELSE

where2 := where2 || cr || 'AND T.TERM\_ID = P.TERM\_ID' ;

where2 := where2 || cr || 'AND TL.TERM\_ID(+) = T.TERM\_ID' ;

END IF;

IF dates\_low IS NOT NULL AND dates\_high IS NOT NULL THEN

where2 := where2 || cr || 'AND P.DUE\_DATE BETWEEN TO\_DATE(''';

where2 := where2 || TO\_CHAR(dates\_low,'DD-MM-YYYY-HH24:MI:SS');

where2 := where2 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

where2 := where2 || cr || ' + NVL (T.PRINTING\_LEAD\_DAYS, 0)';

where2 := where2 || cr || ' AND TO\_DATE(''';

where2 := where2 || TO\_CHAR(dates\_high,'DD-MM-YYYY-HH24:MI:SS');

where2 := where2 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

where2 := where2 || cr || ' + NVL (T.PRINTING\_LEAD\_DAYS, 0)';

ELSIF dates\_low IS NOT NULL THEN

where2 := where2 || cr || 'AND P.DUE\_DATE >= TO\_DATE(''';

where2 := where2 || TO\_CHAR(dates\_low,'DD-MM-YYYY-HH24:MI:SS');

where2 := where2 || ''',''DD-MM-YYYY-HH24:MI:SS'') + NVL (T.PRINTING\_LEAD\_DAYS, 0)';

ELSIF dates\_high IS NOT NULL THEN

where2 := where2 || cr || 'AND P.DUE\_DATE <= TO\_DATE(''';

where2 := where2 || TO\_CHAR(dates\_high,'DD-MM-YYYY-HH24:MI:SS');

where2 := where2 || ''',''DD-MM-YYYY-HH24:MI:SS'') + NVL (T.PRINTING\_LEAD\_DAYS, 0)';

END IF;

IF P\_customer\_class\_code IS NOT NULL THEN

where2 := where2 || cr || 'AND B.CUSTOMER\_CLASS\_CODE = ''' || P\_customer\_class\_code || '''';

END IF;

IF choice = sel\_new THEN

GOTO sel\_new\_invoice;

ELSIF choice = sel\_inv THEN

GOTO sel\_selected\_inv;

ELSIF choice = sel\_sel THEN

GOTO sel\_selected;

ELSIF choice = sel\_batch THEN

GOTO sel\_batch\_invoice;

ELSIF choice = sel\_one THEN

GOTO sel\_one\_invoice;

ELSIF choice = sel\_adj THEN

GOTO sel\_adj\_adjustments;

END IF;

/\*----------------------------------------------------------------------+

| select new invoices |

+----------------------------------------------------------------------\*/

<<sel\_new\_invoice>>

where1 := where1 || cr || 'AND A.PRINTING\_PENDING = ''Y''';

where1 := where1 || cr ||

'AND NVL(TL.SEQUENCE\_NUM, 1) > NVL(A.LAST\_PRINTED\_SEQUENCE\_NUM,0)';

where2 := where2 || cr || 'AND A.PRINTING\_PENDING = ''Y''';

where2 := where2 || cr || 'AND P.TERMS\_SEQUENCE\_NUMBER > NVL(A.LAST\_PRINTED\_SEQUENCE\_NUM,0)';

/\*----------------------------------------------------------------------+

| select selected invoices |

+----------------------------------------------------------------------\*/

<<sel\_selected\_inv>>

where3 := '';

IF trx\_number\_low IS NOT NULL AND trx\_number\_high IS NOT NULL THEN

where3 := where3 || cr || 'AND A.TRX\_NUMBER BETWEEN';

where3 := where3 || ' ''' || trx\_number\_low || '''';

where3 := where3 || ' AND ''' || trx\_number\_high || '''';

ELSIF trx\_number\_low IS NOT NULL THEN

where3 := where3 || cr || 'AND A.TRX\_NUMBER >= ''' || trx\_number\_low || '''';

ELSIF trx\_number\_high IS NOT NULL THEN

where3 := where3 || cr || 'AND A.TRX\_NUMBER <= ''' || trx\_number\_high || '''';

END IF;

where1 := where1 || where3;

where2 := where2 || where3;

GOTO ok\_exit;

/\*----------------------------------------------------------------------+

| otherwise |

+----------------------------------------------------------------------\*/

<<sel\_selected>>

GOTO ok\_exit;

/\*----------------------------------------------------------------------+

| select by batch |

+----------------------------------------------------------------------\*/

<<sel\_batch\_invoice>>

IF P\_batch\_id IS NOT NULL THEN

where1 := where1 || cr || 'AND A.BATCH\_ID = ' || P\_batch\_id;

where2 := where2 || cr || 'AND A.BATCH\_ID = ' || P\_batch\_id;

END IF;

GOTO ok\_exit;

/\*----------------------------------------------------------------------+

| select one invoice |

+----------------------------------------------------------------------\*/

<<sel\_one\_invoice>>

IF P\_customer\_trx\_id IS NOT NULL THEN

where1 := where1 || cr || 'AND A.CUSTOMER\_TRX\_ID = ' || P\_customer\_trx\_id;

where2 := where2 || cr || 'AND A.CUSTOMER\_TRX\_ID = ' || P\_customer\_trx\_id;

END IF;

GOTO ok\_exit;

/\*----------------------------------------------------------------------+

| select adjustments |

+----------------------------------------------------------------------\*/

<<sel\_adj\_adjustments>>

table1 := table1 || ', AR\_ADJUSTMENTS ADJ' || cr ;

table2 := table2 || ', AR\_ADJUSTMENTS ADJ' || cr ;

where3 := cr || 'and a.customer\_trx\_id = adj.customer\_trx\_id ' ;

IF adj\_number\_low IS NOT NULL OR adj\_number\_high IS NOT NULL

THEN

BEGIN

IF adj\_number\_low IS NOT NULL AND adj\_number\_high IS NOT NULL THEN

where3 := where3 || cr || 'AND ADJ.ADJUSTMENT\_NUMBER BETWEEN';

where3 := where3 || ' ''' || adj\_number\_low || '''';

where3 := where3 || ' AND ''' || adj\_number\_high || '''';

ELSIF adj\_number\_low IS NOT NULL THEN

where3 := where3 || cr || 'AND ADJ.ADJUSTMENT\_NUMBER >= ''' || adj\_number\_low || '''';

ELSIF adj\_number\_high IS NOT NULL THEN

where3 := where3 || cr || 'AND ADJ.ADJUSTMENT\_NUMBER <= ''' || adj\_number\_high || '''';

END IF;

END;

END IF;

IF adj\_dates\_low IS NOT NULL OR adj\_dates\_high IS NOT NULL

THEN

BEGIN

IF adj\_dates\_low IS NOT NULL AND adj\_dates\_high IS NOT NULL THEN

where3 := where3 || cr || 'AND ADJ.APPLY\_DATE BETWEEN TO\_DATE(''';

where3 := where3 || TO\_CHAR(adj\_dates\_low,'DD-MM-YYYY-HH24:MI:SS');

where3 := where3 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

where3 := where3 || cr || ' AND TO\_DATE(''';

where3 := where3 || TO\_CHAR(adj\_dates\_high,'DD-MM-YYYY-HH24:MI:SS');

where3 := where3 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

ELSIF adj\_dates\_low IS NOT NULL THEN

where3 := where3 || cr || 'AND ADJ.APPLY\_DATE >= TO\_DATE(''';

where3 := where3 || TO\_CHAR(adj\_dates\_low,'DD-MM-YYYY-HH24:MI:SS');

where3 := where3 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

ELSIF adj\_dates\_high IS NOT NULL THEN

where3 := where3 || cr || 'AND ADJ.APPLY\_DATE <= TO\_DATE(''';

where3 := where3 || TO\_CHAR(adj\_dates\_high,'DD-MM-YYYY-HH24:MI:SS');

where3 := where3 || ''',''DD-MM-YYYY-HH24:MI:SS'')';

END IF;

END;

END IF;

where3 := where3 || cr || 'AND ADJ.STATUS = ''A''';

where1 := where1 || where3;

where2 := where2 || where3;

GOTO sel\_selected\_inv;

/\*----------------------------------------------------------------------+

| exit successfully |

+----------------------------------------------------------------------\*/

<<ok\_exit>>

P\_where1 := where1;

P\_where2 := where2;

p\_table1 := table1;

p\_table2 := table2;

END; /\* end of procedure build\_where\_clause \*/

--end ARP\_TRX\_SELECT\_CONTROL;

---

--

--

---

PROCEDURE RUN ( p\_errbuf OUT VARCHAR2,

p\_retcode OUT NUMBER,

P\_ORDER\_BY IN VARCHAR2,

P\_choice IN VARCHAR2,

p\_cust\_trx\_class IN VARCHAR2,

P\_cust\_trx\_type\_id IN NUMBER,

P\_trx\_number\_low IN VARCHAR2,

P\_trx\_number\_high IN VARCHAR2,

P\_dates\_low IN DATE,

P\_dates\_high IN DATE,

p\_sales\_order\_low IN VARCHAR2,

p\_sales\_order\_high IN VARCHAR2,

P\_DELIVERY\_NUM\_LOW IN VARCHAR2,

P\_DELIVERY\_NUM\_HIGH IN VARCHAR2,

P\_CUST\_PO\_NUM\_LOW IN VARCHAR2,

P\_CUST\_PO\_NUM\_HIGH IN VARCHAR2,

P\_customer\_class\_code IN VARCHAR2,

P\_customer\_id IN NUMBER,

p\_customer\_num\_id IN NUMBER,

P\_installment\_number IN NUMBER,

P\_open\_invoice IN VARCHAR2,

p\_check\_for\_taxyn IN VARCHAR2,

p\_tax\_registration\_number IN VARCHAR2,

p\_header\_pages IN NUMBER,

p\_debug\_flag IN VARCHAR2,

p\_message\_level IN NUMBER,

p\_random\_invoices\_flag IN VARCHAR2,

p\_invoice\_list\_string IN VARCHAR2,

P\_REPORT\_PRINT\_TEMP IN VARCHAR2)

IS

lv\_language VARCHAR2 (6 BYTE);

lv\_TERRITORY VARCHAR2 (6 BYTE);

lv\_set\_layout\_option BOOLEAN;

lv\_conc\_request\_id NUMBER;

lv\_printer\_name VARCHAR2(100);

lv\_printer BOOLEAN;

lv\_request\_id NUMBER:=Fnd\_Global.CONC\_REQUEST\_ID;

lv\_org\_id NUMBER;

lv\_NUMBER\_OF\_COPIES NUMBER;

lv\_interval NUMBER:=10;

lv\_max\_wait NUMBER:=0;

lb\_result BOOLEAN;

lv\_phase VARCHAR2(20);

lv\_status VARCHAR2(20);

lv\_dev\_phase VARCHAR2(20);

lv\_dev\_status VARCHAR2(20);

lv\_message1 VARCHAR2(20);

lv\_user\_id NUMBER;

lv\_resp\_id NUMBER;

lv\_resp\_appl\_id NUMBER;

REPORT\_NOT\_SUBMITTED EXCEPTION;

CHILD\_REQUEST\_FAILED EXCEPTION;

BEGIN

Fnd\_File.PUT\_LINE( Fnd\_File.output,'Order By: '||P\_ORDER\_BY );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_CHOICE: '|| P\_choice );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_cust\_trx\_class: '|| p\_cust\_trx\_class );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_cust\_trx\_type\_id: '||P\_cust\_trx\_type\_id );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_trx\_number\_low : '|| P\_trx\_number\_low );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_trx\_number\_high : '||P\_trx\_number\_high );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_dates\_low: '|| P\_dates\_low );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_dates\_high : '|| P\_dates\_high );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_sales\_order\_low : '|| p\_sales\_order\_low );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_sales\_order\_high: '||p\_sales\_order\_high );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_DELIVERY\_NUM\_LOW: '||P\_DELIVERY\_NUM\_LOW );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_DELIVERY\_NUM\_HIGH : '||P\_DELIVERY\_NUM\_HIGH );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_CUST\_PO\_NUM\_LOW: '||P\_CUST\_PO\_NUM\_LOW );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_CUST\_PO\_NUM\_HIGH: '||P\_CUST\_PO\_NUM\_HIGH );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_customer\_class\_code: '||P\_customer\_class\_code );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_customer\_id : '||P\_customer\_id );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_customer\_num\_id: '||p\_customer\_num\_id );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_installment\_number: '||P\_installment\_number );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_open\_invoice: '||P\_open\_invoice );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_check\_for\_taxyn: '||p\_check\_for\_taxyn );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_tax\_registration\_number: '||p\_tax\_registration\_number );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_header\_pages: '||p\_header\_pages );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_debug\_flag: '||p\_debug\_flag );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_message\_level: '||p\_message\_level );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_random\_invoices\_flag: '||p\_random\_invoices\_flag );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'p\_invoice\_list\_string: '||p\_invoice\_list\_string );

Fnd\_File.PUT\_LINE( Fnd\_File.output,'P\_REPORT\_PRINT\_TEMP: '||P\_REPORT\_PRINT\_TEMP );

SELECT LANGUAGE,

TERRITORY

INTO lv\_language,

lv\_TERRITORY

FROM XDO\_LOBS

WHERE LOB\_TYPE='TEMPLATE\_SOURCE'

AND APPLICATION\_SHORT\_NAME='XXGIL'

AND LOB\_CODE='XXGILARCOMINV'

AND FILE\_NAME = P\_REPORT\_PRINT\_TEMP;

Fnd\_File.PUT\_LINE( Fnd\_File.output, 'After print name..... '||lv\_language||' - '||lv\_TERRITORY);

--Derive the printer Name

BEGIN

SELECT printer,

NVL(NUMBER\_OF\_COPIES,0) ,

REQUESTED\_BY,

RESPONSIBILITY\_APPLICATION\_ID,

RESPONSIBILITY\_ID

INTO lv\_printer\_name,

lv\_NUMBER\_OF\_COPIES,

lv\_user\_id,

lv\_resp\_appl\_id,

lv\_resp\_id

FROM fnd\_concurrent\_requests

WHERE request\_id = lv\_request\_id;

EXCEPTION

WHEN OTHERS THEN

NULL;

END;

Fnd\_Global.APPS\_INITIALIZE (lv\_user\_id,lv\_resp\_id,lv\_resp\_appl\_id);

--Fnd\_Profile.GET ('ORG\_ID', lv\_org\_id); --null

lv\_org\_id:=Mo\_Global.get\_current\_org\_id;

Fnd\_File.PUT\_LINE( Fnd\_File.output, 'Org ID: '|| lv\_org\_id);

Fnd\_Request.set\_org\_id(lv\_org\_id);

-- lv\_set\_layout\_option := Fnd\_Request.add\_layout('XXGIL','Gilead Print Commercial Invoices',lv\_language,lv\_TERRITORY,'PDF');

lv\_set\_layout\_option := Fnd\_Request.add\_layout('XXGIL','XXGILARCOMINV',lv\_language,lv\_TERRITORY,'PDF');

IF lv\_printer\_name IS NOT NULL THEN

lv\_printer := Fnd\_Request.set\_print\_options( lv\_printer\_name,

NULL,

lv\_NUMBER\_OF\_COPIES, -- no of copies

TRUE,

'N');

END IF;

Fnd\_File.PUT\_LINE( Fnd\_File.output, 'Setup printer name '||lv\_printer\_name);

-- CALLING THE CHILD REPORT USING FND\_REQUEST

lv\_conc\_request\_id := Fnd\_Request.SUBMIT\_REQUEST(

application => 'XXGIL',

program => 'XXGILARCOMINV',

description => NULL,

start\_time => TO\_CHAR(SYSDATE, 'DD-MON-YY HH:MI:SS'),

sub\_request => FALSE,

argument1 => P\_ORDER\_BY ,

argument2 => P\_choice ,

argument3 => p\_cust\_trx\_class,

argument4 => P\_cust\_trx\_type\_id,

argument5 => P\_trx\_number\_low,

argument6 => P\_trx\_number\_high,

argument7 => P\_dates\_low,

argument8 => P\_dates\_high,

argument9 => p\_sales\_order\_low ,

argument10 => p\_sales\_order\_high,

argument11 => P\_DELIVERY\_NUM\_LOW,

argument12 => P\_DELIVERY\_NUM\_HIGH ,

argument13 => P\_CUST\_PO\_NUM\_LOW,

argument14 => P\_CUST\_PO\_NUM\_HIGH,

argument15 => P\_customer\_class\_code ,

argument16 => P\_customer\_id,

argument17 => p\_customer\_num\_id,

-- argument18 => P\_installment\_number ,

argument18 => P\_open\_invoice,

argument19 => p\_check\_for\_taxyn,

argument20 => p\_tax\_registration\_number,

argument21 => p\_header\_pages,

argument22 => p\_debug\_flag,

argument23 => p\_message\_level--,

--argument25 => p\_random\_invoices\_flag ,

--argument26 => p\_invoice\_list\_string

);

COMMIT;

IF lv\_conc\_request\_id= 0 THEN

Fnd\_File.put\_line(Fnd\_File.LOG, '\*\*\*\*\*\*\*\*Printing Request Not Submitted. Verify parameters and rerun the report\*\*\*\*\*\*\*\*\*\*\*');

RAISE REPORT\_NOT\_SUBMITTED;

ELSE

lb\_result := Fnd\_Concurrent.wait\_for\_request( lv\_conc\_request\_id,

lv\_interval,

lv\_max\_wait,

lv\_phase,

lv\_status,

lv\_dev\_phase,

lv\_dev\_status,

lv\_message1);

COMMIT;

IF lv\_dev\_phase = 'COMPLETE' THEN

IF lv\_dev\_status = 'WARNING' OR lv\_dev\_status = 'ERROR' THEN

Fnd\_File.put\_line(Fnd\_File.LOG, '\*\*\*\*\*\*\*\*Child Request Failed in '||lv\_dev\_status||' status '||lv\_message1);

RAISE CHILD\_REQUEST\_FAILED;

ELSIF lv\_dev\_status = 'NORMAL' THEN

Fnd\_File.put\_line(Fnd\_File.LOG, 'Child request is scuessful.....');

END IF;

END IF;

END IF;

EXCEPTION

WHEN REPORT\_NOT\_SUBMITTED THEN

RAISE;

WHEN CHILD\_REQUEST\_FAILED THEN

RAISE;

WHEN OTHERS THEN

Fnd\_File.PUT\_LINE(Fnd\_File.OUTPUT,'Error: When others then '||SQLCODE||'-'||SQLERRM);

RAISE;

END RUN;

END Xxgil\_Arp\_Trx\_Select\_Cont\_Pkg ;

**Formula columns:**

* + **Legal entity information:** Create formula column **CF\_LE\_ID** and right following formula to derive legal entity register name and address.

FUNCTION CF\_LE\_IDFORMULA RETURN NUMBER IS

L\_LEGAL\_ENTITY\_ID NUMBER;

L\_ORG\_ID NUMBER;

L\_LE\_NAME VARCHAR2(240);

L\_LE\_REGISTERED\_NAME VARCHAR2(240);

L\_LE\_REGISTER\_NUM VARCHAR2(30);

L\_LE\_ADDRESS1 VARCHAR2(240);

L\_LE\_ADDRESS2 VARCHAR2(240);

L\_LE\_CITY VARCHAR2(30);

L\_LE\_COUNTRY VARCHAR2(60);

L\_LE\_ZIP VARCHAR2(30);

L\_LE\_REGION1 VARCHAR2(120);

L\_LE\_REGION2 VARCHAR2(120);

L\_LE\_TELEPHONES VARCHAR2(120);

L\_SHIP\_DATE\_ACTUAL DATE;

L\_FOB\_POINT VARCHAR2(30);

BEGIN

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: BEFORE LEGAL ENTITY ID... ' || :TRX\_NUMBER);

END IF;

SELECT LEGAL\_ENTITY\_ID

, ORG\_ID

, SHIP\_DATE\_ACTUAL

, FOB\_POINT

INTO L\_LEGAL\_ENTITY\_ID

, L\_ORG\_ID

, L\_SHIP\_DATE\_ACTUAL

, L\_FOB\_POINT

FROM RA\_CUSTOMER\_TRX

WHERE CUSTOMER\_TRX\_ID=:CUSTOMER\_TRX\_ID;

:CP\_SHIP\_DATE\_ACTUAL:=L\_SHIP\_DATE\_ACTUAL;

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: BEFORE LEGAL ENTITY DERIVATION. LEGAL ENTITY ID... ' || L\_LEGAL\_ENTITY\_ID);

END IF;

--LEGAL ENTITY INFORMATION

BEGIN

SELECT PROF.NAME C\_LE\_NAME,

REG.REGISTERED\_NAME,

REG.REGISTRATION\_NUMBER,

HR.ADDRESS\_LINE\_1 C\_LE\_ADDRESS1,

HR.ADDRESS\_LINE\_2 C\_LE\_ADDRESS2,

HR.TOWN\_OR\_CITY C\_LE\_CITY,

HR.COUNTRY C\_LE\_COUNTRY,

HR.POSTAL\_CODE C\_LE\_ZIP,

HR.REGION\_1 C\_LE\_REGION1,

HR.REGION\_2 C\_LE\_REGION2,

NVL(TELEPHONE\_NUMBER\_1,TELEPHONE\_NUMBER\_2) C\_LE\_TELEPHONE

INTO L\_LE\_NAME

, L\_LE\_REGISTERED\_NAME

, L\_LE\_REGISTER\_NUM

, L\_LE\_ADDRESS1

, L\_LE\_ADDRESS2

, L\_LE\_CITY

, L\_LE\_COUNTRY

, L\_LE\_ZIP

, L\_LE\_REGION1

, L\_LE\_REGION2

, L\_LE\_TELEPHONES

FROM XLE\_REGISTRATIONS REG,

XLE\_ENTITY\_PROFILES PROF,

-- HR\_OPERATING\_UNITS HOU,

HR\_LOCATIONS\_ALL HR

WHERE REG.SOURCE\_TABLE='XLE\_ENTITY\_PROFILES'

AND REG.SOURCE\_ID = PROF.LEGAL\_ENTITY\_ID

AND REG.LOCATION\_ID = HR.LOCATION\_ID

AND PROF.LEGAL\_ENTITY\_ID=L\_LEGAL\_ENTITY\_ID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

--:CP\_LEGAL\_ENTITY\_ID:=L\_LEGAL\_ENTITY\_ID;

:CP\_ORG\_ID:=L\_ORG\_ID;

:CP\_LE\_NAME := L\_LE\_NAME;

:CP\_LE\_REGISTERED\_NAME:= L\_LE\_REGISTERED\_NAME;

:CP\_LE\_REGISTER\_NUM :=L\_LE\_REGISTER\_NUM;

:CP\_LE\_ADDRESS1:= L\_LE\_ADDRESS1;

:CP\_LE\_ADDRESS2:=L\_LE\_ADDRESS2;

:CP\_LE\_CITY:= L\_LE\_CITY;

:CP\_LE\_COUNTRY:= L\_LE\_COUNTRY;

:CP\_LE\_ZIP:= L\_LE\_ZIP;

:CP\_LE\_REGION1:= L\_LE\_REGION1;

:CP\_LE\_REGION2:= L\_LE\_REGION2;

:CP\_LE\_TELEPHONES:= L\_LE\_TELEPHONES;

SRW.MESSAGE (5000, 'DEBUG: AFTER LE ID END OF ASSIGNMENT: ' || L\_LE\_REGISTERED\_NAME);

RETURN (L\_LEGAL\_ENTITY\_ID);

END;

**b)** **Header Sales Information:** Create **CF\_head\_sales\_order** formula column and write following code in pl/sql formula column to derive Order details.

FUNCTION CF\_HEAD\_SALES\_ORDERFORMULA RETURN CHAR IS

L\_SALES\_ORDER VARCHAR2(50);

L\_SO\_TYPE VARCHAR2(150);

L\_SALES\_ORDER\_DATE DATE;

L\_DELIVERY\_NAME VARCHAR2(30);

L\_FOB\_CODE VARCHAR2(30);

L\_DELIVERED\_DATE DATE;

L\_ULTIMATE\_DROPOFF\_DATE DATE;

L\_DELIVERY\_TYPE VARCHAR2(30);

L\_SHIP\_METHOD VARCHAR2(30);

L\_DELIVERY\_ID NUMBER;

BEGIN

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: BEFORE SALES ORDER HEADER INFO ' || :TRX\_NUMBER);

END IF;

BEGIN

SELECT DISTINCT TRX\_LINES.SALES\_ORDER,

TRX\_LINES.INTERFACE\_LINE\_ATTRIBUTE2 C\_SO\_TYPE,

TRX\_LINES.SALES\_ORDER\_DATE,

WND.NAME DELIVERY\_NAME,

WND.FOB\_CODE,

WND.DELIVERY\_TYPE,

WND.SHIP\_METHOD\_CODE,

WND.DELIVERY\_ID

INTO L\_SALES\_ORDER,

L\_SO\_TYPE,

L\_SALES\_ORDER\_DATE,

L\_DELIVERY\_NAME,

L\_FOB\_CODE,

L\_DELIVERY\_TYPE,

L\_SHIP\_METHOD,

L\_DELIVERY\_ID

FROM RA\_CUSTOMER\_TRX\_LINES\_ALL TRX\_LINES,

WSH\_NEW\_DELIVERIES WND

WHERE /\*TRX\_LINES.INTERFACE\_LINE\_CONTEXT='ORDER ENTRY'

AND \*/TRX\_LINES.CUSTOMER\_TRX\_ID = :CUSTOMER\_TRX\_ID

AND TRX\_LINES.INTERFACE\_LINE\_ATTRIBUTE3 = WND.NAME;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

WHEN TOO\_MANY\_ROWS THEN

NULL;

END;

:CP\_HEAD\_SO\_TYPE:= L\_SO\_TYPE;

:CP\_HEAD\_SO\_DATE:=L\_SALES\_ORDER\_DATE;

:CP\_HEAD\_DELIVERY\_NAME:=L\_DELIVERY\_NAME;

:CP\_HEAD\_FOB\_CODE:=L\_FOB\_CODE;

:CP\_HEAD\_DELIVERY\_TYPE:=L\_DELIVERY\_TYPE;

:CP\_HEAD\_SHIP\_METHOD:=L\_SHIP\_METHOD;

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: AFTER SALES ORDER HEADER INFO... ' || L\_SALES\_ORDER);

END IF;

BEGIN

SELECT MAX(CREATION\_DATE)

INTO L\_DELIVERED\_DATE

FROM WSHBV\_DOCUMENT\_PACKSLIP

WHERE ENTITY\_NAME = 'WSH\_NEW\_DELIVERIES'

AND ENTITY\_ID = L\_DELIVERY\_ID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

:CP\_HEAD\_DELIVERED\_DATE:=L\_DELIVERED\_DATE;

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: AFTER DELIVERED DATE... ' || L\_DELIVERED\_DATE);

END IF;

RETURN(L\_SALES\_ORDER);

END;

1. **Contract Num:**Derive contract number for sales order.

FUNCTION CF\_CONTRACT\_NUMFORMULA RETURN CHAR IS

L\_AGREEMENT\_NAME VARCHAR2 (30);

BEGIN

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: BEFORE GETTING CONTRACT NUMBER: ' ||:CUSTOMER\_TRX\_ID);

END IF;

BEGIN

SELECT NAME

INTO L\_AGREEMENT\_NAME

FROM SO\_AGREEMENTS SOA,

RA\_CUSTOMER\_TRX RCT

WHERE RCT.CUSTOMER\_TRX\_ID =:CUSTOMER\_TRX\_ID

AND SOA.AGREEMENT\_ID = RCT.AGREEMENT\_ID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

RETURN (L\_AGREEMENT\_NAME);

END;

**d)** **CF\_bill\_to\_tax\_reference:** Derive to bill to customer Tax reference

FUNCTION CF\_BILL\_TO\_TAX\_REFERENCEFORMUL RETURN VARCHAR2 IS

L\_BILL\_TO\_TAX\_REFERENCE VARCHAR2(50);

L\_SHIP\_TO\_TAX\_REFERENCE VARCHAR2(50);

L\_BILL\_TO\_CUST\_CLASS\_CODE VARCHAR2(30);

BEGIN

--BILL TO TAX REFERENCE GV 05/01/2009

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: BEFORE BILL TAX REFERENCE '||:BILL\_TO\_CUSTOMER\_ID);

END IF;

BEGIN

SELECT NVL(HOP.TAX\_REFERENCE,HOP.JGZZ\_FISCAL\_CODE) ,

CUSTOMER\_CLASS\_CODE

INTO L\_BILL\_TO\_TAX\_REFERENCE,

L\_BILL\_TO\_CUST\_CLASS\_CODE

FROM HZ\_ORGANIZATION\_PROFILES HOP,HZ\_CUST\_ACCOUNTS HCA

WHERE HOP.PARTY\_ID = HCA.PARTY\_ID

AND SYSDATE BETWEEN TRUNC(HOP.EFFECTIVE\_START\_DATE) AND TRUNC(NVL(HOP.EFFECTIVE\_END\_DATE,SYSDATE+1))

AND HCA.CUST\_ACCOUNT\_ID = :BILL\_TO\_CUSTOMER\_ID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

--GET THE TRX DATE VALUE

BEGIN

SELECT NVL(HOP.TAX\_REFERENCE,HOP.JGZZ\_FISCAL\_CODE)

INTO L\_BILL\_TO\_TAX\_REFERENCE

FROM HZ\_ORGANIZATION\_PROFILES HOP,HZ\_CUST\_ACCOUNTS HCA

WHERE HOP.PARTY\_ID = HCA.PARTY\_ID

AND :TRX\_DATE BETWEEN tRUNC(HOP.EFFECTIVE\_START\_DATE)

AND TRUNC(NVL(HOP.EFFECTIVE\_END\_DATE,SYSDATE+1))

AND HCA.CUST\_ACCOUNT\_ID = :BILL\_TO\_CUSTOMER\_ID

AND ROWNUM=1;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

End;

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: BEFORE SHIP TAX REFERENCE '||:SHIP\_TO\_CUSTOMER\_ID);

END IF;

--SHIP TO TAX REFERENCE

BEGIN

SELECT NVL(HOP.TAX\_REFERENCE,HOP.JGZZ\_FISCAL\_CODE)

INTO L\_SHIP\_TO\_TAX\_REFERENCE

FROM HZ\_ORGANIZATION\_PROFILES HOP,HZ\_CUST\_ACCOUNTS HCA

WHERE HOP.PARTY\_ID = HCA.PARTY\_ID

AND SYSDATE BETWEEN TRUNC(HOP.EFFECTIVE\_START\_DATE) AND TRUNC(NVL(HOP.EFFECTIVE\_END\_DATE,SYSDATE+1))

AND HCA.CUST\_ACCOUNT\_ID = :SHIP\_TO\_CUSTOMER\_ID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

BEGIN

SELECT NVL(HOP.TAX\_REFERENCE,HOP.JGZZ\_FISCAL\_CODE)

INTO L\_SHIP\_TO\_TAX\_REFERENCE

FROM HZ\_ORGANIZATION\_PROFILES HOP,HZ\_CUST\_ACCOUNTS HCA

WHERE HOP.PARTY\_ID = HCA.PARTY\_ID

AND :TRX\_DATE BETWEEN TRUNC(HOP.EFFECTIVE\_START\_DATE) AND TRUNC(NVL(HOP.EFFECTIVE\_END\_DATE,SYSDATE+1))

AND HCA.CUST\_ACCOUNT\_ID = :SHIP\_TO\_CUSTOMER\_ID

AND ROWNUM=1;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

END;

IF (:P\_DEBUG\_FLAG = 'Y') THEN

SRW.MESSAGE (5000, 'DEBUG: AFTER SHIP TAX REFERENCE '||L\_SHIP\_TO\_TAX\_REFERENCE);

END IF;

:CP\_SHIP\_TO\_TAX\_REFERENCE :=L\_SHIP\_TO\_TAX\_REFERENCE;

:BILL\_TO\_CUST\_CLASS\_CODE:=L\_BILL\_TO\_CUST\_CLASS\_CODE;

RETURN L\_BILL\_TO\_TAX\_REFERENCE;

END;

**E)** **CF\_BANK\_DETAILS:** To derive Remit to Bank Details

FUNCTION CF\_BANK\_DETAILSFORMULA RETURN CHAR IS

L\_PROFIE\_REMITBANK\_DFF VARCHAR2(150);

L\_BANK\_NAME VARCHAR2(100);

L\_BANK\_NUMBER VARCHAR2(30);

L\_BANK\_BRANCH\_NUM VARCHAR2(30);

L\_BANK\_SWIFT\_CODE VARCHAR2(30 BYTE);

L\_BANK\_IBAN\_NUM VARCHAR2(50);

Begin

BEGIN

SELECT SUBSTR(ATTRIBUTE5,1,INSTR(ATTRIBUTE5,'|')-1) BANK\_NAME,

SUBSTR(ATTRIBUTE5,INSTR(ATTRIBUTE5,'|')+1,

(INSTR(ATTRIBUTE5,'|',1,2)-1)- INSTR(ATTRIBUTE5,'|',1,1))

BANK\_NUMBER,

SUBSTR(ATTRIBUTE5,(INSTR(ATTRIBUTE5,'|',1,2)+1))

BANK\_BRANCH\_NUMBER,

ATTRIBUTE5

INTO l\_bank\_name ,

l\_bank\_number,

l\_bank\_branch\_num,

l\_profie\_remitbank\_dff

FROM HZ\_CUSTOMER\_PROFILES

WHERE CUST\_ACCOUNT\_ID=:BILL\_TO\_CUSTOMER\_ID

AND STATUS ='A'

AND SITE\_USE\_ID IS NULL;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

srw.message (5000, 'DEBUG: Remit To Bank data NOT found');

END;

:bill\_to\_bankacct\_num:=l\_bank\_number;

:bill\_to\_bankacct\_name:=l\_bank\_name;

:bill\_to\_bank\_branch\_num:=l\_bank\_branch\_num;

IF l\_profie\_remitbank\_dff is NOT NULL

Then

BEGIN

SELECT b.eft\_swift\_code,

a.iban\_number

INTO l\_bank\_swift\_code,

l\_bank\_IBAN\_num

FROM CE\_BANK\_ACCOUNTS a,CE\_BANK\_BRANCHES\_V B

WHERE A.BANK\_BRANCH\_ID = B.BRANCH\_PARTY\_ID

AND a.BANK\_ACCOUNT\_NAME =l\_bank\_name;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

:bill\_to\_bank\_SWIFT:=l\_bank\_swift\_code;

:bill\_to\_bank\_iban:=l\_bank\_IBAN\_num ;

End If;

RETURN(l\_profie\_remitbank\_dff);

END;

**E)** **C\_TRX\_TYPE:**To derive transaction type

function C\_TRX\_TYPEFormula return Char is

l\_trx\_type VARchar2(100);

begin

srw.message (5000, 'DEBUG: Before getting C\_TRX\_TYPE: ' || :customer\_trx\_id);

SELECT NAME

INTO l\_trx\_type

FROM RA\_CUST\_TRX\_TYPES

WHERE CUST\_TRX\_TYPE\_ID = (SELECT CUST\_TRX\_TYPE\_ID

FROM ra\_customer\_trx

WHERE CUSTOMER\_TRX\_ID = :customer\_trx\_id) ;

RETURN(l\_trx\_type);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN(l\_trx\_type);

end;

**F)** **D\_PRINT\_TRX\_TYPE:**To display local language transaction type, create this formula column.

Function D\_PRINT\_TRX\_TYPEFormula return Char is

l\_print\_trx\_type VARCHAR2(50);

Begin

SRW.MESSAGE(100,'Report Org Country '||:RP\_LE\_COUNTRY||'-'||:REPORT\_ORG\_COUNTRY||'-'||:TRX\_TYPE);

SELECT DESCRIPTION

INTO l\_print\_trx\_type

FROM ar\_lookups

WHERE lookup\_type ='XXGIL\_OTC\_TRX\_TYPE\_TRANSLATION'

AND LOOKUP\_CODE= :REPORT\_ORG\_COUNTRY||'-'||:TRX\_TYPE;

RETURN( l\_print\_trx\_type);

exception

WHEN NO\_DATA\_FOUND THEN

SRW.message(100,'Transaction type not found in XXGIL\_OTC\_TRX\_TYPE\_TRANSLATION');

RAISE srw.program\_abort;

RETURN( null);

END;

G) **CF\_CNTRY\_OF\_ORIGIN:**This formula column has created in G\_LINES query to derive the country of origin to display on the report.

Function CF\_CNTRY\_OF\_ORIGINFormula return Char is

l\_country\_of\_origin varchar2(20);

begin

IF (:P\_Debug\_Flag = 'Y') THEN

srw.message (5000, 'DEBUG: Before Country of Origin. Order Line ID ' || :LINE\_SO\_LINE\_ID);

END IF;

SELECT HL.country

INTO l\_country\_of\_origin

FROM hr\_locations hl,

hr\_organization\_units hou,

OE\_ORDER\_LINES OOL

WHERE hl.location\_id= hou.LOCATION\_id

AND OOL.SHIP\_FROM\_ORG\_ID = HOU.ORGANIZATION\_ID

AND OOL.LINE\_ID=:LINE\_SO\_LINE\_ID;

IF (:P\_Debug\_Flag = 'Y') THEN

srw.message (5000, 'DEBUG: After Country of Origin. Order Line ID ' || l\_country\_of\_origin );

END IF;

RETURN l\_country\_of\_origin;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN l\_country\_of\_origin;

end;

1. **CF\_LINE\_DISC\_AMOUNT**:- This formula column has created in G\_LINES query to derive and display line discount per unit:

FUNCTION LINE\_DISC\_PER\_UNITFORMULA RETURN NUMBER IS

L\_DISC\_PER\_UNIT NUMBER;

BEGIN

--DISCOUNT

IF :LINE\_PRODUCT\_NUM IS NOT NULL THEN

IF NVL(:LINE\_STANDARD\_PRICE,0) <> 0

AND NVL(:LINE\_UNIT\_SELLING\_PRICE,0) <> 0 THEN

IF :LINE\_STANDARD\_PRICE <> :LINE\_UNIT\_SELLING\_PRICE THEN L\_DISC\_PER\_UNIT:=:LINE\_STANDARD\_PRICE - :LINE\_UNIT\_SELLING\_PRICE;

:LINE\_STD\_EXTENDED\_AMT:=:LINE\_STANDARD\_PRICE\* :LINE\_QTY\_INVOICED;

ELSE

:LINE\_STD\_EXTENDED\_AMT:=:LINE\_STANDARD\_PRICE\* :LINE\_QTY\_INVOICED;

END IF;

END IF;

END IF;

RETURN(L\_DISC\_PER\_UNIT);

END;

**I) CF\_LINE\_DISC\_AMOUNT**:- This formula column has created in G\_LINES query to derive and display line discount amount.

FUNCTION CF\_LINE\_DISC\_AMOUNTFORMULA RETURN NUMBER IS

L\_LINE\_DISC\_AMOUNT NUMBER;

L\_LINE\_DISC\_PERCENT NUMBER;

BEGIN

IF (:P\_Debug\_Flag = 'Y') THEN

srw.message (5000, 'DEBUG: Before discount amount. ' );

END IF;

IF nvl(:line\_disc\_per\_unit,0) <> 0 then l\_line\_disc\_amount:=nvl(:line\_qty\_invoiced,0)\*nvl(:line\_disc\_per\_unit,0);

if l\_line\_disc\_amount <> 0 then l\_line\_disc\_percent:=round((nvl(l\_line\_disc\_amount,0)/nvl(:line\_extended\_amount,0)) \*100,2);

:CP\_line\_disc\_percent:=l\_line\_disc\_percent||'%';

end if;

End If;

IF (:P\_Debug\_Flag = 'Y') THEN

srw.message (5000, 'DEBUG: After discount amount. '||l\_line\_disc\_amount );

END IF;

RETURN(l\_line\_disc\_amount);

END;

J) **LINE\_NET\_PRICE**: This formula column has created in G\_LINES query to derive and display line net price.

function LINE\_NET\_PRICEFormula return Number is

begin

return(nvl(:line\_standard\_price,0) - nvl(:line\_disc\_per\_unit,0));

END;

K) **C\_IT\_CONDITION\_TXT2**: This formula column has created in Q\_SUMMARY\_INVOICE to derive and display conditional text specific to Italy.

Function C\_IT\_CONDITION\_TXT2Formula return Char is

l\_it\_condition\_txt VARCHAR2(500);

begin

--This conditional text is for ITALY

IF :REPORT\_ORG\_COUNTRY = 'IT'

-- AND :BILL\_TO\_CUST\_CLASS\_CODE IN ('PUBLIC HOSPITALS' ,'PUBLIC SECTOR COMPANIES') then

AND UPPER(:C\_TRX\_TYPE) LIKE 'GOV%'

THEN

l\_it\_condition\_txt:='FATTURA IN SOSPESO AI SENSI DELL''ART.6, 5° COMMA DEL DPR 633/72 IVA DETRAIBILE DA PARTE DEL CONCESSIONARIO SOLO ALL''ATTO DEL PAGAMENTO';

else

if :inv\_tax\_exemption\_id is not null then

l\_it\_condition\_txt:='NON IMPONIBILE IVA ART. 71/8, 1° COMMA DPR 633';

end if;

end if;

return(l\_it\_condition\_txt);

end;

**Formula columns added for Japan changes:-**

1. CF\_JP\_SHIKIRI\_SHO

This formula column derives either the delivery number or the transaction number (for manual transactions) as Shikiri-sho number.

1. CF\_JP\_CONS\_INV\_NO

This formula column derives either the delivery number or the transaction number (for manual transactions) as Shikiri-sho number.

1. CF\_JP\_BILL\_PARTY\_SITE

This formula column derives the bill to party site name for the respective tranaction

1. CF\_JP\_SO\_DATE

This formula column derives the ordered date of sales order or the transaction date of the transaction if the former is available in Japanese time zone

1. CF\_JP\_WHOLESALER

This formula column derives the wholesaler code from the Site Use Information DFF

1. CF\_JP\_MAKER\_CODE

This formula column derives the Makercode from the lookup XXGIL\_OTC\_JP\_JDNET\_CODES

1. CF\_JP\_TXN\_CODE

This formula column derives the Transaction code for a particular transaction based on the order type, invoice extended amount and transaction type.

1. CF\_JP\_TOTAL

This formula column derives the total invoice amount: total extended amount + total tax amount

1. CF\_JP\_TAX\_AMT

This formula column derives the total invoice tax amount

1. CF\_JP\_TAX\_AMT

This formula column derives the invoice tax rate

1. CF\_JP\_SHIP\_PARTY\_SITE

This formula column derives the Ship to party site name

1. CF\_JP\_UNIFIED\_ITEM

This formula column derives the unified item code for Japan for the transaction/delivery line

1. CF\_JP\_PRODUCT\_NAME

This formula column derives the item name in Kanji for the transaction/delivery line

1. CF\_JP\_UOM

This formula column derives the UOM code for Japan for the transaction/delivery line

1. CF\_JP\_LINE\_PRICE

This forumala column derives the unit selling price for the transaction/delivery line

1. CF\_JP\_EXT\_AMT

This formula column derives the extended amount for each transaction/delivery line

1. CF\_JP\_LOT\_NUMBER

This formula column derives the lot number for each transaction/delivery line

1. CF\_JP\_EXPIRATION\_DATE

This formula column derives the expirationd date for the lot for each transaction/delivery line

## Additional Database Objects

1. Package XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG has created. This package has two procedures.
   1. The RUN procedure used by main program to attach print template, printer and internally call print report.
   2. The BUILD\_WHERE\_CLAUSE procedure will be called from child program to build the where clause for Q\_INVOICE query.

## Log Output

Standard Log output. The report has parameter called “Diagnostics”, if user expects to see more processing messages then select ’Y’ in the parameter.

## Installation Steps

* Execute XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG.pks file to create XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG package specification
* Execute XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG.pkb file to create XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG package body.
* Upload XXGIL\_CM\_CUST\_REMIT\_TO\_BANK.ldt file using FNDLOAD to create XXGIL\_CM\_CUST\_REMIT\_TO\_BANK value set.
* Upload XXGIL\_OTC\_TRX\_TYPE\_TRANSLATION.ldt file using FNDLOAD to create XXGIL\_OTC\_TRX\_TYPE\_TRANSLATION lookup.
* Copy XXGILARCOMINV.rdf file to XXGIL\_TOP/reports/US
* Register the concurrent programs “**Gilead Print Commercial Invoices(Main)”** and “**Gilead Print Commercial Invoices”**
* Register the Data Definition for the above report in XML Publisher responsibility. The data definition Name is **XXGILARCOMINV**.
* Register the Template Definition for the above Data Definition in XML Publisher Responsibility
  + Template Definition Name **– XXGILARCOMINV**
  + Upload rtf template ( Layout template file)

## Registration

### Value Sets

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_SRS\_RAXINV\_ORDER\_BY |
| **Size** | Char(80) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | Table:  AR\_LOOKUPS  Table Columns:  MEANING CHAR(80)  LOOKUP\_CODE CHAR(80)  Where/order by:  where lookup\_type = 'SRS\_RAXINV\_ORDER\_BY'  and meaning in ( 'Customer','Transaction Number')  and lookup\_code != 'ADJUSTMENT\_NUMBER'  order by meaning |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_INV\_PRINT\_CHOICES |
| **Size** | Char(30) |
| **Type** | Independent |
| **Reqd** |  |
| **SQL** | Values:  NEW  SEL |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_RAXINV\_INVOICE\_CLASS |
| **Size** | Char(80) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | Table: ar\_lookups  Table Columns:  Value: Meaning VARCHAR2(80)  ID: LOOKUP\_CODE VARCHAR2(30)    Where/order by:  WHERE lookup\_type = 'INV/CM'  AND lookup\_code != 'PMT'  and lookup\_code in ('CM','INV','DM')  ORDER BY meaning, lookup\_code |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_RAXINV\_TRX\_TYPE |
| **Size** | Char(20) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | Table: RA\_CUST\_TRX\_TYPES CTT  Table Columns:  Value: ctt.name VARCHAR(20)  Meaning: ctt.description VARCHAR2(80)  ID: ctt.cust\_trx\_type\_id NUMBER(38)  Where/order by:  WHERE nvl( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL,  ctt.TYPE) = ctt.TYPE  and ctt.type in ('INV','CM','DM')  ORDER BY ctt.NAME |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_RAXINV\_TRX\_NUMBER |
| **Size** | Char(20) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | **Table:** RA\_CUSTOMER\_TRX TRX, RA\_CUST\_TRX\_TYPES TYPE, AR\_LOOKUPS L1  **Columns:**  Value: TRX.TRX\_NUMBER VARCHAR2(20)  Meaning: SUBSTR( L1.MEANING || ' : ' || TYPE.NAME,1,30) VARCHAR2(30)  ID: TRX.TRX\_NUMBER VARCHAR2(20)  **Where/Order By:**  WHERE TRX.COMPLETE\_FLAG = 'Y'  AND TRX.PRINTING\_OPTION = 'PRI'  AND TRX.CUST\_TRX\_TYPE\_ID = TYPE.CUST\_TRX\_TYPE\_ID  AND TYPE.TYPE = L1.LOOKUP\_CODE  AND L1.LOOKUP\_CODE IN ('INV','CM','DM')  AND L1.LOOKUP\_TYPE = 'INV/CM'  AND TYPE.TYPE =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL, TYPE.TYPE )  AND TRX.CUST\_TRX\_TYPE\_ID =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL, TRX.CUST\_TRX\_TYPE\_ID )  AND TRX.CUSTOMER\_TRX\_ID IN  (SELECT MAX(TRX2.CUSTOMER\_TRX\_ID)  FROM RA\_CUSTOMER\_TRX TRX2,  RA\_CUST\_TRX\_TYPES TYPE2,  AR\_LOOKUPS L2  WHERE TRX2.COMPLETE\_FLAG = 'Y'  AND TRX2.PRINTING\_OPTION = 'PRI'  AND TRX2.CUST\_TRX\_TYPE\_ID =  TYPE2.CUST\_TRX\_TYPE\_ID  AND TYPE2.TYPE = L2.LOOKUP\_CODE  AND L2.LOOKUP\_TYPE = 'INV/CM'  AND TYPE2.TYPE =  NVL( :$FLEX$.AR\_RAXINV\_INVOICE\_CLASS:NULL,  TYPE2.TYPE )  AND TRX2.CUST\_TRX\_TYPE\_ID =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL,  TRX2.CUST\_TRX\_TYPE\_ID)  AND TRX2.TRX\_NUMBER = TRX.TRX\_NUMBER)  ORDER BY TRX.TRX\_NUMBER, L1.MEANING,  TRX.CUST\_TRX\_TYPE\_ID,  TRX.CUSTOMER\_TRX\_ID |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_RAXINV\_SO |
| **Size** | Char(50) |
| **Type** | Table |
| **Reqd** |  |
| **Sql** | **Table:**  RA\_CUSTOMER\_TRX TRX, RA\_CUST\_TRX\_TYPES TYPE, RA\_CUSTOMER\_TRX\_LINES RCTL, AR\_LOOKUPS L1,HZ\_CUST\_ACCOUNTS HCA,HZ\_PARTIES HP  **Columns:**  Value: RCTL.SALES\_ORDER VARCHAR2(50)  ID: RCTL.SALES\_ORDER VARCHAR2(50)  **Where/Order By:**  WHERE TRX.COMPLETE\_FLAG = 'Y'  AND TRX.PRINTING\_OPTION = 'PRI'  AND TRX.CUSTOMER\_TRX\_ID = RCTL.CUSTOMER\_TRX\_ID  AND RCTL.SALES\_ORDER IS NOT NULL  AND TRX.SOLD\_TO\_CUSTOMER\_ID = HCA.CUST\_ACCOUNT\_ID  AND HCA.PARTY\_ID = HP.PARTY\_ID  AND TRX.CUST\_TRX\_TYPE\_ID = TYPE.CUST\_TRX\_TYPE\_ID  AND TYPE.TYPE = L1.LOOKUP\_CODE  AND L1.LOOKUP\_TYPE = 'INV/CM'  AND L1.LOOKUP\_CODE IN ('INV','CM','DM')  AND TYPE.TYPE =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL, TYPE.TYPE )  AND TRX.CUST\_TRX\_TYPE\_ID =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL, TRX.CUST\_TRX\_TYPE\_ID )  AND trx.customer\_Trx\_id IN  (SELECT MAX(RCTL2.customer\_Trx\_id)  FROM RA\_CUSTOMER\_TRX TRX2,  RA\_CUST\_TRX\_TYPES TYPE2,  RA\_CUSTOMER\_TRX\_LINES RCTL2,  AR\_LOOKUPS L2  WHERE TRX2.COMPLETE\_FLAG = 'Y'  AND TRX2.PRINTING\_OPTION = 'PRI'  AND TRX2.CUSTOMER\_TRX\_ID = RCTL2.CUSTOMER\_TRX\_ID  AND RCTL2.SALES\_ORDER IS NOT NULL  AND TRX2.CUST\_TRX\_TYPE\_ID = TYPE2.CUST\_TRX\_TYPE\_ID  AND TYPE2.TYPE = L2.LOOKUP\_CODE  AND L2.LOOKUP\_TYPE = 'INV/CM'  AND L2.LOOKUP\_CODE IN ('INV','CM','DM')  AND TYPE2.TYPE = NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL, TYPE2.TYPE )  AND TRX2.CUST\_TRX\_TYPE\_ID = NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL,  TRX2.CUST\_TRX\_TYPE\_ID)  AND RCTL2.SALES\_ORDER = RCTL.SALES\_ORDER)  ORDER BY RCTL.SALES\_ORDER,hp.party\_name,hca.account\_number  **Additional Columns:**  HP.PARTY\_NAME "Customer Name", HCA.ACCOUNT\_NUMBER "Customer Number"(20) |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_RAXINV\_DELIVERY |
| **Size** | Char(50) |
| **Type** | Table |
| **Verification/Values** |  |
| **SQL** | **Table:**  RA\_CUSTOMER\_TRX TRX, RA\_CUST\_TRX\_TYPES TYPE, RA\_CUSTOMER\_TRX\_LINES RCTL, AR\_LOOKUPS L1  **Columns:**  Value: RCTL.INTERFACE\_LINE\_ATTRIBUTE6 VARCHAR2(50)  ID: RCTL.INTERFACE\_LINE\_ATTRIBUTE6 VARCHAR2(50)  **Where/Order By :**  WHERE TRX.COMPLETE\_FLAG = 'Y'  AND TRX.PRINTING\_OPTION = 'PRI'  AND TRX.CUSTOMER\_TRX\_ID = RCTL.CUSTOMER\_TRX\_ID  AND RCTL.INTERFACE\_LINE\_ATTRIBUTE6 IS NOT NULL  AND TRX.CUST\_TRX\_TYPE\_ID = TYPE.CUST\_TRX\_TYPE\_ID  AND RCTL.INTERFACE\_LINE\_ATTRIBUTE6 IS NOT NULL  AND TYPE.TYPE = L1.LOOKUP\_CODE  AND L1.LOOKUP\_TYPE = 'INV/CM'  AND L1.LOOKUP\_CODE IN ('INV','CM','DM')  AND TYPE.TYPE =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL, TYPE.TYPE )  AND TRX.CUST\_TRX\_TYPE\_ID =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL, TRX.CUST\_TRX\_TYPE\_ID )  AND RCTL.CUSTOMER\_TRX\_LINE\_ID IN  (SELECT MAX(RCTL2.CUSTOMER\_TRX\_LINE\_ID)  FROM RA\_CUSTOMER\_TRX TRX2,  RA\_CUST\_TRX\_TYPES TYPE2,  RA\_CUSTOMER\_TRX\_LINES RCTL2,  AR\_LOOKUPS L2  WHERE TRX2.COMPLETE\_FLAG = 'Y'  AND TRX2.PRINTING\_OPTION = 'PRI'  AND TRX2.CUSTOMER\_TRX\_ID = RCTL2.CUSTOMER\_TRX\_ID  AND RCTL.INTERFACE\_LINE\_ATTRIBUTE6 IS NOT NULL  AND TRX2.CUST\_TRX\_TYPE\_ID = TYPE2.CUST\_TRX\_TYPE\_ID  AND TYPE2.TYPE = L2.LOOKUP\_CODE  AND L2.LOOKUP\_TYPE = 'INV/CM'  AND L2.LOOKUP\_CODE IN ('INV','CM','DM')  AND TYPE2.TYPE = NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL, TYPE2.TYPE )  AND TRX2.CUST\_TRX\_TYPE\_ID = NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL,  TRX2.CUST\_TRX\_TYPE\_ID)  AND RCTL2.INTERFACE\_LINE\_ATTRIBUTE6 = RCTL.INTERFACE\_LINE\_ATTRIBUTE6)  ORDER BY RCTL.INTERFACE\_LINE\_ATTRIBUTE6, RCTL.SALES\_ORDER  **Additional Columns:**  RCTL.SALES\_ORDER "Sales Order" (50) |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_RAXINV\_PO\_NUM |
| **Size** | Char(50) |
| **Type** | Table |
| **Verification/Values** |  |
| **SQL** | **Tables:** RA\_CUSTOMER\_TRX TRX, RA\_CUST\_TRX\_TYPES TYPE, AR\_LOOKUPS L1,HZ\_CUST\_ACCOUNTS HCA,HZ\_PARTIES HP  **Columns:**  Value: TRX.PURCHASE\_ORDER VARCHAR2(50)  ID: TRX.PURCHASE\_ORDER VARCHAR2(50)  **Where/Order By:**  WHERE TRX.COMPLETE\_FLAG = 'Y'  AND TRX.PRINTING\_OPTION = 'PRI'  AND TRX.CUST\_TRX\_TYPE\_ID = TYPE.CUST\_TRX\_TYPE\_ID  AND TYPE.TYPE = L1.LOOKUP\_CODE  AND TRX.PURCHASE\_ORDER IS NOT NULL  AND L1.LOOKUP\_CODE IN ('INV','CM','DM')  AND L1.LOOKUP\_TYPE = 'INV/CM'  AND TYPE.TYPE =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_INVOICE\_CLASS:NULL, TYPE.TYPE )  AND TRX.CUST\_TRX\_TYPE\_ID =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL, TRX.CUST\_TRX\_TYPE\_ID )  AND TRX.SOLD\_TO\_CUSTOMER\_ID = HCA.CUST\_ACCOUNT\_ID  AND HCA.PARTY\_ID = HP.PARTY\_ID  AND TRX.CUSTOMER\_TRX\_ID IN  (SELECT MAX(TRX2.CUSTOMER\_TRX\_ID)  FROM RA\_CUSTOMER\_TRX TRX2,  RA\_CUST\_TRX\_TYPES TYPE2,  AR\_LOOKUPS L2  WHERE TRX2.COMPLETE\_FLAG = 'Y'  AND TRX2.PRINTING\_OPTION = 'PRI'  AND TRX2.CUST\_TRX\_TYPE\_ID =  TYPE2.CUST\_TRX\_TYPE\_ID  AND TYPE2.TYPE = L2.LOOKUP\_CODE  AND L2.LOOKUP\_TYPE = 'INV/CM'  AND TYPE2.TYPE =  NVL( :$FLEX$.AR\_RAXINV\_INVOICE\_CLASS:NULL,  TYPE2.TYPE )  AND TRX2.CUST\_TRX\_TYPE\_ID =  NVL( :$FLEX$.XXGIL\_AR\_RAXINV\_TRX\_TYPE:NULL,  TRX2.CUST\_TRX\_TYPE\_ID)  AND TRX2.PURCHASE\_ORDER = TRX.PURCHASE\_ORDER)  ORDER BY TRX.PURCHASE\_ORDER, HP.PARTY\_NAME |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_CUSTOMER\_NUMBER\_ID |
| **Size** | Char(30) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | **Table Name:**  HZ\_CUST\_ACCOUNTS CUST, HZ\_PARTIES PARTY  **Columns:**  CUST.ACCOUNT\_NUMBER VARCHAR(30)  PARTY.PARTY\_NAME||' ('||CUST.ACCOUNT\_NUMBER||')' VARCHAR(83)  CUST.CUST\_ACCOUNT\_ID NUMBER(22)  **Where/Order By:**  WHERE CUST.PARTY\_ID = PARTY.PARTY\_ID AND  NVL(CUST.CUSTOMER\_CLASS\_CODE,'X') =  nvl(:$FLEX$.AR\_RAXINV\_CUSTOMER\_CLASS:NULL,  nvl(CUST.CUSTOMER\_CLASS\_CODE, 'X'))  ORDER BY PARTY.PARTY\_NAME, CUST.ACCOUNT\_NUMBER, CUST.CUST\_ACCOUNT\_ID  **Additional Columns:**  CUST.ACCOUNT\_NUMBER "Customer Number"(20), PARTY.JGZZ\_FISCAL\_CODE "Taxpayer Id"(15), PARTY.TAX\_REFERENCE "Tax Registration Number"(15) |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_INV\_PRINT\_TEMPLATE |
| **Size** | Char(120) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | Table: XDO\_LOBS  Table Columns:  Value: FILE\_NAME VARCHAR2(120)  Where/order by:  WHERE LOB\_TYPE='TEMPLATE\_SOURCE'  AND APPLICATION\_SHORT\_NAME='XXGIL'  AND LOB\_CODE='XXGILARCOMINV' |

### Concurrent Program Executable

1) **Gilead Print Commercial Invoices (Main)**:

|  |  |
| --- | --- |
| **Executable** | XXGILARCOMINVCP |
| **Short Name** | XXGILARCOMINVCP |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Commercial Invoices (Main) |
| **Execution Method** | PL/SQL Stored Procedure |
| **Execution File Name** | XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG.RUN |

2) **Gilead Print Commercial Invoices:**

|  |  |
| --- | --- |
| **Executable** | XXGILARCOMINV |
| **Short Name** | XXGILARCOMINV |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Commercial Invoices |
| **Execution Method** | Oracle reports |
| **Execution File Name** | XXGILARCOMINV |

### Concurrent Program Definition

1) Gilead Print Commercial Invoices(Main)

|  |  |
| --- | --- |
| **Program** | Gilead Print Commercial Invoices(Main) |
| **Short Name** | XXGILARCOMINVCP |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Commercial Invoices(Main) |
| **Executable Name** | XXGILARCOMINVCP |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | Text |
| **Mls Function** |  |
| **Style** |  |

2) Gilead Print Commercial Invoices:

|  |  |
| --- | --- |
| **Program** | Gilead Print Commercial Invoices |
| **Short Name** | XXGILARCOMINV |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Commercial Invoices |
| **Executable Name** | XXGILARCOMINV |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | XML |
| **Mls Function** |  |
| **Style** | Invoice Print |

### Concurrent Program Definition

1) Gilead Print Commercial Invoices (Main)

|  |  |
| --- | --- |
| **Program** | Gilead Print Commercial Invoices(Main) |
| **Short Name** | XXGILARCOMINVCP |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Commercial Invoices(Main) |
| **Executable Name** | XXGILARCOMINVCP |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | Text |
| **Mls Function** |  |
| **Style** |  |

2) Gilead Print Commercial Invoices:

|  |  |
| --- | --- |
| **Program** | Gilead Print Commercial Invoices |
| **Short Name** | XXGILARCOMINV |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Commercial Invoices |
| **Executable Name** | XXGILARCOMINV |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | XML |
| **Mls Function** |  |
| **Style** | Invoice Print |

### Concurrent Program Parameters

1. **Gilead Print Commercial Invoices(Main)**:

|  |  |
| --- | --- |
| **Seq** | 5 |
| **Parameter Name** | Order By |
| **Parameter** | P\_ORDER\_BY |
| **Value Set/Item Class** | XXGIL\_AR\_SRS\_RAXINV\_ORDER\_BY |
| **Default Value** | select meaning from ar\_lookups where lookup\_type = 'SRS\_RAXINV\_ORDER\_BY' and lookup\_code = 'TRX\_NUMBER' |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Order By |

|  |  |
| --- | --- |
| **Seq** | 7 |
| **Parameter Name** | Print Choice |
| **Parameter** | P\_CHOICE |
| **Value Set/Item Class** | XXGIL\_AR\_INV\_PRINT\_CHOICES |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | Y |
| **Prompt** | Print Choice |

|  |  |
| --- | --- |
| **Seq** | 10 |
| **Parameter Name** | Transaction Class |
| **Parameter** | P\_CUST\_TRX\_CLASS |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_INVOICE\_CLASS |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Transaction Class |

|  |  |
| --- | --- |
| **Seq** | 20 |
| **Parameter Name** | Transaction Type |
| **Parameter** | P\_CUST\_TRX\_TYPE\_ID |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_TRX\_TYPE |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Transaction Type |

|  |  |
| --- | --- |
| **Seq** | 30 |
| **Parameter Name** | Transaction Number Low |
| **Parameter** | P\_TRX\_NUMBER\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_TRX\_NUMBER |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Transaction Number Low |

|  |  |
| --- | --- |
| **Seq** | 40 |
| **Parameter Name** | Transaction Number High |
| **Parameter** | P\_TRX\_NUMBER\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_TRX\_NUMBER |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Transaction Number High |

|  |  |
| --- | --- |
| **Seq** | 50 |
| **Parameter Name** | Trx Date Low |
| **Parameter** | P\_TRX\_DATE\_LOW |
| **Value Set/Item Class** | FND\_STANDARD\_DATE |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Trx Date Low |

|  |  |
| --- | --- |
| **Seq** | 60 |
| **Parameter Name** | Trx Date High |
| **Parameter** | P\_TRX\_DATE\_HIGH |
| **Value Set/Item Class** | FND\_STANDARD\_DATE |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Trx Date High |

|  |  |
| --- | --- |
| **Seq** | 62 |
| **Parameter Name** | Sales Order Low |
| **Parameter** | P\_SALES\_ORDER\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_SO |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Sales Order Low |

|  |  |
| --- | --- |
| **Seq** | 63 |
| **Parameter Name** | Sales Order High |
| **Parameter** | P\_SALES\_ORDER\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_SO |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Sales Order High |

|  |  |
| --- | --- |
| **Seq** | 64 |
| **Parameter Name** | Delivery Number Low |
| **Parameter** | P\_DELIVERY\_NUM\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_DELIVERY |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Delivery Number Low |

|  |  |
| --- | --- |
| **Seq** | 65 |
| **Parameter Name** | Delivery Number High |
| **Parameter** | P\_DELIVERY\_NUM\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_DELIVERY |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Delivery Number High |

|  |  |
| --- | --- |
| **Seq** | 66 |
| **Parameter Name** | Customer PO Num Low |
| **Parameter** | P\_CUST\_PO\_NUM\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_PO\_NUM |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer PO Num Low |

|  |  |
| --- | --- |
| **Seq** | 67 |
| **Parameter Name** | Customer PO Num High |
| **Parameter** | P\_CUST\_PO\_NUM\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_PO\_NUM |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer PO Num High |

|  |  |
| --- | --- |
| **Seq** | 70 |
| **Parameter Name** | Customer Class |
| **Parameter** | P\_CUSTOMER\_CLASS\_CODE |
| **Value Set/Item Class** | AR\_RAXINV\_CUSTOMER\_CLASS |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer Class |

|  |  |
| --- | --- |
| **Seq** | 80 |
| **Parameter Name** | Customer |
| **Parameter** | P\_CUSTOMER\_ID |
| **Value Set/Item Class** | AR\_CUSTOMER\_NAME\_ID |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Customer |

|  |  |
| --- | --- |
| **Seq** | 82 |
| **Parameter Name** | Customer Number |
| **Parameter** | P\_CUSTOMER\_NUM\_ID |
| **Value Set/Item Class** | XXGIL\_AR\_CUSTOMER\_NUMBER\_ID |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer Number |

|  |  |
| --- | --- |
| **Seq** | 150 |
| **Parameter Name** | Installment Number |
| **Parameter** | P\_INSTALLMENT\_NUMBER |
| **Value Set/Item Class** | 7/Number |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Installment Number |

|  |  |
| --- | --- |
| **Seq** | 160 |
| **Parameter Name** | Open Invoices Only |
| **Parameter** | P\_OPEN\_INVOICE |
| **Value Set/Item Class** | XXGIL\_AR\_SRS\_RAXINV\_ORDER\_BY |
| **Default Value** | select meaning from fnd\_lookups where lookup\_type = 'YES\_NO' and lookup\_code = 'Y' |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Open Invoices Only |

|  |  |
| --- | --- |
| **Seq** | 170 |
| **Parameter Name** | Print Tax Yes/No Flag |
| **Parameter** | P\_CHECK\_FOR\_TAXYN |
| **Value Set/Item Class** | AR\_SRS\_YES\_NO |
| **Default Value** | select meaning from fnd\_lookups where lookup\_type = 'YES\_NO' and lookup\_code = 'N' |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Print Tax Yes/No Flag |

|  |  |
| --- | --- |
| **Seq** | 190 |
| **Parameter Name** | Tax Registration Number |
| **Parameter** | P\_TAX\_REGISTRATION\_NUMBER |
| **Value Set/Item Class** | AP\_SRS\_ALPHANUMERIC\_OPT |
| **Default Value** | SELECT TAX\_REGISTRATION\_NUMBER FROM AR\_SYSTEM\_PARAMETERS |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Tax Registration Number |

|  |  |
| --- | --- |
| **Seq** | 220 |
| **Parameter Name** | Number of alignment pages |
| **Parameter** | P\_HEADER\_PAGES |
| **Value Set/Item Class** | FND\_NUMBER15 |
| **Default Value** | Constant - 1 |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Number of alignment pages |

|  |  |
| --- | --- |
| **Seq** | 290 |
| **Parameter Name** | Diagnostics |
| **Parameter** | P\_DEBUG\_FLAG |
| **Value Set/Item Class** | 30 Characters Optional |
| **Default Value** | Constant - N |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Diagnostics |

|  |  |
| --- | --- |
| **Seq** | 300 |
| **Parameter Name** | Message Level |
| **Parameter** | P\_MESSAGE\_LEVEL |
| **Value Set/Item Class** | 7/Number |
| **Default Value** | Constant - 10 |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Message Level |

|  |  |
| --- | --- |
| **Seq** | 310 |
| **Parameter Name** | Random Invoices Flag |
| **Parameter** | P\_RANDOM\_INVOICES\_FLAG |
| **Value Set/Item Class** | AR\_SRS\_YES\_NO |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Random Invoices Flag |

|  |  |
| --- | --- |
| **Seq** | 320 |
| **Parameter Name** | Invoice List |
| **Parameter** | P\_INVOICE\_LIST\_STRING |
| **Value Set/Item Class** | 240 Characters |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | 240 Characters |
| **Seq** | 330 |
| **Parameter Name** | Template Name |
| **Parameter** | P\_REPORT\_PRINT\_TEMP |
| **Value Set/Item Class** | XXGIL\_INV\_PRINT\_TEMPLATE |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | Y |
| **Prompt** | Print Template |

1. **Gilead Print Commercial Invoices:** Following are parameters

|  |  |
| --- | --- |
| **Seq** | 5 |
| **Parameter Name** | Order By |
| **Parameter** | P\_ORDER\_BY |
| **Value Set/Item Class** | XXGIL\_AR\_SRS\_RAXINV\_ORDER\_BY |
| **Default Value** | select meaning from ar\_lookups where lookup\_type = 'SRS\_RAXINV\_ORDER\_BY' and lookup\_code = 'TRX\_NUMBER' |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Order By |

|  |  |
| --- | --- |
| **Seq** | 7 |
| **Parameter Name** | Print Choice |
| **Parameter** | P\_CHOICE |
| **Value Set/Item Class** | XXGIL\_AR\_INV\_PRINT\_CHOICES |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | Y |
| **Prompt** | Print Choice |

|  |  |
| --- | --- |
| **Seq** | 10 |
| **Parameter Name** | Transaction Class |
| **Parameter** | P\_CUST\_TRX\_CLASS |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_INVOICE\_CLASS |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Transaction Class |

|  |  |
| --- | --- |
| **Seq** | 20 |
| **Parameter Name** | Transaction Type |
| **Parameter** | P\_CUST\_TRX\_TYPE\_ID |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_TRX\_TYPE |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Transaction Type |

|  |  |
| --- | --- |
| **Seq** | 30 |
| **Parameter Name** | Transaction Number Low |
| **Parameter** | P\_TRX\_NUMBER\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_TRX\_NUMBER |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Transaction Number Low |

|  |  |
| --- | --- |
| **Seq** | 40 |
| **Parameter Name** | Transaction Number High |
| **Parameter** | P\_TRX\_NUMBER\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_TRX\_NUMBER |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Transaction Number High |

|  |  |
| --- | --- |
| **Seq** | 50 |
| **Parameter Name** | Trx Date Low |
| **Parameter** | P\_TRX\_DATE\_LOW |
| **Value Set/Item Class** | FND\_STANDARD\_DATE |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Trx Date Low |

|  |  |
| --- | --- |
| **Seq** | 60 |
| **Parameter Name** | Trx Date High |
| **Parameter** | P\_TRX\_DATE\_HIGH |
| **Value Set/Item Class** | FND\_STANDARD\_DATE |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Trx Date High |

|  |  |
| --- | --- |
| **Seq** | 62 |
| **Parameter Name** | Sales Order Low |
| **Parameter** | P\_SALES\_ORDER\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_SO |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Sales Order Low |

|  |  |
| --- | --- |
| **Seq** | 63 |
| **Parameter Name** | Sales Order High |
| **Parameter** | P\_SALES\_ORDER\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_SO |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Sales Order High |

|  |  |
| --- | --- |
| **Seq** | 64 |
| **Parameter Name** | Delivery Number Low |
| **Parameter** | P\_DELIVERY\_NUM\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_DELIVERY |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Delivery Number Low |

|  |  |
| --- | --- |
| **Seq** | 65 |
| **Parameter Name** | Delivery Number High |
| **Parameter** | P\_DELIVERY\_NUM\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_DELIVERY |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Delivery Number High |

|  |  |
| --- | --- |
| **Seq** | 66 |
| **Parameter Name** | Customer PO Num Low |
| **Parameter** | P\_CUST\_PO\_NUM\_LOW |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_PO\_NUM |
| **Default Value** |  |
| **Range** | Low |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer PO Num Low |

|  |  |
| --- | --- |
| **Seq** | 67 |
| **Parameter Name** | Customer PO Num High |
| **Parameter** | P\_CUST\_PO\_NUM\_HIGH |
| **Value Set/Item Class** | XXGIL\_AR\_RAXINV\_PO\_NUM |
| **Default Value** |  |
| **Range** | High |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer PO Num High |

|  |  |
| --- | --- |
| **Seq** | 70 |
| **Parameter Name** | Customer Class |
| **Parameter** | P\_CUSTOMER\_CLASS\_CODE |
| **Value Set/Item Class** | AR\_RAXINV\_CUSTOMER\_CLASS |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer Class |

|  |  |
| --- | --- |
| **Seq** | 80 |
| **Parameter Name** | Customer |
| **Parameter** | P\_CUSTOMER\_ID |
| **Value Set/Item Class** | AR\_CUSTOMER\_NAME\_ID |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Customer |

|  |  |
| --- | --- |
| **Seq** | 82 |
| **Parameter Name** | Customer Number |
| **Parameter** | P\_CUSTOMER\_NUM\_ID |
| **Value Set/Item Class** | XXGIL\_AR\_CUSTOMER\_NUMBER\_ID |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Customer Number |

|  |  |
| --- | --- |
| **Seq** | 150 |
| **Parameter Name** | Installment Number |
| **Parameter** | P\_INSTALLMENT\_NUMBER |
| **Value Set/Item Class** | 7/Number |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Installment Number |

|  |  |
| --- | --- |
| **Seq** | 160 |
| **Parameter Name** | Open Invoices Only |
| **Parameter** | P\_OPEN\_INVOICE |
| **Value Set/Item Class** | XXGIL\_AR\_SRS\_RAXINV\_ORDER\_BY |
| **Default Value** | select meaning from fnd\_lookups where lookup\_type = 'YES\_NO' and lookup\_code = 'Y' |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Open Invoices Only |

|  |  |
| --- | --- |
| **Seq** | 170 |
| **Parameter Name** | Print Tax Yes/No Flag |
| **Parameter** | P\_CHECK\_FOR\_TAXYN |
| **Value Set/Item Class** | AR\_SRS\_YES\_NO |
| **Default Value** | select meaning from fnd\_lookups where lookup\_type = 'YES\_NO' and lookup\_code = 'N' |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Print Tax Yes/No Flag |

|  |  |
| --- | --- |
| **Seq** | 190 |
| **Parameter Name** | Tax Registration Number |
| **Parameter** | P\_TAX\_REGISTRATION\_NUMBER |
| **Value Set/Item Class** | AP\_SRS\_ALPHANUMERIC\_OPT |
| **Default Value** | SELECT TAX\_REGISTRATION\_NUMBER FROM AR\_SYSTEM\_PARAMETERS |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Tax Registration Number |

|  |  |
| --- | --- |
| **Seq** | 220 |
| **Parameter Name** | Number of alignment pages |
| **Parameter** | P\_HEADER\_PAGES |
| **Value Set/Item Class** | FND\_NUMBER15 |
| **Default Value** | Constant - 1 |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | N |
| **Prompt** | Number of alignment pages |

|  |  |
| --- | --- |
| **Seq** | 290 |
| **Parameter Name** | Diagnostics |
| **Parameter** | P\_DEBUG\_FLAG |
| **Value Set/Item Class** | 30 Characters Optional |
| **Default Value** | Constant - N |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Diagnostics |

|  |  |
| --- | --- |
| **Seq** | 300 |
| **Parameter Name** | Message Level |
| **Parameter** | P\_MESSAGE\_LEVEL |
| **Value Set/Item Class** | 7/Number |
| **Default Value** | Constant - 10 |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Message Level |

|  |  |
| --- | --- |
| **Seq** | 310 |
| **Parameter Name** | Random Invoices Flag |
| **Parameter** | P\_RANDOM\_INVOICES\_FLAG |
| **Value Set/Item Class** | AR\_SRS\_YES\_NO |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Random Invoices Flag |

|  |  |
| --- | --- |
| **Seq** | 320 |
| **Parameter Name** | Invoice List |
| **Parameter** | P\_INVOICE\_LIST\_STRING |
| **Value Set/Item Class** | 240 Characters |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | 240 Characters |

## Report Change Requests:

* + 1. **OTC CR#65:** Need to have separate remittance information for customers with whom we have direct debit relationship-Germany (DE)

**Functional Requirement:**

Depending on the receipt Method and the Invoice class a different footer text should be printed on the Invoice (Credit Memo) in DE  
**Detailed logic:**

1. If Invoice Class = Credit Memo do not print any text
2. Else If Invoice Class = Invoice then
   1. If the Creation Method of the Receipt Class of the receipt Method of the Invoice is “Automatic”then  
      print: Der Rechnungsbetrag wird bei Fälligkeit von Ihrem Konto eingezogen.
   2. Else print: Bitte überweisen Sie den fälligen Betrag unter Angabe der Rechnungsnummer.
3. **OTC CR#103b:** Changes to complete the French Customer Invoice.

**Functional Requirement:**

* + 1. "Payment Method”to be added as one of the columns on the report – This value will be hard coded for go-live
    2. Split the aggregated quantity per lot quantity - Similar to Turkish requirement and to support reconciliation of product, units and lots between pack slip (delivery note) and the invoice. The business would request the invoice to have quantity information split by lot for a given line item.
    3. Changes to Customer facing Invoice report to show discounts for FOC on a separate line from the product. Additionally, the percent discount must show in the “REMISE” column on the invoice.
    4. Invoice copies number – France business would like to collate invoices for printing purposes, at the invoice level. This means that for a list of 10 invoices, 5 copies each, the first invoice will print5 times before moving to printing the second invoice, and so on. Current printing solution prints the entire group of invoices once, before printing the entire group again based on the number of copies requested.This is requested to facilitate distribution of a given invoice to multiple parties without having to collate these manually from multiple print runs. The solution will require splitting the invoice print file into individual print files for each invoice before sending to the print queue.

**Detailed Logic:**

Split the aggregated quantity per lot quantity – This logic has applied to five countries (France, Germany, Austria, Switzerland and Italy). Below queries added to retrieve appropriate data for invoice lines in lines query,

CR1936 – Added the Logic to include the Lot numbers for Kite Drop Shop Order and added additional condition to filter the charge lines

--Created by Suresh Kandru for split by lot quantity for products. Seven individual queries are developed and join with UNION please see comments before each query

-- Modified by Skandru on Feb14-2011 for freight lines and added code for order by logic

--Modified by Skandru on Apr28-2011 for invoiced qty and extended amount columns added decode function to get correct qty from UOM conversion INT6 defect #4212

--Modified by Skandru on May04-2011 for processing charge lines these lines are not having delivery name so adding extra query to get order/any process charge lines defect #4397

--Modified by Skandru on May31-2011 to display split shipment lines for defect #5380

--Modified by Suresh Kandru on Oct29-2011 if product shipped from single lot it should show single line on invoice.Production defect #7089

--modified by Pragathi Kumar in Sept2013 to correct discrepancy between total taxable amount and sum of line taxable amounts

/\* Senthil : CR1936 - Kite Drop Shop Process - Missing Lot number and Lot Expiry date \*/

SELECT t.line\_customer\_trx\_id

,t.line\_customer\_trx\_line\_id

,t.line\_number

,t.line\_type

,t.line\_item\_description

,t.line\_qty\_ordered

,t.line\_qty\_invoiced

,t.line\_uom

,t.line\_delivery\_name

,t.line\_so\_line\_id

,t.inventory\_item\_id

,t.line\_product\_num

,t.line\_uom\_code

,t.line\_standard\_price

,t.line\_unit\_selling\_price

,round(t.line\_extended\_amount,2) line\_extended\_amount

,t.line\_net\_amount

,t.line\_sales\_order

,t.line\_sales\_order\_date

,t.line\_tax\_rate

,t.line\_vat\_tax\_id

,t.line\_tax\_exemption\_id

,t.line\_location\_rate\_id

,t.line\_tax\_precedence

,t.lot\_number

,t.line\_is\_a\_child\_flag

,t.link\_to\_line

,t.line\_child\_indicator

,t.link\_to\_cust\_trx\_line\_id

,t.line\_of\_type\_frt

,t.order\_by1

,t.dummy

,t.line\_tax\_inclusive

,t.line\_detail\_id

FROM (SELECT c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, decode( c2.line\_number,

null, c.line\_number,

null) line\_number

, c.line\_type line\_type

, nvl(c.translated\_description,c.description) line\_item\_description

, nvl(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, nvl(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, u.unit\_of\_measure line\_uom

--gv added 01/03/10

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

, c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

/\*, NVL(c.unit\_standard\_price,0) \* nvl(c.quantity\_invoiced, c.quantity\_credited) line\_std\_extended\_amt

, ( NVL(c.unit\_standard\_price,0) - NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price)) line\_disc\_per\_unit\*/

, c.unit\_standard\_price line\_standard\_price

/\*end of gv\*/

, nvl (c.unit\_selling\_price, c.gross\_unit\_selling\_price) /\* Bug 2335596 \*/

line\_unit\_selling\_price

, nvl (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount /\* Bug 2335596 \*/

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, null lot\_number

, decode( to\_char(c2.line\_number),

null, 'N', 'Y') line\_is\_a\_child\_flag

, nvl(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, decode(c.line\_type,

'LINE', 0,

1) line\_child\_indicator,

nvl(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

/\* comment out for bug 745960 :

, decode(msi.item\_type,

'FRT', 'Z',

'A') line\_of\_type\_frt

\*/

, decode(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, decode(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 dummy

--gv added nvl 01/05/10

, nvl( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

,NULL line\_detail\_id

from ra\_customer\_trx\_lines c

, ra\_customer\_trx\_lines c2

, mtl\_units\_of\_measure u

, mtl\_system\_items msi

where c.customer\_trx\_id = :customer\_trx\_id

and c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

and c.uom\_code = u.uom\_code(+)

and c.inventory\_item\_id = msi.inventory\_item\_id (+)

and msi.organization\_id(+) = :so\_organization\_id

and NVL(c.interface\_line\_context,'X') !='ORDER ENTRY'

and :CP\_TRX\_TYPE\_CLASS ='INV'

-- Added by skandru for lines split by lot qty and product qty converted by uom conversion, These lines context is order entry and transaction type is Invoice

UNION

SELECT distinct c.customer\_trx\_id line\_customer\_trx\_id

,c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

-- NULL line\_number

,c.line\_number line\_number

,c.line\_type line\_type

,NVL(c.translated\_description,c.description) line\_item\_description

--Suresh Kandru added code for defect #7089

--,NVL(wdd.requested\_quantity,wdd.shipped\_quantity) line\_qty\_ordered

,(SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)) line\_qty\_ordered

,DECODE(wdd.src\_requested\_quantity\_uom,wdd.requested\_quantity\_uom,(SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),NVL(Xxgil\_Otc\_Print\_Global\_Pkg.xxgil\_uom\_conversion\_qty(c.inventory\_item\_id,wdd.requested\_quantity\_uom),1)\*NVL((SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),0)) line\_qty\_invoiced

,u.unit\_of\_measure line\_uom

,c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

,c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

,c.inventory\_item\_id

,msi.segment1 line\_product\_num

,u.uom\_code line\_uom\_code

--,NULL -- c.unit\_standard\_price line\_standard\_price

,c.unit\_standard\_price line\_standard\_price

, nvl (c.unit\_selling\_price, c.gross\_unit\_selling\_price) line\_unit\_selling\_price

, NVL (c.unit\_selling\_price, NVL(c.gross\_unit\_selling\_price,0))\* DECODE(wdd.src\_requested\_quantity\_uom,wdd.requested\_quantity\_uom,(SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),(NVL(Xxgil\_Otc\_Print\_Global\_Pkg.xxgil\_uom\_conversion\_qty(c.inventory\_item\_id,wdd.requested\_quantity\_uom),1)\*NVL((SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),0))) line\_extended\_amount

, NULL--c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, wdd.lot\_number lot\_number

, NULL -- line\_is\_a\_child\_flag

, NULL -- link\_to\_line

, NULL -- line\_child\_indicator,

, NULL -- link\_to\_cust\_trx\_line\_id,

, NULL --

, c.line\_number order\_by1

, 1 dummy

, NVL( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

--, wdd.delivery\_detail\_id line\_detail\_id -- comment for defect #7089

,NULL line\_detail\_id

FROM ra\_customer\_trx\_lines c,

mtl\_system\_items msi,

wsh\_new\_deliveries wnd,

wsh\_delivery\_assignments wda,

wsh\_delivery\_details wdd,

mtl\_lot\_numbers mln,

mtl\_units\_of\_measure u

WHERE c.customer\_trx\_id = :customer\_trx\_id

AND c.interface\_line\_attribute3 = wnd.NAME

AND c.interface\_line\_attribute6 = wdd.source\_line\_id

AND wdd.inventory\_item\_id = mln.inventory\_item\_id(+)

AND wnd.delivery\_id = wda.delivery\_id

AND wda.delivery\_detail\_id=wdd.delivery\_detail\_id

AND wdd.lot\_number = mln.lot\_number(+)

AND wdd.organization\_id = mln.organization\_id(+)

AND c.uom\_code = u.uom\_code(+)

AND c.inventory\_item\_id = msi.inventory\_item\_id (+)

AND msi.organization\_id(+) = :so\_organization\_id

AND NVL(c.interface\_line\_context,'X') ='ORDER ENTRY'

AND :CP\_TRX\_TYPE\_CLASS ='INV'

-- Added by suresh kandru for processing charge lines these lines are not having delivery name so adding extra query to get order/any process charge lines defect #4397

UNION

SELECT c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, DECODE( c2.line\_number,

NULL, c.line\_number,

NULL) line\_number

, c.line\_type line\_type

, NVL(c.translated\_description,c.description) line\_item\_description

, NVL(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, NVL(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, u.unit\_of\_measure line\_uom

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

, c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

, c.unit\_standard\_price line\_standard\_price

, NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price) /\* Bug 2335596 \*/

line\_unit\_selling\_price

, NVL (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount /\* Bug 2335596 \*/

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, NULL lot\_number

, DECODE( TO\_CHAR(c2.line\_number),

NULL, 'N', 'Y') line\_is\_a\_child\_flag

, NVL(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, DECODE(c.line\_type,

'LINE', 0,

1) line\_child\_indicator

, NVL(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

, DECODE(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, DECODE(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 Dummy

, NVL( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

,NULL line\_detail\_id

FROM ra\_customer\_trx\_lines c

, ra\_customer\_trx\_lines c2

, mtl\_units\_of\_measure u

, mtl\_system\_items msi

WHERE c.customer\_trx\_id = :customer\_trx\_id

AND c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

AND c.uom\_code = u.uom\_code(+)

AND c.inventory\_item\_id = msi.inventory\_item\_id (+)

AND msi.organization\_id(+) = :so\_organization\_id

AND NVL(c.interface\_line\_context,'X') ='ORDER ENTRY'

AND NVL(c.interface\_line\_attribute3,1)=0

/\* Senthil : CR1936 - Drop Ship Orders will delivery name as 0, hence adding one more condtion for charge lines \*/

AND msi.shippable\_item\_flag = ‘N’

AND :CP\_TRX\_TYPE\_CLASS ='INV'

/\* Senthil : CR1936 - Drop Ship Order Changes\*/

UNION

SELECT distinct c.customer\_trx\_id line\_customer\_trx\_id

,c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

-- NULL line\_number

,c.line\_number line\_number

,c.line\_type line\_type

,NVL(c.translated\_description,c.description) line\_item\_description

--Suresh Kandru added code for defect #7089

--,NVL(wdd.requested\_quantity,wdd.shipped\_quantity) line\_qty\_ordered

,NVL(oola.shipping\_quantity, oola.shipped\_quantity) line\_qty\_ordered

,c.quantity\_invoiced line\_qty\_invoiced

--,(SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)) line\_qty\_ordered

--,DECODE(wdd.src\_requested\_quantity\_uom,wdd.requested\_quantity\_uom,(SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),NVL(Xxgil\_Otc\_Print\_Global\_Pkg.xxgil\_uom\_conversion\_qty(c.inventory\_item\_id,wdd.requested\_quantity\_uom),1)\*NVL((SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),0)) line\_qty\_invoiced

,u.unit\_of\_measure line\_uom

,c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

,c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

,c.inventory\_item\_id

,msi.segment1 line\_product\_num

,u.uom\_code line\_uom\_code

--,NULL -- c.unit\_standard\_price line\_standard\_price

,c.unit\_standard\_price line\_standard\_price

, nvl (c.unit\_selling\_price, c.gross\_unit\_selling\_price) line\_unit\_selling\_price

,c.extended\_amount line\_extended\_amount

--, NVL (c.unit\_selling\_price, NVL(c.gross\_unit\_selling\_price,0))\* DECODE(wdd.src\_requested\_quantity\_uom,wdd.requested\_quantity\_uom,(SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),(NVL(Xxgil\_Otc\_Print\_Global\_Pkg.xxgil\_uom\_conversion\_qty(c.inventory\_item\_id,wdd.requested\_quantity\_uom),1)\*NVL((SUM(wdd.shipped\_quantity) OVER (PARTITION BY c.line\_number,msi.inventory\_item\_id,wdd.lot\_number)),0))) line\_extended\_amount

, NULL--c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, mln.lot\_number lot\_number

, NULL -- line\_is\_a\_child\_flag

, NULL -- link\_to\_line

, NULL -- line\_child\_indicator,

, NULL -- link\_to\_cust\_trx\_line\_id,

, NULL --

, c.line\_number order\_by1

, 1 dummy

, NVL( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

--, wdd.delivery\_detail\_id line\_detail\_id -- comment for defect #7089

,NULL line\_detail\_id

FROM ra\_customer\_trx\_lines c,

mtl\_system\_items msi,

apps.oe\_order\_lines\_all oola,

apps.mtl\_parameters mp,

apps.rcv\_transactions rt,

apps.rcv\_lot\_transactions rlt,

apps.OE\_DROP\_SHIP\_SOURCES os,

apps.ra\_customer\_trx\_all rct,

mtl\_lot\_numbers mln,

mtl\_units\_of\_measure u

WHERE c.customer\_trx\_id = :customer\_trx\_id

AND oola.line\_id = c.interface\_line\_attribute6

AND oola.inventory\_item\_id = mln.inventory\_item\_id

AND mp.organization\_id = mln.organization\_id

AND os.header\_id = oola.header\_id

AND rt.po\_header\_id = os.po\_header\_id

AND rt.transaction\_id = rlt.transaction\_id

AND rlt.lot\_num = mln.lot\_number

AND rct.customer\_trx\_id = :customer\_trx\_id

AND c.uom\_code = u.uom\_code(+)

AND c.inventory\_item\_id = msi.inventory\_item\_id (+)

AND msi.organization\_id(+) = :so\_organization\_id

AND NVL(c.interface\_line\_context,'X') ='ORDER ENTRY'

AND :CP\_TRX\_TYPE\_CLASS ='INV'

--Added by skandru below query retrieve lines those are having context order entry and transaction type is credit memo

UNION

SELECT c.customer\_trx\_id line\_customer\_trx\_id

,c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

--,NULL -- LINE NUMBER

,c.line\_number line\_number

,c.line\_type line\_type

,NVL(c.translated\_description,c.description) line\_item\_description

,NVL(rlt.quantity,0) line\_qty\_ordered

,NVL(rlt.quantity,0) \*-1 line\_qty\_invoiced

,u.unit\_of\_measure line\_uom

,c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

,c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

,c.inventory\_item\_id

,msi.segment1 line\_product\_num

,u.uom\_code line\_uom\_code

--,NULL -- c.unit\_standard\_price line\_standard\_price

,c.unit\_standard\_price line\_standard\_price

, NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price) line\_unit\_selling\_price

, NVL (c.unit\_selling\_price, NVL(c.gross\_unit\_selling\_price,0))\* NVL(rlt.quantity,0) \* -1 line\_extended\_amount

, NULL--c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, rlt.lot\_num lot\_number

, NULL -- line\_is\_a\_child\_flag

, NULL -- link\_to\_line

, NULL -- line\_child\_indicator,

, NULL -- link\_to\_cust\_trx\_line\_id,

, NULL --

, c.line\_number order\_by1

, 1 dummy

, NVL( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

,NULL line\_detail\_id

FROM apps.ra\_customer\_trx\_lines c,

apps.mtl\_system\_items msi,

apps.mtl\_units\_of\_measure u,

apps.rcv\_lot\_transactions rlt,

apps.rcv\_transactions rt,

apps.oe\_order\_lines\_all oola,

apps.rcv\_shipment\_lines rsl

WHERE c.customer\_trx\_id = :customer\_trx\_id

AND rsl.oe\_order\_line\_id = oola.line\_id

AND rt.shipment\_line\_id= rsl.shipment\_line\_id

AND rt.transaction\_id=rlt.transaction\_id

AND rt.transaction\_type='RECEIVE'

AND TO\_CHAR(oola.line\_id)=c.INTERFACE\_LINE\_ATTRIBUTE6

AND rlt.item\_id=c.inventory\_item\_id

AND rsl.source\_document\_code='RMA'

AND c.uom\_code = u.uom\_code(+)

AND c.inventory\_item\_id = msi.inventory\_item\_id (+)

AND msi.organization\_id(+) = :so\_organization\_id

AND NVL(c.interface\_line\_context,'X') ='ORDER ENTRY'

AND :CP\_TRX\_TYPE\_CLASS ='CM'

--Added by skandru below query retrieve return processing charge lines for credit memos defect#5339

UNION

SELECT c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, DECODE( c2.line\_number,

NULL, c.line\_number,

NULL) line\_number

, c.line\_type line\_type

, NVL(c.translated\_description,c.description) line\_item\_description

, NVL(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, NVL(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, u.unit\_of\_measure line\_uom

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

, c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

, c.unit\_standard\_price line\_standard\_price

, NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price) /\* Bug 2335596 \*/

line\_unit\_selling\_price

, NVL (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount /\* Bug 2335596 \*/

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, NULL lot\_number

, DECODE( TO\_CHAR(c2.line\_number),

NULL, 'N', 'Y') line\_is\_a\_child\_flag

, NVL(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, DECODE(c.line\_type,

'LINE', 0,

1) line\_child\_indicator

, NVL(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

, DECODE(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, DECODE(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 Dummy

, NVL( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

,NULL line\_detail\_id

FROM apps.ra\_customer\_trx\_lines c

, apps.ra\_customer\_trx\_lines c2

, apps.mtl\_units\_of\_measure u

, apps.mtl\_system\_items msi

, apps.oe\_order\_lines oola

WHERE c.customer\_trx\_id = :customer\_trx\_id

AND c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

AND c.uom\_code = u.uom\_code(+)

AND c.inventory\_item\_id = msi.inventory\_item\_id (+)

AND msi.organization\_id(+) = :so\_organization\_id

AND NVL(c.interface\_line\_context,'X') ='ORDER ENTRY'

AND oola.line\_id=TO\_CHAR(c.interface\_line\_attribute6)

AND NVL(oola.shippable\_flag,'Y')='N'

AND UPPER(oola.line\_category\_code) ='RETURN'

AND :CP\_TRX\_TYPE\_CLASS ='CM'

-- Added by skandru below query retrieve lines context is not order entry and transaction type is credit memo

UNION

SELECT c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, decode( c2.line\_number,

null, c.line\_number,

null) line\_number

, c.line\_type line\_type

, nvl(c.translated\_description,c.description) line\_item\_description

, nvl(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, nvl(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, u.unit\_of\_measure line\_uom

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

, c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

/\*, NVL(c.unit\_standard\_price,0) \* nvl(c.quantity\_invoiced, c.quantity\_credited) line\_std\_extended\_amt

, ( NVL(c.unit\_standard\_price,0) - NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price)) line\_disc\_per\_unit\*/

, c.unit\_standard\_price line\_standard\_price

, nvl (c.unit\_selling\_price, c.gross\_unit\_selling\_price) /\* Bug 2335596 \*/

line\_unit\_selling\_price

, nvl (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount /\* Bug 2335596 \*/

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, null lot\_number

, decode( to\_char(c2.line\_number),

null, 'N', 'Y') line\_is\_a\_child\_flag

, nvl(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, decode(c.line\_type,

'LINE', 0,

1) line\_child\_indicator,

nvl(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

/\* comment out for bug 745960 :

, decode(msi.item\_type,

'FRT', 'Z',

'A') line\_of\_type\_frt

\*/

, decode(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, decode(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 dummy

, nvl( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

,NULL line\_detail\_id

from ra\_customer\_trx\_lines c

, ra\_customer\_trx\_lines c2

, mtl\_units\_of\_measure u

, mtl\_system\_items msi

where c.customer\_trx\_id = :customer\_trx\_id

and c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

and c.uom\_code = u.uom\_code(+)

and c.inventory\_item\_id = msi.inventory\_item\_id (+)

and msi.organization\_id(+) = :so\_organization\_id

and NVL(c.interface\_line\_context,'X') !='ORDER ENTRY'

and :CP\_TRX\_TYPE\_CLASS ='CM'

-- Added skandru on Feb14-2011 for freight lines ( freight lines are not retrieving due to lot split join conditions )

UNION

SELECT c.customer\_trx\_id line\_customer\_trx\_id

, c.customer\_trx\_line\_id line\_customer\_trx\_line\_id

, DECODE( c2.line\_number,

NULL, c.line\_number,

NULL) line\_number

, c.line\_type line\_type

, NVL(c.translated\_description,c.description) line\_item\_description

, NVL(c.quantity\_ordered,

c.quantity\_invoiced) line\_qty\_ordered

, NVL(c.quantity\_invoiced,

c.quantity\_credited) line\_qty\_invoiced

, u.unit\_of\_measure line\_uom

, c.INTERFACE\_LINE\_ATTRIBUTE3 line\_delivery\_name

, c.INTERFACE\_LINE\_ATTRIBUTE6 line\_so\_line\_id

, c.inventory\_item\_id

, msi.segment1 line\_product\_num

, u.uom\_code line\_uom\_code

, c.unit\_standard\_price line\_standard\_price

, NVL (c.unit\_selling\_price, c.gross\_unit\_selling\_price) /\* Bug 2335596 \*/

line\_unit\_selling\_price

, NVL (c.extended\_amount, c.gross\_extended\_amount)

line\_extended\_amount /\* Bug 2335596 \*/

, c.extended\_amount line\_net\_amount

, c.sales\_order line\_sales\_order

, c.sales\_order\_date line\_sales\_order\_date

, c.tax\_rate line\_tax\_rate

, c.vat\_tax\_id line\_vat\_tax\_id

, c.tax\_exemption\_id line\_tax\_exemption\_id

, c.sales\_tax\_id line\_location\_rate\_id

, c.tax\_precedence line\_tax\_precedence

, NULL lot\_number

, DECODE( TO\_CHAR(c2.line\_number),

NULL, 'N', 'Y') line\_is\_a\_child\_flag

, NVL(c.link\_to\_cust\_trx\_line\_id,

c.customer\_trx\_line\_id) link\_to\_line

, DECODE(c.line\_type,

'LINE', 0,

1) line\_child\_indicator,

NVL(c.link\_to\_cust\_trx\_line\_id , -1) link\_to\_cust\_trx\_line\_id

, DECODE(msi.item\_type,

'FRT', 'A',

'A') line\_of\_type\_frt

, DECODE(c.link\_to\_cust\_trx\_line\_id,

'', c.line\_number ,

c2.line\_number ) order\_by1

, 1 dummy

, NVL( c.amount\_includes\_tax\_flag,'N') line\_tax\_inclusive

,NULL line\_detail\_id

FROM ra\_customer\_trx\_lines c

, ra\_customer\_trx\_lines c2

, mtl\_units\_of\_measure u

, mtl\_system\_items msi

WHERE c.customer\_trx\_id = :customer\_trx\_id

AND c.link\_to\_cust\_trx\_line\_id = c2.customer\_trx\_line\_id(+)

AND c.uom\_code = u.uom\_code(+)

AND c.inventory\_item\_id = msi.inventory\_item\_id (+)

AND msi.organization\_id(+) = :so\_organization\_id

AND NVL(c.interface\_line\_context,'X') ='ORDER ENTRY'

AND c.deferral\_exclusion\_flag = 'Y'

AND :CP\_TRX\_TYPE\_CLASS ='INV'

) t

ORDER BY

DECODE( t.line\_number,NULL,DECODE( t.line\_type,'LINE', 0,

'TAX',8000,

'FREIGHT',9000),DECODE( t.line\_type,'LINE', 0,

'TAX',8000,

'FREIGHT',9000) + t.line\_number )

* + France business would like to collate invoices for printing purposes, at the invoice level. This means that for a list of 10 invoices, 5 copies each, the first invoice will print5 times before moving to printing the second invoice

To fulfill above requirement used BI-Publisher Bursting feature and this Bursting Control file assigned to corresponding data definition.

Printer name and print copies are going to mention in lookup and these lookup values retrieving from formula column(CF\_FRANCE\_PRINTER) query and put it into place holder columns CP\_FR\_PRINTER\_NAME and CP\_FR\_PRINT\_COPIES. Lookup type mentioned in EBS is “XXGIL\_FRANCE\_PRINT\_COPIES”. Print copies number mentioned under Tag column and IPP printer name entered under description column in lookup.

**Bursting Control File :**

<xapi:requestset xmlns:xapi="http://xmlns.oracle.com/oxp/xapi" type="bursting">

<xapi:request select="/TMP003/LIST\_G\_ORDER\_BY/G\_ORDER\_BY/LIST\_G\_INVOICE/G\_INVOICE">

<xapi:delivery>

<xapi:print id="GIL" printer="${CP\_FR\_PRINTER\_NAME}" copies="${CP\_FR\_PRINT\_COPIES}"/>

</xapi:delivery>

<xapi:document output="France Invoice ${TRX\_NUMBER}" output-type="pdf" delivery="GIL">

<xapi:template type="rtf" location="xdo://XXGIL.XXGILARCOMINVSPLIT.fr.FR/?getSource=true">

</xapi:template>

</xapi:document>

</xapi:request>

</xapi:requestset>

Bursting control file will be called by using below code. This is going to write in after report trigger in RDF.

l\_req\_id number;

begin

srw.user\_exit( 'FND SRWEXIT' );

-- Code Added for CR-103b by Suresh Kandru

IF :REPORT\_ORG\_COUNTRY = 'FR' AND :RP\_PRINT\_COPIES!=0 THEN

l\_req\_id := FND\_REQUEST.SUBMIT\_REQUEST('XDO','XDOBURSTREP',NULL,NULL,FALSE,'Y',:P\_CONC\_REQUEST\_ID,'N');

IF l\_req\_id> 0 THEN

srw.message(20002, 'Submitted request\_id ' || l\_req\_id);

COMMIT;

ELSE

srw.message(20002, 'Failed to submit request');

END IF;

END IF;

* + - End of code for CR-103b

1. **OTC -113 -** The invoice print template is to be modified to include the Legal Justification Message from the Tax Rate Code in Oracle E-Business Tax in the invoice footer. The customer VAT number should also be added which is now held in the customers Tax Profile as the customer is liable to pay the tax via the reverse charge process.

Modified template and rdf to display legal justification message from tax rate code and customer VAT number also added to templates

1. **OTC-106 -** Display the different quantities by lot on the invoice when the shipment was done picking the same product from different lots

Modified ITALY rtf template for lot and expiration date columns xml tags and split by lot quantity logic applied to ITALY country. Split by lot quantity logic code already added under CR-103B

1. **OTC -74 –**

Currently discounts and surcharges applied on Orders are combined into single line and shows net amount on the Invoice document. We need to display separate line from the gross amount, and show the discounts on a separate line on the invoice below the WAC price

Gilead charges tax for state of Minnesota for Letairis and Cayston and that also should appear as a separate line on invoice document.

We are displaying discount name and discount value in separate line. If we have discount amount more than ‘0’ and discount percenatage is not null we are displaying line net amount. This information will display after discount lines. Below code is used to get discount lines under G\_LINES query in RDF.

SELECT opa.adjusted\_amount line\_discount,

opa.operand,

qsl.NAME discount\_name,

opa.line\_id,

opa.list\_line\_id,

opa.pricing\_group\_sequence,

opa.arithmetic\_operator

FROM oe\_price\_adjustments opa, qp\_secu\_list\_headers\_vl qsl

WHERE opa.list\_header\_id=qsl.list\_header\_id

AND opa.list\_line\_type\_code IN ('DIS' ,'SUR')

AND opa.line\_id=:line\_so\_line\_id

AND opa.header\_id=:c\_order\_header\_id

AND NVL(opa.applied\_flag,'N')='Y'

AND NVL(opa.accrual\_flag,'N')='N'

order by opa.pricing\_group\_sequence

As part of this CR we have added tax lines by using below code. Added two formula columns (CF\_TAX\_AMT1 and CF\_TAX\_DESC) to this group for currency format purpose.

SELECT tax\_rate\_code,

trx\_line\_id link\_to\_cust\_trx\_line\_id,

tax\_rate tax\_rate,

tax\_amt

FROM zx\_lines\_v

WHERE trx\_id =:customer\_trx\_id

AND trx\_line\_id=:line\_customer\_trx\_line\_id

AND tax\_rate >0

1. **OTC-26**

**Description:** Several European Countries need to show a breakdown of tax by tax code on their Customer invoices. The correct format provided by Oracle is 'European Tax Format'. 'Itemize and Sum' may also be acceptable. Customer Profiles currently set to 'Total Tax Only' -Need to modify current invoice program to allow for 2 or more tax summary lines at the footer.

Added below query to satisfy above requirement

SELECT tax\_rate\_code,

tax\_rate tax\_rate,

sum(tax\_amt) tax\_amount,

SUM(unrounded\_taxable\_amt) taxable\_amt,SUM(unrounded\_tax\_amt) tax\_amt

FROM zx\_lines

WHERE trx\_id =:customer\_trx\_id

AND tax\_rate >0

GROUP BY

tax\_rate\_code,

tax\_rate

1. **CR 547**

A new invoice template for South Korea has been added. This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices-Standard(Print)”.

****

Template details for South Korea:

|  |  |
| --- | --- |
| Template Name | Korea Invoice |
| Code | XXGILARCOMINVS\_KOREA |
| Data Definition | Gilead Commercial Invoices- DD |
| Application | Gilead Custom Application |
| Type | RTF |
| Default File | KOREA.rtf |
| Default File Language | English |
| Default Territory | Republic of Korea |
| Default Output Type | PDF |
| File Name | KOREA.rtf |
| Language | English |
| Territory | Republic of Korea |

1. **CR 1096: Portugal Billing Certification (Commercial Invoice as per Portuguese requirements)**

**Functional Requirement:**

Oracle has released patches to its E-Business Suite Applications (Portuguese Receivables) to comply with the following legislation.

Portuguese Legislation published in the Official Journal, 1. Grade – N. º 120 – 23 June 2010, Decree (Portaria) No. 363/2010 of the 23rd June Series I nº 120, regulates the Billing Software Certification Process.

The patches introduce the following functionality:

* The registration of sales invoices or similar documents recording, through an asymmetric RSA encryption algorithm and a private key known only by the software provider.
* Generate the encrypted signature for Receivables transactions:.
* Generate the signature by concatenating and encrypting the following data elements Invoice Date, System Entry Date, Invoice Number, Invoice Amount, Hash Key (signature of the previously sequenced document).
* Store required signature-related elements for subsequent SAFT reporting (SAFT reporting is not in the scope of this functional design).
* Store the required certification information for subsequent invoice printing;
* **This functional design is for the modification of the existing custom Gilead invoice print program and template used by Portugal so that:**
  + - 1. Four characters from the encrypted signature (references to 1st(a) , 11th(W) , 21st (g) and 31st (C) position, separated by ‘-‘) are printed on the invoice.
      2. Application certificate number preceded by the expression ‚Processado por programa certificado n.º‛ (‘Processed by program with certification number ‘) is printed on the invoice.
      3. A transaction re-print control is added such that the first time a transaction is printed, it should print 2 copies. On the first copy word ‘Original’ must be printed and on the second copy word ‘Duplicado’ must be printed. Next times it is printed: 2nd and 3rd time is printed it should print word ‘2ª via’ and 3ª via’ respectively; After 4th time it should print word ‚Reimpressão‛;
      4. A version of the template to be used in non-production environments which includes the following statement printed at the header level: ‘Documento emitido para fins de Formação‛ ( ‘Document issued for training purposes‛).

#### **SEPA Requirement – Need to change the invoice template to add BIC/IBAN information for SEPA countries , the below attached excel file details on the changes required**

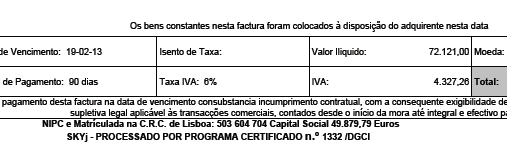


#### Signature printing on Portugal invoice template

The GIL Print Commercial Invoices-Standard(Print) program and Portugal template is altered so the following is printed on the footer of the Portugal invoice template.

**SKYj - PROCESSADO POR PROGRAMA CERTIFICADO n.º 1332 /DGCI**

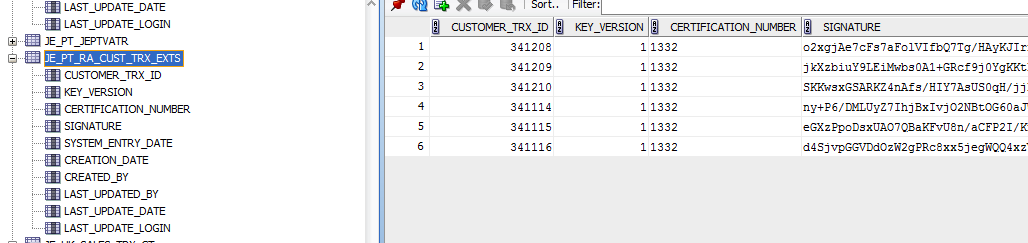
E.g.



**‘SKYj’ are the**  1st , 11th , 21st and 31st characters from the digital signature stored in table JE\_PT\_RA\_CUST\_TRX\_EXTS for each transaction being printed.

E.g.

SKKwsxGSARKZ4nAfs/HIY7AsUS0qH/jjEpbP5SwBcdnWOh0j2r8T8+5fD9Bkzkyjqdkx1He4V/7beaePxoYkMcRT4G/Q6CzaoF5sIkpFzEU5jGwT4XuhM4BetVuCXcMOpp7OFmsbale7ZfK68JcuBrTAgQefufq3qp3oOjzrrxM=



**Reprint Control**

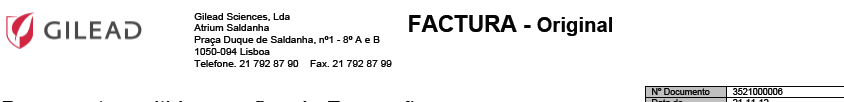
The GIL Print Commercial Invoices-Standard(Print) program should be altered in such a way that it doesn’t impact on any invoice printing carried out by other countries, i.e. changes to the output of the XML generated by the GIL Print Commercial Invoices-Standard(Print) program will not require updates to the invoice templates used by other countries or impact how these countries print invoices.

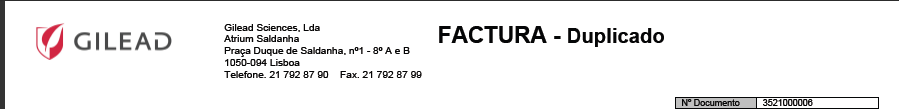
A transaction re-print control is added such that the first time a transaction is printed, it should print 2 copies. On the first copy word ‘Original’ must be printed and on the second copy word ‘Duplicado’ must be printed. Next times it is printed: 2nd and 3rd time is printed it should print word ‘2ª via’ and 3ª via’ respectively; After 4th time it should print word ‚Reimpressão‛.

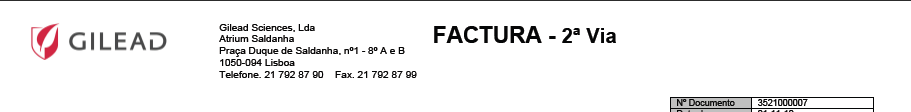
The GIL Print Commercial Invoices-Standard(Print) program should be altered so it works along the same lines as the Oracle “Portugal: Invoice Print Selected Invoices” program, i.e.

1. The first time a PT-OU-01 transaction is printed the XML output will contain two copies of the transaction XML data. One copy of the XML data will be labeled ‘Original’ and the second copy of the XML data will be labeled ‘Duplicado’.
2. The second time a PT-OU-01 transaction is printed the XML output will contain one copy of the transaction XML data. This copy of the XML data will be labeled ‘2ª via’.
3. The third time a PT-OU-01 transaction is printed the XML output will contain one copy of the transaction XML data. This copy of the XML data will be labeled ‘3ª via’.
4. The fourth and subsequent times a PT-OU-01 transaction is printed the XML output will contain one copy of the transaction XML data. This copy of the XML data will be labeled ‘Reimpressão’.

The Portugal Invoice template will be altered to print the wording above to indicate the copy being printed:







**Non Production Version of the template**

A version of the Portugal template will be created for printing Portugal invoices outside of the Production environment. This version of the template will include the following wording: ‘Documento emitido para fins de Formação‛



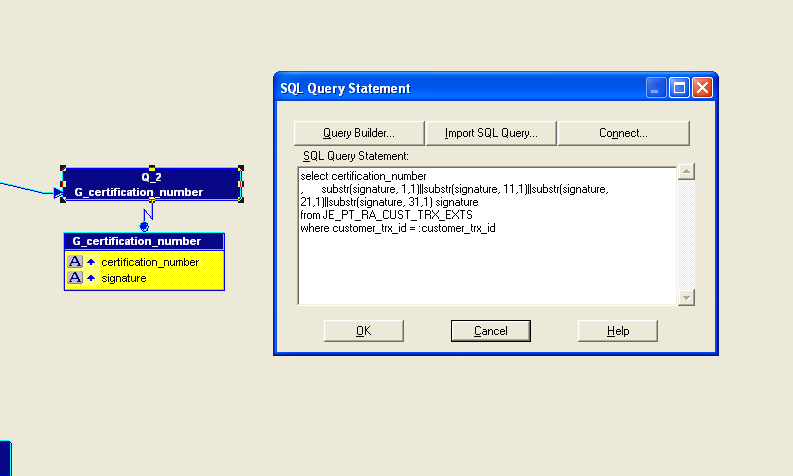
**Technical Design:**

1. Create a new RDF: XXGILARCOMINVPT.rdf that caters to Portuguese localization requirements specified in functional requirements section. XXGILARCOMINVPT.rdf uses XXGILARCOMINV.rdf as base and additional requirements for Portugal have been incorporated.
2. Create a new concurrent program executable XXGILARCOMINVPT with Execution Method as ‘Oracle Reports’ and Execution File Name as ‘XXGILARCOMINVPT’**.**
3. Create a new concurrent program – ‘GIL Print Commercial Invoices-Portugal’ with executable name as ‘XXGILARCOMINVPT’ and set the output format to ‘XML’.
4. Create a new XML Publisher data definition with XXGILARCOMINVPT as code.
5. Create a new XML Publisher template definition using the data definition in step 4.
6. Make changes to PORTUGAL.rtf xml publisher rtf template as per the requirement and upload it to the template definition defined in step 5.
7. Add concurrent program ‘GIL Print Commercial Invoices-Portugal’ to the required request groups.

**Requirement1 : Signature and Certificate number Printing**

This requirement calls for printing the 1st , 11th , 21st and 31st characters from the digital signature stored in table JE\_PT\_RA\_CUST\_TRX\_EXTS for each transaction being printed. This signature appears in the footer sector followed by “**PROCESSADO POR PROGRAMA CERTIFICADO n.º <CERTIFICATE\_NUMER> /DGCI”** where <CERTIFICATE\_NUMBER> is the software certificate number stored in table JE\_PT\_RA\_CUST\_TRX\_EXTS.

Both encrypted signature and software certification number are fetched using query Q\_2(Group: G\_certificate\_number) in XXGILARCOMINVPT.rdf

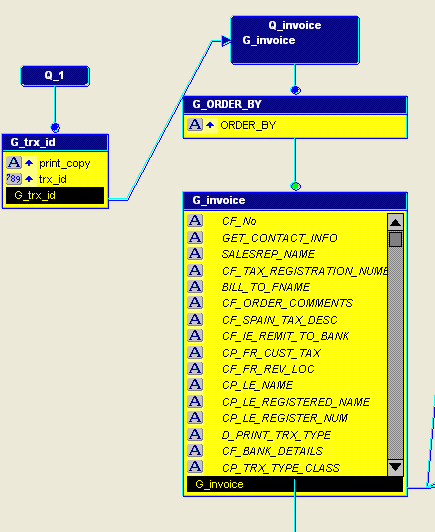


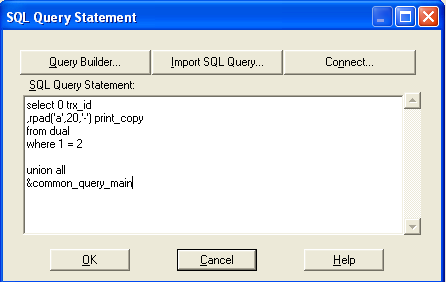
**Requirement2: Reprint Control**

This requirement is covered by building a master query Q\_1 (Group G\_trx\_id).

This query controls the xml stream generated for all possible report submission scenarios. If the invoice is being printed for the first time the query Q\_2 using ra\_customer\_trx\_all.printing\_count field will generate xml stream such that two identical sets of data are produced: one to be stamped as Original (Factura – Original) and other as Duplicate (Factura – Duplicado).

For invoices that have been printed at least once, the xml steam will contain only one data set. In this case based on the printing\_count the invoice prints will be stamped ‘2ª via’ for 2nd copy, 3ª via’ for 3rd copy and Reimpressão‛ for all subsequent copies.

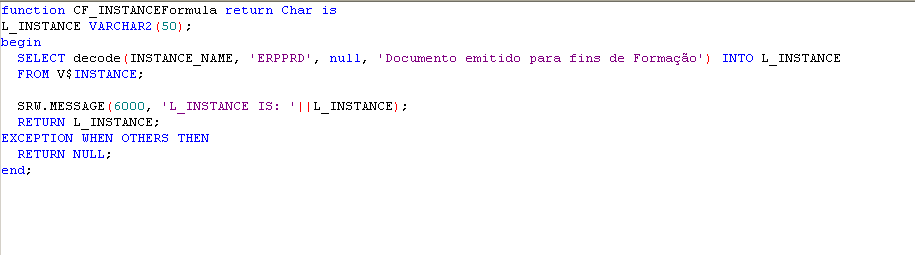




**Requirement 3: Non Production Version of the template**

This requirement calls for printing “Documento emitido para fins de Formação” on the invoices printed from any non production environments.

Formula Column CF\_INSTANCE checks for the instance from which the program was submitted. If instance is ‘ERPPRD’ no message is printed and if instance is not ‘ERPPRD’, above said message will be printed.



REQ0386648-Modified Legal Entity Address change in Switzerland Template

From:

Gilead Sciences Switzerland Sàrl

Turmstrasse 28

CH-6300 Zug

Tel: +41 41 580 02 00 Fax: +41 41 580 02 98

MWST-Nr.: 675 526

UID-Nr.: CHE – 112.939.367

TO:

Gilead Sciences Switzerland Sàrl

Turmstrasse 28

6312 Steinhausen

Switzerland

Tel: +41 41 580 02 00 Fax: +41 41 580 02 98

MWST-Nr.: 675 526

UID-Nr.: CHE – 112.939.367

#### Concurrent Program Executable:

|  |  |
| --- | --- |
| **Executable** | XXGILARCOMINVPT |
| **Short Name** | XXGILARCOMINVPT |
| **Application** | Gilead Custom Application |
| **Description** |  |
| **Execution Method** | Oracle reports |
| **Execution File Name** | XXGILARCOMINVPT |

**Concurrent Program Definition:**

|  |  |
| --- | --- |
| **Program** | GIL Print Commercial Invoices-Portugal |
| **Short Name** | XXGILARCOMINVPT |
| **Application** | Gilead Custom Application |
| **Description** | GIL Print Commercial Invoices - Portugal |
| **Executable Name** | XXGILARCOMINVPT |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | XML |
| **Mls Function** |  |
| **Style** |  |

**Concurrent Program Paramters:**

**Same as “GIL Print Commercial Invoices-Standard(Print)” program.**

**Template details:**

|  |  |
| --- | --- |
| Template Name | Gilead Commercial Invoice - Portugal |
| Code | XXGILARCOMINVPT |
| Data Definition | Gilead Commercial Invoice - Portugal |
| Application | Gilead Custom Application |
| Type | RTF |
| Default File | PORTUGAL.rtf |
| Default File Language | Portuguese |
| Default Territory | Portugal |
| Default Output Type | PDF |
| File Name | PORTUGAL.rtf |
| Language | Portuguese |
| Territory | Portugal |

1. **Turkey e-Invoice Project:**

As part of Turkey e-invoice project, certain pre-identified customers will be enabled for electronic invoicing. These customers will receive Gilead invoices via EDI integration and will not require printing of paper invoices. A new AR transaction type ‘COM e-INV TR-OU-01’ will be created for this purpose.

Commercial invoice for Turkey “GIL Print Commercial Invoices -Turkey” will be modified to have an additional parameter: “Print e-Invoices”.

If Print e-Invoices = ‘Yes’ then the report will print for all Turkish AR transaction types (i.e. print for all customers – e-invoice and non e-invoice enabled).

If Print e-Invoices = ‘No’ then the report will not print for ‘COM e-INV TR-OU-01’ transaction type (i.e. print only for non e-invoice enabled customers).

1. **Turkey e-Invoice Project Phase 2:**

GIL Print Commercial Invoices –Turkey (XXGILARCOMINVTR.rdf)

Change in Turkey Phase2 is to Print Delivery number instead of Sales Order Number. For multiple deliveries, first delivery is to be printed.

Column mapping of field F\_SALES\_ORDER, is changed to CP\_HEAD\_DELIVERY\_NAME.

Also Formula column CF\_HEAD\_SALES\_ORDER is modified adding an order by clause for below query

SELECT DISTINCT TRX\_LINES.sales\_order,

TRX\_LINES.INTERFACE\_LINE\_ATTRIBUTE2 C\_SO\_TYPE,

TRX\_LINES.Sales\_order\_date,

wnd.name DELIVERY\_NAME ,

WND.FOB\_CODE,

-- WND.DELIVERED\_DATE,

-- Wnd.ULTIMATE\_DROPOFF\_DATE,

Wnd.DELIVERY\_TYPE,

Wnd.SHIP\_METHOD\_CODE,

WND.DELIVERY\_ID

INTO l\_sales\_order,

l\_so\_type,

l\_sales\_order\_date,

l\_delivery\_name,

l\_fob\_code,

-- L\_DELIVERED\_DATE,

-- l\_ultimate\_dropoff\_date,

l\_delivery\_type,

l\_ship\_method ,

l\_delivery\_id

FROM ra\_customer\_trx\_lines TRX\_LINES,

WSH\_NEW\_DELIVERIES wnd

WHERE TRX\_LINES.INTERFACE\_LINE\_CONTEXT='ORDER ENTRY'

AND TRX\_LINES.customer\_trx\_id = :customer\_trx\_id

AND TRX\_LINES.INTERFACE\_LINE\_ATTRIBUTE3 = WND.NAME

ORDER BY wnd.name asc;

1. **CR#1441 Prohibition On Export Terms on Canadian Invoices**

**Functional Requirement :**

To print prohibition on export terms note on the Invoices when any item is shipped to Canada Customers from US-OU-01 Operating Unit.

* Need to modify the existing US Invoice template, USA.rtf
* For any item sold from US-OU-01 Operating Unit, to Canada (Ship To Country), print the Prohibition on export terms note in the lines.

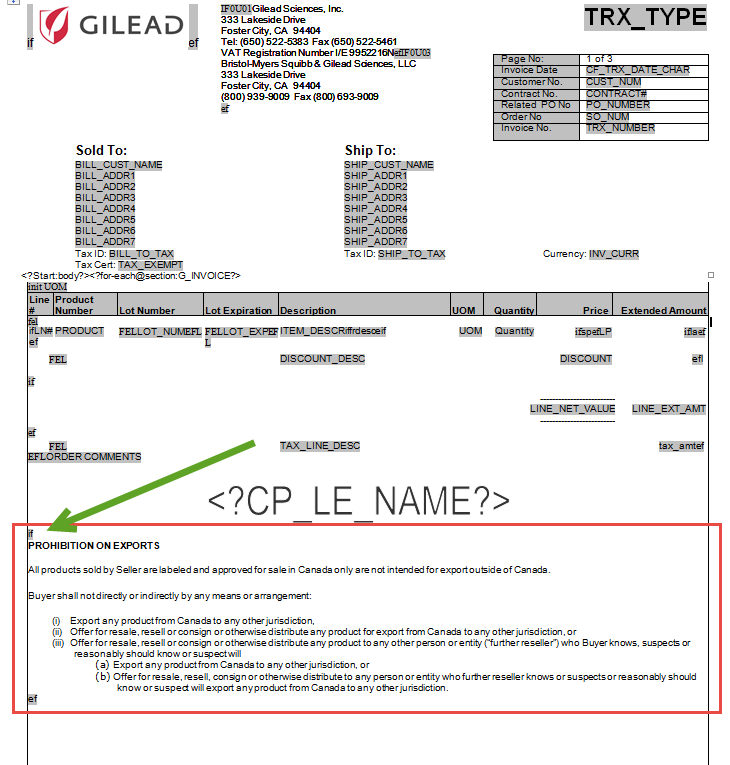
**This functional design is for the modification of the existing custom Gilead invoice print program, GIL Print Commercial Invoice (Standard) and template, USA.rtf used by US-OU-01 so that:**

1. For all Canadian customers, the Prohibition on Export Terms will be printed on the invoices at line level.
2. **Added IF condition statement to the USA.rtf template to print the Prohibition on Export terms note :**

**If**

**<?if: (../../../../RP\_ORG\_NAME='US-OU-01') and (SHIP\_COUNTRY='CA')?>**

**ef**



1. **CR 1406**

New invoice template file is added for Hong Kong. This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices-Standard(Print)”.

The existing Hong Kong template was modified as per the TW SG SK EBS Implementation project requirement. The new requirement was to change the rtf layout to remove the shipping header lines and move the Delivery Note No. field to the right hand header table.

The modified template is attached below

****

Template details for Hong Kong:

|  |  |
| --- | --- |
| File Name | HONGKONG.rtf |
| Language | English |
| Territory | Hong Kong |

1. CR 976: Ireland Invoice of GIL Print Commercial Invoices-Standard (Print) requirement.

Functional Requirement:

Dynamically changing Company’s Address, VAT Registration number, Company’s Bank Details and Authorization code based on operating unit in which the Invoices are created.

Conversion of VAT amount into EURO currency by using GL Daily rates for Invoice Date.

Printing additional text in Invoice print for Zero VAT when Tri-Supply Limited issues invoices to Gilead Sciences Limited.

The Business requirement can precisely be displayed in below table.

| **Seq** | **Invoice From** | **Invoice To** | **Requirement Details** |
| --- | --- | --- | --- |
| 1 | GSL | Tri-Supply | Current GSL template + (VAT amounts in €, address, VAT number) |
| 2 | GSL | BMS | Current GSL template + (VAT amounts in €, address, VAT number) |
| 3 | Tri-Supply | GSL | Current GSL template + (Tri-Supply company details, bank details, authorization number) |
| 4 | Tri-Supply | GSL | Below should be printed in invoice after Invoice Lines.  **Zero rate supply – reference VAT13B/56B authorisation number: 90/14410/10 2015** |
| 5 | Tri-Supply | BMS | Current GSL template + (Tri-Supply company details, bank details, authorization number |
| 6 | Tri-Supply | BMS Company | Current GSL template + (Tri-Supply company details, bank details, authorization number |
| 7 | BMS | GSL | Current GSL template + (BMS company details, bank details, authorization number)  **Zero rate supply – reference VAT13B/56B authorisation number: 90/14410/10 2015** |
| 8 | BMS | BMS Company | Current GSL template + (BMS company details, bank details, authorization number) |
| 9 | BMS | Tri-Supply | Current GSL template + (BMS company details, bank details, authorization number) |

**Technical Design:**

1. Modify the RDF: XXGILARCOMINVPT.rdf with new formula column for VAT amount in EURO requirements specified in functional requirements section.
   1. If Invoice currency is not in EURO
   2. And VAT Amount is greater than zero
   3. And sold to country is Ireland
2. Make changes to display VAT amounts in €, address, VAT number to IRELAND.rtf xml publisher rtf template as per the requirement and upload it.
3. CR 1481 REPORTING OF PURCHASE DATA and BREACH OF TERMS OF SALE terms on Canadian invoices.

New invoice template file with the updated verbiage is added for US Invoices(For Canada Customers). This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices-Standard(Print)”.

Template details for US Invoice:



File Name USA.rtf

Language English

1. **CR1448 :Spain\_Legal\_Requirement\_for\_Addition\_3\_fields\_Invoice**

**Business Need**

Three new fields are required to be added in current invoice format of Spain Invoice Print report. These fields will contain customer specific information for the public administration to register and account the invoice.

**Technical Design**

1. Existing rdf XXGILARCOMINV, is modified to add 3 new fields as below

|  |  |
| --- | --- |
| **Fields** | **Mapping** |
| Organo Gestor | APPS.HZ\_CUST\_SITE\_USES\_ALL.ATTRIBUTE25 |
| Oficina Contable | APPS.HZ\_CUST\_SITE\_USES\_ALL. ATTRIBUTE24 |
| Unidad Tramitadora | APPS.HZ\_CUST\_SITE\_USES\_ALL. ATTRIBUTE23 |

Formula column CF\_BILL\_ATTRIBUTE is added to G\_invoice group.

Below is the code for the same.

function CF\_BILL\_ATTRIBUTEFormula return Char is

lv\_attribute23 VARCHAR2(240);

lv\_attribute24 VARCHAR2(240);

lv\_attribute25 VARCHAR2(240);

begin

:CP\_BILL\_ATTRIBUTE23 :=NULL;

:CP\_BILL\_ATTRIBUTE24 :=NULL;

:CP\_BILL\_ATTRIBUTE25 :=NULL;

srw.message(1001,'Bill\_to\_site\_use\_id '||:BILL\_TO\_SITE\_USE\_ID);

SELECT attribute23,

attribute24,

attribute25

INTO

lv\_attribute23,

lv\_attribute24,

lv\_attribute25

FROM HZ\_CUST\_SITE\_USES

WHERE site\_use\_id=:BILL\_TO\_SITE\_USE\_ID;

:CP\_BILL\_ATTRIBUTE23 :=lv\_attribute23;

:CP\_BILL\_ATTRIBUTE24 :=lv\_attribute24;

:CP\_BILL\_ATTRIBUTE25 :=lv\_attribute25;

srw.message(1001,'CP\_BILL\_ATTRIBUTE23 '||:CP\_BILL\_ATTRIBUTE23);

srw.message(1001,'CP\_BILL\_ATTRIBUTE24 '||:CP\_BILL\_ATTRIBUTE24);

srw.message(1001,'CP\_BILL\_ATTRIBUTE25'||:CP\_BILL\_ATTRIBUTE25);

RETURN 'Y';

EXCEPTION

WHEN OTHERS THEN

:CP\_BILL\_ATTRIBUTE23 :=NULL;

:CP\_BILL\_ATTRIBUTE24 :=NULL;

:CP\_BILL\_ATTRIBUTE25 :=NULL;

srw.message(1001,'Error in CF\_BILL\_ATTRIBUTEFormula '||SQLERRM);

RETURN 'N';

end;

1. Also Spain.rtf is modified to add the above three fields below bill\_to address.
2. **CR1462 :Spain e-Invoicing**

**Business Need**

Enhance ‘GIL Print Commercial Invoices-Standard (Print)’ to stop printing transactions belonging to customers enabled for electronic invoicing.

**Technical Design**

1. Concurrent program ‘GIL Print Commercial Invoices-Standard(Print)’ is modified to have an additional parameter: 'Print eInvoices' with the default value as ‘No’
2. XXGILARCOMINV.rdf is modified to add new parameter ‘P\_PRINT\_EINVOICES’
3. XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG package is modified as below
   1. Rename ‘build\_where\_clause procedure’ to ‘build\_where\_clause\_standard’
   2. Add new parameter ‘p\_print\_einvoices’ to build\_where\_clause\_standard
   3. Create a new wrapper procedure ‘build\_where\_clause’which will call ‘build\_where\_clause\_standard’ by passing p\_print\_einvoices parameter as ‘Y’. This is to ensure any existing objects referring to existing procedure ‘build\_where\_clause’ will continue to work as per the existing functionality.
4. Logic for ‘build\_where\_clause\_standard’ procedure

For an eInvoice customer, Attribute1 on the 'Customer Credit Profile Information' DFF will be ‘Invoice Only’ or ‘All AR transactions’ and Invoice creation date has to be greater than or equal to the value in Attribute2 column (AR EDI Effective Date) on the 'Customer Credit Profile Information' DFF.

* If p\_print\_einvoices = ‘Yes’ then the report will print all transactions as per the existing functionality
* If p\_print\_einvoices = ‘No’ then the report will not print relevant transactions for eInvoice enabled customers
  + If the Attribute1 (Invoice Electronic Submission) is ‘Invoice Only’ and the Invoice creation date is greater than or equal to Attribute2 column (AR EDI Effective Date) on the 'Customer Credit Profile Information' DFF, then Invoice Print program will exclude Invoice transactions for the print. Other transactions will still be printed.
  + If the Attribute1 (Invoice Electronic Submission) is ‘All AR transactions’ and the Invoice creation date is greater than or equal to Attribute2 column (AR EDI Effective Date) on the 'Customer Credit Profile Information' DFF, then Invoice Print program will not print any transactions.

Additional logic for Spain only:

If the transaction has a different ship-to and bill-to customers, and the attribute6 on ‘Customer Credit Profile Information’ DFF for bill-to customer is ‘ES\_FACTURAE3.2’ then the invoice print program will only validate bill-to customer to verify if it is an eInvoice customer.

If the transaction has a different ship-to and bill-to customers, and the attribute6 on ‘Customer Credit Profile Information’ DFF for bill-to customer is ‘ES\_EDIFACT’ or NULL then the invoice print program will validate both the bill-to and ship-to customer to verify if it is an eInvoice customer.

1. **CR 1543 e-invoice for Italy**

**Business Need**

Enhance ‘GIL Print Commercial Invoices-Split (Print)’ will be modified to add 2 new fields CIG code and IPA code. The IPA code will be on Invoice header section just below the field ‘/Cod. Fiscale’ and CIG code will be at line level of report layout.

**Technical Design**

1. Modify the RDF: XXGILARCOMINVSPLIT.rdf with 2 new formula column for IPA code at invoice header and CIG code at line level.
2. Formula column CF\_IPA\_CODE add to G\_INV\_TERM group

function CF\_IPA\_CODEFormula return Char is

l\_ipa\_code VARCHAR2(240);

l\_err\_msg VARCHAR2(4000);

l\_sales\_order ra\_customer\_trx.ct\_reference%TYPE;

BEGIN

BEGIN

SELECT ooh.attribute2

INTO l\_ipa\_code

FROM oe\_order\_headers ooh

WHERE ooh.order\_number = :CF\_HEAD\_SALES\_ORDER

AND ooh.Context = '113';

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

BEGIN

SELECT attribute3

INTO l\_ipa\_code

FROM ra\_customer\_trx\_all

WHERE customer\_trx\_id = :customer\_trx\_id

AND attribute\_category = '113';

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

srw.message (5000, 'DEBUG: IPA code error "no data found" ' );

l\_ipa\_code := '';

END;

END;

return (l\_ipa\_code);

EXCEPTION

WHEN OTHERS THEN

l\_err\_msg := sqlerrm;

srw.message (5000, 'DEBUG: CF\_IPA\_CODEFormula - Error :- '||l\_err\_msg);

l\_ipa\_code := '';

END;

1. Formula column CF\_CIG\_CODE add to G\_LINES group

function CF\_CIG\_CODEFormula return Char is

l\_cig\_code VARCHAR2(240);

l\_err\_msg VARCHAR2(4000);

l\_sales\_order ra\_customer\_trx.ct\_reference%TYPE;

l\_sales\_order\_id oe\_order\_headers.header\_id%TYPE;

l\_Blanket\_Number oe\_order\_headers.Blanket\_Number%TYPE;

BEGIN

BEGIN

SELECT obl.cust\_po\_number

INTO l\_cig\_code

FROM oe\_order\_headers ooh,

oe\_order\_lines\_all oel,

oe\_blanket\_headers\_all Obh,

oe\_blanket\_lines\_all obl

WHERE oel.line\_id = :line\_so\_line\_id

AND oel.header\_id = ooh.header\_id

AND obh.order\_number = ooh.Blanket\_Number

AND obl.header\_id = obh.header\_id

AND oel.inventory\_item\_id = obl.inventory\_item\_id

AND ooh.order\_number = :CF\_HEAD\_SALES\_ORDER;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

BEGIN

SELECT attribute1

INTO l\_cig\_code

FROM ra\_customer\_trx\_Lines\_all

WHERE customer\_trx\_id = :customer\_trx\_id

AND customer\_trx\_line\_id = :line\_customer\_trx\_line\_id

AND attribute\_category = 'IT-OU-01';

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

srw.message (5000, 'DEBUG: CIG\_CODE error "no data found" for '||:LINE\_PRODUCT\_NUM||' So Line ID: '||:line\_so\_line\_id);

l\_cig\_code := '';

END;

END;

return (l\_cig\_code);

EXCEPTION

WHEN OTHERS THEN

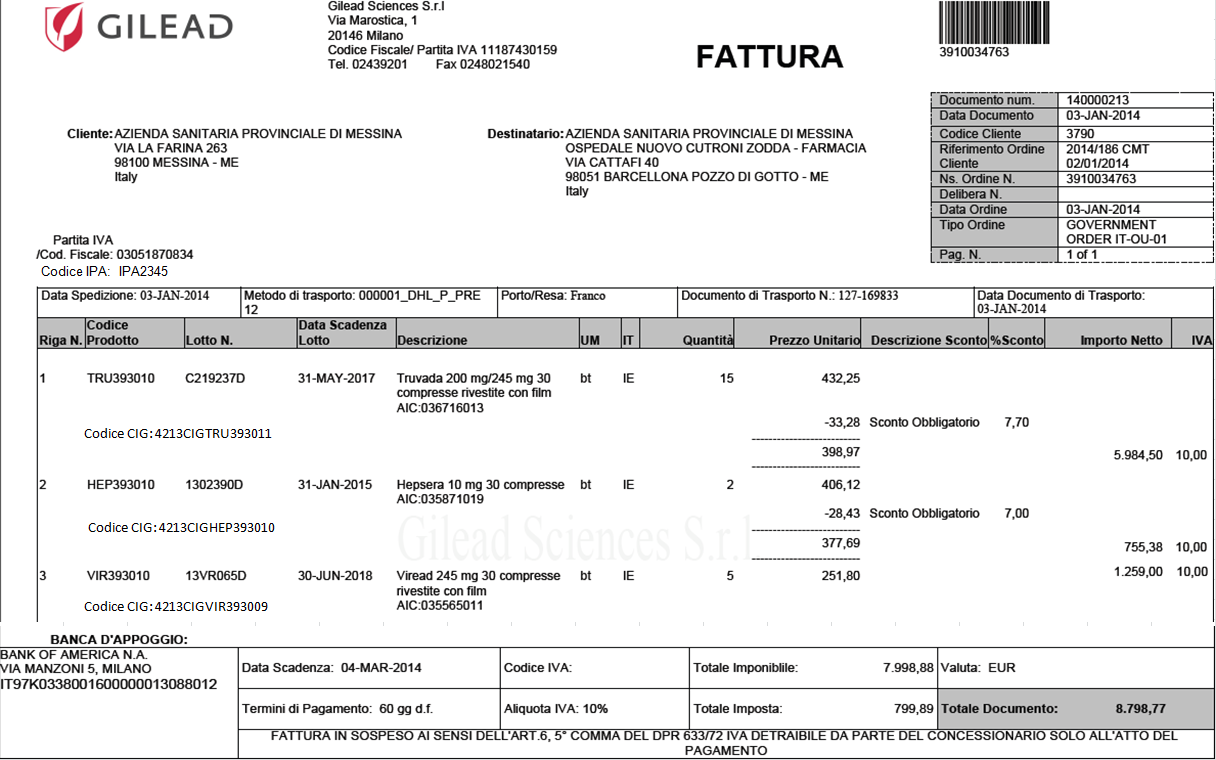
l\_err\_msg := sqlerrm;

srw.message (5000, 'DEBUG: CF\_CIG\_CODEFormula :'||:LINE\_PRODUCT\_NUM||' So Line ID: '||:line\_so\_line\_id ||' Error :- '||l\_err\_msg);

l\_cig\_code := '';

END;

1. Modified ITALY.rtf to display IPA and CIP code.



1. **REQ0114682**

Modification is done on the invoice template for Germany.As per the user request . They want to modify the text at the template level for germany only .This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices and GIL Print Commercial Invoices-Split(Print)

Correct Imprint:

"Gilead Sciences Ireland UC: Großhandelsbetriebserlaubnis No. W00442/00001 wurde am 07.04.2016 durch die Health Products Regulatory Authority (HPRA) erteilt"

 Current Imprint:

"Gilead Sciences Ltd.: Herstellungserlaubnis No. M589 wurde am 14. März 2005 durch das Irish Medicines Board erteilt."



1. VAT Compliant Project, project # 206330.

There is a business requirement to enhance concurrent program invoice print program to support multiple tax registrations for intercompany account receivables transactions for countries Ireland and US respectively. Further to this, the US invoice print for intercompany transactions will be enhanced to display VAT related fields similar to that of Ireland. There is also a business requirement to display Ship from Address for all intercompany transactions, this requirement is specific to US and Ireland.

1) Enhancement of ‘GIL Print Commercial Invoices-Standard (Print)’ ’ w.r.t to country US.

Report will display new VAT related fields when ALL below conditions are true.

1.1) Operating unit is ‘US-OU-01’ or ‘US-OU-03.

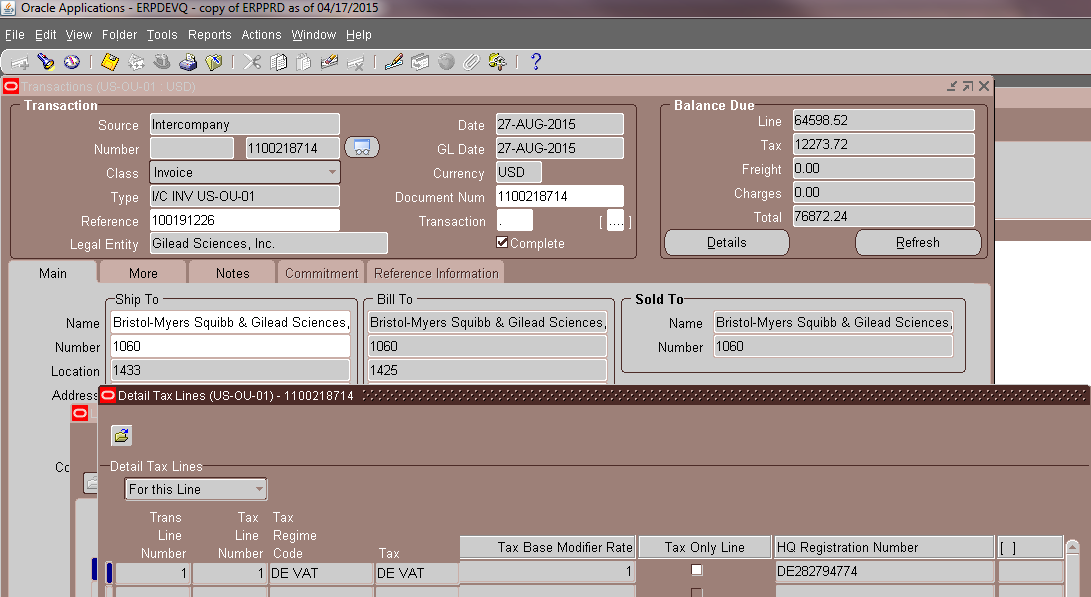
Existing formula column “RP\_ORG\_NAME” will be used to determine the OU name.

1.2) AND a EU country VAT code applied on AR Transaction (the country code will be derived from the tax code of the invoice).

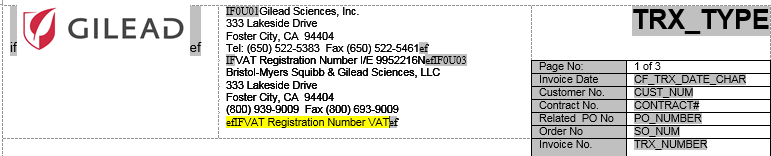
A new function CF\_VAT\_US has been created in XXGILARCOMINV.rdf to validate the above conditions and determine whether the invoice is VAT complaint or not. Function also checks if respective country exists in 'EUROPEAN ECONOMIC COMMUNITY'.

Below customizations will be enabled only when the newly created function CF\_VAT\_US validates true.

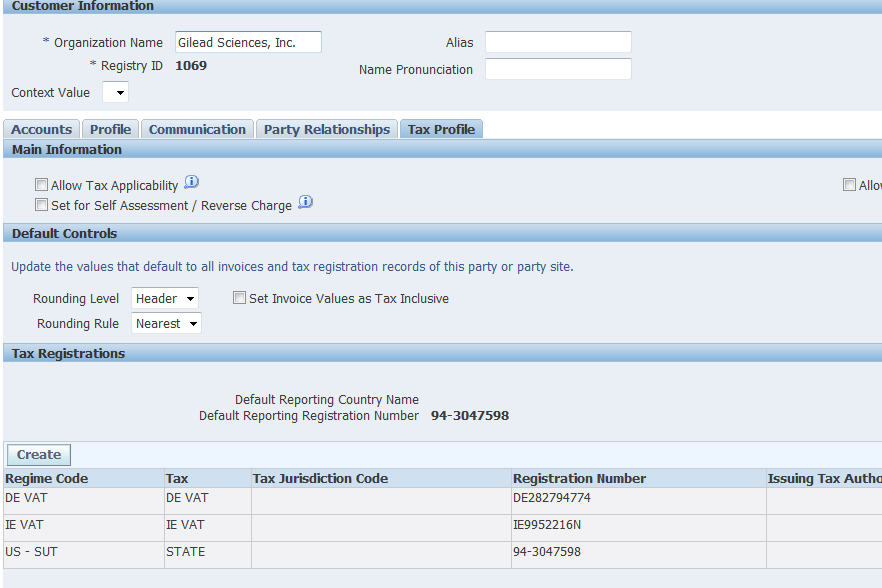
1.3) Display VAT registration number for Legal Entities ‘Gilead Sciences, Inc.’ or ‘Bristol-Myers Squibb & Gilead Sciences, LLC’ .



The value under HQ registration number will be picked up and displayed as VAT registration number, the existing hardcoded value will not be displayed in this case. A new formula column CF\_VAT\_REG\_NUM has been added to derive the HQ registration number. The value in column “hq\_estb\_reg\_number” will be picked up from the table zx\_lines\_v by passing customer\_trx\_id and trx\_number as the parameters.

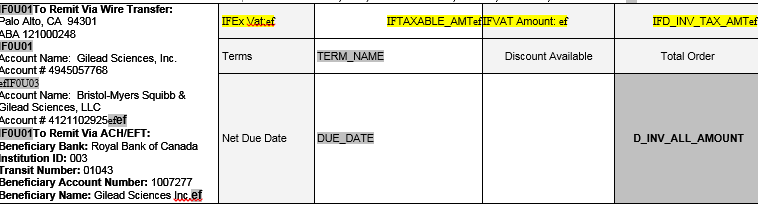


1.4) New logic will be introduced to derive tax registration number of trading partner (value of fields BILL\_TO\_TAX and SHIP\_TO\_TAX respectively). These values will be derived from customer setup ‘Tax Profile’ tab based on ship to country.



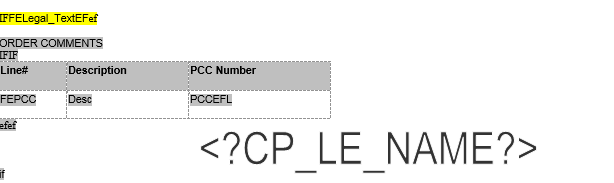
The registration number will be derived by passing the customer information (party id) and corresponding regime code. The country code and ship to customer details will be passed to profile tables to derive the registration number.

1.4) Field TAXABLE\_AMT (prompt ‘Ex Vat’) & D\_INV\_TAX\_AMT (prompt ‘Vat Amount’), will be added similar to that of Ireland template.XML tags D\_INV\_ITEM\_AMOUNT & D\_INV\_TAX\_AMOUNT used in Ireland template will be mapped to these newly added fields.

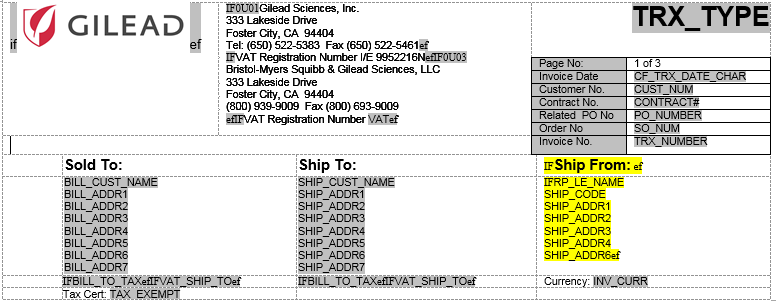


1.5) The Legal Message associated with the AR transaction line will be printed on US invoice print, this is a existing functionality for Ireland invoice print and same logic will be used for US invoice print.Field “REPORTING\_CODE\_NAME” from the existing .rdf (used in Ireland template) will be mapped to this newly added field.

In RTF template USA.rtf legal message will be displayed just above ‘Order Comments’ field (see below screenshot).

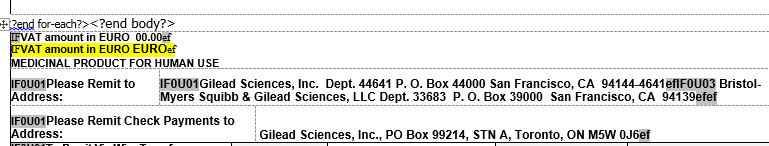


1.6) Enhance US Invoice print to display Ship from address. Enhance the USA.rtf file to create new fields for ship from address as shown below. Ship from information will be obtained from the shipping warehouse information (Column “INTERFACE\_HEADER\_ATTRIBUTE3” in ra\_customer\_trx\_all). The location code, address information will be derived from hz\_locations\_all by passing inventory organization details, a place holder column CP\_SHIP\_FRM\_CODE holds the address details.

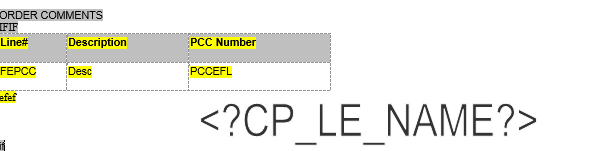


The ship from details will be displayed only if the newly added formula column CF\_VAT\_ADD\_US validates to true (This formula column validates the OU and order type)

1.7) A new formula column VAT\_EUR\_AMT has been added to derive the euro amount of the transaction. Exchange rates are derived from gl\_daily\_rates for the corresponding invoice date.



1.8) PCC Transactions details are displayed in the invoice print just below the order comments as shown below. PCC number is derived from attribute13 of AR invoice lines table.Customer\_trx\_id and org\_id are passed to the lines table to retrieve the PCC number A new group q\_g\_line\_number1 has been added to derive the below details.



2) Enhancement of Invoice print w.r.t to Ireland intercompany Invoice print.

Concurrent program ‘GIL Print Commercial Invoices-Standard (Print)’ will display new VAT related fields when ALL below conditions are true. The newly created function CF\_VAT\_IE will be used to validate the transaction.

ISO Logic

1) Operating unit is ‘IE-OU-01' or'IE-OU-02’.

Existing formula column “RP\_ORG\_NAME” will be used to determine the OU name.

2) If it is an INTERCOMPANY transaction.

3) If the order is an internal sales order i.e. order\_id =10.

4) AND a EU country VAT code applied on AR Transaction

OR

ESO with ORDER TYPE %API%

1) Check the order type , this will exist in our custom lookup - XXGIL\_ORD\_TYPE\_NEW\_AR\_DOC\_SEQ

OR

MANUAL AR TRANSACTIONS

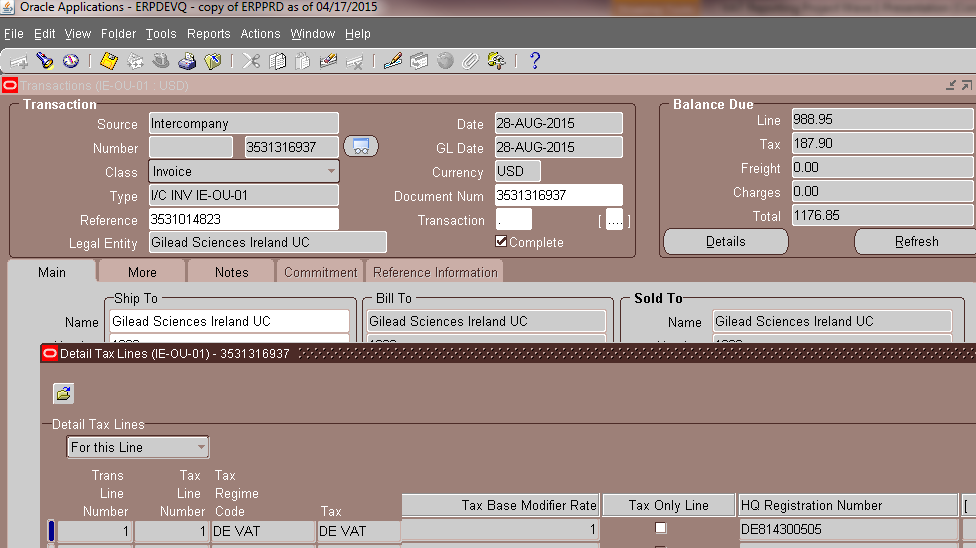
1) Operating unit is ‘IE-OU-01' or'IE-OU-02’

2) AND AR Transaction Source is ‘MANUAL-OTHER’

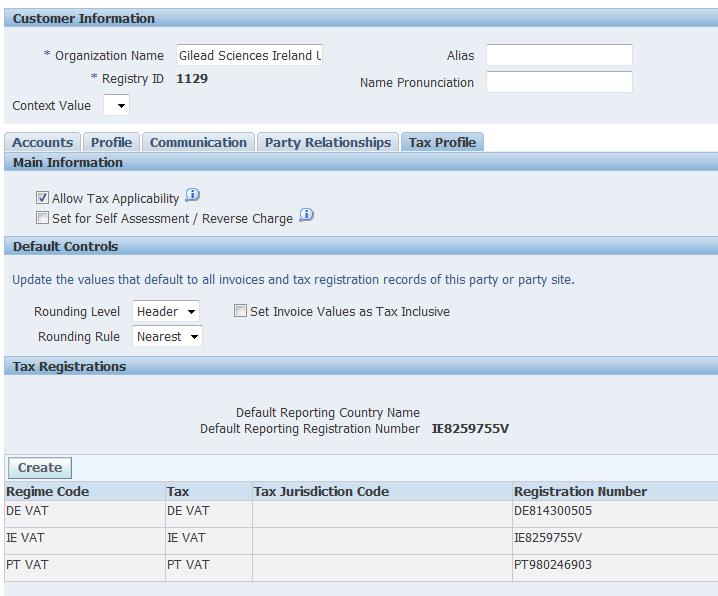
3) AND an EU country VAT code applied on AR Transaction.

4) AND Bill to Customer Account type will be ‘Internal’

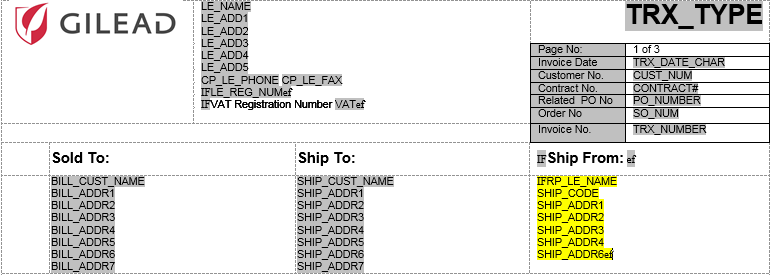
2.1) Display VAT registration number for Legal Entities ‘Gilead Sciences Ireland UC’ or ‘Tri-Supply Limited’ respectively depending upon operating unit where account receivables transaction exists. The logic developed from US template will be used for Ireland too.



2.2) New logic will be introduced to derive tax registration number of trading partner (value of fields BILL\_TO\_TAX and SHIP\_TO\_TAX respectively). Ex: If ‘Ship to’ country is ‘IE’ then registration number for ‘Gilead Sciences Ireland UC’ will be ‘IE8259755V’.

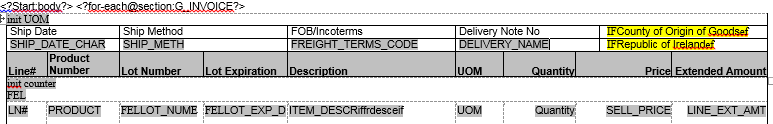


2.3) Enhance Ireland Invoice print to display Ship from address. Ship from address will be the physical location of inventory organization where materials are shipped from. Enhance the Ireland.rtf file to create new fields for ship from address.The address details will be displayed only when the newly added formula column CF\_VAT\_ADD\_IE validates true.



2.4) PCC Transactions details are displayed in the invoice print just below the order comments as shown below. Same logic will be used as explained above in US template customization.

2.5) The field country of origin will be hidden if it is a VAT complaint invoice.



3) Generic enhancement of ‘GIL Print Commercial Invoices-Standard (Print)’ for intercompany Invoice print.

3.1) For all intercompany transactions, invoice print will display the ‘Ship To’ address. This is applicable for both, intercompany transactions for internal sales orders and direct distribution orders. The ‘Ship To’ address will be displayed where goods are shipped i.e ship address from sales order.

Following changes are done in the RDF to derive the ship to address.

3.1.1) SHIP\_TO\_CUSTOMER\_ID and SHIP\_TO\_SITE\_USED\_ID are derived from OE\_ORDER\_HEADERS\_ALL by passing order number, the same query validates for intercompany invoices as explained below.

nvl(upper(A.interface\_header\_context),''INTERCOMPANY'') = ''INTERCOMPANY'' )

3.1.2) A new union query has been added to q\_ship\_customer (data model) to derive the address details.In the new union query \_ALL tables were used to derive the shipping information.(hz\_cust\_acct\_sites\_ALL & hz\_cust\_site\_uses\_ALL )

:SHIP\_TO\_SITE\_USE\_ID & :SHIP\_TO\_CUSTOMER\_ID which was derived as explained above will be passed as a parameter to retrieve the shipping to details for the corresponding transaction.

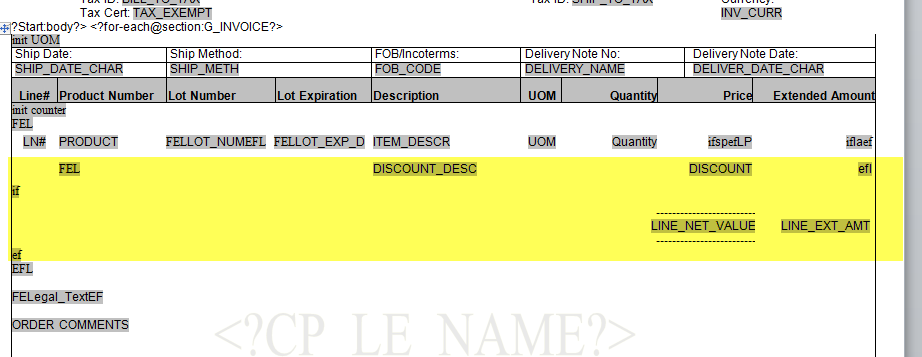
Template details for South Korea:

|  |  |
| --- | --- |
| Template Name | Korea Invoice |
| Code | XXGILARCOMINVS\_KOREA |
| Data Definition | Gilead Commercial Invoices- DD |
| Application | Gilead Custom Application |
| Type | RTF |
| Default File | KOREA.rtf |
| Default File Language | English |
| Default Territory | Republic of Korea |
| Default Output Type | PDF |
| File Name | KOREA.rtf |
| Language | English |
| Territory | Republic of Korea |

1. **SR#REQ0130219: Newzeland template should display discounts**

**New zeland is going to start providing discounts for the the customers from Jul 1,2016 and they requested to show discount applied on the sales order line for Newzeland to display correct amount on the invoice print.**

**NEWZELAND.rtf is modified to display discount modifier and amount details as shown below.**



1. **A New template is required for printing Japan Invoices**

**Functional Requirement:**

A new invoice template for Japan has been added. This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices-Standard(Print)”.

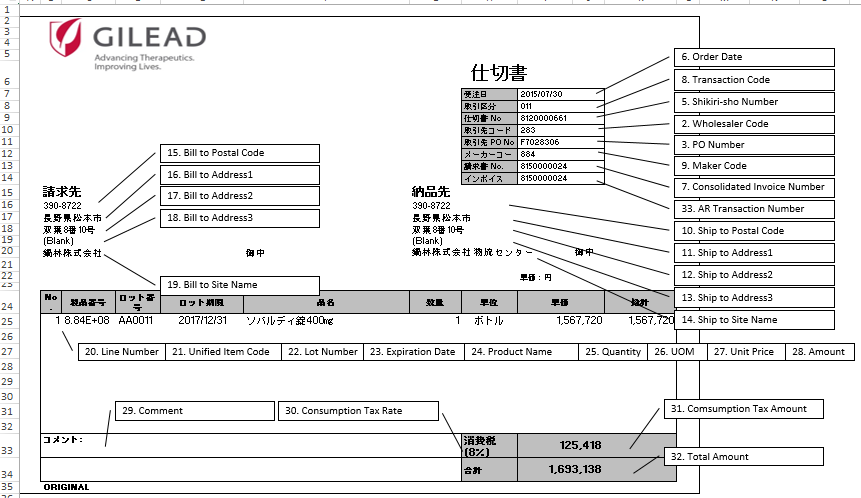
Modify the existing RDF XXGILARCOMINV.rdf to include the new fields and changes required for the Japanese commercial invoice report.

A new template:”Japan Invoice” is created for the above program.

New fields:

* + 1. Transaction code
    2. Shikiri-sho number
    3. Wholesaler code
    4. Maker code
    5. Consolidated Invoice No.
    6. AR Transaction number
    7. Unified item code
    8. Product name in Japanese (Kanji)
    9. UOM code for Japan
    10. Comments
    11. Consumption tax rate
    12. Consumption tax amount

Register the new template for the Japanese commercial invoice report “GIL Print Commercial Invoices-Standard(Print)” and file name “Japan.rtf”. The template will be registered under territory Japan and language Japanese.



Template details for Japan:

|  |  |
| --- | --- |
| Template Name | Japan Invoice |
| Code | XXGILARCOMINVJP |
| Data Definition | Gilead Commercial Invoices- DD |
| Application | Gilead Custom Application |
| Type | RTF |
| Default File | JAPAN.rtf |
| Default File Language | Japanese |
| Default Territory | Japan |
| Default Output Type | PDF |
| File Name | JAPAN.rtf |
| Language | Japanese |
| Territory | Japan |

**Technical Design:**

1. Update an existing RDF: XXGILARCOMINV.rdf that caters to Japanese localization requirements specified in functional requirements section.
2. Existing concurrent program – ‘GIL Print Commercial Invoices-Standard(Print)’ will be able to access the new template attached
3. Update the existing data definition with the new template and define it for Japan territory

Make changes to JAPAN.rtf xml publisher rtf template as per the requirement and upload it to the template definition updated in step 3.

### Executable File Location

The files used by this Report reside at the following location:

$XXGIL\_TOP/reports/US

## Incompatibilities

None

## Performance Considerations

NA

## Restart/Recovery Procedures

No special steps to be taken in case of crash. Resubmit the report if it errors out

<Mention crash recovery steps if any, in this section mention about steps how to resubmit the report if report errors out. Note: This is very important incase report is generated from temporary table and / or storing the process out put data in table

1. **REQ0189897**

Modification is done on the invoice template for ITALY .As per the user request . They want to modify the text at the template level for Italy only .This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices and GIL Print Commercial Invoices-Split(Print)

Correct Imprint:

" Gilead Sciences S.r.l

via Melchiorre Gioia

26 20124 Milano

Codice Fiscale/ Partita IVA 11187430159

Tel. 02439201 Fax 0248021540."

“

 Current Imprint:

" Gilead Sciences S.r.l

Via Marostica, 1

20146 Milano

Codice Fiscale/ Partita IVA 11187430159

Tel. 02439201 Fax 0248021540."



1. **REQ0184923**

Modification is done on the invoice template for NORWAY .As per the user request . They want to modify the text at the template level for Norway only .This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices and GIL Print Commercial Invoices-Standard(Print)

Correct Imprint:

Gilead Sciences Norway AS

C.J. Hambros Plass 2 C

0164 Oslo

Norway

Tel: +46-8-505 718 00

VAT Reg Number: 991461841MVA

Current Imprint :

Gilead Sciences Norway AS

Jernbanegatan 8

2004 Lillestrom

Norway

Tel: +46-8-505 718 00

VAT Reg Number: 991461841MVA



1. **INC0467041**

Updated KOREA.rtf file to remove the decimal units in all the price/amount fields. Korea business has informed to remove the decimal units from the commercial print invoice as there are no decimal units for KOREA business.

Added below code to remove decimal units from amount/price fields.

<?xdofx:format\_number(replace(D\_LINE\_EXTENDED\_AMOUNT,',',''), '##,##0')?>

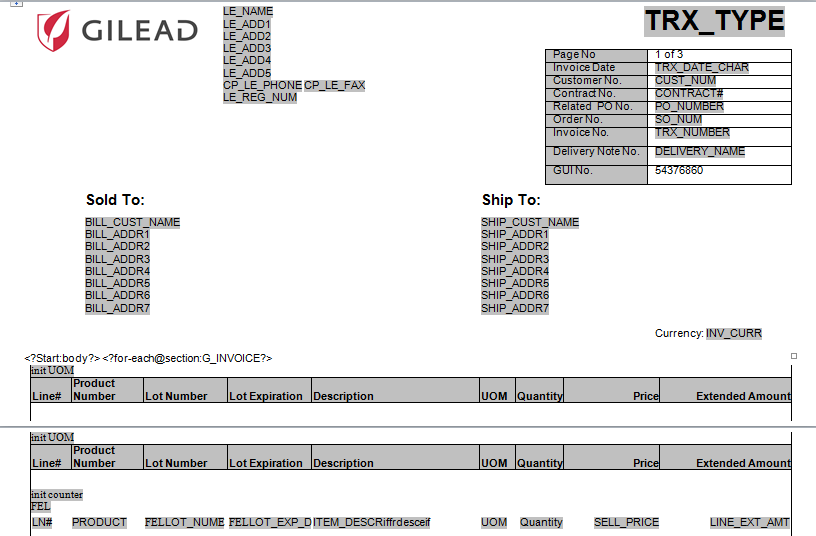


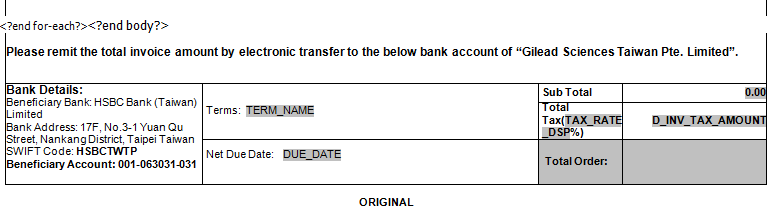
1. **TW\_SG\_SK EBS Implementation Project – Taiwan Rollout**

**Functional Requirement:**

A new invoice template for Taiwan has been created as per the requirement for TW SG SK Implementation project. As per the requirement the invoice template has been kept similar to existing Hong Kong invoice template (mentioned in point no. 13 ).

Register the new template for the Taiwan’s commercial invoice report “GIL Print Commercial Invoices-Standard (Print)” and file name “TAIWAN.rtf”. The template will be registered under territory Taiwan and language English





New invoice template file is added for Taiwan. This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices-Standard (Print)”.

****

Template details for Taiwan:

|  |  |
| --- | --- |
| Template Name | TAIWAN Invoice |
| Code | XXGILARCOMINVS\_TAIWAN |
| Data Definition | Gilead Commercial Invoices- DD |
| Application | Gilead Custom Application |
| Type | RTF |
| Default File | TAIWAN.rtf |
| Default File Language | English |
| Default Territory | Taiwan |
| Default Output Type | PDF |
| File Name | TAIWAN.rtf |
| Language | English |
| Territory | Taiwan |

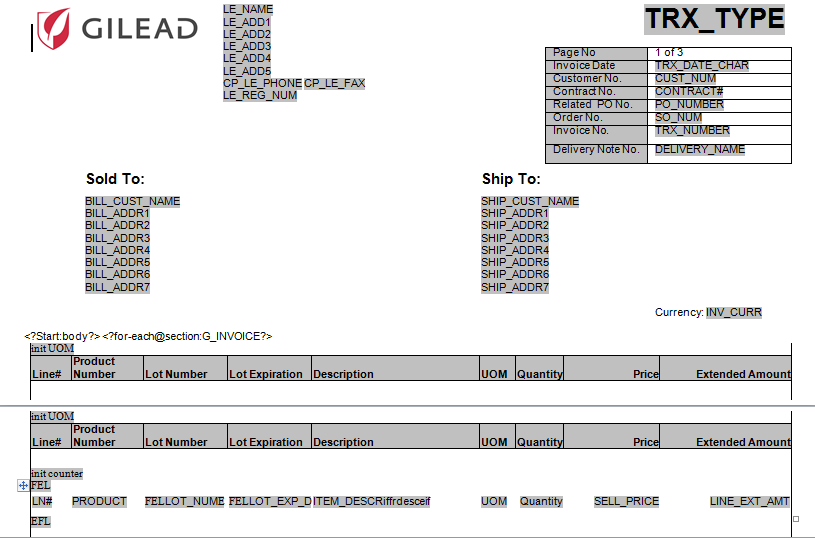
Technical Design:

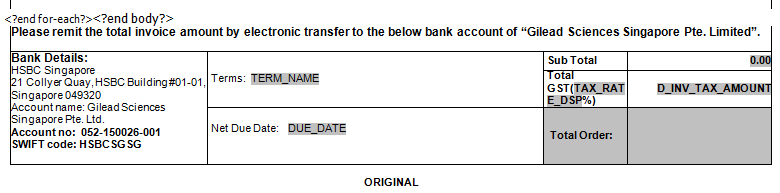
1. Existing concurrent program – ‘GIL Print Commercial Invoices-Standard(Print)’ will be able to access the new template attached
2. Update the existing data definition with the new template TAIWAN.rtf and define it for Taiwan territory
3. **TW\_SG\_SK EBS Implementation Project – Singapore Rollout**

Functional Requirement:

A new invoice template for Singapore has been created as per the requirement for TW SG SK Implementation project. As per the requirement the invoice template has been kept similar to existing Hong Kong invoice template (mentioned in point no. 13 ).

Register the new template for the Singapore’s commercial invoice report “GIL Print Commercial Invoices-Standard (Print)” and file name “SINGAPORE.rtf”. The template will be registered under territory Singapore and language English





New invoice template file is added for Singapore. This template will be available for selection when users submit concurrent program “GIL Print Commercial Invoices-Standard (Print)”.

****

Template details for Singapore:

|  |  |
| --- | --- |
| Template Name | Singapore Invoice |
| Code | XXGILARCOMINVS\_SG |
| Data Definition | Gilead Commercial Invoices- DD |
| Application | Gilead Custom Application |
| Type | RTF |
| Default File | SINGAPORE.rtf |
| Default File Language | English |
| Default Territory | Singapore |
| Default Output Type | PDF |
| File Name | SINGAPORE.rtf |
| Language | English |
| Territory | Singapore |

Technical Design:

1. Existing concurrent program – ‘GIL Print Commercial Invoices-Standard(Print)’ will be able to access the new template attached
2. Update the existing data definition with the new template SINGAPORE.rtf and define it for Singapore territory
3. **CR-1832 Print end customer information on AR Credit Memos**

**Business Need**

Three new fields added for ‘**GIL Print Commercial Invoices-Standard (Print)**’ to display the end customer details - Indirect customer name, indirect customer DEA and Indirect customer address.

The end customer details are currently being captured in RMA received from GENCO. This change is applicable only for US-OU-01 AND US-OU-03.

**Technical Design**

Existing XXGILARCOMINV.rdf has been modified to add 3 new fields as mentioned below

REASON\_CODE field has been added at line level to display the return reason code for RMA orders

|  |  |
| --- | --- |
| **Fields** | **Mapping** |
| INDIRECTCUSTOMER DEA | RA\_CUSTOMER\_TRX.ATTRIBUTE2 |
| INDIRECT CUSTOMER NAME | RA\_CUSTOMER\_TRX.ATTRIBUTE8 |
| INDIRECT CUSTOMER ADDRESS | RA\_CUSTOMER\_TRX.ATTRIBUTE3 |
| REASON\_CODE | OEORDER\_LINES\_ALL.RETURN\_REASON\_CODE |

* Reason code, Indirect customer DEA, Indirect customer name and Indirect customer address fields gets displayed only for Order Type - EDI RMA CR/NCR US-OU-0 and EDI RMA CR/NCR US-OU-03 for US-OU-01 and US-OU-03 operating unit. These Three new column values get displayed in report output only when transaction type is ‘Credit Memo’.
* Existing lookup ‘XXGIL\_EDI\_180\_TRANS\_MAP’ will have the order type as lookup values and operating unit set as DFF attribute2. ’XXGIL\_EDI\_180\_TRANS\_MAP’ containing EDI order type and OU information will determine which transactions would contain end customer information. (Lookup Reference: GENCO RMA design documents: - IOTC151 (Inbound Interface to create RMA via EDI 810 from GENCO), IOTC 152 (Outbound Interface to send credit memos))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lookup Type** | **Lookup Code** | **Description** | **DFF (Operating Unit)** | **DFF (Transaction Identifier)** |
| XXGIL\_EDI\_180\_TRANS\_MAP | 97H | EDI RMA CR/NCR US-OU-01 | US-OU-01 | ORDER\_TYPE |
| XXGIL\_EDI\_180\_TRANS\_MAP | 99H | EDI RMA CR/NCR US-OU-03 | US-OU-03 | ORDER\_TYPE |

* Existing formula column “CF\_ORDER\_COMMENTS” will be used to determine the end customer details - Indirect customer name, indirect customer DEA and Indirect customer address.
* Existing formula column “CF\_LOT\_NUMBER” will be used to determine the reason code and line transaction type of order

**SQL Statements**

**Formula Column - CF\_ORDER\_COMMENTS (report)**

* Get the values of comments, attribute2, attribute3, attribute8 from ra\_customer\_trx\_all table for the current transaction and assign them to placeholder columns - CP\_INDIRECT\_CUSTOMER\_DEA, CP\_INDIRECT\_CUSTOMER\_ADDRESS and CP\_INDIRECT\_CUSTOMER\_NAME respectively.
* Get the transaction type of order header and store in place holder column - CP\_ORDER\_TYPE.

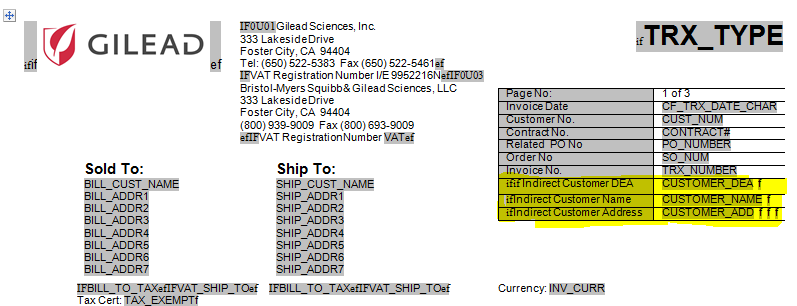
**Formula Column – CF\_LOT\_NUMBER (report)**

* Get the value of return reason at order line level and store in place holder column - CP\_REASON\_CODE.
* Get the transaction type of order line and store in place holder column - CP\_LINE\_ORDER\_TYPE

Three new column values get displayed in report output only transaction type is ‘Credit Memo’.

**RTF Layout**

Modified USA.rtf to display indirect customer DEA, Indirect customer name and indirect customer address details





1. **CR#1814 – EBS Changes for HCV Orders in Turkey**

* **Business requirement:**

1. Currently for Turkey customer invoice report, discounts are displayed as line item. Need to suppress the discount lines in the invoices and display the net amount only.
2. In existing report for Free Of Charge (FOC) invoices, the amounts are displayed as ,00. Need to modify the report so that FOC invoice amounts are displayed as 0,00
3. In existing report for Free of Charge (FOC) invoices, the amount in words for 0 value invoices was displayed as blank. Need to modify the report so that the amount in words is displayed as ‘Sifir’ for FOC invoices.

* **Technical Design:**

1. Modification has been done in XXGILARCOMINVTR .rdf. data model query Q\_LINES has been modified so that it picks up only those line items with line context as order entry and transaction type as Invoice or Credit Memo and line type code not being ‘Discount’.
2. A new formula column - CF\_DISC\_UNIT\_PRICE has been added in main line query ‘Q\_Lines’ to pick the unit selling price from order line.
3. Changes have been done in formula columns - ‘D\_Line\_Unit\_Selling\_Price’ and ‘D\_Line\_Extended\_Amount’ to format the discounted unit seling price and discount net amount of the sales order line item.
4. New Printer configuration details along with its new ip address at Gilead Turkey office has been registered in EBS application. The same printer name (TKWHS-DOT-MATRIX) has been added in concurrent program definition.

* **SQL Query:**
* **Q\_Lines** data model queryhas been modified to pick only the invoice / credit memo line having net unit price of the line item**.** Any discount line of the item should not get picked up for display in report.
* **CF\_DISC\_UNIT\_PRICE** formula column has been created to pick unit selling price of line item from sales order line. This value will always have the net unit price shown even when any discount is applied on the item.
* **D\_Line\_Unit\_Selling\_Price** formula column has been modified to format the unit selling price obtained from formula columnCF\_DISC\_UNIT\_PRICE.
* **D\_Line\_Extended\_Amount** formula column has been modified to format the net line amount obtained by multiplying quantity with unit selling price, i.e **–** price from formula column CF\_DISC\_UNIT\_PRICE.

1. **France e-Invoicing Project**

* **Business Need**

1. Enhance ‘GIL Print Commercial Invoices-Split(Print)’ to stop printing transactions belonging to customers enabled for electronic invoicing.

* **Technical Design**

1. Concurrent program ‘GIL Print Commercial Invoices-Split(Print)’ is modified to have an additional parameter: 'Print eInvoices' with the default value as ‘No’
2. XXGILARCOMINVSPLIT.rdf is modified to add new parameter ‘P\_PRINT\_EINVOICES’
3. Modified the BUILD\_WHERE\_CLAUSE program unit in the XXGILARCOMINVSPLIT.rdf to call the procedure 'build\_where\_clause\_standard' instead of 'build\_where\_clause'
4. Logic for ‘build\_where\_clause\_standard’ procedure
   1. For an eInvoice customer, Attribute1 on the 'Customer Credit Profile Information' DFF will be ‘Invoice Only’ or ‘All AR transactions’ and Invoice creation date has to be greater than or equal to the value in Attribute2 column (AR EDI Effective Date) on the 'Customer Credit Profile Information' DFF.
   2. If p\_print\_einvoices = ‘Yes’ then the report will print all transactions as per the existing functionality
   3. If p\_print\_einvoices = ‘No’ then the report will not print relevant transactions for eInvoice enabled customers
5. **CR#1688 – Automation – Email AR Transactions for non-EDI customer**

* **Business requirement:**

1. New parameter “Email Address for Copy” needs to be added for both GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email). This is an optional parameter and will have only Gilead email id and will be utilized as CC email address while sending email.
2. New parameter “Print/Email” needs to be added for both GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email). This parameter will be non-displayed and will have only one value ‘Email’.
3. The Invoice print program and split program needs to have the following logic to send email

* If the customer is set up as EDI, this will overwrite all other information and the transaction should not be selected for email (unless Print eInvoices Parameter is set as Yes)
* If the customer has an email address available at bill-to site/ customer level then send invoices/CM via email. If no email address found either at bill-to site/ customer level then sends email to the common email id address set at operating unit level.
* A new DFF ‘AR Transaction Email Option‘ will be set up at customer profile level to categorize the transactions which will be sent via email. Following logic will be utilized to determine the transactions that will be sent via email.
  + If customer profile level DFF value is ‘Email Invoices Only’ then only invoices will be sent via email. The credit memos will be printed as usual.
  + If customer profile level DFF value is ‘Email Credit Memos Only’ then only credit memos will be sent via email. The invoices will be printed as usual.
  + If customer profile level DFF value is ‘Email All Transactions ‘then invoices and credit memos will be sent via email.

1. Derivation logic to determine email address to which the invoice/CM will be emailed to is as follows –

* For Invoices :The email address of the contact defined at the customer bill to site level with Role as “Invoices” will be checked first. If it exists then the transactions will be mailed to that email address.
* For Credit Memos :The email address of the contact defined at the customer bill to site level with Role as “Credit memos” will be checked first. If it exists then the transactions will be mailed to that email address.
* For Invoices or Credit Memos: If the above mentioned email address is not present then check for the email address under “statements” at the customer level.

If there is no email address then use the common email address for the Operating unit.

1. Logic for sending email invoice/CM –

* Emails to customers will be grouped by Bill-To site.
* If there is more than one Invoice/CM for a bill to site, then there will be multiple attachments in email for each invoice/CM
* There is no requirement to limit on the no. of attachments in an email. However, for flexibility, site level profile value of “XXGIL\_AR\_EMAIL\_ATTACHMENT\_LIMIT” will be used to control maximum no. of attachment in an email. The value will be set as 50 during go-live.

1. Terms and conditions file will be attached along with the invoices for a given bill-to site, only for those operating units for which it is defined and there will be only one copy attached per email sent.
2. Attachment file name
   1. If PO number is available : PO#XXXXXX - <Transaction Class>#XXXXXX
   2. If PO number is not available : <Transaction Class>#XXXXXX

Transaction class could be INV, CM, etc

1. Once the Email is sent
   1. Print related columns – Print Pending flag, Print dates are updated. Print date at the transaction level indicates the date on which the Email was sent.
   2. Update AR Transaction Descriptive Flexfield (attribute10) “Transaction sent Via email” to Yes
2. GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email) programs when run in “NEW”/”SEL” mode, then email will be sent only when there is an email id available either at bill-to site/ customer level/ operating unit level.
3. GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email) programs when run in ‘SEL’ mode will have the following additional validations and the program should end in error if the validation fails
   1. To limit significant number of emails to our customers, usage of at least one of the following parameters should be mandatory : Transaction number Low/High, Transaction date Low/High, Sales Order Low/High, Delivery Number Low/High, or Customer PO Num Low/High
   2. A threshold of 100 emails is setup to avoid accidental emails to customers. The parameters selected in this run generate xxx emails that exceed the threshold. Please narrow the number of emails by using additional parameters (The Threshold value of 100 will be maintained in a Site level Profile “GIL AR Transaction Limit”)
4. GIL Print Commercial Invoices-Standard (Print) and GIL Print Commercial Invoices-Split (Print) programs should use the following logic
   1. Print einvoices – “No” : Ignore the EDI and Email transactions
   2. Print einvoices – “Yes” : Print all the transactions (including EDI / Email)
5. GIL email for eInvoices
   1. Email einvoices – “No” : Ignore the EDI transactions
   2. Email einvoices – “Yes” : Email all the transactions (including EDI / Email)
6. From email address recepient need to receive the failure notification if the email bounces back from the customer contact (this is not EBS functionality but standard email functionality).
7. After end of every run, a summary of the transactions need to be emailed along with bill-to site and recipient email address to the common email address of operating unit.

* **Technical Design:**

1. Create custom table xxgil\_ar\_email\_details under XXGIL schema to hold the details of the invoices to be emailed.
2. Create two new custom concurrent programs - GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email). These will be similar to their existing print programs except two additional parameters added to these.
3. Add two new parameters 'Print/Email' and ‘Email Address for Copy' in GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email).

'Print/Email' will be a non-displayed parameter and will have only one value ‘Email’

‘Email Address for Copy' will be a free text field and holding only Gilead email address.

1. Add a new parameter ‘P\_SUBMIT\_STATUS’ to send emails with invoices as attachment only when it is having value ‘Y’
2. Modified the prompt of parameter 'Print Choice' to 'Email Choice' in GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email).
3. Following changes have been done in XXGILARCOMINV.rdf and XXGILARCOMINVSPLIT .rdf for the derivation logic of email address and to email invoice/CM/DM files as attachment.
   1. Add new user parameters ‘P\_PRINT\_EMAIL’, ‘P\_SUBMIT\_STATUS’ and ‘P\_CC\_EMAIL\_ADDRESS ‘in XXGILARCOMINV.rdf and XXGILARCOMINVSPLIT .rdf. ‘P\_PRINT\_EMAIL’ will hold only one value ‘Email’.
   2. Change lexical parameter ‘lp\_where10’ of program unit ‘BUILD\_QUERY’ to have an additional filter condition to pick either invoices or only CM or all, based on the value of DFF (hz\_customer\_profile.attribute8)set at customer profile level.
   3. Add logic in ‘After Parameter Form’ trigger to allow sending emails to customers when submitted with ‘SEL’ mode, when at least one of the following parameters have value –

* Transaction number Low/High
* Transaction date Low/High
* Sales Order Low/High
* Delivery Number Low/High,
* Customer PO Num Low/High

If at least one of the above parameter has value, then set parameter – P\_SUBMIT\_STATUS as ‘Y’, else as ‘N’.

In ‘NEW’ mode, parameter P\_SUBMIT\_STATUS is set as ‘Y’ always.

* 1. Create new formula column ‘CF\_Email\_Info’ under data query block ‘Q\_INVOICE’. This calls custom package function xxgil\_email\_comm\_invoice\_pkg.update\_table to insert invoice details and the email addresses.
  2. Create new formula column ‘CF\_TEMPLATE\_LOC’ under data query block ‘Q\_INVOICE ’to get the location of the rtf template through a call to custom package function xxgil\_email\_comm\_invoice\_pkg.get\_template\_loc
  3. Create new formula column ‘CF\_ATTACHMENT\_TITLE’ under data query block ‘Q\_INVOICE’ to get the title of the invoice files generated through bursting through a call to custom package function xxgil\_email\_comm\_invoice\_pkg. attachment\_title
  4. Create new formula column ‘CF\_ORG\_NAME’ under data query block ‘Q\_INVOICE’ to pull the operating unit name from report level placeholder column ‘RP\_ORG\_NAME’
  5. Create new formula column ‘CF\_LE\_NAME’ under data query block ‘Q\_INVOICE’ to pull the legal entity name from report level placeholder column ‘RP\_LE\_NAME’
  6. Modified ‘AfterReport’ trigger of GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email) programs to call custom email program ‘GIL Email Commercial Invoices-Standard’ only when
     1. When submitted with ‘SEL’ mode, the number of invoices processed by the current request is less than the limit set by the custom profile ‘GIL AR Transaction Email Limit‘
     2. There is at least one invoice/CM/DM to be emailed in current request and P\_SUBMIT\_STATUS as ‘Y’.
  7. Additional logic in custom package procedure XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG.build\_where\_clause\_standard, such that for existing invoice print programs –
     1. Print Choice - “NEW” : Print the transactions for the customers not setup as EDI or Email
     2. Print Choice - “SEL” : Print the transactions even if the email id is available
  8. Create bursting control file ‘GIL\_Print\_Comm\_Invoice.xml’ to burst the invoice print program at invoice level with file system as "FILE\_DELIVERY" and output as specific server path $XXGIL\_TOP/outbound/bursting, for all individual invoice files to be generated through bursting.

1. Update USA.rtf and IRELAND.rtf by replacing the XML source of fields where ‘RP\_ORG\_NAME’ and ‘RP\_LE\_NAME’ being used, by ‘CF\_ORG\_NAME’ and ‘CF\_LE\_NAME’ respectively.
2. Create custom concurrent program ‘GIL Email Commercial Invoices-Standard’ to email invoices as attachment to the email address set at bill-to site/ customer level / operating unit level.

This program calls custom shell script XXGIL\_AR\_EMAIL\_ATTACHMENTS.prog placed under $XXGIL\_TOP/bin directory in server.

* + 1. Call standard bursting program “XML Publisher Report Bursting Program” only after parent program GIL Email Commercial Invoices-Standard (Email) or GIL Email Commercial Invoices-Split (Email) program is complete.
    2. Once bursting program completes, custom shell script picks the directory path where invoice files are generated through bursting program, based on custom profile – ‘XXGIL\_AR\_TRX\_FILE\_LOC’ set at site level.
    3. It loads the invoice files into the custom table – xxgil\_ar\_email\_details as a BLOB data against the request\_id and invoice number its being generated. This is done through SQL Loader command and control file - XXGIL\_AR\_EMAIL\_DETAILS\_LOAD.ctl placed under same $XXGIL\_TOP/bin directory.
    4. Custom package procedure xxgil\_email\_comm\_invoice\_pkg. send\_mail\_with\_invoice being called to email invoices along with T&C files as attachment, if any, available for the operating unit, maintained .in custom value set “XXGIL\_AR\_EMAIL\_TRANSACTIONS”.

Once invoice email is sent, update AR Transaction Descriptive Flexfield (attribute10 of ra\_customer\_trx\_all) to ‘Yes’, print pending flag, print dates against the transaction.

* + 1. A summary email is sent to the email address set at operating unit level using custom package function xxgil\_email\_comm\_invoice\_pkg. send\_summary\_email

1. Create custom value set “XXGIL\_AR\_EMAIL\_TRANSACTIONS” to hold the email information like – From email address, email subject/body and T&C attachment, if any, for required operating unit.

* **Assumptions:**

1. Chargeback transactions are out of scope of this report
2. No changes to the Invoice Layout has been done
3. Customers that are EDI-enabled are out of scope of this report
4. No email will be sent when “Email/Print Choice” is set as ‘Print’.

* **SQL Query**

1. Add additional logic in lexical parameter ‘lp\_where10’ under ‘BUILD\_QUERY’program unit to determine if the transaction type matches with the DFF (hz\_customer\_profile.attribute8)set at customer profile level
2. In ‘AfterReport’ trigger

IF at least one transaction is there to be emailed then

IF p\_print\_email = ‘EMAIL’ and P\_SUBMIT\_STATUS = ‘Y’

IF (P\_CHOICE = 'SEL') and (total number of invoices in current request less than the value of custom profile ‘GIL AR Invoices Email Attachment Limit’ THEN

Call custom concurrent program ‘GIL Email Commercial Invoices-Standard’ to email invoices

END IF

IF (P\_CHOICE = ‘NEW’) THEN

Call custom concurrent program ‘GIL Email Commercial Invoices-Standard’ to email invoices

END IF

END IF

1. Derive email address
   1. Call custom package procedure xxgil\_email\_comm\_invoice\_pkg.customer\_email\_address to derive the email address from bill-to site /customer level/ operating unit level.

If email address is available at bill-to site level based on the transaction then send invoices to that email address

If email address is not available at bill-to site level then check if it’s available at customer level and if so then send invoices to that email address

If email address is not available either at bill-to site/ customer level then pick email address from operating unit level and send invoices to that email address.

1. Insert into custom table xxgil\_ar\_email\_address invoice details
   1. Create formula column ‘CF\_Email\_Info’ under query block ‘Q\_INVOICE’
   2. Call custom package procedure xxgil\_email\_comm\_invoice\_pkg.update\_table within ‘CF\_Email\_Info’
      1. This procedure has parameters – parent concurrent program request id, transaction id, bill-to site use id, customer number, CC email address, transaction type, term count, term sequence number, printing count and printing original date.
      2. Call xxgil\_email\_comm\_invoice\_pkg.customer\_email\_address within xxgil\_email\_comm\_invoice\_pkg.update\_table, based on transaction id to derive email address.
      3. Call xxgil\_email\_comm\_invoice\_pkg.attachement\_title within xxgil\_email\_comm\_invoice\_pkg.update\_table to derive invoice/CM/DM file name based on the transaction number and its related PO
      4. Insert parent concurrent program request id, transaction number, operating unit name and id, transaction class, email address, cc email address, bill-to site, transaction type, term count, term sequence number, printing count, printing original date and attachment file name based on the transaction id, concurrent request id and bill-to site use id.
2. Call custom emailing program ‘GIL Email Commercial Invoices-Standard’ to email invoices as attachment to the email address set at bill-to site/ customer level / operating unit level.
   1. Submit standard bursting program “XML Publisher Report Bursting Program” only after parent program GIL Email Commercial Invoices-Standard (Email) or GIL Email Commercial Invoices-Split (Email) program is complete.
   2. Once bursting program completes, pick invoice files from server generated through bursting program.
   3. Load the invoice files into the custom table – xxgil\_ar\_email\_details as a BLOB data against the request\_id and invoice number it’s being generated using SQL Loader command and control file - XXGIL\_AR\_EMAIL\_DETAILS\_LOAD.ctl.
   4. Call custom package procedure xxgil\_email\_comm\_invoice\_pkg. send\_mail\_with\_invoice to email invoices along with T&C files as attachment.

Once email is sent for invoices, update attribute10 = ‘Yes’ of ra\_customer\_trx\_all against the transaction and send summary email to the email address set at operating unit level.

1. **CR#837 – Outbound EDI 812 CM Interface**

* **Business Requirement:**

1. Report need not to print the EDI 812 Credit memosOU if the following Customer Profile Values are available at Customer level
2. EDI 812 (Yes/No) – Attribute6 in Customer Profile
3. EDI 812 Effective Date - Attribute 7 in Customer profile
4. GIL Email Commercial Invoices-Standard (Email) and GIL Email Commercial Invoices-Split (Email) programs should use the following logic

a. Email edi Invoices – “No” : Ignore the EDI transactions

b. Email edi Invoices – “Yes” : Email the transactions (including EDI if the customer is setup to email Credit Memos)

1. GIL Print Commercial Invoices-Standard (Print) and GIL Print Commercial Invoices-Split (Print) programs should use the following logic
   1. Print einvoices – “No” : Ignore the EDI and Email transactions

Print einvoices – “Yes” : Print all the transactions (including EDI / Email)

* **Assumption:**

. a) The above profile options to be enabled for EDI 812 US Trading Partner

b) Customers that are EDI-enabled are out of scope of this report

* **Technical Design:**
  + Validation logic added in custom package procedure XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG.build\_where\_clause\_standard, so that for existing invoice print programs not to pick credit memos –
    1. Attribute6 in the Customer profile for the Credit memos should be set as “Yes”
    2. Attribute7 in the Customer profile for the Credit memos should be greater than or equal to Transaction date.

1. **CR#1962 – Kite Drop Ship Orders – Austria and Germany template changes.**

* **Business Requirement:**

Gilead Registered address is being printed in the Austria and Germany Invoice for Kite Invoices. The RTF for Germany and Austria need to be changed to include Kite registered address for Kite Invoices.

* **Technical Design:**

A new calculated column is being introduced in the ‘G\_INVOICES’ group. CF\_KITE\_ITEM. This flag will have ‘Y’ for Kite items.

The Program unit for derivation logic is as follows:

function CF\_KITE\_ITEM\_Formula return char is

BEGIN

SELECT INVENTORY\_ITEM\_ID

INTO lv\_inventory\_item\_id

FROM ra\_customer\_trx\_lines\_all

WHERE customer\_Trx\_id = :customer\_trx\_id

AND line\_type = 'LINE';

-- Get the Inventory item and its Category

SELECT distinct mc.segment3

INTO lv\_prod\_cat

FROM mtl\_system\_items\_b msib,

mtl\_item\_categories mic,

mtl\_category\_sets\_tl mcst,

mtl\_category\_sets\_b mcs,

mtl\_categories\_b mc

WHERE 1=1

AND mic.inventory\_item\_id = msib.inventory\_item\_id

AND mic.category\_set\_id = mcs.category\_set\_id

AND mcs.category\_set\_id = mcst.category\_set\_id

AND mcst.LANGUAGE = USERENV ('LANG')

AND mic.category\_id = mc.category\_id

AND mcst.category\_set\_name = 'Inventory'

and msib.inventory\_item\_id = lv\_inventory\_item\_id;

IF lv\_prod\_cat = 'Yescarta' THEN

lv\_kite\_item := 'Y';

END IF;

return lv\_kite\_item;

## Table and View Usage

| **Table Name** | **Owner** | **Select** | **Insert** | **Update** | **Delete** |
| --- | --- | --- | --- | --- | --- |
| XXGIL\_AR\_EMAIL\_DETAILS | XXGIL | X | X | X | X |

## Table Structure

| **Table Name** | **Column Name** | **Data Type** |
| --- | --- | --- |
| XXGIL\_AR\_EMAIL\_DETAILS | REQUEST\_ID | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | TRANSACTION\_NUMBER | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | BILL\_TO\_SITE | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | CUSTOMER\_NUMBER | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | EMAIL\_ADDRESS | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | CC\_EMAIL\_ADDRESS | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | ORG\_ID | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | ORG\_NAME | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | FILE\_NAME | VARCHAR2(4000) |
| XXGIL\_AR\_EMAIL\_DETAILS | TRANSACTION\_CLASS | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | FILE\_DATA | BLOB |
| XXGIL\_AR\_EMAIL\_DETAILS | TERM\_SEQUENCE\_NUMBER | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | TRX\_TYPE | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | TERM\_COUNT | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | PRINTING\_COUNT | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | PRINTING\_ORIGINAL\_DATE | DATE |
| XXGIL\_AR\_EMAIL\_DETAILS | EMAIL\_SENT | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | EMAIL\_ERROR | VARCHAR2(2000) |
| XXGIL\_AR\_EMAIL\_DETAILS | LAST\_UPDATED\_DATE | DATE |
| XXGIL\_AR\_EMAIL\_DETAILS | LAST\_UPDATED\_BY | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | LAST\_UPDATE\_LOGIN | NUMBER |
| XXGIL\_AR\_EMAIL\_DETAILS | CREATION\_DATE | DATE |
| XXGIL\_AR\_EMAIL\_DETAILS | CREATED\_BY | NUMBER |

## Package

| **Package Name** | **Owner** |
| --- | --- |
| XXGIL\_EMAIL\_COMM\_INVOICE\_PKG | APPS |
| XXGIL\_ARP\_TRX\_SELECT\_CONT\_PKG | APPS |

## Host File

| **File Name** | **Directory** |
| --- | --- |
| XXGIL\_AR\_EMAIL\_ATTACHMENTS.prog | $XXGIL\_TOP/bin |

## Control File

| **File Name** | **Directory** |
| --- | --- |
| XXGIL\_AR\_EMAIL\_DETAILS\_LOAD.ctl | $XXGIL\_TOP/bin |

## Profile

| **Profile Name** | **Application** | **Level** | **Value** |
| --- | --- | --- | --- |
| XXGIL\_AR\_EMAIL\_ATTACHMENT\_LIMIT | Gilead Custom Application | Site | 50 |
| XXGIL\_AR\_EMAIL\_PURGE\_DATA | Gilead Custom Application | Site | 30 |
| XXGIL\_AR\_TRX\_FILE\_LOC | Gilead Custom Application | Site | /outbound/bursting/ |
| XXGIL\_AR\_TRX\_LIMIT\_ERROR | Gilead Custom Application | Site | 100 |

## Value Set

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_EMAIL\_REQUEST |
| **Size** | Number(15) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | Table:  (select request\_id from xxgil\_ar\_email\_details) xx  Table Columns:  REQUEST\_ID NUMBER(15)  Where/order by:  order by request\_id desc |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_OPERATING\_UNITS |
| **Size** | Char(50) |
| **Type** | Table |
| **Reqd** |  |
| **SQL** | Table:  fnd\_concurrent\_requests fcr, hr\_operating\_units hou  Table Columns:  HOU.NAME CHAR(50)  Where/order by:  WHERE fcr.org\_id = hou.organization\_id  AND fcr.request\_id = :$FLEX$.XXGIL\_AR\_EMAIL\_REQUEST |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_VALIDATE\_EMAIL |
| **Size** | Char(10) |
| **Type** | Special |
| **Reqd** |  |
| **Validate (SQL)** | FND PLSQL "DECLARE  lv\_str varchar2(100) := :!VALUE;  lv\_com varchar2(100);  lv\_standard varchar2(100) :='@GILEAD.COM';  BEGIN  IF (lv\_str IS NOT NULL) THEN  SELECT UPPER(substr(lv\_str,instr(lv\_str,'@' ,1),length(lv\_str)))  INTO lv\_com  FROM dual;  IF lv\_com<>lv\_standard THEN  fnd\_message.set\_name( 'FND', 'FND\_GENERIC\_MESSAGE') ;  fnd\_message.set\_token( 'MESSAGE', 'Enter only Gilead Email id') ;  fnd\_message.raise\_error;  END IF;  END IF;  END;" |

|  |  |
| --- | --- |
| **Value Set** | XXGIL\_AR\_EMAIL\_TRANSACTIONS |
| **Size** | Char(50) |
| **Type** | Independent |
| **Reqd** |  |
| **SQL** |  |

Messages

| **Name** | **Language** | **Application** | **Message Text** |
| --- | --- | --- | --- |
| XXGIL\_AR\_ALL\_INV\_TITLE\_MSG | US | Gilead Custom Application | &TRX\_TYPE from Gilead Sciences, Inc. Customer number &CUST\_NUM |
| XXGIL\_AR\_BE\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Gilead's AR Standard Terms & Conditions also attached.  Kind Regards,  Customer Service Belgium  Gilead Sciences Belgium SPRL-BVBA |
| XXGIL\_AR\_DE\_MSG | US | Gilead Custom Application | Sehr geehrte Damen und Herren,  Im Anhang erhalten Sie Ihre Rechnung/en zu Ihrem Auftrag/Ihren Aufträgen  Mit freundlichen Grüßen  Gilead Sciences GmbH |
| XXGIL\_AR\_DE\_TITLE\_MSG | US | Gilead Custom Application | Ihre Rechnung von Gilead Sciences GmbH, Ihre Kundennummer: &CUST\_NUM |
| XXGIL\_AR\_DK\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Kind Regards,  Customer Service Nordics  Gilead Denmark |
| XXGIL\_AR\_EMAIL\_PARAM\_ERROR | US | Gilead Custom Application | To limit significant no. of emails to our customers, usage of at least one of the following parameters is mandatory:Transaction Number Low/High, Trx date Low/High, Sales Order Low/High, Delivery Number Low/High, or Customer PO Num Low/High. |
| XXGIL\_AR\_FI\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Kind Regards,  Customer Service Nordics  Gilead Finland |
| XXGIL\_AR\_IE\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Kind Regards,  Customer Service  Gilead Sciences Ireland UC |
| XXGIL\_AR\_NL\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Kind Regards,  Customer Service Netherlands  Gilead Sciences Netherlands |
| XXGIL\_AR\_NO\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Kind Regards,  Customer Service Nordics  Gilead Norway |
| XXGIL\_AR\_SE\_MSG | US | Gilead Custom Application | Hello,  Please see your invoice(s) attached.  Kind Regards,  Customer Service Nordics  Gilead Sweden |
| XXGIL\_AR\_TRX\_LIMIT\_ERROR | US | Gilead Custom Application | A threshold of &PROFILE\_LIMIT emails is setup to avoid accidental emails to our customers. The parameters selected in this run generate &PROG\_COUNT emails that exceed the threshold. Please narrow the number of emails by using additional parameters. |
| XXGIL\_AR\_US01\_MSG | US | Gilead Custom Application | Hello,  Please see your &TRX\_TYPE attached.  Please see link below for Gilead's AR Standard Terms & Conditions:  http://www.gilead.com/~/media/files/pdfs/other/standard%20terms%20and%20conditions\_2015.pdf  Kind Regards,  Accounts Receivable  Gilead Sciences, Inc. |
| XXGIL\_AR\_US03\_MSG | US | Gilead Custom Application | Hello,  Please see your &TRX\_TYPE attached.  Please see link below for BMS-Gilead Sciences, LLC's AR Standard Terms & Conditions:  http://www.gilead.com/~/media/files/pdfs/other/standard%20terms%20and%20conditions\_2015.pdf  Kind Regards,  Accounts Receivable  BMS-Gilead Sciences, LLC |
| XXGIL\_AR\_US03\_TITLE\_MSG | US | Gilead Custom Application | &TRX\_TYPE from BMS-Gilead Sciences, LLC |
| XXGIL\_AR\_US\_TITLE\_MSG | US | Gilead Custom Application | &TRX\_TYPE from Gilead Sciences, Inc. |

1. **CR 1936 Kite Drop Ship Order Setup**

**Business Need**

GIL Print Commercial Invoices-Standard (Print) and ‘GIL Print Commercial Invoices-Split (Print)’ is not printing the Lot Number and Lot Expiry Date for Kite Drop Ship Orders

**Technical Design**

For Drop Ship Orders there will not be any Shipping Details (Delivery Name = 0) as the Order is sourced and shipped externally. Hence the Lot Number information cannot be fetched from Shipping Module.

The Lot Query has been enhanced to derived the Lot details from the Receiving Trancsactions and included in the ‘Q\_LOTS’ Query of the RDF file.

The new query would be

--Created by Suresh Kandru to get lot number and expiration date for line item. Union below queries to get lot number and expiration date for credit memos and invoices.

-- Added by Suresh Kandru on May12 2011, query to get lot number and expiration date for intercompany invoices defect #4871

-- Query modified by Vikas Sathidev on 04 May,2017 as per REQ0227814/PRB12622

/\* Senthil : CR1936 - Kite Drop Shop Process - Missing Lot number and Lot Expiry date \*/

SELECT

wdd.source\_line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

wdd.date\_requested,

wdd.lot\_number,

mln.expiration\_date,

wdd.REQUESTED\_QUANTITY line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814

,-1 transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.wsh\_new\_deliveries wnd ,

apps.wsh\_delivery\_assignments wda,

apps.wsh\_delivery\_details wdd,

apps.mtl\_lot\_numbers mln

WHERE wnd.delivery\_id = wda.delivery\_id

AND wda.delivery\_detail\_id=wdd.delivery\_detail\_id

AND wdd.lot\_number = mln.lot\_number

AND wdd.organization\_id = mln.organization\_id

AND wnd.name=:line\_delivery\_name

AND :CP\_TRX\_TYPE\_CLASS ='INV'

UNION ALL -- Query split to handle INTERCOMPANY invoices separately from other invoices as INTERCOMPANY invoices alone have the transaction\_id

SELECT distinct wdd.source\_line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

wdd.date\_requested,

wdd.lot\_number,

mln.expiration\_date,

wdd.REQUESTED\_QUANTITY line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,

mmt.transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.wsh\_new\_deliveries wnd,

apps.wsh\_delivery\_assignments wda,

apps.wsh\_delivery\_details wdd,

apps.mtl\_lot\_numbers mln,

apps.mtl\_material\_transactions mmt,

apps.mtl\_transaction\_lot\_numbers mtln

WHERE wnd.delivery\_id = wda.delivery\_id

AND wda.delivery\_detail\_id = wdd.delivery\_detail\_id

AND wdd.lot\_number = mln.lot\_number

AND wdd.organization\_id = mln.organization\_id

AND wnd.name = :line\_delivery\_name

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

AND mmt.TRX\_SOURCE\_LINE\_ID = wdd.source\_line\_id

AND mmt.transaction\_id = mtln.transaction\_id

AND mln.lot\_number = mtln.lot\_number

UNION ALL -- Query split to handle INTERCOMPANY invoices which are 'Logical Sales Order Issue/Receipt'

SELECT distinct wdd.source\_line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

wdd.date\_requested,

wdd.lot\_number,

mln.expiration\_date,

wdd.REQUESTED\_QUANTITY line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,

mmt.transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.wsh\_new\_deliveries wnd,

apps.wsh\_delivery\_assignments wda,

apps.wsh\_delivery\_details wdd,

apps.mtl\_lot\_numbers mln,

apps.mtl\_material\_transactions mmt

WHERE wnd.delivery\_id = wda.delivery\_id

AND wda.delivery\_detail\_id = wdd.delivery\_detail\_id

AND wdd.lot\_number = mln.lot\_number

AND wdd.organization\_id = mln.organization\_id

AND wnd.name = :line\_delivery\_name

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

AND mmt.TRX\_SOURCE\_LINE\_ID = wdd.source\_line\_id

and not exists (select 1

from mtl\_transaction\_lot\_numbers mtln

where 1=1

and mmt.transaction\_id = mtln.transaction\_id)

AND mln.lot\_number in (select d.lot\_number

from mtl\_material\_transactions a,

mtl\_material\_transactions b,

wsh\_delivery\_details d

where 1=1

and a.transaction\_id = mmt.transaction\_id

and b.transaction\_id = a.transaction\_batch\_id

and d.delivery\_detail\_id = b.picking\_line\_Id)

UNION

SELECT

oola.line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

oola.request\_date,

oels.lot\_number,

mln.expiration\_date,

nvl(oola.shipping\_quantity,oola.shipped\_quantity) line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,-1 transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.mtl\_lot\_numbers mln,

apps.oe\_lot\_serial\_numbers oels,

apps.oe\_order\_lines\_all oola,

apps.mtl\_parameters mp

WHERE oels.line\_id = oola.line\_id

AND oels.lot\_number = mln.lot\_number

AND mp.organization\_id= mln.organization\_id

AND oola.ship\_from\_org\_id = mp.organization\_id

AND :CP\_TRX\_TYPE\_CLASS ='INV'

UNION ALL-- Query split to handle INTERCOMPANY and other invoices separately

SELECT distinct oola.line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

oola.request\_date,

oels.lot\_number,

mln.expiration\_date,

NVL(oola.shipping\_quantity, oola.shipped\_quantity) line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,

mmt.transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.mtl\_lot\_numbers mln,

apps.oe\_lot\_serial\_numbers oels,

apps.oe\_order\_lines\_all oola,

apps.mtl\_parameters mp,

apps.mtl\_material\_transactions mmt

WHERE oels.line\_id = oola.line\_id

AND oels.lot\_number = mln.lot\_number

AND mp.organization\_id = mln.organization\_id

AND oola.ship\_from\_org\_id = mp.organization\_id

AND mmt.source\_line\_id = oola.line\_id

AND mmt.inventory\_item\_id = mln.inventory\_item\_id

AND mmt.organization\_id = mln.organization\_id

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

UNION

SELECT

oola.line\_id line\_so\_line\_id,

rlt.item\_id inventory\_item\_id,

oola.request\_date,

rlt.lot\_num,

rlt.expiration\_date,

nvl(oola.shipping\_quantity,oola.shipped\_quantity) line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,-1 transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.rcv\_lot\_transactions rlt,

apps.rcv\_transactions rt,

apps.oe\_order\_lines\_all oola,

apps.rcv\_shipment\_lines rsl

WHERE rsl.oe\_order\_line\_id = oola.line\_id

AND rt.shipment\_line\_id= rsl.shipment\_line\_id

AND rt.transaction\_id=rlt.transaction\_id

AND rt.transaction\_type='RECEIVE'

AND :CP\_TRX\_TYPE\_CLASS ='CM'

AND rsl.source\_document\_code='RMA'

UNION ALL -- Query split to handle INTERCOMPANY and other invoice types separately

SELECT distinct oola.line\_id line\_so\_line\_id,

rlt.item\_id inventory\_item\_id,

oola.request\_date,

rlt.lot\_num,

rlt.expiration\_date,

NVL (oola.shipping\_quantity, oola.shipped\_quantity) line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,

mmt.transaction\_id --Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.rcv\_lot\_transactions rlt,

apps.rcv\_transactions rt,

apps.oe\_order\_lines\_all oola,

apps.rcv\_shipment\_lines rsl,

apps.mtl\_material\_transactions mmt

WHERE rsl.oe\_order\_line\_id = oola.line\_id

AND rt.shipment\_line\_id = rsl.shipment\_line\_id

AND rt.transaction\_id = rlt.transaction\_id

AND rt.transaction\_type = 'RECEIVE'

AND :CP\_TRX\_TYPE\_CLASS = 'CM'

AND rsl.source\_document\_code = 'RMA'

AND rt.transaction\_id = mmt.rcv\_transaction\_id

UNION

SELECT

wdd.source\_line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

wdd.date\_requested,

wdd.lot\_number,

mln.expiration\_date,

XXGIL\_OTC\_PRINT\_GLOBAL\_PKG.xxgil\_delsum\_qty(wdd.source\_line\_id,mln.inventory\_item\_id) line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622

,-1 transaction\_id -- Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.wsh\_delivery\_details wdd,

apps.mtl\_lot\_numbers mln

WHERE wdd.lot\_number = mln.lot\_number

AND wdd.organization\_id = mln.organization\_id

AND :CP\_TRX\_TYPE\_CLASS ='INV'

UNION ALL -- Query split to handle INTERCOMPANY and other invoice types separately

SELECT distinct wdd.source\_line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

wdd.date\_requested,

wdd.lot\_number,

mln.expiration\_date,

wdd.REQUESTED\_QUANTITY line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622 ,

,mmt.transaction\_id transaction\_id -- Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.wsh\_delivery\_details wdd,

apps.mtl\_lot\_numbers mln,

apps.mtl\_material\_transactions mmt,

apps.mtl\_transaction\_lot\_numbers mtln

WHERE wdd.lot\_number = mln.lot\_number

AND wdd.organization\_id = mln.organization\_id

AND wdd.inventory\_item\_id = mln.inventory\_item\_id

and wdd.organization\_id = mmt.organization\_id

AND mmt.organization\_id = mln.organization\_id

AND mmt.TRX\_SOURCE\_LINE\_ID = wdd.source\_line\_id

and mmt.transaction\_id = mtln.transaction\_id

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

AND mln.lot\_number = mtln.lot\_number

UNION ALL

SELECT distinct wdd.source\_line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

wdd.date\_requested,

wdd.lot\_number,

mln.expiration\_date,

wdd.REQUESTED\_QUANTITY line\_qty\_ordered -- Added by Vikas Sathidev for REQ0227814/PRB12622 ,

,mmt.transaction\_id transaction\_id -- Added by Vikas Sathidev for REQ0227814/PRB12622

FROM apps.wsh\_delivery\_details wdd,

apps.mtl\_lot\_numbers mln,

apps.mtl\_material\_transactions mmt

WHERE wdd.lot\_number = mln.lot\_number

AND wdd.organization\_id = mln.organization\_id

AND wdd.inventory\_item\_id = mln.inventory\_item\_id

and wdd.organization\_id = mmt.organization\_id

AND mmt.organization\_id = mln.organization\_id

AND mmt.TRX\_SOURCE\_LINE\_ID = wdd.source\_line\_id

and not exists (select 1

from mtl\_transaction\_lot\_numbers mtln

where 1=1

and mmt.transaction\_id = mtln.transaction\_id)

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

AND mln.lot\_number in (select d.lot\_number

from mtl\_material\_transactions a,

mtl\_material\_transactions b,

wsh\_delivery\_details d

where 1=1

and a.transaction\_id = mmt.transaction\_id

and b.transaction\_id = a.transaction\_batch\_id

and d.delivery\_detail\_id = b.picking\_line\_Id)

/\* Senthil : CR1936 - Kite Drop Shop Process - Missing Lot number and Lot Expiry date \*/

UNION ALL

SELECT distinct oola.line\_id line\_so\_line\_id,

mln.inventory\_item\_id inventory\_item\_id,

oola.request\_date date\_requested,

mln.lot\_number,

mln.expiration\_date,

NVL(oola.shipping\_quantity, oola.shipped\_quantity) line\_qty\_ordered,

-1 transaction\_id

FROM apps.mtl\_lot\_numbers mln,

apps.oe\_order\_lines\_all oola,

apps.mtl\_parameters mp,

-- apps.mtl\_material\_transactions mmt,

-- apps.MTL\_TRANSACTION\_LOT\_NUMBERS mtln,

apps.rcv\_transactions rt,

apps.rcv\_lot\_transactions rlt,

apps.OE\_DROP\_SHIP\_SOURCES os,

apps.ra\_customer\_trx\_all rct,

apps.ra\_customer\_trx\_lines\_all rctl

WHERE 1=1

AND mp.organization\_id = mln.organization\_id

AND oola.ship\_from\_org\_id = mp.organization\_id

-- AND mmt.source\_line\_id = oola.line\_id

-- AND mmt.inventory\_item\_id = mln.inventory\_item\_id

-- AND mmt.organization\_id = mln.organization\_id

-- AND mmt.source\_line\_id = rctl.interface\_line\_attribute6

AND oola.line\_id = rctl.interface\_line\_attribute6

AND oola.inventory\_item\_id = mln.inventory\_item\_id

AND mp.organization\_id = mln.organization\_id

AND os.header\_id = oola.header\_id

AND rt.po\_header\_id = os.po\_header\_id

AND rt.transaction\_id = rlt.transaction\_id

AND rlt.lot\_num = mln.lot\_number

-- AND mtln.lot\_number = mln.lot\_number

-- and mtln.transaction\_id = mmt.TRANSACTION\_ID

and rctl.customer\_trx\_id = rct.customer\_trx\_id

AND rct.customer\_trx\_id = :customer\_trx\_id

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

/\*and 0 = :line\_delivery\_name \*/

and not exists (select 1 from wsh\_new\_deliveries where 1=1

and name = :line\_delivery\_name)

The same enhancement has been made to calculated field in the RDF

function CF\_lot\_numberFormula

Cursor c1 is

SELECT wdd.date\_requested,

WDD.LOT\_NUMBER,

mln.EXPIRATION\_DATE

FROM WSH\_NEW\_DELIVERIES wnd ,

WSH\_DELIVERY\_ASSIGNMENTS wda,

WSH\_DELIVERY\_DETAILS wdd,

MTL\_LOT\_NUMBERS mln

WHERE WND.NAME = :line\_delivery\_name

AND wnd.delivery\_id = wda.delivery\_id

AND WDA.DELIVERY\_DETAIL\_ID = WDD.DELIVERY\_DETAIL\_ID

AND WDD.SOURCE\_LINE\_ID = :line\_so\_line\_id

AND MLN.INVENTORY\_ITEM\_ID = :INVENTORY\_ITEM\_ID

AND Wdd.lot\_number = mln.lot\_number

AND WDD.ORGANIZATION\_ID = MLN.ORGANIZATION\_ID

UNION

--added by Pratap on 05/20/2010 for CM Lot numbers

SELECT oola.request\_date,

oels.lot\_number,

mln.expiration\_date

FROM apps.mtl\_lot\_numbers mln,

apps.oe\_lot\_serial\_numbers oels,

apps.oe\_order\_lines\_all oola,

apps.mtl\_parameters mp

WHERE oels.line\_id = oola.line\_id

AND oola.line\_id = :line\_so\_line\_id

AND mln.inventory\_item\_id = :inventory\_item\_id

AND oels.lot\_number = mln.lot\_number

AND mp.organization\_id = mln.organization\_id

AND oola.ship\_from\_org\_id = mp.organization\_id

UNION

/\* Senthil : CR 1936 : Kite Drop Ship Process Changes \*/

SELECT distinct oola.request\_date,

rlt.lot\_num,

mln.expiration\_date

FROM apps.mtl\_lot\_numbers mln,

apps.oe\_order\_lines\_all oola,

apps.mtl\_parameters mp,

-- apps.mtl\_material\_transactions mmt,

-- apps.MTL\_TRANSACTION\_LOT\_NUMBERS mtln,

apps.rcv\_transactions rt,

apps.rcv\_lot\_transactions rlt,

apps.OE\_DROP\_SHIP\_SOURCES os,

apps.ra\_customer\_trx\_all rct,

apps.ra\_customer\_trx\_lines\_all rctl

WHERE 1=1

AND mp.organization\_id = mln.organization\_id

AND oola.ship\_from\_org\_id = mp.organization\_id

AND oola.line\_id = rctl.interface\_line\_attribute6

AND oola.inventory\_item\_id = mln.inventory\_item\_id

AND mp.organization\_id = mln.organization\_id

AND os.header\_id = oola.header\_id

AND rt.po\_header\_id = os.po\_header\_id

AND rt.transaction\_id = rlt.transaction\_id

AND rlt.lot\_num = mln.lot\_number

and rctl.customer\_trx\_id = rct.customer\_trx\_id

AND rct.customer\_trx\_id = :customer\_trx\_id

AND :CP\_TRX\_TYPE\_CLASS = 'INV'

and not exists (select 1 from wsh\_new\_deliveries where 1=1

and name = :line\_delivery\_name)

ORDER BY 2,3;

### Concurrent Program Executable

### 1) GIL Email Commercial Invoices-Standard:

|  |  |
| --- | --- |
| **Executable** | XXGILARSENDMAILTRX |
| **Short Name** | XXGILARSENDMAILTRX |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Email AR Transactions |
| **Execution Method** | Host |
| **Execution File Name** | XXGIL\_AR\_EMAIL\_ATTACHMENTS |

### 2) GIL Email Commercial Invoices-Standard (Email)

|  |  |
| --- | --- |
| **Executable** | XXGILARCOMINV |
| **Short Name** | XXGILARCOMINV |
| **Application** | Gilead Custom Application |
| **Description** | GIL Print Commercial Invoices |
| **Execution Method** | Oracle Reports |
| **Execution File Name** | XXGILARCOMINV |

### 3) GIL Email Commercial Invoices-Split (Email):

|  |  |
| --- | --- |
| **Executable** | XXGILARCOMINVSPLIT |
| **Short Name** | XXGILARCOMINVSPLIT |
| **Application** | Gilead Custom Application |
| **Description** | Gilead Print Invoices For Split Quantity By Lot |
| **Execution Method** | Oracle Reports |
| **Execution File Name** | XXGILARCOMINVSPLIT |

### Concurrent Program Definition

1) **GIL Email Commercial Invoices-Standard**

|  |  |
| --- | --- |
| **Program** | GIL Email Commercial Invoices-Standard |
| **Short Name** | XXGILARSENDCOMMINV |
| **Application** | Gilead Custom Application |
| **Description** | GIL Email Commercial Invoices-Standard |
| **Executable Name** | XXGILARSENDMAILTRX |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | Text |
| **Mls Function** |  |
| **Style** |  |

**2) GIL Email Commercial Invoices-Standard (Email)**

|  |  |
| --- | --- |
| **Program** | GIL Email Commercial Invoices-Standard (Email) |
| **Short Name** | XXGILARCOMINVMAIL |
| **Application** | Gilead Custom Application |
| **Description** | GIL Email Commercial Invoices |
| **Executable Name** | XXGILARCOMINV |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | XML |
| **Mls Function** |  |
| **Style** |  |

**3) GIL Email Commercial Invoices-Split (Email)**

|  |  |
| --- | --- |
| **Program** | GIL Email Commercial Invoices-Split (Email) |
| **Short Name** | XXGILARCOMINVLOTSPLIT |
| **Application** | Gilead Custom Application |
| **Description** | GIL Email Commercial Invoices Split Quantity By Lot |
| **Executable Name** | XXGILARCOMINVSPLIT |
| **Request, ‘Use In SRS’ Checkbox** | Y |
| **Output Format** | XML |
| **Mls Function** |  |
| **Style** |  |

### Concurrent Program Parameters

1. **GIL Email Commercial Invoices-Standard:**

|  |  |
| --- | --- |
| **Seq** | 10 |
| **Parameter Name** | Parent Request Id |
| **Parameter** | XXGIL\_AR\_EMAIL\_REQUEST |
| **Value Set/Item Class** | 10/number |
| **Default Value** |  |
| **Required (Y/N)** | Y |
| **Display (Y/N)** | Y |
| **Prompt** | Parent Request Id |

|  |  |
| --- | --- |
| **Seq** | 20 |
| **Parameter Name** | Operating Unit |
| **Parameter** | Operating Unit |
| **Value Set/Item Class** | XXGIL\_AR\_OPERATING\_UNITS |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Operating Unit |

1. **GIL Email Commercial Invoices-Standard (Email): (Only changed ones)**

|  |  |
| --- | --- |
| **Seq** | 8 |
| **Parameter Name** | Print/Email |
| **Parameter** | P\_PRINT\_EMAIL |
| **Value Set/Item Class** | 10 Characters |
| **Default Value** | Email |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Print/Email |

|  |  |
| --- | --- |
| **Seq** | 90 |
| **Parameter Name** | Email Address for Copy |
| **Parameter** | P\_CC\_EMAIL\_ADDRESS |
| **Value Set/Item Class** | XXGIL\_VALIDATE\_EMAIL |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Email Address for Copy |

1. **GIL Email Commercial Invoices-Split (Email): (Only changed ones)**

|  |  |
| --- | --- |
| **Seq** | 8 |
| **Parameter Name** | Print/Email |
| **Parameter** | P\_PRINT\_EMAIL |
| **Value Set/Item Class** | 10 Characters |
| **Default Value** | Email |
| **Required (Y/N)** | N |
| **Display (Y/N)** | N |
| **Prompt** | Print/Email |

|  |  |
| --- | --- |
| **Seq** | 90 |
| **Parameter Name** | Email Address for Copy |
| **Parameter** | P\_CC\_EMAIL\_ADDRESS |
| **Value Set/Item Class** | XXGIL\_VALIDATE\_EMAIL |
| **Default Value** |  |
| **Required (Y/N)** | N |
| **Display (Y/N)** | Y |
| **Prompt** | Email Address for Copy |

XML Publisher Data Definition

1) GIL Email Commercial Invoices-Standard (Email)

|  |  |
| --- | --- |
| **Name** | Gilead Email Commercial Invoices- DD |
| **Code** | XXGILARCOMINVMAIL |
| **Application** | Gilead Custom Application |
| **Start Date** | SYSDATE |
| **End Date** |  |
| **XML Schema** |  |
| **Data Template** |  |
| **Preview Data** |  |
| **Bursting Control File** | GIL\_AR\_Email\_Commercial\_Invoices |

2) GIL Email Commercial Invoices-Split (Email)

|  |  |
| --- | --- |
| **Name** | Gilead Commercial Invoice-Split (Email) |
| **Code** | XXGILARCOMINVLOTSPLIT |
| **Application** | Gilead Custom Application |
| **Start Date** | SYSDATE |
| **End Date** |  |
| **XML Schema** |  |
| **Data Template** |  |
| **Preview Data** |  |
| **Bursting Control File** | GIL Email Commercial Invoices |

XML Publisher Template

1) GIL Email Commercial Invoices-Standard (Email)

| **Name** | **Application** | **Data Definition** | **Type** | **Output Type** | **Default File** | **Language** | **Territory** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Belgium Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | BELGIUM.rtf | English | Belgium |
| Denmark Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | DENMARK,rtf | Danish | Denmark |
| Finland Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | FINLAND.rtf | Finnish | Finland |
| Ireland Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | IRELAND.rtf | English | Ireland |
| Netherlands Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | NETHERLANDS.rtf | Dutch | Netherlands |
| US Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | USA.rtf | English | United States |
| Norway Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | NORWAY.rtf | Norwegian | Norway |
| Sweden Email Invoice | Gilead Custom Application | Gilead Email Commercial Invoices- DD | RTF | PDF | SWEDEN.rtf | Swedish | Sweden |

1. GIL Email Commercial Invoices-Split (Email)

| **Name** | **Application** | **Data Definition** | **Type** | **Output Type** | **Default File** | **Language** | **Territory** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Germany Invoice Template(Email) | Gilead Custom Application | Gilead Commercial Invoice-Split (Email) | RTF | PDF | GERMANY.rtf | English | Germany |

# BI Publisher Approach

The BI publisher templates will be created to display report output. The approach is:

* Using BI publisher table wizard all the report level columns in free form group for displaying headers arrange them as per the report requirement.
* Select columns required from G\_MAIN group and G\_LAST\_RECEIPT group. G\_LAST\_GROUP will have LAST\_RECEIPT\_DT and C\_LAST\_REC\_AMT\_DISP. Rest of the columns from G\_MAIN group. This is repeating frame for all the records.
* For each change of Project display project level totals. For each change of input parameter there will be line break and totals.
* There will be two templates will developed one for EXCEL and another for PDF output. PDF will have all design requirements like page numbering, sort by totals, logo and appearance standards. Excel is direct output without much formatting and easy accessible to excel.

# Unit /Link Test Scenario

| **Sr. No.** | **Name of the Test Case** | **Test Conditions** | **Expected Results** |
| --- | --- | --- | --- |
|  | No input data | Run report but do not select any input data | Error message to indicate at least one input data value must be selected |
|  | Select invalid data | Select invalid data on report (such as PO that does not exist), run report | Error message to indicate that PO number does not exist |
|  | Project has a change in Project Manager | Run report for initial manager, then make change to new project manager, re-run report | The same report should be produced (all history should move to new manager) |
|  | Invalid combination of input data | User should select an invalid combination of input data items | Error message to indicate combination of input values is invalid |
|  | Run report prior to the receiving against a PO, but a PO has been opened for $100 | Run report prior to the receiving of a PO amount, but there is a PO opened for a certain value | Report should contain data for the PO for the full value |
|  | Receive $50 against PO and run report | Receive against a PO valued at $100, receive $50, then immediately run Open PO Report | Report should display the original amount of $100, the $50 receipt amount and date of the receipt and the $50 open amount |
|  | Run: GIL Print Commercial Invoices-Standard(Print) for Hong Kong | Field: Order By, Action: Enter ‘Transasction Number’  Field: Print Choice, Action: Enter ‘SEL’  Field: Trx Date Low, Action: Enter ’01-Nov-2016’  Field: Trx Date High, Action: Enter ’10-Nov-2016’  Field: Open Invoices Only, Action: Enter ‘No’  Field: Diagnostics, Action: Enter ‘N’  Field: Print eInvoices, Action: Enter ‘No’  Field: Print Template, Action: Select the print template for Hong Kong | Gilead specific invoice is printed with modified Hong Kong layout |
|  | Create an invoice manually in Taiwan operating unit  TW-OU-01 AR INVOICES >> Transactions | Enter the required field for invoice creation and save. Note the invoice number | Invoice should be created. |
|  | Run: GIL Print Commercial Invoices-Standard(Print) for Taiwan | Field: Order By, Action: Enter ‘Transasction Number’  Field: Print Choice, Action: Enter ‘SEL’  Field: Trx Date Low, Action: Enter ’01-Jan-2017’  Field: Trx Date High, Action: Enter ’10-Jan-2017’  Field: Open Invoices Only, Action: Enter ‘No’  Field: Diagnostics, Action: Enter ‘N’  Field: Print eInvoices, Action: Enter ‘No’  Field: Print Template, Action: Select the print template for Taiwan | Gilead specific invoice is printed with Taiwan layout. Also the invoice created in the previous step should appear |
|  | Create an invoice manually in Singapore operating unit  SG-OU-01 AR INVOICES >> Transactions | Enter the required field for invoice creation and save. Note the invoice number | Invoice should be created. |
|  | Run: GIL Print Commercial Invoices-Standard(Print) for Singapore | Field: Order By, Action: Enter ‘Transasction Number’  Field: Print Choice, Action: Enter ‘SEL’  Field: Trx Date Low, Action: Enter ’01-Jan-2017’  Field: Trx Date High, Action: Enter ’10-Jan-2017’  Field: Open Invoices Only, Action: Enter ‘No’  Field: Diagnostics, Action: Enter ‘N’  Field: Print eInvoices, Action: Enter ‘No’  Field: Print Template, Action: Select the print template for Singapore | Gilead specific invoice is printed with Singapore layout. Also the invoice created in the previous step should appear |
|  | Run the report “GIL Print Commercial Invoices-Standard (Print)” for non Genco RMA transactions US OU1 and US OU 3 | Verify that Indirect Customer Details are not printed on the Credit Memo | Indirect Customer Attributes are NOT printed on Credit Memo successfully. |
|  | Run the report “GIL Print Commercial Invoices-Standard (Print)” for Credit Memos submitted manually | Verify that Indirect Customer Details are not printed on the Credit Memo | Indirect Customer Attributes are NOT printed on Credit Memo successfully. |
|  | Run the report “GIL Print Commercial Invoices-Standard (Print)” for Standard Invoice | Verify that Indirect Customer Details are not printed on the Invoice | Verify that Indirect Customer Details are not printed on the Invoice |
|  | Run the Report to test Maximum characters getting printed on the Credit Memo | Verify that Indirect Customer Details are printed correctly  Indirect Customer Address : 94 characters  Indirect Customer Name : 40 characters  Indirect Customer DEA : 9 characters | Indirect Customer Attributes are printed on Credit Memo successfully |
|  | Discount line of sales order | Run report for sales order having line item with a discount on it. | Only the line for item should be displayed in output. Discount line with its amount should not be displayed. |
|  | Discounted unit price and net price of sales order | Run report for sales order having line item with a discount on it. | Only line for item should be displayed in output with its discounted unit price and net amount |
|  | Discount unit price and net price of RMA | Run report for a RMA order with line item having a discount on it. | Only the line for item should be displayed in output with its discounted unit price and net amount. Discount line with its amount should not be displayed. |
|  | Output gets printed from dot matrix printer located at Gilead Turkey office | Run concurrent program for any sales order under Turkey OU. | Output should get printed from dot matrix printer |
|  | Execute the concurrent program ‘GIL Print Commercial Invoices-Split(Print)’ from FR-OU-01 AR Invoices responsibility for an EDI Invoice with following parameters:  Order By: Transaction Number  Print Choice: SEL  Transaction Number Low: 3310159910  Transaction Number High: 3310159910  Open Invoices Only: No  Diagnostics: N  Print eInvoices: Yes | Run the concurrent program | The report should be generated and bsting program should be fired. The invoice should appear in the report output |
|  | Execute the concurrent program ‘GIL Print Commercial Invoices-Split(Print)’ from FR-OU-01 AR Invoices responsibility for an EDI Invoice with following parameters:  Order By: Transaction Number  Print Choice: SEL  Transaction Number Low: 3310159910  Transaction Number High: 3310159910  Open Invoices Only: No  Diagnostics: N  Print eInvoices: No | Run the concurrent program | The report should be completed successfully. The invoice should not appear in the report output |
|  | Execute the concurrent program ‘GIL Print Commercial Invoices-Split(Print)’ from FR-OU-01 AR Invoices responsibility for a non EDI Invoice with following parameters:  Order By: Transaction Number  Print Choice: SEL  Transaction Number Low: 3310159909  Transaction Number High: 3310159909  Open Invoices Only: No  Diagnostics: N  Print eInvoices: Yes | Run the concurrent program | The report should be generated and bsting program should be fired. The invoice should appear in the report output |
|  | Execute the concurrent program ‘GIL Print Commercial Invoices-Split(Print)’ from FR-OU-01 AR Invoices responsibility for a non EDI Invoice with following parameters:  Order By: Transaction Number  Print Choice: SEL  Transaction Number Low: 3310159909  Transaction Number High: 3310159909  Open Invoices Only: No  Diagnostics: N  Print eInvoices: No | Run the concurrent program | The report should be generated and bsting program should be fired. The invoice should appear in the report output |
|  | Setup the customer for “Email All AR Transactions”, setup a bill to site contact with Role Invoice and Credit Memo. Run “GIL Email Commercial Invoices-Standard (Email)” | Verify that system run the report successfully | Verify that the Invoices and Credit Memos are emailed to the bill to site contact |
|  | Setup the customer for “Email All AR Transactions”, setup only customer statement contact. Run “GIL Email Commercial Invoices-Standard (Email)” | Verify that system run the report successfully | Verify that the Invoices and Credit Memos are emailed to the customer statement contact |
|  | Setup the customer for “Email All AR Transactions” and donot setup any email contact. Run “GIL Email Commercial Invoices-Standard (Email)” | Verify that system run the report successfully | Verify that the Invoices and Credit Memos are emailed to the from email address for the OU |
|  | Setup the customer for “Email Invoices”. Run “GIL Email Commercial Invoices-Standard (Email)” for a Credit Memo | Verify that system run the report successfully | Credit Memo should not be picked up by the program |
|  | Setup the customer for “Email Credit Memos”. Run “GIL Email Commercial Invoices-Standard (Email)” for a Credit Memo | Verify that system run the report successfully | Verify that the Credit Memos are emailed |
|  | Setup the customer for “Email Invoices”. Run “GIL Email Commercial Invoices-Standard (Email)” for an Invoice | Verify that system run the report successfully | Verify that the Invoice is emailed |
|  | Setup the customer for “Email Credit Memos”. Run “GIL Email Commercial Invoices-Standard (Email)” for an Invoice | Verify that system run the report successfully | Invoice should not be picked up by the program |
|  | Run “GIL Email Commercial Invoices-Standard (Email)” for an EDI Invoice and “Email einvoices” is “No” | Verify that system run the report successfully | Transaction should not be picked up by the program |
|  | Run “GIL Email Commercial Invoices-Standard (Email)” in Email mode for an EDI Invoice and “Email einvoices” is “Yes” | Verify that system run the report successfully | Transaction should be picked up by the program and emailed to the from email address for the Operating Unit |
|  | Run “GIL Email Commercial Invoices-Standard (Email)” | Verify that system run the report successfully | Summary of the transactions emailed should be emailed to the common email address for the Operating Unit |
|  | Setup the customer for “Email All AR Transactions” and Run “GIL Print Commercial Invoices-Standard (Print)” and Print eInvoices as No | Verify that system run the report successfully | Transaction should not be picked up by the program |
|  | Setup the customer for “Email All AR Transactions” and Run “GIL Print Commercial Invoices-Standard (Print)” and Print eInvoices as Yes | Verify that system run the report successfully | Transaction should be picked up by the program |
|  | Setup the customer for “Email All AR Transactions” and Run “GIL Email Commercial Invoices-Standard (Email)” in SEL mode and no other parameters | Report should end in error | Error message should be displayed on click of Diagnostic or View details : To limit significant no. of emails to our customers, usage of at least one of the following parameters is mandatory : Transaction number Low/High, Transaction date Low/High, Sales Order Low/High, Delivery Number Low/High, or Customer PO Num Low/High |
|  | Setup the customer for “Email All AR Transactions” and Run “GIL Email Commercial Invoices-Standard (Email)” in SEL mode and select a parameter range that will send more than 100 Invoices | Report should end in error | Error message should be displayed on click of Diagnostic or View details: A threshold of 100 emails is setup to avoid accidental emails to our customers. The parameters selected in this run generate XX emails that exceed the threshold. Please narrow the number of emails by using additional parameters |
|  | Setup the Attribute6 and Attribute7 for Credit memo customers and Run “GIL Print Commercial Invoices-Standard (Print)” or “GIL Email Commercial Invoices-Standard (Email)” and Print eInvoices as No | Verify that system run the report successfully | Transaction should not be picked up by the program |
|  | Setup the Attribute6 and Attribute7 for Credit memo customers as null and Run “GIL Email/Print Commercial Invoices-Standard (Email/Print)” or “GIL Email Commercial Invoices-Standard (Email)” and Print eInvoices as Yes | Verify that system run the report successfully | Transaction should be picked up by the program |
|  | Execute the concurrent program ‘GIL Print Commercial Invoices-Standard(Print)’ from UK-OU-03 AR Invoices responsibility for a Kite Drop Ship Order | Run the concurrent program | The report should be generated and bsting program should be fired. The invoice should appear in the report output with Lot number and Lot Expiry date |
|  | Execute the concurrent program ‘GIL Print Commercial Invoices-Split(Print)’ from DE-OU-01 AR Invoices responsibility for Kite Drop Ship Order | Run the concurrent program | The report should be generated and bsting program should be fired. The invoice should appear in the report output with Lot number and Lot Expiry date |

# Open/Closed Issues

## Open Issues

## N/A

| **ID** | **Issue** | **Resolution** | **Responsibility** | **Target Date** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Closed Issues

## N/A

| **ID** | **Issue** | **Resolution** | **Resolved By** | **Closed Date** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

| REVISION HISTORY | | | | |
| --- | --- | --- | --- | --- |
| **Rev.** | | **Effective Date** | **Author** | **Description of Changes** | **Justification for Changes** |
| 16.0 | | <See EDMS> | Saumya Mishra | Updated the Approach section 1.3 for 12.2.6 upgrade | Modified the RTF layout to handle page-break issue for 12.2.6 upgrade |
| 15.0 | | 31 May 2018 | Pradeep Kumar | Change in the Legal Entity Address for Switzerland Template at 2.15.1.1 | Requirement for change in Legal entity address mentioned in REQ0386648 |
| 14.0 | | 11 May 2018 | Senthilkumar Shanmugam | CR1962 – Included a new calculated field to identify the Kite Invoice | CR1962 – Print Kite Registered address for Kite Invoices |
| 13.0 | | 29 Mar 2018 | Senthilkumar Shanmugam | CR1936 – Enhanced the Q\_LINE and Q\_LOT query to include Lot number and Lot Expiry date for Kite Drop Ship Orders | CR 1936 Kite Drop Ship Orders |
| 12.0 | | 18 Nov 2017 | Raja K U | Added point 32 in Section 2.18 to mention the changes to print credit mems ino invoice print report | As per ITSM - 32674 |
| 11.0 | | 19 Oct 2017 | Sankalpa Saha | Added point 31 in section 2.18 to mention the change related to email commercial invoices for non-EDI customers.  Added test cases under section 4 | As per requirement in CR 1688 – Email AR Transactions for Non-EDI Customers (ITSM-32027) |
| 10.0 | | 18 Aug 2017 | Ranajit Sinha | Added point 30 in section 2.18 to mention the changes for France E-Invoicing project. Inclusion of new parameter Print eInvoices in the “GIL Print Commercial Invoices-Split(Print)” program. | As per requirement in France E-Invoicing project |
| 9.0 | | 23 Jun 2017 | Sankalpa Saha | Added point 29 in section 2.18 to mention the change in invoice print for Turkey. The discount line suppression logic is also mentioned. | As per requirement in CRS 1814 – Turkey HCV Order Changes |
| 8.0 | | 07 Apr 2017 | Ranajit Sinha | TW\_SG\_SK EBS Implementation Project.   1. Added new template details for Taiwan. Section 26 2. Added new template details for Singapore. Section 27 3. Added modified template details for Hong Kong. Updated 2.15 Report Change Requests. Point 13 removed shipment header information from template   CR 1832: Print end customer information on AR Credit Memos  Added modified template details for US section 28 | Taiwan and Singapore EBS Roll Out  CR 1832: Print end customer information on AR Credit Memos |
| 7.0 | | 04-Apr-2017 | Madhu Vankudothu | Updated the latest KOREA template to remove decimal units in Section 25 | As per incident INC0467041, made changes to KOREA template to remove Decimal Units. |
| 6.0 | | 05 Jan 2017 | Sanjay Bedi | Updated the latest Italy template in Section 24 | As per SR# REQ0184923 to make modifications Imprint for NORWAY Template characters |
| 5.0 | | 20 Dec 2016 | Sanjay Bedi | Updated the latest Italy template in Section 23 | As per SR# REQ0189897 to make modifications Imprint for ITALY Template characters |
| 4.0 | | 14 Jul 2016 | Rupen Patil | Japan Localization changes  The following sections have been updated :-   1. Section 1.2 Changes for Japan on the objects 2. Section 1.3   Lookups and DFF for Japan   1. Section 2.3   Multi-language template addition   1. Section 2.4   Table and view usage for Japan   1. Section 2.5   Field Mapping   1. Section 2.9   SQL Statements  Formula columns | Japan Roll Out |
| 3.0 | | 12 Jul 2016 | Sunny Shanigaram | Newzeland Invoice Print to show discounts on order line. Updated 2.15 Report change Requests, Point 21 | As per SR#REQ0130219, Newzeland has started providing discounts to customers and therefore, Newzeland invoice should display discounts to show the correct amount |
| 2.0 | | 21 Jun 2016 | S. Prasad | Invoice Print capability to support multiple tax registrations. Updated Section 2.2 Report Change Requests, Point 20 | VAT Compliant Project, project # 206330. |
| 1.0 | | 13 Jun 2016 | Vikas Sathid | 1. New ITSDS to replace ITD-2447. 2. Updated the latest Germen template in Section 19 | 1. To remove the ITD from the archive workflow into a controlled workflow.  2. As per SR# REQ0114682 to make modifications Imprint for German Template characters |