Table of Contents

Scripts for: Concurrent Manager 2

Scripts for: FND - Script to add responsibility using fnd\_user\_pkg with validation 8

Script for: Purchasing 11

Scripts for : Apps Menu and Functions Related Scripts 16

Scripts for: Order Management 17

Scripts for: Responsabilities 24

Scripts for: Account Receivables 33

Scripts for: iExpenses 36

Scripts for: General Ledger 37

Scripts for: Cash Management 38

Scripts for: Payables 39

Scripts for: Inventory 40

## Scripts for: Concurrent Manager

- - - - - - - - - - - - - - - - Code begins here - - - - - - - - - - - - - - -

spool ccm.lst

prompt Step 1 Checking how many rows in FND\_CONCURRENT\_REQUEST.

select count(\*) from fnd\_concurrent\_requests;

prompt

-----------------------------------------

prompt Step 2 Checking how many rows in FND\_CONCURRENT\_PROCESSES table.

select count(\*) from fnd\_concurrent\_processes;

prompt

-----------------------------------------------

prompt Step 3 Checking sys.dual table which should have one and only one row.

select count(\*) from sys.dual;

prompt If you have more than one row in sys.dual, please delete it

prompt sql> delete rownum from SYS.DUAL;

Prompt rownum= the row number to delete

prompt

prompt

prompt

---------------------------------------------

prompt Step 4 Checking fnd\_dual. There must be at lest one row:

select count(\*) from fnd\_dual;

prompt If there are no record selected,

prompt Update fnd\_dual table to have at lest one record

prompt

----------------------------------------------

prompt Step 5 Checking the Internal Manager queue name "FNDICM" which should be=1

select concurrent\_queue\_id from fnd\_concurrent\_queues

where concurrent\_queue\_name='FNDICM';

prompt

----------------------------------------------

prompt Step 6 Checking for Active processes under the Internal Manager queue

prompt in fnd\_concurrent\_proceses table:

prompt

select a.concurrent\_queue\_name

, substr(b.os\_process\_id,0,10) "OS Proc"

, b.oracle\_process\_id "Oracle ID"

, b.process\_status\_code

from fnd\_concurrent\_queues a

, fnd\_concurrent\_processes b

where a.concurrent\_queue\_id=b.concurrent\_queue\_id

and a.concurrent\_queue\_name='FNDICM'

and b.process\_status\_code='A'

order by b.process\_status\_code;

prompt If any rows found with process\_status\_code with value = 'A' (= Active)

prompt The internal Manager will not start up ,so to avoide this issue

prompt update these rows to have process\_status\_code value ='K'(terminated)

prompt

prompt

-----------------------------------------

prompt Step 7 Checking for Active processes under the Standard Manager queue

prompt in fnd\_concurrent\_proceses table:

prompt

select a.concurrent\_queue\_name

, substr(b.os\_process\_id,0,10) "OS Proc"

, b.oracle\_process\_id "Oracle ID"

, b.process\_status\_code

from fnd\_concurrent\_queues a

, fnd\_concurrent\_processes b

where a.concurrent\_queue\_id=b.concurrent\_queue\_id

and a.concurrent\_queue\_name='STANDARD'

and b.process\_status\_code='A'

order by b.process\_status\_code;

prompt If any rows found with process\_status\_code with value = 'A' (= Active)

prompt The internal Manager will not start up ,so to avoide this issue

prompt update these rows to have process\_status\_code value ='K'(terminated)

prompt

prompt

------------------------------------------

prompt Step 8 Checking for Active processes under the Conflict Manager queue

prompt in fnd\_concurrent\_proceses table:

prompt

select a.concurrent\_queue\_name

, substr(b.os\_process\_id,0,10) "OS Proc"

, b.oracle\_process\_id "Oracle ID"

, b.process\_status\_code

from fnd\_concurrent\_queues a

, fnd\_concurrent\_processes b

where a.concurrent\_queue\_id=b.concurrent\_queue\_id

and a.concurrent\_queue\_name='FNDCRM'

and b.process\_status\_code='A'

order by b.process\_status\_code;

prompt If any rows found with process\_status\_code with value = 'A' (= Active)

prompt The internal Manager will not start up ,so to avoide this issue

prompt update these rows to have process\_status\_code value ='K'(terminated)

prompt

prompt

---------------------------------------------------

prompt Step 9 Checking Actual and Target Processes for Internal Manager:

select MAX\_PROCESSES,RUNNING\_PROCESSES

from FND\_CONCURRENT\_QUEUES

where CONCURRENT\_QUEUE\_NAME='FNDICM';

prompt If the MAX\_PROCESSES=RUNNING\_PROCESSES that means the manager is UP.

prompt

prompt

--------------------------------------------------------

prompt Step 10 Checking Actual and Target Processes for the Standard Manager:

select MAX\_PROCESSES,RUNNING\_PROCESSES

from FND\_CONCURRENT\_QUEUES

where CONCURRENT\_QUEUE\_NAME='STANDARD';

prompt If the MAX\_PROCESSES=RUNNING\_PROCESSES that means the manager is UP.

prompt

prompt

---------------------------------------------------------

prompt Step 11 Checking Actual and Target Processes for Conflict Resolution Manager:

select MAX\_PROCESSES,RUNNING\_PROCESSES

from FND\_CONCURRENT\_QUEUES

where CONCURRENT\_QUEUE\_NAME='FNDCRM';

prompt If the MAX\_PROCESSES=RUNNING\_PROCESSES that means the manager is UP.

prompt

prompt

---------------------------------------------------------

Prompt Step 12 Checking if the control\_code set to 'N':

select control\_code from fnd\_concurrent\_queues

where control\_code='N';

prompt

prompt If any rows selected, please update the table fnd\_concurrent\_queues:

prompt Update fnd\_concurrent\_queues set control\_code = null

prompt where control\_code ='N';

PROMPT Update fnd\_concurrent\_queues set target\_node = null;

PROMPT commit;

prompt

prompt

--------------------------------

PROMPT Step 13 Checking terminated processes:

PROMPT

select count (\*) from fnd\_concurrent\_requests

where status\_code='T';

prompt

prompt If you have terminated processes run the following sql statement:

prompt

prompt SQL> Update fnd\_concurrent\_requests

prompt set status\_code = 'E', phase\_code = 'C'

prompt where status\_code = 'T';

prompt

------------------------------------------

prompt Step 14 Checking pending requests:

select count(\*) from fnd\_concurrent\_requests

where status\_code='P';

prompt If any rows selected please run the following sql statement:

prompt SQL> Update fnd\_concurrent\_requests

prompt set status\_code = 'E', phase\_code = 'C'

prompt where status\_code = 'P';

prompt

------------------------------------------------------

prompt Step 15 Checking Running processes:

prompt

select count (\*) from fnd\_concurrent\_requests

where status\_code='R';

prompt

prompt If you have Running processes run the following sql statement

prompt SQL> Update fnd\_concurrent\_requests

prompt set status\_code = 'E', phase\_code = 'C'

prompt where status\_code = 'R';

prompt

------------------------------------------

prompt Step 16 Checking the PMON method, which should be set to LOCK:

prompt

select profile\_option\_id , profile\_option\_value

from FND\_PROFILE\_OPTION\_VALUES

where profile\_option\_id= (select profile\_option\_id

from FND\_PROFILE\_OPTIONS

where profile\_option\_name='CONC\_PMON\_METHOD');

prompt

prompt If the PROFILE\_OPTION\_VALUE was't LOCK please

prompt Reset PMON to LOCK by running afimpmon.sql script(The manager should be down)

prompt 1-At UNIX command prompt:

prompt 2-cd $FND\_TOP/sql

prompt 3-Log into SQLPLUS as apps/

prompt SQL> @afimpmon.sql

prompt prompt1:dual

prompt prompt2:LOCK (LOCK MUST BE ALL UPPERCASE)

prompt For Oracle Applications Release 11.5, when you check the PMON

prompt Method you may get no rows selected which is normal,

prompt because in apps 11.5 the Pmon Method is hard coded to Lock at

prompt the Operating System level.

prompt

prompt

-------------------------------------------------------

prompt Step-17 Checking how many FNDLIBR processes are running:

prompt -For Unix :From unix command prompt $ ps -ef grep -i fndlibr

prompt If you have any FNDLIBR processes running,please kill them before

prompt starting or shuting down the internal manager

prompt

prompt

prompt -For NT, through Task Manager, check the entries under the Processes tab

for FNDLIBR.exe processes.

prompt If there are any, Highlight and click [End Process] button to kill processes

prompt

----------------------------------------------------------

prompt Step-18 Checking how many "FND\_%"invalid objects:

select substr(owner,1, 12) owner, substr(object\_type,1,12) type,

substr(status,1,8) status, substr(object\_name, 1, 25) name

from dba\_objects

where object\_name like 'FND\_%'

and status='INVALID';

prompt If you have any invalied objects please see note#113947.1 via Metalink

prompt

--------------------------------------------------------------

prompt Step-19-How to find the PID in the O/S for request\_id:

prompt If you do not like to check this enter any number then click Enter to Exit

select r.request\_id, p.os\_process\_id

from FND\_CONCURRENT\_REQUESTS r,FND\_CONCURRENT\_PROCESSES p

where r.controlling\_manager = p.concurrent\_process\_id

and request\_id=&request\_id;

prompt

prompt Please upload the "ccm.lst" output to Support, Thanks.

prompt

spool off

- - - - - - - - - - - - - - - Code ends here - - - - - - - - - - - - - - - -

## Scripts for: FND - Script to add responsibility using fnd\_user\_pkg with validation

DECLARE

v\_user\_name VARCHAR2 (10) := 'Enter\_User\_Name';

v\_resp\_name VARCHAR2 (50) := 'Enter\_Existing\_Responsibility\_Name';

v\_req\_resp\_name VARCHAR2 (50) := 'Enter\_required\_Responsibility\_Name';

v\_user\_id NUMBER (10);

v\_resp\_id NUMBER (10);

v\_appl\_id NUMBER (10);

v\_count NUMBER (10);

v\_resp\_app VARCHAR2 (50);

v\_resp\_key VARCHAR2 (50);

v\_description VARCHAR2 (100);

RESULT BOOLEAN;

BEGIN

SELECT fu.user\_id, frt.responsibility\_id, frt.application\_id

INTO v\_user\_id, v\_resp\_id, v\_appl\_id

FROM fnd\_user fu,

fnd\_responsibility\_tl frt,

fnd\_user\_resp\_groups\_direct furgd

WHERE fu.user\_id = furgd.user\_id

AND frt.responsibility\_id = furgd.responsibility\_id

AND frt.LANGUAGE = 'US'

AND fu.user\_name = v\_user\_name

AND frt.responsibility\_name = v\_resp\_name;

fnd\_global.apps\_initialize (v\_user\_id, v\_resp\_id, v\_appl\_id);

SELECT COUNT (\*)

INTO v\_count

FROM fnd\_user fu,

fnd\_responsibility\_tl frt,

fnd\_user\_resp\_groups\_direct furgd

WHERE fu.user\_id = furgd.user\_id

AND frt.responsibility\_id = furgd.responsibility\_id

AND frt.LANGUAGE = 'US'

AND fu.user\_name = v\_user\_name

AND frt.responsibility\_name = v\_req\_resp\_name;

IF v\_count = 0

THEN

SELECT fa.application\_short\_name,

frv.responsibility\_key,

frv.description

INTO v\_resp\_app, v\_resp\_key, v\_description

FROM fnd\_responsibility\_vl frv, fnd\_application fa

WHERE frv.application\_id = fa.application\_id

AND frv.responsibility\_name = v\_req\_resp\_name;

fnd\_user\_pkg.addresp (username => v\_user\_name,

resp\_app => v\_resp\_app,

resp\_key => v\_resp\_key,

security\_group => 'STANDARD',

description => v\_description,

start\_date => SYSDATE - 1,

end\_date => NULL);

RESULT :=

fnd\_profile.SAVE (x\_name => 'APPS\_SSO\_LOCAL\_LOGIN',

x\_value => 'BOTH',

x\_level\_name => 'USER',

x\_level\_value => v\_user\_id);

RESULT :=

fnd\_profile.SAVE (x\_name => 'FND\_CUSTOM\_OA\_DEFINTION',

x\_value => 'Y',

x\_level\_name => 'USER',

x\_level\_value => v\_user\_id);

RESULT :=

fnd\_profile.SAVE (x\_name => 'FND\_DIAGNOSTICS',

x\_value => 'Y',

x\_level\_name => 'USER',

x\_level\_value => v\_user\_id);

RESULT :=

fnd\_profile.SAVE (x\_name => 'DIAGNOSTICS',

x\_value => 'Y',

x\_level\_name => 'USER',

x\_level\_value => v\_user\_id);

RESULT :=

fnd\_profile.SAVE (x\_name => 'FND\_HIDE\_DIAGNOSTICS',

x\_value => 'N',

x\_level\_name => 'USER',

x\_level\_value => v\_user\_id);

DBMS\_OUTPUT.put\_line (

'The responsibility added to the user '

|| v\_user\_name

|| ' is '

|| v\_req\_resp\_name);

COMMIT;

ELSE

DBMS\_OUTPUT.put\_line (

'The responsibility has already been added to the user');

END IF;

END;

## Script for: Purchasing

Purchase Requisition details

SELECT prh.segment1 "Req #", prh.creation\_date, prh.created\_by, poh.segment1 "PO #", ppx.full\_name "Requestor Name",prh.description "Req Description", prh.authorization\_status, prh.note\_to\_authorizer, prh.type\_lookup\_code, prl.line\_num,prl.line\_type\_id, prl.item\_description, prl.unit\_meas\_lookup\_code, prl.unit\_price, prl.quantity, prl.quantity\_delivered,prl.need\_by\_date, prl.note\_to\_agent, prl.currency\_code, prl.rate\_type, prl.rate\_date, prl.quantity\_cancelled, prl.cancel\_date,prl.cancel\_reason

FROM po\_requisition\_headers\_all prh,

po\_requisition\_lines\_all prl,

po\_req\_distributions\_all prd,

per\_people\_x ppx,

po\_headers\_all poh,

po\_distributions\_all pda

WHERE prh.requisition\_header\_id = prl.requisition\_header\_id

AND ppx.person\_id = prh.preparer\_id

AND prh.type\_lookup\_code = 'PURCHASE'

AND prd.requisition\_line\_id = prl.requisition\_line\_id

AND pda.req\_distribution\_id = prd.distribution\_id

AND pda.po\_header\_id = poh.po\_header\_id

AND TO\_CHAR (prh.creation\_date, 'YYYY') IN ('2010', '2011')

Internal Requisition details

SELECT prh.segment1 "Req #", prh.creation\_date, prh.created\_by, poh.segment1 "PO #", ppx.full\_name "Requestor Name",

prh.description "Req Description", prh.authorization\_status, prh.note\_to\_authorizer, prl.line\_num,

prl.line\_type\_id, prl.source\_type\_code, prl.item\_description, prl.unit\_meas\_lookup\_code, prl.unit\_price, prl.quantity, prl.quantity\_delivered,

prl.need\_by\_date, prl.note\_to\_agent, prl.currency\_code, prl.rate\_type, prl.rate\_date, prl.quantity\_cancelled, prl.cancel\_date,

prl.cancel\_reason

FROM po\_requisition\_headers\_all prh,

po\_requisition\_lines\_all prl,

po\_req\_distributions\_all prd,

per\_people\_x ppx,

po\_headers\_all poh,

po\_distributions\_all pda

WHERE prh.requisition\_header\_id = prl.requisition\_header\_id

AND ppx.person\_id = prh.preparer\_id

AND prh.type\_lookup\_code = 'INTERNAL'

AND prd.requisition\_line\_id = prl.requisition\_line\_id

AND pda.req\_distribution\_id (+) = prd.distribution\_id

AND pda.po\_header\_id = poh.po\_header\_id (+)

AND TO\_CHAR (prh.creation\_date, 'YYYY') IN ('2010', '2011')

Purchase Order details

SELECT

ph.SEGMENT1 po\_num

, ph.CREATION\_DATE

, hou.name "Operating Unit"

, ppx.full\_name "Buyer Name"

, ph.type\_lookup\_code "PO Type"

, plc.displayed\_field "PO Status"

, ph.COMMENTS

, pl.line\_num

, plt.order\_type\_lookup\_code "Line Type"

, NULL "Item Code"

, pl.item\_description

, pl.unit\_meas\_lookup\_code "UOM"

, pl.base\_unit\_price

, pl.unit\_price

, pl.quantity

, ood.organization\_code "Shipment Org Code"

, ood.organization\_name "Shipment Org Name"

, pv.vendor\_name supplier

, pvs.vendor\_site\_code

, (pl.unit\_price \* pl.quantity) "Line Amount"

, prh.segment1 req\_num

, prh.type\_lookup\_code req\_method

, ppx1.full\_name "Requisition requestor"

FROM po\_headers\_all ph

, po\_lines\_all pl

, po\_distributions\_all pda

, po\_vendors pv

, po\_vendor\_sites\_all pvs

, po\_distributions\_all pd

, po\_req\_distributions\_all prd

, po\_requisition\_lines\_all prl

, po\_requisition\_headers\_all prh

, hr\_operating\_units hou

, per\_people\_x ppx

, po\_line\_types\_b plt

, org\_organization\_definitions ood

, per\_people\_x ppx1

, po\_lookup\_codes plc

WHERE

1=1

AND TO\_CHAR(ph.creation\_date, 'YYYY') IN (2010, 2011)

AND ph.vendor\_id = pv.vendor\_id

AND ph.po\_header\_id = pl.po\_header\_id

AND ph.vendor\_site\_id = pvs.vendor\_site\_id

AND ph.po\_header\_id = pd.po\_header\_id

and pl.po\_line\_id = pd.po\_line\_id

AND pd.req\_distribution\_id = prd.distribution\_id (+)

AND prd.requisition\_line\_id = prl.requisition\_line\_id (+)

AND prl.requisition\_header\_id = prh.requisition\_header\_id (+)

and hou.organization\_id = ph.org\_id

and ph.agent\_id = ppx.person\_id

and pda.po\_header\_id = ph.po\_header\_id

and pda.po\_line\_id = pl.po\_line\_id

and pl.line\_type\_id = plt.line\_type\_id

and ood.organization\_id = pda.destination\_organization\_id

and ppx1.person\_id (+) = prh.preparer\_id

and plc.lookup\_type = 'DOCUMENT STATE'

and plc.LOOKUP\_CODE = ph.closed\_code

and pl.item\_id is null

UNION

-- Purchase Orders for inventory items

SELECT

ph.SEGMENT1 po\_num

, ph.CREATION\_DATE

, hou.name "Operating Unit"

, ppx.full\_name "Buyer Name"

, ph.type\_lookup\_code "PO Type"

, plc.displayed\_field "PO Status"

, ph.COMMENTS

, pl.line\_num

, plt.order\_type\_lookup\_code "Line Type"

, msi.segment1 "Item Code"

, pl.item\_description

, pl.unit\_meas\_lookup\_code "UOM"

, pl.base\_unit\_price

, pl.unit\_price

, pl.quantity

, ood.organization\_code "Shipment Org Code"

, ood.organization\_name "Shipment Org Name"

, pv.vendor\_name supplier

, pvs.vendor\_site\_code

, (pl.unit\_price \* pl.quantity) "Line Amount"

, prh.segment1 req\_num

, prh.type\_lookup\_code req\_method

, ppx1.full\_name "Requisition requestor"

FROM po\_headers\_all ph

, po\_lines\_all pl

, po\_distributions\_all pda

, po\_vendors pv

, po\_vendor\_sites\_all pvs

, po\_distributions\_all pd

, po\_req\_distributions\_all prd

, po\_requisition\_lines\_all prl

, po\_requisition\_headers\_all prh

, hr\_operating\_units hou

, per\_people\_x ppx

, mtl\_system\_items\_b msi

, po\_line\_types\_b plt

, org\_organization\_definitions ood

, per\_people\_x ppx1

, po\_lookup\_codes plc

WHERE

1=1

AND TO\_CHAR(ph.creation\_date, 'YYYY') IN (2010, 2011)

AND ph.vendor\_id = pv.vendor\_id

AND ph.po\_header\_id = pl.po\_header\_id

AND ph.vendor\_site\_id = pvs.vendor\_site\_id

AND ph.po\_header\_id = pd.po\_header\_id

and pl.po\_line\_id = pd.po\_line\_id

AND pd.req\_distribution\_id = prd.distribution\_id (+)

AND prd.requisition\_line\_id = prl.requisition\_line\_id (+)

AND prl.requisition\_header\_id = prh.requisition\_header\_id (+)

and hou.organization\_id = ph.org\_id

and ph.agent\_id = ppx.person\_id

and pda.po\_header\_id = ph.po\_header\_id

and pda.po\_line\_id = pl.po\_line\_id

and pl.line\_type\_id = plt.line\_type\_id

and ood.organization\_id = pda.destination\_organization\_id

and ppx1.person\_id (+) = prh.preparer\_id

and pda.destination\_organization\_id = msi.organization\_id (+)

and msi.inventory\_item\_id = nvl(pl.item\_id, msi.inventory\_item\_id)-- OR pl.item\_id is null)

and plc.lookup\_type = 'DOCUMENT STATE'

and plc.LOOKUP\_CODE = ph.closed\_code

and pl.item\_id is not null

Receiving transactions with PO and requisition information

SELECT

ph.segment1 po\_num,

ood.organization\_name,

pol.po\_line\_id,

pll.quantity,

rsh. receipt\_source\_code,

rsh. vendor\_id,

rsh. vendor\_site\_id,

rsh. organization\_id,

rsh. shipment\_num,

rsh. receipt\_num,

rsh. ship\_to\_location\_id,

rsh. bill\_of\_lading,

rsl.shipment\_line\_id,

rsl.QUANTITY\_SHIPPED,

rsl.QUANTITY\_RECEIVED ,

rct.transaction\_type,

rct.transaction\_id,

nvl(rct.source\_doc\_quantity,0) transaction\_qty

from rcv\_transactions rct

, rcv\_shipment\_headers rsh

, rcv\_shipment\_lines rsl

, po\_lines\_all pol

, po\_line\_locations\_all pll

, po\_headers\_all ph

, org\_organization\_definitions ood

where 1=1

and to\_char(rct.creation\_date, 'YYYY') in ('2010', '2011')

and rct.po\_header\_id = ph.po\_header\_id

and rct.po\_line\_location\_id = pll.line\_location\_id

and rct.po\_line\_id = pol.po\_line\_id

and rct.shipment\_line\_id=rsl.shipment\_line\_id

and rsl.shipment\_header\_id=rsh.shipment\_header\_id

and rsh.ship\_to\_org\_id = ood.organization\_id

order by rct.transaction\_id

## Scripts for : Apps Menu and Functions Related Scripts

SELECT DISTINCT

fmep.menu\_id,

DECODE (

fmep.function\_id,

NULL, DECODE (

fmec.function\_id,

NULL, DECODE (fmec1.function\_id,

NULL, 'No Func',

fmec1.function\_id),

fmec.function\_id),

fmep.function\_id)

funcID,

fff.user\_function\_name,

fff.description

FROM fnd\_form\_functions\_tl fff,

fnd\_menu\_entries fmec1,

fnd\_menu\_entries fmec,

fnd\_menu\_entries fmep

WHERE fmep.menu\_id = (SELECT menu\_id

FROM fnd\_menus

WHERE menu\_name = 'INV\_NAVIGATE' --Change the menu according to your requirement

AND ROWNUM = 1)

AND fmep.sub\_menu\_id = fmec.menu\_id(+)

AND fmec.sub\_menu\_id = fmec1.menu\_id(+)

AND fff.function\_id =

DECODE (

fmep.function\_id,

NULL, DECODE (

fmec.function\_id,

NULL, DECODE (fmec1.function\_id,

NULL, -999,

fmec1.function\_id),

fmec.function\_id),

fmep.function\_id)

ORDER BY DECODE (

fmep.function\_id,

NULL, DECODE (

fmec.function\_id,

NULL, DECODE (fmec1.function\_id,

NULL, 'No Func',

fmec1.function\_id),

fmec.function\_id),

fmep.function\_id)

## Scripts for: Order Management

Order header info

Select \* from oe\_order\_headers\_all ooh

where order\_number= :p\_order\_number

Operating unit info

select \* from hr\_operating\_units

where organization\_id=oe\_order\_headers\_all.org\_id

Order type info

select \* from apps.oe\_transaction\_types\_tl

where transaction\_type\_id=oe\_order\_headers\_all.order\_type\_id

Price list info

select \* from apps.qp\_list\_headers\_tl

where list\_header\_id=oe\_order\_headers\_all.price\_list\_id

select \* from apps.qp\_list\_lines

where list\_header\_id=oe\_order\_headers\_all.price\_list\_id

Find customer info

select \* from hz\_cust\_accounts hca

where cust\_account\_id=oe\_order\_headers\_all.sold\_to\_org\_id

select \* from apps.hz\_parties

where party\_id=hz\_cust\_accounts.party\_id

Find Ship to location info

select \* from hz\_cust\_site\_uses\_all

where site\_use\_id=oe\_order\_headers\_all.ship\_to\_org\_id

select \* from apps.hz\_cust\_acct\_sites\_all

where cust\_acct\_site\_id=hz\_cust\_site\_uses\_all.cust\_acct\_site\_id

select \* from hz\_party\_sites

where party\_site\_id=hz\_cust\_acct\_sites\_all.party\_site\_id

Find Bill to location

select \* from hz\_cust\_site\_uses\_all

where site\_use\_id=oe\_order\_headers\_all.invoice\_to\_org\_id

select \* from hz\_cust\_acct\_sites\_all

where cust\_acct\_site\_id=hz\_cust\_site\_uses\_all.cust\_acct\_site\_id

select \* from hz\_party\_sites

where party\_site\_id=hz\_cust\_acct\_sites\_all.party\_site\_id

actual address

select \* from hz\_locations

where location\_id=hz\_party\_sites.location\_id

Sales rep id

select name from apps.ra\_salesreps\_all salerep where

salesrep\_id = oe\_order\_headers\_all.salesrep\_id and rownum =1

Payment terms

select name from apps.ra\_terms\_tl

where term\_id =oe\_order\_headers\_all.payment\_term\_id

and language = 'US'

Order source

select name from apps.oe\_order\_sources

where order\_source\_id= oe\_order\_headers\_all.order\_source\_id

and enabled\_flag= 'Y'

Order Source Reference

select orig\_sys\_document\_ref from oe\_order\_headers\_all ooh

where order\_number='&oracle order number'

FOB Point Code

select lookup\_code from ar\_lookups

where lookup\_type = 'FOB' and enabled\_flag = 'Y'

and upper(meaning) = upper(oe\_order\_headers\_all.fob\_point\_code)

Freight terms

select lookup\_code from apps.oe\_lookups

where upper (lookup\_type) = 'FREIGHT\_TERMS' and enabled\_flag = 'Y'

and upper (lookup\_code) = upper (oe\_order\_headers\_all.freight\_terms\_code)

For sales channel code validation

select lookup\_code from apps.oe\_lookups

where lookup\_type = 'SALES\_CHANNEL' and enabled\_flag = 'Y'

upper(lookup\_code) = upper(oe\_order\_headers\_all.sales\_channel\_code)

Ship method

select ship\_method\_code from wsh.wsh\_carrier\_services

where ship\_method\_code = oe\_order\_headers\_all.shipping\_method\_code

Warehouse Info

select \* from org\_organization\_definitions

where organization\_id = oe\_order\_headers\_all.ship\_from\_org\_id

Sales order Lines Details

select \* from apps.oe\_order\_lines\_all

where header\_id=oe\_order\_headers\_all.header\_id

Transactional currency code

select ota.price\_list\_id, qhb.currency\_code

from ont.oe\_transaction\_types\_all ota, qp.qp\_list\_headers\_b qhb

where ota.transaction\_type\_id = oe\_order\_headers\_all.order\_type\_id

and ota.price\_list\_id = qhb.list\_header\_id(+)

and NVL(qhb.list\_type\_code, 'PRL') = 'PRL'

and qhb.currency\_code =oe\_order\_headers\_all.transactional\_curr\_code

Item info

select \* from apps.mtl\_system\_items\_b

where segment1 like oe\_order\_lines\_all.ordered\_item

and organization\_id=oe\_order\_lines\_all.ship\_from\_org\_id

UOM

select uom\_code from inv.mtl\_units\_of\_measure\_tl

where upper(uom\_code)= upper(oe\_order\_lines\_all.order\_quantity\_uom)

and language= 'US' and nvl(disable\_date, (sysdate + 1)) > sysdate

Item type code validation

select lookup\_code from apps.oe\_lookups

where upper(lookup\_type) = 'ITEM\_TYPE'

and enabled\_flag = 'Y'

and upper(lookup\_code)= oe\_order\_lines\_all.item\_type\_code

On hand quantities

select \* from apps.mtl\_onhand\_quantities

where inventory\_item\_id=oe\_order\_lines\_all.inventory\_item\_id

and organization\_id=oe\_order\_lines\_all.ship\_from\_org\_id

Shipping

select \* from wsh\_delivery\_details

where source\_header\_id=oe\_order\_headers\_all.header\_id

select \* from wsh\_delivery\_assignments

where delivery\_detail\_id=wsh\_delivery\_details.delivery\_detail\_id

select \* from wsh\_new\_deliveries

where delivery\_id=wsh\_delivery\_assignments.delivery\_id

select \* from wsh\_delivery\_legs

where delivery\_id=wsh\_new\_deliveries.delivery\_id

select \* from wsh\_trip\_stops wts

where stop\_id=wsh\_delivery\_legs.pick\_up\_stop\_id

select \* from wsh\_trips wt

where trip\_id=wsh\_trip\_stops.trip\_id

select \* from org\_organization\_definitions

where organization\_id = wsh\_new\_deliveries.organization\_id

Material transactions

select \* from mtl\_material\_transactions

where inventory\_item\_id=oe\_order\_lines\_all.inventory\_item\_id

and organization\_id=oe\_order\_lines\_all.ship\_from\_org\_id

select \* from mtl\_transaction\_types

where transaction\_type\_id = mmt.transaction\_type\_id

select \* from apps.mtl\_txn\_source\_types

where transaction\_source\_type\_id= mmt.transaction\_source\_type\_id

mmt = mtl\_material\_transactions

Join between OM, WSH, AR Tables

SELECT ooh.order\_number

,ool.line\_id

,ool.ordered\_quantity

,ool.shipped\_quantity

,ool.invoiced\_quantity

,wdd.delivery\_detail\_id

,wnd.delivery\_id

,rctl.interface\_line\_attribute1

,rctl.interface\_line\_attribute3

,rctl.interface\_line\_attribute6

,rct.org\_id

,rct.creation\_date

,trx\_number

,rctl.quantity\_ordered

,rct.interface\_header\_context

FROM oe\_order\_headers\_all ooh

,oe\_order\_lines\_all ool

,wsh\_delivery\_details wdd

,wsh\_new\_deliveries wnd

,wsh\_delivery\_assignments wda

,ra\_customer\_trx\_all rct

,ra\_customer\_trx\_lines\_all rctl

WHERE ooh.header\_Id=ool.header\_id

AND wdd.source\_header\_id=ooh.header\_id

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

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

AND rctl.interface\_line\_attribute1=to\_char(ooh.order\_number)

AND rctl.interface\_line\_attribute6=to\_char(ool.line\_id)

AND rctl.interface\_line\_attribute3=to\_char(wnd.delivery\_id)

AND rctl.customer\_trx\_id=rct.customer\_trx\_id

AND rct.interface\_header\_context='ORDER ENTRY'

Purchase release concurrent program will transfer the details from OM to PO requisitions interface. The following query will verify the same:

SELECT interface\_source\_code,

interface\_source\_line\_id,

quantity,

destination\_type\_code,

transaction\_id,

process\_flag,

request\_id,

TRUNC (creation\_date)

FROM po\_requisitions\_interface\_all

WHERE interface\_source\_code = 'ORDER ENTRY'

AND interface\_source\_line\_id IN (SELECT drop\_ship\_source\_id

FROM oe\_drop\_ship\_sources

WHERE header\_id = &order\_hdr\_id

AND line\_id = &order\_line\_id);

The following sql is used to review the requisition, sales order, and receipt number. It shows the joins between various tables in Internal Sales order (ISO)

SELECT porh.segment1,

porl.line\_num,

pord.distribution\_num,

ooh.order\_number

sales\_order,

ool.line\_number so\_line\_num,

rsh.receipt\_num,

rcv.transaction\_type

FROM oe\_order\_headers\_all ooh,

po\_requisition\_headers\_all porh,

po\_requisition\_lines\_all porl,

po\_req\_distributions\_all pord,

oe\_order\_lines\_all ool,

po\_system\_parameters\_all posp,

rcv\_shipment\_headers rsh,

rcv\_transactions rcv

WHERE ooh.order\_source\_id = posp.order\_source\_id

AND porh.org\_id = posp.org\_id

AND porh.requisition\_header\_id = ool.source\_document\_id

AND porl.requisition\_line\_id = ool.source\_document\_line\_id

AND porh.requisition\_header\_id = porl.requisition\_header\_id

AND porl.requisition\_line\_id = pord.requisition\_line\_id

AND porl.requisition\_line\_id = rcv.requisition\_line\_id

AND pord.distribution\_id = rcv.req\_distribution\_id

AND rcv.shipment\_header\_id = rsh.shipment\_header\_id

PO Matched Invoice Query

SELECT POV.VENDOR\_NAME

, AIA.INVOICE\_TYPE\_LOOKUP\_CODE AS INVOICE\_TYPE

, AIA.INVOICE\_NUM

, AIA.INVOICE\_DATE

, AID.ACCOUNTING\_DATE AS GL\_DATE

, AIA.INVOICE\_CURRENCY\_CODE

, PHA.SEGMENT1 AS PO\_NUMBER

, RSH.RECEIPT\_NUM

, NVL(AIA.EXCHANGE\_RATE,1) AS EXCHANGE\_RATE

, GCC.CONCATENATED\_SEGMENTS

, AID.AMOUNT AS ENTERED\_AMOUNT

, AID.BASE\_AMOUNT AS ACCOUNTED\_AMOUNT

FROM AP\_INVOICE\_DISTRIBUTIONS\_ALL AID

, AP\_INVOICES\_ALL AIA

, PO\_VENDORS POV

, GL\_CODE\_COMBINATIONS\_KFV GCC

, PO\_DISTRIBUTIONS\_ALL PDA

, PO\_HEADERS\_ALL PHA

, (SELECT TRANSACTION\_ID, SHIPMENT\_HEADER\_ID

FROM RCV\_TRANSACTIONS WHERE TRANSACTION\_TYPE = 'RECEIVE') RCV

, RCV\_SHIPMENT\_HEADERS RSH

WHERE

AID.PO\_DISTRIBUTION\_ID IS NOT NULL

AND AID.BASE\_AMOUNT IS NOT NULL

AND AID.INVOICE\_ID = AIA.INVOICE\_ID

AND AIA.VENDOR\_ID = POV.VENDOR\_ID

AND AID.DIST\_CODE\_COMBINATION\_ID = GCC.CODE\_COMBINATION\_ID

AND TRUNC(AID.ACCOUNTING\_DATE) BETWEEN '01-APR-2011' AND '31-MAR-2013'

AND AID.PO\_DISTRIBUTION\_ID = PDA.PO\_DISTRIBUTION\_ID

AND PDA.PO\_HEADER\_ID = PHA.PO\_HEADER\_ID

AND AID.RCV\_TRANSACTION\_ID = RCV.TRANSACTION\_ID (+)

AND RCV.SHIPMENT\_HEADER\_ID = RSH.SHIPMENT\_HEADER\_ID (+)

UNION ALL

SELECT POV.VENDOR\_NAME

, AIA.INVOICE\_TYPE\_LOOKUP\_CODE AS INVOICE\_TYPE

, AIA.INVOICE\_NUM

, AIA.INVOICE\_DATE

, AID.ACCOUNTING\_DATE AS GL\_DATE

, AIA.INVOICE\_CURRENCY\_CODE

, PHA.SEGMENT1 AS PO\_NUMBER

RSH.RECEIPT\_NUM

, NVL(AIA.EXCHANGE\_RATE,1) AS EXCHANGE\_RATE

, GCC.CONCATENATED\_SEGMENTS

, AID.AMOUNT AS ENTERED\_AMOUNT

, AID.AMOUNT AS ACCOUNTED\_AMOUNT

FROM AP\_INVOICE\_DISTRIBUTIONS\_ALL AID

, AP\_INVOICES\_ALL AIA

, PO\_VENDORS POV

, GL\_CODE\_COMBINATIONS\_KFV GCC

, PO\_DISTRIBUTIONS\_ALL PDA

, PO\_HEADERS\_ALL PHA

, (SELECT TRANSACTION\_ID, SHIPMENT\_HEADER\_ID

FROM RCV\_TRANSACTIONS WHERE TRANSACTION\_TYPE = 'RECEIVE') RCV

, RCV\_SHIPMENT\_HEADERS RSH

WHERE

AID.PO\_DISTRIBUTION\_ID IS NOT NULL

AND AID.BASE\_AMOUNT IS NULL

AND AID.INVOICE\_ID = AIA.INVOICE\_ID

AND AIA.VENDOR\_ID = POV.VENDOR\_ID

AND AID.DIST\_CODE\_COMBINATION\_ID = GCC.CODE\_COMBINATION\_ID

AND TRUNC(AID.ACCOUNTING\_DATE) BETWEEN '01-APR-2011' AND '31-MAR-2013'

AND AID.PO\_DISTRIBUTION\_ID = PDA.PO\_DISTRIBUTION\_ID

AND PDA.PO\_HEADER\_ID = PHA.PO\_HEADER\_ID

AND AID.RCV\_TRANSACTION\_ID = RCV.TRANSACTION\_ID (+)

AND RCV.SHIPMENT\_HEADER\_ID = RSH.SHIPMENT\_HEADER\_ID (+)

ORDER BY INVOICE\_NUM, GL\_DATE

## Scripts for: Responsabilities

//\*

1. Responsibilities Listing

Purpose/Description:

Retrieve a list of all responsibilities.

Parameters

None

\*//

SELECT

(SELECT application\_short\_name

FROM fnd\_application fa

WHERE fa.application\_id = frt.application\_id)

application

, frt.responsibility\_id

, frt.responsibility\_name

FROM apps.fnd\_responsibility\_tl frt;

//\*

2. Menus Listing

Purpose/Description:

To see the Menus associated with a given responsibility

Parameters

responsibility\_id that you can retrieve from query nr 1 (Responsibilities Listing)

\*//

SELECT DISTINCT

a.responsibility\_name

, c.user\_menu\_name

FROM

apps.fnd\_responsibility\_tl a

, apps.fnd\_responsibility b

, apps.fnd\_menus\_tl c

, apps.fnd\_menus d

, apps.fnd\_application\_tl e

, apps.fnd\_application f

WHERE

a.responsibility\_id(+) = b.responsibility\_id

AND a.responsibility\_id = 50103

AND b.menu\_id = c.menu\_id

AND b.menu\_id = d.menu\_id

AND e.application\_id = f.application\_id

AND f.application\_id = b.application\_id

AND a.LANGUAGE = ‘US’;

//\*

3. Submenu And Function Listing

Purpose/Description:

By using this query you can check function and submenus attached to a specific menu

Parameters

User\_menu\_name that you can get by running query 2 (Menu Listing)

\*//

SELECT

c.prompt

, c.description

FROM

apps.fnd\_menus\_tl a

, fnd\_menu\_entries\_tl c

WHERE

a.menu\_id = c.menu\_id

AND a.user\_menu\_name = ‘Navigator Menu – System Administrator GUI’;

SELECT c.prompt, c.description

FROM apps.fnd\_menus\_tl a, fnd\_menu\_entries\_tl c

WHERE a.menu\_id = c.menu\_id AND a.user\_menu\_name = 'F4 UK PAY Navigator';

//\*

4.User and Assigned Responsibility Listing

Purpose/Description:

You can use this query to check responsibilities assigned to users.

Parameters

None

\*//

SELECT UNIQUE

u.user\_id

, SUBSTR (u.user\_name, 1, 30) user\_name

, SUBSTR (r.responsibility\_name, 1, 60) responsiblity

, SUBSTR (a.application\_name, 1, 50) application

FROM

fnd\_user u

, fnd\_user\_resp\_groups g

, fnd\_application\_tl a

, fnd\_responsibility\_tl r

WHERE

g.user\_id(+) = u.user\_id

AND g.responsibility\_application\_id = a.application\_id

AND a.application\_id = r.application\_id

AND g.responsibility\_id = r.responsibility\_id

ORDER BY

SUBSTR (user\_name, 1, 30)

, SUBSTR (a.application\_name, 1, 50)

, SUBSTR (r.responsibility\_name, 1, 60);

//\*

5. Responsibility and assigned request group listing

Purpose/Description:

To find responsibility and assigned request groups.

Every responsibility contains a request group (The request group is basis of submitting requests)

Parameters

None

\*//

SELECT

responsibility\_name responsibility

, request\_group\_name

, frg.description

FROM

fnd\_request\_groups frg

, fnd\_responsibility\_vl frv

WHERE

frv.request\_group\_id = frg.request\_group\_id

ORDER BY responsibility\_name

//\*

6. Profile option with modification date and user

Purpose/Description:

Query that can be used to audit profile options.

Parameters

None

\*//

SELECT

t.user\_profile\_option\_name

, profile\_option\_value

, v.creation\_date

, v.last\_update\_date

, v.creation\_date – v.last\_update\_date "Change Date"

, (SELECT UNIQUE user\_name

FROM fnd\_user

WHERE user\_id = v.created\_by) "Created By"

, (SELECT user\_name

FROM fnd\_user

WHERE user\_id = v.last\_updated\_by) "Last Update By"

FROM

fnd\_profile\_options o

, fnd\_profile\_option\_values v

, fnd\_profile\_options\_tl t

WHERE

o.profile\_option\_id = v.profile\_option\_id

AND o.application\_id = v.application\_id

AND start\_date\_active <= SYSDATE

AND NVL (end\_date\_active, SYSDATE) >= SYSDATE

AND o.profile\_option\_name = t.profile\_option\_name

AND level\_id = 10001

AND t.LANGUAGE IN (SELECT language\_code

FROM fnd\_languages

WHERE installed\_flag = ‘B’

UNION

SELECT nls\_language

FROM fnd\_languages

WHERE installed\_flag = ‘B’)

ORDER BY user\_profile\_option\_name;

//\*

7. Forms personalization Listing

Purpose/Description:

To get modified profile options.

Personalization is a feature available in 11.5.10.X.

Parameters

None

\*//

SELECT

ffft.user\_function\_name "User Form Name"

, ffcr.SEQUENCE

, ffcr.description

, ffcr.rule\_type

, ffcr.enabled

, ffcr.trigger\_event

, ffcr.trigger\_object

, ffcr.condition

, ffcr.fire\_in\_enter\_query

, (SELECT user\_name

FROM fnd\_user fu

WHERE fu.user\_id = ffcr.created\_by) "Created By”

FROM

fnd\_form\_custom\_rules ffcr

, fnd\_form\_functions\_vl ffft

WHERE ffcr.ID = ffft.function\_id

ORDER BY 1;

//\*

8. Patch Level Listing

Purpose/Description:

Query that can be used to view the patch level status of all modules

Parameters

None

\*//

SELECT

a.application\_name

, DECODE (b.status, ‘I’, ‘Installed’, ‘S’, ‘Shared’, ‘N/A’) status

, patch\_level

FROM

apps.fnd\_application\_vl a

, apps.fnd\_product\_installations b

WHERE

a.application\_id = b.application\_id;

//\*

9. Request attached to responsibility listing

Purpose/Description:

To see all requests attached to a responsibility

Parameters

None

\*//

SELECT

responsibility\_name

, frg.request\_group\_name

, fcpv.user\_concurrent\_program\_name

, fcpv.description

FROM

fnd\_request\_groups frg

, fnd\_request\_group\_units frgu

, fnd\_concurrent\_programs\_vl fcpv

, fnd\_responsibility\_vl frv

WHERE

frgu.request\_unit\_type = ‘P’

AND frgu.request\_group\_id = frg.request\_group\_id

AND frgu.request\_unit\_id = fcpv.concurrent\_program\_id

AND frv.request\_group\_id = frg.request\_group\_id

ORDER BY responsibility\_name;

//\*

10. Request listing application wise

Purpose/Description:

View all request types application wise

Parameters

None

\*//

SELECT

fa.application\_short\_name

, fcpv.user\_concurrent\_program\_name

, description

, DECODE (fcpv.execution\_method\_code

,’B', ‘Request Set Stage Function’

,’Q', ‘SQL\*Plus’

,’H', ‘Host’

,’L', ‘SQL\*Loader’

,’A', ‘Spawned’

,’I', ‘PL/SQL Stored Procedure’

,’P', ‘Oracle Reports’

,’S', ‘Immediate’

,fcpv.execution\_method\_code) exe\_method

, output\_file\_type

, program\_type

, printer\_name

, minimum\_width

, minimum\_length

, concurrent\_program\_name

, concurrent\_program\_id

FROM

fnd\_concurrent\_programs\_vl fcpv

, fnd\_application fa

WHERE

fcpv.application\_id = fa.application\_id

ORDER BY description

//\*

11. Count Reports per module

Purpose/Description:

To Count Reports

Parameters

None

\*//

SELECT

fa.application\_short\_name

, DECODE (fcpv.execution\_method\_code

,’B', ‘Request Set Stage Function’

,’Q', ‘SQL\*Plus’

,’H', ‘Host’

,’L', ‘SQL\*Loader’

,’A', ‘Spawned’

,’I', ‘PL/SQL Stored Procedure’

,’P', ‘Oracle Reports’

,’S', ‘Immediate’

,fcpv.execution\_method\_code) exe\_method

, COUNT (concurrent\_program\_id) COUNT

FROM

fnd\_concurrent\_programs\_vl fcpv

, fnd\_application fa

WHERE

fcpv.application\_id = fa.application\_id

GROUP BY

fa.application\_short\_name

, fcpv.execution\_method\_code

ORDER BY 1;

//\*

12. Request Status Listing

Purpose/Description:

This query returns report/request processing time

Parameters

None

\*//

SELECT

f.request\_id

, pt.user\_concurrent\_program\_name user\_concurrent\_program\_name

, f.actual\_start\_date actual\_start\_date

, f.actual\_completion\_date actual\_completion\_date

, floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)

|| ‘ HOURS ‘ ||

floor((((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60) -

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)\*3600)/60)

|| ‘ MINUTES ‘ ||

round((((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60) -

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)\*3600 -

(floor((((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60) -

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)\*3600)/60)\*60) ))

|| ‘ SECS ‘ time\_difference

, DECODE(p.concurrent\_program\_name

,’ALECDC’

,p.concurrent\_program\_name||’['||

f.description||']‘

,p.concurrent\_program\_name) concurrent\_program\_name

, decode(f.phase\_code

,’R',’Running’

,’C',’Complete’

,f.phase\_code) Phase

, f.status\_code

FROM

apps.fnd\_concurrent\_programs p

, apps.fnd\_concurrent\_programs\_tl pt

, apps.fnd\_concurrent\_requests f

WHERE

f.concurrent\_program\_id = p.concurrent\_program\_id

AND f.program\_application\_id = p.application\_id

AND f.concurrent\_program\_id = pt.concurrent\_program\_id

AND f.program\_application\_id = pt.application\_id

AND pt.language = USERENV(‘Lang’)

AND f.actual\_start\_date is not null

ORDER by f.actual\_completion\_date-f.actual\_start\_date desc;

//\*

13. User and responsibility listing

Purpose/Description:

Check responsibilities assigned to users

Parameters

None

\*//

SELECT UNIQUE

u.user\_id

, SUBSTR (u.user\_name, 1, 30) user\_name

, SUBSTR (r.responsibility\_name, 1, 60) responsiblity

, SUBSTR (a.application\_name, 1, 50) application

FROM

fnd\_user u

, fnd\_user\_resp\_groups g

, fnd\_application\_tl a

, fnd\_responsibility\_tl r

WHERE g.user\_id(+) = u.user\_id

AND g.responsibility\_application\_id = a.application\_id

AND a.application\_id = r.application\_id

AND g.responsibility\_id = r.responsibility\_id

–AND a.application\_name like ‘%Order Man%’

ORDER BY SUBSTR (user\_name, 1, 30),

SUBSTR (a.application\_name, 1, 50),

SUBSTR (r.responsibility\_name, 1, 60)

//\*

14. Applied Patch Listing

Purpose/Description:

Check Current Applied Patches

Parameters

None

\*//

SELECT

patch\_name

, patch\_type

, maint\_pack\_level

, creation\_date

FROM applsys.ad\_applied\_patches

ORDER BY creation\_date DESC

Users Connected through application

SELECT user\_name username,

description name,

to\_char(b.first\_connect,'MM/DD/RR HH24:MI') firstconnect,

to\_char(b.last\_connect,'MM/DD/RR HH24:MI') lastconnect

FROM apps.fnd\_user a,

(SELECT MIN (first\_connect) first\_connect,

MAX (last\_connect) last\_connect, last\_updated\_by user\_id

FROM apps.icx\_sessions

GROUP BY last\_updated\_by) b

WHERE a.user\_id = b.user\_id

AND last\_connect>SYSDATE-3/12

ORDER BY 4 DESC

Function Listing

SELECT function\_id, user\_function\_name, creation\_date, description

FROM applsys.fnd\_form\_functions\_tl

order by order by user\_function\_name;

Request Attached To Responsibility Listing

SELECT responsibility\_name , frg.request\_group\_name,

fcpv.user\_concurrent\_program\_name, fcpv.description

FROM fnd\_request\_groups frg,

fnd\_request\_group\_units frgu,

fnd\_concurrent\_programs\_vl fcpv,

fnd\_responsibility\_vl frv

WHERE frgu.request\_unit\_type = 'P'

AND frgu.request\_group\_id = frg.request\_group\_id

AND frgu.request\_unit\_id = fcpv.concurrent\_program\_id

AND frv.request\_group\_id = frg.request\_group\_id

ORDER BY responsibility\_name;

Request Listing Application Wise

SELECT fa.application\_short\_name, fcpv.user\_concurrent\_program\_name,

description,

DECODE (fcpv.execution\_method\_code,

'B', 'Request Set Stage Function',

'Q', 'SQL\*Plus',

'H', 'Host',

'L', 'SQL\*Loader',

'A', 'Spawned',

'I', 'PL/SQL Stored Procedure',

'P', 'Oracle Reports',

'S', 'Immediate',

fcpv.execution\_method\_code

) exe\_method,

output\_file\_type, program\_type, printer\_name, minimum\_width,

minimum\_length, concurrent\_program\_name, concurrent\_program\_id

FROM fnd\_concurrent\_programs\_vl fcpv, fnd\_application fa

WHERE fcpv.application\_id = fa.application\_id

ORDER BY description

Count Module Wise Reports

SELECT fa.application\_short\_name,

DECODE (fcpv.execution\_method\_code,

'B', 'Request Set Stage Function',

'Q', 'SQL\*Plus',

'H', 'Host',

'L', 'SQL\*Loader',

'A', 'Spawned',

'I', 'PL/SQL Stored Procedure',

'P', 'Oracle Reports',

'S', 'Immediate',

fcpv.execution\_method\_code

) exe\_method,

COUNT (concurrent\_program\_id) COUNT

FROM fnd\_concurrent\_programs\_vl fcpv, fnd\_application fa

WHERE fcpv.application\_id = fa.application\_id

GROUP BY fa.application\_short\_name, fcpv.execution\_method\_code

ORDER BY 1;

Request Status Listing

SELECT f.request\_id , pt.user\_concurrent\_program\_name user\_concurrent\_program\_name

, f.actual\_start\_date actual\_start\_date

, f.actual\_completion\_date actual\_completion\_date,

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)

|| ' HOURS ' ||

floor((((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60) -

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)\*3600)/60)

|| ' MINUTES ' ||

round((((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60) -

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)\*3600 -

(floor((((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60) -

floor(((f.actual\_completion\_date-f.actual\_start\_date)\*24\*60\*60)/3600)\*3600)/60)\*60) ))

|| ' SECS ' time\_difference

, DECODE(p.concurrent\_program\_name,'ALECDC',p.concurrent\_program\_name||'['||f.description||']',p.concurrent\_program\_name)concurrent\_program\_name

, decode(f.phase\_code,'R','Running','C','Complete',f.phase\_code) Phase

, f.status\_code

FROM apps.fnd\_concurrent\_programs p

, apps.fnd\_concurrent\_programs\_tl pt

, apps.fnd\_concurrent\_requests f

WHERE f.concurrent\_program\_id = p.concurrent\_program\_id

and f.program\_application\_id = p.application\_id

and f.concurrent\_program\_id = pt.concurrent\_program\_id

and f.program\_application\_id = pt.application\_id

AND pt.language = USERENV('Lang')

and f.actual\_start\_date is not null

ORDER by f.actual\_completion\_date-f.actual\_start\_date desc;

## Scripts for: Account Receivables

Run the following SQL Query to get all active AR Memo Lines.

Navigation: Receivables Manager > Setup > Transactions > Memo Lines

=> Set the Org context first.

begin

mo\_global.set\_policy\_context('S',<org\_id>);

end;

/

Script:

SELECT hou.name operating\_unit,

amlv.name,

amlv.description,

amlv.line\_type type,

amlv.tax\_code tax\_classification,

amlv.uom\_code unit\_of\_measure,

amlv.unit\_std\_price unit\_list\_price,

gcck.concatenated\_segments revenue\_account,

amlv.start\_date,

amlv.end\_date

FROM ar\_memo\_lines\_vl amlv,

gl\_code\_combinations\_kfv gcck,

hr\_operating\_units hou

WHERE end\_date IS NULL

AND hou.organization\_id = amlv.org\_id

AND amlv.GL\_ID\_REV = gcck.CODE\_COMBINATION\_ID;

1. Provide AR Transaction to the following script.

It lists all CM's applied against a single AR Transaction.

SELECT rcta2.trx\_number CMs\_Applied\_to\_AR\_Trx\_No

FROM RA\_CUSTOMER\_TRX\_ALL rcta1,

RA\_CUSTOMER\_TRX\_ALL rcta2,

AR\_RECEIVABLE\_APPLICATIONS\_ALL araa

WHERE rcta1.trx\_number = '&Enter\_AR\_Txn\_No'

AND rcta1.customer\_trx\_id = araa.APPLIED\_CUSTOMER\_TRX\_ID

AND rcta2.customer\_trx\_id = araa.customer\_trx\_id;

2. Provide Credit Memo number to the following query

It lists all AR Transactions/Invoices applied with this Credit Memo given

SELECT rcta2.trx\_number AR\_Trx\_Nos

FROM RA\_CUSTOMER\_TRX\_ALL rcta1,

RA\_CUSTOMER\_TRX\_ALL rcta2,

AR\_RECEIVABLE\_APPLICATIONS\_ALL araa

WHERE rcta1.trx\_number = '&Enter\_Credit\_Memo'

AND rcta1.customer\_trx\_id = araa.customer\_trx\_id

AND rcta2.customer\_trx\_id = araa.APPLIED\_CUSTOMER\_TRX\_ID;

AR INVOICE QUERY

select

'Opening Balance' "BATCH SOURCE NAME"

,rtl.invoice\_currency\_code "CURRENCY CODE"

,(select name from ra\_cust\_trx\_types\_all where cust\_trx\_type\_id=rtl.cust\_trx\_type\_id and

rownum=1)customer\_trx\_type\_name

,ac.customer\_name customer\_name

,ac.customer\_number customer\_number

,rtl.trx\_number trx\_number

,rtl.trx\_date transaction\_date

,rgd.Gl\_date gl\_date

,rgd.percent dist\_percent

,(SELECT hcs.location

FROM hz\_parties hp

,hz\_cust\_accounts hca

,hz\_cust\_acct\_sites\_all hcas

,hz\_party\_sites hps

,HZ\_CUST\_SITE\_USES\_ALL hcs

WHERE hca.cust\_account\_id=hcas.cust\_account\_id

AND hcas.party\_site\_id=hps.party\_site\_id

AND hcs.cust\_acct\_site\_id=hcas.cust\_acct\_site\_id

AND hps.party\_id = hp.party\_id

AND hcs.site\_use\_code='BILL\_TO'

AND hca.cust\_account\_id=rtl.bill\_to\_customer\_id

AND HCS.ORG\_ID = rtl.org\_id

AND HCS.STATUS='A'

AND hcas.status='A'

AND hca.status = 'A'

AND upper(hp.party\_name) = trim(upper(ac.CUSTOMER\_NAME))

and rownum=1)location\_number

,rtl.comments "INV\_DESC"

,rtl.interface\_header\_attribute1

,(select name from ra\_terms where term\_id=rtl.term\_id and rownum=1) "TERM\_NAME"

,rct.line\_number "LINE\_NUMBER"

,rct.quantity\_invoiced "LINE\_QUANTITY"

,rct.unit\_selling\_price "UNIT\_SELLING\_PRICE"

,(rct.revenue\_amount) line\_amount

,(rct.extended\_amount) dist\_amount

,(select concatenated\_segments from gl\_code\_combinations\_kfv where

code\_combination\_id=rgd.code\_combination\_id)acccount

,(select name from hr\_operating\_units where organization\_id=rtl.org\_id)"OPERATING\_UNIT"

from ra\_customer\_trx\_all rtl,ar\_customers ac,RA\_CUSTOMER\_TRX\_LINES\_ALL

rct,RA\_CUST\_TRX\_LINE\_GL\_DIST\_ALL rgd

where ac.customer\_id=rtl.bill\_to\_customer\_id

and rct.customer\_trx\_id = rtl.customer\_trx\_id

and rct.customer\_trx\_line\_id=rgd.customer\_trx\_line\_id

and rtl.customer\_trx\_id=rgd.customer\_trx\_id

order by rtl.org\_id

AR RECEIPT QUERY

select

a.name "RECEIPT\_METHOD\_NAME"

,acr.receipt\_number

,ac.customer\_name

,ac.customer\_number

,acr.receipt\_date

,acr.type

,(SELECT hcs.location

FROM hz\_parties hp

,hz\_cust\_accounts hca

,hz\_cust\_acct\_sites\_all hcas

,hz\_party\_sites hps

,HZ\_CUST\_SITE\_USES\_ALL hcs

WHERE hca.cust\_account\_id=hcas.cust\_account\_id

AND hcas.party\_site\_id=hps.party\_site\_id

AND hcs.cust\_acct\_site\_id=hcas.cust\_acct\_site\_id

AND hps.party\_id = hp.party\_id

AND hcs.site\_use\_code='BILL\_TO'

AND hcs.site\_use\_id=acr.customer\_site\_use\_id

AND HCS.ORG\_ID = acr.org\_id

AND HCS.STATUS='A'

AND hcas.status='A'

AND hca.status = 'A'

AND upper(hp.party\_name) = trim(upper(ac.CUSTOMER\_NAME))

and rownum=1)location\_number

,acr.amount

,(select name from hr\_operating\_units where organization\_id=acr.org\_id)"operating\_unit"

,acr.currency\_code "CURRENCY CODE"

,'Opening Balance' "REMITANCE\_BANK\_NAME"

,cba.bank\_account\_num "REMITANCE\_BANK\_ACC\_NUM"

,acr.status "RECEIPT TYPE"

from

AR\_RECEIPT\_METHODS a ,

AR\_CASH\_RECEIPTS\_ALL acr,ar\_customers ac,CE\_BANK\_ACCT\_USES\_all

REMIT\_BANK,CE\_BANK\_ACCOUNTS cba

where acr.pay\_from\_customer = ac.customer\_id(+)

AND a.receipt\_method\_id=acr.receipt\_method\_id

AND REMIT\_BANK.BANK\_ACCT\_USE\_ID (+) = acr.REMIT\_BANK\_ACCT\_USE\_ID

AND remit\_bank.bank\_account\_id = CBA.bank\_account\_id (+)

order by acr.org\_id;

## Scripts for: iExpenses

Use the following sql script/query to find Signing Limits you assign to employees who are responsible for approving expense reports entered in Oracle Self-Service Expenses

SQL Query/Script:

SELECT pap.full\_name,

awsla.cost\_center,

awsla.org\_id,

hou.name organization\_name,

awsla.signing\_limit

FROM ap\_web\_signing\_limits\_all awsla,

per\_all\_people\_f pap,

hr\_organization\_units hou

WHERE awsla.employee\_id = pap.person\_id

AND awsla.org\_id = hou.organization\_id

AND pap.effective\_start\_date = (SELECT max(pap1.effective\_start\_date)

FROM apps.per\_all\_people\_f pap1

WHERE pap1.person\_id = pap.person\_id)

AND awsla.document\_type = 'APEXP'

ORDER BY hou.name,

pap.full\_name,

awsla.cost\_center,

awsla.signing\_limit;

## Scripts for: General Ledger

Following SQL query helps you to find Recurring Journals created in a Period

Also, shows who Created them and who Posted the same along with Dr Account, Cr Account & GL Code Combination information.

SQL Script:

=> Provide Period Name to the following script. Eg. 'Jan-12'

SELECT gjh.period\_name,

gjh.name JOURNAL,

fu1.user\_name created\_by,

gjh.date\_created,

DECODE (gjl.status, 'P', 'Posted', 'U', 'Unposted') status,

fu2.user\_name posted\_by,

gjh.posted\_date,

gjl.je\_line\_num,

gcck.concatenated\_segments GL\_CODE\_COMBINATION,

gjl.accounted\_dr,

gjl.accounted\_cr,

gjl.description

FROM gl\_je\_headers gjh, gl\_je\_lines gjl, gl\_code\_combinations\_kfv gcck, fnd\_user fu1, fnd\_user fu2

WHERE gjh.period\_name in ('&Enter\_Period\_Name')

AND gjh.je\_source = 'Recurring'

AND gjh.je\_header\_id = gjl.je\_header\_id

AND gcck.code\_combination\_id = gjl.code\_combination\_id

AND fu1.user\_id = gjh.created\_by

AND fu2.user\_id = gjh.last\_updated\_by;

## Scripts for: Cash Management

When we upgrade from 11.5.10 to R12, Bank Branches, Bank Accounts of type INTERNAL/PAYROLL did NOT migrate/upgrade.

Use the following script to identify the same.

Scripts:

1. Query to find the Branches that did not upgrade:

SELECT bb.bank\_branch\_id,

bb.bank\_name,

bb.bank\_branch\_name,

bb.country

FROM ap\_bank\_branches bb, ce\_upg\_bank\_rec cb, ce\_bank\_branches\_v cbv

WHERE cb.source\_pk\_id(+) = bb.bank\_branch\_id

AND cb.party\_id = cbv.branch\_party\_id(+)

AND cb.bank\_entity\_type(+) = 'BRANCH'

AND cbv.bank\_branch\_name IS NULL

ORDER BY bb.bank\_name;

2. Query to find the Bank Accounts that did not upgrade:

SELECT pay.bank\_account\_id, pay.bank\_account\_name, pay.bank\_account\_num

FROM ap\_bank\_accounts\_all pay,

ce\_bank\_accounts cash,

ce\_upg\_bank\_accounts cuba

WHERE pay.bank\_account\_id = cuba.source\_pk\_id(+)

AND cuba.ce\_bank\_account\_id = cash.bank\_account\_id(+)

AND cash.bank\_account\_name IS NULL

ORDER BY pay.bank\_account\_name;

Note: You can run following queries to reconcile between 11i and R12

11i

select count(\*) from ap\_bank\_accounts\_all;

select count(\*) from ap\_bank\_account\_uses\_all;

R12

select count(\*) from IBY\_EXT\_BANK\_ACCOUNTS;

select count(\*) from CE\_BANK\_ACCOUNTS;

select count(\*) from CE\_BANK\_ACCT\_USES\_ALL;

## Scripts for: Payables

AP INVOICE QUERY

select a.vendor\_name "SUPPLIER NAME"

,d.vendor\_site\_code "SUPPLIER SITE"

,b.invoice\_num "INVOICE NUMBER"

,b.invoice\_date "INVOICE DATE"

,b.invoice\_currency\_code "CURRENCY CODE"

,b.gl\_date,substr(b.description,1,50) DESCRIPTION

,b.invoice\_type\_lookup\_code "TRANSACTION TYPE"

,(c.amount) "AMOUNT"

,(select concatenated\_segments from gl\_code\_combinations\_kfv

where code\_combination\_id=c.dist\_code\_combination\_id)ACCOUNT

,(select name from hr\_operating\_units where organization\_id=b.org\_id)OPERATING\_UNIT\_NAME

from ap\_suppliers a,ap\_supplier\_sites\_all d,ap\_invoices\_all b,ap\_invoice\_distributions\_all c

where a.vendor\_id=b.vendor\_id

and c.invoice\_id=b.invoice\_id

and a.vendor\_id=d.vendor\_id

order by b.invoice\_type\_lookup\_code

## Scripts for: Inventory

ON-HAND Quantities Query

SELECT

MTL\_SYSTEM\_ITEMS\_B.ORGANIZATION\_ID,

ORG\_ORGANIZATION\_DEFINITIONS.ORGANIZATION\_NAME,

MTL\_ONHAND\_QUANTITIES\_DETAIL.SUBINVENTORY\_CODE,

MTL\_SYSTEM\_ITEMS\_B.SEGMENT1 ITEM\_NO,

MTL\_SYSTEM\_ITEMS\_B.DESCRIPTION,

MTL\_SYSTEM\_ITEMS\_B.PRIMARY\_UOM\_CODE UOM,

MTL\_ONHAND\_QUANTITIES\_DETAIL.TRANSACTION\_QUANTITY ON\_HAND

,MTL\_MATERIAL\_TRANSACTIONS.TRANSACTION\_QUANTITY ON\_HAND

,MTL\_MATERIAL\_TRANSACTIONS.TRANSACTION\_QUANTITY \*

MTL\_MATERIAL\_TRANSACTIONS.TRANSACTION\_COST TOTAL\_COST

FROM

MTL\_SYSTEM\_ITEMS\_B

,ORG\_ORGANIZATION\_DEFINITIONS

, MTL\_ONHAND\_QUANTITIES\_DETAIL

,mtl\_material\_transactions

WHERE

MTL\_SYSTEM\_ITEMS\_B.ORGANIZATION\_ID=ORG\_ORGANIZATION\_DEFINITIONS.ORGANIZATION\_ID

ANDMTL\_SYSTEM\_ITEMS\_B.INVENTORY\_ITEM\_ID=MTL\_ONHAND\_QUANTITIES\_DETAIL.INVENTORY\_ITEM\_ID

AND

MTL\_SYSTEM\_ITEMS\_B.ORGANIZATION\_ID=MTL\_ONHAND\_QUANTITIES\_DETAIL.ORGANIZATION\_ID

AND

MTL\_SYSTEM\_ITEMS\_B.ORGANIZATION\_ID=MTL\_MATERIAL\_TRANSACTIONS.ORGANIZATION\_ID

AND

MTL\_SYSTEM\_ITEMS\_B.INVENTORY\_ITEM\_ID=MTL\_MATERIAL\_TRANSACTIONS.INVENTORY\_ITEM\_ID

AND MTL\_ONHAND\_QUANTITIES\_DETAIL.INVENTORY\_ITEM\_ID =

MTL\_MATERIAL\_TRANSACTIONS.INVENTORY\_ITEM\_ID

AND MTL\_ONHAND\_QUANTITIES\_DETAIL.ORGANIZATION\_ID =

MTL\_MATERIAL\_TRANSACTIONS.ORGANIZATION\_ID

AND

MTL\_ONHAND\_QUANTITIES\_DETAIL.create\_transaction\_id=MTL\_MATERIAL\_TRANSACTIONS.transaction\_id

AND ORG\_ORGANIZATION\_DEFINITIONS.ORGANIZATION\_ID =

MTL\_MATERIAL\_TRANSACTIONS.ORGANIZATION\_ID

AND ORG\_ORGANIZATION\_DEFINITIONS.ORGANIZATION\_ID =

MTL\_ONHAND\_QUANTITIES\_DETAIL.ORGANIZATION\_ID

ORDER BY 1, 3, 4

Receipt Query

SELECT DISTINCT rcta.trx\_number

invoice\_no,rcta.EXCHANGE\_RATE\_TYPE,rcta.EXCHANGE\_RATE,rcta.exchange\_date,

rcta.bill\_to\_customer\_id cust\_id,rcta.customer\_trx\_id l\_trx\_id,

rcta.trx\_date receipt\_date, apsa.due\_date,

apsa.amount\_due\_remaining receipt\_amount

FROM ra\_customer\_trx\_all rcta,

ar\_payment\_schedules\_all apsa,

ra\_batch\_sources\_all rba

WHERE rcta.batch\_source\_id = rba.batch\_source\_id

AND rcta.customer\_trx\_id = apsa.customer\_trx\_id

---AND rcta.orig\_system\_batch\_name = 'TABS'

AND apsa.amount\_due\_remaining > 0

AND UPPER (apsa.status) = 'OP'

AND rcta.INVOICE\_CURRENCY\_CODE = 'MA'

--AND UPPER (apsa.CLASS) = 'INV'

--AND rcta.org\_id = 101

ORDER BY rcta.bill\_to\_customer\_id,

rcta.bill\_to\_customer\_id,

rcta.trx\_date,

apsa.due\_date,

rcta.trx\_number;