Amar Ujala Repeated Issue List

Issue: Pl. provide approval for given new agencies for billing rate & cover price.

This agency is started from 01-Feb-2015, so, pl enter cover price and billing rate from 01-Feb-15.

121	LKN – Sultanpur	574	5947960	Gangaram Yadav	219873	Baldirai

							Cover Price												
Org Code	Dak Name	DAK Code	Cust ID	Customer Name	Ship To	Location Name	Mon	Tue	Wed	Thur	Fri	Sat	Sun	Mon	Tue	Wed	Thur	Fri	Sat	Sun
549	LKU – UDAN UPC	676	5947890	Amit News Agency	218950	Newada			1.00							0.67				
121	LKN – Sultanpur	574	5947960	Gangaram Yadav	219873	Baldirai	4.00	4.00	4.00	4.00	4.00	4.00	4.00	2.68	2.68	2.68	2.68	2.68	2.68	2.68

Solution: edit au_order_process_trans where AU_ORD_AR_INT_CUST_ID=5947960

Note: usually this kind of issue is solved by Priyanka. But she said I cannot do this for previous dates.

As the user wanted it from feb 1 2015 and issue has come to us on 3 feb.

Issue: Approval for delete VNS-Rupayan suppliment one entry PO Dt.- 30-01-15

Please allow delete one entry of VNS rupayan suppliment " Amar Ujala Suppliment Print Order" as below :

Date : 30-Jan-15 VNS PO - 90,000 copy will be delete.

File also enclosed for your reference.

Solution: delete from au_suppliment where AU_SUPPLIMENT_PUB_DATE='30-jan-2015' and AU_SUPPLIMENT_UNIT='VNS' and AU_SUPPLIMENT_REGULAR=90000

Issue Please Change following in City Dak Cust Supply

Unit	Dak	Cust ID	Ship To	Customer Name	Location	Supply Change Show	Actual Supply Change Show
117	125	5770755	220206	Bala Ji News Agency(Quila)	C.B.Ganj(P)	270	135

Current 125 Dak Total PO 54182 & As Per delivery Sheet 54182

Actual 125 Dak Total PO 54182 & As Per delivery Sheet 54182

Note : Compact Jan 2015 Bill Register not shows bill number while posting is OK

Solution: delete from au_supply_change
where AU_SUPCHNG_SHIP_TO='220206'
and AU_SUPCHNG_POST_CHANGES=135.

Below table is for confirmation.

select * from au_order_process_trans where AU_ORD_AR_INT_SHIP_TO=220206 and AU_ORD_AR_INT_PUB_DATE='04-feb-2015'.

Issue: select *from po_line_locations_all@hrtoebs

where po_header_id=1

select *from po_headers_all@hrtoebs

where SEGMENT1='300000'

PO number=Segment1

Change match_id = r into P

----------FINAL----------

```
update po_line_locations_all set MATCH_OPTION ='P'
where po_header_id in (select po_header_id from po_headers_all
where ATTRIBUTE9 is not null
and ATTRIBUTE9 like 'WM%')
and MATCH_OPTION ='R'
```

Issue: Please provide all units MRR and Print order data dated 21-Jan-15 to 31-Jan-15 in excel file

Solution: select *from CR_MRRDET

where MR_DATE between '21-jan-2015' and '31-jan-2015'

select *from AU_PRINT_ORDER

where AU_PRINT_ORDER_PUB_DATE between '21-jan-2015' and '31-jan-2015'

then copy and paste it into XLS.

Issue: Asset number 259647 is coming in 'Amar Ujala Rectification Asset Listing Report' in Jan-15 which should not come. Please rectify the same.

Solution: select * from fa_additions

Attribute 14 – y.

Issue : Vodafone monthly bill.

Solution: select line_number,attribute4,amount,unit,naturalac,accounting_date,tax_code_id,description,invoice_num from ap_inv_lines_t where 1 = 2 for update

Insert a ROW in excel sheet provided (Column A downwards has to be blanked)

Execute and commit

```
  select   c2.unit||'.999.999.'||c2.tax_code_id||'.'||c2.naturalac||'.99.9999' from ap_inv_lines_t c2 where flag is null
 minus
select
gc.segment1||'.'||gc.segment2||'.'||gc.segment3||'.'||gc.segment4||'.'||gc.segment5||'.'||gc.segment6||'.'||gc.segment7  from gl_code_combinations gc.
```

In toad if you get any data, then mail it to Priyanka and Vinaysheel for 'OPEN GL CODE ACCOUNT COMBINATION'.

Once it is done.

VODA/JV/Feb-15 is last column in excel sheet copy and paste down there.

```
begin
XXA_AP_INT_p_001('VODA/JV/Feb-15');
end;
```

```
select * from ap_invoices_interface
where INVOICE_DATE='31-mar-2015'
```

This date comes in the mail as GL date.

Check in apps production.

In front End go to PAYABLES
RUN
PAYABLE OPEN INTERFACE_ IMPORT (NOT SURE IMPORT IS THERE OR NOT)

SELECT quick invoices.

SUBMIT AND CHECK OUTPUT.

(INSERT COLUMN IN EXCEL SHEET.

PASTE THE SHEET IN PL/SQL.

UNLOCK.

CLICK SELECT AND PASTE IT.

COMMIT;

CODE IF ANY THEN

EMPTY

WE NEED TO CHECK IT ON INTERFACE ALSO.

```
select * from ap_invoices_interface
where invoice_num like 'VODA%'
```

```
select sum(amount) from ap_invoice_lines_interface
where invoice_id in (649433,649434)
```

aucustom.xxau_order_lines@ebstohr b

aucustom.xxau_order_headers@ebstohr

SECOND QUERY

CODE IF ANY THEN

EMPTY

WE NEED TO CHECK IT ON INTERFACE ALSO.

)

Issue: Pre Approve Purchase Order

Solution:

Save this query in c as named PO. This will run in SQL Plus, only

In SQL Plus, SQL> @C:\po\poxrespo

Please enter the PO number to reset : 349109 (as example)

Please enter the organization id to which the PO belongs (Default NULL) : 101

Do you want to delete the action history since the last approval ? (Y/N) Y

```
REM dbdrv: none
      /*=========================================================================+
           |  Copyright (c) 1997 Oracle Corporation Redwood Shores, California, USA|
           |                       All rights reserved.                  |

+=========================================================================*/

 /* $Header: poxrespo.sql 115.7 2006/09/19 05:28:13 ambansal noship $ */
SET SERVEROUTPUT ON
SET VERIFY OFF;


/* PLEASE READ NOTE 390023.1 CAREFULLY BEFORE EXECUTING THIS SCRIPT.

 This script will:
* reset the document to incomplete/requires reapproval status.
* delete/update action history as desired (refere note 390023.1 for more details).
* abort all the related workflows

If using encumbrance it will also:
* correct po encumbrance values if corrupted
* Suggest gl manual postings to have the gl data in synch.
*/

set serveroutput on size 100000
prompt
prompt
accept sql_po_number prompt 'Please enter the PO number to reset : ';
accept sql_org_id default NULL prompt 'Please enter the organization id to which the PO belongs
(Default NULL) : ';
```

```
accept delete_act_hist prompt 'Do you want to delete the action history since the last approval ? (Y/N)
';
prompt

declare
x_stmt varchar2(3000);
NAME_ALREADY_USED EXCEPTION;
PRAGMA Exception_Init(NAME_ALREADY_USED,-955);
begin
x_stmt:= 'create table po_manual_postings_temp
        (po_distribution_id number,
         budget_account_id  number,
         debit_amount       number,
         timestamp          date,
             type           varchar2(5),
             account          varchar2(285))';

EXECUTE immediate(x_stmt);
exception
WHEN NAME_ALREADY_USED then
  dbms_output.put_line('table already exists, hence not creating again');
when others then
  dbms_output.put_line('An exception occured creating po_manual_postings_temp table');
  dbms_output.put_line(SQLCODE || '-'|| SQLERRM);
  dbms_output.put_line ('Please contact Oracle Support');
  rollback;
  return;

end;
/


DECLARE

/* select only the POs which are in preapproved, in process state and are not finally closed
   cancelled */

CURSOR potoreset(po_number varchar2, x_org_id number) is
SELECT wf_item_type, wf_item_key, po_header_id, segment1,
      revision_num, type_lookup_code,approved_date
FROM po_headers_all
WHERE segment1 = po_number
```

```
and NVL(org_id,-99) = NVL(x_org_id,-99)
-- bug 5015493: Need to allow reset of blankets and PPOs also.
-- and type_lookup_code = 'STANDARD'
and authorization_status IN ('IN PROCESS', 'PRE-APPROVED')
and NVL(cancel_flag, 'N') = 'N'
and NVL(closed_code, 'OPEN') <> 'FINALLY_CLOSED';

/* select the max sequence number with NULL action code */

CURSOR maxseq(id number, subtype po_action_history.object_sub_type_code%type) is
SELECT nvl(max(sequence_num), 0)
FROM   po_action_history
WHERE  object_type_code IN ('PO', 'PA')
AND    object_sub_type_code = subtype
AND    object_id = id
AND    action_code is NULL;

/* select the max sequence number with submit action */

CURSOR poaction(id number, subtype po_action_history.object_sub_type_code%type) is
SELECT nvl(max(sequence_num), 0)
FROM   po_action_history
WHERE  object_type_code IN ('PO', 'PA')
AND    object_sub_type_code = subtype
AND    object_id = id
AND    action_code = 'SUBMIT';

cursor wfstoabort(st_item_type varchar2,st_item_key varchar2) is
select level,item_type,item_key,end_date
from wf_items
start with
   item_type = st_item_type and
   item_key =  st_item_key
connect by
   prior item_type = parent_item_type and
   prior item_key = parent_item_key
order by level desc;

wf_rec wfstoabort%ROWTYPE;

submitseq po_action_history.sequence_num%type;
nullseq po_action_history.sequence_num%type;
```

```
x_organization_id number ;
x_po_number varchar2(15);
po_enc_flag varchar2(1);
x_open_notif_exist varchar2(1);
pos potoreset%ROWTYPE;

x_progress varchar2(500);
x_cont varchar2(10);
x_active_wf_exists varchar2(1);
l_delete_act_hist varchar2(1);
l_change_req_exists varchar2(1);
l_res_seq po_action_history.sequence_num%TYPE;
l_sub_res_seq po_action_history.sequence_num%TYPE;
l_res_act po_action_history.action_code%TYPE;
l_del_res_hist varchar2(1);


/* For encumbrance actions */

NAME_ALREADY_USED EXCEPTION;
PRAGMA Exception_Init(NAME_ALREADY_USED,-955);
X_STMT VARCHAR2(2000);
disallow_script VARCHAR2(1);

TYPE enc_tbl_number is TABLE OF NUMBER;
TYPE enc_tbl_flag   is TABLE OF VARCHAR2(1);

l_dist_id                enc_tbl_number;
l_enc_flag               enc_tbl_flag;
l_enc_amount             enc_tbl_number;
l_gl_amount              enc_tbl_number;
l_manual_cand                    enc_tbl_flag;
l_req_dist_id            enc_tbl_number;
l_req_enc_flag           enc_tbl_flag;
l_req_enc_amount         enc_tbl_number;
l_req_gl_amount          enc_tbl_number;
l_req_qty_bill_del       enc_tbl_number;
l_rate_table             enc_tbl_number;
l_price_table            enc_tbl_number;
l_qty_ordered_table      enc_tbl_number;
l_req_price_table        enc_tbl_number;
```

```
l_req_encumbrance_flag          varchar2(1);
l_purch_encumbrance_flag varchar2(1);
l_remainder_qty        NUMBER;
l_bill_del_amount         NUMBER;
l_req_bill_del_amount   NUMBER;
l_qty_bill_del     NUMBER;
l_timestamp               date;
l_eff_quantity            NUMBER;
l_rate                    NUMBER;
l_price                   NUMBER;
l_ordered_quantity        NUMBER;
l_tax                     NUMBER;
l_amount                  NUMBER;
l_precision               fnd_currencies.precision%type;
l_min_acc_unit            fnd_currencies.minimum_accountable_unit%TYPE;
l_approved_flag           po_line_locations_all.approved_flag%TYPE;
i                         number;
j                         number;
k                         number;



BEGIN

select '&delete_act_hist'
          into l_delete_act_hist
          from dual;

select &sql_org_id
  into x_organization_id
  from dual;

select '&sql_po_number'
  into x_po_number
  from dual;


x_progress := '010: start';

   begin
   select 'Y'
    into x_open_notif_exist
```

```
    from dual
    where exists (select 'open notifications'
                          from wf_item_activity_statuses wias,
                                wf_notifications wfn,
                                po_headers_all poh
                          where wias.notification_id is not null
                            and wias.notification_id = wfn.group_id
                            and wfn.status = 'OPEN'
                            and wias.item_type = 'POAPPRV'
                            and wias.item_key = poh.wf_item_key
                            and NVL(poh.org_id,-99) = NVL(x_organization_id,-99)
                            and poh.segment1=x_po_number
                            and poh.authorization_status IN ('IN PROCESS', 'PRE-APPROVED'));
   exception
   when NO_DATA_FOUND then
     null;
   end;

x_progress := '020: selected open notif';

if (x_open_notif_exist = 'Y') then
   dbms_output.put_line('  ');
   dbms_output.put_line('An Open notification exists for this document, you may want to use the
notification to process this document. Do not commit if you wish to use the notification');
end if;

begin
select 'Y'
  into l_change_req_exists
  from dual
  where exists (select 'po with change request'
                        from po_headers_all h
                        where h.segment1 = x_po_number
                          and nvl(h.org_id, -99) = NVL(x_organization_id, -99)
                          and h.change_requested_by in ('REQUESTER', 'SUPPLIER'));
exception
   when NO_DATA_FOUND then
     null;
end;

if (l_change_req_exists = 'Y') then
   dbms_output.put_line('  ');
```

```
   dbms_output.put_line('ATTENTION !!! There is an open change request against this PO. You should
respond to the notification for the same.');
   return;
--   dbms_output.put_line('If you are running this script unaware of the change request, Please
ROLLBACK');
end if;




open potoreset(x_po_number, x_organization_id);

fetch potoreset into pos;
if potoreset%NOTFOUND then
   dbms_output.put_line('No PO with PO Number '||x_po_number ||
                                ' exists in org '||to_char(x_organization_id)
                                || ' which requires to be reset');
   return;
end if;
close potoreset;

/* check if any distribution with USSGL code exists - if it does then exit */

disallow_script := 'N';
begin
select 'Y'
  into disallow_script
  from dual
  where exists (select 'dist with USSGL code'
                        from po_distributions_all
                       where po_header_id = pos.po_header_id
                         and ussgl_transaction_code is not null);

Exception
when NO_DATA_FOUND then
   null;
end;

if disallow_script = 'Y' then
  dbms_output.put_line('You have a public sector installation and USSGL transaction codes are
used');
  dbms_output.put_line('The reset script is not allowed in such a scenario, please contact Oracle
Support');
   return;
```

```
end if;


    dbms_output.put_line('Processing '||pos.type_lookup_code
                         ||' PO Number: '
                         ||pos.segment1);
    dbms_output.put_line('.....................................');


    begin
     select 'Y'
       into x_active_wf_exists
       from wf_items wfi
      where wfi.item_type = pos.wf_item_type
            and wfi.item_key = pos.wf_item_key
            and wfi.end_date is null;

    exception
    when NO_DATA_FOUND then
    x_active_wf_exists := 'N';
    end;

    if (x_active_wf_exists = 'Y') then
       dbms_output.put_line('Aborting Workflow...');
       open wfstoabort(pos.wf_item_type,pos.wf_item_key);
       loop
       fetch wfstoabort into wf_rec;
            if wfstoabort%NOTFOUND then
              close wfstoabort;
              exit;
            end if;

            if (wf_rec.end_date is null) then
            BEGIN
             WF_Engine.AbortProcess(wf_rec.item_type, wf_rec.item_key);
       EXCEPTION
        WHEN OTHERS THEN
            dbms_output.put_line(' workflow not aborted :'
                    ||wf_rec.item_type ||'-'||wf_rec.item_key);

       END;
```

```
        end if;
        end loop;
end if;

dbms_output.put_line('Updating PO Status..');
UPDATE po_headers_all
   SET authorization_status = decode(pos.approved_date, NULL, 'INCOMPLETE',
                              'REQUIRES REAPPROVAL'),
    wf_item_type = NULL,
    wf_item_key = NULL,
        approved_flag = decode(pos.approved_date, NULL, 'N', 'R')
 WHERE po_header_id = pos.po_header_id;

OPEN  maxseq(pos.po_header_id, pos.type_lookup_code);
FETCH maxseq into nullseq;
CLOSE maxseq;

OPEN  poaction(pos.po_header_id, pos.type_lookup_code);
FETCH poaction into submitseq;
CLOSE poaction;
IF nullseq > submitseq THEN

      if nvl(l_delete_act_hist,'N') = 'N' then
        Update po_action_history
          set action_code = 'NO ACTION',
              action_date = trunc(sysdate),
              note = 'updated by reset script on '||to_char(trunc(sysdate))
    WHERE object_id = pos.po_header_id
      AND  object_type_code = decode(pos.type_lookup_code,
                                          'STANDARD','PO',
                                          'PLANNED', 'PO', --future plan to enhance for planned PO
                                          'PA')
      AND object_sub_type_code = pos.type_lookup_code
      AND sequence_num = nullseq
      AND action_code is NULL;
  else

        Delete po_action_history
         where object_id = pos.po_header_id
           and object_type_code = decode(pos.type_lookup_code,
                                          'STANDARD','PO',
                                          'PLANNED', 'PO', --future plan to enhance for planned PO
```

```
                                                    'PA')
                and object_sub_type_code = pos.type_lookup_code
                and sequence_num >= submitseq
                and sequence_num <= nullseq;

        end if;

    END IF;

    dbms_output.put_line('Done Approval Processing.');


 select nvl(purch_encumbrance_flag,'N')
   into l_purch_encumbrance_flag
   from financials_system_params_all fspa
   where NVL(fspa.org_id,-99) = NVL(x_organization_id,-99);

   if (l_purch_encumbrance_flag='N')
     -- bug 5015493 : Need to allow reset for blankets also
     OR (pos.type_lookup_code = 'BLANKET') then

     if (pos.type_lookup_code = 'BLANKET') then
          dbms_output.put_line('document reset successfully');
          dbms_output.put_line('If you are using Blanket encumbrance, Please ROLLBACK, else
COMMIT');
     else
          dbms_output.put_line('document reset successfully');
          dbms_output.put_line('please COMMIT data');
     end if;
     return;
   end if;

 -- <start description>

 /*
 dist flag   dist amount   gl tables amount  solution
 ---------   -----------   ----------------  --------

 Y                  NN                NN
              amount same          No problem
              amount not same
```

```
Y               NN              N       Update N, 0
Y               N               NN      Update N    eligible for manual posting
Y               N               N       Update N
N               NN              NN
            amount same         Update Y, 0 eligible for manual posting
                amount not same         Update N, 0 eligible for manual posting

N               NN              N       Update N, 0
N               N               NN      Eligible for manual posting.
N               N               N       No problem


quantity*price (at shipment level) *rate + (nrtax/Quant.ordered)*effective_quantity*rate



 Exit and advice the user to run the program import Journals
  if not done already */


-- <end description>



 x_progress := '040 starting enc actions ';

disallow_script := 'N';
begin
select 'Y'
into disallow_script
from dual
where not exists (select 'encumbrance data'
             from gl_je_lines
            where reference_3 in
               (select to_char(po_distribution_id)
                  from po_distributions_all pod
                 where pod.po_header_id = pos.po_header_id)
              and reference_1 = 'PO'
           );
EXCEPTION
when NO_DATA_FOUND THEN
```

```
  NULL;
end;

if disallow_script = 'Y' then
   dbms_output.put_line('Could not find Encumbrance data - it is possible that ');
   dbms_output.put_line('The concurrent program - program create Journals has not been run.');
   dbms_output.put_line('If that is the case, please rollback, run the program and reexecute ');
   dbms_output.put_line('the script, Otherwise commit the data.');
end if;

/* Collect all the data into the plsql tables */

x_progress := '050 enc actions- collect data ';
begin
select pod.po_distribution_id,
       nvl(pod.encumbered_flag,'N'),
       nvl(pod.encumbered_amount,'0'),
       sum(nvl(gl.entered_dr,0)-nvl(gl.entered_cr,0)),
       'N' -- initialize manual posting flag
bulk collect into l_dist_id,l_enc_flag, l_enc_amount, l_gl_amount,l_manual_cand
from po_distributions_all pod, gl_je_lines gl, gl_je_headers glh,po_line_locations_all pll
where pod.po_header_id = pos.po_header_id
  and pod.line_location_id = pll.line_location_id
  and nvl(pll.closed_code,'OPEN') <> 'FINALLY CLOSED'
  and nvl(pll.cancel_flag,'N') = 'N'
  and gl.reference_1(+) = 'PO'
  and glh.actual_flag = 'E'
  and gl.reference_3(+) = to_char(pod.po_distribution_id)
  and gl.je_header_id = glh.je_header_id
  and glh.je_category = 'Purchases'
  and glh.je_source = 'Purchasing'
  and nvl(pod.prevent_encumbrance_flag,'N') <> 'Y'
  and pod.ussgl_transaction_code is null  -- ignore distributions will ussgl transaction code
group by pod.po_distribution_id, pod.encumbered_flag, pod.encumbered_amount;

EXCEPTION
when NO_DATA_FOUND then
dbms_output.put_line (' No enc data exists ');
RETURN;
end ;

 /*Initialize the counter for table that stores dist ids
```

```
        which require a manual posting of data */

/* find the precision and minimum accountable unit of the functional
   currency, so that it can be used later for rounding */
select fc.precision, fc.minimum_accountable_unit
  into l_precision, l_min_acc_unit
from   fnd_currencies fc,
       gl_sets_of_books gsob,
       financials_system_params_all fspa
where NVL(fspa.org_id,-99) = NVL(x_organization_id,-99)
  and fspa.set_of_books_id = gsob.set_of_books_id
  and gsob.currency_code = fc.currency_code;

x_progress := '060 -enc actions, find the right values';

for i in 1..l_dist_id.count loop
   x_progress:= '062 processing distribution :'||l_dist_id(i);
   x_progress:= '062 gl amount '||l_gl_amount(i);

   /*********************************************************************
     calculating the total amount l_amount, which is based on the qty
     ordered and the bill/delivered amount which is based on the qty
     billed or delivered.

     PART 1: total amount
   *********************************************************************/

       select pod.quantity_ordered - nvl(pod.quantity_cancelled, 0),
                  nvl(pod.rate, 1),
                  nvl(poll.price_override, 0),
                  nvl(pod.quantity_ordered, 0)
             into l_eff_quantity, l_rate, l_price, l_ordered_quantity
             from po_distributions_all pod, po_line_locations_all poll
            where pod.line_location_id = poll.line_location_id
              and pod.po_distribution_id = l_dist_id(i);

            x_progress := '080 after select before tax';
            l_tax      := po_tax_sv.get_tax('PO', l_dist_id(i));

            /*Calculate the amount in functional currency */
            l_amount := (l_tax / l_ordered_quantity) * l_rate * l_eff_quantity +
                        l_price * l_rate * l_eff_quantity;
```

```
        /*Take care of rounding */
          if (l_min_acc_unit is NULL) then
            l_amount := round(l_amount, l_precision);
          else
            l_amount := round(l_amount / l_min_acc_unit) * l_min_acc_unit;
        end if;



/*********************************************************************
  calculating the total amount l_amount, which is based on the qty
  ordered and the bill/delivered amount which is based on the qty
  billed or delivered.

  PART 2 : billed/delivered amount
*********************************************************************/

        select  (decode(nvl(poll.accrue_on_receipt_flag,'N'),
                'N',nvl(pod.quantity_billed,0),
                 greatest(nvl(pod.quantity_billed,0),
                          nvl(pod.quantity_delivered,0))))
        into l_qty_bill_del
        from po_distributions_all pod,
             po_line_locations_all poll
       where pod.line_location_id = poll.line_location_id
         and pod.po_distribution_id = l_dist_id(i);

      l_bill_del_amount := (l_tax/l_ordered_quantity)*l_rate*l_qty_bill_del +
                           l_price*l_rate*l_qty_bill_del;


      /*Take care of rounding */
        if (l_min_acc_unit is NULL) then
          l_bill_del_amount := round(l_bill_del_amount, l_precision);
        else
          l_bill_del_amount := round(l_bill_del_amount / l_min_acc_unit) * l_min_acc_unit;
    end if;

    dbms_output.put_line('dist id '||l_dist_id(i)||'gl amount '||
                         l_gl_amount(i)||' bill_del_amount '||l_bill_del_amount
                         ||' l_amount '||l_amount);
```

```
        if (l_gl_amount(i) <> 0) then

/* we are using the comparison between abs value of difference of the 2 amounts
   and the min accountable unit to see if the two amount values are equal */

            if (abs(l_gl_amount(i) - l_enc_amount(i))>nvl(l_min_acc_unit,0)) then
                dbms_output.put_line('067 dist id '||to_char(l_dist_id(i))
                                ||' amounts unequal and gl > 0');


 /*****************************************************************************
  modify:
  It is possible that we have gl amount > 0 and dist amount > 0 for a particular
  distribution, but still the encumbrance flag is N.
  This is possible in the case when after a partial recpt or billing, the
  distribution is unreserved.
  In such a case the encumbrance amount at the dist level would be equal to the
  amount billed/received and also the gl amount, but the encumb. flag would be N.

 *****************************************************************************/

                    x_progress := '090 updating amounts';
                    if ((l_bill_del_amount = l_enc_amount(i)) and (l_bill_del_amount = l_gl_amount(i)))
then
                      l_enc_flag(i) := 'N';
                    elsif (l_bill_del_amount = l_enc_amount(i)) then
                      l_enc_flag(i) := 'N';
                      l_manual_cand(i) := 'Y';
                    elsif (l_bill_del_amount = l_gl_amount(i)) then
                      l_enc_flag(i) := 'N';
                      l_enc_amount(i) := l_bill_del_amount;
                    else
                      l_enc_amount(i) := l_amount;
                      l_enc_flag(i) := 'Y';
                      /*Now compare the new amount if its not equal to gl amount
                       then mark this dist for manual posting */
                     if (abs((l_amount - l_gl_amount(i))) > nvl(l_min_acc_unit, 0)) then
                       l_manual_cand(i) := 'Y';
                     end if;
                    end if;
```

```
            else /* amounts equal and both not null */

                        if (l_bill_del_amount = l_gl_amount(i)) then
                           l_enc_flag(i) := 'N';
                        else
                           l_enc_amount(i) := l_amount;
                           l_enc_flag(i) := 'Y';
                           l_manual_cand(i) := 'Y';
                        end if;
            end if;   /* is GL amt = dist enc amount ? */

         else /* gl amount is 0 */
               dbms_output.put_line('dist id ' || l_dist_id(i) || 'gl amount 0');
               dbms_output.put_line('l_enc_amount ' || l_enc_amount(i)
                                    ||' l_bill_del_amount '||l_bill_del_amount);

                 if (l_bill_del_amount > 0) then
                   l_enc_amount(i) := l_bill_del_amount;
                   l_enc_flag(i) := 'N';
                   l_manual_cand(i) := 'Y';
                 else
                   l_enc_amount(i) := 0;
                       l_enc_flag(i) := 'N';
                 end if;
            end if; /* Is GL amount > 0 ? */

end loop;

x_progress := '130 enc actions - update po_distributions_all ';

-- updated the distributions
forall upd in 1..l_dist_id.count

   update po_distributions_all
     set encumbered_flag = l_enc_flag(upd),
         encumbered_amount = l_enc_amount(upd)
    where po_distribution_id = l_dist_id(upd);

--update the shipment encumbered_flag
UPDATE po_line_locations_all poll
    SET encumbered_flag = (SELECT Min(encumbered_flag)
                           FROM po_distributions_all
```

```
                        WHERE line_location_id = poll.line_location_id)
WHERE poll.po_header_id = pos.po_header_id;


x_progress := '140 create table ';
l_timestamp := sysdate;

x_progress := '150 get manual posting data '||
            to_char(l_manual_cand.count) ||' **';

if l_dist_id.count <> 0 then
  dbms_output.put_line('Dist is a Cand for manual Post Enc Amt is.....');
  forall k in 1..l_manual_cand.count

        INSERT into po_manual_postings_temp
             (po_distribution_id,
              budget_account_id,
              debit_amount,
              timestamp,
                    type,
                    account)
        select pod.po_distribution_id,
             pod.budget_account_id,
             nvl(pod.encumbered_amount,0) - sum(nvl(entered_dr,0) - nvl(entered_cr,0)),
             L_timestamp,
                    'PO',
                    glcck.concatenated_segments
         from po_distributions_all pod,
             gl_je_lines gl,
                    gl_je_headers glh,
             gl_code_combinations_kfv glcck
        where gl.reference_3(+) = to_char(pod.po_distribution_id)
          and gl.reference_1(+) = 'PO'
          and pod.po_distribution_id = l_dist_id(k)
              and nvl(l_manual_cand(k),'N') = 'Y'
              and gl.je_header_id = glh.je_header_id
              and glh.je_category = 'Purchases'
              and glh.actual_flag = 'E'
              and glh.je_source = 'Purchasing'
              and pod.budget_account_id = glcck.code_combination_id
          group by po_distribution_id, pod.budget_account_id,
                pod.encumbered_amount,L_timestamp,
```

```
                glcck.concatenated_segments
        having nvl(pod.encumbered_amount,0) - sum(nvl(entered_dr,0) - nvl(entered_cr,0))<> 0;
end if;

-- reserve action history stuff
-- check the action history and delete any reserve to submit actions if all the distributions
-- are now unencumbered, this should happen only if we are deleting the action history

if l_delete_act_hist = 'Y' then

  -- first get the last sequence and action code from action history
  begin
    select sequence_num, action_code
     into l_res_seq, l_res_act
         from po_action_history pah
         WHERE pah.object_id = pos.po_header_id
      AND  pah.object_type_code = decode(pos.type_lookup_code,
                                         'STANDARD','PO',
                                         'PLANNED', 'PO', --future plan to enhance for planned PO
                                         'PA')
      AND pah.object_sub_type_code = pos.type_lookup_code
          AND sequence_num in (select max(sequence_num)
                                      from po_action_history pah1
                                      where pah1.object_id = pah.object_id
                                 AND  pah1.object_type_code =pah.object_type_code
                               AND  pah1.object_sub_type_code =pah.object_sub_type_code);
  exception
  when TOO_MANY_ROWS then
   dbms_output.put_line('action history needs to be corrected separately ');
  when NO_DATA_FOUND then
   null;
  end;

  -- now if the last action is reserve get the last submit action sequence

  if (l_res_act = 'RESERVE') then
  begin
    select max(sequence_num)
     into l_sub_res_seq
    from  po_action_history pah
    where action_code = 'SUBMIT'
     and  pah.object_id = pos.po_header_id
```

```
    and  pah.object_type_code = decode(pos.type_lookup_code,
                                       'STANDARD','PO',
                                       'PLANNED', 'PO', --future plan to enhance for planned PO
                                       'PA')
    and pah.object_sub_type_code = pos.type_lookup_code;
exception
when NO_DATA_FOUND then
  null;
end;

   -- check if we need to delete the action history, ie. if all the distbributions
   -- are unreserved

   if ((l_sub_res_seq is not null ) and (l_res_seq > l_sub_res_seq)) then

        begin
     select 'Y'
       into l_del_res_hist
      from dual
      where not exists (select 'encumbered dist'
                        from po_distributions_all pod
                        where pod.po_header_id = pos.po_header_id
                          and nvl(pod.encumbered_flag,'N') = 'Y'
                          and nvl(pod.prevent_encumbrance_flag,'N')='N');
     exception
         when NO_DATA_FOUND then
           l_del_res_hist := 'N';
     end;

     if l_del_res_hist = 'Y' THEN

          dbms_output.put_line('deleting reservation action history ... ');

       delete po_action_history pah
         where pah.object_id = pos.po_header_id
           and  pah.object_type_code = decode(pos.type_lookup_code,
                                       'STANDARD','PO',
                                       'PLANNED', 'PO', --future plan to enhance for planned PO
                                       'PA')
         and pah.object_sub_type_code = pos.type_lookup_code
         and sequence_num >= l_sub_res_seq
         and sequence_num <= l_res_seq;
```

```
        end if;

      end if; -- l_res_seq > l_sub_res_seq

   end if;

end if;


/*****************************************************************************
 now check the backing req data
 We have two values(correct) of encumbered flag in the table l_enc_flag() -Y and N
 If the value is Y then req should have been unreserved. if it is N then the req
 should be reserved.
*****************************************************************************/

/*first start with the reqs that should be unreserved, ie. corresponding po dist
encumbered flag = Y */

l_manual_cand:= null;
j:=1;

begin
select prd.distribution_id,
       nvl(prd.encumbered_flag,'N'),
       nvl(prd.encumbered_amount,'0'),
       sum(nvl(gl.entered_dr,0)-nvl(gl.entered_cr,0)),
       'N' -- initialize manual postings flag
bulk collect into l_req_dist_id,l_req_enc_flag, l_req_enc_amount, l_req_gl_amount,l_manual_cand
from po_req_distributions_all prd, gl_je_lines gl,gl_je_headers glh,po_distributions_all pod
where pod.po_header_id = pos.po_header_id
  AND prd.distribution_id = Nvl(pod.req_distribution_id,-999)
  and gl.reference_1(+) = 'REQ'
  and gl.reference_3(+) = to_char(prd.distribution_id)
  AND pod.encumbered_flag = 'Y'
  AND nvl(prd.prevent_encumbrance_flag,'N') = 'N'
  and prd.ussgl_transaction_code is null
  and glh.je_header_id    = gl.je_header_id
  and glh.je_category = 'Purchases'
  and glh.je_source = 'Purchasing'
  and glh.actual_flag = 'E'
group by prd.distribution_id, prd.encumbered_flag, prd.encumbered_amount;
```

```
exception
WHEN NO_DATA_FOUND THEN
  DBMS_OUTPUT.PUT_LINE('no req to unreserve ');
WHEN OTHERS THEN
  dbms_output.put_line('exception get req '||sqlerrm||x_progress);
end;

if l_req_dist_id is not null then
  for i in 1..l_req_dist_id.count loop
  dbms_output.put_line('unres req: dist id'||l_req_dist_id(i)||
                                ' l_req_enc_amount '||l_req_enc_amount(i)||
                                ' l_req_gl_amount  '||l_req_gl_amount(i));

    if (l_req_gl_amount(i) <> 0)  then
      l_manual_cand(i):= 'Y';
    end if;
    if l_req_enc_amount(i) <> 0 then
      l_req_enc_amount(i) := 0;
          l_req_enc_flag(i):='N';
    end if;
    if l_req_enc_flag(i) <> 'N' then
      l_req_enc_flag(i):='N';
    end if;
  end loop;

  forall k in 1..l_req_dist_id.count
    update po_req_distributions_all
      set encumbered_amount = l_req_enc_amount(k),
            encumbered_flag = l_req_enc_flag(k)
     where distribution_id = l_req_dist_id(k);

  if l_manual_cand is not null then

  forall k in 1..l_manual_cand.count
    insert into po_manual_postings_temp
              (po_distribution_id,
               budget_account_id,
               debit_amount,
               timestamp,
                     type,
                     account)
```

```
          select prd.distribution_id,
            prd.budget_account_id,
            nvl(prd.encumbered_amount,0) - sum(nvl(entered_dr,0) - nvl(entered_cr,0)),
            L_timestamp,
                  'REQ',
                  glcck.concatenated_segments
        from po_req_distributions_all prd,
            gl_je_lines gl,
                  gl_je_headers glh,
            gl_code_combinations_kfv glcck
       where gl.reference_3(+) = to_char(prd.distribution_id)
         and gl.reference_1(+) = 'REQ'
         and prd.distribution_id = l_req_dist_id(k)
             and nvl(l_manual_cand(k),'N') = 'Y'
             and prd.budget_account_id = glcck.code_combination_id
             and glh.je_header_id    = gl.je_header_id
             and glh.je_category = 'Purchases'
             and glh.je_source = 'Purchasing'
             and glh.actual_flag = 'E'
         group by distribution_id, prd.budget_account_id,
                prd.encumbered_amount,L_timestamp,
                glcck.concatenated_segments;
  end if;


end if;


/*******************************************************************************************
  Now we have set those req distributions right, which shoul have been unreserved but
  were not due to some error - Now check for those requisitions that should be reserved
*******************************************************************************************/
--cleanup the tables used
l_req_dist_id    := NULL;
l_req_enc_flag   := NULL;
l_req_enc_amount := NULL;
l_req_gl_amount  := NULL;

l_manual_cand:= NULL;
j:=1;

begin
```

```
            select prd.distribution_id,
                   nvl(prd.encumbered_flag,'N'),
                   nvl(prd.encumbered_amount,'0'),
                   sum(nvl(gl.entered_dr,0)-nvl(gl.entered_cr,0)),
                   'N'
            bulk collect
              into l_req_dist_id,
                   l_req_enc_flag,
                   l_req_enc_amount,
                   l_req_gl_amount,
                   l_manual_cand
              from po_req_distributions_all prd,
                   gl_je_lines gl,
                   gl_je_headers glh,
                   po_distributions_all pod
             where pod.po_header_id = pos.po_header_id
               and prd.distribution_id = Nvl(pod.req_distribution_id,-999)
               and gl.reference_1(+) = 'REQ'
               and gl.reference_3(+) = to_char(prd.distribution_id)
               and pod.encumbered_flag = 'N'
               and nvl(prd.prevent_encumbrance_flag,'N') = 'N'
               and prd.ussgl_transaction_code is null
               and glh.je_header_id    = gl.je_header_id
               and glh.je_category = 'Purchases'
               and glh.je_source = 'Purchasing'
               and glh.actual_flag = 'E'
            group by prd.distribution_id, prd.encumbered_flag, prd.encumbered_amount;
exception
when no_data_found then
  dbms_output.put_line('no reqs to do reservation check');
end;

if (l_req_dist_id.count <> 0) then

    begin
      select   prd.distribution_id,
               (decode(nvl(poll.accrue_on_receipt_flag,'N'),
                  'N',nvl(pod.quantity_billed,0),
                  greatest(nvl(pod.quantity_billed,0),
                           nvl(pod.quantity_delivered,0)))) q,
               nvl(pod.rate, 1),
               nvl(poll.price_override, 0),
```

```
            prl.unit_price,
            nvl(prd.req_line_quantity, 0)
        bulk collect into
         l_req_dist_id        ,
         l_req_qty_bill_del   ,
         l_rate_table         ,
         l_req_price_table    ,
         l_price_table        ,
         l_qty_ordered_table
         from po_req_distributions_all prd,
             po_distributions_all pod,
             po_line_locations_all poll,
             po_requisition_lines_all prl
        where prd.requisition_line_id = prl.requisition_line_id
          and pod.encumbered_flag = 'N'
          and nvl(prd.prevent_encumbrance_flag, 'N') = 'N'
          and pod.po_header_id = pos.po_header_id
          and pod.req_distribution_id = prd.distribution_id
          and pod.line_location_id = poll.line_location_id
          order by prd.distribution_id;

    exception
      when no_data_found then
        dbms_output.put_line('no reqs to do reservation check');
    end;

end if;

if (l_req_dist_id.count <> 0) then
    dbms_output.put_line('dist count after 2nd sql :'||l_req_dist_id.count);
    for i in 1 .. l_req_dist_id.count loop

      l_tax     :=  po_tax_sv.get_tax('REQ',l_req_dist_id(i));
      l_amount  :=  l_qty_ordered_table(i)*l_req_price_table(i) + l_tax;

/* we would not take PO tax into consideration when calculating the req
   encumbrance amount since that is added to the PO enc amount and should not
   be included in the req reserved amount calculation */

   -- logging start
   dbms_output.put_line('reserve req id '||l_req_dist_id(i)||
                        ' l_req_enc_amount '||l_req_enc_amount(i)||
```

```
                                ' l_req_gl_amount '||l_req_gl_amount(i)||
                                ' req line qty '||l_qty_ordered_table(i)||
                        ' qty billed delivered on PO '|| l_req_qty_bill_del(i)||
                        ' req price '||l_req_price_table(i));

  -- logging end;
    l_remainder_qty       := (l_qty_ordered_table(i)-l_req_qty_bill_del(i));

    l_req_bill_del_amount := l_remainder_qty *l_req_price_table(i)
                                  + (l_remainder_qty/l_qty_ordered_table(i))*l_tax;

  -- logging start
  dbms_output.put_line('bck req id :'||l_req_dist_id(i) || 'remainder qty :'||
                          l_remainder_qty||' enc amount :'||l_req_bill_del_amount);
  -- logging end

    /* take care of rounding */

     if (l_min_acc_unit is NULL) then
                l_amount := round(l_amount, l_precision);
                l_req_bill_del_amount := round(l_req_bill_del_amount, l_precision);
     else
                l_amount := round(l_amount / l_min_acc_unit) * l_min_acc_unit;
                l_req_bill_del_amount:= round (l_req_bill_del_amount/l_min_acc_unit) *
                                                                  l_min_acc_unit;
     end if;

--logging start
dbms_output.put_line('bck req id :'||l_req_dist_id(i) || 'amount :'||
                          l_amount||' enc amount :'||l_req_bill_del_amount);
-- logging end
dbms_output.put_line('before enc flag comparison count'||l_req_enc_flag.count);
    if l_req_enc_flag(i) <> 'Y' then
       l_req_enc_flag(i) := 'Y';
    end if;

    --l_req_bill_del_amount = 0 is not possible here since this value will be
    -- only if the entire qty is rcd/billed in which case the PO cannot be unreserved.

          dbms_output.put_line('before enc amount comparison count'||l_req_enc_amount.count);

            if (l_req_enc_amount(i) <> l_req_bill_del_amount) then
```

```
            l_req_enc_amount(i) := l_req_bill_del_amount;

          -- logging start
                dbms_output.put_line('assigned bck req id :'||l_req_dist_id(i) ||
                                ' enc amount :'||  l_req_enc_amount(i));
          -- logging end

         end if;

            dbms_output.put_line('before gl amount comparison count'||l_req_gl_amount.count);
         if (l_req_gl_amount(i) <> l_req_bill_del_amount) then
            l_manual_cand(i) := 'Y';
        end if;

end loop;

  forall k in 1..l_req_dist_id.count
    update po_req_distributions_all
      set encumbered_amount = l_req_enc_amount(k),
            encumbered_flag = l_req_enc_flag(k)
     where distribution_id = l_req_dist_id(k);

 if l_manual_cand is not null then

 forall k in 1..l_manual_cand.count
   insert into po_manual_postings_temp
             (po_distribution_id,
              budget_account_id,
              debit_amount,
              timestamp,
                  type,
                  account)
        select prd.distribution_id,
           prd.budget_account_id,
           nvl(prd.encumbered_amount,0) - sum(nvl(entered_dr,0) - nvl(entered_cr,0)),
           L_timestamp,
                'REQ',
                glcck.concatenated_segments
       from po_req_distributions_all prd,
           gl_je_lines gl,
                gl_je_headers glh,
           gl_code_combinations_kfv glcck
```

```
        where gl.reference_3(+) = to_char(prd.distribution_id)
          and gl.reference_1(+) = 'REQ'
          and prd.distribution_id = l_req_dist_id(k)
              and nvl(l_manual_cand(k),'N') = 'Y'
              and prd.budget_account_id = glcck.code_combination_id
              and glh.je_header_id    = gl.je_header_id
              and glh.je_category = 'Purchases'
              and glh.je_source = 'Purchasing'
              and glh.actual_flag = 'E'
          group by distribution_id, prd.budget_account_id,
                   prd.encumbered_amount,L_timestamp,
                   glcck.concatenated_segments;

   end if;
 end if;


  dbms_output.put_line('encumbrance actions completed, Please commit if no Issue found');


EXCEPTION
WHEN OTHERS THEN
  dbms_output.put_line('some exception occured '||sqlerrm||' rolling back'||x_progress);
  rollback;
END;
/
```

Issue: At the time of closing DAK, it ask for MRR

Solution:

```
select *--upper(nvl(attribute2,'Y')) into v_check
    from au_erp_fnd_flex_values_vl
    where flex_value_set_id=1009968
    and flex_value_meaning='690' for update
```

Issue: This query is used to reopen the DAY Close.

Solution:

```
select *--upper(nvl(attribute2,'Y')) into v_check
from au_erp_fnd_flex_values_vl
     where flex_value_set_id=1009968
     and flex_value_meaning='690' for update--
to_char(&AU_PRINT_ORDER.AU_PRINT_ORDER_DAK);
```

ORDER_PROCESS_TRANS_DAY_CLOSE

```
select * from AU_ORDER_PROCESS_TRANs
where AU_ORD_AR_INT_DAK='690'

update AU_ORDER_PROCESS_TRANs
set au_ord_ar_day_close='N'
where AU_ORD_AR_INT_DAK='690'
AND AU_ORD_AR_INT_PUB_DATE='03-FEB-2015'
```