

```

1  select * from dba_objects
2  where object_name like 'PO_HEADERS%'
3  AND (OBJECT_TYPE LIKE 'T%' OR OBJECT_TYPE LIKE 'SY%')
4  AND OWNER IN ('APPS', 'PO')
5  AND OBJECT_ID IN (41472,1945578);
6
7  apps.fnd_program.add_to_group (program_short_name => l_program_short_name,
8                                program_application =>
8                                l_program_application,
9
9                                request_group      =>
9                                l_request_group,
10
10                               group_application   =>
10                               l_group_application
11
11                               );
12
13  TRUNC((rq.actual_completion_date - rq.actual_start_date) * 24)
14  || ' Hour '
15  || MOD(
16      TRUNC(
17          (rq.actual_completion_date - rq.actual_start_date) * 1440),
18      60)
19  || ' Min '
20  || MOD(
21      TRUNC(
22          (rq.actual_completion_date - rq.actual_start_date) * 86400),
23      60)
24  || ' Sec'
25  "Time Taken"
26  SELECT SYS_CONTEXT ('USERENV', 'DB_NAME')
27
28
29  SELECT SYS_CONTEXT ('USERENV', 'DB_NAME')
30  db_name INTO gc_database FROM DUAL
31
32
33  /*
34  fnd_user                      -----> user information
35  PER_ALL_PEOPLE_F              -----> connected user information
36  FND_USER_RESP_GROUPS_direct  -----> contains responsibility details
37  FND_RESPONSIBILITY            -----> contains responsibility id, key (table)
38  FND_RESPONSIBILITY_VL        -----> contains applications id, responsibility id, key and
39  name (view)
39  FND_APPLICATION_VL           -----> contains applications id, short name, name and
40  product code (view)
40  fnd_user_pkg.createuser
41
42  */
43
44  ;
45
46  select *
47  from per_all_people_f;
48
49  select user_name, user_id
50  from fnd_user;
51
52  select user_name, user_id, start_date, end_date, employee_id
53  from fnd_user
54  where lower(user_name) = 'operations';
55
56  select employee_number, npw_number, current_employee_flag, current_npw_flag, person_id,
57  effective_start_date, effective_end_date, start_date, BUSINESS_GROUP_ID, first_name,
58  last_name, full_name
59  from per_all_people_f;
58
59  select fu.user_name,

```

```

60         fu.user_id,
61         fu.start_date,
62         fu.end_date,
63         fu.employee_id,
64         papf.employee_number,
65         papf.npw_number,
66         papf.current_employee_flag,
67         papf.current_npw_flag,
68         papf.person_id,
69         papf.effective_start_date,
70         papf.effective_end_date,
71         papf.start_date,
72         papf.BUSINESS_GROUP_ID,
73         papf.first_name,
74         papf.last_name,
75         papf.full_name
76 from per_all_people_f papf,
77      fnd_user fu
78 where 1=1
79        and fu.employee_id = &id
80        and fu.employee_id = PAPF.person_id;
81
82
83
84 select *
85 from fnd_users;fnd_program.delete_program
86 fnd_program.delete_executable
87 fnd_program.program_exists
88 fnd_program.executable_exists
89 apps.fnd_program.remove_from_group
90 ;
91
92 -- delete program
93 begin
94
95     apps.fnd_program.delete_program( 'XXQGEN_TEST_DK', 'PO');
96     DBMS_OUTPUT.PUT_LINE('Program successfully deleted.');

```

```

129         dbms_output.put_line('error in remove the group : ' || sqlcode || ' - ' ||
130                                sqlerrm);
131     end;
132     /
133
134     -- done
135
136     BEGIN
137         -- Output the result of the program_exists function
138         DBMS_OUTPUT.PUT_LINE(
139             'Program Exists: ' ||
140             CASE
141                 WHEN apps.fnd_program.program_exists(
142                     program      => 'XXQGEN_TEST_DK',
143                     application => 'PO' -- 'Purchasing' maps to 'PO' in short name
144                 ) THEN 'Yes'
145             ELSE 'No'
146             END
147         );
148     END;
149     /
150
151     -- done
152
153
154     BEGIN
155         -- Output the result of the program_exists function
156         DBMS_OUTPUT.PUT_LINE(
157             'Executable Exists: ' ||
158             CASE
159                 WHEN apps.fnd_program.executable_exists(
160                     executable_short_name => 'XXQGEN_TEST_DK',
161                     application => 'PO' -- 'Purchasing' maps to 'PO' in short name
162                 ) THEN 'Yes'
163             ELSE 'No'
164             END
165         );
166     END;
167     /
168
169
170
171     SELECT DISTINCT UPCVL.CONCURRENT_PROGRAM_ID,
172                    UPCVL.CONCURRENT_PROGRAM_NAME,
173                    UPCVL.EXECUTABLE_APPLICATION_ID,
174                    UPCVL.EXECUTABLE_ID,
175                    FEFV.EXECUTION_FILE_NAME,
176                    FEFV.DESCRPTION,
177                    UPCVL.APPLICATION_ID,
178                    FAVL.APPLICATION_NAME,
179                    FAVL.APPLICATION_SHORT_NAME ,
180                    FCR.REQUEST_ID,
181                    SC.MEANING STATUS_CODE,
182                    PC.MEANING PHASE_CODE,
183                    FCR.ACTUAL_COMPLETION_DATE - FCR.ACTUAL_START_DATE
184     FROM      FND_CONCURRENT_PROGRAMS_VL UPCVL,
185              FND_APPLICATION_VL FAVL,
186              FND_EXECUTABLES_FORM_V FEFV,
187              FND_CONCURRENT_REQUESTS FCR,
188              (SELECT LOOKUP_CODE, MEANING
189               FROM FND_LOOKUP_VALUES
190               WHERE LOOKUP_TYPE = 'CP_STATUS_CODE' AND LANGUAGE = 'US'
191               AND ENABLED_FLAG = 'Y') SC,
192              (SELECT LOOKUP_CODE, MEANING
193               FROM FND_LOOKUP_VALUES
194               WHERE LOOKUP_TYPE = 'CP_PHASE_CODE' AND LANGUAGE = 'US'
195               AND ENABLED_FLAG = 'Y') PC
196     WHERE 1=1

```

```

197 AND      UPCVL.CONCURRENT_PROGRAM_NAME = 'XXQGEN_USER_DATA_PGK_DK'
198 AND      UPCVL.CONCURRENT_PROGRAM_ID = FCR.CONCURRENT_PROGRAM_ID
199 AND      UPCVL.APPLICATION_ID = FAVL.APPLICATION_ID
200 AND      UPCVL.EXECUTABLE_ID = FEFV.EXECUTABLE_ID
201 AND      SC.LOOKUP_CODE = FCR.STATUS_CODE
202 AND      PC.LOOKUP_CODE = FCR.PHASE_CODE;select prha.requisition_header_id,
203 PRLA.LINE_NUM,
204 PRLA.PURCHASE_BASIS,
205     prha.preparer_id,
206     PAPFPRE.FULL_NAME,
207     lub.user_name LAST_UPDATED_BY,
208     cb.user_name CREATED_BY,
209     PRHA.ORG_ID,
210     HOU.NAME ORGANIZATION_NAME,
211     PRHA.TYPE_LOOKUP_CODE,
212     PRHA.AUTHORIZATION_STATUS,
213     PLC1.DESCRPTION,
214     PLC1.DISPLAYED_FIELD,
215     PRHA.DESCRPTION,
216     PRHA.ENABLED_FLAG,
217     PRHA.SUMMARY_FLAG,
218     PRHA.SEGMENT1,
219     PRLA.REQUISITION_LINE_ID,
220     PRLA.CATEGORY_ID,
221     MC.SEGMENT1 || ' ' || MC.SEGMENT2 CATEGORY_NAME,
222     PRLA.QUANTITY,
223     PRLA.UNIT_PRICE,
224     PRLA.QUANTITY * PRLA.UNIT_PRICE AMOUNT,
225     PRLA.TO_PERSON_ID,
226     PAPF.FULL_NAME,
227     MSIB.SEGMENT1 || ' ' || MSIB.DESCRPTION ITEM_ID,
228     PRLA.ITEM_DESCRIPTION,
229     PRLA.NEED_BY_DATE,
230     PRLA.DELIVER_TO_LOCATION_ID,
231     HLA.ADDRESS_LINE_1,
232     HLA.DESCRPTION,
233     HLA.POSTAL_CODE,
234     PRLA.LINE_NUM,
235     PRDA.DISTRIBUTION_ID,
236     PRDA.REQUISITION_LINE_ID,
237     PRDA.SET_OF_BOOKS_ID,
238     PRDA.DISTRIBUTION_NUM,
239     PRLA.REQUEST_ID,
240     gcc.ACCOUNT_TYPE,
241     gcc.ENABLED_FLAG,
242     gcc.SUMMARY_FLAG,
243     gcc.COMPANY_COST_CENTER_ORG_ID
244 from
245     po_requisition_headers_all prha,
246     po_requisition_lines_all prla,
247     po_req_distributions_all prda,
248     PER_ALL_PEOPLE_F PAPFPRE,
249     fnd_user lub,
250     fnd_user cb,
251     HR_LOCATIONS_ALL HLA,
252     PER_ALL_PEOPLE_F PAPF,
253     HR_ORGANIZATION_UNITS HOU,
254     MTL_CATEGORIES MC,
255     MTL_SYSTEM_ITEMS_B MSIB,
256     ORG_ORGANIZATION_DEFINITIONS OOD,
257     GL_CODE_COMBINATIONS gcc,
258     PO_LOOKUP_CODES PLC1
259 --     PER_ALL_PEOPLE_F PAPFREQ
260 where
261     1=1
262 and     prha.requisition_header_id = prla.requisition_header_id
263 and     prha.segment1 = '14438'
264 AND     PAPFPRE.PERSON_ID = PRHA.PREPARER_ID
265 and     prla.last_updated_by = lub.USER_id

```

```

266 and prla.created_by = cb.USER_id
267 and prda.requisition_line_id = prla.requisition_line_id
268 AND PRLA.TO_PERSON_ID = PAPF.PERSON_ID
269 AND HLA.LOCATION_ID = PRLA.DELIVER_TO_LOCATION_ID
270 AND HOU.ORGANIZATION_ID = PRHA.ORG_ID
271 AND PRHA.AUTHORIZATION_STATUS = PLC1.LOOKUP_CODE
272 AND PLC1.LOOKUP_TYPE = 'AUTHORIZATION STATUS'
273 AND PRLA.CATEGORY_ID = MC.CATEGORY_ID
274 --AND HLA.PERSON_ID = PRLA.DELIVER_TO_LOCATION_ID
275 AND MSIB.INVENTORY_ITEM_ID = PRLA.ITEM_ID
276 AND MSIB.ORGANIZATION_ID = OOD.ORGANIZATION_ID
277 AND OOD.ORGANIZATION_ID = PRLA.ORG_ID
278 and prda.code_combination_id = gcc.code_combination_id;
279 ;
280
281
282
283 SELECT *
284 FROM PO_LOOKUP_CODES
285 WHERE LOOKUP_CODE = 'INCOMPLETE'
286 AND LOOKUP_TYPE = 'AUTHORIZATION STATUS'
287 ;
288
289 select *
290 from dba_objects
291 where object_name like 'HR%LOC%';
292
293 SELECT *
294 FROM HR_LOCATIONS_ALL
295 WHERE LOCATION_ID = 204;
296
297 select *
298 from po_req_distributions_all;
299
300 select *
301 from gl_code_combinations;
302
303 select *
304 from po_req_distributions_all prda,
305      gl_code_combinations gcc
306 where prda.CODE_COMBINATION_ID = gcc.code_combination_id;
307
308 SELECT
309
310      --prda.deliver_to_location_id,
311      --prda.quantity,
312      --prda.amount,
313      --prda.gl_encumbrance_id,
314      --prda.charge_account_id,
315      prda.budget_account_id,
316      --prda.status_code,
317      gcc.code_combination_id,
318      gcc.segment1 AS company_code,
319      gcc.segment2 AS cost_center,
320      gcc.segment3 AS account,
321      gcc.segment4 AS sub_account,
322      gcc.segment5 AS product_code,
323      gcc.description AS gl_account_description
324 FROM
325      po_req_distributions_all prda,
326      gl_code_combinations gcc
327 WHERE
328      prda.code_combination_id = gcc.code_combination_id;
329
330 SELECT *
331 FROM ORG_ORGANIZATION_DEFINITIONS;
332 ORG_ORGANISATION_DEFINITIONS
333
334

```

```

335  ORG_ORGANIZATION_DEFINITIONS
336  ORG_ORGANIZATION_DEFINITIONS
337
338  SELECT *
339  FROM TABS
340  WHERE TABLE_NAME LIKE '%ORG_%_OR%_D';
341
342  SELECT *
343  FROM MTL_SYSTEM_ITEMS_B;
344  select *
345  from
346      po_requisition_headers_all prha,
347      po_requisition_lines_all prla,
348      po_req_distributions_all prda
349  where
350      1=1
351  and   prha.requisition_header_id = prla.requisition_header_id
352  and   prha.segment1 = '14438'
353  and   prda.requisition_line_id = prla.requisition_line_id;
354
355  select *
356  from po_requisition_headers_all;
357
358  SELECT *
359  FROM MTL_CATEGORIES
360  WHERE CATEGORY_ID = 1;
361
362  select *
363  from po_requisition_lines_all;
364
365  select *
366  from po_req_distributions_all;
367
368  select *
369  from fnd_user
370  where user_id = 1318;
371
372  select *
373  from ap_suppliers
374  where vendor_id = 30163;
375
376  select org_id, hou.name
377  from ap_supplier_sites_all assa,
378       hr_operating_units hou
379  where vendor_id = 30163
380  and    HOU.ORGANIZATION_ID = assa.org_id;
381
382  SELECT *
383  FROM MTL_SYSTEM_ITEMS_B;
384
385  GL_CODE_COMBINATIONS
386
387  hr_operating_units
388
389  select *
390  from ap_supplier_sites_all;
391
392
393
394  select assa.vendor_id, assa.vendor_site_code, ass.vendor_name ,count(assa.vendor_id)
395  from ap_supplier_sites_all assa, ap_suppliers ass
396  where
397      assa.vendor_id = ass.vendor_id
398  group by assa.vendor_id, assa.vendor_site_code, ass.vendor_name
399  having count(assa.vendor_id) >1;
400
401  select * from po_requisition_lines_all;
402
403  --question : vendor all data from ap_supplier_sites_all, ap_supplier without using group

```

```

403 by
404 -- HOW MANY PARAMETER CAN BE ENTER IN CONCURRENT PROGRAM
405 -- FETCH PARAMETER WHICH IS CONNECTED WITH CONCURRENT PROGRAM FROM THE SQL QUERRY
406 -- PRINT PATAMETER WHICH IS USED IN CONCUREENT PROGRAM
407
408 SELECT
409 REQUISITION_HEADER_IDLINE_NUMLINE_TYPE IDCATEGORY_IDITEM_DESCRIPTIONUNIT_MEAS_LOOKUP_CODE
410 UNIT_PRICEQUANTITYDELIVER_TO_LOCATION_IDTO_PERSON_IDLAST_UPDATED_BYCREATED_BYORG_IDREQUIS
411 ITION_HEADER_IDREQUISITION_LINE_ID
412 FROM po_requisition_lines_all;
413
414 select prha.requisition_header_id
415 ,prla.line_num
416 ,prla.requisition_line_id
417 ,mc.category_id
418 ,papf.full_name
419 ,mc.segment2
420 ,prla.unit_meas_lookup_code
421 ,prla.unit_price
422 ,prla.quantity
423 ,prla.unit_price*prla.quantity Amount
424 ,prla.need_by_date
425 ,msib.segment1
426 ,msib.description
427 ,ood.organization_code
428 ,ood.organization_name
429 ,prla.to_person_id
430 ,hl.location_code
431 ,plt.line_type
432 from po_requisition_lines_all prla
433 ,po_requisition_headers_all prha
434 ,mtl_categories mc
435 ,per_all_people_f papf
436 ,mtl_system_items_b msib
437 ,org_organization_definitions ood
438 ,hr_locations hl
439 ,po_line_types plt
440 where prla.requisition_header_id=prha.requisition_header_id
441 and mc.category_id=prla.category_id
442 and msib.inventory_item_id=prla.item_id
443 and prla.to_person_id=papf.person_id
444 and ood.organization_id=msib.organization_id
445 and ood.organization_id = prla.org_id
446 and prla.deliver_to_location_id=hl.location_id
447 and prla.line_type_id=plt.line_type_id
448 and prha.segment1 = '14438'
449 and PRLA. -- ADD EXIST CODE
450 ;
451
452 DESC FND_USER;
453 PER_ALL_PEOPLE_F
454 FND_FLEX_VALUE_SET
455
456 SELECT *
457 FROM MTL_SYSTEM_ITEMS_B
458 WHERE segment1 = 'P2P Cycle';
459
460 SELECT *
461 FROM PO_REQUISITION_LINES_ALL
462 WHERE ITEM_ID = 210959;
463
464 SELECT
465 PRD.REQ_LINE_QUANTITY
466 ,GCC.CHART_OF_ACCOUNTS_ID
467
468 ,GCC.SEGMENT1 || '-' || GCC.SEGMENT2 || '-' || GCC.SEGMENT3 || '-' || GCC.SEGMENT4 || '-' || GCC.SEGM

```

```

ENT5 ACCOUNT_DETAIL
468 FROM
469 PO_REQ_DISTRIBUTIONS_ALL PRD
470 ,GL_CODE_COMBINATIONS GCC
471 ,PO_REQUISITION_LINES_ALL PRL
472 ,PO_REQUISITION_HEADERS_ALL PRH
473 WHERE
474 PRD.CODE_COMBINATION_ID=GCC.CODE_COMBINATION_ID
475 AND PRL.REQUISITION_LINE_ID=PRD.REQUISITION_LINE_ID
476 AND PRL.REQUISITION_HEADER_ID=PRH.REQUISITION_HEADER_ID
477 AND PRH.SEGMENT1='14438';
478
479 select prha.requisition_header_id,
480        prha.preparer_id,
481        PAPFPRE.FULL_NAME PREPARER_FULL_NAME,
482        lub.user_name LAST_UPDATED_BY,
483        cb.user_name CREATED_BY,
484        PRHA.ORG_ID,
485        HOU.NAME ORGANIZATION_NAME,
486        PRHA.TYPE LOOKUP_CODE,
487        PRHA.AUTHORIZATION_STATUS AUTH_STATUS,
488        PLC1.DESCRPTION,
489        PLC1.DISPLAYED_FIELD,
490        PRHA.DESCRPTION HEADER_DESC,
491        PRHA.ENABLED_FLAG,
492        PRHA.SUMMARY_FLAG,
493        PRHA.SEGMENT1,
494        PRLA.REQUISITION_LINE_ID,
495        PRLA.CATEGORY_ID,
496        PRLA.PURCHASE_BASIS,
497        MC.SEGMENT1 || ' ' || MC.SEGMENT2 CATEGORY_NAME,
498        PRLA.QUANTITY,
499        PRLA.UNIT_PRICE,
500        PRLA.QUANTITY * PRLA.UNIT_PRICE AMOUNT,
501        PRLA.TO_PERSON_ID,
502        PAPF.FULL_NAME REQ_PERSON,
503        MSIB.SEGMENT1 || ' ' || MSIB.DESCRPTION ITEM_ID,
504        PRLA.ITEM_DESCRIPTION,
505        PRLA.DESTINATION_TYPE_CODE,
506        PRLA.NEED_BY_DATE,
507        PRLA.DELIVER_TO_LOCATION_ID,
508        HLA.ADDRESS_LINE_1,
509        HLA.DESCRPTION LOC_DESC,
510        HLA.POSTAL_CODE,
511        PRLA.LINE_NUM,
512        PRDA.DISTRIBUTION_ID,
513        PRDA.REQUISITION_LINE_ID,
514        PRDA.SET_OF_BOOKS_ID,
515        PRDA.DISTRIBUTION_NUM,
516        PRLA.REQUEST_ID,
517        gcc.ACCOUNT_TYPE,
518        gcc.ENABLED_FLAG,
519        gcc.SUMMARY_FLAG,
520        gcc.COMPANY_COST_CENTER_ORG_ID,
521
522        GCC.SEGMENT1||'-'||GCC.SEGMENT2||'-'||GCC.SEGMENT3||'-'||GCC.SEGMENT4||'-'||GCC.
523        SEGMENT5 ACCOUNT_DETAIL
524 from
525 po_requisition_headers_all prha,
526 po_requisition_lines_all prla,
527 po_req_distributions_all prda,
528 PER_ALL_PEOPLE_F PAPFPRE,
529 fnd_user lub,
530 fnd_user cb,
531 HR_LOCATIONS_ALL HLA,
532 PER_ALL_PEOPLE_F PAPF,
533 HR_ORGANIZATION_UNITS HOU,
534 MTL_CATEGORIES MC,
535 MTL_SYSTEM_ITEMS_B MSIB,

```



```

534         ORG_ORGANIZATION_DEFINITIONS OOD,
535         GL_CODE_COMBINATIONS gcc,
536         PO_LOOKUP_CODES PLC1
537     --     PER_ALL_PEOPLE_F PAPFREQ
538 where
539     1=1
540 and     prha.requisition_header_id = prla.requisition_header_id
541 and     prha.segment1 = '14438'
542 AND     PAPFPRE.PERSON_ID = PRHA.PREPARER_ID
543 and     prla.last_updated_by = lub.USER_id
544 and     prla.created_by = cb.USER_id
545 and     prda.requisition_line_id = prla.requisition_line_id
546 AND     PRLA.TO_PERSON_ID = PAPF.PERSON_ID
547 AND     HLA.LOCATION_ID = PRLA.DELIVER_TO_LOCATION_ID
548 AND     HOU.ORGANIZATION_ID = PRHA.ORG_ID
549 AND     PRHA.AUTHORIZATION_STATUS = PLC1.LOOKUP_CODE
550 AND     PLC1.LOOKUP_TYPE = 'AUTHORIZATION STATUS'
551 AND     PRLA.CATEGORY_ID = MC.CATEGORY_ID
552 --AND     HLA.PERSON_ID = PRLA.DELIVER_TO_LOCATION_ID
553 AND     MSIB.INVENTORY_ITEM_ID = PRLA.ITEM_ID
554 AND     MSIB.ORGANIZATION_ID = OOD.ORGANIZATION_ID
555 AND     OOD.ORGANIZATION_ID = PRLA.ORG_ID
556 and     prda.code_combination_id = gcc.code_combination_id;CREATE OR REPLACE PACKAGE BODY
xxqgen_api_pkg_dk AS
557
558
559     /*****
560     * Procedure: CREATE_USER
561     * Description: Creates a new user using FND_USER_PKG API.
562     *****/
563
564     PROCEDURE create_user (
565         p_user_name          IN VARCHAR2,
566         p_unencrypted_password IN VARCHAR2
567     ) IS
568 BEGIN
569     fnd_user_pkg.createuser(
570         x_user_name          => p_user_name,
571         x_owner              => NULL,
572         x_unencrypted_password => p_unencrypted_password,
573         x_session_number    => NULL,
574         x_start_date        => NULL,
575         x_end_date          => NULL,
576         x_last_logon_date   => NULL,
577         x_description       => NULL,
578         x_password_date     => NULL,
579         x_password_accesses_left => NULL,
580         x_password_lifespan_accesses => NULL,
581         x_password_lifespan_days => NULL,
582         x_employee_id       => NULL,
583         x_email_address     => NULL,
584         x_fax               => NULL,
585         x_customer_id       => NULL,
586         x_supplier_id       => NULL
587     );
588
589     -- Debug message
590     DBMS_OUTPUT.PUT_LINE('User created successfully.');
```

```

596      * Description: Adds a program to a request group.
597
598      ****
599      ****/
600  PROCEDURE add_to_group (
601      p_program_short_name  IN VARCHAR2,
602      p_program_application IN VARCHAR2,
603      p_request_group       IN VARCHAR2,
604      p_group_application   IN VARCHAR2
605  ) IS
606  BEGIN
607      apps.fnd_program.add_to_group(
608          program_short_name => p_program_short_name,
609          program_application => p_program_application,
610          request_group       => p_request_group,
611          group_application   => p_group_application
612      );
613      DBMS_OUTPUT.PUT_LINE('Program added to group successfully.');
```

```

614      COMMIT;
615  EXCEPTION
616      WHEN OTHERS THEN
617          DBMS_OUTPUT.PUT_LINE('Error adding program to group: ' || SQLCODE || ' - '
618              || SQLERRM);
619  END add_to_group;
620
621  /*****
622  ****
623  ****/
624  * Procedure: SUBMIT_REQUEST
625  * Description: Submits a concurrent request for a specific program.
626
627  ****
628  ****/
629  PROCEDURE submit_request (
630      p_application IN VARCHAR2,
631      p_program     IN VARCHAR2,
632      p_user_id     IN NUMBER,
633      p_resp_id     IN NUMBER,
634      p_resp_appl_id IN NUMBER
635  ) IS
636      l_request_id NUMBER;
637  BEGIN
638      apps.fnd_global.apps_initialize(p_user_id, p_resp_id, p_resp_appl_id);
639
640      l_request_id := fnd_request.submit_request(
641          application => p_application,
642          program     => p_program
643      );
644
645      IF l_request_id = 0 THEN
646          DBMS_OUTPUT.PUT_LINE('Request submission failed.');
```

```

647      ELSE
648          DBMS_OUTPUT.PUT_LINE('Request submitted successfully. Request ID: ' ||
649              l_request_id);
650      END IF;
651  EXCEPTION
652      WHEN OTHERS THEN
653          DBMS_OUTPUT.PUT_LINE('Error submitting request: ' || SQLCODE || ' - ' ||
654              SQLERRM);
655  END submit_request;
656
657  /*****
658  ****
659  ****/
660  * Procedure: EXECUTABLE_EXISTS
661  * Description: Checks if an executable exists.
662
663  ****
664  ****/

```

```

652 PROCEDURE executable_exist (
653     p_executable_short_name IN VARCHAR2,
654     p_application           IN VARCHAR2
655 ) IS
656 BEGIN
657     IF apps.fnd_program.executable_exists(
658         executable_short_name => p_executable_short_name,
659         application           => p_application
660     ) THEN
661         DBMS_OUTPUT.PUT_LINE('Executable exists.');
```

662 ELSE

663 DBMS_OUTPUT.PUT_LINE('Executable does not exist.');

664 END IF;

665 EXCEPTION

666 WHEN OTHERS THEN

667 DBMS_OUTPUT.PUT_LINE('Error checking executable existence: ' || SQLCODE || ' - ' || SQLERRM);

668 END executable_exist;

669

670

671

672

673

```

/*****
*****
* Procedure: PROGRAM_EXIST
* Description: Checks if a program exists.
*****
*****/
674 PROCEDURE program_exist (
675     p_program IN VARCHAR2,
676     p_application IN VARCHAR2
677 ) IS
678 BEGIN
679     IF apps.fnd_program.program_exists(
680         program      => p_program,
681         application => p_application
682     ) THEN
683         DBMS_OUTPUT.PUT_LINE('Program exists.');
```

684 ELSE

685 DBMS_OUTPUT.PUT_LINE('Program does not exist.');

686 END IF;

687 EXCEPTION

688 WHEN OTHERS THEN

689 DBMS_OUTPUT.PUT_LINE('Error checking program existence: ' || SQLCODE || ' - ' || SQLERRM);

690 END program_exist;

691

692

693

694

695

```

/*****
*****
* Procedure: REMOVE_FROM_GROUP
* Description: Removes a program from a request group.
*****
*****/
696 PROCEDURE remove_from_group (
697     p_program_short_name IN VARCHAR2,
698     p_program_application IN VARCHAR2,
699     p_request_group      IN VARCHAR2,
700     p_group_application  IN VARCHAR2
701 ) IS
702 BEGIN
703     apps.fnd_program.remove_from_group(
704         program_short_name => p_program_short_name,
705         program_application => p_program_application,
706         request_group      => p_request_group,
707         group_application  => p_group_application
708     );
709     DBMS_OUTPUT.PUT_LINE('Program removed from group successfully.');
```

710 EXCEPTION

```

711         WHEN OTHERS THEN
712             DBMS_OUTPUT.PUT_LINE('Error removing program from group: ' || SQLCODE || ' - ' || SQLERRM);
713     END remove_from_group;
714
715
716 /*****
717  * Procedure: DELETE_EXECUTABLE
718  * Description: Deletes an executable.
719
720 *****/
721 PROCEDURE delete_executable (
722     p_executable_short_name IN VARCHAR2,
723     p_application            IN VARCHAR2
724 ) IS
725 BEGIN
726     apps.fnd_program.delete_executable(
727         executable_short_name => p_executable_short_name,
728         application           => p_application
729     );
730     DBMS_OUTPUT.PUT_LINE('Executable successfully deleted.');
```

```

731 EXCEPTION
732     WHEN OTHERS THEN
733         DBMS_OUTPUT.PUT_LINE('Error deleting executable: ' || SQLCODE || ' - ' || SQLERRM);
734 END delete_executable;
735
736 /*****
737  * Procedure: DELETE_PROGRAM
738  * Description: Deletes a program from the system.
739
740 *****/
741 PROCEDURE delete_program (
742     p_program_short_name IN VARCHAR2,
743     p_application_code   IN VARCHAR2
744 ) IS
745 BEGIN
746     apps.fnd_program.delete_program(
747         program_short_name => p_program_short_name,
748         application        => p_application_code
749     );
750     DBMS_OUTPUT.PUT_LINE('Program successfully deleted.');
```

```

751 EXCEPTION
752     WHEN OTHERS THEN
753         DBMS_OUTPUT.PUT_LINE('Error deleting program: ' || SQLCODE || ' - ' || SQLERRM);
754 END delete_program;
755
756 /*****
757  * Procedure: MAIN
758  * Description: Main procedure.
759
760 *****/
761 PROCEDURE main IS
762 BEGIN
763     DBMS_OUTPUT.PUT_LINE('Executing main process...');
```

```

764 END main;
765
766 END xxqgen_api_pkg_dk;
767 /
768 CREATE OR REPLACE PACKAGE BODY XXQGEN_INVOICE_AP_STATUS
```

```

765 IS
766 FUNCTION beforereport (P_END_CREATION_DATE IN DATE, P_START_CREATION_DATE IN DATE,
P_TRADING_PARTNER IN VARCHAR2) RETURN BOOLEAN
767 AS
768     CURSOR CUR_INVOICE_DATA IS
769         SELECT
770             inv1.vendor_id AS c_vendor_id,
771             hp.party_name AS trading_partner,
772             TO_CHAR(inv1.invoice_date, 'DD-MON-RRRR HH24:MI') AS invoice_date,
773             b.batch_name AS batch_name,
774             inv1.invoice_id AS c_invoice_id,
775             inv1.invoice_num AS invoice_num,
776             (SELECT DISTINCT poh.segment1 AS c_po_number
777              FROM po_headers poh,
778                   po_distributions_all pda,
779                   ap_invoice_distributions_all aida
780              WHERE poh.po_header_id = pda.po_header_id
781                    AND pda.po_distribution_id = aida.po_distribution_id
782                    AND aida.invoice_id = inv1.invoice_id) AS po_number,
783             DECODE(inv1.invoice_currency_code,
784                   'USD', inv1.invoice_amount,
785                   inv1.base_amount) AS original_amount,
786             DECODE(inv1.invoice_currency_code,
787                   'USD', inv1.invoice_amount,
788                   inv1.base_amount)
789             - DECODE(inv1.payment_currency_code,
790                   'USD', NVL(inv1.amount_paid, 0) + NVL(discount_amount_taken, 0),
791                   ROUND((DECODE(inv1.payment_cross_rate_type,
792                                'EMU FIXED', 1 / inv1.payment_cross_rate,
793                                inv1.exchange_rate) * NVL(inv1.amount_paid, 0)),
794                                f.precision)
795                   + ROUND((DECODE(inv1.payment_cross_rate_type,
796                                'EMU FIXED', 1 / inv1.payment_cross_rate,
797                                inv1.exchange_rate) *
798                                NVL(inv1.discount_amount_taken, 0)),
799                                f.precision)
800             ) AS amount_remaining,
801             inv1.description AS c_description,
802             apps.ap_invoices_pkg.get_approval_status(
803                 inv1.invoice_id, inv1.invoice_amount,
804                 inv1.payment_status_flag, inv1.invoice_type_lookup_code
805             ) AS approval_status,
806             aiaha.last_update_date AS due_date,
807             apt.name AS payment_term,
808             aiaha.response AS notes,
809             aiaha.approver_name
810         FROM
811             hz_parties hp,
812             ap_invoices_all inv1,
813             ap_batches_all b,
814             ap_payment_schedules_all s,
815             ap_terms apt,
816             ap_inv_aprvl_hist_all aiaha,
817             fnd_currencies_vl f
818         WHERE hp.party_id = inv1.party_id
819               AND (hp.party_id = P_TRADING_PARTNER OR P_TRADING_PARTNER IS NULL)
820               AND b.batch_id(+) = inv1.batch_id
821               AND s.invoice_id(+) = inv1.invoice_id
822               AND apt.term_id = inv1.terms_id
823               AND aiaha.invoice_id(+) = inv1.invoice_id
824               AND f.currency_code = inv1.invoice_currency_code
825               AND f.currency_code = 'USD'
826               AND TRUNC(inv1.creation_date) >= NVL(TRUNC(P_START_CREATION_DATE),
TRUNC(inv1.creation_date))
827               AND TRUNC(inv1.creation_date) <= NVL(TRUNC(P_END_CREATION_DATE),
TRUNC(inv1.creation_date));
828
829     TYPE CUR_INVOICE_DATA_TYPE IS TABLE OF CUR_INVOICE_DATA%ROWTYPE INDEX BY
PLS_INTEGER;

```

```

827         INV_TBL CUR_INVOICE_DATA_TYPE;
828
829 BEGIN
830     -- Free unused memory
831     DBMS_SESSION.free_unused_user_memory;
832
833     -- Open the cursor and fetch data in bulk
834     OPEN CUR_INVOICE_DATA;
835     LOOP
836         FETCH CUR_INVOICE_DATA BULK COLLECT INTO INV_TBL LIMIT 100;
837
838         -- Process the fetched data
839         FORALL IDX IN 1 .. INV_TBL.COUNT
840             INSERT INTO XXQGEN_PO_APPROV_STATUS_DK -- Replace with actual table name
841                 (VENDOR_ID,
842                 TRADING_PARTNER,
843                 INVOICE_DATE,
844                 BATCH_NAME,
845                 INVOICE_ID,
846                 INVOICE_NUM,
847                 PO_NUMBER,
848                 ORIGINAL_AMOUNT,
849                 AMOUNT_REMAINING,
850                 DESCRIPTION,
851                 APPROVAL_STATUS,
852                 DUE_DATE,
853                 payment_term,
854                 NOTES,
855                 APPROVER_NAME)
856             VALUES
857                 (INV_TBL(IDX).c_vendor_id,
858                 INV_TBL(IDX).trading_partner,
859                 INV_TBL(IDX).invoice_date,
860                 INV_TBL(IDX).batch_name,
861                 INV_TBL(IDX).c_invoice_id,
862                 INV_TBL(IDX).invoice_num,
863                 INV_TBL(IDX).po_number,
864                 INV_TBL(IDX).original_amount,
865                 INV_TBL(IDX).amount_remaining,
866                 INV_TBL(IDX).c_description,
867                 INV_TBL(IDX).approval_status,
868                 INV_TBL(IDX).due_date,
869                 INV_TBL(IDX).payment_term,
870                 INV_TBL(IDX).notes,
871                 INV_TBL(IDX).approver_name);
872
873         EXIT WHEN INV_TBL.COUNT = 0;
874     END LOOP;
875
876     -- Close the cursor
877     CLOSE CUR_INVOICE_DATA;
878
879     RETURN TRUE;
880
881 EXCEPTION
882     WHEN OTHERS THEN
883         FND_FILE.PUT_LINE(FND_FILE.LOG, 'ERROR: ' || SQLCODE || ' : ' || SQLERRM);
884         RETURN FALSE;
885 END beforereport;
886
887 FUNCTION afterreport (p_orgid IN NUMBER) RETURN BOOLEAN
888 AS
889 BEGIN
890     -- Uncomment and specify the table name if truncation is needed
891     -- EXECUTE IMMEDIATE 'TRUNCATE TABLE XXQGEN_PO_APPROV_STATUS_DK';
892     RETURN TRUE;
893 EXCEPTION
894     WHEN OTHERS THEN
895         FND_FILE.PUT_LINE(FND_FILE.LOG, 'ERROR in afterReport: ' || SQLERRM);

```

```

896         RETURN FALSE;
897     END afterreport;
898
899 END XXQGEN_INVOICE_AP_STATUS;
900 CREATE OR REPLACE PACKAGE BODY XXQGEN_PO_REQ_BULK_COLL_DK IS
901     FUNCTION beforereport (
902         P_REQ_TYPE           IN VARCHAR2,
903         P_REQUESTER_NAME     IN VARCHAR2,
904         P_LINE_NUM           IN VARCHAR2,
905         P_REQ_HDR_ID         IN VARCHAR2,
906         P_ORG_ID             IN VARCHAR2
907     ) RETURN BOOLEAN
908 AS
909     -- Cursor for Q_MAIN
910     CURSOR CUR_HEAD_DATA IS
911         SELECT
912             PRHA.REQUISITION_HEADER_ID HEADER_ID,
913             PRHA.PREPARER_ID,
914             PAPF.EMAIL_ADDRESS,
915             PAPF.FULL_NAME,
916             PRHA.SEGMENT1,
917             PRHA.DESCRPTION,
918             PRHA.AUTHORIZATION_STATUS AUTH_STATUS,
919             PRHA.TYPE_LOOKUP_CODE LOOKUP_CODE,
920             PRHA.ORG_ID,
921             HOU.NAME_ORG_NAME
922         FROM
923             PO_REQUISITION_HEADERS_ALL PRHA,
924             PER_ALL_PEOPLE_F PAPF,
925             HR_ORGANIZATION_UNITS HOU,
926             PO_DOCUMENT_TYPES_ALL_VL PDTAV
927         WHERE
928             PRHA.PREPARER_ID = PAPF.PERSON_ID
929             AND PRHA.ORG_ID = HOU.ORGANIZATION_ID
930             AND PDTAV.DOCUMENT_SUBTYPE = PRHA.TYPE_LOOKUP_CODE
931             AND PRHA.ORG_ID = PDTAV.ORG_ID
932             AND PDTAV.DOCUMENT_TYPE_CODE = 'REQUISITION'
933             AND (PRHA.REQUISITION_HEADER_ID = P_REQ_HDR_ID OR P_REQ_HDR_ID IS NULL)
934             AND (PRHA.ORG_ID = P_ORG_ID OR P_ORG_ID IS NULL)
935             AND (PRHA.TYPE_LOOKUP_CODE = P_REQ_TYPE OR P_REQ_TYPE IS NULL)
936             AND (PAPF.FULL_NAME = P_REQUESTER_NAME OR P_REQUESTER_NAME IS NULL);
937
938     -- Cursor for Q_MAIN2
939     CURSOR CUR_LINE_DATA (HEADER_ID IN NUMBER) IS
940         SELECT
941             PRLA.LINE_NUM,
942             MSIB.SEGMENT1 NAME,
943             PRLA.ITEM_DESCRIPTION DESCRIPTION,
944             PRLA.ITEM_ID,
945             PRLA.QUANTITY,
946             PRLA.UNIT_PRICE,
947             PRLA.QUANTITY * PRLA.UNIT_PRICE AMOUNT,
948             GCC.SEGMENT1 || '-' || GCC.SEGMENT2 || '-' || GCC.SEGMENT3 || '-' ||
949             GCC.SEGMENT4 AS CHARGE_ACCOUNT
950         FROM
951             PO_REQUISITION_LINES_ALL PRLA,
952             PO_REQ_DISTRIBUTIONS_ALL PRDA,
953             MTL_SYSTEM_ITEMS_B MSIB,
954             GL_CODE_COMBINATIONS GCC
955         WHERE
956             PRDA.REQUISITION_LINE_ID = PRLA.REQUISITION_LINE_ID
957             AND PRDA.CODE_COMBINATION_ID = GCC.CODE_COMBINATION_ID
958             AND MSIB.INVENTORY_ITEM_ID = PRLA.ITEM_ID
959             AND (PRLA.REQUISITION_HEADER_ID = HEADER_ID OR HEADER_ID IS NULL);
960
961     -- Variable to store fetched data
962     TYPE CUR_HEAD_DATA_TYPE IS TABLE OF CUR_HEAD_DATA%ROWTYPE INDEX BY PLS_INTEGER;
963     HEAD_TBL CUR_HEAD_DATA_TYPE;

```

```

964         TYPE CUR_LINE_DATA_TYPE IS TABLE OF CUR_LINE_DATA%ROWTYPE INDEX BY PLS_INTEGER;
965         LINE_TBL          CUR_LINE_DATA_TYPE;
966
967 BEGIN
968     -- Fetch data from Q_MAIN
969     OPEN CUR_HEAD_DATA;
970     LOOP
971         FETCH CUR_HEAD_DATA BULK COLLECT INTO HEAD_TBL LIMIT 100;
972
973         -- Exit when no more rows are found
974         EXIT WHEN HEAD_TBL.COUNT = 0;
975
976         FOR I IN 1 .. HEAD_TBL.COUNT LOOP
977             -- Fetch line data for the current header
978             OPEN CUR_LINE_DATA(HEAD_TBL(I).HEADER_ID);
979             LOOP
980                 FETCH CUR_LINE_DATA BULK COLLECT INTO LINE_TBL LIMIT 100;
981
982                 -- Exit when no more rows are found
983                 EXIT WHEN LINE_TBL.COUNT = 0;
984
985                 FOR J IN 1 .. LINE_TBL.COUNT LOOP
986                     -- Insert consolidated data into staging table
987                     INSERT INTO XXQGEN_PO_REQ_DK (
988                         HEADER_ID,
989                         PREPARER_ID,
990                         EMAIL_ADDRESS,
991                         FULL_NAME,
992                         SEGMENT1,
993                         DESCRIPTION,
994                         AUTH_STATUS,
995                         LOOKUP_CODE,
996                         ORG_ID,
997                         ORG_NAME,
998                         LINE_NUM,
999                         ITEM_NAME,
1000                        ITEM_DESCRIPTION,
1001                        ITEM_ID,
1002                        QUANTITY,
1003                        UNIT_PRICE,
1004                        AMOUNT,
1005                        CHARGE_ACCOUNT
1006                    )
1007                    VALUES (
1008                        HEAD_TBL(I).HEADER_ID,
1009                        HEAD_TBL(I).PREPARER_ID,
1010                        HEAD_TBL(I).EMAIL_ADDRESS,
1011                        HEAD_TBL(I).FULL_NAME,
1012                        HEAD_TBL(I).SEGMENT1,
1013                        HEAD_TBL(I).DESCRIPTION,
1014                        HEAD_TBL(I).AUTH_STATUS,
1015                        HEAD_TBL(I).LOOKUP_CODE,
1016                        HEAD_TBL(I).ORG_ID,
1017                        HEAD_TBL(I).ORG_NAME,
1018                        LINE_TBL(J).LINE_NUM,
1019                        LINE_TBL(J).NAME,
1020                        LINE_TBL(J).DESCRIPTION,
1021                        LINE_TBL(J).ITEM_ID,
1022                        LINE_TBL(J).QUANTITY,
1023                        LINE_TBL(J).UNIT_PRICE,
1024                        LINE_TBL(J).AMOUNT,
1025                        LINE_TBL(J).CHARGE_ACCOUNT
1026                    );
1027                 END LOOP;
1028             END LOOP;
1029             CLOSE CUR_LINE_DATA;
1030         END LOOP;
1031     END LOOP;
1032     CLOSE CUR_HEAD_DATA;

```



```

1033
1034         RETURN TRUE;
1035     EXCEPTION
1036         WHEN OTHERS THEN
1037             FND_FILE.PUT_LINE(FND_FILE.LOG, 'Error in beforeTrigger: ' || SQLCODE || ' -
                ' || SQLERRM);
1038             RETURN FALSE;
1039     END beforereport;
1040
1041     FUNCTION afterreport
1042     RETURN BOOLEAN
1043     IS
1044     BEGIN
1045         RETURN TRUE;
1046     EXCEPTION
1047         WHEN OTHERS THEN
1048             RETURN FALSE;
1049     END afterreport;
1050 END XXQGEN_PO_REQ_BULK_COLL_DK;
1051 -- 1st Query
1052 SELECT
1053     RQHA.SEGMENT1,
1054     PAPF.PERSON_ID PREPARER_ID,
1055     FU.USER_ID LAST_UPDATED_BY,
1056     FU.USER_NAME CREATED_BY,
1057     RQHA.DESCRPTION,
1058     RQHA.TYPE_LOOKUP_CODE TYPEEE,
1059     PLC.LOOKUP_CODE,
1060     HOU.ORGANIZATION_ID ORG_ID
1061 FROM
1062     PO_REQUISITION_HEADERS_ALL RQHA,
1063     PER_ALL_PEOPLE_F PAPF,
1064     FND_USER FU,
1065     PO_DOCUMENT_TYPES_ALL_VL PDT,
1066     PO_LOOKUP_CODES PLC,
1067     HR_OPERATING_UNITS HOU
1068 WHERE
1069     RQHA.PREPARER_ID=PAPF.PERSON_ID
1070     AND RQHA.LAST_UPDATED_BY = FU.USER_ID
1071     AND RQHA.AUTHORIZATION_STATUS =PLC.LOOKUP_CODE
1072     AND PLC.LOOKUP_TYPE ='AUTHORIZATION STATUS'
1073     AND RQHA.TYPE_LOOKUP_CODE=PDT.DOCUMENT_SUBTYPE
1074     AND RQHA.ORG_ID=PDT.ORG_ID
1075     AND PDT.DOCUMENT_TYPE_CODE = 'REQUISITION'
1076     AND RQHA.ORG_ID=HOU.ORGANIZATION_ID
1077     AND RQHA.SEGMENT1='14438'
1078 ;
1079
1080 -- 2nd query
1081 select prha.requisition_header_id
1082 ,prla.line_num
1083 ,prla.requisition_line_id
1084 ,mc.category_id
1085 ,papf.full_name
1086 ,mc.segment2
1087 ,prla.unit_meas_lookup_code
1088 ,prla.unit_price
1089 ,prla.quantity
1090 ,prla.unit_price*prla.quantity Amount
1091 ,prla.need_by_date
1092 ,msib.segment1
1093 ,msib.description
1094 ,ood.organization_code
1095 ,ood.organization_name
1096 ,prla.to_person_id
1097 ,hl.location_code
1098 ,plt.line_type
1099 from po_requisition_lines_all prla
1100 ,po_requisition_headers_all prha

```

```

1101 ,mtl_categories mc
1102 ,per_all_people_f papf
1103 ,mtl_system_items_b msib
1104 ,org_organization_definitions ood
1105 ,hr_locations hl
1106 ,po_line_types plt
1107 where prla.requisition_header_id=prha.requisition_header_id
1108 and mc.category_id=prla.category_id
1109 and msib.inventory_item_id=prla.item_id
1110 and prla.to_person_id=papf.person_id
1111 and ood.organization_id=msib.organization_id
1112 and ood.organization_id = prla.org_id
1113 and prla.deliver_to_location_id=hl.location_id
1114 and prla.line_type_id=plt.line_type_id
1115 and prha.segment1 = '14438';
1116
1117 -- 3rd query
1118 SELECT
1119     PRD.REQ_LINE_QUANTITY
1120     ,GCC.CHART_OF_ACCOUNTS_ID
1121
1122     ,GCC.SEGMENT1||'-'||GCC.SEGMENT2||'-'||GCC.SEGMENT3||'-'||GCC.SEGMENT4||'-'||GCC.SEGM
1123     ENT5 ACCOUNT_DETAIL
1124 FROM
1125     PO_REQ_DISTRIBUTIONS_ALL PRD
1126     ,GL_CODE_COMBINATIONS GCC
1127     ,PO_REQUISITION_LINES_ALL PRL
1128     ,PO_REQUISITION_HEADERS_ALL PRH
1129 WHERE
1130     PRD.CODE_COMBINATION_ID=GCC.CODE_COMBINATION_ID
1131     AND PRL.REQUISITION_LINE_ID=PRD.REQUISITION_LINE_ID
1132     AND PRL.REQUISITION_HEADER_ID=PRH.REQUISITION_HEADER_ID
1133     AND PRH.SEGMENT1='14438'
1134 ;
1135
1136 -- 4th Query
1137 select prha.requisition_header_id,
1138     prha.preparer_id,
1139     PAPFPRE.FULL_NAME PREPARER_FULL_NAME,
1140     lub.user_name LAST_UPDATED_BY,
1141     cb.user_name CREATED_BY,
1142     PRHA.ORG_ID,
1143     HOU.NAME ORGANIZATION_NAME,
1144     PRHA.TYPE_LOOKUP_CODE,
1145     PRHA.AUTHORIZATION_STATUS AUTH_STATUS,
1146     PLC1.DESCRPTION,
1147     PLC1.DISPLAYED_FIELD,
1148     PRHA.DESCRPTION HEADER_DESC,
1149     PRHA.ENABLED_FLAG,
1150     PRHA.SUMMARY_FLAG,
1151     PRHA.SEGMENT1,
1152     PRLA.REQUISITION_LINE_ID,
1153     PRLA.CATEGORY_ID,
1154     PRLA.PURCHASE BASIS,
1155     MC.SEGMENT1 || ' ' || MC.SEGMENT2 CATEGORY_NAME,
1156     PRLA.QUANTITY,
1157     PRLA.UNIT PRICE,
1158     PRLA.QUANTITY * PRLA.UNIT_PRICE AMOUNT,
1159     PRLA.TO_PERSON_ID,
1160     PAPF.FULL_NAME REQ_PERSON,
1161     MSIB.SEGMENT1 || ' ' || MSIB.DESCRPTION ITEM_ID,
1162     PRLA.ITEM_DESCRIPTION,
1163     PRLA.DESTINATION_TYPE_CODE,
1164     PRLA.NEED_BY_DATE,
1165     PRLA.DELIVER_TO_LOCATION_ID,
1166     HLA.ADDRESS_LINE_1,
1167     HLA.DESCRPTION LOC_DESC,
1168     HLA.POSTAL_CODE,
1169     PRLA.LINE_NUM,

```

```

1168         PRDA.DISTRIBUTION_ID,
1169         PRDA.REQUISITION_LINE_ID,
1170         PRDA.SET_OF_BOOKS_ID,
1171         PRDA.DISTRIBUTION_NUM,
1172         PRLA.REQUEST_ID,
1173         gcc.ACCOUNT_TYPE,
1174         gcc.ENABLED_FLAG,
1175         gcc.SUMMARY_FLAG,
1176         gcc.COMPANY_COST_CENTER_ORG_ID,
1177
1178         GCC.SEGMENT1||'-'||GCC.SEGMENT2||'-'||GCC.SEGMENT3||'-'||GCC.SEGMENT4||'-'||GCC.
1179         SEGMENT5 ACCOUNT_DETAIL
1180
1181 from
1182     po_requisition_headers_all prha,
1183     po_requisition_lines_all prla,
1184     po_req_distributions_all prda,
1185     PER_ALL_PEOPLE_F PAPFPRE,
1186     fnd_user lub,
1187     fnd_user cb,
1188     HR_LOCATIONS_ALL HLA,
1189     PER_ALL_PEOPLE_F PAPF,
1190     HR_ORGANIZATION_UNITS HOU,
1191     MTL_CATEGORIES MC,
1192     MTL_SYSTEM_ITEMS_B MSIB,
1193     ORG_ORGANIZATION_DEFINITIONS OOD,
1194     GL_CODE_COMBINATIONS gcc,
1195     PO_LOOKUP_CODES PLC1
1196 --     PER_ALL_PEOPLE_F PAPFREQ
1197 where
1198     1=1
1199 and prha.requisition_header_id = prla.requisition_header_id
1200 and prha.segment1 = '14438'
1201 AND PAPFPRE.PERSON_ID = PRHA.PREPARER_ID
1202 and prla.last_updated_by = lub.USER_id
1203 and prla.created_by = cb.USER_id
1204 and prda.requisition_line_id = prla.requisition_line_id
1205 AND PRLA.TO_PERSON_ID = PAPF.PERSON_ID
1206 AND HLA.LOCATION_ID = PRLA.DELIVER_TO_LOCATION_ID
1207 AND HOU.ORGANIZATION_ID = PRHA.ORG_ID
1208 AND PRHA.AUTHORIZATION_STATUS = PLC1.LOOKUP_CODE
1209 AND PLC1.LOOKUP_TYPE = 'AUTHORIZATION STATUS'
1210 AND PRLA.CATEGORY_ID = MC.CATEGORY_ID
1211 --AND HLA.PERSON_ID = PRLA.DELIVER_TO_LOCATION_ID
1212 AND MSIB.INVENTORY_ITEM_ID = PRLA.ITEM_ID
1213 AND MSIB.ORGANIZATION_ID = OOD.ORGANIZATION_ID
1214 AND OOD.ORGANIZATION_ID = PRLA.ORG_ID
1215 and prda.code_combination_id = gcc.code_combination_id;
1216 --1. Basic Query: Fetching Requisition Headers
1217 --Retrieve requisition details like requisition number, description, and status.
1218
1219 SELECT
1220     prh.requisition_header_id,
1221     prh.segment1 AS requisition_number,
1222     prh.description,
1223     prh.authorization_status,
1224     prh.preparer_id,
1225     prh.creation_date
1226 FROM
1227     po_requisition_headers_all prh
1228 WHERE
1229     prh.authorization_status = 'APPROVED'; -- Filter for approved requisitions
1230
1231 --2. Intermediate Query: Requisition Lines with Items
1232 --Fetch requisition lines along with item details.
1233
1234 SELECT
1235     prh.segment1 AS requisition_number,
1236     prh.description AS requisition_description,

```

```

1235     prl.line_num AS requisition_line_number,
1236     prl.item_description,
1237     prl.quantity,
1238     prl.unit_price,
1239     prl.need_by_date
1240 FROM
1241     po_requisition_headers_all prh
1242 JOIN
1243     po_requisition_lines_all prl
1244 ON
1245     prh.requisition_header_id = prl.requisition_header_id
1246 WHERE
1247     prh.authorization_status = 'APPROVED'
1248 AND
1249     prl.item_id IS NOT NULL; -- Ensure the line is tied to an item
1250
1251 --3. Advanced Query: Requisition to Purchase Order Link
1252 --Track requisitions converted to purchase orders.
1253
1254 SELECT
1255     prh.segment1 AS requisition_number,
1256     prh.description AS requisition_description,
1257     prl.line_num AS requisition_line_number,
1258     prl.quantity AS req_quantity,
1259     pol.po_header_id,
1260     poh.segment1 AS po_number,
1261     pol.quantity AS po_quantity,
1262     pol.unit_price AS po_unit_price,
1263     poh.authorization_status AS po_status
1264 FROM
1265     po_requisition_headers_all prh
1266 JOIN
1267     po_requisition_lines_all prl
1268 ON
1269     prh.requisition_header_id = prl.requisition_header_id
1270 JOIN
1271     po_req_distributions_all prd
1272 ON
1273     prl.requisition_line_id = prd.requisition_line_id
1274 JOIN
1275     po_distributions_all pd
1276 ON
1277     prd.distribution_id = pd.req_distribution_id
1278 JOIN
1279     po_lines_all pol
1280 ON
1281     pd.po_line_id = pol.po_line_id
1282 JOIN
1283     po_headers_all poh
1284 ON
1285     pol.po_header_id = poh.po_header_id
1286 WHERE
1287     prh.authorization_status = 'APPROVED'
1288 AND
1289     poh.authorization_status IN ('APPROVED', 'IN PROCESS'); -- Filter for PO status
1290
1291 --4. Advanced Query: Requisition to Receiving and Invoices
1292 --Track the full P2P flow from requisitions to receiving and invoices.
1293
1294 SELECT
1295     prh.segment1 AS requisition_number,
1296     prl.line_num AS requisition_line_number,
1297     prl.item_description,
1298     prl.quantity AS req_quantity,
1299     poh.segment1 AS po_number,
1300     pol.quantity AS po_quantity,
1301     rcv.transaction_quantity AS received_quantity,
1302     ap.invoice_num,
1303     ap.invoice_amount,

```

```

1304     ap.payment_status_flag
1305 FROM
1306     po_requisition_headers_all prh
1307 JOIN
1308     po_requisition_lines_all prl
1309 ON
1310     prh.requisition_header_id = prl.requisition_header_id
1311 JOIN
1312     po_req_distributions_all prd
1313 ON
1314     prl.requisition_line_id = prd.requisition_line_id
1315 JOIN
1316     po_distributions_all pd
1317 ON
1318     prd.distribution_id = pd.req_distribution_id
1319 JOIN
1320     po_lines_all pol
1321 ON
1322     pd.po_line_id = pol.po_line_id
1323 JOIN
1324     po_headers_all poh
1325 ON
1326     pol.po_header_id = poh.po_header_id
1327 LEFT JOIN
1328     rcv_transactions rcv
1329 ON
1330     pol.po_line_id = rcv.po_line_id
1331 LEFT JOIN
1332     ap_invoices_all ap
1333 ON
1334     pd.po_distribution_id = ap.po_distribution_id
1335 WHERE
1336     prh.authorization_status = 'APPROVED'
1337 AND
1338     poh.authorization_status = 'APPROVED'
1339 AND
1340     rcv.transaction_type IN ('RECEIVE', 'DELIVER')
1341 AND
1342     ap.invoice_status = 'VALIDATED';
1343 Explanation of Key Tables:
1344 PO_REQUISITION_HEADERS_ALL: Stores requisition headers.
1345 PO_REQUISITION_LINES_ALL: Stores requisition line details.
1346 PO_HEADERS_ALL: Stores purchase order headers.
1347 PO_LINES_ALL: Stores purchase order line details.
1348 RCV_TRANSACTIONS: Stores receiving transactions.
1349 AP_INVOICES_ALL: Stores invoice information.
1350 Notes:
1351 Adjust filters (authorization_status, transaction_type, etc.) based on your requirements.
1352 Use appropriate date filters to limit the data set.
1353 Always test these queries in a non-production environment before use.
1354
1355
1356
1357
1358
1359 You said:
1360 more questions
1361 ChatGPT said:
1362 ChatGPT
1363 Here are additional P2P requisition-related questions, ranging from basic to advanced,
    that you might encounter during an interview or while working on Oracle EBS:
1364
1365 Basic Questions
1366 What is the P2P process?
1367 What is a requisition, and why is it created in the P2P cycle?
1368 What are the key tables involved in storing requisition details in Oracle EBS?
1369 What is the difference between requisition types: Internal and Purchase Requisition?
1370 What is the significance of the authorization_status column in the
    PO_REQUISITION_HEADERS_ALL table?

```

1371 How is the preparer of a requisition identified in the database?
1372 What is the purpose of the item_id in PO_REQUISITION_LINES_ALL?
1373 Intermediate Questions
1374 How do you identify approved requisitions in Oracle EBS?
1375 How can you fetch the list of requisitions created within a specific date range?
1376 How do you link requisitions to purchase orders in the database?
1377 What are the different statuses a requisition can have, and what do they mean?
1378 How do you differentiate between standard purchase orders and blanket purchase agreements in a query?
1379 Explain the relationship between PO_REQUISITION_LINES_ALL and PO_REQ_DISTRIBUTIONS_ALL.
1380 What is the significance of the org_id column in requisition-related tables?
1381 Advanced Questions
1382 How can you track a requisition's progress from creation to invoice payment?
1383 How would you write a query to find requisitions pending approval?
1384 How do you identify requisitions that are fully received but not invoiced?
1385 How can you optimize a query linking requisitions, purchase orders, receiving, and invoices?
1386 What is the difference between RECEIVE and DELIVER transaction types in the receiving process?
1387 How would you find out if a requisition line has multiple distributions?
1388 Explain how financials (GL codes, cost centers, etc.) are tied to requisitions in the database.
1389 How do you handle requisitions created with invalid items or suppliers?
1390 Scenario-Based Questions
1391 A requisition has been approved but isn't showing up in the buyer's worklist. What steps would you take to debug this?
1392 A requisition is created, but the approval workflow fails. How would you identify the issue?
1393 Some requisitions are stuck in a pre-approved status. How can you find and resolve these requisitions?
1394 How do you identify requisitions that have not been converted to purchase orders?
1395 A purchase order linked to a requisition is closed, but the requisition line is still open. How would you debug this?
1396 Technical Questions
1397 How do you enable debugging for a P2P workflow in Oracle EBS?
1398 What are the APIs used to create and update requisitions programmatically?
1399 How do you handle changes to a requisition after it's been approved?
1400 Explain the role of PO_REQAPPROVAL_INIT1 in requisition approval workflows.
1401 What is the difference between po_requisition_headers_all and po_requisition_headers views?
1402 How do you ensure data consistency between requisitions and purchase orders in a multi-org setup?
1403 Analytical Questions
1404 How do you create a report that lists all requisitions and their associated purchase orders, including their statuses?
1405 How would you find the total spend by supplier based on requisitions?
1406 How can you identify trends in requisition creation by department or preparer?
1407 How would you identify high-value requisitions that require additional approvals?
1408 How can you create a dashboard to monitor requisitions pending in various statuses?
1409 Let me know if you'd like detailed queries or solutions to any of these questions!'
1410
1411
1412
1413
1414
1415
1416 You said:
1417 query questions on fnd user , people , concurrent program, request program etc all ebs operation
1418 ChatGPT said:
1419 ChatGPT
1420 Here's a comprehensive list of query questions related to Oracle EBS operations involving FND Users, People, Concurrent Programs, Request Programs, and more. These queries cover various functional areas and help understand EBS processes:
1421
1422 FND_USER Queries
1423 Retrieve a list of all active users:
1424
1425 sql

```

1426 Copy code
1427 SELECT
1428     user_id,
1429     user_name,
1430     start_date,
1431     end_date,
1432     last_logon_date
1433 FROM
1434     fnd_user
1435 WHERE
1436     account_status = 'ACTIVE';
1437 Find user details based on responsibilities assigned:
1438
1439 sql
1440 Copy code
1441 SELECT
1442     fu.user_name,
1443     fu.description AS user_description,
1444     fr.responsibility_name,
1445     frv.start_date,
1446     frv.end_date
1447 FROM
1448     fnd_user fu
1449 JOIN
1450     fnd_user_resp_groups_direct frv
1451 ON
1452     fu.user_id = frv.user_id
1453 JOIN
1454     fnd_responsibility_tl fr
1455 ON
1456     frv.responsibility_id = fr.responsibility_id
1457 WHERE
1458     frv.end_date IS NULL;
1459 Track users who haven't logged in for a long time:
1460
1461 sql
1462 Copy code
1463 SELECT
1464     user_name,
1465     last_logon_date,
1466     account_status
1467 FROM
1468     fnd_user
1469 WHERE
1470     last_logon_date < SYSDATE - 90; -- No login in the last 90 days
1471 Identify locked user accounts:
1472
1473 sql
1474 Copy code
1475 SELECT
1476     user_name,
1477     start_date,
1478     end_date
1479 FROM
1480     fnd_user
1481 WHERE
1482     account_status = 'LOCKED';
1483 HR People (PER_ALL_PEOPLE_F) Queries
1484 Retrieve active employees and their user accounts:
1485
1486 sql
1487 Copy code
1488 SELECT
1489     paf.person_id,
1490     papf.full_name,
1491     fnd_user.user_name,
1492     papf.date_of_birth
1493 FROM
1494     per_all_people_f papf

```

```

1495 JOIN
1496     per_all_assignments_f paf
1497 ON
1498     papf.person_id = paf.person_id
1499 JOIN
1500     fnd_user
1501 ON
1502     paf.person_id = fnd_user.employee_id
1503 WHERE
1504     SYSDATE BETWEEN papf.effective_start_date AND papf.effective_end_date
1505 AND
1506     paf.assignment_status_type IN ('ACTIVE');
1507 Find employees who don't have user accounts:
1508
1509 sql
1510 Copy code
1511 SELECT
1512     papf.person_id,
1513     papf.full_name
1514 FROM
1515     per_all_people_f papf
1516 WHERE
1517     papf.person_id NOT IN (
1518         SELECT
1519             employee_id
1520         FROM
1521             fnd_user
1522         WHERE
1523             employee_id IS NOT NULL
1524     );
1525 Concurrent Programs Queries
1526 Fetch details of all concurrent programs:
1527
1528 sql
1529 Copy code
1530 SELECT
1531     concurrent_program_id,
1532     concurrent_program_name,
1533     description,
1534     enabled_flag
1535 FROM
1536     fnd_concurrent_programs;
1537 Retrieve details of requests submitted for a specific program:
1538
1539 sql
1540 Copy code;
1541 SELECT
1542     fcr.request_id,
1543     fcr.concurrent_program_id,
1544     fcr.status_code,
1545     fcr.phase_code,
1546     fcr.request_date
1547 FROM
1548     fnd_concurrent_requests fcr
1549 JOIN
1550     fnd_concurrent_programs fcp
1551 ON
1552     fcr.concurrent_program_id = fcp.concurrent_program_id
1553 WHERE
1554     fcp.concurrent_program_name = 'YOUR_PROGRAM_NAME';
1555 Identify programs with the most failures:
1556
1557 sql
1558 Copy code
1559 SELECT
1560     fcp.concurrent_program_name,
1561     COUNT(*) AS failure_count
1562 FROM
1563     fnd_concurrent_requests fcr

```



```

1564 JOIN
1565     fnd_concurrent_programs fcp
1566 ON
1567     fcr.concurrent_program_id = fcp.concurrent_program_id
1568 WHERE
1569     fcr.phase_code = 'COMPLETED'
1570 AND
1571     fcr.status_code = 'ERROR'
1572 GROUP BY
1573     fcp.concurrent_program_name
1574 ORDER BY
1575     failure_count DESC;
1576 Retrieve running requests:
1577
1578 sql
1579 Copy code
1580 SELECT
1581     request_id,
1582     concurrent_program_id,
1583     phase_code,
1584     status_code,
1585     logfile_name
1586 FROM
1587     fnd_concurrent_requests
1588 WHERE
1589     phase_code = 'RUNNING';
1590 Request Sets and Programs Queries
1591 Fetch details of request sets:
1592
1593 sql
1594 Copy code
1595 SELECT
1596     request_set_id,
1597     request_set_name,
1598     application_id
1599 FROM
1600     fnd_request_sets;
1601 List all requests in a request set:
1602
1603 sql
1604 Copy code
1605 SELECT
1606     frs.request_set_id,
1607     frs.request_set_name,
1608     frs.stage_id,
1609     fcp.concurrent_program_name
1610 FROM
1611     fnd_request_set_stages frss
1612 JOIN
1613     fnd_request_sets frs
1614 ON
1615     frss.request_set_id = frs.request_set_id
1616 JOIN
1617     fnd_concurrent_programs fcp
1618 ON
1619     frss.concurrent_program_id = fcp.concurrent_program_id;
1620 Security and Responsibility Queries
1621 Retrieve responsibilities assigned to a user:
1622
1623 sql
1624 Copy code
1625 SELECT
1626     fu.user_name,
1627     fr.responsibility_name
1628 FROM
1629     fnd_user fu
1630 JOIN
1631     fnd_user_resp_groups frg
1632 ON

```

```

1633         fu.user_id = frg.user_id
1634 JOIN
1635     fnd_responsibility_tl fr
1636 ON
1637     frg.responsibility_id = fr.responsibility_id
1638 WHERE
1639     fu.user_name = 'YOUR_USER_NAME';
1640 Check if a responsibility is end-dated:
1641
1642 sql
1643 Copy code
1644 SELECT
1645     responsibility_name,
1646     start_date,
1647     end_date
1648 FROM
1649     fnd_responsibility
1650 WHERE
1651     responsibility_id = 'RESPONSIBILITY_ID';
1652 Diagnostic and Utility Queries
1653 Check invalid logins:
1654
1655 sql
1656 Copy code
1657 SELECT
1658     user_name,
1659     last_logon_date,
1660     failed_login_attempts
1661 FROM
1662     fnd_user
1663 WHERE
1664     failed_login_attempts > 0;
1665 Audit changes to user accounts:
1666
1667 sql
1668 Copy code
1669 SELECT
1670     user_id,
1671     user_name,
1672     last_update_date,
1673     last_updated_by
1674 FROM
1675     fnd_user
1676 WHERE
1677     last_update_date > SYSDATE - 30; -- Changes in the last 30 days
1678 Monitor long-running concurrent requests:
1679
1680 sql
1681 Copy code
1682 SELECT
1683     request_id,
1684     user_name,
1685     concurrent_program_name,
1686     requested_start_date,
1687     elapsed_time
1688 FROM
1689     fnd_concurrent_requests
1690 WHERE
1691     elapsed_time > 30; -- Requests running for more than 30 minutes
1692 Fetch error messages for failed requests:
1693
1694 sql
1695 Copy code
1696 SELECT
1697     request_id,
1698     phase_code,
1699     status_code,
1700     log_file_name,
1701     oracle_process_id

```

```

1702 FROM
1703     fnd_concurrent_requests
1704 WHERE
1705     phase_code = 'COMPLETED'
1706 AND
1707     status_code = 'ERROR';
1708 Let me know which area you'd like to explore further or if you need explanations or more
    queries!'
1709
1710
1711
1712
1713
1714
1715 You said:
1716 more query questions
1717 ChatGPT said:
1718 ChatGPT
1719 Here's an extended set of query questions for Oracle EBS that you can explore across
    different domains such as FND Users, Concurrent Programs, Responsibilities, and
    Security, along with some more advanced operational queries:
1720
1721 FND_USER and User-Related Queries
1722 List users with expired accounts:
1723
1724 sql
1725 Copy code
1726 SELECT
1727     user_id,
1728     user_name,
1729     end_date
1730 FROM
1731     fnd_user
1732 WHERE
1733     end_date < SYSDATE;
1734 Find the last logon details for a specific user:
1735
1736 sql
1737 Copy code
1738 SELECT
1739     user_name,
1740     last_logon_date
1741 FROM
1742     fnd_user
1743 WHERE
1744     user_name = 'YOUR_USER_NAME';
1745 Identify all system administrators in the application:
1746
1747 sql
1748 Copy code
1749 SELECT
1750     fu.user_name,
1751     fr.responsibility_name
1752 FROM
1753     fnd_user fu
1754 JOIN
1755     fnd_user_resp_groups_direct furg
1756 ON
1757     fu.user_id = furg.user_id
1758 JOIN
1759     fnd_responsibility fr
1760 ON
1761     furg.responsibility_id = fr.responsibility_id
1762 WHERE
1763     fr.responsibility_name LIKE '%System Administrator%';
1764 Find users linked to terminated employees:
1765
1766 sql
1767 Copy code

```

```

1768 SELECT
1769     fu.user_name,
1770     fu.employee_id
1771 FROM
1772     fnd_user fu
1773 JOIN
1774     per_all_people_f papf
1775 ON
1776     fu.employee_id = papf.person_id
1777 WHERE
1778     papf.effective_end_date < SYSDATE;
1779 Responsibilities and Roles Queries
1780 List responsibilities not assigned to any user:
1781
1782 sql
1783 Copy code
1784 SELECT
1785     responsibility_name
1786 FROM
1787     fnd_responsibility
1788 WHERE
1789     responsibility_id NOT IN (
1790         SELECT
1791             DISTINCT responsibility_id
1792         FROM
1793             fnd_user_resp_groups_direct
1794     );
1795 Fetch a user's responsibilities with effective dates:
1796
1797 sql
1798 Copy code
1799 SELECT
1800     fu.user_name,
1801     fr.responsibility_name,
1802     frg.start_date,
1803     frg.end_date
1804 FROM
1805     fnd_user fu
1806 JOIN
1807     fnd_user_resp_groups frg
1808 ON
1809     fu.user_id = frg.user_id
1810 JOIN
1811     fnd_responsibility fr
1812 ON
1813     frg.responsibility_id = fr.responsibility_id
1814 WHERE
1815     fu.user_name = 'YOUR_USER_NAME';
1816 Check if a specific responsibility is available in a given application:
1817
1818 sql
1819 Copy code
1820 SELECT
1821     responsibility_name
1822 FROM
1823     fnd_responsibility
1824 WHERE
1825     application_id = 'APPLICATION_ID'
1826 AND
1827     responsibility_name = 'RESPONSIBILITY_NAME';
1828 Concurrent Programs Queries
1829 Fetch all disabled concurrent programs:
1830
1831 sql
1832 Copy code
1833 SELECT
1834     concurrent_program_name,
1835     description
1836 FROM

```

```

1837         fnd_concurrent_programs
1838 WHERE
1839     enabled_flag = 'N';
1840 Find completed concurrent requests with warnings:
1841
1842 sql
1843 Copy code
1844 SELECT
1845     request_id,
1846     phase_code,
1847     status_code
1848 FROM
1849     fnd_concurrent_requests
1850 WHERE
1851     phase_code = 'COMPLETED'
1852 AND
1853     status_code = 'WARNING';
1854 Get the execution time of a specific request:
1855
1856 sql
1857 Copy code
1858 SELECT
1859     request_id,
1860     requested_start_date,
1861     actual_start_date,
1862     actual_completion_date,
1863     (actual_completion_date - actual_start_date) * 24 * 60 AS execution_time_minutes
1864 FROM
1865     fnd_concurrent_requests
1866 WHERE
1867     request_id = 'REQUEST_ID';
1868 Retrieve logs of failed concurrent requests:
1869
1870 sql
1871 Copy code
1872 SELECT
1873     request_id,
1874     logfile_name,
1875     logfile_node
1876 FROM
1877     fnd_concurrent_requests
1878 WHERE
1879     status_code = 'ERROR';
1880 People (PER_ALL_PEOPLE_F) Queries
1881 Find all employees under a specific manager:
1882
1883 sql
1884 Copy code
1885 SELECT
1886     employee_id,
1887     full_name,
1888     assignment_id
1889 FROM
1890     per_all_assignments_f
1891 WHERE
1892     manager_id = 'MANAGER_PERSON_ID';
1893 List employees in a specific department:
1894
1895 sql
1896 Copy code
1897 SELECT
1898     papf.person_id,
1899     papf.full_name,
1900     per.organization_id,
1901     hrorg.name AS department_name
1902 FROM
1903     per_all_people_f papf
1904 JOIN
1905     per_all_assignments_f per

```

```

1906     ON
1907         papf.person_id = per.person_id
1908     JOIN
1909         hr_all_organization_units hrorg
1910     ON
1911         per.organization_id = hrorg.organization_id
1912     WHERE
1913         hrorg.name = 'DEPARTMENT_NAME';
1914     Get details of all terminated employees:
1915
1916     sql
1917     Copy code
1918     SELECT
1919         papf.person_id,
1920         papf.full_name,
1921         paf.effective_end_date
1922     FROM
1923         per_all_people_f papf
1924     JOIN
1925         per_all_assignments_f paf
1926     ON
1927         papf.person_id = paf.person_id
1928     WHERE
1929         paf.assignment_status_type = 'TERMINATED';
1930     System Administration Queries
1931     Find top N requests consuming the most runtime:
1932
1933     sql
1934     Copy code
1935     SELECT
1936         request_id,
1937         concurrent_program_id,
1938         elapsed_time,
1939         requested_start_date
1940     FROM
1941         fnd_concurrent_requests
1942     ORDER BY
1943         elapsed_time DESC
1944     FETCH FIRST 10 ROWS ONLY;
1945     List all custom concurrent programs in the system:
1946
1947     sql
1948     Copy code
1949     SELECT
1950         concurrent_program_name,
1951         description
1952     FROM
1953         fnd_concurrent_programs
1954     WHERE
1955         concurrent_program_name LIKE 'XX%';
1956     Find all requests submitted by a specific user:
1957
1958     sql
1959     Copy code
1960     SELECT
1961         fcr.request_id,
1962         fcr.concurrent_program_id,
1963         fcr.phase_code,
1964         fcr.status_code,
1965         fcr.request_date
1966     FROM
1967         fnd_concurrent_requests fcr
1968     JOIN
1969         fnd_user fu
1970     ON
1971         fcr.requested_by = fu.user_id
1972     WHERE
1973         fu.user_name = 'YOUR_USER_NAME';
1974     Audit which users have added or updated concurrent programs:

```

```

1975
1976 sql
1977 Copy code
1978 SELECT
1979     concurrent_program_id,
1980     concurrent_program_name,
1981     last_updated_by,
1982     last_update_date
1983 FROM
1984     fnd_concurrent_programs
1985 WHERE
1986     last_update_date > SYSDATE - 30; -- Programs updated in the last 30 days
1987 Diagnostic and Validation Queries
1988 Check users with invalid configurations:
1989
1990 sql
1991 Copy code
1992 SELECT
1993     fu.user_name,
1994     fu.account_status
1995 FROM
1996     fnd_user fu
1997 WHERE
1998     NVL(fu.end_date, SYSDATE) < SYSDATE
1999 OR
2000     NVL(fu.start_date, SYSDATE) > SYSDATE;
2001 Validate if a concurrent program is associated with any request group:
2002
2003 sql
2004 Copy code
2005 SELECT
2006     fcp.concurrent_program_name,
2007     frg.request_group_name
2008 FROM
2009     fnd_request_group_units frgu
2010 JOIN
2011     fnd_request_groups frg
2012 ON
2013     frgu.request_group_id = frg.request_group_id
2014 JOIN
2015     fnd_concurrent_programs fcp
2016 ON
2017     frgu.request_unit_id = fcp.concurrent_program_id
2018 WHERE
2019     fcp.concurrent_program_name = 'YOUR_PROGRAM_NAME';
2020 Check for unused responsibilities:
2021
2022 sql
2023 Copy code
2024 SELECT
2025     responsibility_id,
2026     responsibility_name
2027 FROM
2028     fnd_responsibility
2029 WHERE
2030     responsibility_id NOT IN (
2031         SELECT
2032             DISTINCT responsibility_id
2033         FROM
2034             fnd_user_resp_groups_direct
2035     );
2036 Would you like additional questions on any specific area? Or do you need explanations or
step-by-step solutions? Let me know!
2037
2038
2039
2040
2041
2042

```

```

2043 You said:
2044 more query questions
2045 ChatGPT said:
2046 ChatGPT
2047 Here are more query questions for Oracle EBS focusing on FND User, Concurrent Programs,
Responsibilities, Requests, Data Security, and other operational aspects:

2048
2049 FND User and Security Queries
2050 Find all inactive users:
2051
2052 sql
2053 Copy code
2054 SELECT
2055     user_name,
2056     start_date,
2057     end_date
2058 FROM
2059     fnd_user
2060 WHERE
2061     NVL(end_date, SYSDATE) < SYSDATE;
2062 Check if a specific user has logged in within the past 30 days:
2063
2064 sql
2065 Copy code
2066 SELECT
2067     user_name,
2068     last_logon_date
2069 FROM
2070     fnd_user
2071 WHERE
2072     user_name = 'USERNAME'
2073 AND
2074     last_logon_date >= SYSDATE - 30;
2075 List all users with SYSADMIN privileges:
2076
2077 sql
2078 Copy code
2079 SELECT
2080     fu.user_name,
2081     fr.responsibility_name
2082 FROM
2083     fnd_user fu
2084 JOIN
2085     fnd_user_resp_groups_direct furg
2086 ON
2087     fu.user_id = furg.user_id
2088 JOIN
2089     fnd_responsibility fr
2090 ON
2091     furg.responsibility_id = fr.responsibility_id
2092 WHERE
2093     fr.responsibility_name LIKE '%SYSADMIN%';
2094 Audit changes to user details in the last 7 days:
2095
2096 sql
2097 Copy code
2098 SELECT
2099     user_name,
2100     last_update_date,
2101     last_updated_by
2102 FROM
2103     fnd_user
2104 WHERE
2105     last_update_date >= SYSDATE - 7;
2106 List users linked to multiple employees:
2107
2108 sql
2109 Copy code
2110 SELECT

```



```

2111         user_name,
2112         employee_id
2113 FROM
2114     fnd_user
2115 GROUP BY
2116     user_name,
2117     employee_id
2118 HAVING
2119     COUNT(employee_id) > 1;
2120 Responsibilities and Role Management
2121 Get responsibilities assigned to users for a specific application:
2122
2123 sql
2124 Copy code
2125 SELECT
2126     fu.user_name,
2127     fr.responsibility_name,
2128     frg.start_date,
2129     frg.end_date
2130 FROM
2131     fnd_user fu
2132 JOIN
2133     fnd_user_resp_groups frg
2134 ON
2135     fu.user_id = frg.user_id
2136 JOIN
2137     fnd_responsibility fr
2138 ON
2139     frg.responsibility_id = fr.responsibility_id
2140 WHERE
2141     fr.application_id = 'APPLICATION_ID';
2142 Check users with responsibilities that have expired:
2143
2144 sql
2145 Copy code
2146 SELECT
2147     fu.user_name,
2148     fr.responsibility_name,
2149     frg.end_date
2150 FROM
2151     fnd_user fu
2152 JOIN
2153     fnd_user_resp_groups frg
2154 ON
2155     fu.user_id = frg.user_id
2156 JOIN
2157     fnd_responsibility fr
2158 ON
2159     frg.responsibility_id = fr.responsibility_id
2160 WHERE
2161     frg.end_date < SYSDATE;
2162 Find responsibilities assigned to multiple users:
2163
2164 sql
2165 Copy code
2166 SELECT
2167     responsibility_name,
2168     COUNT(user_id) AS user_count
2169 FROM
2170     fnd_user_resp_groups
2171 GROUP BY
2172     responsibility_name
2173 HAVING
2174     COUNT(user_id) > 1;
2175 Concurrent Programs and Requests
2176 List all concurrent programs created or modified in the last 6 months:
2177
2178 sql
2179 Copy code

```

```

2180 SELECT
2181     concurrent_program_name,
2182     description,
2183     last_update_date
2184 FROM
2185     fnd_concurrent_programs
2186 WHERE
2187     last_update_date >= SYSDATE - 180;
2188 Retrieve all scheduled concurrent requests for a specific user:
2189
2190 sql
2191 Copy code
2192 SELECT
2193     request_id,
2194     phase_code,
2195     status_code,
2196     requested_start_date
2197 FROM
2198     fnd_concurrent_requests
2199 WHERE
2200     requested_by = (SELECT user_id FROM fnd_user WHERE user_name = 'USERNAME');
2201 Identify requests that failed due to errors:
2202
2203 sql
2204 Copy code
2205 SELECT
2206     request_id,
2207     phase_code,
2208     status_code,
2209     log_file_name
2210 FROM
2211     fnd_concurrent_requests
2212 WHERE
2213     phase_code = 'COMPLETED'
2214 AND
2215     status_code = 'ERROR';
2216 Audit request groups containing custom concurrent programs:
2217
2218 sql
2219 Copy code
2220 SELECT
2221     frg.request_group_name,
2222     fcp.concurrent_program_name
2223 FROM
2224     fnd_request_group_units frgu
2225 JOIN
2226     fnd_request_groups frg
2227 ON
2228     frgu.request_group_id = frg.request_group_id
2229 JOIN
2230     fnd_concurrent_programs fcp
2231 ON
2232     frgu.request_unit_id = fcp.concurrent_program_id
2233 WHERE
2234     fcp.concurrent_program_name LIKE 'XX%';
2235 Retrieve historical runtime for a specific program:
2236
2237 sql
2238 Copy code
2239 SELECT
2240     request_id,
2241     actual_start_date,
2242     actual_completion_date,
2243     (actual_completion_date - actual_start_date) * 24 * 60 AS runtime_minutes
2244 FROM
2245     fnd_concurrent_requests
2246 WHERE
2247     concurrent_program_id = (SELECT concurrent_program_id FROM fnd_concurrent_programs
2248                             WHERE concurrent_program_name = 'PROGRAM_NAME');

```

```

2248 Data Security and Auditing
2249 List security profiles associated with a specific responsibility:
2250
2251 sql
2252 Copy code
2253 SELECT
2254     fr.responsibility_name,
2255     fp.security_profile_name
2256 FROM
2257     fnd_responsibility fr
2258 JOIN
2259     fnd_profile_options fpo
2260 ON
2261     fr.responsibility_id = fpo.level_id
2262 JOIN
2263     fnd_security_profiles fp
2264 ON
2265     fpo.profile_option_value = fp.security_profile_id
2266 WHERE
2267     fr.responsibility_name = 'RESPONSIBILITY_NAME';
2268 Find all profile options and their values for a specific user:
2269
2270 sql
2271 Copy code
2272 SELECT
2273     fpo.profile_option_name,
2274     fu.user_name,
2275     fpor.profile_option_value
2276 FROM
2277     fnd_profile_options fpo
2278 JOIN
2279     fnd_profile_option_responsibilities fpor
2280 ON
2281     fpo.profile_option_id = fpor.profile_option_id
2282 JOIN
2283     fnd_user fu
2284 ON
2285     fpor.level_value = fu.user_id
2286 WHERE
2287     fu.user_name = 'USERNAME';
2288 Track changes to responsibilities for a specific user:
2289
2290 sql
2291 Copy code
2292 SELECT
2293     fu.user_name,
2294     fr.responsibility_name,
2295     furg.creation_date,
2296     furg.last_update_date
2297 FROM
2298     fnd_user fu
2299 JOIN
2300     fnd_user_resp_groups_direct furg
2301 ON
2302     fu.user_id = furg.user_id
2303 JOIN
2304     fnd_responsibility fr
2305 ON
2306     furg.responsibility_id = fr.responsibility_id
2307 WHERE
2308     fu.user_name = 'USERNAME';
2309 People and Assignment Queries
2310 Fetch active employees with assigned responsibilities:
2311
2312 sql
2313 Copy code
2314 SELECT
2315     papf.person_id,
2316     papf.full_name,

```

```

2317         fr.responsibility_name
2318 FROM
2319     per_all_people_f papf
2320 JOIN
2321     fnd_user fu
2322 ON
2323     papf.person_id = fu.employee_id
2324 JOIN
2325     fnd_user_resp_groups frg
2326 ON
2327     fu.user_id = frg.user_id
2328 JOIN
2329     fnd_responsibility fr
2330 ON
2331     frg.responsibility_id = fr.responsibility_id
2332 WHERE
2333     papf.effective_end_date IS NULL;
2334 List terminated employees still linked to active user accounts:
2335
2336 sql
2337 Copy code
2338 SELECT
2339     fu.user_name,
2340     papf.full_name,
2341     papf.effective_end_date
2342 FROM
2343     fnd_user fu
2344 JOIN
2345     per_all_people_f papf
2346 ON
2347     fu.employee_id = papf.person_id
2348 WHERE
2349     papf.effective_end_date < SYSDATE
2350 AND
2351     NVL(fu.end_date, SYSDATE) > SYSDATE;
2352 Would you like more advanced queries, or help with any of these?
2353
2354
2355
2356
2357
2358
2359 You said:
2360 yes more advance querry questions
2361 ChatGPT said:
2362 ChatGPT
2363 Here are advanced query questions related to Oracle EBS for various operations, diving
2364 deeper into real-world use cases:
2365
2366 Advanced FND User Queries
2367 Identify users with overlapping responsibilities:
2368
2369 sql
2370 Copy code
2371 SELECT
2372     fu.user_name,
2373     fr.responsibility_name,
2374     COUNT(*)
2375 FROM
2376     fnd_user_resp_groups frg
2377 JOIN
2378     fnd_user fu
2379 ON
2380     frg.user_id = fu.user_id
2381 JOIN
2382     fnd_responsibility fr
2383 ON
2384     frg.responsibility_id = fr.responsibility_id
2385 WHERE

```

```

2385         frg.start_date <= NVL(frg.end_date, SYSDATE)
2386 GROUP BY
2387     fu.user_name,
2388     fr.responsibility_name
2389 HAVING
2390     COUNT(*) > 1;
2391 Track inactive users assigned to critical responsibilities:
2392
2393 sql
2394 Copy code
2395 SELECT
2396     fu.user_name,
2397     fr.responsibility_name,
2398     frg.start_date,
2399     frg.end_date
2400 FROM
2401     fnd_user fu
2402 JOIN
2403     fnd_user_resp_groups frg
2404 ON
2405     fu.user_id = frg.user_id
2406 JOIN
2407     fnd_responsibility fr
2408 ON
2409     frg.responsibility_id = fr.responsibility_id
2410 WHERE
2411     fu.end_date IS NOT NULL
2412 AND
2413     fr.responsibility_name IN ('SYSADMIN', 'SECURITY ADMINISTRATOR');
2414 Fetch user account lock/unlock activity logs:
2415
2416 sql
2417 Copy code
2418 SELECT
2419     fu.user_name,
2420     fl.message_text,
2421     fl.creation_date
2422 FROM
2423     fnd_logins fl
2424 JOIN
2425     fnd_user fu
2426 ON
2427     fl.user_id = fu.user_id
2428 WHERE
2429     fl.message_text LIKE '%Account%Locked%'
2430 OR
2431     fl.message_text LIKE '%Account%Unlocked%';
2432 Concurrent Program and Request Queries
2433 Fetch the top 10 longest-running concurrent requests:
2434
2435 sql
2436 Copy code
2437 SELECT
2438     request_id,
2439     concurrent_program_id,
2440     actual_start_date,
2441     actual_completion_date,
2442     ROUND((actual_completion_date - actual_start_date) * 1440, 2) AS runtime_minutes
2443 FROM
2444     fnd_concurrent_requests
2445 WHERE
2446     phase_code = 'C'
2447 AND
2448     status_code = 'C'
2449 ORDER BY
2450     runtime_minutes DESC
2451 FETCH FIRST 10 ROWS ONLY;
2452 List concurrent programs without request groups:
2453

```

```

2454 sql
2455 Copy code
2456 SELECT
2457     concurrent_program_name
2458 FROM
2459     fnd_concurrent_programs fcp
2460 WHERE
2461     NOT EXISTS (
2462         SELECT
2463             1
2464         FROM
2465             fnd_request_group_units frgu
2466         WHERE
2467             frgu.request_unit_id = fcp.concurrent_program_id
2468     );
2469 Identify failed requests for a specific program in the last 30 days:
2470
2471 sql
2472 Copy code
2473 SELECT
2474     request_id,
2475     log_file_name,
2476     phase_code,
2477     status_code
2478 FROM
2479     fnd_concurrent_requests
2480 WHERE
2481     concurrent_program_id = (SELECT concurrent_program_id FROM fnd_concurrent_programs
2482                             WHERE concurrent_program_name = 'PROGRAM_NAME')
2483 AND
2484     phase_code = 'C'
2485 AND
2486     status_code = 'E'
2487 AND
2488     requested_start_date >= SYSDATE - 30;
2489 Retrieve concurrent programs with a specific parameter value:
2490
2491 sql
2492 Copy code
2493 SELECT
2494     fcp.concurrent_program_name,
2495     fcp.description,
2496     fcr.argument_text
2497 FROM
2498     fnd_concurrent_requests fcr
2499 JOIN
2500     fnd_concurrent_programs fcp
2501 ON
2502     fcr.concurrent_program_id = fcp.concurrent_program_id
2503 WHERE
2504     fcr.argument_text LIKE '%VALUE%';
2505 Responsibility and Security
2506 Find responsibilities without any users assigned:
2507
2508 sql
2509 Copy code
2510 SELECT
2511     responsibility_name
2512 FROM
2513     fnd_responsibility
2514 WHERE
2515     NOT EXISTS (
2516         SELECT
2517             1
2518         FROM
2519             fnd_user_resp_groups_direct
2520         WHERE
2521             fnd_user_resp_groups_direct.responsibility_id =
2522             fnd_responsibility.responsibility_id

```

```

2521 );
2522 Audit changes to responsibility assignments in the last 90 days:
2523
2524 sql
2525 Copy code
2526 SELECT
2527     fu.user_name,
2528     fr.responsibility_name,
2529     frg.creation_date,
2530     frg.last_update_date
2531 FROM
2532     fnd_user_resp_groups_direct frg
2533 JOIN
2534     fnd_user fu
2535 ON
2536     frg.user_id = fu.user_id
2537 JOIN
2538     fnd_responsibility fr
2539 ON
2540     frg.responsibility_id = fr.responsibility_id
2541 WHERE
2542     frg.last_update_date >= SYSDATE - 90;
2543 Find users with expired responsibilities still active:
2544
2545 sql
2546 Copy code
2547 SELECT
2548     fu.user_name,
2549     fr.responsibility_name,
2550     frg.end_date
2551 FROM
2552     fnd_user_resp_groups_direct frg
2553 JOIN
2554     fnd_user fu
2555 ON
2556     frg.user_id = fu.user_id
2557 JOIN
2558     fnd_responsibility fr
2559 ON
2560     frg.responsibility_id = fr.responsibility_id
2561 WHERE
2562     NVL(frg.end_date, SYSDATE) < SYSDATE
2563 AND
2564     NVL(fu.end_date, SYSDATE) > SYSDATE;
2565 Employee and Person Queries
2566 Fetch users without employees linked:
2567
2568 sql
2569 Copy code
2570 SELECT
2571     user_name
2572 FROM
2573     fnd_user
2574 WHERE
2575     employee_id IS NULL;
2576 Identify terminated employees still associated with active responsibilities:
2577
2578 sql
2579 Copy code
2580 SELECT
2581     papf.full_name,
2582     papf.person_id,
2583     fu.user_name,
2584     fr.responsibility_name
2585 FROM
2586     per_all_people_f papf
2587 JOIN
2588     fnd_user fu
2589 ON

```

```

2590     papf.person_id = fu.employee_id
2591 JOIN
2592     fnd_user_resp_groups frg
2593 ON
2594     fu.user_id = frg.user_id
2595 JOIN
2596     fnd_responsibility fr
2597 ON
2598     frg.responsibility_id = fr.responsibility_id
2599 WHERE
2600     papf.effective_end_date < SYSDATE
2601 AND
2602     NVL(fu.end_date, SYSDATE) > SYSDATE;
2603 Fetch employees with roles across multiple responsibilities:
2604
2605 sql
2606 Copy code
2607 SELECT
2608     papf.person_id,
2609     papf.full_name,
2610     fr.responsibility_name,
2611     COUNT(*) AS responsibility_count
2612 FROM
2613     per_all_people_f papf
2614 JOIN
2615     fnd_user fu
2616 ON
2617     papf.person_id = fu.employee_id
2618 JOIN
2619     fnd_user_resp_groups frg
2620 ON
2621     fu.user_id = frg.user_id
2622 JOIN
2623     fnd_responsibility fr
2624 ON
2625     frg.responsibility_id = fr.responsibility_id
2626 GROUP BY
2627     papf.person_id,
2628     papf.full_name,
2629     fr.responsibility_name
2630 HAVING
2631     COUNT(*) > 1;
2632 Advanced Logging and Debugging
2633 Retrieve failed concurrent requests with log file details:
2634
2635 sql
2636 Copy code
2637 SELECT
2638     fcr.request_id,
2639     fcp.concurrent_program_name,
2640     fcr.log_file_name,
2641     fcr.phase_code,
2642     fcr.status_code
2643 FROM
2644     fnd_concurrent_requests fcr
2645 JOIN
2646     fnd_concurrent_programs fcp
2647 ON
2648     fcr.concurrent_program_id = fcp.concurrent_program_id
2649 WHERE
2650     fcr.phase_code = 'C'
2651 AND
2652     fcr.status_code = 'E';
2653 Track changes to system profile values in the past 30 days:
2654
2655 sql
2656 Copy code
2657 SELECT
2658     fpo.profile_option_name,

```



```

2659         fps.level_value,
2660         fps.profile_option_value,
2661         fps.last_update_date,
2662         fps.last_updated_by
2663 FROM
2664     fnd_profile_option_values fps
2665 JOIN
2666     fnd_profile_options fpo
2667 ON
2668     fps.profile_option_id = fpo.profile_option_id
2669 WHERE
2670     fps.last_update_date >= SYSDATE - 30; /* Formatted on 12/18/2024 12:07:52 PM (QP5
        v5.163.1008.3004) */
2671 SELECT *
2672 FROM fnd_user
2673 here;
2674
2675 SELECT *
2676     FROM fnd_user
2677     WHERE user_name = 'OPERATIONS';
2678
2679 SELECT *
2680     FROM fnd_user
2681     WHERE user_id = 1318;
2682
2683 --1006
2684 SELECT * FROM PER_ALL_PEOPLE_F
2685 WHERE PERSON_ID = 10061;
2686
2687 SELECT SYSDATE FROM DUAL;
2688
2689
2690 SELECT * FROM operations;
2691
2692 SELECT *
2693     FROM fnd_user
2694     WHERE user_name = 'ADITYA'
2695           AND TRIM (SYSDATE) BETWEEN start_date AND end_date;
2696
2697 SELECT *
2698     FROM fnd_user
2699     WHERE user_name = 'OPERATIONS'
2700           AND TRIM (SYSDATE) BETWEEN start_date AND NVL (end_date, SYSDATE + 1);
2701
2702 SELECT *
2703     FROM PER_ALL_PEOPLE_F;
2704
2705 --WHERE PERSON_ID = 25;
2706
2707 SELECT * FROM per_all_people_f;
2708
2709 SELECT *
2710     FROM PER_ALL_PEOPLE_F
2711     WHERE PERSON_ID = 25
2712           AND TRUNC (SYSDATE) BETWEEN EFFECTIVE_START_DATE
2713                                     AND EFFECTIVE_END_DATE
2714           AND (CURRENT_EMPLOYEE_FLAG = 'Y' OR CURRENT_NPW_FLAG = 'Y');
2715
2716
2717 /*****
2718      *
2719      *****/
2720
2721 SELECT
2722     fu.user_id,
2723     fu.user_name user_name,
2724     fu.start_date,
2725     fu.end_date,
2726     fu.description,

```

```

2727         papf.person_id,
2728         NVL (papf.EMPLOYEE_NUMBER, papf.NPW_NUMBER) employee_id,
2729         papf.full_name,
2730         papf.EFFECTIVE_START_DATE,
2731         papf.EFFECTIVE_END_DATE,
2732         papf.CURRENT_EMPLOYEE_FLAG,
2733         papf.CURRENT_NPW_FLAG,
2734         papf.EMAIL_ADDRESS
2735 FROM fnd_user fu, per_all_people_f papf
2736 WHERE
2737         fu.employee_id = papf.person_id
2738         AND user_name = 'ADITYA'
2739         AND TRIM (SYSDATE) BETWEEN fu.start_date
2740                                AND NVL (fu.end_date, SYSDATE + 1)
2741         AND TRIM (SYSDATE) BETWEEN papf.EFFECTIVE_START_DATE
2742                                AND papf.EFFECTIVE_END_DATE
2743         AND (papf.CURRENT_EMPLOYEE_FLAG = 'Y' OR papf.CURRENT_NPW_FLAG = 'Y');
2744
2745
2746
2747     SELECT START_DATE,
2748            END_DATE,
2749            RESPONSIBILITY_APPLICATION_ID,
2750            RESPONSIBILITY_ID,
2751            SECURITY_GROUP_ID,
2752            LAST_UPDATE_DATE,
2753            LAST_UPDATED_BY,
2754            CREATED_BY,
2755            CREATION_DATE,
2756            LAST_UPDATE_LOGIN,
2757            USER_ID
2758     FROM FND_USER_RESP_GROUPS_direct
2759     WHERE (responsibility_id, responsibility_application_id) IN
2760            (SELECT responsibility_id, application_id
2761             FROM fnd_responsibility
2762             WHERE USER_ID = 1014243)
2763     ORDER BY responsibility_application_id, responsibility_id, security_group_id;
2764
2765
2766     --*****
2767
2768
2769
2770     SELECT START_DATE,
2771            END_DATE,
2772            RESPONSIBILITY_APPLICATION_ID,
2773            RESPONSIBILITY_ID,
2774            SECURITY_GROUP_ID,
2775            LAST_UPDATE_DATE,
2776            LAST_UPDATED_BY,
2777            CREATED_BY,
2778            CREATION_DATE,
2779            LAST_UPDATE_LOGIN,
2780            USER_ID
2781     FROM FND_USER_RESP_GROUPS_direct
2782     WHERE (responsibility_id, responsibility_application_id) IN
2783            (SELECT responsibility_id, application_id
2784             FROM fnd_responsibility
2785             WHERE ( version = '4'
2786                   OR version = 'W'
2787                   OR version = 'M'
2788                   OR version = 'H'))
2789            -- AND (USER_ID = 1318)
2790     ORDER BY responsibility_application_id, responsibility_id, security_group_id;
2791
2792     select * from dba_objects where object_type LIKE '%TABLE%';
2793
2794
2795     SELECT responsibility_id, application_id, version

```

```

2796             FROM fnd_responsibility
2797             WHERE (   version = '4'
2798                    OR version = 'W'
2799                    OR version = 'M'
2800                    OR version = 'H');
2801
2802
2803 select * from dba_objects where object_type LIKE '%TABLE%';
2804
2805 select * from dba_objects where object_NAME LIKE '%FND%RES%VL%'; /*for language*/
2806
2807 select * from dba_objects where object_NAME LIKE '%FND%RES%TL%';
2808
2809 select * from dba_objects where object_NAME LIKE '%FND%APP%VL%';
2810
2811 select * from dba_objects where object_NAME LIKE '%FND%USER%RESP%';
2812
2813 select *
2814 from FND_USER_RESP_GROUPS;
2815
2816 SELECT * FROM FND_RESPONSIBILITY;
2817 SELECT * FROM FND_USER_RESP_GROUPS_direct;
2818 --where user_id = 1014254
2819 --order by last_update_date desc;
2820 SELECT * FROM FND_RESPONSIBILITY_VL;
2821 SELECT * FROM FND_APPLICATION_VL;
2822 select *
2823 from fnd_user
2824 where user_name = 'DEV';
2825
2826 fnd_user_resp_groups_api
2827
2828 xxqgen_user_data_pkg_ ;
2829
2830 --FND_APPLICATION_VL
2831 select * from fnd_responsibility;
2832
2833 SELECT
2834     fu.user_id,
2835     fu.user_name user_name,
2836     fu.start_date,
2837     fu.end_date,
2838     fu.description,
2839     papf.person_id,
2840     NVL (papf.EMPLOYEE_NUMBER, papf.NPW_NUMBER) employee_id,
2841     papf.full_name,
2842     papf.EFFECTIVE_START_DATE,
2843     papf.EFFECTIVE_END_DATE,
2844     papf.CURRENT_EMPLOYEE_FLAG,
2845     papf.CURRENT_NPW_FLAG,
2846     papf.EMAIL_ADDRESS,
2847     frv.RESPONSIBILITY_NAME,
2848     fav.APPLICATION_NAME,
2849     frv.RESPONSIBILITY_KEY,
2850     frv.DESCRPTION,
2851     frv.START_DATE
2852 FROM
2853     fnd_user fu,
2854     per_all_people_f papf,
2855     FND_USER_RESP_GROUPS_direct furg,
2856     FND_RESPONSIBILITY_VL frv,
2857     FND_APPLICATION_VL fav
2858 where
2859     furg.responsibility_id = frv.RESPONSIBILITY_ID
2860     and fu.user_id=furg.user_id
2861     and frv.APPLICATION_ID=fav.APPLICATION_ID
2862     and fu.employee_id = papf.person_id
2863     -- and user_name = 'DEV'
2864     AND TRIM (SYSDATE) BETWEEN fu.start_date

```

```

2865             AND NVL (fu.end_date, SYSDATE + 1)
2866         AND TRIM (SYSDATE) BETWEEN papf.EFFECTIVE_START_DATE
2867             AND papf.EFFECTIVE_END_DATE
2868         AND (papf.CURRENT_EMPLOYEE_FLAG = 'Y' OR papf.CURRENT_NPW_FLAG = 'Y');
2869
2870     ;
2871
2872     select * from fnd_user;
2873
2874     select * from dba_objects where object_name like '%fnd%';
2875
2876     select * from all_objects where object_name like '%fnd%';
2877
2878     select *
2879     from fnd_user_resp_groups;
2880
2881     select *
2882     from FND_RESPONSIBILITY_VL frv;
2883
2884     select *
2885     from fnd_concurrent_program_vl;
2886
2887     SELECT *
2888     FROM FND_APPLICATION_VL
2889     WHERE APPLICATION_ID = 201;
2890
2891     SELECT * FROM FND_CONCURRENT_PROGRAMS_VL
2892     WHERE USER_CONCURRENT_PROGRAM_NAME like 'XXQGEN%DK';
2893
2894     SELECT * FROM FND_EXECUTABLES_FORM_V
2895     WHERE EXECUTABLE_ID = 16642;
2896
2897
2898     SELECT * FROM FND_CONCURRENT_REQUESTS
2899     WHERE 1=1;
2900     -- AND REQUEST_ID =5912035 -- 5912104
2901     --AND CONCURRENT_PROGRAM_ID = 218704;
2902
2903
2904     -- PHASE
2905     SELECT LOOKUP_CODE, MEANING
2906     FROM FND_LOOKUP_VALUES
2907     WHERE LOOKUP_TYPE = 'CP_PHASE_CODE' AND LANGUAGE = 'US'
2908     AND ENABLED_FLAG = 'Y';
2909
2910     --STATUS
2911     SELECT LOOKUP_CODE, MEANING
2912     FROM FND_LOOKUP_VALUES
2913     WHERE LOOKUP_TYPE = 'CP_STATUS_CODE' AND LANGUAGE = 'US'
2914     AND ENABLED_FLAG = 'Y';
2915
2916     SELECT DISTINCT
2917     UPCVL.CONCURRENT_PROGRAM_ID,
2918     UPCVL.CONCURRENT_PROGRAM_NAME,
2919     UPCVL.EXECUTABLE_APPLICATION_ID,
2920     UPCVL.EXECUTABLE_ID,
2921     FEFV.EXECUTION_FILE_NAME,
2922     FEFV.DESRIPTION,
2923     UPCVL.APPLICATION_ID,
2924     FAVL.APPLICATION_NAME,
2925     FAVL.APPLICATION_SHORT_NAME,
2926     FCR.REQUEST_ID,
2927     SC.MEANING AS STATUS_CODE,
2928     PC.MEANING AS PHASE_CODE,
2929     FCR.ACTUAL_COMPLETION_DATE - FCR.ACTUAL_START_DATE AS EXECUTION_DURATION
2930 FROM FND_CONCURRENT_PROGRAMS_VL UPCVL
2931 JOIN FND_APPLICATION_VL FAVL
2932 ON UPCVL.APPLICATION_ID = FAVL.APPLICATION_ID
2933 JOIN FND_EXECUTABLES_FORM_V FEFV

```

```

2934         ON UPCVL.EXECUTABLE_ID = FEFV.EXECUTABLE_ID
2935 JOIN      FND_CONCURRENT_REQUESTS FCR
2936         ON UPCVL.CONCURRENT_PROGRAM_ID = FCR.CONCURRENT_PROGRAM_ID
2937 JOIN      (SELECT LOOKUP_CODE, MEANING
2938           FROM FND_LOOKUP_VALUES
2939           WHERE LOOKUP_TYPE = 'CP_STATUS_CODE'
2940             AND LANGUAGE = 'US'
2941             AND ENABLED_FLAG = 'Y') SC
2942         ON SC.LOOKUP_CODE = FCR.STATUS_CODE
2943 JOIN      (SELECT LOOKUP_CODE, MEANING
2944           FROM FND_LOOKUP_VALUES
2945           WHERE LOOKUP_TYPE = 'CP_PHASE_CODE'
2946             AND LANGUAGE = 'US'
2947             AND ENABLED_FLAG = 'Y') PC
2948         ON PC.LOOKUP_CODE = FCR.PHASE_CODE
2949 WHERE     UPCVL.CONCURRENT_PROGRAM_NAME = 'XXQGEN_USER_DATA_PGK_DK';
2950
2951
2952 SELECT DISTINCT UPCVL.CONCURRENT_PROGRAM_ID,
2953                UPCVL.CONCURRENT_PROGRAM_NAME,
2954                UPCVL.EXECUTABLE_APPLICATION_ID,
2955                UPCVL.EXECUTABLE_ID,
2956                FEFV.EXECUTION_FILE_NAME,
2957                FEFV.DESCRPTION,
2958                UPCVL.APPLICATION_ID,
2959                FAVL.APPLICATION_NAME,
2960                FAVL.APPLICATION_SHORT_NAME ,
2961                FCR.REQUEST_ID,
2962                SC.MEANING STATUS_CODE,
2963                PC.MEANING PHASE_CODE,
2964                FCR.ACTUAL_COMPLETION_DATE - FCR.ACTUAL_START_DATE
2965 FROM      FND_CONCURRENT_PROGRAMS_VL UPCVL,
2966          FND_APPLICATION_VL FAVL,
2967          FND_EXECUTABLES_FORM_V FEFV,
2968          FND_CONCURRENT_REQUESTS FCR,
2969          (SELECT LOOKUP_CODE, MEANING
2970            FROM FND_LOOKUP_VALUES
2971            WHERE LOOKUP_TYPE = 'CP_STATUS_CODE' AND LANGUAGE = 'US'
2972              AND ENABLED_FLAG = 'Y') SC,
2973          (SELECT LOOKUP_CODE, MEANING
2974            FROM FND_LOOKUP_VALUES
2975            WHERE LOOKUP_TYPE = 'CP_PHASE_CODE' AND LANGUAGE = 'US'
2976              AND ENABLED_FLAG = 'Y') PC
2977 WHERE 1=1
2978 AND    UPCVL.CONCURRENT_PROGRAM_NAME = 'XXQGEN%'
2979 AND    UPCVL.CONCURRENT_PROGRAM_ID = FCR.CONCURRENT_PROGRAM_ID
2980 AND    UPCVL.APPLICATION_ID = FAVL.APPLICATION_ID
2981 AND    UPCVL.EXECUTABLE_ID = FEFV.EXECUTABLE_ID
2982 AND    SC.LOOKUP_CODE = FCR.STATUS_CODE
2983 AND    PC.LOOKUP_CODE = FCR.PHASE_CODE;
2984
2985
2986
2987 SELECT DISTINCT UPCVL.CONCURRENT_PROGRAM_ID,
2988                UPCVL.CONCURRENT_PROGRAM_NAME,
2989                UPCVL.EXECUTABLE_APPLICATION_ID,
2990                UPCVL.EXECUTABLE_ID,
2991                FEFV.EXECUTION_FILE_NAME,
2992                FEFV.DESCRPTION,
2993                UPCVL.APPLICATION_ID,
2994                FAVL.APPLICATION_NAME,
2995                FAVL.APPLICATION_SHORT_NAME ,
2996                FCR.REQUEST_ID
2997 FROM      FND_CONCURRENT_PROGRAMS_VL UPCVL,
2998          FND_APPLICATION_VL FAVL,
2999          FND_EXECUTABLES_FORM_V FEFV,
3000          FND_CONCURRENT_REQUESTS FCR
3001 WHERE 1=1
3002 AND    UPCVL.CONCURRENT_PROGRAM_NAME = 'XXQGEN_USER_DATA_PGK_DK'

```

```

3003 AND      UPCVL.CONCURRENT_PROGRAM_ID = FCR.CONCURRENT_PROGRAM_ID
3004 AND      UPCVL.APPLICATION_ID = FAVL.APPLICATION_ID
3005 AND      UPCVL.EXECUTABLE_ID = FEFV.EXECUTABLE_ID;
3006
3007
3008 SELECT *
3009 FROM FND_CONCURRENT_REQUESTS
3010 WHERE CONCURRENT_PROGRAM_ID = 218711;
3011
3012 -- FND_CONCURRENT_PROGRAMS_TL T, FND_CONCURRENT_PROGRAMS B
3013 SELECT *
3014 FROM FND_CONCURRENT_PROGRAMS;
3015
3016 SELECT *
3017 FROM FND_CONCURRENT_PROGRAMS_TL;
3018
3019 SELECT *
3020 FROM V$INSTANCE;
3021
3022 SELECT *
3023 FROM ICX_PARAMETERS;
3024
3025
3026
3027
3028 SELECT fcpv.application_id,
3029        fcpv.concurrent_program_name,
3030        fcpv.last_update_date,
3031        fcpv.executable_application_id,
3032        fcpv.executable_id,
3033        FCPV.USER_CONCURRENT_PROGRAM_NAME,
3034        fefv.EXECUTION_FILE_NAME,
3035        fefv.description,
3036        fefv.application_name,
3037        FCPV.EXECUTABLE_APPLICATION_ID,
3038        fcr.request_id,
3039        fcr.last_updated_by,
3040        fcr.STATUS_CODE,
3041        fcr.PHASE_CODE,
3042        fcr.PP_END_DATE,
3043        fcr.PP_START_DATE,
3044        fcr.EXP_DATE,
3045        fcr.CRM_TSTMP,
3046        fcr.ACTUAL_COMPLETION_DATE,
3047        fcr.ACTUAL_START_DATE,
3048        fcr.NUMBER_OF_ARGUMENTS,
3049        fcr.RESPONSIBILITY_ID,
3050        fcr.RESPONSIBILITY_APPLICATION_ID,
3051        fcr.HOLD_FLAG
3052 FROM      FND_CONCURRENT_PROGRAMS_VL fcpv,
3053        FND_EXECUTABLES_FORM_V FEFV,
3054        FND_CONCURRENT_REQUESTS FCR
3055 WHERE USER_CONCURRENT_PROGRAM_NAME like 'XXQGEN%DK'
3056 and      fcpv.executable_id = fefv.executable_id
3057 and      FCPV.CONCURRENT_PROGRAM_ID = FCR.CONCURRENT_PROGRAM_ID;
3058
3059 select *
3060 from fnd_executables_form_v;
3061
3062 select *
3063 from FND_CONCURRENT_REQUESTS;
3064
3065 select *
3066 from fnd_concurrent_programs_vl
3067 where user_concurrent_program_name like 'XXQGEN%DK';
3068
3069
3070 select *
3071 from XDO_DS_DEFINITIONS_B

```

```
3072  where data_source_code like '%dk';
3073
3074  select *
3075  from XDO_DS_DEFINITIONS_TL
3076  where data_source_code like '%dk'
3077  order by creation_date desc;
3078
3079  SELECT
3080  FROM FND_USER FU , PER_ALL_PEOPLE_F PAPF
3081  WHERE FU.EMPLOYEE_ID = PAPF.PERSON_ID
3082  AND FU.USER_NAME ='OPERATIONS';
3083
3084
3085
3086  select fu.user_name, fu.creation_date, papf.full_name, papf.sex, papf.nationality,
3087  papf.start_date, fu.end_date
3087  from fnd_user fu, per_all_people_f papf
3088  where fu.employee_id = papf.person_id
3089  and    fu.user_name = 'OPERATIONS';
```