🡺 **To display table size:**

Select owner as "Schema", segment\_name as "Object Name", segment\_type as "Object Type", round(bytes/1024/1024,2) as "Object Size (Mb)", tablespace\_name as "Tablespace"

from dba\_segments;

Optional: [where] [group by] [ order by]

🡺 **To display all tables with table counts and table space name in one server:**

select \* from all\_tables;

Optional: [where] [group by] [ order by]

🡺 **To display packages, procedures, functions and indexes:**

select \* from user\_identifiers;

Optional: [where] [group by] [ order by]

🡺 **To display packages, procedures, functions and indexes for current user:**

select \* from user\_objects;

Optional: [where] [group by] [ order by]

🡺 **To display packages, procedures, functions and indexes in one serer:**

select \* from dba\_procedures;

Optional: [where] [group by] [ order by]

🡺 **To display packages, procedures, functions and indexes for current user:**

select \* from user\_procedures;

Optional: [where] [group by] [ order by]

🡺 **To display free space for table spaces:**

SELECT \* FROM USER\_FREE\_SPACE

Optional: [where] [group by] [ order by]

🡺 **To display free space for table spaces:**

SELECT TABLESPACE\_NAME, TO\_CHAR(SUM(NVL(BYTES,0))/1024/1024/1024, '99,999,990.99') AS “FREE\_BYTES in GB” FROM USER\_FREE\_SPACE

GROUP BY TABLESPACE\_NAME;

Optional: [where] [group by] [ order by]

🡺 **To display the list of tablespaces:**

select distinct tablespace\_name from dba\_tablespaces;

🡺 **To check maximum string length:**

select id, (max(vsize(column\_name))) cnt from TABLE\_NAME group by

id having ↓ ↓ Example (column name) Example (column name)

(max(vsize(column\_name)))>50 order by cnt desc;

↓

Example (length of string)

🡺 **To delete internal duplicates in a table:**

delete from TABLE\_NAME where rowid in (select min(rowid) from TABLE\_NAME

group by <Main columns list> having count (\*)>1);

🡺 **To export in csv format in POSTGRES:**

EX:

copy (SELECT \* FROM targets.locus\_ref) to 'D:\Internal1\MONTHLY\targets\_locus\_ref.csv' DELIMITER ',' CSV HEADER;

copy (SELECT \* FROM persons) to 'C:\tmp\persons\_client.csv' with csv

NOTE: PERMISSION DENIED

🡺 Ascii values query:

begin

for i in 1..400 loop

dbms\_output.put\_line(chr(i) || ' = ' || i);

end loop;

end;

select ascii('°') from dual

select to\_char('1'||chr(186)||chr(176)) from dual;

select to\_char('1'||chr(174)||chr(176)) from dual;

**🡺INDEX DETAILS:**

select distinct tablespace\_name

from all\_indexes

where owner = 'GOSTAR\_REVAMP'

🡺 TRIGGER:

TRIGGER structure\_details\_chk\_trg

BEFORE INSERT OR UPDATE

ON structure\_details

FOR EACH ROW

BEGIN

IF (:NEW.compound\_status = 'DISCOVERY' AND :NEW.cmpd\_status <> 1)

OR (:NEW.compound\_status = 'DEVELOPMENT' AND :NEW.cmpd\_status <> 2)

OR (:NEW.compound\_status = 'DRUG' AND :NEW.cmpd\_status <> 3)

OR (:NEW.compound\_status = 'AGRI' AND :NEW.cmpd\_status <> 4)

THEN

raise\_application\_error (-20010, 'Please check CMPD\_STATUS value');

ELSE

IF :NEW.SUB\_SMILES IS NOT NULL AND (:NEW.MOL\_FORMULA IS NULL

OR :NEW.MOL\_WEIGHT IS NULL) THEN

raise\_application\_error (-20020, 'MOL\_FORMULA/MOL\_WEIGHT should be present');

END IF;

END IF;

END structure\_details\_chk\_trg;

🡺PROCEDURE for export csv files:

CREATE OR REPLACE PROCEDURE export\_sdf\_smiles (dir VARCHAR2, file\_name VARCHAR2)

IS

FILE UTL\_FILE.file\_type;

*-- CURSOR cur is select GVK\_ID,SMILES SMILES FROM ||t\_name;*

CURSOR cur

IS

SELECT gvk\_id, sub\_smiles

FROM structure\_details

WHERE sub\_smiles IS NOT NULL AND database\_name = '' || file\_name || '';

BEGIN

*-- Open the file for writing*

FILE := UTL\_FILE.fopen (UPPER (dir), file\_name || '.txt', 'w', 32767);

*-- Export rows one by one*

BEGIN

FOR rec IN cur

LOOP

*-- All columns were concatenated into single value in SELECT*

UTL\_FILE.put\_line (FILE, rec.gvk\_id || CHR (9) || rec.sub\_smiles);

END LOOP;

UTL\_FILE.fclose (FILE);

END;

END;

/

🡺EXECUTE:

Begin

export\_tox\_smiles ('smiles\_data');

END;

/

**select** gvk\_id,**count**(\*),**count**(**distinct** assay\_type),**array\_agg**(**distinct** assay\_type) **from** gostarui.all\_activity\_gostar **where** assay\_type **in** ('B','FC') **group** **by** gvk\_id **having** **count**(**distinct** assay\_type)>1;

)

**select** gvk\_id,**count**(\*),**count**(**distinct** assay\_type),**json\_agg**(**distinct** assay\_type) **from** gostarui.all\_activity\_gostar **where** assay\_type **in** ('B','FC') **group** **by** gvk\_id **having** **count**(**distinct** assay\_type)>1;

POSTGRES:

ALTER TABLE reference DISABLE TRIGGER ALL;

ALTER TABLE reference ENABLE TRIGGER ALL;

POSTGRES:

**select** gvk\_id, **count** (\*), **count**(**distinct** assay\_type),**array\_agg**(**distinct** assay\_type) **from** gostarui.all\_activity\_gostar **where** assay\_type **in** ('B','FC') **group** **by** gvk\_id **having** **count**(**distinct** assay\_type)>1;

)

**select** gvk\_id,**count**(\*),**count**(**distinct** assay\_type),**json\_agg**(**distinct** assay\_type) **from** gostarui.all\_activity\_gostar **where** assay\_type **in** ('B','FC') **group** **by** gvk\_id **having** **count**(**distinct** assay\_type)>1;

To map the data type in postgres :

Update table\_name set col1=col1::<data\_type>;

Ex:

Update act\_uom\_check\_data set activity\_value= activity\_value1::float8;

**↓**

Varchar

**Counts:**

navakanth.suram@HYD6ODC3060D /cygdrive/d/Internal1/MONTHLY/Taxdump/new\_taxdump

$ wc -l \*.dmp

Enter Space Query:

Ex:

select ASSAY\_ID,RADIO\_LIGAND\_UOM from A\_ACTIVITY\_ASSAY where length(RADIO\_LIGAND\_UOM)-length(replace(RADIO\_LIGAND\_UOM,chr(13)||chr(10) ,''))>0;

update DRUG\_INFORMATION\_MASTER set STD\_HIGHEST\_PHASE\_ALL=trim(replace(STD\_HIGHEST\_PHASE\_ALL,chr(13)||chr(10) ,' ')) where GVK\_ID in (30117788,30109142);

OPERATOR classes in POSTGRESQL:

**SELECT** am.amname **AS** index\_method,

opc.opcname **AS** opclass\_name

**FROM** pg\_am am, pg\_opclass opc

**WHERE** opc.opcmethod = am.**oid**

**ORDER** **BY** index\_method, opclass\_name;

**ALTER** **INDEX** targets.AAG\_gvk\_id **RENAME** **TO** AAG\_ASSAY\_TYPES\_IDX

Ex:

**select** **count**(gvk\_id) **from** targets.AAG\_ASSAY\_TYPE\_BFC\_A **where** assay\_types @> '{B,FC}';

**Array Functions and Operators**

**Array Operators:**

| **Operator** | **Description** | **Example** | **Result** |
| --- | --- | --- | --- |
| = | **equal** | ARRAY[1.1,2.1,3.1]::int[] = ARRAY[1,2,3] | t |
| <> | **not equal** | ARRAY[1,2,3] <> ARRAY[1,2,4] | t |
| < | **less than** | ARRAY[1,2,3] < ARRAY[1,2,4] | t |
| > | **greater than** | ARRAY[1,4,3] > ARRAY[1,2,4] | t |
| <= | **less than or equal** | ARRAY[1,2,3] <= ARRAY[1,2,3] | t |
| >= | **greater than or equal** | ARRAY[1,4,3] >= ARRAY[1,4,3] | t |
| @> | **contains** | ARRAY[1,4,3] @> ARRAY[3,1] | t |
| <@ | **is contained by** | ARRAY[2,7] <@ ARRAY[1,7,4,2,6] | t |
| && | **overlap (have elements in common)** | ARRAY[1,4,3] && ARRAY[2,1] | t |
| || | **array-to-array concatenation** | ARRAY[1,2,3] || ARRAY[4,5,6] | {1,2,3,4,5,6} |
| || | **array-to-array concatenation** | ARRAY[1,2,3] || ARRAY[[4,5,6],[7,8,9]] | {{1,2,3},{4,5,6},{7,8,9}} |
| || | **element-to-array concatenation** | 3 || ARRAY[4,5,6] | {3,4,5,6} |
| || | **array-to-element concatenation** | ARRAY[4,5,6] || 7 | {4,5,6,7} |

Operator Description Example

~ Matches regular expression, case sensitive

~\* Matches regular expression, case insensitive

!~ Does not match regular expression, case sensitive

!~\* Does not match regular expression, case insensitive

SELECT

   COLUMN\_NAME

FROM

   information\_schema.COLUMNS;

table\_schema

table\_name

column\_name

change column datatype:

ALTER TABLE assets ALTER COLUMN name TYPE VARCHAR;

JCHEM cd cartridge

In oracle:

## select ref\_id,reference,count(\*)

## from gostarui.all\_activity\_gostar

## where assay\_type in ('A','D','M','E','ESL','ELP','ELD','EKA')

## group by ref\_id,reference

## having count(distinct assay\_type)>1;

Postgres:

**Assay\_types:**

**select** ref\_id,reference,**count**(\*),**count**(**distinct** assay\_type),**array\_agg**(**distinct** assay\_type) **from** gostarui.all\_activity\_gostar **where** assay\_type **in** ('A','D','M','E','ESL','ELP','ELD','EKA') **group** **by** ref\_id,reference **having** **count**(**distinct** assay\_type)>1;

**select** a.ref\_id,a.reference,**count**(\*),**count**(**distinct** b.assay\_type),**array\_agg**(**distinct** b.assay\_type)

**from** gostarui.reference\_master a,gostarui.all\_activity\_gostar b

**where** b.assay\_type **in** ('A','D','M','E','T') **and** a.reference **like** '%2018%' **and** a.ref\_id=b.ref\_id

**group** **by** a.ref\_id,a.reference

**having** **count**(**distinct** b.assay\_type)>1;

**select** a.ref\_id,a.reference,a.ref\_type,**count**(\*),**count**(**distinct** b.assay\_type),**unnest**(**array\_agg**(**distinct** b.assay\_type))

**from** gostarui.reference\_master a,gostarui.all\_activity\_gostar b

**where** b.assay\_type **in** ('A','D','M','E','T') **and** a.reference **like** '%2018%' **and** a.ref\_id=b.ref\_id **and** a.ref\_type **in** ('J','P')

**group** **by** a.ref\_id,a.reference

**having** **count**(**distinct** b.assay\_type)>1;

<https://pubchem.ncbi.nlm.nih.gov/edit3/index.html>

select \* from DBA\_DATA\_FILES;

select DISTINCT a.Task\_ID,a.curator\_id,a.QC\_ID,b.USER\_ID,c.USER\_NAME,b.TAsk\_id,e. USER\_NAME,C.USER\_ID

from TASK\_ALLOCATION a,TASK\_STATUS b, USER\_MASTER c,Z\_ACT\_TYPE\_UOM\_REFS\_COMPLTD d,USER\_MASTER e

where a.task\_id = d.task\_id and a.TASK\_ID=b.TASK\_ID and a.curator\_id=c.user\_id and a.QC\_ID=e.USER\_ID;

SELECT distinct UM.USER\_NAME,RM.REF\_TYPE,COUNT(distinct T.GVK\_ID) cnt\_gvk\_id,COUNT(distinct aag.act\_ID) cnt\_act\_id,COUNT(distinct aag.assay\_ID) cnt\_assay\_id,COUNT(distinct aag.REF\_ID)

FROM

TASK\_MASTER TM,

TASK\_ALLOCATION TA,

USER\_MASTER UM,

TASK\_STATUS G,

REF\_GVK T,

all\_activity\_gostar aag,

REFERENCE\_MASTER RM

WHERE

TM.TASK\_ID=TA.TASK\_ID

AND TA.CURATOR\_ID=UM.USER\_ID

AND TA.TASK\_ALLOCATION\_ID=G.TASK\_ALLOCATION\_ID

AND TM.REF\_ID=T.REF\_ID

AND (G.END\_DATE BETWEEN to\_date('01-JAN-19','DD-MON-YY') AND to\_date('10-SEP-19','DD-MON-YY'))

AND (G.START\_DATE BETWEEN to\_date('01-JAN-19','DD-MON-YY') AND to\_date('10-SEP-19','DD-MON-YY'))

and TM.REF\_ID=aag.REF\_ID and RM.REF\_ID=aag.REF\_ID and aag.REF\_ID=T.REF\_ID and upper(USER\_NAME) ='AYANA.RAJ'

GROUP BY UM.USER\_NAME,RM.REF\_TYPE;

**Commit for every row:**

declare

l\_counter PLS\_INTEGER;

commit\_after\_in PLS\_INTEGER;

cursor c1

is

select \*

from THERAPEUTIC\_AREA

where THERAPEUTIC\_USE in (select DRUG\_USE from DRUG\_USE where ICD10\_CODE is null);

begin

l\_counter := 1;

for i in c1

loop

update DRUG\_USE

set ICD10\_CODE=i.ICD10\_CODE

where DRUG\_USE=i.THERAPEUTIC\_USE and ICD10\_CODE is null;

IF l\_counter >= commit\_after\_in

THEN

COMMIT;

l\_counter := 1;

ELSE

l\_counter := l\_counter + 1;

END IF;

end loop;

commit;

end;

/

Count:

Semicolons in a column :

SELECT REGEXP\_COUNT(HIGHEST\_PHASE\_ALL,';') REGEXP\_COUNT

FROM drug\_information\_master;

Maximum semicolons in a column :

SELECT max(REGEXP\_COUNT(HIGHEST\_PHASE\_ALL,';')) REGEXP\_COUNT

FROM drug\_information\_master;

SEMICOLON separate (And convert into separate columns):

select gvk\_id,HIGHEST\_PHASE\_ALL,trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 1)) HIGHEST\_A,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 2)) HIGHEST\_B,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 3)) HIGHEST\_C,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 4)) HIGHEST\_D,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 5)) HIGHEST\_E,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 6)) HIGHEST\_F,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 7)) HIGHEST\_G,

trim(regexp\_substr(HIGHEST\_PHASE\_ALL, '[^;]+', 1, 8)) HIGHEST\_I,

STD\_HIGHEST\_PHASE\_ALL

from drug\_information\_master;

HYPHEN separate (And convert into separate columns):

select gvk\_id,HIGHEST\_PHASE\_ALL,trim(regexp\_substr(HIGHEST\_A, '[^-]+', 1, 1)) HIGHEST\_A,

trim(regexp\_substr(HIGHEST\_B, '[^-]+', 1, 1)) HIGHEST\_B,

trim(regexp\_substr(HIGHEST\_C, '[^-]+', 1, 1)) HIGHEST\_C,

trim(regexp\_substr(HIGHEST\_D, '[^-]+', 1, 1)) HIGHEST\_D,

trim(regexp\_substr(HIGHEST\_E, '[^-]+', 1, 1)) HIGHEST\_E,

trim(regexp\_substr(HIGHEST\_F, '[^-]+', 1, 1)) HIGHEST\_F,

trim(regexp\_substr(HIGHEST\_G, '[^-]+', 1, 1)) HIGHEST\_G,

trim(regexp\_substr(HIGHEST\_I, '[^-]+', 1, 1)) HIGHEST\_I,

STD\_HIGHEST\_PHASE\_ALL

from DIM\_SEMI\_SEPARATE;

select \* from DIM\_SEMI\_SEPARATE where REGEXP\_COUNT(HIGHEST\_PHASE\_ALL,';') in

(

SELECT max(REGEXP\_COUNT(HIGHEST\_PHASE\_ALL,';')) REGEXP\_COUNT

FROM drug\_information\_master);

select \* from DIM\_HYPHEN\_SEPARATE where REGEXP\_COUNT(HIGHEST\_PHASE\_ALL,';') in

(

SELECT 7 REGEXP\_COUNT

FROM drug\_information\_master);

select \* from DIM\_HYPHEN\_SEPARATE where gvk\_id not in(

select gvk\_id from

(

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A)=trim(STD\_HIGHEST\_PHASE\_ALL))

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B||'; '||HIGHEST\_C)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B||'; '||HIGHEST\_C||'; '||HIGHEST\_D)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B||'; '||HIGHEST\_C||'; '||HIGHEST\_D||'; '||HIGHEST\_E)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B||'; '||HIGHEST\_C||'; '||HIGHEST\_D||'; '||HIGHEST\_E||'; '||HIGHEST\_F)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B||'; '||HIGHEST\_C||'; '||HIGHEST\_D||'; '||HIGHEST\_E||'; '||HIGHEST\_F||'; '||HIGHEST\_G)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_A||'; '||HIGHEST\_B||'; '||HIGHEST\_C||'; '||HIGHEST\_D||'; '||HIGHEST\_E||'; '||HIGHEST\_F||'; '||HIGHEST\_G||'; '||HIGHEST\_H)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where trim(HIGHEST\_B||'; '||HIGHEST\_A)=trim(STD\_HIGHEST\_PHASE\_ALL)

and STD\_HIGHEST\_PHASE\_ALL like '%;%')

union all

(select gvk\_id

from DIM\_HYPHEN\_SEPARATE

where HIGHEST\_PHASE\_ALL is null and STD\_HIGHEST\_PHASE\_ALL is null)

));

To display last word :

Type1:

select regexp\_substr(

'ThisSentence.ShouldBe.SplitAfterLastPeriod.Sentence',

'[^.]+$')

from dual;

Output: Sentence

Type2:

select (case when str like '%.' then ''

else substr(str, - instr(reverse(str), '.') + 1)

end)

from (select 'ThisSentence.ShouldBe.SplitAfterLastPeriod.Sentence' as str from dual) t;

Output: Sentence

SELECT

REGEXP\_SUBSTR('http://www.oracle.com/products',

'http://([[:alnum:]]+\.?){3,4}/?') "REGEXP\_SUBSTR"

FROM DUAL;

Output:

REGEXP\_SUBSTR

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

<http://www.oracle.com/>

The following query returns the first and last names for those employees with a first name of Steven or Stephen (where first\_name begins with Ste and ends with en and in between is either v or ph):

SELECT first\_name, last\_name

FROM employees

WHERE REGEXP\_LIKE (first\_name, '^Ste(v|ph)en$');

OUTPUT:

FIRST\_NAME LAST\_NAME

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

Steven King

Steven Markle

Stephen Stiles

The following example examines the string, looking for two or more spaces. Oracle replaces each occurrence of two or more spaces with a single space.

SELECT

REGEXP\_REPLACE('500 Oracle Parkway, Redwood Shores, CA',

'( ){2,}', ' ') "REGEXP\_REPLACE"

FROM DUAL;

OUTPUT:

REGEXP\_REPLACE

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

500 Oracle Parkway, Redwood Shores, CA

The following example examines phone\_number, looking for the pattern xxx.xxx.xxxx. Oracle reformats this pattern with (xxx) xxx-xxxx:

SELECT

REGEXP\_REPLACE(phone\_number,

'([[:digit:]]{3})\.([[:digit:]]{3})\.([[:digit:]]{4})',

'(\1) \2-\3') "REGEXP\_REPLACE"

FROM employees;

OUTPUT:

REGEXP\_REPLACE

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

(515) 123-4567

(515) 123-4568

(515) 123-4569

(590) 423-4567

The following query returns the last name for those employees with a double vowel in their last name (where last\_name contains two adjacent occurrences of either a, e, i, o, or u, regardless of case):

SELECT last\_name

FROM employees

WHERE REGEXP\_LIKE (last\_name, '([aeiou])\1', 'i');

OUTPUT:

LAST\_NAME

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

De Haan

Greenberg

Khoo

Gee

Greene

Lee

Bloom

Feeney

The following example examines the string, looking for the first substring bounded by commas. Oracle Database searches for a comma followed by one or more occurrences of non-comma characters followed by a comma. Oracle returns the substring, including the leading and trailing commas.:

SELECT

REGEXP\_SUBSTR('500 Oracle Parkway, Redwood Shores, CA',

',[^,]+,') "REGEXPR\_SUBSTR"

FROM DUAL;

OUTPUT:

REGEXPR\_SUBSTR

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

, Redwood Shores,

How to extract the string after a dash ?

SELECT SUBSTR(your\_column, INSTR(your\_column,'-', -1) + 1)

FROM your\_table;

semicolon separator and convert into rows:

select

t.gvk\_id,HIGHEST\_PHASE\_ALL,

regexp\_substr(t.HIGHEST\_PHASE\_ALL, '[^;]+', 1, commas.column\_value) as A

from

DIM\_SEMI\_SEPARATE t,

table(cast(multiset(select level from dual connect by level <= length (regexp\_replace(t.HIGHEST\_PHASE\_ALL, '[^;]+')) + 1) as sys.OdciNumberList)) commas;

select

t.gvk\_id,THERAPEUTIC\_USE,

trim(regexp\_substr(t.THERAPEUTIC\_USE, '[^,]+', 1, commas.column\_value)) as A

from

all\_THERAPEUTIC\_USE\_ICD10 t,

table(cast(multiset(select level from dual connect by level <= length (regexp\_replace(t.THERAPEUTIC\_USE, '[^,]+')) + 1) as sys.OdciNumberList)) commas;

**To match a preceding character exactly n times, you use the char{n} pattern. The following example returns the first names that contain exactly two letters L or 'l':**

SELECT

  first\_name

FROM

  employees

WHERE

  REGEXP\_LIKE( first\_name, 'l{2}', 'i' )

ORDER BY

  first\_name;

**Table convert into JSON format:**

**(In POSTGRES)**

SELECT array\_to\_json(array\_agg(fred), FALSE) AS ok\_json FROM fred;

-- <-- "TRUE" here will produce plus

("+) signs in the output. "FALSE"

is the default anyway.

SELECT department\_id

, LISTAGG(employee\_id, ',')

WITHIN GROUP (ORDER BY employee\_id)

AS employees

FROM employees

GROUP BY department\_id;

ADD constraint in POSTGRERS:

ALTER TABLE table\_name

ADD CONSTRAINT constraint\_name UNIQUE (column1, column2, ... column\_n);

**To convert a table into JSON format:**

**SELECT** **array\_to\_json**(**array\_agg**(t), **true**) **AS** ok\_json **FROM** cdm.cas t;

**For particular gvk\_id:**

**SELECT** **array\_to\_json**(**array\_agg**(t), **true**) **AS** ok\_json **FROM** cdm.cas t **where** gvk\_id=30141793;

**To converts rows into JSON:**

**select** **row\_to\_json**(t.\*) **as** **line** **from** cdm.cas **as** t

**GOSTAR\_REVAMP**

**create** **table** public.templates (Table\_Name **varchar**(100),

Column\_s\_no **integer**,

Column\_name **varchar**(100),

Data\_type **varchar**(100),

Order\_to\_follow **integer**,

Export\_option\_for **varchar**(100)

);

**create** **table** public.Template\_Master (Template\_id **bigint**,

Template\_Name **varchar**(100),

Table\_Name **Varchar**,

Description **varchar**(1000)

);

**ALTER** **TABLE** public.Template\_Master

**ADD** **CONSTRAINT** TEMPLATE\_ID\_PK **primary** **key** (TEMPLATE\_ID);

**insert** **into** public.Template\_Master (Template\_id,template\_name,Table\_Name,Description)

(**select** **row\_number** () **over**(**order** **by** table\_name **asc**) **as** "Template\_id",Table\_Name **as** "TM",Table\_Name,Table\_Name **as** "DES"

**from**

(**select** **distinct** table\_name **from** public.templates **order** **by** table\_name **asc**)a);

**create** **table** public.Template\_Columns (Template\_column\_id **bigint**,Template\_id **bigint**,

Template\_column\_name **varchar**(100),

Column\_Name **Varchar**,

Columns\_no **integer**,

Data\_type **varchar**(100),

Display\_order **integer**,

Is\_export\_required **boolean**

);

**ALTER** **TABLE** public.Template\_Columns

**ADD** **CONSTRAINT** TEMPLATE\_COLUMN\_ID\_PK **primary** **key** (TEMPLATE\_COLUMN\_ID);

**ALTER** **TABLE** public.Template\_Columns

**ADD** **CONSTRAINT** TEMPLATE\_ID\_FK **FOREIGN** **KEY** (TEMPLATE\_ID) **REFERENCES** public.Template\_Master (TEMPLATE\_ID);

**insert** **into** public.Template\_Columns (Template\_id,Template\_column\_name,Column\_Name,Columns\_no,Data\_type,Display\_order,Is\_export\_required,Template\_column\_id)

(**select** a.Template\_id,b.Column\_name **as** "TCN",b.Column\_name,b.Column\_s\_no,b.Data\_type,b.Order\_to\_follow,

**CASE**

**WHEN** b.Export\_option\_for='Yes' **THEN** 'true'::**boolean**

**when** b.Export\_option\_for **is** **null** **then** 'false'::**boolean**

**ELSE** 'false'::**boolean**

**END** **AS** Export\_option\_for,

(**row\_number** () **over**(**order** **by** (a.template\_id,b.Column\_s\_no) )) **as** "ID"

**from** public.Template\_master a,public.templates b

**where** a.Table\_Name=b.Table\_Name);

**SELECT** Export\_option\_for,

**CASE**

**WHEN** Export\_option\_for='Yes' **THEN** 'true'::**boolean**

**when** Export\_option\_for **is** **null** **then** 'false'::**boolean**

**ELSE** 'false'::**boolean**

**END** **AS** Export\_option\_for

**from** public.templates;

**DB LINKS:**

select distinct db\_link, username, host from DBA\_DB\_LINKS;

SELECT DB\_LINK, USERNAME, HOST FROM ALL\_DB\_LINKS

**Postgres:**

**Cursor QRY:**

**DO** $$**DECLARE**

z\_cas\_upd **CURSOR** **FOR** **SELECT** \* **FROM** dbo.cco\_act\_id;

eno dbo.cco\_act\_id%ROWTYPE;

**BEGIN**

**OPEN** z\_cas\_upd;

**LOOP**

**FETCH** z\_cas\_upd **INTO** eno;

EXIT **WHEN** **NOT** **FOUND**;

**update** dbo.admet\_task\_data **set** cells\_cellline\_organ=eno.cells\_cellline\_organ **where** act\_id=eno.act\_id;

**END** **LOOP**;

**close** z\_cas\_upd;

END$$;

**FILE TAR TO DMP CONVERT:**

tar xvzf chembl\_25\_oracle11g.tar.gz

**Mol convert:**

**select** gvk\_id,molconvert(sub\_smiles\_mol,'mol') **from** "gostarui".STRUCTURE\_DETAILS **where** sub\_smiles **is** **not** **null** **and** gvk\_id <10;

Add Constraints in Oracle:

alter table TARGET\_PROTEIN\_MASTER1 add CONSTRAINT TAR\_PROTN\_MASTER\_TARID\_PK1 PRIMARY KEY (TARGET\_ID);

alter table TARGET\_PROTEIN\_MASTER1 add CONSTRAINT TAR\_PROTN\_MASTER\_STDNMID\_FK1 FOREIGN KEY (STDNAME\_ID) REFERENCES STANDARD\_NAME\_MASTER (STDNAME\_ID) ;

ALTER TABLE TARGET\_PROTEIN\_MASTER1

ADD CONSTRAINT check\_target\_id

CHECK (TARGET\_ID is not null);

**Pg\_dump:**

/usr/pgsql-11/bin/pg\_dump --host=172.21.0.60 --username=gostar\_client\_user --format=p --verbose --schema=fulcrum --file=/disk2/postgres\_gostar\_backups/OCT19/fulcrum.sql gostar\_client\_db

**Mol Convert IN Oracle:**

CREATE INDEX OCT19\_SMILES\_qidxsm ON GOSTAR\_WEEK3.TABLE\_NAME

(sub\_SMILES)

INDEXTYPE IS JCHEM.JC\_IDXTYPE

PARAMETERS('tableType=anyStructures,fp\_size=72,fp\_bit=2,pat\_length=6,std\_config=aromatize:d,absoluteStereo=n,haltOnError=n,duplicateFiltering=n,TDF=n,exclusiveDF=n,TABLESPACE=GOSTAR\_TB')

NOPARALLEL;

alter table GOSTAR\_WEEK3.TABLE\_NAME add

(

mol clob,

ext\_fname varcahr2(100)

);

update GOSTAR\_WEEK3.TABLE\_NAME set mol=jc\_molconvertc(jc\_molconvertc(sub\_SMILES,'smiles:u'),'mol:a') where sub\_SMILES is not null;

commit;

update GOSTAR\_WEEK3.TABLE\_NAME set ext\_fname=gvk\_id||'.mol';

commit;

DECLARE

F UTL\_FILE.FILE\_TYPE;

V\_LINE VARCHAR2 (1000);

V\_SOURCE VARCHAR2;

V\_SCI\_SOURCE\_NAME VARCHAR2;

BEGIN

F := UTL\_FILE.FOPEN ('DP\_DUMPS\_DIR', 'tpm\_source\_sciname\_corr.CSV', 'R');

IF UTL\_FILE.IS\_OPEN(F) THEN

LOOP

BEGIN

UTL\_FILE.GET\_LINE(F, V\_LINE, 1000);

IF V\_LINE IS NULL THEN

EXIT;

END IF;

V\_SOURCE := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 1);

V\_SCI\_SOURCE\_NAME := REGEXP\_SUBSTR(V\_LINE, '[^,]+', 1, 2);

INSERT INTO tpm\_source\_sciname\_corr VALUES(V\_SOURCE, V\_SCI\_SOURCE\_NAME);

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

EXIT;

END;

END LOOP;

END IF;

UTL\_FILE.FCLOSE(F);

END;

/

Table size :

select sum(bytes)

from user\_segments

where segment\_type='TABLE'

and segment\_name = 'Z\_DSP\_KINASE\_TARGET\_MCD\_ACT';

select a.activity\_value\_new,b.activity\_value\_new1,decode(round(to\_number(a.activity\_value\_new),12),round(to\_number(b.activity\_value\_new1),12),'Same','Changed') status

from act\_type\_uom\_check\_data\_final a full outer join act\_type\_uom\_check\_data\_final b on (a.act\_id=b.act\_id) where a.activity\_value\_new is not null;

**Split:**

split -l 250 filename.csv

split -l 100000 STRUCTURE\_DETAILS\_TARGETS.txt --additional-suffix=.txt sd\_1

LISTAGG: (Convert multiple rows into single row)

Ex:

select ref\_id,listagg(authors,'; ') within group (order by(ref\_id)) as authors from refs\_table\_authors where ref\_id =12076 group by ref\_id;

Last DML Operation:

select \* from ALL\_TAB\_MODIFICATIONS where table\_owner='GOSTAR\_WEEK5' and lower(table\_name)='a\_gvk\_formid\_map';

SELECT

   name,

   amount,

   NTILE(3) OVER(

      ORDER BY amount)

FROM

   sales\_stats

--binary\_interger

--pls\_integer

--simple\_interger

Decimal length:

SELECT MAX(LENGTH(SUBSTR(STR\_RATE, INSTR(STR\_RATE, '.')+ 1)))

FROM your\_table;

Ex:

select act\_id,ref\_id,reference,activity\_value,MAX(LENGTH(SUBSTR(activity\_value, INSTR(activity\_value, '.')+1))) a

from all\_activity\_gostar;

where activity\_value is not null group by act\_id,ref\_id,reference,activity\_value order by a desc;

**Split the dat file :**

Using below command in linux:

split -b 1500M CID-InChI-Key

**To read top 10 rows in linux:**

head -n 10 xaa.txt

**To read last 10 rows in linux:**

tail -n 10 xaa.txt

**file converting to txt:**

mv xaj xaj.txt

**Sql loader :**

Step 1: Create control file

In control file:

LOAD DATA

INFILE <"File Name">

APPEND INTO TABLE <Table Name>

FIELDS TERMINATED BY <'\t'>

TRAILING NULLCOLS

(Column List)

And Save as control file with extension .ctl

**Ex:**

LOAD DATA

INFILE "xah.txt"

APPEND INTO TABLE CID\_InChI\_Key\_DEC09

FIELDS TERMINATED BY '\t'

TRAILING NULLCOLS

(CID,INCHI char(10000),INCHIKEY char(10000))

**To skip column use below command:**

LOAD DATA

INFILE "xah.txt"

APPEND INTO TABLE CID\_InChI\_Key\_DEC09

FIELDS TERMINATED BY '\t'

TRAILING NULLCOLS

(CID,INCHI FILLER,INCHIKEY char(10000))

Step 2:

After Step 1 use below command in linux (putty) to load the data into table.

sqlldr USER\_NAME/PASSWORD control=<file\_name.ctl> log=<file\_name.log> bad=<file\_Name.bad> direct=true

Ex:

sqlldr GOSTAR\_WEEK5/GOSTAR\_WEEK5 control=cid\_inchi\_ctl.ctl log=loader\_abc.log bad=loader\_abc.bad direct=true

SELECT

COUNT(DISTINCT gvk\_id),

COUNT(DISTINCT CASE

WHEN database\_name = 'DD' THEN str\_id

ELSE NULL

END) dd\_count,

COUNT(DISTINCT CASE

WHEN database\_name = 'CCD' THEN str\_id

ELSE NULL

END) ccd\_count

FROM

structure\_details;

SELECT act\_id,activity\_remarks,

trim(REGEXP\_REPLACE(activity\_remarks,

'( ){2,}', ' ')) "REGEXP\_REPLACE"

FROM all\_activity\_gostar where activity\_remarks is not null and REGEXP\_LIKE(activity\_remarks, '( ){2,}');

update all\_activity\_gostar set activity\_remarks=trim(REGEXP\_REPLACE(activity\_remarks,'( ){2,}', ' ')) where act\_id in (SELECT act\_id

FROM all\_activity\_gostar where activity\_remarks is not null and REGEXP\_LIKE(activity\_remarks, '( ){2,}'));

Query V$SESSION supplying the username for the session you want to terminate:

SELECT SID, SERIAL#, STATUS, SERVER

FROM V$SESSION

WHERE USERNAME = '<username>';

Execute the ALTER SYSTEM command to terminate the session:

ALTER SYSTEM KILL SESSION '<sid, serial#>'

EX:

SELECT SID, SERIAL#, STATUS, SERVER

FROM V$SESSION

WHERE USERNAME = 'GOSTARUI' and PROGRAM='SQL Developer' and OSUSER='navakanth.suram';

select ref\_id,reference,ref\_type,a1\_cnt,listagg(assay\_types,',')within group (order by ref\_id) assaytypes from(

select distinct a.ref\_id ref\_id,a.reference reference ,a.ref\_type ref\_type ,count(distinct b.assay\_type) a1\_cnt ,b.assay\_type,decode(lag(b.assay\_type)over (partition by null order by b.assay\_type),b.assay\_type,null,b.assay\_type) assay\_types

from gostarui.reference\_master a,gostarui.all\_activity\_gostar b

where a.ref\_id in (2115014,2106785,2055842)

and a.ref\_id=b.ref\_id

group by a.ref\_id,a.reference,a.ref\_type,b.assay\_type) group by ref\_id,reference,ref\_type,a1\_cnt order by assaytypes;

**select** **row\_number**() **over** (**order** **by** m.ref\_id, m.auth\_s\_no, m.comp\_s\_no) racd\_id,

        m.ref\_id, m.auth\_s\_no, m.comp\_s\_no,

        a.authors, c.company\_name, c.company\_address,

        ac.ind\_address, ac.email, **coalesce**(ac.phone1, ac.phone2) phone

**from** gostarui.refs\_auth\_comp\_ref\_mapping m **inner** **join** gostarui.refs\_table\_authors a **on** (a.ref\_id = m.ref\_id **and** a.auth\_s\_no = m.auth\_s\_no)

**left** **outer** **join** gostarui.refs\_table\_company\_address c **on** (c.ref\_id = m.ref\_id **and** c.comp\_s\_no = m.comp\_s\_no)

**left** **outer** **join** gostarui.refs\_authors\_contact ac **on** (ac.ref\_id = a.ref\_id **and** ac.auth\_s\_no = a.auth\_s\_no);

**NUMBER:**

regexp\_like(synonyms,'^[[:digit:]]+$')

**formid export sqlldr:**

rm xaa.dat

rm xaa1.dat

scala -cp . StrIdsSmi2Smi xab xab.dat

gawk -F "|" '{ print $1 "|<startlob>" $2 "<endlob>|<startlob>" $3 "<endlob>|" $4 "|" $5 }' xab.dat > xaa.dat

sqlldr GOSTARDCM\_DEC19/GOSTARDCM\_DEC19 control=FEB12\_2\_nosalts.ctl log=FEB12\_2\_nosalts.log bad=FEB12\_2\_nosalts.bad errors=9999999 direct=true

rm xaa.dat

rm xab.dat

scala -cp . StrIdsSmi2Smi xac xac.dat

gawk -F "|" '{ print $1 "|<startlob>" $2 "<endlob>|<startlob>" $3 "<endlob>|" $4 "|" $5 }' xac.dat > xaa.dat

sqlldr GOSTARDCM\_DEC19/GOSTARDCM\_DEC19 control=FEB12\_2\_nosalts.ctl log=FEB12\_2\_nosalts.log bad=FEB12\_2\_nosalts.bad errors=9999999 direct=true

**Double Space:**

SELECT act\_id,activity\_remarks,

trim(REGEXP\_REPLACE(activity\_remarks,

'( ){2,}', ' ')) "REGEXP\_REPLACE"

FROM all\_activity\_gostar where activity\_remarks is not null and REGEXP\_LIKE(activity\_remarks, '( ){2,}');

--Update Query :

update all\_activity\_gostar set activity\_remarks=trim(REGEXP\_REPLACE(activity\_remarks,'( ){2,}', ' ')) where act\_id in (SELECT act\_id

FROM all\_activity\_gostar where activity\_remarks is not null and REGEXP\_LIKE(activity\_remarks, '( ){2,}'));

**UPDATE QRY FOR EXTRA CHAR(PRINT):**

select \* from REFS\_TABLE\_COMPANY\_ADDRESS\_GP where REGEXP\_like(company\_name, '[^[:print:]]');

select company\_name,REGEXP\_REPLACE(company\_name, '[^[:print:]]', ' ') from REFS\_TABLE\_COMPANY\_ADDRESS where REGEXP\_like(company\_name, '[^[:print:]]');

update REFS\_TABLE\_COMPANY\_ADDRESS\_GP set company\_name = REGEXP\_REPLACE(company\_name, '[^[:print:]]', ' '),

company\_address = REGEXP\_REPLACE(company\_address, '[^[:print:]]', ' ')

where REGEXP\_like(company\_name, '[^[:print:]]');

**UPDATE QUERY FOR MULTIPLE COLUMNS FROM DIFFERENT TABLE:**

update REFS\_TABLE\_COMPANY\_ADDRESS\_GP r

set (company\_name, company\_address) =

(select company\_name, company\_address

from refs\_table\_company\_address\_u ri where ri.ref\_id = r.ref\_id and ri.comp\_s\_no = r.comp\_s\_no and ri.company\_name <> r.company\_name and ref\_id = 51907 and rownum = 1)

where exists (select distinct company\_name, company\_address

from refs\_table\_company\_address\_u ri

where ri.ref\_id = r.ref\_id and ri.comp\_s\_no = r.comp\_s\_no and ri.company\_name <> r.company\_name and ref\_id = 51907 and rownum = 1);

Query for Extra chars:

Example:

select assay\_id, replace(translate(ENZYME\_CELL\_ASSAY,' `1234567890~!@#$%^&\*()abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ,.;''[]|}{":?></\+-\_=',' '), ' ', '') as ENZYME\_CELL\_ASSAY ,INCUBATION\_TIME,MEASURED,ASSAY\_METHOD\_NAME,DOSE,PH,TEMPERATURE,DOSE\_UOM,RADIO\_LIGAND\_CONC,BUFFER,RADIO\_LIGAND,ACTIVITY,IN\_PRESENCE\_OF,CO\_ADMINISTERED,DOSE\_CONCENTRATION,RADIO\_LIGAND\_UOM,SUBSTRATE,SUBSTRATE\_UOM,IN\_PRESENCE\_OF\_UOM

from ACTIVITY\_ASSAY

where replace(translate(ENZYME\_CELL\_ASSAY,' `1234567890~!@#$%^&\*()abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ,.;''[]|}{":?></\+-\_=',' '), ' ', '') is not null;

**GENERATE EXISTS INDEX SCRIPTS:**

select

pg\_get\_indexdef(pg\_index.indexrelid)|| ';'

from

pg\_index

inner join pg\_class on

indexrelid = pg\_class.oid

inner join pg\_namespace on

pg\_namespace.oid = pg\_class.relnamespace

where

indisprimary = false

and indisvalid = true

and nspname not like 'pg\_%25' and nspname not like 'pg\_%'

order by

nspname,

relname;

**GENERATE EXISTS CONSTARINTS SCRIPTS:**

select

'ALTER TABLE ' || nspname || '.' || relname || ' ADD CONSTRAINT ' || conname || ' ' || pg\_get\_constraintdef(pg\_constraint.oid)|| ';'

from

pg\_constraint

inner join pg\_class on

conrelid = pg\_class.oid

inner join pg\_namespace on

pg\_namespace.oid = pg\_class.relnamespace

order by

case

when contype = 'f' then 0

else 1

end desc,

contype desc,

nspname desc,

relname desc,

conname desc;

**To remove duplicates in POSTGRES:**

**delete**

**from**

iktos\_stg.DRUG\_ADVERSE\_EVENTS DAEA

**using** (

**select** **min**(ctid) **as** ctid,

gvk\_id,

ICD10\_CODE,

CLASSIFICATION\_CODE,

ADVERSE\_EVENTS,

physiological\_use

**from**

iktos\_stg.DRUG\_ADVERSE\_EVENTS

**group** **by**

gvk\_id,

ICD10\_CODE,

CLASSIFICATION\_CODE,

ADVERSE\_EVENTS,

physiological\_use

**having**

**count**(\*)>1) DAEB

**where**

DAEA.gvk\_id = DAEB.gvk\_id

**and** DAEA.ICD10\_CODE = DAEB.ICD10\_CODE

**and** DAEA.CLASSIFICATION\_CODE = DAEB.CLASSIFICATION\_CODE

**and** DAEA.ADVERSE\_EVENTS = DAEB.ADVERSE\_EVENTS

**and** DAEA.physiological\_use = DAEB.physiological\_use

**and** DAEA.ctid <> DAEB.ctid;

**Extra Enter:**

select PHONE1

from REFS\_AUTHORS\_CONTACT

where instr(PHONE1, chr(10)) > 0;

**ALTER** **TABLE** "TARGETS\_MCD\_CDM".et\_mcd\_activity\_jan2020 **ALTER** **COLUMN** activity\_value **TYPE** **float8** **USING** activity\_value::**float8**;

**INDEX SCRIPT:**

select dbms\_metadata.get\_ddl('INDEX', index\_name, owner)

from all\_indexes

where owner in ('GILEADCLIENT');

**CONSTRAINTS SCRIPT:**

**--FOREGIN KEY SCRIPT:**

SELECT dbms\_metadata.get\_dependent\_ddl('REF\_CONSTRAINT', table\_name)

FROM user\_tables t

WHERE table\_name IN ('BINDING\_SITE','CAS','CLINICAL\_STUDY','CLINICAL\_STUDY\_DESIGN','COMPD\_MET\_TOX\_ACTIVITY','COMPOUND\_SYNONYMS','DRUG\_ADVERSE\_EVENTS','DRUG\_COMPANY','DRUG\_FORM','DRUG\_INFORMATION\_MASTER','DRUG\_INFORMATION\_TRIAL\_REFS','DRUG\_INTERACTIONS','DRUG\_ROA','DRUG\_ROE','DRUG\_TARGETS','GENERAL\_TERMS','GVK\_DUPES\_TABLE','LIPINSKI\_CALC','LIPINSKI\_EXP','LOCUS\_REF','MECHANISM','MECHANISM\_TOXICITY','METABOLISM\_MBT','METABOLISM\_RXN','METABOLISM\_SCHEME','METABOLITE\_ENZYME','METABOLITE\_REFERENCE','METABOLITE\_SPECIES','MULTIPLELOCI','PRIMARY\_TARGET','REFERENCE\_MAPPING','REFS\_AUTHORS\_CONTACT','STRUCTURE\_GVKID','STRUCTURE\_REFERENCE','TABLE\_AUTHORS','TABLE\_COMPANY\_ADDRESS','TABLE\_SIMILAR\_PATENTS','TARGET\_SYNONYMS','THERAPEUTIC\_AREA','TOX\_STRUCTURE','ALL\_ACTIVITY\_GOSTAR')

AND EXISTS (SELECT 1

FROM user\_constraints

WHERE table\_name = t.table\_name

AND constraint\_type = 'R');

**--UNIQUE CONSTRAINT SCRIPT:**

SELECT DBMS\_METADATA.get\_ddl ('CONSTRAINT', constraint\_name, owner)

FROM all\_constraints

WHERE owner = UPPER('GILEADCLIENT')

AND constraint\_type IN ('U');

**--CHECK CONSTRAINT SCRIPT:**

select dbms\_metadata.get\_ddl('CONSTRAINT', c.constraint\_name)

from

user\_constraints c

where

c.constraint\_type = 'C';

**PRIMARY KEY CONSTARINT SCRIPT:**

select dbms\_metadata.get\_ddl('CONSTRAINT', c.constraint\_name)

from

user\_constraints c

where

c.constraint\_type = 'P';

**INITIAL CAP (INITCAP):**

select upper(substr(columnn\_name,1,1))||lower(substr(columnn\_name,2,length(columnn\_name))) from table\_name;

**pg\_dump with jobs:**

sudo pg\_dump -h 172.21.0.170 -U gs\_client\_user -d gostar\_client\_db --verbose --no-acl --no-owner --jobs=4 -Fd --schema=gostarui -f gostarui\_202003140-1545

**pg\_restore:**

nohup pg\_restore --host=localhost --port=5432 --dbname=gostar\_db\_mig --username=gostar\_mig\_user --no-acl --no-owner --verbose --jobs=8 gostarui\_202003140-1545 > gs\_202003140-1545.log 2>&1 &

**MD5sum: (CheckSum)**

md5sum \* > md\_checksum.txt

**pg\_dump without jobs:(172.21.0.170 – server)**

(dump path : /disk1/postgres\_backups)

pg\_dump -h 172.21.0.170 -U gs\_client\_user -d gostar\_client\_db --verbose --no-acl --no-owner -Fd --schema=gostarui\_1 -f gostarui\_1\_202003160-0603

**Restore at 56 server:**

Path : /data/postgres\_backups/data\_20200314

nohup pg\_restore --host=localhost --port=5432 --dbname=gostar\_db\_mig --username=gostar\_mig\_user --no-acl --no-owner --verbose gostarui\_1\_202003160-0603 > gostarui\_1\_202003160-0603.log 2>&1

**For single table dump:**

pg\_dump -h 172.21.0.170 -U gs\_client\_user -d gostar\_client\_db --verbose --no-acl --no-owner --jobs=4 -Fd --schema=gostarui -t 'gostarui.logp\_logd\_solubililty' -f gostarui\_202003160-1545

**SPACE after FIRST LETTER:**

select substr(names,1,1)||' '||substr(names,2)from DI\_rp;

Table schema size:

SELECT pg\_size\_pretty(sum(pg\_relation\_size(quote\_ident(schemaname) || '.' || quote\_ident(tablename)))::bigint) FROM pg\_tables

WHERE schemaname = 'yourschema';

select owner as schema\_name,

object\_name as table\_name,

created

from sys.all\_objects

where object\_type = 'TABLE'

-- excluding some Oracle maintained schemas

and owner='GOSTARDCM\_FEB20'

and created > sysdate - 60

order by created desc,

owner,

object\_name;

CREATE USER GOSTAR\_CLIENT IDENTIFIED BY GOSTAR\_CLIENT

DEFAULT TABLESPACE GOSTAR\_TB\_P

TEMPORARY TABLESPACE TEMP2

QUOTA UNLIMITED ON GOSTAR\_TB\_P;

GRANT "CONNECT" TO GOSTAR\_CLIENT WITH ADMIN OPTION;

GRANT "DBA" TO GOSTAR\_CLIENT WITH ADMIN OPTION;

GRANT "RESOURCE" TO GOSTAR\_CLIENT WITH ADMIN OPTION;

ALTER USER GOSTAR\_CLIENT DEFAULT ROLE "CONNECT", "DBA", "RESOURCE";

**SELECT** pg\_size\_pretty(**sum**(pg\_relation\_size(**quote\_ident**(schemaname) || '.' || **quote\_ident**(tablename)))::**bigint**) **FROM** pg\_tables

**WHERE** schemaname = 'smiles\_upsert';

**SELECT** pg\_size\_pretty(pg\_database\_size('gostar\_client\_db'));

**select** spcname

,pg\_tablespace\_location(**oid**)

**from** pg\_tablespace;

**select** table\_name,column\_name,udt\_name,**coalesce**(NUMERIC\_PRECISION,CHARACTER\_MAXIMUM\_LENGTH) datasize

**from** information\_schema.**columns** **where** table\_schema ='target\_search'

**and** **upper**(table\_name) **in** ('ACTIVITY\_MECHANISM\_ALL\_CHILD','ACTIVITY\_MECHANISM\_MASTER','ACTIVITY\_VALUE\_ONTOLOGY\_MASTER','ADME\_ONTOLOGY\_ALL\_CHILDS','ADME\_ONTOLOGY\_MASTER','ADVERSE\_EVENT\_STRUCTURE\_MAP','ALL\_MAPPING\_IDS','ASSAY\_METHOD\_MASTER','ASSAY\_TYPE\_ONTOLOGY\_ALL\_CHILDS','ASSAY\_TYPE\_ONTOLOGY\_MASTER','BIBLIOGRAPH\_ONTOLOGY\_ALL\_CHILDS','BIBLIOGRAPH\_ONTOLOGY\_MASTER','BILBIOGRAPH\_REFERENCE\_MAP','COMPOUND\_PHASE\_ALL\_CHILD','COMPOUND\_PHASE\_MASTER','DRUG\_INDICATIONS','ENDPOINT\_ALL\_CHILDS','ENDPOINT\_MASTER','INDICATION\_ONTOLOGY\_ALL\_CHILDS','INDICATION\_ONTOLOGY\_MASTER','LIST\_ACTIVITY\_MECHANISMS','LIST\_ACTIVITY\_TYPE\_UOM\_PREFIX','LIST\_ACTIVITY\_TYPES','LIST\_ALL\_FAMILIES','LIST\_CLINICAL\_PHASES','LIST\_COMPOUND\_CATEGORIES','LIST\_FAMILIES','LIST\_ICD10\_CODES','LIST\_LOCUS\_ID','LIST\_OFFICIAL\_NAME','LIST\_PDB\_IDS','LIST\_SOURCE\_SYNONYMS','LIST\_SOURCE\_TAX\_IDS','LIST\_SUB\_SUBFAMILIES','LIST\_SUBFAMILIES','LIST\_SUPERFAMILIES','LIST\_TARGET\_SYNONYMS','LIST\_THERAPEUTIC\_USE','LIST\_UNIPROT\_ID','METABOLISM\_STRUCTURE\_MAP','MOL\_WEIGHT\_ONTOLOGY\_MASTER','ONTO\_ASSAYTYPE','PHYSICO\_CHEMICAL\_MASTER','PLOT\_ANALYSIS\_ACTIVITIES','REF\_YEAR\_ONTOLOGY\_ALL\_CHILDS','REF\_YEAR\_ONTOLOGY\_MASTER','REF\_YEAR\_REFERENCE\_MAP','REFERENCE\_COMPANIES','REFERENCE\_ICD10\_CODES','ROLE','SOURCE\_CLASSIFICATION\_MASTER','STANDARD\_NAME\_MASTER','STATISTICS\_SUMMARIES','STR\_COMPOUND\_PHASES','STR\_ID\_BASED','STRUCTURE\_IMAGE\_UPDATES','STRUCTURE\_WEIGHTS','TARGET\_ONTOLOGY\_ALL\_CHILDS','TARGET\_ONTOLOGY\_MASTER','TARGET\_PROTEIN\_MASTER','TARGET\_PROTEIN\_PDB\_IDS','TARGET\_SYNONYMS','TEMPLATE\_COLUMNS','TEMPLATE\_MASTER','THERA\_AREA\_STRUCTURE\_MAP','THERAPEUTIC\_AREA','THERAPEUTIC\_AREA\_AGG','THERAPEUTIC\_AREA\_WITH\_STR','TOXICITY\_ONTOLOGY\_ALL\_CHILDS','TOXICITY\_ONTOLOGY\_MASTER','USER\_EXPORT\_LOG','USERS','USERS\_ROLES')

**order** **by** table\_name,column\_name;

--Mysql:

**select** **upper**(TABLE\_NAME),**upper**(COLUMN\_NAME),**concat**(dtype,sizes) **from** (

**select** TABLE\_NAME,COLUMN\_NAME,dtype,**concat**(datasize,')') sizes **from** (

**select** TABLE\_NAME,COLUMN\_NAME,**concat**(DATA\_TYPE,'(') dtype,

**case** **when** NUMERIC\_PRECISION **is** **null** **then** CHARACTER\_MAXIMUM\_LENGTH **else** NUMERIC\_PRECISION **end** datasize

**from** INFORMATION\_SCHEMA.**Columns**

**where** TABLE\_SCHEMA='IInl\_gov\_260320' ) types) schema\_details **order** **by** TABLE\_NAME;

---Rank: ( In MySQL)

**SELECT** t.\*,

@rownum := @rownum + 1 **AS** **rank**

**FROM** IInl\_gov\_260320.structure\_details t,

(**SELECT** @rownum := 0) r;

**select** **count**(\*) **from** IInl\_gov\_240720.structure\_details sd

**where** gvk\_id **in** (**select** gvk\_id **from** (

**select** t.gvk\_id,@rownum:=@rownum+1 **as** rnk

**from** IInl\_gov\_260320.structure\_details t,

(**SELECT** @rownum :=0) r)b **where** rnk **BETWEEN** 1 **and** 10);

History:

select trunc(hist.sample\_time,'DD') time ,hist.sql\_id,sql.sql\_text

from dba\_hist\_active\_sess\_history hist,

dba\_hist\_sqltext sql

where hist.sql\_id = sql.sql\_id order by time asc ;

mysql database size:

**SELECT** table\_schema **AS** "Database", **SUM**(data\_length + index\_length) / 1024 / 1024 **AS** "Size (MB)"

**FROM** information\_schema.TABLES **GROUP** **BY** table\_schema;

Note Pad++:

Remove all text before comm:

Ctr+f : .+(\,)

Replace :

After where for remove: [where].\*

**Jchem:63 server**

1)

**In Putty**

1.$**oracle/oracle123**

2.goto path:

**cd jchem/cartridge/**

3. **./server.sh stop**

4. **./server.sh start &**

Press enter

II)

**In Toad**

call jchem\_core\_pkg.use\_password('gostarclient\_gsk2019');

---Intersect in Mysql:

**select** gvk\_id **from** integral\_therapeutics.structure\_details\_conn\_table sdc

**inner** **join**

integral\_therapeutics.sd\_cnct\_table\_3mon\_pg sdpg

**using** (gvk\_id);

**create** **table** gostar\_client\_stg.all\_activity\_gostar\_wout\_semi

(

**SELECT**

**DISTINCT** activity\_mechanism,activity\_prefix,activity\_remarks,activity\_type,activity\_uom,activity\_value,act\_id,assay\_id,assay\_no,assay\_type,cells\_cellline\_organ,common\_name,cyp\_enzyme,cyp\_gene\_id,enzyme\_cell\_assay,gvk\_id,micro\_molarvalue,micro\_molar\_prefix,micro\_molar\_remarks,new\_residue,old\_residue,operation,other\_names,**parameter**,pdb\_id,**position**,protein,p\_s\_m,range\_lower,range\_lower\_uom,range\_upper,range\_upper\_uom,reference,ref\_id,sd,source,standard\_name,standard\_uom,stdname\_id,target\_id,std\_activity\_type,std\_act\_prefix,flag,**trim**(**SUBSTRING\_INDEX**(**SUBSTRING\_INDEX**(criteria, ';', n.digit+1), ';', -1)) criteria

**FROM**

gostar\_client\_stg.all\_activity\_gostar aag

**INNER** **JOIN**

(**SELECT** 0 digit **UNION** **ALL** **SELECT** 1 **UNION** **ALL** **SELECT** 2 **UNION** **ALL** **SELECT** 3 **UNION** **ALL** **SELECT** 4 **UNION** **ALL** **SELECT** 5 **UNION** **ALL** **SELECT** 6 **UNION** **ALL** **SELECT** 7 **UNION** **ALL** **SELECT** 8) n

**ON** **LENGTH**(**REPLACE**(criteria, ';' , '')) <= **LENGTH**(criteria)-n.digit);

**SELECT** **distinct** (**CHAR\_LENGTH**(criteria) - **CHAR\_LENGTH**(**REPLACE**(criteria, ';','')) + 1) **as** semicount **from** gostar\_client\_stg.all\_activity\_gostar;

--Oracle column partition”

select REFERENCE,COMPOUND\_NAME,SUB\_SMILES,ACTIVITY\_TYPE,STD\_ACTIVITY\_TYPE,ACTIVITY\_UOM,STANDARD\_UOM,ACTIVITY\_PREFIX,

ACTIVITY\_VALUE,SD,ACTIVITY\_REMARKS,ENZYME\_CELL\_ASSAY,PROTEIN,COMMON\_NAME,SOURCE,ASSAY\_TYPE,CELLS\_CELLLINE\_ORGAN,

ASSAY\_METHOD\_NAME,IN\_PRESENCE\_OF,IN\_PRESENCE\_OF\_PREFIX,IN\_PRESENCE\_OF\_CONC,IN\_PRESENCE\_OF\_UOM,DOSE,DOSE\_PREFIX,DOSE\_UOM,

ROA,CO\_ADMINISTERED,CO\_ADMINISTERED\_PREFIX,CO\_ADMINISTERED\_UOM,SUBSTRATE,SUBSTRATE\_CONC,SUBSTRATE\_PREFIX,SUBSTRATE\_UOM,

RADIO\_LIGAND,RADIO\_LIGAND\_CONC,RADIO\_LIGAND\_PREFIX,RADIO\_LIGAND\_UOM,BUFFER,BUFFER\_PREFIX,BUFFER\_CONC,BUFFER\_UOM,PH\_PREFIX,

PH,INCUBATION\_PREFIX,INCUBATION\_TIME,INCUBATION\_UOM,TEMPERATURE\_PREFIX,TEMPERATURE,TEMPERATURE\_UOM,ACTIVITY,REF\_ID,

ASSAY\_ID,ACT\_ID,GVK\_ID,DOI,PUBMED\_ID,COMPANY\_NAMES,MICRO\_MOLARVALUE

from

(

select REFERENCE,COMPOUND\_NAME,SUB\_SMILES,count(\*) OVER ( PARTITION BY ACTIVITY\_TYPE order by rownum ) n,

ACTIVITY\_TYPE,STD\_ACTIVITY\_TYPE,ACTIVITY\_UOM,STANDARD\_UOM,ACTIVITY\_PREFIX,ACTIVITY\_VALUE,SD,ACTIVITY\_REMARKS,

ENZYME\_CELL\_ASSAY,PROTEIN,COMMON\_NAME,SOURCE,ASSAY\_TYPE,CELLS\_CELLLINE\_ORGAN,ASSAY\_METHOD\_NAME,IN\_PRESENCE\_OF,

IN\_PRESENCE\_OF\_PREFIX,IN\_PRESENCE\_OF\_CONC,IN\_PRESENCE\_OF\_UOM,DOSE,DOSE\_PREFIX,DOSE\_UOM,ROA,CO\_ADMINISTERED,

CO\_ADMINISTERED\_PREFIX,CO\_ADMINISTERED\_UOM,SUBSTRATE,SUBSTRATE\_CONC,SUBSTRATE\_PREFIX,SUBSTRATE\_UOM,RADIO\_LIGAND,

RADIO\_LIGAND\_CONC,RADIO\_LIGAND\_PREFIX,RADIO\_LIGAND\_UOM,BUFFER,BUFFER\_PREFIX,BUFFER\_CONC,BUFFER\_UOM,PH\_PREFIX,PH,

INCUBATION\_PREFIX,INCUBATION\_TIME,INCUBATION\_UOM,TEMPERATURE\_PREFIX,TEMPERATURE,TEMPERATURE\_UOM,ACTIVITY,REF\_ID,ASSAY\_ID,

ACT\_ID,GVK\_ID,DOI,PUBMED\_ID,COMPANY\_NAMES,MICRO\_MOLARVALUE

from OMEROS\_PREFINAL

) where n =1;

Mol to smile convert:

select gvk\_id,jc\_molconvertc(mol\_new,'smiles') sub\_smiles from gostar\_week3.POSTERA\_SUBSMILES\_22MAR21;

tablespace creation:

CREATE TABLESPACE GOWYETH\_TB DATAFILE '/media/disk2/oracle/oradata/dotnetdb/gostar\_gowyeth\_tb\_01.dbf'

SIZE 20G AUTOEXTEND OFF

LOGGING ONLINE EXTENT MANAGEMENT LOCAL AUTOALLOCATE BLOCKSIZE 8K SEGMENT SPACE MANAGEMENT AUTO FLASHBACK ON;

alter tablespace GOWYETH\_TB add DATAFILE '/media/disk2/oracle/oradata/dotnetdb/gostar\_gowyeth\_tb\_02.dbf' SIZE 20G AUTOEXTEND OFF;

------

mysql> USE db1;

mysql> LOAD DATA INFILE 't1.txt' INTO TABLE t1

FIELDS TERMINATED BY ',' FIELDS ENCLOSED BY '"'

LINES TERMINATED BY '\r\n';

Cartridge:

<https://docs.chemaxon.com/display/docs/cartridge-api.md#src-1803469-cartridgeapi-jc-contains>

to know character set in oracle:

select \* from nls\_database\_parameters;

select osuser,sesion.sid,

sqltext.\*

from v$sqltext sqltext, v$session sesion

where sesion.sql\_hash\_value = sqltext.hash\_value

and sesion.sql\_address = sqltext.address

and sesion.username is not null

and osuser=’**osuser**’

order by sqltext.piece;

schema update in oracle:

select user#,NAME from SYS.user$ WHERE NAME='CISTIM';

USER# NAME

--modify the schema's name

select user#,NAME from SYS.user$ WHERE USER#=288;

UPDATE SYS.USER$ SET NAME='SINEQUA' WHERE USER#=288;

ALTER USER SINEQUA identified by SINEQUA account unlock;

---chemaxon in postgres:

--extension for chemaxon

**Create** **extension** chemaxon\_type;

**Create** **extension** chemaxon\_framework;

--column add

smiles\_mol molecule('sample')

--insert data into column

smiles::molecule('sample') smiles\_mol

--chemindex create

**CREATE** **INDEX** fm2\_smiles\_indx **ON** structure\_search.framework2\_master **USING** chemindex (smiles\_mol);

**select** **trim**(**replace**(**split\_part**(ik\_aux,'AuxInfo',1),**chr**(10),''))inchi,**trim**(**replace**('AuxInfo'||**split\_part**(ik\_aux,'AuxInfo',2),**chr**(10),'')) auxinfo,inchikey **from**(

**select** molconvert(sub\_smiles\_mol,'inchi')ik\_aux,molconvert(sub\_smiles\_mol,'inchikey') inchikey

**from** gostarui.structure\_details

**where** gvk\_id=2446355)a;

PostgreSQL:

status:

/\*pga4dash\*/

**SELECT**

pid,

datname,

usename,

application\_name,

client\_addr,

**to\_char**(backend\_start, 'YYYY-MM-DD HH24:MI:SS TZ') **AS** backend\_start,

state,

wait\_event\_type || ': ' || wait\_event **AS** wait\_event,

pg\_blocking\_pids(pid) **AS** blocking\_pids,

query,

**to\_char**(state\_change, 'YYYY-MM-DD HH24:MI:SS TZ') **AS** state\_change,

**to\_char**(query\_start, 'YYYY-MM-DD HH24:MI:SS TZ') **AS** query\_start,

backend\_type

**FROM**

pg\_stat\_activity

**WHERE**

datname = (**SELECT** datname **FROM** pg\_database **WHERE** datname = 'gostarnext\_migtest\_db')**ORDER** **BY** pid;

**SDF:**

split -l 100000 STRUCTURE\_DETAILS\_GVKID\_SMILES.txt --additional-suffix=.txt sd\_1

cat \*.sdf >> STRUCTURE\_DETAILS\_twofiles.sdf

split -b 100M STRUCTURE\_DETAILS\_twofiles.sdf

cat \*.sdf >> STRUCTURE\_DETAILS.sdf

grep -E -o "<GVK\_ID>" STRUCTURE\_DETAILS\_SDF\_1.sdf >outfile\_merge\_counts\_10L.txt

XML format:

select dbms\_xmlgen.getxml('select GVK\_ID, MOL, EXT\_FNAME, COMPOUND\_NAME, FLAG from gostardcm\_oct20.ELS\_STRUCTURE\_CONN\_AUG21') xml from dual;

attach job:

impdp UNIVERSAL\_MIGRATION/UNIVERSAL\_MIGRATION#123 attach=SYS\_IMPORT\_FULL\_01

Including Zero rows table for export in exp format:

SELECT \* FROM user\_tables WHERE segment\_created = 'NO';

SELECT 'ALTER TABLE '||table\_name||' ALLOCATE EXTENT;' FROM user\_tables WHERE segment\_created = 'NO';

--mol null updation:

(select a.GVK\_ID,a.gvk\_id||'.mol',b.mol,'NOSMILES' from SDC\_NOVA a, gostar\_dcm\_oct21.A\_STR\_DET\_CONN\_TABLE b where a.sub\_smiles is null and b.gvk\_id=30003963);

User lock in Oracle:

select

c.owner,

c.object\_name,

c.object\_type,

b.sid,

b.serial#,

b.status,

b.osuser,

b.machine

from

v$locked\_object a ,

v$session b,

dba\_objects c

where

b.sid = a.session\_id

and a.object\_id = c.object\_id;

ALTER SYSTEM KILL SESSION '581,60531' IMMEDIATE;

ALTER SYSTEM KILL SESSION '37,33419' IMMEDIATE;