Single row Functions

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

Single row Functions returns a value based on a single row in a query

1)character functions.

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

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

1)lower

select ename, lower(ename)

from scott.emp;

2]upper

select ename upper(ename) as "Updated Name"

from scott.emp;

3]initcap

Select ename,initcap(ename)

from scott.emp;

4]lpad

select sal, lpad(sal,10,'X') from scott.emp;

5]rpad

select sal, rpad(sal,10,'#') from scott.emp;

6]LTRIM

select LTRIM(' Aakash Bhatt') from dual'

7]RTRIM

select RTRIM('Aakash Bhatt ') from dual;

8]CONCAT

select CONCAT(ename,job) From Scott.Emp;

select CONCAT(ename,CONCAT('',job)) FROM SCOTT.EMP;

9]LENGTH

SELECT ENAME,LENGTH(ENAME) FROM SCOTT.EMP;

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

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

2]STRING operators

1]SUBSTR

select ENAME,SUBSTR(ENAME,1,2)

FROM SCOTT.EMP;

SELECT ENAME,SUBSTR(ENAME,3,-1) FROM SCOTT.EMP;

SELECT EMPNO,ENAME,

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

q is a quote operator which ignores

SELECT EMPNO||q'[ EMPNO AND THE EMPNAME IS ]'||ENAME||q'[ AND THE DESIGNATION IS ]'||JOB

FROM SCOTT.EMP;

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

2]INSTR

SELECT INSTR('AAKASH BHATT','K') FROM DUAL;

SELECT INSTR(ENAME,'L') FROM SCOTT.EMP;

SELECT INSTR(ENAME,'L',1,2) FROM SCOTT.EMP

3]REPLACE

SELECT REPLACE('JACK AND JUE','J','BL') FROM DUAL;

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

3]Numeric FUNCTIONS

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1]CEIL--> IT RETURNS THE LARGEST INTEGER.

SELECT CEIL(15.3) FROM DUAL;

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

2] FLOOR--> IT RETURNS THE SMALLEST INTEGER.

SELECT FLOOR(15.7) FROM DUAL;

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

3]ROUND-->ROUNDING TO THE NEAREST DECIMAL POINT.

SELECT ROUND(EMPNO) FROM SCOTT.EMP;

SELECT ROUND(12.5578,3) FROM DUAL;

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

4]TRUNC-->WILL NOT ROUND TO NEAREST DECIMAL POINT

SELECT TRUNC(12.96,1) FROM DUAL;

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

5]MOD-->

SELECT MOD(12,6) FROM DUAL;

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

4]DATE FUNCTIONS

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1]SYSDATE-->IT RETURNS THE ORACLE DATABASE SERVER'S DATE.

SELECT SYSDATE FROM DAUL;

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

2]CURRENT\_DATE-->IT RETURNS THE OS DATE.

SELECT CURRENT\_DATE FROM DUAL;

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

3]CURRENT\_TIMESTAMP-->IT WILL RETURN SERVERS TIME STAMP.

SELECT CURRENT\_TIMESTAMP FROM DUAL;

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

4]ADD\_MONTHS

SELECT SYSDATE,ADD\_MONTHS(SYSDATE,3) FROM DUAL;

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

5]SUB\_MONTHS

SELECT SYSDATE,ADD\_MONTHS(SYSDATE,-1) FROM DUAL;

6]LAST\_DAY

SELECT LAST\_DAY('29-SEP-2020') FROM DUAL;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

QUERY

FIRST WE USE LAST\_DAY TO COMPUTE THE LAST DATE OF THE

CURRENT MONTH,THEN WE SUBTRACTED -1 MONTH(USING ADD\_MONTHS),

WHICH IS GUARANTEED TO GIVE US THE LAST fay of the PREVIOUS MONTH

FINALLY WE ADD +1 TO ADVANCE THE DAY

SELECT ADD\_MONTHS(LAST\_DAY(SYSDATE),-1)+1 FROM DUAL;

SELECT LAST\_DAY(ADD\_MONTHS(SYSDATE, -1))+1 FROM DUAL;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TO PRINT FIRST DAY OF THE MONTH OF 30 DAY MONTH.

SELECT LAST\_DAY(SYSDATE)-30 +1 FROM DUAL;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7]MONTHS\_BETWEEN

NO OF MONTHS

SELECT HIREDATE,MONTHS\_BETWEEN(SYSDATE,HIREDATE) FROM SCOTT.EMP

NO\_OF\_WEEKS

SELECT ROUND((SYSDATE - HIREDATE)/7)FROM SCOTT.EMP;

8]FOR CALCULATING THE AGE

SELECT ROUND(MONTHS\_BETWEEN((SYSDATE),TO\_DATE('13-JUN-1999'))) FROM DUAL;

SELECT ROUND(MONTHS\_BETWEEN((SYSDATE),TO\_DATE('13-JUN-1999'))/12) AS "AGE" FROM DUAL;

9]NEXT\_DAY

SELECT NEXT\_DAY(SYSDATE,'WEDNESDAY')FROM DUAL;

-\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CONVERSION FUNCTIONS

1]TO\_CHAR-TO CHANGE NUMBER INTO CHARACTERS.

SELECT EMPNO,ENAME,HIREDATE,TO\_CHAR(HIREDATE,'DD/MM/YYYY') FROM SCOTT.EMP;

SELECT EMPNO,ENAME,HIREDATE,TO\_CHAR(HIREDATE,'MONTH/YEAR') FROM SCOTT.EMP;

WEEK

SELECT SYSDATE, TO\_CHAR(SYSDATE,'IW') FROM DUAL;

DAYS

SELECT SYSDATE, TO\_CHAR(SYSDATE,'DDD') FROM DUAL;

QUATER

SELECT SYSDATE, TO\_CHAR(SYSDATE,'Q') FROM DUAL;

FOR ALTERING SYSTEM DATE

ALTER SYSTEM SET FIXED\_DATE='01-JAN-2017';

ALTER SYSTEM SET FIXED\_DATE = NONE ;

ALTER SESSION SET NLS\_TERRITORY = 'SPAIN'

SELECT SAL,TO\_CHAR(SAL, '099999') FROM SCOTT.EMP;

FOR PRINTING CURRENCY

SELECT SAL,TO\_CHAR(SAL, 'L99999') FROM SCOTT.EMP;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

FILL MODEOR FORMAT MASK PARAMETER ->IT REMOVES THE BLANKS

SELECT HIREDATE,TO\_CHAR(HIREDATE,'MONTH DD, YYYY')

FROM SCOTT.EMP

/

SELECT HIREDATE,TO\_CHAR(HIREDATE,'FMMONTH DD, YYYY')

FROM SCOTT.EMP

/

2]TO\_DATE

SELECT TO\_CHAR(HIREDATE,'FMDDTH MON,YYYY') FROM SCOTT.EMP;

SELECT TO\_CHAR(HIREDATE,'FMMON DDTH,YYYY') FROM SCOTT.EMP;

WRITE A QUERY TO DIPLAY THE I WAS BORN ON THURSDAY

select concat ('I was born on', to\_char(to\_date('23-sep-1998'),'day')) from dual/

SELECT CONCAT('I WAS BORN ON',TO\_CHAR(TO\_DATE('03-SEP-2020'),'DAY')) FROM DUAL;

SELECT TO\_CHAR(TO\_DATE('01-JAN-2019')-TO\_DATE('02-FEB-2018')) FROM DUAL;

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

3]TO\_NUMBER

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Aggregate FUNCTIONS

1)Avg

SELECT ROUND(AVG(SAL)) FROM EMP;

2)SUM

SELECT SUM(SAL) FROM EMP;

3)COUNT

SELECT COUNT(SAL) FROM EMP;

4)MAX

SELECT MAX(SAL) FROM EMP;

5)MIN

SELECT MIN(SAL) FROM EMP;

6)GREATEST

SELECT ENAME, SAL,COMM,GREATEST(SAL,COMM) FROM EMP;

FUNCTIONS TO DEAL WITH NULL

NVL & NVL2

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

1]NVL

SELECT ENAME, SAL, COMM,NVL(COMM,0) FROM EMP;

SELECT COMM,NVL(TO\_CHAR(COMM),'NO COMMISSION') FROM SCOTT.EMP;

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

2]NVL2(ARG1,ARG2,ARG3)

A]IF THE FIRST EXPRESSION IS NOT NULL THEN IT RETURNS

THE SECOND EXPRESSION.

B]IF THE FIRST EXPRESSIONIS NULL THEN IT RETURNS THIRD EXPRESSION

SELECT COMM,NVL2(COMM,COMM,100) FROM EMP;

QUERY

TO DISPLAY SALRAY + COMM OR ONLY SAL

SELECT ENAME, SAL, COMM, NVL2(COMM,(SAL+COMM),SAL) FROM SCOTT.EMP;

3]NULLIF

SELECT NULLIF('MADRID','MADRID') FROM DUAL;

SELECT COMM,SAL, NVL(NULLIF(COMM,SAL),0) FROM SCOTT.EMP;

SELECT COMM,SAL, NULLIF(COMM,SAL) FROM SCOTT.EMP;

4] COALESCE

-->IF THE FIRST EXPRESSION

SELECT COALESCE(NULL,NULL,NULL,2,3,4,NULL) FROM DUAL;

SELECT ENAME, MGR, COMM, COALESCE(MGR,COMM,1) FROM SCOTT.EMP;

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

SUBSTITUTION VARIABLE

1]&

SELECT \* FROM SCOTT.EMP

WHERE SAL> &VAL;

2]&&

SELECT \* FROM SCOTT.EMP

WHERE SAL > &&VAL;

SELECT \* FROM &TABLENAME

WHERE &COL = &A

ORDER BY &COL2 &Y;

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

CASE & DECODE FUNCTIONS

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

1]DECODE FUNCTIONS

IT TAKES 4 ARGUMENTS

SELECT DECODE('AAKASH','AAKASH','TRUE','FALSE') FROM DUAL;

SELECT ENAME, JOB, DECODE(JOB,

'CLERK','GOOD JOB',

'SALESMAN','YOU ARE WORKING HARD',

'MANAGER','FREE KI SALARY',

'NO MATCH')

AS "DETAILS"

FROM SCOTT.EMP;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2]CASE STATEMENTS

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A]FIXED

-------

SELECT

(

CASE 'DOG'

WHEN 'CAT' THEN 'FALSE'

WHEN 'DOG' THEN 'TRUE'

WHEN 'COW' THEN 'FALSE'

ELSE 'NO MATCH FOUND'

END) AS "ANIMALS"

FROM DUAL;

SELECT ENAME,JOB,

(

CASE JOB

WHEN 'CLERK' THEN 'GOOD JOB'

WHEN 'SALESMAN' THEN 'HARD WORKING'

WHEN 'MANAGER' THEN 'RESPONSIBLE JOB'

ELSE 'SORRY NOT FOUND'

END) AS DESIGNATION

FROM SCOTT.EMP

WHERE SAL >2000;

)

select EMPNO,ENAME,SAL,

(

CASE

WHEN EMPNO BETWEEN 7000 AND 7500 THEN SAL+50

WHEN EMPNO BETWEEN 7600 AND 7800 THEN SAL-100

ELSE SAL

END) AS "REVISED SALARY"

FROM SCOTT.EMP;

select EMPNO,ENAME,SAL,

(

CASE

WHEN SAL<=1500 THEN SAL+500

WHEN ENAME ='KING' THEN SAL-2000

WHEN EMPNO BETWEEN 7600 AND 7700 THEN SAL

WHEN SAL IN(3000) THEN SAL+3000

ELSE SAL

END) AS "REVISED SALARY"

FROM SCOTT.EMP;

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

GROUPING FUNCTIONS

1]GROUP BY

SELECT SAL, DEPTNO FROM SCOTT.EMP

ORDER BY DEPTNO

SELECT DEPTNO, SUM(SAL) FROM SCOTT.EMP

GROUP BY DEPTNO

ORDER BY DEPTNO;

COUNT NO OF EMPLOYEES IN DEPRATMENT

SELECT DEPTNO, COUNT(EMPNO)

FROM SCOTT.EMP

GROUP BY DEPTNO;

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

HAVING CLAUSE

SELECT DEPTNO ,JOB,SUM(SAL)

FROM SCOTT.EMP

GROUP BY DEPTNO,JOB

ORDER BY 1,2;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ROLLUP

SELECT nvl(to\_char(deptno),'Total'), SUM(SAL)

FROM SCOTT.EMP

GROUP BY ROLLUP(DEPTNO)

ORDER BY 1;

SELECT DEPTNO, JOB, SUM(SAL)

FROM SCOTT.EMP

GROUP BY DEPTNO, JOB

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

CUBE

SELECT deptno,JOB, SUM(SAL)

FROM SCOTT.EMP

GROUP BY CUBE(DEPTNO,JOB)

ORDER BY 1,2;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

===========================================================

JOINS

------

1]NATURAL JOINS

->IT JOINS TO TABLES BASED ON THE SAME COLUMN NAME,SAME VALUES,

SAME DATATYPES.

SELECT EMPNO,ENAME, DEPTNO, DNAME

FROM SCOTT.EMP NATURAL JOIN SCOTT.DEPT

WHERE DEPTNO IN(10,30)

ORDER BY DEPTNO;

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

2]INNER JOINS

SELECT E.EMPNO, E.ENAME, D.DNAME, D.DEPTNO

FROM SCOTT.EMP E INNER JOIN SCOTT.DEPT D

ON(E.DEPTNO = D.DEPTNO);

ALTERNATE

SELECT E.EMPNO, E.ENAME, D.DNAME,

FROM SCOTT.EMP E INNER JOIN SCOTT.DEPT D

USING(DEPTNO);

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

3] SELF JOINS

=================================================

EQUI JOIN

-> THOSE JOINS WHICH USES THE EQUALITY OPERATOR

SELECT

E.EMPNO "EMPLOYEE NO"

E.ENAME "EMPLOYEE NAME"

D.ENAME "MANAGER NAME"

FROM SCOTT.EMP E JOIN SCOTT.EMP

ON(E.MGR = D.EMPNO);

SELECT

D.ENAME "MANAGER NAME"

COUNT(D.EMPNO) "COUNT"

FROM SCOTT.EMP E JOIN SCOTT.EMP D

ON(E.MGR = D.EMPNO)

GROUP BY D.ENAME

HAVING COUNT(D.EMPNO) >= 2;

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

NON-EQUI jOIN

->THOSE JOINS WHICH USES OTHER OPERATORS(OTHER THAN THE = OPERATOR)

SELECT E,EMPNO, E.ENAME, E.SAL, S.GRADE

FROM SCOTT.EMP E JOIN SCOTT.SALGRADE S

ON E.SAL BETWEEN S.LOSAL AND S.HISAL;

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

Outer JOINS

=============================

1]left outer join

select e.empno, e.ename, d.dname, d.deptno

from scott.emp e left outer join scott.dept d

on(e.deptno = d.deptno);

2]right outer join

select e.empno, e.ename, d.dname, d.deptno

from scott.emp e right outer join scott.dept d

on(e.deptno = d.deptno);

3]full outer join

select d.dname, d.deptno, e.empno, e.ename

from scott.dept d full outer join scott.emp e

on(e.deptno = d.deptno);

select e.first\_name, e.last\_name, d.department\_name, l.city

from employees e join departments day

on(e.departments\_id = d.department\_id)

join locations

on(l.location\_id = d.location\_id);

==================================================

===================================================

DDL COMMANDS(Data Definition language)(It cannot be rolled back)

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

create,alter,drop,truncate,RENAME

==============

1]CREATE

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

create table emp1

(

fname varchar2(20),

lname varchar2(20)

);

more eg

CREATE TABLE emp2

(

EID number NOT NULL,

Ename VARCHAR2(20)

);

insert into emp2 values(1,'Rahul');

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

ASSIGNING PRIMARY KEY AT COLUMN LEVEL

CREATE TABLE EMP3(

EID NUMBER PRIMARY KEY,

ENAME VARCHAR2(20)

);

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

ASIGNING PRIMARY KEY AT TABLE LEVEL

CREATE TABLE EMP3(

EID NUMBER,

ENAME VARCHAR2(20),

CONSTRAINT PK PRIMARY KEY(EID)

);

create Table DEPT1(

DEPT\_ID NUMBER,

DNAME VARCHAR2(20),

EID NUMBER,

CONSTRAINT PK\_DEPT1 PRIMARY KEY(DEPT\_ID) ON DELETE CASCADE,

CONSTRAINT FK\_DEPT1 FOREIGN KEY(EID) REFERENCES EMP3(EID)

);

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

Creation of table using check constraint

CREATE TABLE EMP4(

EID NUMBER,

ENAME VARCHAR2(20),

CITY VARCHAR2(20),

CONSTRAINT CHK\_CITY CHECK(CITY IN ('MUMBAI','DELHI'))

);

=============================

Creation of table using default and Unique CONSTRAINT

CREATE TABLE EMP5(

EID NUMBER,

ENAME VARCHAR2(20),

CITY VARCHAR2(20),

CONSTRAINT CHK\_CITY2 CHECK(CITY IN ('MUMBAI','DELHI')),

CONSTRAINT UK\_emp5 UNIQUE(EID)

);

create table emp10

(

NAME VARCHAR2(20),

CITY VARCHAR2(20) DEFAULT 'MUMBAI'

);

====================================================

Constraints->It means set of rules and restrictions

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

1]not null:

2]Primary Key:

3]Foreign key:

4]check :it will check the data is full filling any conditions specified

5]default:

6]unique key:allows null one time.

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

=====================================

ALTER-TO ADD OR DROP A COLUMN,TO RENAME A COLUMN,TO MODIFY A DATATYPES

ALTER TABLE EMP11

ADD LAST\_NAME VARCHAR2(20);

ALTER TABLE EMP11

DROP COLUMN LAST\_NAME;

ALTER TABLE EMP11

RENAME COLUMN NAME TO FIRST\_NAME;

ALTER TABLE EMP11

MODIFY FIRST\_NAME VARCHAR2(40);

alter table emp1

ADD EID NUMBER CONSTRAINT PK\_EMP11 PRIMARY KEY

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

ALTER TABLE EMP1

ADD EID NUMBER

alter table emp1

ADD CONSTRAINT PK\_EM11111 PRIMARY KEY(EID)

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

=======================================

duplicate a table

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

create table emp\_dummy as select \* from emp;

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

drop table emp\_dummy ---> to drop the table completely

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

to create a duplicate the existing table without copying the data

create table emp\_dummy as select \* from emp where 1 = 0;

======================================

DML (Data Manupalation language) It can be rolled back

===================

insert,update,delete

===============

1]Insert

insert into emp1(fname,lname)

values('aakash','bhatt');

alternate way

insert into emp1 values ('Vivek','Bhatt');

==============================================

UPDATE-- TO UPDATE A ROW

UPDATE EMP11

SET LAST\_NAME ='SHAH',CITY ='DELHI'

WHERE NAME='HIREN'

=============================================

-----(rollback,commit)-------

Transaction control language(TCL)

==============================================

SAVEPOINT(CHECKPOINT)

INSERT INTO EMP11 VALUES('AAKASH','MUMBAI')

SAVEPOINT A;

============================================

SET OPERATORS

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

CREATE TABLE SALES\_2005

(

PERSON\_NAME VARCHAR2(30),AMOUNT NUMBER

);

CREATE TABLE SALES\_2006

(

PERSON VARCHAR2(30),AMOUNT NUMBER

);

INSERT INTO SALE\_2005('JOE',1000);

1]Union-It will delete the duplicates

SELECT PERSON\_NAME,AMOUNT

FROM SALES\_2005

UNION

SELECT PERSON,AMOUNT

FROM SALES\_2006;

2]union all-It will also include duplicates.

SELECT PERSON\_NAME,AMOUNT

FROM SALES\_2005

UNION ALL

SELECT PERSON,AMOUNT

FROM SALES\_2006;

3]intersect-

SELECT PERSON\_NAME,AMOUNT

FROM SALES\_2005

INTERSECT

SELECT PERSON,AMOUNT

FROM SALES\_2006;

EG:2-

SELECT DEPTNO

FROM EMP

INTERSECT

SELECT DEPTNO

FROM DEPT;

4]minus-it will return the rows which are there in first table but not in second tables.

SELECT DEPTNO

FROM DEPT

MINUS

SELECT DEPTNO

FROM EMP;

========================================================================================

SUB-QUERIES

================

Q1]find out who earning greater salary than blake.(SINGLE ROW QUERY)

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

1]find the salary of blake-(inner query).

2]find those salaries greater than blake(outer query).

INNER query

select sal from emp

where ename = 'BLAKE';

outer query

select empno,ename,sal

from emp

where sal >

Nested query

SELECT EMPNO,ENAME,SAL

FROM EMP

WHERE SAL > (SELECT SAL FROM EMP

WHERE ENAME = 'BLAKE');

TYPES OF SUB-QUERIES

====================

1]SINGLE ROW SUBQUERY-IT RETURNS SINGLE SET OF RECORDS

SELECT ENAME,JOB

FROM EMP

WHERE JOB =(SELECT JOB FROM EMP WHERE EMPNO = 7369);

QUERY-2

SELECT EMPNO,ENAME,SAL,JOB

FROM EMP

WHERE JOB = (SELECT JOB FROM EMP WHERE EMPNO = 7369 )

AND

SAL > ()

Q3- FIND HIGHEST SALARY EARNED BY AN EMPLOYEE

SELECT MAX(SAL) FROM EMP;

Q4-FIND THE SECOND HIGHEST SALARY EARNED BY AN EMPLOYEE

SELECT EMPNO,ENAME,DEPTNO,SAL

FROM EMP

WHERE SAL =( SELECT MAX(SAL) FROM EMP

WHERE SAL <> (

SELECT MAX(SAL) FROM EMP));

Q5-DISPLAY ENAME JOB SAL DNAME OF THAT EMPLOYEE

WHO EARNS LEAST SALARY IN THE ORGANISATION

select e.ename,e.job,e.sal,d.dname

from emp e join dept d

on(e.deptno = d.deptno)

where e.sal = (select min(sal) from emp);

q6- find all the departments having a min salary less than dept 10

PART 1: SELECT DEPTNO,MIN(SAL) FROM EMP

GROUP BY DEPTNO

PART 2: HAVING MIN(SAL) < (SELECT MIN(SAL) FROM EMP

WHERE DEPTNO =10);

2]MULTIPLE ROW SUBQUERY-ITRETURNS MULTIPLE SET OF RECORDS.(MORE THAN ONE ROW)

MULTI ROW COMPARISON OPERATORS

1]IN-EQUAL TO ANY ROW IN THE LIST

SELECT EMPNO,SAL,DEPTNO,ENAME

FROM EMP

WHERE SAL IN (SELECT MAX(SAL) FROM EMP GROUP BY DEPNO);

2]ANY-COMPARE VALUE TO EACH VALUE RETURNED BY THE SUBQUERY(IT WORKS LIKE OR OPERATOR)

SELECT EMPNO,SAL,DEPTNO,ENAME

FROM EMP

WHERE SAL > ANY(SELECT MAX(SAL) FROM EMP GROUP BY DEPNO);

3]ALL-COMPARE VALUE TO EVERY VALUE RETURNED BY THE SUBQUERY(AND).

SELECT EMPNO,SAL,DEPTNO,ENAME

FROM EMP

WHERE SAL < ALL(SELECT MAX(SAL) FROM EMP GROUP BY DEPNO);

=======================================================================

PAIR VISE COMPARISON

--------

SELECT ENAME,MGR,DEPTNO

FROM EMP

WHERE (MGR.DEPTNO) IN (SELECT MGR,DEPTNO FROM EMP

WHERE MGR = 7839 AND DEPTNO = 30);

=============================================================

CO-RELATED SUBQUERIES

-------

IN CO-RELATED SUB QUERY OUTER QUERY GETS EXECUTED AND THE INNER QUERY USES THAT RESULT OF

THE OUTER QUERY TO CHECK WHETHER THE DATA IS AVAILABLE OR NOT IF DATA IS AVAILABLE THEN IT RETURNS TRUE OTHERWISE FALSE

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

USES 2 OPERATORS

1]EXISTS-

1]EXIST OPERATOR TESTS FOR EXISTENCE OF ROWS IN THE RESULT IN SUBQUERY.

2]IF A SUBQUERY ROW IS FOUND THEN THE SEARCH DOESNOT CONTINUE IN THE INNER QUERY.THEN IT RETURNS TRUE.OTHERWISE VICE-VERSA.

2] NOT EXISTS-

1]IT TESTS WHETHER A VALUE RETRIEVED BY THE OUTER QUERY IS NOT

PART OF THE RESULT SET OF THE VALUES RETURNED BY INNER QUERY.

CREATE TABLE EMPLOYEE

(

EID NUMBER PRIMARY KEY;

)

CREATE TABLE PROJECT

(

EID NUMBER ,

PID VARCHAR2(10) PRIMARY KEY,

PNAME VARCHAR2(20),

LOCATION VARCHAR2(30),

CONSTRAINT FK\_PROJ FOREIGN KEY(PID) REFERENCES EMPLOYEE

);

Q]FIND THE DETAILS OF EMP WHO IS WORKING ON ATLEAST ONE PROJECT.

SELECT \* FROM EMPLOYEE

WHERE EXISTS (SELECT EID FROM PROJECT

WHERE EMPLOYEE.EID = PROJECT.EID);

Q]FIND ALL THE EMPLOYEE WHO EARN MORE THAN AVG SALARY IN THE DEPT

SELECT EMPNO,ENAME FROM EMP

WHERE EXISTS (SELECT SAL FROM EMP

WHERE SAL > AVG(SAL));

SELECT EMPNO,ENAME,DEPTNO,SAL FROM EMP E

WHERE SAL > (SELECT AVG(SAL) FROM EMP

WHERE DEPTNO = E.DEPTNO);

=====================================================================

WITH CLAUSE

-----------

WITH A AS (SELECT DUMMY FROM DUAL)

SELECT \* FROM A;

WITH DEPT\_COUNT AS (SELECT DEPTNO,COUNT(\*) AS DEPT\_COUNT FROM EMP

GROUP BY DEPTNO)

SELECT E.ENAME AS "EMPLOYEE NAME",

DC.DEPT\_COUNT AS "DEPRATMENT COUNT"

FROM EMP E,DEPT\_COUNT DC

WHERE E.DEPTNO = DC.DEPTNO;

=================================

QUERY-FIND Nth HIGHEST SALRAY

SELECT EMPNO,ENAME,SAL

FROM EMP E1

WHERE & n-1 = (SELECT COUNT(DISTINCT SAL) FROM EMP E2

WHERE E2.SAL > E1.SAL);

===================================

DCL(DDATA CONTROL LANGUAGE)

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

1]GRANT-TO GIVE PERMISSION,PRIVELLEGES

2]REVOKE-TAKING BACK THE PERMISSION.

WHO IS A USER??

ANY PERSON WHO CAN ACCESS TABLES,ETC.

EG:

-HR

-SCOTT

-MODI

-AAKASH.

WHAT IS A DATABASE?

COLLECTION OF RELATED TABLES IS A DATABASE.

WHAT IS SCHEMA?

IT IS NOTHING BUT A COLLECTION OF TABLES,VIEWS ,SEQUENCES,PROCEDURES,FUNCTIONS,TRIGGERS ETC

command to connect to sysdba

conn / as sysdba

Create a user

Username password

create user aakash identified by aakash;

grant create session to aakash; ---->session privellege

grant create table to aakash; -------->table creation permission.

grant unlimited tablespace to aakash;-->permission to insert in the tables.

grant create session to aakash with admin option;--->session privellege with admin option

so that the user can grant session to other user also

revoke create session from aakash;--> to take back privellege.

to drop the user

drop user username;

Concept of role

grant all privilege to manager;-->all priveleges.

create role manager;

grant create session, create table, unlimited tablespace, create view to manager;

grant manager to aakash;--->assigning a role to a user

create role executive;

to change password

alter user aakash identified by 1234;

to unlock the account

alter user aakash identified by 1234 account unlock;

grant select on user1.abcd to user2;--> select person to other user to see table of other users

grant update(fname) on user1.abcd with grant option;

update user1.abcd

set fname = 'aakash'

where fname = 'jovin';

commit;

===========================================================

data dictionary views

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

data dictionary:- collection of tables and views created

and maintained by the oracle server and contains info about the db.

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

use\_objects:-info about the objects that the user owns or has created

all\_objects-info about all objects like tabes,views,sequence to which u have access

dba\_objects:-info about all objects that are owned by all the users.

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

describe dictionary;

select \* from dictionary

where table\_name = 'DBA\_OBJECTS';

select \* from USER\_TABLES;

select \* from ALL\_TABLES:

SELECT \* FROM USER\_TAB\_COLUMNS;

SELECT \* FROM USER\_CONSTRAINTS;

SELECT \* FROM USER\_SYNONYMS;

SELECT \* FROM USER\_SEQUENCES;

COMMENT ON TABLE EMP1IS 'EMPLOYEE INFO TABLE SO PLEASE BE CAREFUL';

SELECT COMMENTS FROM USER\_TAB\_COMMENTS

WHERE TABLE\_NAME = 'EMP1';

TO DROP A COMMENT

COMMENT ON TABLE EMP1 IS '';

DATABASE OBJECTS

==================

1]TABLE-> BASIC UNIT OF STORAGE COMPOSED OF ROWS.

2]VIEW:->IT LOGICALLY REPRESENTS SUBSET OF DATA FROM ONE OR MORE TABLES

3]SEQUENCES->ITS IS USED GENERATE NUMERICAL VALLUES.

4]SYNONYMS:GIVES ALTERNATE NAME TO OBJECTS.

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

VIEW:1] IT CANNOT CONTAIN HIS OWN DATA.

TYPES OF VIEW

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

1]SIMPLE VIEW:IT DERIVES DATA FROM A SINGLE TANLE IT CAN PERFORM DML ON VIEWS

2]COMPLEX VIEWS:MORE THAN ONE TABLE.IT DOES NOT ALLOW, ANY DML OPERATIONS ON VIEW

EXAMPLE:

CREATE VIEW EMP\_VW1

AS

SELECT EMPNO,ENAME,JOB,SAL

FROM EMP;

CREATE OR REPLACE VIEW EMP\_VW1

AS

SELECT EMPNO AS "EMPLOYEE NUMBER",ENAME AS "EMPLOYEE NAME",

JOB AS "dESIGNATION",SAL AS "SALARY"

FROM EMP;

DIFFERENT WAY OF GIVING ALIASES

CREATE OR REPLACE VIEW EMP\_VW1("EMPLOYEE NUMBER","EMPLOYEE NAME","dESIGNATION","SALARY")

AS

SELECT EMPNO,ENAME ,JOB,SAL

FROM EMP;

COMPLEX VIEW EXAMPLE

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

CREATE OR REPLACE VIEW DEPT\_SUM\_VIEW(DNAME,MINIMUM\_SALARY,MAXIMUM\_SALARY,AVERAGE\_SALARY)

AS

SELECT D.DNAME,MIN(E.SAL),MAX(E.SAL),AVG(E.SAL)

FROM EMP E JOIN DEPT D

ON(E.DEPTNO = D.DEPTNO)

GROUP BY D.DNAME;

============================================

SEQUENCES

---------

IT IS A DB OBJECT THAT GENERATE INTEGER.

CREATE SEQUENCE MM

START WITH 1

INCREMENT BY 1

MINVALUE 1

MAXVALUE 99999 BY DEFAULT IT IS

CYCLE/NOCYCLE

CACHE/NOCACHE

SELECTTHANKYOU SIR