ORACLE(OAK RIDGE ANALYTICALE COMPUTING AND LOGICLE ENGINE)
==>SQL
==>PL/SQL

SQL(STRUCTURED QUERY LANGUAGE)

It Supports to all Environments.
It is a Non Procedure Level Language.
It is Unified Language.
It is Common for all RDBMS.

DATA OBJECTS :

1.TABLES
2.INDEXEs
3.VIEWS
4.CLUSTERS
5.SYNONYMS
6.SEQUENCES

1.TABLES :
A Table is the data structure that holds data in a relational database. A table is comprised of rows and cols.

Each row represents one occurance of the Entity.
Each col represents the property of the Entity.

ON TABLES WE CAN APPLY MAINLY THREE CAT COMMNADS THEY ARE.....
1.DDLC (DATA DEFINITAION LANGUAGE COMMANDS)
2.DMLC (DATA MANIPLATION LANGUAGE COMMANDS);
3.DCLC(DATA CONTROL LANGA.....).

===>**DDL COMMANDS** ON TABLE :

=====>CREATING THE TABLE :

1.**CREATE TABLE** CAT: DDL TYPE : SQL

SYN:.....
CREATE TABLE <TABNAME>
(COL1 DATATYPE(SIZE),COL1 DATATYPE(SIZE),.....);

CREATE TABLE STUDENT
(RNO NUMBER(3),SNAME VARCHAR(20),M1 NUMBER(3),M2 NUMBER(3),M3 NUMBER(3));

DATA TYPES :

1.CHARACTER(n) : Character string. Fixed-length n
2.VARCHAR(n) or : CHARACTER VARYING(n)
Character string. Variable length. Maximum length n
3.INTEGER(p) : Integer numerical (no decimal). Precision p

4.DECIMAL(p,s)   : Exact numerical, precision p, scale s. Example: decimal(5,2) is a number that has 3 digits before the decimal and 2 digits after the decimal
5.NUMERIC(p,s)   : Exact numerical, precision p, scale s. (Same as DECIMAL)
or
NUMBER()
6.DATE   : Stores year, month, and day values

Eg: CREATE TABLE STUDENT
(RNO NUMBER(3),SNAME VARCHAR(20),STREET VARCHAR(20),STATION VARCHAR(20));

CREATE TABLE EMP
(ENO NUMBER(3),ENAME VARCHAR(20),ESAL NUMBER)

====>DISPLAY THE STRUCTURE

2.**DESCRIBE C**AT :DDL TYPE :SQL*PLUS

SYN:.....
DESC[RIBE] <TABNAME>[;]

====>**RENAMING THE TABLE :**

3.**RENAME** CAT :DDL TYPE :SQL

SYN:.......
RENAME <OLDFILE> TO <NEWFILE>

====>MODIFICATION OF TABLE :

4.**ALTER TABLE** CAT :DDL TYPE :SQL

SYN:...
ALTER TABLE <TABNAME>
ADD(COL1 DATATYPE(SIZE),COL2 DATATYPE(SIZE),,,.....);
MODIFY(COL1 DATATYPE(SIZE),,,,,....);

====>DELETING THE TABLE...

5.**DROP TABLE** CAT :DDL TYPE :SQL

SYN:.....
DROP TABLE <TABNAME>;

===>DMLC
====>INSERTING DATA INTO THE TABLE...

6.**INSERT** CAT :DML TYPE :SQL

SYN:
INSERT INTO <TABNAME>(COL1,COL2,......)
VALUES(VAL1,VAL2,VAL3.......);

```
Eg : INSERT INTO STUDENT(RNO,SNAME)
     VALUES(1,'BHANU');

     INSERT INTO STUDENT(RNO,SNAME)
     VALUES(2,'RAMU);
```

MARCOSUBSTITUTION VARIABLES : &,&&

```
      & FOR SQL
      && FOR PL/SQL

EX: INSERT INTO EMP(ENO,ENAME)
    VALUES(&ENO,'&ENAME')

    INSERT INTO BHANU
    VALUES(&ENO,'&ENAME',&SAL,'&ADR');
```

====>**DISPLAY THE RECORDS FROM THE TABLE ....**

7.**SELECT** CAT :DQL TYPE :SQL

```
SYN:
SELECT[DISTINCT] * | [TABLNAME .] COLNAME [ALIAS],COLNAME[ALIAS], , ,
.....
FROM <TABNAME>[,<TABNAME2>]
[WHERE <EXPRESSION>]
[GROUP BY COLNAME[ , COL ] ]
[HAVING <EXPRESSION>]
[ORDERED BY COL | POSITION[ASC|DESC]]
[UNION | UNIONALL | INTERSECT | MINUS ]
[FOR UPDATE COLNAME ] ;
```

```
DISTINCT : To Avoid the Duplicate Records.
ALL : All Rows .
* : All Rows and cols.
ALIAS : Duplicate Name.

Eg: SELECT * FROM EMP;

SELECT ENO,ENAME FROM EMP;

SELECT DISTICT FROM EMP;// IT WILL ELIMINATE THE DUPLICATE RECORDS

SELECT * FROM EMP WHERE SAL>5000;

SELECT * FROM EMP WHERE ENO=1 OR ENO=4 OR ENO=5;

WHERE : Whenever u want to retrive the records on any condition then
we can apply this.
```

====>**OPERATORS**

```
1.RELATIONAL OPERATORS (>, <, >=, <=, !=)
2.LOGICAL OPERATORS (AND, OR, NOT)
```

SPECIAL OPERATORS :

1.IN, NOT IN
2.LIKE, NOT LIKE
3.BETWEEN, NOT BETWEEN
4.IS NULL, IS NOT NULL

1.IN, NOT IN :
COLNAME [NOT] IN (EXP1,EXP2,,,,,,,,,.....);

SELECT * FROM STUDENT WHERE RNO IN(1,4,7,90)

2.LIKE NOT LIKE (only for character expressions)
COLNAME [NOT] LIKE(EXP1,EXP2,,,,,,,,);

in this we have the _(UNDERSCORE),%;

_ for single character or any character
% for no.of characters OR any character

==>ex : select * from student where name like 'k%';
(in this it will display the records whose name is starts with k.)

3.BETWEEN,NOT BETWEEN for Range values

===>select * from emp
where sal between 4000 and 10000;

4.IS NULL, IS NOT NULL it is usefull to find the no.of NULL values records.

select * from emp where eno is null;

====>UPDATING THE TABLE : ......

8.**UPDATE** CAT : DML TYPE : SQL

SYN:
UPDATE <TABNAME>
SET COLNAME=EXPRESSION,COLNAME=EXPRESSION,.......
[WHERE <CONDITION>];

EX UPDATE BHANU
SET SAL=4500 WHERE ENO=1;

UPDATE BHANU
SET SAL=5500 WHERE SAL BETWEEN 4000 AND 5000;

====>DELTING THE RECORD FROM THE TABLE

9.**DELETE** CAT : DML TYPE : SQL

SYN:
DELETE [FROM] <TABNAME>

DELETE FROM BHANU; // TOTAL TABLE OF DATE WILL BE DELETED

DELETE FROM BHANU WHERE SAL<5000;

GROUP BY CLAUSE : This clause is used with select to combine a group of rows based on the vlues of particular column.

GROUP FUNCTIONS/AGGREGATE FUNCTIONS :

1.MAX()
2.MIN()
3.COUNT()
4.SUM()
5.AVG()

1.MAX():This function returns the largest value of all selected values of a column.

SYN: MAX(COLOUMN);
EX : SELECT MAX(SAL) FROM EMP;
SELECT MAX(SAL) MAX _SAL FROM EMP GROUP BY DEPTNO;
==>in this it will return the maximum sal from the dept wise.

2.MIN():This function returns the smallest value of all selected values of a column.

SYN: MIN(COLOUMN);

EX: SELECT MIN(SAL) FROM EMP;

3.COUNT(): This function returns the no.of rows selected.

SYN: COUNT(COL | *)

EX: SELECT COUNT(*), COUNT(SAMT) FROM XYZ;

SELECT DEPTNO,COUNT(*) FROM EMP GROUP BY DEPTNO;

==> in this it will returns no.of persons in each dept;

4.SUM(): This function determines the SUM of selected coloumns.

SYN: SUM(COLUMN | V ALUE);

EX: SELECT SUM(SAL) FROM EMP;

SELECT DEPTNO,SUM(SAL) FROM EMP GROP BY DEPTNO;

5.AVG():This function determines the AVERAGE of selected coloumns.

SYN: AVG(COLUMN | V ALUE);

EX: SELECT AVG(SAL) FROM EMP;

SELECT DEPTNO,AVG(SAL) FROM EMP GROP BY DEPTNO;

==>**HAVING CLAUSE:**

It is used with only Group by clause for conditions.

```
EX : SELECT DEPTNO,SUM(SAL) FROM SALES GROUP BY DEPTNO
         HAVING  SUM(SAL)>20000;
```

====>**ORDER BY CLAUSE :**

This clause is used to diplay the data either assinding or descinding order based on the column or postion.

```
EX: SELECT * FROM STUDENT ORDER BY SNO;

    SELECT * FROM STUDNET ORDER BY 1 DESC;
```

========>**TYPES OF FUNCTIONS**

1.**NUMERIC FUNCTIONS**
2.STRING FUNCTIONS
3.DATE FUNCTIONS
4.CONVERSION FUNCTIONS
5.MISLENIOUS FUNCTIONS
6.GROUP FUNCTIONS
7.SPECIAL FUNCTIONS

1.NUMERIC FUNCTIONS :

=>1.ABS() :
SELECT ABS(-3) FROM DUAL; output is 3.

=>2.SQRT() :
SELECT SQRT(25) FROM DUAL;

=>3.POWER() :
SELECT POWER(2,3) FROM DUAL;
OUTPUT IS 8

=>4.MOD() :
SELECT MOD(123,10) FROM DUAL;
OUTPUT IS 3

=>5.CEIL() :
SELECT CEIL(2.4) FROM DUAL;
OUTPUT IS 3

=>6.FLOOR() :
SELECT FLOOR(2.4) FROM DUAL;
OUTPUT IS 2

=>7.TRUNC() :
SELECT TRUNC(2.3456,2) FROM DUAL;
OUTPUT IS 2.34

=>8.ROUND() :
SELECT ROUND(2.345),ROUND(2.678) FROM DUAL;
OUTPUT 2,3

NOTE : IF THE VALUE IS >0.5 THEN THE RESULT IS CEIL
       OTHERWISE FLOOR VALUE.

=>9.SIGN() :
SELECT SIGN(-234),SIGN(234),SIGN(0) FROM DUAL;
OUTPUT IS -1,1,0

=====>DUAL : It is a system table for public.

2.CHARACTER FUNCTIONS

=>1.UPPER() : SELECT UPPER('bhanu') FROM DUAL;

=>2.LOWER() : SELECT LOWER('BHANU') FROM DUAL;

=>3.INITCAP(): SELECT INITCAP('bhanu prakash') FROM DUAL;
       OUTPUT Bhanu Prakash

=>4.LENGTH() : SELECT LENGTH('BHANU') FROM DUAL;
OUTPUT IS 5

=>5.LPAD()

The first character of given string will be
left padded with char,upto length n.
EX :
SELECT ENO,LPAD(ENAME,15,'-') FROM EMP;

=>6.RPAD()
SELECT RPAD(ENO,15,'-'),ENAME FROM EMP;

=>7.SUBSTR()
SUBSTR(STRING | COL | m[,n]);

SELECT SUBSTR('PRAKASH',2) FROM DUAL;
RAKASH
SELECT SUBSTR('PRAKASH',2,3) FROM DUAL;
RAK

=>8.INSTR():
This is usefull to findout the first occurance of
the given character.

SELECT INSTR('BHANU PRAKASH','A') FROM DUAL;
3
SELECT INSTR('BHANU PRAKASH','A',4) FROM DUAL;
9

>9.LTRIM() :
  This will eliminates the left blank spaces.

```
SELECT LTRIM('   BHANU') FROM DUAL;
BHANU
```

=>10.RTRIM() :
  This will eliminates the right blank spaces.

```
SELECT RTRIM('BHANU   ') FROM DUAL;
BHANU
```

=>11.TRIM() :
THIS WILL ELIMINATE THE LEFT AND RIGHT BLANK SPACES.

```
SELECT TRIM('   BHANU    ') FROM DUAL;
SELECT * FROM EMP WHERE TRIM(ENAME)='BHANU';
```

=>12.CONCAT(COLOUMN|VALUE)
  This function concat coloumn1 with coloumn2.

  ex: SELECT CONCAT(EMPNO,JOB) FROM EMP;

      SELECT EMPNO||JOB FROM EMP;

   WAQ TO DISPLAY EMPNO,JOB IS FOLLOWING.

```
EMPNO JOB
---- ---
213-----Designation is clerk
214-----Designation is manager.

SELECT CONCAT(ENO,'----Designation is'),JOB FROM EMP;
```

wAQ TO DISPLAY THE JOB AND SAL BY THE FOLLOWING.

```
JOB SAL
---- ------
CLERK 400
MANAGER 1000

SELECT JOB,CONCAT(JOB,'(RS.'),CONCAT(SAL,')') FROM EMP;

OR
SELECT JOB,JOB||'(RS.'||SAL||')' FROM EMP;

JOB JOB_SAL
--- -------
CLERK CLERK(RS.400)
MANAGER MANAGER(RS.1000)
```

=>13.ASCII(CHAR) :
THIS FUNCTION RETURNS THE ASCII VALUE TO THE CHARACTER.

```
SELECT ASCII('A'),ASCII('B') FROM DUAL;
65 66
```

WAQ TO DISPLAY ALL THE EMPLOYEES DETAILS WITH
WHOSE EMPLOYEE NAME STARTS WITH FIRST CHARACTER 'A';

```
SELECT * FROM EMP
WHERE ASCII(ENAME)=65;
OR
SELECT * FROM EMP
WHERE ASCII(ENAME) IN (65,66,67);
OR
SELECT * FROM EMP
WHERE ASCII(ENAME) BETWEEN 65 AND 70;
```

=>14.CHR(NUMBER_EXP);
THIS FUNCTION RETURNS ASCII CHARACTER OF THE GIVEN NUMERIC VALUE.

```
SELECT CHR(65) FROM DUAL;
  A
OR
SLECT * FROM EMP
WHERE CHR(ASCII(ENAME)) IN ('A','B','C');
OR
SELECT * FROM EMP
WHERE CHR(ASCII(ENAME)) BETWEEN 'A' AND 'F';
```

=>15.REPLACE(STRING|COLUMN|EXSTING STRING[,REPLACE_STRING]);

```
SELECT REPLACE('SRINIVASA RAO','RAO') FROM DUAL;
SRINIVASA
SELECT REPLACE('SRINIVASA RAO','RAO','REDDY') FROM DUAL;
SRINIVASA REDDY
```

=>16.SOUNDEX(COLOUMN)
THIS FUNCTION RETURNS A CHARACTER STRING
REPRASANTING A SOUND OF WORKS FROM EACH COLOUMN
OR THIS FUNCTION RETURNS A PHONETIC REPRASANTAION
EACH WORD AND ALLOWS YOU TO WORDS COMPARE.

```
SELECT * FROM EMP
WHERE SOUNDEX(ENAME)=SOUNDEX('SUBBA RAO');
   SUBRAO
   SUBBARAO
   SUBBBAARAO
```

3.DATE FUNCTIONS :

ARTHMETIC OPERATORS ON DATES :

1.DATE + NUMBER :

```
SELECT SYSDATE+2 FROM DUAL;
OUTPUT:
SYSDATE+2
---------
11-FEB-08
```

2.DATE - NUMBER :

```
SELECT SYSDATE-2 FROM DUAL;

SYSDATE-2
---------
07-FEB-08
```

3.DATE - DATE :

```
SELECT SYSDATE-TO_DATE('10-AUG-00') FROM DUAL;

SYSDATE-TO_DATE('10-AUG-00')
----------------------------
                 2739.3814
```

4.MONTHS_BETWEEN(DATE1,DATE2);

```
SELECT MONTHS_BETWEEN(SYSDATE,JDATE) FROM EMP;
```

5.NEXT_DAY(DATE1,CHAR)

THIS FUNCTION RETURNS THE DAY OF NEXT SPECIFIED DATE OF WEEK;

```
SELECT NEXT_DAY(SYSDATE,'THURSDAY') FROM DUAL;
11-MAY-00
```

6.ADD_MONTHS(DATE1,N)

```
SELECT ADD_MONTHS(SYSDATE,3) FROM DUAL;
SELECT ADD_MONTHS(SYSDATE,-3) FROM DUAL;
```

7.LAST_DAY(DATE)

```
SELECT LAST_DAY(SYSDATE) FROM DUAL;

LAST_DAY(
---------
29-FEB-08
```

4.CONVERSION FUNCTION :

1.TO_NUMBER(CHR)

THIS FUNCTION CONVERTS CHARACTER WHICH CONTINES A NUMBER.

```
      ITEM_TABLE
--------------------------------
INO   I_NAME      RATE  QTY
--------------------------------
1     BOOKS       RS.50       10
2     PENS  RS.15       50
3     SLATES      RS.5  12
--------------------------------

SELECT I_NAME,TO_NUMBER(SUBSTR(RATE,4))*QTY TOT_AMT FROM ITEM_TABLE;
```

2.TO_CHAR(NUMERIC EXPRESSION[,FMT])

THIS FUNCTION CONVERTS THE NUMERIC TO CHARACTER BY GIVEN FORMAT.

WAQ TO DISPLAY THE EMPLOYEE DETAILS WHOSE JOINED IN THE MONTH OF JUL AND AUGUST.

```
SELECT * FROM EMP
WHERE TO_CHAR(JDATE,'MON') IN ('JUL','AUG');
OR
SELECT * FROM EMP
WHERE TO_CHAR(JDATE,'MM') IN (7,8);
```

SPECIAL DATE FORMATS :

DEFAULT FORMAT ---- DD:MON:YY;

DAY:

D - DAY OF WEEK(1 TO 7)
DD - DAY OF THE MONTH(1 TO 31)
DDD - DAY OF THE YEAR(1 TO 365)
DY - NAME OF THE DAY,THREE LETTER ABBRIVITAION
SUN,MON,THU
DAY - NAME OF THE DAY(MONDAY,SUNDAY,----)

```
SELECT TO_CHAR(SYSDATE,'DAY') FROM DUAL;
TUESDAY
```

MONTH:

MM - MONTH OF NUMBER(1 TO 12)
MON - NAME OF THE MONTH WITH THREE LETTERS(JAN,FEB,MAR----)
MONTH - FULL NAME OF THE MONTH.

YEAR :

Y - LAST DIGIT OF THE YEAR(2000-0)
YY - TWO DIGITS OF THE YEAR(2000-00)
YYY - THREE DIGITS OF THE YEAR(2000-000)
YYYY - FULL
Y,YYY - 2,000

```
SELECT TO_CHAR(SYSDATE,'YEAR') FROM DUAL;
```

WEEK :

W - WEEK OF MONTH(1 TO 5)
WW - WEEK OF YEAR(1 TO 52)

```
SELECT TO_CHAR(SYSDATE,'WW') FROM DUAL;
```

TIME FORMAT :

1.HH : HOUR OF DAY (1 TO 12)
2.HH24 : HOUR OF DAY (1 TO 24)
3.MI : MINUTS
4.SS : SECONDS
5.AM/PM : MERIDIUMS

DEFUALT ORACLE TIME : HH:MI:SS;

```
SELECT TO_CHAR(SYSDATE,'HH:MI:SS') FROM DUAL;
```

SPECIAL DATE FORMATS :

1.TH - ORDINAL NUMBER
  DDTH - 9TH

```
SELECT TO_CHAR(SYSDATE,'DDTH') FROM DUAL;
```

2.SP - SPELD OUT
  DDSP - NINE

```
SELECT TO_CHAR(SYSDATE,'DDSP') FROM DUAL;
```

```
SELECT SAL,TO_CHAR(TO_DATE(SAL,'YYYY'),'YEAR') FROM DUAL;
```

```
SAL SALARY
--- ------
5000 FIVETHOUSAND
2000 TWOTHOUSAND
```

ASSIGNMENT ABOUT MONTHS AND NUMBERS

1 - JAN
2 - FEB AND SO LIKE THAT.

```
SELECT TO_CHAR(TO_DATE(&K,'MM'),'MONTH') FROM DUAL;
```

1-JANUA
2-FEBR

```
SELECT TO_CHAR(TO_DATE(&K,'DD'),'DDSP') FROM DUAL;
```

1-ONE
3-THREE

1 TO 31

```
SELECT TO_CHAR(TO_DATE(&K,'DDD'),'DDDSP') FROM DUAL;
```

123-ONE HUNDRED TWENTY THREE

IN THIS WE CAN GIVE BELLOW 365

MISLENIOUS FUNCTIONS.

1.GREATEST(VLAUE1,VALUE2,VALUE3, , ,)

THIS FUNCTION RETURNS THE LARGEST VALUE IN THE GIVEN VALUES
OR COLOUMNS.

```
SELECT GREATEST(1,2,33,44) FROM  DUAL;

      SALES
-------------------------
SNO   SAMT  COMM
-------------------------
1     5000  4500
2     6000  6100
3     7000  2000
4     500   NULL
---------------------------
SELECT GREATEST(SAMT,COMM) LARGE FROM SALES;


LARGE
-----
5000
6100
7000
```

2.LEAST(COLUMN|VALUE)

THIS FUNCTION IS RETURNS THE LEAST VALUE FROM THE GIVEN VALUES OR
COLOUMNS.

```
SELECT LEAST(33,2,11) FROM DUAL;
```

3.NVL(COLOUMN,EXPRESSION)
  This function is used tot find the Null values. If the coloumn value is null, It returns the given expresssion. If the coloumn
   value is not null it returns the column value.

```
SELECT SLNO,COMM,NVL(COMM,O) NULL_COMM FROM SALES;

SNO   COMM  NULL_COMM
1     4500  4500
2      0
3     4000  4000
```

4.DECODE(COLOUMN|VALUE|,SEARCH_VALUE,RESULT[SEARCH_VALUE,RESULT----
][,DEFAULT VALUE])

THIS FUNCTION IS EQUL OR SIMILAR TO "IF" STATEMENT.

```
SELECT DECODE(1000,1000,'EQUAL','NOT EQUAL') FROM DUAL;
```

OUTPUT : EQUAL

```
SELECT DECODE (&NUMBER,100,'OK','NO') FROM DUAL;
```

```
SELECT
DECODE('&NAME','BHANU','SAMENAME','SRINU','GOOD','RAMU','BETTER','UGLY')
FROM DUAL;
```

WAQ TO CHECK EMPLOYEE SALARIES.

CONDITIONS:

1.IF THE SAL=5000 THEN GRADE-B
2.IF THE SAL>5000 THEN GRADE-A
3.IF THE SAL<5000 THEN GRADE-C;

EMPLOYEE

ENO	SALARY
100	5000
200	4000
300	7000

USING SIGN AND DECODE FUNCTION:

```
SELECT ENO,SALARY,DECODE(SIGN(SALARY-500),0,'GRADE-B',1,'GRAD-A','GRADE-
C') FROM EMPLOYEE;
```

USING SUBSTR

```
SELECT SALARY,DECODE(SUBSTR(TO_CHAR(SAL-5000),1,1),'-','GRADE-
C',0,'GRADE-B','GRADE-A) FROM EMPLOYEE;
SELECT DECODE(SAL,5000,'GRADE-B',GREATEST(SAL,5000),'GRADE-A','GRADE-C')
FROM EMPLOYEE;
SELECT DOCODE(MOD(5000,SAL),0,'B',5000,'A','C') FROM EMPLOYEE;
```

TRY TO IMPLEMENT TRIM,CHR FUNCTIONS;

****

CONSTRAINTS:
1.DOMAIN INTEGRITY
2.ENTITY INTEGRITY
3.REFRENTIAL INTEGRITY

1.DOMAIN INTEGRITY:
   A.NOT NULL CONSTRAINT
   B.CHECK CONSTRAINT

```
CREATE TABLE EMP_MASTER
(ENO NUMBER(3) CONSTRAINT NT_ENO NOT NULL,NAME CHAR(10),
 SAL NUMBER(9,2) CONSTRAINT NT_SAL NOT NULL);
```

EX:
INSERT INTO EMP_MASTER VALUES(100,'VASU',NULL); N
INSERT INTO EMP_MASTER VAUES(NULL,'BOSU',2000); N
INSERT INTO EMP_MASTER VAUES(200,NULL,2000);   y

CHECK CONSTRAINT:

IT IS CONDITIONAL CONSTRAINT IT ALLOWS CONDITION SATISFY VALUES AND NULL VALUES ONLY.

SYNTAX :CREATE TABLE EMP_MASTER
        (ENO NUMBER(3) CONSTRAINT CH_ENO CHECK(ENO>100),
         NAME VARCHAR(20));

EX :
INSERT INTO EMP_MASTER VAUES(100,'VASU'); N
INSERT INTO EMP_MASTER VAUES(101,'SRI'); Y
INSERT INTO EMP_MASTER VAUES(66,'HARI'); N
INSERT INTO EMP_MASTER VAUES(NULL,'SIVA'); Y

ENTITY INTIGIRITY CONSTRAINTS:

THIS CONSTRAINT MAINTAINANCE UNIQUENESS IN A RECORD.
UNIQUE OR PRIMARY KEY FALL UNDER THIS SECTION.

A.**UNIQUE** : THIS CONSTRAINT IS USED TO PREVENT THE DUPLICATE VALUES
WITHIN THE ROWS OF A SPECIFIED COLOUMNS OR IN A TABLE.
THIS CONSTRAINT IS DEFINED TO MORE THAN ONE COLOUMN OR
COMBINATION COLOUMNS IS SAID TO COMP0SITE UNIQUE KEY.

B.**PRIMARY KEY** : THIS CONSTRAINT PREVENTS DUPLICATION OF ROWS AND
DOES NOT ALLOW NULL VALUES.
A TABLE CAN HAVE ONLY ONE PRIMARY KEY.
IF A PRIMARY KEY CONSTRAINT IS ASSINED TO MORE THAN
ONE COLOUMN OR COMBINATION OF COLOUMNS,
IT WILL SAID TO BE COMPOSITE PRIMARY KEY.
WHICH CAN CONTAINE A MAXIMUM OF 16 COLOUMNS.

EX : CREATE TABLE EMP_SAL
     (ENO NUMBER(4) UNIQUE,NAME VARCHAR(20));

INSERT INTO EMP_SAL VALUES(1,'VASU'); y
INSERT INTO EMP_SAL VALUES(1,'SIVA'); n
INSERT INTO EMP_SAL VALUES(NULL,'SRINU'); y
INSERT INTO EMP_SAL VALUES(2,'RAMU'); y

EX :CREATE TABLE EMP_SAL
    (ENO NUMBER(3) PRIMARY KEY,NAME VARCHAR(20));

INSERT INTO EMP_SAL VALUES(1,'VASU'); Y
INSERT INTO EMP_SAL VALUES(1,'VIVA'); N
INSERT INTO EMP_SAL VALUES(NULL,'VASU'); N
INSERT INTO EMP_SAL VALUES(2,'RAMU'); Y

MANY COLOUMNS / MULTIPLE COLOUMNS :

```
CREATE TABLE EMP_SAL
(ENO NUMBER(3),NAME VARCHAR(20),SAL NUMBER(9,2),
 CONSTARINT U_ENO_SAL UNIQUE(ENO,SAL));

INSERT INTO EMP_SAL VALUES(1,'VASU',5000); Y
INSERT INTO EMP_SAL VALUES(NULL,'KKKK',5000); Y
INSERT INTO EMP_SAL VALUES(1,'RAMU',5000); N
INSERT INTO EMP_SAL VALUES(2,'SUBBU',500); Y
```

REFERENTIAL INTITIGRITY CONSTRAINTS :

THIS CONSTRAINT ENFORCES RELATIONSHIP B/W THE TWO TABLES.
FOREIGN KEY IS FALLS UNDER THIS SECTION.

A.**FOREIGN KEY:** THIS CONSTRAINT IS USED TO ESTABLISH A PARENT AND CHILD OR MASTER AND DETAILED RELATIONSHIP B/W TWO TABLES.

SYNTAX :

```
[,][CONSTRAINT < CONST_NAME>FOREIGN KEY(CHILD TABLE COL,....)]
REFERENCES MASTER_TABLE_NAME[PRIMARY KEY COLOUMNS]
[ON DELETE CASCADE].
```

EX:

```
CREATE TABLE EMP_CHILD
(ENO NUMBER(3),SAL NUMBER(9,2),CONSTRAINT F_EMPNO FOREIGN KEY(EMPNO)
 REFERECES EMP_MAST(ENO) ON DELETE CASCADE);
       OR
CREATE TABLE EMP_CHILD
(ENO NUMBER(3) REFERENCEs EMP_MAST(ENO),SAL NUMBER(9,2)
 ON DELETE CASCADE);

DELETE FROM EMP_MAST WHERE ENO=3;
```

****

SET OPERATORS

1.UNION
2.UNION ALL
3.INTERSECT
4.MINUS

1.**UNION :-**

THIS OPERATOR COMBINES THE RESULT WHICH REMOVES DUPLICATE SELECTED ROWS.

EMP_JOB

ENO	NAME	DESIG	DEPNO	SAL
1	VASU	MANAGER	10	5000
2	RAVI	CLERK	20	3000
3	KSS	CLERK	10	2500
4	VAMSI	SVISOR	30	4000
5	BOSU	CLERK	10	3000
6	RAJU	CLERK	30	2000

```
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=20
UNION
SELECT DESIG FROM EMP_JOB
WHERE DEPNO=30;

20 - CLERK
30 - CLERK,SVISOR
OUTPUT - CLERK,SVISOR

SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=20
UNION
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=30
UNION
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=10;

OUTPUT - CLERK,SVISOR,MANAGER,
```

UNION ALL:

```
----------------
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=20
UNION ALL
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=30
UNION ALL
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=10;

OUTPUT - ALL ROWS FROM DESIG
```

MULTIPLE COLOUMNS:

```
----------------------------------

SELECT DESIG,SAL FROM EMP_JOB
WHERE DEPTNO=10
UNION
SELECT DESIG,SAL FROM EMP_JOB
WHERE DEPTNO=30;

10TH DEPT
-----------------------
MANAGER 5000
CLERK 2500
CLERK 3000

30TH DEPT
------------------
SVISOR 4000
CLERK 2000
CLERK 3000

OUTPUT : CLERK 3000
```

INTERSECTION:

```
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=20
INTERSECT
SELECT DESIG FROM EMP_JOB
WHERE DEPNO=30;
```

20 - CLERK
30 - CLERK,SVISOR
OUTPUT - CLERK

MINUS:

```
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=10
MINUS
SELECT DESIG FROM EMP_JOB
WHERE DEPTNO=30;
```

10 - MANAGER,CLERK,CLERK
30 - SVISOR,CLERK
OUTPUT - MANAGER.

VIEWS

A VIEW IS QUERY OF ANOTHER ONE OR MORE TABLES
THAT PROVIDES ANOTHER WAY OF PRESENTING INFORMATION.

A VIEW DOES NOT CONTINE ANY SPACE TO STORE DATA.

1.VIEW IS A LOGICAL WINDOW.
2.VIEW DOES NOT OCCUPIED ANY MEMORY SPACE.
3.WE CAN USE A VIEW TO PERFORM THE FOLLOWING TASKS

A.MAINTAINE SECURITY
B.RENAME COLOUMNS
C.HIDE COMPLEXITY

WE CAN HAVE THE TWO TYPES OF VIEWS

1.UPDATABLE VIEW
2.READ ONLY VIEW

SYN:

```
CREATE[OR REPLACE][FORCE|NOFORCE] VIEW<FILE_NAME>
[COLOUMN ALIAS,------] AS QUERY;
[WITH CHECK OPTION];
```

```
CREATE OR REPLACE VIEW SAL_VIEW(ITEM_NAME,ITEM_CODE)
AS SELECT ITNAME,ITCODE FROM SALES;
```

MANIPLATION OF VIEW

```
INSERT INTO SAL_VIEW VAUES('LIRIL',789);

SELECT * FROM SAL_VIEW;

UPDATE SAL_VIEW SET ITNAME='RIN'
WHERE IT_NAME='LIRIL';
```

WHERE CONDITION:

```
CREATE OR REPLACE VIEW SAL_VIEW AS SELECT * FROM SALES WHERE ITCODE>7004;

INSERT INTO SAL_VIEW
VALUES('106,'MARGO',7000,20);

SELECT * FROM SALES;
SELECT * FROM SAL_VIEW;
```

FORCE:

```
DROP TABLE EMP;

CREATE OR REPLACE VIEW SAL_VIEW AS SELECT * FROM EMP;

ERROR : TABLE OR VIEW DOES NOT EXIST.

CREATE OR REPLACE FORCE VIEW SAL_VIEW AS SELECT * FROM EMP;

ORACLE WARNNIG :
VIEW CREATED WITH COMPLIATION ERRORS.

CREATE TABLE EMP(ENO NUMBER(3));

INSERT INTO EMP
VALUES(100);

INSERT INTO EMP
VALUES(200);

SELECT * FROM SAL_VIEW;

CREATE A VIEW WITH MULTIPLE TABLES :(READ ONLY)

CREATE OR REPLACE VIEW MT
AS SELECT MASTER.ENO,MASTER.SAL,
STUDENT.NAME FROM MASTER,STUDENT;

SELECT * FROM MT;
```

NOTE : IN THIS WE CANNOT INSERT ANY RECORDS.

WITH CHECK OPTION :

```
CREATE OR REPLACE VIEW SAL_VIEW
AS SELECT IT_NAME,IT_CODE FROM SALES
WHERE ITCODE>=7004;
WITH CHECK OPTION;

INSERT INTO SAL_VIEW(IT_NAME,IT_CODE)
VALUES('LIRIL',3000);
```

ERROR : VIEW WITH CHECK OPTION

GROUP BY CLAUSE :

```
CREATE OR REPLACE VIEW SAMPLE
AS SELECT DEPTNO,SUM(SAL) FROM EMP
GROUP BY DEPTNO;
```

IN THIS ALSO WE CONT ENTER ANY DATA.

DCLC COMMANDS :

DATA CONTROL LANGUAGE COMMANDS

THESE COMMNADS ARE USED TO CONTROLLING AND
ACCESSING THE ORACLE DATABASE.

1.**CREATE USER** : TYPE : SQL

SYN :

```
CREATE USER<USER_NAME>IDENTIFIED BY <PWD>;
```

NOTE : BEFORE USING THIS COMMAND THE USER MUST HAVE DBA
PREVILIZES MEANS DBA.

2.**GRANT** :

SYN :

```
GRANT CONNECT[,RESOURCE FILE][,DBA] TO USER[USER,,,,,]
[IDENTIFIED BY<PWD>[,<PWD>----]];
```

ORACLE HAS TWO CATEGIRES OF PREVILIZES

1.SYSTEM PRE
2.OBJECT PRE

1.SYSTEM PRE: THESE PREVIELIZES ENABLE AN ORECLE
USER TO CONNECT AND EXICUTE STATEMENTS
SUCH AS CREATE USER,CREATE TABLE...

```
CREATE USER PRAKASH IDENTIFIED BY BHANU;

GRANT CONNECT TO PRAKASH;
GRANT CONNECT,RESOURCE TO PRAKASH;
GRANT CONNECT,RESOURCE,DBA TO PRAKASH;
GRANT CONNECT,RESOURCE TO X,Y IDENTIFIED BY A,B;
GRANT CONNECT,RESOURCE,DBA TO X,Y IDENTIFIED BY A,B;

CONNECT SYSTEM/bhanu;
```

2.**SHOW USER;**

```
SHOW<USER>[;]
```

3.EXIT OR QUIT

4.CL SCR

5.CLEAR BUFFER

6.REVOKE T:SQL CAT:DCL

```
REVOKE CONNECT[,RESOURCE][,DBA] FROM <USER>[,USER,,,,];

REVOKE CONNECT FROM PRAKASH;
REVOKE DBA FROM PRAKASH;
```

7.GRANT -II FORM

```
GRANT INSERT[,DELETE][,UPDATE,SELECT,ALTER,INDEX]|ALL ON <TABLE_NAME>
TO USER[,USER,,,,];
```

THIS FORMAT OF GRANT COMMAND GRANTS PREVILIZES TO USERS WITH RESPECT TO TABLE OR VIEWS.
THIS PREVILIZES KNOWN AS OBJECT PREVILIZES.

8.COMMIT :saves the previous actions;

9.ROLLBACK :like undo command.

```
ROLLBACK[,TO SAVE POINT<SAVEPOINT_NAME>]
```

THIS COMMAND IS USED TO DISCARDS[CANCELS] OR UNDOS DML TRANSCATIONS AFTER COMMIT;

NOTE: COMMIT AND ROLLBACK COMMNADS ARE CALLED AS DCLC OR TCLC CATEGORY COMMNADS.

10.SAVEPOINT

THIS COMMAND IS USED TO CREATE THE SAVEPOINT LOCATION.

SYN: SAVEPOINT <SAVEPOINTNAME>

EX:

```
SQL>INSERT 2REC
SQL>UPDATE 2REC
SQL>COMMIT
SQL>INSERT 4REC
SQL>UPDATE 2REC
SQL>ROLLBACK
```

JOINS:

HERE WE HAVE THE 3 TBALE

TABLE 1:

```
SQL> DESC COURSE_FEE
 Name                           Null?    Type
 ------------------------------ -------- ----
 SNO                                     NUMBER(3)
 COURSE                         NOT NULL VARCHAR2(20)
 TFEES                                   NUMBER(9)
-------------------------------------------------
```

COURSE_FEE

SNO	COURSE	TFEES
1	DOA	6000
2	ADCA	16000
3	ADDA	6500
4	HDCA	22000
5	PGDCA	14000
6	C	1500
7	C++	2000
8	MS_OFFICE	1500

TABLE 2:

```
SQL> DESC STUD_INFO
 Name                           Null?    Type
 ------------------------------ -------- ----
 RNO                                     NUMBER(3)
 NAME                                    VARCHAR2(20)
 COURSE                                  VARCHAR2(20)
 FPAID                                   NUMBER(9)
```

STUD_INFO

RNO	NAME	COURSE	FPAID
101	VASU	DOA	2000

```
  102 BHANU                ADDA                        1500
  103 SRINU                DOA                         1000
  104 SIVA                 HDCA                        6000
  105 PRAKASH              DOA                         3000
  106 RAMU                 PGDCA                       2500
-----------------------------------------------------------
```

TABLE 3:

```
SQL> DESC FEE_DIS
 Name                           Null?    Type
 ------------------------------ -------- ----
 COURSE                                  VARCHAR2(20)
 DIS                                     NUMBER(9)

COURSE                    DIS
------------------- ---------
DOA                        20
HDCA                       50
------------------------------
```

JOINs :

Joins are used to combine coloumns from multiple tables.
There are 4 types of joins.

1.Inner joins(Equi joins/Non Equi joins)
2.Outer joins
3.Self Joins
4.Cartician joins.

1.**EQUI-JOINS/NON EQUI-JONS:**

```
SQL>SELECT RNO,NAME,COURSE_FEE.COURSE,TFEES,FPAID,
TFEES-FPAID FROM STUD_INFO,COURSE_FEE
WHERE STUD_INFO.COURSE=COURSE_FEE.COURSE
```

```
  RNO NAME    COURSE                TFEES     FPAID TFEES-FPAID
----- ------------------- -------------------- --------- --------- -----
  101 VASU    DOA                   6000      2000        4000
  102 BHANU   ADDA                  6500      1500        5000
  103 SRINU   DOA                   6000      1000        5000
  104 SIVA    HDCA                 22000      6000       16000
  105 PRAKASH DOA                   6000      3000        3000
  106 RAMU    PGDCA                14000      2500       11500
-------------------------------------------------------------------------
```

2.CARTICIAN JOINS :

```
SQL>SELECT RNO,NAME,COURSE_FEE.COURSE,TFEES,FPAID,
TFEES-FPAID FROM STUD_INFO,COURSE_FEE;
```

```
   RNO NAME   COURSE                TFEES     FPAID TFEES-FPAID
------ ------------------- -------------------- --------- --------- ----
   101 VASU    DOA                   6000      2000        4000
   102 BHANU   DOA                   6000      1500        4500
   103 SRINU   DOA                   6000      1000        5000
   104 SIVA    DOA                   6000      6000           0
```

```
   105 PRAKASH   DOA                    6000      3000        3000
   106 RAMU      DOA                    6000      2500        3500
   101 VASU      ADCA                  16000      2000       14000
   102 BHANU     ADCA                  16000      1500       14500
   103 SRINU     ADCA                  16000      1000       15000
   104 SIVA      ADCA                  16000      6000       10000
   105 PRAKASH   ADCA                  16000      3000       13000
   106 RAMU      ADCA                  16000      2500       13500
   101 VASU      ADDA                   6500      2000        4500
   102 BHANU     ADDA                   6500      1500        5000
   103 SRINU     ADDA                   6500      1000        5500
   104 SIVA      ADDA                   6500      6000         500
   105 PRAKASH   ADDA                   6500      3000        3500
   106 RAMU      ADDA                   6500      2500        4000
   101 VASU      HDCA                  22000      2000       20000
   102 BHANU     HDCA                  22000      1500       20500
   103 SRINU     HDCA                  22000      1000       21000

   RNO NAME     COURSE                TFEES     FPAID TFEES-FPAID
------ -------------------- -------------------- --------- --------- ----
   104 SIVA     HDCA                  22000      6000       16000
   105 PRAKASH  HDCA                  22000      3000       19000
```

3.OUTER JOINS :

```
SQL>SELECT RNO,NAME,COURSE_FEE.COURSE,TFEES,FPAID,
TFEES-FPAID FROM STUD_INFO,COURSE_FEE
WHERE STUD_INFO.COURSE(+)=COURSE_FEE.COURSE;
```

```
   RNO NAME   COURSE                TFEES     FPAID TFEES-FPAID
------ ------------------- -------------------- --------- --------- ----
                           ADCA                    16000
   102 BHANU  ADDA                   6500      1500        5000
              C                      1500
                           C++                      2000
   101 VASU   DOA                    6000      2000        4000
   103 SRINU  DOA                    6000      1000        5000
   105 PRAKASH DOA                    6000      3000        3000
   104 SIVA    HDCA                  22000      6000       16000
               MS_OFFICE              1500
   106 RAMU    PGDCA                 14000      2500       11500
```

4.**SELF JOINS** :by using single table we can create the join.

```
SQL>SELECT * FROM BHANU;

 ENO ENAME                SAL ADR
------ -------------------- ------ ----------------------
     1 BHANU                 6700 R.NAGAR
     2 SRINU                23000 K.NAGAR
     3 RAMU                  4500 J.NAGAR
     4 HANU                  8900 R.NAGAR
     5 SIVA                  6789 K.NAGAR
     8 HANUMAN               4567 E.NAGAR
```

```
SQL>SELECT A1.ENO,A2.ENAME,A2.SAL FROM BHANU A1,BHANU A2
WHERE A1.ENO=A2.ENO

  ENO ENAME                    SAL
----- -------------------- ------
    1 BHANU                  6700
    1 BHANU                  6700
    1 BHANU                  6700
    1 BHANU                  6700
    2 SRINU                 23000
    3 RAMU                   4500
    4 HANU                   8900
    5 SIVA                   6789
    8 HANUMAN                4567
-----------------------------------
```

MULTIPLE TABLES:

```
SELECT RNO,NAME,STUD_INFO.COURSE,TFEES,FPAID,TFEES-FPAID,DIS,TFEES-
(TFEES*DIS/100) ACTUALFEES
FROM STUD_INFO,COURSE_FEE,FEE_DIS
WHERE STUD_INFO.COURSE=COURSE_FEE.COURSE AND
STUD_INFO.COURSE=FEE_DIS.COURSE

 RNO NAME    COURSE      TFEES  FPAID TFEES-FPAID    DIS ACTUALFEES
---- -------------------- ------------------- ------ ------ -----------
 101 VASU    DOA         6000   2000        4000     20       4800
 103 SRINU   DOA         6000   1000        5000     20       4800
 105 PRAKASH DOA         6000   3000        3000     20       4800
 104 SIVA    HDCA        22000   6000       16000     50      11000
-------------------------------------------------------------------------
```

SUBQUERIES :

A Query within anothor query is known as sub query.

1. The result of one query is dynamically substitude in the condition of another query.
2.SQL first evaluetes the sub query(or inner query) within the where clause.
3.The return value of sub query is then substituted in the condition of the outer query.
4.There is no limitation to the level of nesting queries.
5.When using relational operators, ensures that the sub query returns a single row output.

Syntax : SELECT Column_list from table where column=(select column from table where---);

Note : The sub query always must be with in the parenthesis.

Sub query Operators :

1.IN
2.ANY
3.ALL

1.IN :This operator defines set of values in which a value may be existed or not.

SYN : Column_Name[not] IN(value1,value2...)

Ex:SELECT * FROM EMP1 WHERE ESAL IN(SELECT MAX(ESAL) FROM EMP1);

2.ANY : >ANY >=ANY <ANY <=ANY

>=ANY : Greater than or equal to minimum salaries returned by the subquery.

<ANY : In this example it displays all rows from emp1 where the salary is less than max salary returned by a sub query salaries.

<=ANY : Less than or equal to maxsalary returned by a sub query salaries.

3.ALL : >ALL >=ALL <ALL <=ALL

```
SQL> SELECT * FROM EMP1;

       ENO ENAME                DESIG                       ESAL
---------- -------------------- -------------------- ----------
         1 BHANU               MANAGER                    15000
         2 RAMU                CLERK                       4500
         3 HARI                MANAGER                    14500
         4 SIVA                CLERK                       3400
         5 BABU                ACCOUNTANT                  5000
         6 JK                  ACCOUNTANT                  4500
         7 GG                  MANAGER                    17500
         8 SRI                 CLERK                       2300
         9 NANDA               ACCOUNTANT                  4600
        10 TTE                 MANAGER                    16500

10 rows selected.

SQL>  SELECT * FROM EMP1 WHERE ESAL>ANY(SELECT ESAL FROM EMP1 WHERE
DESIG='MANAGER');

      ENO   ENAME                DESIG                       ESAL
      ------ -------------------- -------------------- ----------
         7  GG                   MANAGER                    17500
        10  TTE                  MANAGER                    16500
         1  BHANU                MANAGER                    15000

SQL> SELECT * FROM EMP1 WHERE ESAL>=ANY(SELECT ESAL FROM EMP1 WHERE
DESIG='MANAGER');

       ENO ENAME                DESIG                       ESAL
---------- -------------------- -------------------- ----------
         7 GG                   MANAGER                    17500
        10 TTE                  MANAGER                    16500
         1 BHANU                MANAGER                    15000
         3 HARI                 MANAGER                    14500
```

```
SQL>  SELECT * FROM EMP1 WHERE ESAL<ANY(SELECT ESAL FROM EMP1 WHERE
DESIG='MANAGER');

       ENO ENAME                DESIG                     ESAL
---------- -------------------- -------------------- ----------
         8 SRI                  CLERK                     2300
         4 SIVA                 CLERK                     3400
         2 RAMU                 CLERK                     4500
         6 JK                   ACCOUNTANT                4500
         9 NANDA                ACCOUNTANT                4600
         5 BABU                 ACCOUNTANT                5000
         3 HARI                 MANAGER                  14500
         1 BHANU                MANAGER                  15000
        10 TTE                  MANAGER                  16500

9 rows selected.

SQL>  SELECT * FROM EMP1 WHERE ESAL<=ANY(SELECT ESAL FROM EMP1 WHERE
DESIG='MANAGER');

       ENO ENAME                DESIG                     ESAL
---------- -------------------- -------------------- ----------
         8 SRI                  CLERK                     2300
         4 SIVA                 CLERK                     3400
         2 RAMU                 CLERK                     4500
         6 JK                   ACCOUNTANT                4500
         9 NANDA                ACCOUNTANT                4600
         5 BABU                 ACCOUNTANT                5000
         3 HARI                 MANAGER                  14500
         1 BHANU                MANAGER                  15000
        10 TTE                  MANAGER                  16500
         7 GG                   MANAGER                  17500

10 rows selected.


SQL>  SELECT * FROM EMP1 WHERE ESAL>ALL(SELECT ESAL FROM EMP1 WHERE
DESIG='MANAGER');

no rows selected

SQL>  SELECT * FROM EMP1 WHERE ESAL>=ALL(SELECT ESAL FROM EMP1 WHERE
DESIG='MANAGER');

       ENO ENAME                DESIG                     ESAL
---------- -------------------- -------------------- ----------
         7 GG                   MANAGER                  17500

SQL>

       ENO ENAME                DESIG                     ESAL
---------- -------------------- -------------------- ----------
         2 RAMU                 CLERK                     4500
         4 SIVA                 CLERK                     3400
         5 BABU                 ACCOUNTANT                5000
         6 JK                   ACCOUNTANT                4500
```

```
         8 SRI                  CLERK                      2300
         9 NANDA                ACCOUNTANT                 4600

6 rows selected.

SQL>  SELECT * FROM EMP1 WHERE ESAL>=ALL(SELECT ESAL FROM EMP1 WHERE
DESIG='CLERK');

       ENO ENAME                DESIG                       ESAL
---------- -------------------- -------------------- ----------
         1 BHANU                MANAGER                   15000
         2 RAMU                 CLERK                      4500
         3 HARI                 MANAGER                   14500
         5 BABU                 ACCOUNTANT                 5000
         6 JK                   ACCOUNTANT                 4500
         7 GG                   MANAGER                   17500
         9 NANDA                ACCOUNTANT                 4600
        10 TTE                  MANAGER                   16500

8 rows selected.

SQL>  SELECT * FROM EMP1 WHERE ESAL>ALL(SELECT ESAL FROM EMP1 WHERE
DESIG='CLERK');

       ENO ENAME                DESIG                       ESAL
---------- -------------------- -------------------- ----------
         1 BHANU                MANAGER                   15000
         3 HARI                 MANAGER                   14500
         5 BABU                 ACCOUNTANT                 5000
         7 GG                   MANAGER                   17500
         9 NANDA                ACCOUNTANT                 4600
        10 TTE                  MANAGER                   16500

6 rows selected.

SECOND MAXIMUM SALARY :

SQL> SELECT * FROM EMP1;

  ENO ENAME                DESIG                ESAL
----- -------------------- -------------------- -----
    1 BHANU                MANAGER             15000
    2 RAMU                 CLERK                6666
    3 HARI                 MANAGER             14500
    4 SIVA                 CLERK                3400
    5 BABU                 ACCOUNTANT           5000
    6 JK                   ACCOUNTANT           4500
    7 GG                   MANAGER             17500
    8 SRI                  CLERK                2300
    9 NANDA                ACCOUNTANT           4600
   10 TTE                  MANAGER             16500
   11 KKK                  CLERK                2300

11 rows selected.

SQL> SELECT * FROM EMP1 WHERE ESAL IN(SELECT MAX(ESAL) FROM EMP1);
```

```
ENO ENAME                DESIG                ESAL
----- -------------------- -------------------- -----
    7 GG                   MANAGER              17500

SQL> SELECT * FROM EMP1 WHERE ESAL NOT IN(SELECT MAX(ESAL) FROM EMP1);

  ENO ENAME                DESIG                ESAL
----- -------------------- -------------------- -----
    1 BHANU                MANAGER              15000
    2 RAMU                 CLERK                 6666
    3 HARI                 MANAGER              14500
    4 SIVA                 CLERK                 3400
    5 BABU                 ACCOUNTANT            5000
    6 JK                   ACCOUNTANT            4500
    8 SRI                  CLERK                 2300
    9 NANDA                ACCOUNTANT            4600
   10 TTE                  MANAGER              16500
   11 KKK                  CLERK                 2300

10 rows selected.

SQL> SELECT MAX(ESAL) FROM EMP1 WHERE ESAL NOT IN(SELECT MAX(ESAL) FROM
EMP1);

MAX(ESAL)
---------
    16500

SQL> SELECT * FROM EMP1 WHERE ESAL IN(SELECT MAX(ESAL) FROM EMP1 WHERE
ESAL NOT IN(SELECT MAX(ESAL) FROM EMP1));

  ENO ENAME                DESIG                ESAL
----- -------------------- -------------------- -----
   10 TTE                  MANAGER              16500
```