**DATABASE MANAGEMENT SYSTEM**

**PRACTICAL FILE**

Delhi Technological University

Student: VU DUY DU

Roll No: 2k16/CO/364

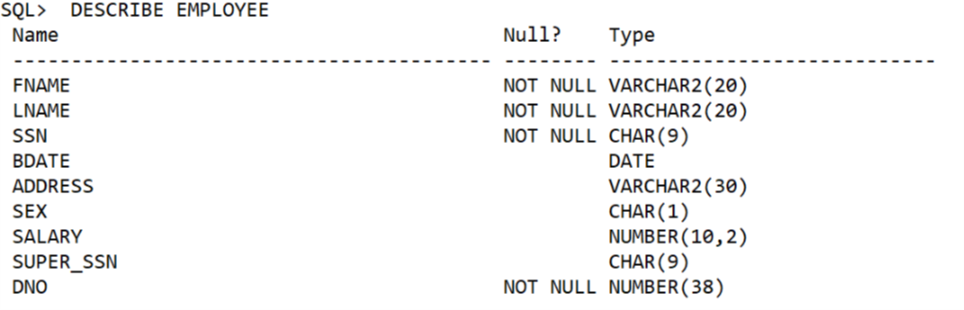
**Practical No. I**

**Introduction:-** Display Schema

**Query:-**

DESCRIBE <table\_name>;

**Result:-**



**Learning:-**

By using this SQL command, we can display the specific schema of a table in the database. The Keyword DESCRIBE can be shorten as DESC.

**Practical No. II**

**Introduction:-** Create Tables: Department, dependent, dept\_locations, employee, project, works\_on

**Query:-**

CREATE TABLE <table\_name>(<attribute\_1> <datatype>, <attribute\_2> <datatype>,…);

SQL> CREATE TABLE Employee(

2 Fname VARCHAR(20) NOT NULL,

3 Lname VARCHAR(20) NOT NULL,

4 Ssn CHAR(9) NOT NULL,

5 Bdate DATE,

6 Address VARCHAR(30),

7 Sex CHAR,

8 Salary DECIMAL(10,2),

9 Super\_ssn CHAR(9),

10 Dno INT NOT NULL,

11 PRIMARY KEY(Ssn),

12 FOREIGN KEY(Super\_Ssn) REFERENCES EMPLOYEE(Ssn),

13 FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber));

SQL> CREATE TABLE DEPARTMENT(

2 Dnumber INT NOT NULL,

3 Dname VARCHAR(15) NOT NULL,

4 Mgr\_Ssn CHAR(9) NOT NULL,

5 Mgr\_start\_date DATE,

6 UNIQUE (Dname),

7 FOREIGN KEY (Mgr\_Ssn) REFERENCES EMPLOYEE(Ssn));

SQL> CREATE TABLE DEPT\_LOCATIONS(

2 Dnumber INT NOT NULL,

3 Dlocation VARCHAR(15) NOT NULL,

4 PRIMARY KEY(Dnumber, Dlocation),

5 FOREIGN KEY(Dnumber) REFERENCES DEPARTMENT(Dnumber));

SQL> CREATE TABLE PROJECT(

2 Pname VARCHAR(15) NOT NULL,

3 Pnumber INT NOT NULL,

4 Plocation VARCHAR(15),

5 Dnum INT NOT NULL,

6 PRIMARY KEY(Pnumber),

7 FOREIGN KEY(Dnum) REFERENCES DEPARTMENT(Dnumber));

SQL> CREATE TABLE WORKS\_ON(

2 Essn CHAR(9) NOT NULL,

3 Pno INT NOT NULL,

4 HOURS DECIMAL(3,1) NOT NULL,

5 PRIMARY KEY(Essn, Pno),

6 FOREIGN KEY(Essn) REFERENCES EMPLOYEE(Ssn),

7 FOREIGN KEY(Pno) REFERENCES PROJECT(Pnumber));

SQL> CREATE TABLE DEPENDENT(

2 Essn CHAR(9) NOT NULL,

3 Dependent\_name VARCHAR(15) NOT NULL,

4 Sex CHAR,

5 Bdate DATE,

6 Relationship VARCHAR(8),

7 PRIMARY KEY(Essn, Dependent\_name),

8 FOREIGN KEY(Essn) REFERENCES EMPLOYEE(Ssn));

**Result:-**

SQL> DESC EMPLOYEE

Name Null? Type

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

FNAME NOT NULL VARCHAR2(20)

LNAME NOT NULL VARCHAR2(20)

SSN NOT NULL CHAR(9)

BDATE DATE

ADDRESS VARCHAR2(30)

SEX CHAR(1)

SALARY NUMBER(10,2)

SUPER\_SSN CHAR(9)

DNO NOT NULL NUMBER(38)

SQL> DESC DEPARTMENT

Name Null? Type

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

DNUMBER NOT NULL NUMBER(38)

DNAME NOT NULL VARCHAR2(15)

MGR\_SSN NOT NULL CHAR(9)

MGR\_START\_DATE DATE

SQL> DESC DEPT\_LOCATIONS;

Name Null? Type

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

DNUMBER NOT NULL NUMBER(38)

DLOCATION NOT NULL VARCHAR2(15)

SQL> DESC PROJECT;

Name Null? Type

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

PNAME NOT NULL VARCHAR2(15)

PNUMBER NOT NULL NUMBER(38)

DNUM NOT NULL NUMBER(38)

PLOCATION VARCHAR2(15)

SQL> DESC WORKS\_ON

Name Null? Type

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

ESSN NOT NULL CHAR(9)

PNO NOT NULL NUMBER(38)

HOURS NOT NULL NUMBER(3,1)

SQL> DESC DEPENDENT

Name Null? Type

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

ESSN NOT NULL CHAR(9)

DEPENDENT\_NAME NOT NULL VARCHAR2(15)

SEX CHAR(1)

BDATE DATE

RELATIONSHIP VARCHAR2(8)

SQL> SELECT TABLE\_NAME FROM USER\_TABLES;

TABLE\_NAME

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

DEPARTMENT

EMPLOYEE

DEPT\_LOCATIONS

PROJECT

WORKS\_ON

DEPENDENT

6 rows selected.

**Learning:-**

To create a new table in SQL we use the command CREATE TABLE. The names of the columns and the datatypes must be specified. Some columns are not null data, they require the entered values are not null. Primary key and foreign key are defined by using constraints:

PRIMARY KEY(<attribute\_name>)

FOREIGN KEY(<attribute\_name>) REFERENCES <other\_table>(<attribute\_name>);

**Practical No. III**

**Introduction:-** Insert values into tables

**Query:-**

INSERT INTO <table\_name> VALUES(<value\_1>,<value\_2>,…);

SQL>INSERT INTO DEPARTMENT(Dname, Dnumber, Mgr\_start\_date) VALUES(‘Headquarters',1,TO\_DATE('1981-06-19','yyyy-mm-dd'));

SQL>INSERT INTO DEPARTMENT(Dname, Dnumber, Mgr\_start\_date) VALUES('Administration',4,TO\_DATE('1995-01-01','yyyy-mm-dd'));

SQL> INSERT INTO DEPARTMENT(Dname, Dnumber, Mgr\_start\_date) VALUES('Research',5,TO\_DATE('1988-05-22','yyyy-mm-dd'));

SQL> INSERT INTO Employee(Fname, Lname, Ssn, Bdate, Address, Sex, Salary, Dno) VALUES('James','Borg',888665555,TO\_DATE('1937-11-10','yyyy-mm-dd'),'450 Stone, Houston,TX','M',55000,1);

SQL> INSERT INTO EMPLOYEE VALUES('Franklin','Wong',333445555,TO\_DATE('1955-12-08','yyyy-mm-dd'),'638 Voss, Houston, TX','M',40000,888665555,5);

SQL> INSERT INTO EMPLOYEE VALUES('Jennifer','Wallace',987654321,TO\_DATE('1941-06-20','yyyy-mm-dd'),'291 Berry,Bellaire, TX','F',43000,888665555,4);

SQL> INSERT INTO EMPLOYEE VALUES('John','Smith',123456789,TO\_DATE('1965-01-09','yyyy-mm-dd'),'731 Fondren, Houston, TX','M',30000,333445555,5);

SQL> INSERT INTO EMPLOYEE VALUES('Ramesh','Narayan',666884444,TO\_DATE('1962-09-15','yyyy-mm-dd'),'975 Fire Oak, Humble, TX','M',38000,333445555,5);

SQL> INSERT INTO EMPLOYEE VALUES('Joyce','English',453453453,TO\_DATE('1972-07-31','yyyy-mm-dd'),'5631 Rice, Houston, TX','F',25000,333445555,5);

SQL> INSERT INTO EMPLOYEE VALUES('Alicia','Zelaya',999887777,TO\_DATE('1968-01-19','yyyy-mm-dd'),'321 Castle, Spring, TX','F',25000,987654321,4);

SQL> INSERT INTO EMPLOYEE VALUES('Ahmad','Jabbar',987987987,TO\_DATE('1969-03-29','yyyy-mm-dd'),'980 Dallas, Houston, TX','M',25000,987654321,4);

SQL> INSERT INTO PROJECT VALUES('ProductX',1,5,'Bellaire');

SQL> INSERT INTO PROJECT VALUES('ProductY',2,5,'Sugarland');

SQL> INSERT INTO PROJECT VALUES('ProductZ',3,5,'Houston');

SQL> INSERT INTO PROJECT VALUES('Computerization',10,4,'Stafford');

SQL> INSERT INTO PROJECT VALUES('Reorganization',20,1,'Houston');

SQL> INSERT INTO PROJECT VALUES('Newbenefits',30,4,'Stafford');

SQL> INSERT INTO DEPT\_LOCATIONS VALUES(1,'Houston');

SQL> INSERT INTO DEPT\_LOCATIONS VALUES(4,'Stafford');

SQL> INSERT INTO DEPT\_LOCATIONS VALUES(5,'Bellaire');

SQL> INSERT INTO DEPT\_LOCATIONS VALUES(5,'Sugarland');

SQL> INSERT INTO DEPT\_LOCATIONS VALUES(5,'Houston');

SQL> INSERT INTO WORKS\_ON VALUES(123456789,1,32.5);

SQL> INSERT INTO WORKS\_ON VALUES(123456789,2,7.5);

SQL> INSERT INTO WORKS\_ON VALUES(666884444,3,40.0);

SQL> INSERT INTO WORKS\_ON VALUES(453453453,1,20.0);

SQL> INSERT INTO WORKS\_ON VALUES(453453453,2,20.0);

SQL> INSERT INTO WORKS\_ON VALUES(333445555,2,10.0);

SQL> INSERT INTO WORKS\_ON VALUES(333445555,3,10.0);

SQL> INSERT INTO WORKS\_ON VALUES(333445555,10,10.0);

SQL> INSERT INTO WORKS\_ON VALUES(333445555,20,10.0);

SQL> INSERT INTO WORKS\_ON VALUES(999887777,30,30.0);

SQL> INSERT INTO WORKS\_ON VALUES(999887777,10,10.0);

SQL> INSERT INTO WORKS\_ON VALUES(987987987,10,35.0);

SQL> INSERT INTO WORKS\_ON VALUES(987987987,30,5.0);

SQL> INSERT INTO WORKS\_ON VALUES(987654321,30,20.0);

SQL> INSERT INTO WORKS\_ON VALUES(987654321,20,15.0);

SQL> INSERT INTO WORKS\_ON(Essn,Pno) VALUES(888665555,20);

SQL> INSERT INTO DEPENDENT VALUES(333445555,'Alice','F',TO\_DATE('1986-04-05','yyyy-mm-dd'),'Daughter');

SQL> INSERT INTO DEPENDENT VALUES(333445555,'Theodore','M',TO\_DATE('1983-10-25','yyyy-mm-dd'),'Son');

SQL> INSERT INTO DEPENDENT VALUES(333445555,'Joy','F',TO\_DATE('1958-05-03','yyyy-mm-dd'),'Spouse');

SQL> INSERT INTO DEPENDENT VALUES(987654321,'Abner','M',TO\_DATE('1942-02-28','yyyy-mm-dd'),'Spouse');

SQL> INSERT INTO DEPENDENT VALUES(123456789,'Michael','M',TO\_DATE(‘1988-01-04','yyyy-mm-dd'),’Son');

SQL> INSERT INTO DEPENDENT VALUES(123456789,'Alice','F',TO\_DATE('1988-12-30','yyyy-mm-dd'),'Daughter');

SQL> INSERT INTO DEPENDENT VALUES(123456789,'Elizabeth','F',TO\_DATE('1967-05-05','yyyy-mm-dd'),'Spouse');

**Learning:-**

By inserting values in to tables, the database is almost complete, and ready to be used. For the null values, we have to list the column name of the table which is not null, and then insert values into that particular column. The unlisted column name is the null value to be entered.

**Practical No. IV**

**Introduction:-** Display all tables

**Query:-**

SELECT \* FROM <table\_name>;

**Result:-**

SQL> SELECT \* FROM EMPLOYEE;

FNAME LNAME SSN BDATE

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

ADDRESS S SALARY SUPER\_SSN DNO

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

James Borg 888665555 10-NOV-37

450 Stone, Houston,TX M 55000 1

Franklin Wong 333445555 08-DEC-55

638 Voss, Houston, TX M 40000 888665555 5

John Smith 123456789 09-JAN-65

731 Fondren, Houston, TX M 30000 333445555 5

Ramesh Narayan 666884444 15-SEP-62

975 Fire Oak, Humble, TX M 38000 333445555 5

Joyce English 453453453 31-JUL-72

5631 Rice, Houston, TX F 25000 333445555 5

Jennifer Wallace 987654321 20-JUN-41

291 Berry,Bellaire, TX F 43000 888665555 4

Alicia Zelaya 999887777 19-JAN-68

321 Castle, Spring, TX F 25000 987654321 4

Ahmad Jabbar 987987987 29-MAR-69

980 Dallas, Houston, TX M 25000 987654321 4

8 rows selected.

SQL> SELECT \* FROM DEPARTMENT;

DNUMBER DNAME MGR\_SSN MGR\_START

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

1 Headquarters 888665555 19-JUN-81

4 Administration 987654321 01-JAN-95

5 Research 333445555 22-MAY-88

SQL> SELECT \* FROM PROJECT;

PNAME PNUMBER DNUM PLOCATION

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

ProductX 1 5 Bellaire

ProductY 2 5 Sugarland

ProductZ 3 5 Houston

Computerization 10 4 Stafford

Reorganization 20 1 Houston

Newbenefits 30 4 Stafford

SQL> SELECT \* FROM DEPT\_LOCATIONS;

DNUMBER DLOCATION

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

1 Houston

4 Stafford

5 Bellaire

5 Houston

5 Sugarland

SQL> select \* from works\_on;

ESSN PNO HOURS

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

123456789 1 32.5

123456789 2 7.5

666884444 3 40

453453453 1 20

453453453 2 20

333445555 2 10

333445555 3 10

333445555 10 10

333445555 20 10

999887777 30 30

999887777 10 10

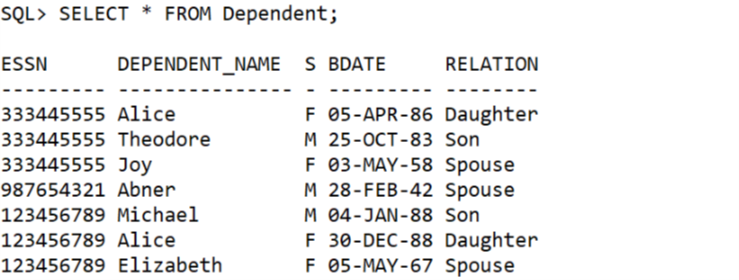
987987987 10 35

987987987 30 5

987654321 30 20

987654321 20 15

888665555 20



**Learning:-**

By using SELECT key word, we can display any row and column in the tables. To display the whole table we use symbol ‘\*’ instead of the column name.

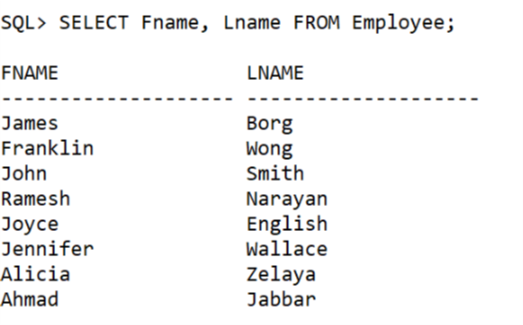
**Practical No. V**

**Introduction:-** Find names of all employees

**Query:-**

SELECT Fname, Lname FROM Employee;

**Result:-**



**Learning:-**

SELECT keyword allows to display the information in a table. Columns are selected according to the list of column names after SELECT keyword.

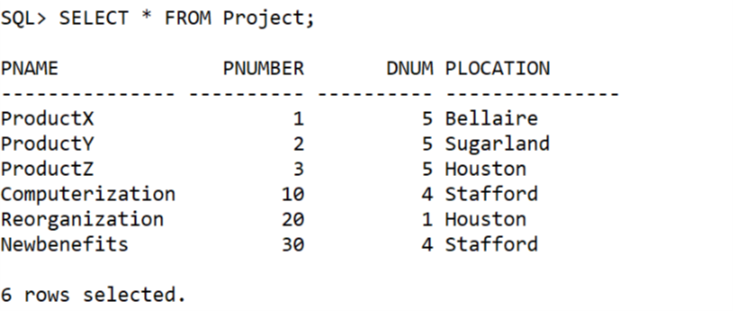
**Practical No. VI**

**Introduction:-** Print entire project table

**Query:-**

SELECT \* FROM Project;

**Result:-**



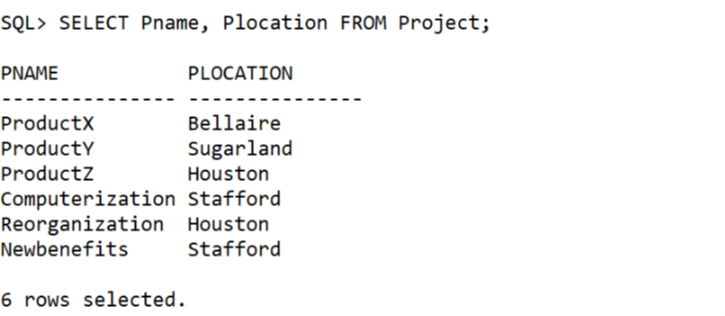
**Practical No. VII**

**Introduction:-** Retrieve list of names and locations of all projects

**Query:-**

SELECT Pname, Plocation FROM Project;

**Result:-**



**Practical No. VIII**

**Introduction:-** Find names of all projects having ‘a’ as the second letter in their names.

**Query:-**

SELECT Pname FROM Project WHERE Pname LIKE ‘\_a%’;

**Result:-**

No rows selected.

**Learning:-**

WHERE key word allows to apply condition for the clause. LIKE keyword is used to compare the strings. Here to find the string having letter ‘a’ at the second position, we use ‘\_‘ at the first position of the string to specify this position can be any character. We use ‘%’ at the end position to replace for other characters.

**Practical No. IX**

**Introduction:-** Find Employees who stay in city whose second letter is ‘a’.

**Query:-**

SELECT Fname, Lname FROM Employee WHERE Addess LIKE ‘%,\_a%’;

**Result:-**

No rows selected.

**Learning:-**

City name is at the position after the first comma, thus we use the comparing string ‘%,\_a%’ for the condition city whose second letter is ‘a’.

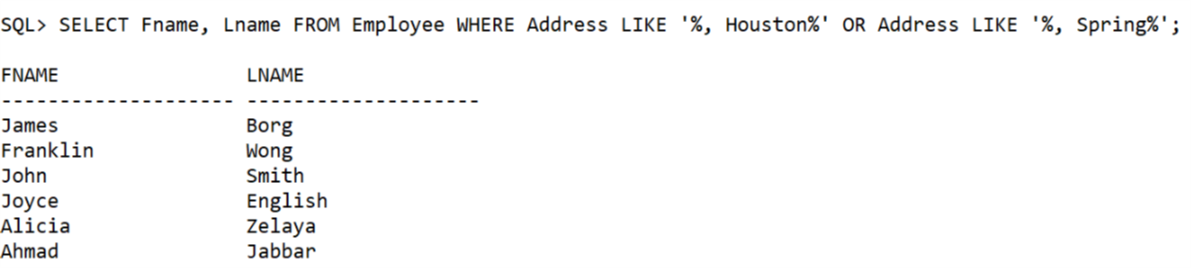
**Practical No. X**

**Introduction:-** Find Employees who stay Houston or Spring.

**Query:-**

SELECT Fname, Lname FROM Employee WHERE Address LIKE ‘%, Houston%’ OR Address LIKE ‘%, Spring%’;

**Result:-**



**Learning:-**

OR operation is used for logical expression in the condition of the statement. AND operation is used in the same way.

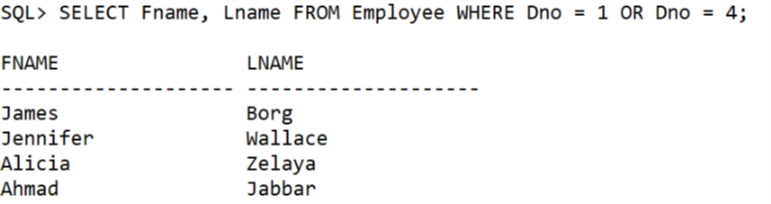
**Practical No. XI**

**Introduction:-** Find List of all employees whose dept no. is 1 or 4

**Query:-**

SELECT Fname, Lname FROM Employee WHERE Dno = 1 OR Dno = 4;

**Result:-**



**Learning:-**

OR operation is used for logical expression in the condition of the statement. AND operation is used in the same way.

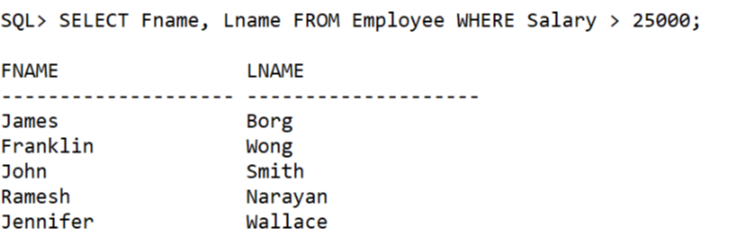
**Practical No. XII**

**Introduction:-** Find all employees whose salary is greater than 25000.

**Query:-**

SELECT Fname, Lname FROM Employee WHERE Salary > 25000;

**Result:-**



**Learning:-**

Greater than ‘>’ operator is used to compared the values in the WHERE clause. Other comparators are ‘<’, ‘<=’,‘>=’,’<>’