**Introduction to DBMS Lab**

**About SQL**

SQL is short for Structured Query Language.SQL is the standard language used to communicate with relational database. SQL is a simple English like language that provides a means of data manipulation (store, retrieve, update, delete) and database creation. The SQL was first developed by IBM in 1970’s for the DB2 platform RDBMS.

Some examples of modern Relational Database Management Systems are MS SQL Server, Microsoft Access, MSDE, Oracle, DB2, Sybase, MySQL, Postgres and Informix. SQL is the standard language used for all the commercial RDBMS’s. MySQL and PostgreSQL are open source RDBMS. Although all those RDBMS use SQL, they use different SQL dialects. For example MS SQL Server specific version of the SQL is called T-SQL, Oracle version of SQL is called PL/SQL, MS Access version of SQL is called JET SQL, etc.

The foundation of every Relational Database Management System is a database object called table. Every database consists of one or more tables, which store the database’s data/information. Each table has its own unique name and consists of columns and rows. The database table columns (called also table fields) have their own unique names and have pre-defined data types. Table columns can have various attributes defining the column functionality (the column is primary key, there is an index defined on the column has certain default value, etc). While table columns describe the data types, the table rows contain the actual data for the columns.

SQL is a comprehensive database language i.e it has statements for the data definitions, queries and updates. Hence, it is a Data Definition Language (DML), a Data Manipulation Language(DML). SQL also provides statements security of database. Hence it is also a Data Control Language(DCL). SQL uses the terms table, row and column for the formal relational model terms relation, tuple and attributes respectively.

In 1986, ANSI released a standard version of SQL called SQL-86 or SQL1. The next version was SQL-89. A revised and much expanded standard called SQL-2 or SQL-92 was developed. The next standard that is well recognized is SQL-99.

**E-R Diagram**

The E-R diagram represents the conceptual database viewed by the end user. ERDs depict the database’s main components: entities, attributes and relationships. The notations used to draw E-R diagrams are shown in the diagram.

**Relational Database Schema**

The description of a database is called as database schema. The database schema is specified during database design. A displayed schema is called a schema diagram. Each object or table or relation in the schema is called a schema entity. Foe Ex, The different schema entities for the database COMPANY are EMPLOYEE, DEPARTMENT, DEPT\_LOCATIONS, PROJECTS, WORKS\_ON and DEPENDENT.

**EXPERIMENT NO: 01**

**Problem definition:**

**Create the following tables for a COMPANY DATABASE**

1. **EMPLOYEE**
2. **DEPARTMENT**
3. **DEPT-LOCATIONS**
4. **PROJECT**
5. **WORKS ON**
6. **DEPENDENT**

**EXPERIMENT NO: 01**

**Create Table:**

This command is used to create table using different attributes name type size.

**SYNTAX:**

(CREATE TABLE TABLE\_NAME

ATTRIBUTE\_NAME\_1 DATA\_TYPE(SIZE),

ATTRIBUTE\_NAME\_2 DATA\_TYPE(SIZE)

ATTRIBUTE\_NAME\_N DATA\_TYPE(SIZE));

**TABLE: - EMPLOYEE**

SQL> CREATE TABLE EMPLOYEE(FNAME VARCHAR(15) NOT NULL,

MNAME CHAR(1),LNAME VARCHAR(15) NOT NULL,

SSN CHAR(9) NOT NULL,BDATE DATE,

ADDRESS VARCHAR(25),SEX CHAR(1),

SALARY NUMBER(10,2), SUPER\_SSN CHAR(9),

DNO INT NOT NULL,PRIMARY KEY(SSN));

Table created.

SQL> DESC EMPLOYEE;

Name Null? Type

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

FNAME NOT NULL VARCHAR2(15)

MNAME CHAR(1)

LNAME NOT NULL VARCHAR2(15)

SSN NOT NULL CHAR(9)

BDATE DATE

ADDRESS VARCHAR2(25)

SEX CHAR(1)

SALARY NUMBER(10,2)

SUPER\_SSN CHAR(9)

DNO NOT NULL NUMBER(38)

**TABLE: DEPARTMENT**

SQL> CREATE TABLE DEPARTMENT(DNAME VARCHAR(15) NOT NULL, DNUMBER INT NOT NULL, MGR\_SSN CHAR(9) NOT NULL,

MGR\_START\_DATE DATE, PRIMARY KEY(DNUMBER), UNIQUE(DNAME), FOREIGN KEY(MGR\_SSN) REFERENCES EMPLOYEE(SSN));

Table created.

SQL> DESC DEPARTMENT;

Name Null? Type

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

DNAME NOT NULL VARCHAR2(15)

DNUMBER NOT NULL NUMBER(38)

MGR\_SSN NOT NULL CHAR(9)

MGR\_START\_DATE DATE

**TABLE: DEPT\_LOCATIONS**

SQL> CREATE TABLE DEPT\_LOCATIONS (DNUMBER INT NOT NULL,

DLOCATION VARCHAR(15) NOT NULL, PRIMARY KEY(DNUMBER,DLOCATION),

FOREIGN KEY(DNUMBER) REFERENCES DEPARTMENT(DNUMBER));

Table created.

SQL> DESC DEPT\_LOCATIONS;

Name Null? Type

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

DNUMBER NOT NULL NUMBER(38)

DLOCATION NOT NULL VARCHAR2(15)

**TABLE: PROJECT**

SQL> CREATE TABLE PROJECT(PNAME VARCHAR(15) NOT NULL,

PNUMBER INT NOT NULL, PLOCATION VARCHAR(15),

DNUM INT NOT NULL,PRIMARY KEY(PNUMBER),

UNIQUE(PNAME),FOREIGN KEY(DNUM) REFERENCES DEPARTMENT(DNUMBER));

Table created.

SQL> DESC PROJECT;

Name Null? Type

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

PNAME NOT NULL VARCHAR2(15)

PNUMBER NOT NULL NUMBER(38)

PLOCATION VARCHAR2(15)

DNUM NOT NULL NUMBER(38)

**TABLE: WORKS\_ON**

SQL> create table works\_on

(essn char(9) not null,

pno int not null,

hours decimal(4,1) not null,

foreign key(essn) references employee(ssn),

foreign key(pno) references project(pnumber));

Table created.

SQL> desc works\_on;

Name Null? Type

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

ESSN NOT NULL CHAR(9)

PNO NOT NULL NUMBER(38)

HOURS NOT NULL NUMBER(4,1)

**TABLE: DEPENDENT**

SQL> create table dependent

(essn char(9) not null,

dependent\_name varchar(15) not null,

sex char(1),

bdate date,

relationship varchar(8),

primary key(essn,dependent\_name),

foreign key(essn) references employee(ssn));

Table created.

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)

**EXPERIMENT NO: 02**

**CREATE THE FOLLOWING TABLES FOR A COMPANY DATABASE**

1. **EMPLOYEE**
2. **DEPARTMENT**
3. **DEPT\_LOCATION**
4. **PROJECT**
5. **WORKS\_ON**
6. **DEPENDENT**

**Using the following constraints**

**- Not null**

**- Primary key**

**- Unique**

**- check**

**- Default**

**- References**

**SQL NOT NULL Constraint**

The NOT NULL constraint uniquely enforces a column to NOT accept NULL values. The NOT NULL constraint enforces a field to always contain a value. This means that you cannot insert a new record or update a record without adding a value to this field.

**SQL UNIQUE Constraint**

The UNIQUE constraint uniquely identifies each record in a database table. The UNIQUE and PRIMARY KEY constraints both provide a guarantee for uniqueness for a column or set columns. A PRIMARY KEY constraint automatically has UNIQUE constraint defined on it. NOTE that you can have many UNIQUE constraints per table but only one PRIMARY KEY constraints per table.

**SQL PRIMARY KEY Constraint**

The PRIMARY KEY constraint uniquely identifies each record in a database table. Primary keys must contain unique values. A primary key column cannot contain NULL values. Each table should have a primary key and each table can have only ONE primary key.

**SQL FOREIGN KEY Constraint**

A FOREIGN KEY in one table points to a PRIMARY KEY in another table.

**SQLCHECK Constraint**

The CHECK constraint is used to limit the values range that can be placed in a column. If you define a CHECK constraint on single column it allows only certain values for this column. If you define a CHECK constraint on a table it can limit the values in certain column based on values in other columns in the row.

**TABLE: - EMPLOYEE**

SQL> CREATE TABLE EMPLOYEE(FNAME VARCHAR(15) NOT NULL,

MNAME CHAR(1),LNAME VARCHAR(15) NOT NULL,

SSN CHAR(9) NOT NULL,BDATE DATE,

ADDRESS VARCHAR(25),SEX CHAR(1),

SALARY NUMBER(10,2), SUPER\_SSN CHAR(9),

DNO INT NOT NULL,PRIMARY KEY(SSN));

Table created.

SQL> DESC EMPLOYEE;

Name Null? Type

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

FNAME NOT NULL VARCHAR2(15)

MNAME CHAR(1)

LNAME NOT NULL VARCHAR2(15)

SSN NOT NULL CHAR(9)

BDATE DATE

ADDRESS VARCHAR2(25)

SEX CHAR(1)

SALARY NUMBER(10,2)

SUPER\_SSN CHAR(9)

DNO NOT NULL NUMBER(38)

**TABLE: DEPARTMENT**

SQL> CREATE TABLE DEPARTMENT(DNAME VARCHAR(15) NOT NULL, DNUMBER INT NOT NULL, MGR\_SSN CHAR(9) NOT NULL,

MGR\_START\_DATE DATE, PRIMARY KEY(DNUMBER), UNIQUE(DNAME), FOREIGN KEY(MGR\_SSN) REFERENCES EMPLOYEE(SSN));

Table created.

SQL> DESC DEPARTMENT;

Name Null? Type

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

DNAME NOT NULL VARCHAR2(15)

DNUMBER NOT NULL NUMBER(38)

MGR\_SSN NOT NULL CHAR(9)

MGR\_START\_DATE DATE

**EXPERIMENT NO: 03**

**Problem definition: -** Inserting values into a table.

**Aim: -** To illustrate the use of insert command.

**Theory: -**

Data manipulation language (DML) is a part of a SQL that manipulates the data in the database. The different DML statements are SELECT, INSERT, DELETE & UPDATE.

One form of insert command is used to add single tuple to a relation we need to specify relation name & a list of values for the tuple.

The syntax is as follows:-

**Syntax:**

INSERT INTO table\_name.

VALUES (value1, value2, value3………..);

The second form of insert command allows the user to specify the attributes names & the corresponding values.

The syntax is as follows:-

INSERT INTO table\_name (column1, column2, column3……)

Values (value1, value2, value3………..);

**EXPERIMENT NO: 03**

**INSERT VALUES INTO THE FOLLOWING TABLES FOR A COMPANY DATABASE**

1. **EMPLOYEE**
2. **DEPARTMENT**
3. **DEPT\_LOCATION**
4. **PROJECT**
5. **WORKS\_ON**
6. **DEPENDENT**

**(TABLES CREATED IN PREVIOUS EXPERIMENT.)**

SQL> select \*from tab;

TNAME TABTYPE CLUSTERID

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

DEPARTMENT TABLE

DEPT\_LOCATIONS TABLE

DEPENDENT TABLE

EMPLOYEE TABLE

PROJECT TABLE

WORKS\_ON TABLE

6 rows selected.

**TABLE: EMPLOYEE**

SQL> /

Enter value for fname: JOHN

Enter value for mname: B

Enter value for lname: SMITH

Enter value for ssn: 1231

Enter value for bdate: 09-JAN-1965

Enter value for address: 731,FONDREN,HOUSTON

old 1: INSERT INTO EMPLOYEE VALUES('&FNAME','&MNAME','&LNAME','&SSN','&BDATE','&ADDRESS',

new 1: INSERT INTO EMPLOYEE VALUES('JOHN','B','SMITH','1231','09-JAN-1965','731,FONDREN,HOUSTON',

Enter value for sex: M

Enter value for salary: 30000

Enter value for super\_ssn: 1231

Enter value for dno: 1

old 2: '&SEX',&SALARY,'&SUPER\_SSN',&DNO)

new 2: 'M',30000,'1231',1)

1 row created.

SQL> /

Enter value for fname: JOHN

Enter value for mname: B

Enter value for lname: SMITH

Enter value for ssn: 1231

Enter value for bdate: 09-JAN-1965

Enter value for address: 731,FONDREN,HOUSTON

old 1: INSERT INTO EMPLOYEE VALUES('&FNAME','&MNAME','&LNAME','&SSN','&BDATE','&ADDRESS',

new 1: INSERT INTO EMPLOYEE VALUES('JOHN','B','SMITH','1231','09-JAN-1965','731,FONDREN,HOUSTON',

Enter value for sex: M

Enter value for salary: 30000

Enter value for super\_ssn: 1231

Enter value for dno: 1

old 2: '&SEX',&SALARY,'&SUPER\_SSN',&DNO)

new 2: 'M',30000,'1231',1)

1 row created.

SQL> /

Enter value for fname: FRANKLIN

Enter value for mname: T

Enter value for lname: WONG

Enter value for ssn: 1232

Enter value for bdate: 08-DEC-1955

Enter value for address: 638,VOSS,HOUSTON

old 1: INSERT INTO EMPLOYEE VALUES('&FNAME','&MNAME','&LNAME','&SSN','&BDATE','&ADDRESS',

new 1: INSERT INTO EMPLOYEE VALUES('FRANKLIN','T','WONG','1232','08-DEC-1955','638,VOSS,HOUSTON',

Enter value for sex: M

Enter value for salary: 40000

Enter value for super\_ssn: 1232

Enter value for dno: 2

old 2: '&SEX',&SALARY,'&SUPER\_SSN',&DNO)

new 2: 'M',40000,'1232',2)

1 row created.

SQL> SELECT \*FROM EMPLOYEE;

FNAME M LNAME SSN BDATE ADDRESS S SALARY

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

SUPER\_SSN DNO

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

JOHN B SMITH 1231 09-JAN-65 731,FONDREN,HOUSTON M 30000

1231 1

FRANKLIN T WONG 1232 08-DEC-55 638,VOSS,HOUSTON M 40000

1232 2

ALICIA J ZALAYA 1233 19-JAN-68 3321,CASTLE,HOUSTON F 25000

1233 3

JENNIFER S WALLACA 1234 20-JUL-74 291,BERRY,HUMBLE F 43000

1234 4

SUSHANT Y KAMBLE 1235 21-APR-97 ABBIHAL,ATHANI M 38000

1235 1

**TABLE: DEPARTMENT**

SQL> INSERT INTO DEPARTMENT VALUES('&DNAME',&DNUMBER,'&MGR\_SSN','&MGR\_START\_DATE');

Enter value for dname: RESEARCH

Enter value for dnumber: 1

Enter value for mgr\_ssn: 1231

Enter value for mgr\_start\_date: 22-JAN-1988

old 1: INSERT INTO DEPARTMENT VALUES('&DNAME',&DNUMBER,'&MGR\_SSN','&MGR\_START\_DATE')

new 1: INSERT INTO DEPARTMENT VALUES('RESEARCH',1,'1231','22-JAN-1988')

1 row created.

SQL> /

Enter value for dname: ADMINISTRATION

Enter value for dnumber: 2

Enter value for mgr\_ssn: 1232

Enter value for mgr\_start\_date: 01-JAN-1995

old 1: INSERT INTO DEPARTMENT VALUES('&DNAME',&DNUMBER,'&MGR\_SSN','&MGR\_START\_DATE')

new 1: INSERT INTO DEPARTMENT VALUES('ADMINISTRATION',2,'1232','01-JAN-1995')

1 row created.

SQL> /

Enter value for dname: HEADQUARTERS

Enter value for dnumber: 3

Enter value for mgr\_ssn: 1233

Enter value for mgr\_start\_date: 19-JUL-1968

old 1: INSERT INTO DEPARTMENT VALUES('&DNAME',&DNUMBER,'&MGR\_SSN','&MGR\_START\_DATE')

new 1: INSERT INTO DEPARTMENT VALUES('HEADQUARTERS',3,'1233','19-JUL-1968')

1 row created.

SQL> /

Enter value for dname: COMPUTERIZATION

Enter value for dnumber: 4

Enter value for mgr\_ssn: 1234

Enter value for mgr\_start\_date: 18-JAN-1992

old 1: INSERT INTO DEPARTMENT VALUES('&DNAME',&DNUMBER,'&MGR\_SSN','&MGR\_START\_DATE')

new 1: INSERT INTO DEPARTMENT VALUES('COMPUTERIZATION',4,'1234','18-JAN-1992')

1 row created.

SQL> SELECT \*FROM DEPARTMENT;

DNAME DNUMBER MGR\_SSN MGR\_START

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

RESEARCH 1 1231 22-JAN-88

ADMINISTRATION 2 1232 01-JAN-95

HEADQUARTERS 3 1233 19-JUL-68

COMPUTERIZATION 4 1234 18-JAN-92

**TABLE: DEPT\_LOCATIONS**

**SQL>descdept\_locations;**

Name Null? Type

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

DNUMBER NOT NULL NUMBER(38)

DLOCATION NOT NULL VARCHAR2(15)

SQL> INSERT INTO DEPT\_LOCATIONS VALUES(&DNUMBER,'&DLOCATION');

Enter value for dnumber: 1

Enter value for dlocation: BANGALORE

old 1: INSERT INTO DEPT\_LOCATIONS VALUES(&DNUMBER,'&DLOCATION')

new 1: INSERT INTO DEPT\_LOCATIONS VALUES(1,'BANGALORE')

1 row created.

SQL> /

Enter value for dnumber: 2

Enter value for dlocation: MUMBAI

old 1: INSERT INTO DEPT\_LOCATIONS VALUES(&DNUMBER,'&DLOCATION')

new 1: INSERT INTO DEPT\_LOCATIONS VALUES(2,'MUMBAI')

1 row created.

SQL> /

Enter value for dnumber: 3

Enter value for dlocation: HUBLI

old 1: INSERT INTO DEPT\_LOCATIONS VALUES(&DNUMBER,'&DLOCATION')

new 1: INSERT INTO DEPT\_LOCATIONS VALUES(3,'HUBLI')

1 row created.

SQL> /

Enter value for dnumber: 4

Enter value for dlocation: PUNE

old 1: INSERT INTO DEPT\_LOCATIONS VALUES(&DNUMBER,'&DLOCATION')

new 1: INSERT INTO DEPT\_LOCATIONS VALUES(4,'PUNE')

1 row created.

SQL> /

Enter value for dnumber: 1

Enter value for dlocation: ATHANI

old 1: INSERT INTO DEPT\_LOCATIONS VALUES(&DNUMBER,'&DLOCATION')

new 1: INSERT INTO DEPT\_LOCATIONS VALUES(1,'ATHANI')

1 row created.

SQL> SELECT \*FROM DEPT\_LOCATIONS;

DNUMBER DLOCATION

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

1 BANGALORE

2 MUMBAI

3 HUBLI

4 PUNE

1 ATHANI

**TABLE: PROJECT**

**SQL>desc project;**

Name Null? Type

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

PNAME NOT NULL VARCHAR2(15)

PNUMBER NOT NULL NUMBER(38)

PLOCATION VARCHAR2(15)

DNUM NOT NULL NUMBER(38)

SQL> INSERT INTO PROJECT VALUES('&PNAME',&PNUMBER,'&PLOCATION',&DNUM);

Enter value for pname: ProductX

Enter value for pnumber: 1

Enter value for plocation: Bangalore

Enter value for dnum: 1

old 1: INSERT INTO PROJECT VALUES('&PNAME',&PNUMBER,'&PLOCATION',&DNUM)

new 1: INSERT INTO PROJECT VALUES('ProductX',1,'Bangalore',1)

1 row created.

SQL> /

Enter value for pname: ProductY

Enter value for pnumber: 2

Enter value for plocation: Mumbai

Enter value for dnum: 2

old 1: INSERT INTO PROJECT VALUES('&PNAME',&PNUMBER,'&PLOCATION',&DNUM)

new 1: INSERT INTO PROJECT VALUES('ProductY',2,'Mumbai',2)

1 row created.

SQL> /

Enter value for pname: ProductZ

Enter value for pnumber: 3

Enter value for plocation: Pune

Enter value for dnum: 4

old 1: INSERT INTO PROJECT VALUES('&PNAME',&PNUMBER,'&PLOCATION',&DNUM)

new 1: INSERT INTO PROJECT VALUES('ProductZ',3,'Pune',4)

1 row created.

SQL> /

Enter value for pname: Computerisation

Enter value for pnumber: 4

Enter value for plocation: belgaum

Enter value for dnum: 3

old 1: INSERT INTO PROJECT VALUES('&PNAME',&PNUMBER,'&PLOCATION',&DNUM)

new 1: INSERT INTO PROJECT VALUES('Computerisation',4,'belgaum',3)

1 row created.

SQL> /

Enter value for pname: Reorganisation

Enter value for pnumber: 5

Enter value for plocation: Hubli

Enter value for dnum: 3

old 1: INSERT INTO PROJECT VALUES('&PNAME',&PNUMBER,'&PLOCATION',&DNUM)

new 1: INSERT INTO PROJECT VALUES('Reorganisation',5,'Hubli',3)

1 row created.

SQL> select \*from project;

PNAME PNUMBER PLOCATION DNUM

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

ProductX 1 Bangalore 1

ProductY 2 Mumbai 2

ProductZ 3 Pune 4

Computerisation 4 belgaum 3

**TABLE: WORKS\_ON**

**SQL>descworks\_on;**

Name Null? Type

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

ESSN NOT NULL CHAR(9)

PNO NOT NULL NUMBER(38)

HOURS NOT NULL NUMBER(3,1)

SQL> insert into works\_on values('&Essn',&pno,&hours);

Enter value for essn: 1231

Enter value for pno: 1

Enter value for hours: 30.5

old 1: insert into works\_on values('&Essn',&pno,&hours)

new 1: insert into works\_on values('1231',1,30.5)

1 row created.

SQL> /

Enter value for essn: 1232

Enter value for pno: 2

Enter value for hours: 14.6

old 1: insert into works\_on values('&Essn',&pno,&hours)

new 1: insert into works\_on values('1232',2,14.6)

1 row created.

SQL> /

Enter value for essn: 1233

Enter value for pno: 3

Enter value for hours: 52.4

old 1: insert into works\_on values('&Essn',&pno,&hours)

new 1: insert into works\_on values('1233',3,52.4)

1 row created.

SQL> /

Enter value for essn: 1234

Enter value for pno: 4

Enter value for hours: 29.4

old 1: insert into works\_on values('&Essn',&pno,&hours)

new 1: insert into works\_on values('1234',4,29.4)

1 row created.

SQL> /

Enter value for essn: 1235

Enter value for pno: 5

Enter value for hours: 11.3

old 1: insert into works\_on values('&Essn',&pno,&hours)

new 1: insert into works\_on values('1235',5,11.3)

1 row created.

SQL> select \*from works\_on;

ESSN PNO HOURS

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

1231 1 30.5

1232 2 14.6

1233 3 52.4

1234 4 29.4

1235 5 11.3

**TABLE: DEPENDENT**

**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> insert into dependent values('&Essn','&dependent\_name','&sex','&bdate','&relationship');

Enter value for essn: 1231

Enter value for dependent\_name: SAMARTH

Enter value for sex: M

Enter value for bdate: 12-JAN-1995

Enter value for relationship: SON

old 1: insert into dependent values('&Essn','&dependent\_name','&sex','&bdate','&relationship')

new 1: insert into dependent values('1231','SAMARTH','M','12-JAN-1995','SON')

1 row created.

SQL> /

Enter value for essn: 1232

Enter value for dependent\_name: BHARATI]

Enter value for sex:

Enter value for bdate:

Enter value for relationship:

old 1: insert into dependent values('&Essn','&dependent\_name','&sex','&bdate','&relationship')

new 1: insert into dependent values('1232','BHARATI]','','','')

1 row created.

SQL> /

Enter value for essn: 1233

Enter value for dependent\_name: CHANDRAKANT

Enter value for sex: M

Enter value for bdate: 15-JUN-1975

Enter value for relationship: FATHER

old 1: insert into dependent values('&Essn','&dependent\_name','&sex','&bdate','&relationship')

new 1: insert into dependent values('1233','CHANDRAKANT','M','15-JUN-1975','FATHER')

1 row created.

SQL> /

Enter value for essn: 1234

Enter value for dependent\_name: SMITHA

Enter value for sex: F

Enter value for bdate: 15-SEP-2000

Enter value for relationship: DAUGHTER

old 1: insert into dependent values('&Essn','&dependent\_name','&sex','&bdate','&relationship')

new 1: insert into dependent values('1234','SMITHA','F','15-SEP-2000','DAUGHTER')

1 row created.

SQL> /

Enter value for essn: 1235

Enter value for dependent\_name: VIJAYALAXMI

Enter value for sex: F

Enter value for bdate: 14-JAN-1980

Enter value for relationship: MOTHER

old 1: insert into dependent values('&Essn','&dependent\_name','&sex','&bdate','&relationship')

new 1: insert into dependent values('1235','VIJAYALAXMI','F','14-JAN-1980','MOTHER')

1 row created.

SQL> SELECT \*FROM DEPENDENT;

ESSN DEPENDENT\_NAME S BDATE RELATIONSHIP

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

1231 SAMARTH M 12-JAN-95 SON

1232 BHARATI F 15-FEB-85 WIFE

1233 CHANDRAKANT M 15-JUN-75 FATHER

1234 SMITHA F 15-SEP-00 DAUGHTER

1235 VIJAYALAXMI F 14-JAN-80 MOTHER

**EXPERIMENT NO: 04**

**SELECT COMMAND:**It selects row and column from database table

**SYNTAX:**

SELECT<STTRIBUTE NAMES>

FROM<TABLE NAME>

WHERE<CONDITION>;

**ILLUSTRATE THE USE OF SELECT STATEMENT:**

* The basic statement for retrieving the information from database is: SELECT statement.
* It is sometimes called a **mapping** or a **select-from block**.
* The basic form of the select statement is:

SELECT <ATTRIBUTE\_LIST>

FROM <TABLE\_LIST>;

<ATTRIBUTE\_LIST> is a list of attribute names whose values are to be retrieved by the query.

<TABLE\_LIST> is a list of relation names required to process the query.

* The SELECT specifies which column of the table/s has to be retrieved and FROM specifies the name of the table/tables.
* If there are more than one attribute are to be listed then every attribute names are separated by comma (,).
* If there are more than one table are involved in retrieving the data, each names of the table are separated by comma(,).

**Query No :01**

**Retrive first name,birth date from employee table.**

SQL> select fname,bdate from employee;

FNAME BDATE

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

JOHN 09-JAN-65

FRANKLIN 08-DEC-55

ALICIA 19-JAN-68

JENNIFER 20-JUL-74

SUSHANT 21-APR-97

**Query No :02**

**Display the data from department table.**

SQL> select \*from department;

DNAME DNUMBER MGR\_SSN MGR\_START

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

RESEARCH 1 1231 22-JAN-88

ADMINISTRATION 2 1232 01-JAN-95

HEADQUARTERS 3 1233 19-JUL-68

COMPUTERIZATION 4 1234 18-JAN-92

**Query No :03**

**Retrive the details of all the employees from employee table.**

SQL> select \*from employee;

FNAME M LNAME SSN BDATE ADDRESS S SALARY

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

SUPER\_SSN DNO

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

JOHN B SMITH 1231 09-JAN-65 731,FONDREN,HOUSTON M 30000

1231 1

FRANKLIN T WONG 1232 08-DEC-55 638,VOSS,HOUSTON M 40000

1232 2

ALICIA J ZALAYA 1233 19-JAN-68 3321,CASTLE,HOUSTON F 25000

1233 3

JENNIFER S WALLACA 1234 20-JUL-74 291,BERRY,HUMBLE F 43000

1234 4

SUSHANT Y KAMBLE 1235 21-APR-97 ABBIHAL,ATHANI M 38000

1235 1

**Query No :04**

**Display the data from dependant table.**

SQL> select \*from dependent;

ESSN DEPENDENT\_NAME S BDATE RELATIONSHIP

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

1231 SAMARTH M 12-JAN-95 SON

1232 BHARATI F 15-FEB-85 WIFE

1233 CHANDRAKANT M 15-JUN-75 FATHER

1234 SMITHA F 15-SEP-00 DAUGHTER

1235 VIJAYALAXMI F 14-JAN-80 MOTHER

**Query No :05**

**List all the department names from department table.**

SQL> select dname from department;

DNAME

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

RESEARCH

ADMINISTRATION

HEADQUARTERS

COMPUTERIZATION

**Expt. No : -5. CONDITIONAL RETRIEVAL – WHERE CLAUSE:**

* WHERE clause is used for CONDITIONAL SELECTION of columns to retrieve the data from the database.
* GENERAL SYNTAX:

SELECT <ATTRIBUTE\_LIST>

FROM <TABLE\_LIST>

WHERE <CONDITION>;

* <CONDITION> is a conditional (Boolean) expression that identifies the tuples (records) to be retrieved by the query.
* Only those columns specified in the SELECT statement are retrieved from the table/s listed in FROM clause which satisfies the condition specified in WHERE clause.
* The comparison used to specify in the WHERE clause can be a comparison with the values or with the column. (Column specifies comparison with the values present in the column).
* WHERE clause is optional, i.e., if there is no condition specified in query then the columns listed in the SELECT statements are retrieved from the respective tables listed in FROM clause.
* If more than one condition is specified in WHERE clause, then they are placed in between two keywords AND/ OR.

AND: specifies every condition have to be TRUE.

OR: specifies at least one condition to be TRUE.

* The basic logical comparison operators for comparing attribute values with one another.
* There are two types of operators

1. RELATIONAL OPERATORS
2. SPECIAL OPERATORS

**Relational Operators:**

|  |  |
| --- | --- |
| **OPERATOR** | **PURPOSE** |
| = | Test for equality |
| != | Test for inequality |
| ^= | Same as != |
| <> | Same as != |
| < | Less than |
| > | Greater than |
| <= | Less than or equal to |
| >= | Greater than or equal to |

**SPECIAL OPERATORS:**

|  |  |
| --- | --- |
| **OPERATOR** | **PURPOSE** |
| In | Equal to any member in parentheses. |
| Not In | Not equal to any member in parentheses. |
| Between A and B | Greater than or equal to A and less than or equal to B. |
| Not Between A and B | Not greater than or equal to A and Not less than or equal to B. |
| Like %XYZ% | Matching a pattern from a column. |

**Query No :01**

**Retrive the birth date and address of the employee whose name is ANAND C KAMMAR from employee table.**

SQL> SELECT BDATE,ADDRESS FROM EMPGPT1 WHERE FNAME='ANAND' AND MINIT='C' AND LNAME='KAMMAR';

BDATE ADDRESS

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

09-JAN-93 UGAR KHURD

**Query No :02**

**Retrive first name, birth date and salary of the employees working in department number 3.**

SQL> SELECT FNAME,BDATE,SALARY FROM EMPGPT1 WHERE DNO=3;

FNAME BDATE SALARY

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

VINAYAK 03-APR-97 20000

MAHANTESH 15-DEC-95 28000

**Query No :03**

**List the male employees whose salary is greater than 25000.**

**SQL> SELECT \*FROM EMPGPT1 WHERE SEX='M' AND SALARY>25000;**

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

**Query No :04**

**List the female employees whose salary is less than 20000.**

SQL> SELECT \*FROM EMPGPT1 WHERE SEX='F' AND SALARY<20000;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

**Query No :05**

**List all the employees who are born before january 25th 1995.**

SQL> SELECT \*FROM EMPGPT1 WHERE BDATE<'25-JAN-1995';

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

**Query No :06**

**List all the employees who are born before january 25th 1996.**

SQL> SELECT \*FROM EMPGPT1 WHERE BDATE<'25-JAN-1996';

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

Query No :07

**List all the employees except those who are working in the department number 2.**

SQL> SELECT \*FROM EMPGPT1 WHERE DNO<>2;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

6 rows selected.

**Query No :08**

**List all the employees whose salary is in between 20000 and 30000.**

SQL> SELECT \*FROM EMPGPT1 WHERE SALARY BETWEEN 20000 AND 30000;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

**Query No :09**

**Retrive the name and address of all the employees who works for the research department.**

SQL> SELECT FNAME,LNAME,ADDRESS FROM EMPGPT1,DEPGPT1 WHERE DNAME='RESEARCH' AND DNO=DNUMBER;

FNAME LNAME ADDRESS

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

VENKATESH SANNAKKI ATHANI

MAHANTESH MUCHANDI ATHANI

**Query No :10**

**List all the employees whose last name is KAMMAR and Department number is 4.**

SQL> SELECT \*FROM EMPGPT1 WHERE LNAME='KAMMAR' AND DNO=4;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

**Query No :11**

**List all the employees whose name is ANAND or VINAYAK.**

SQL> select fname,lname from empgpt1 where fname='VINAYAK' or fname='ANAND';

FNAME LNAME

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

VINAYAK SOUDHAGAR

ANAND KAMMAR

**Query No :12**

**List all the employees except who works for department number 4 using NOT keyword**.

SQL> SELECT \*FROM EMPGPT1 WHERE NOT DNO=4;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

6 rows selected.

**Query No :13**

**List all the employees except who works for department number 4 using less then(<>) operator**.

SQL> SELECT \*FROM EMPGPT1 WHERE DNO<>4;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

6 rows selected.

**Expt. No : 6. QUERY SORTED: ORDER BY CLAUSE**

* SQL allows user to order the result of the Query by the values of one or more attributes.
* **ORDER BY** is a clause used to order the result of query in ascending or descending.
* By default the order will be ascending.
* For ascending and descending, **ASC** and **DSC** keywords are used.

**Query No :01**

**Sort all the employees according to their first name.**

SQL> select \*from empgpt1 order by fname;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

8 rows selected.

**Query No :02**

**Sort all the employees working in department number 4 by their last name.**

SQL> select \*from empgpt1 where dno=4 order by lname;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

**Query No :03**

**Select all the employees according to their salaries in descending order.**

SQL> select \*from empgpt1 order by salary desc;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

8 rows selected.

**Query No :04**

**Select all the employees according to their salaries in ascending order.**

SQL> select \*from empgpt1 order by salary;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

8 rows selected.

**Query No :05**

**List the employees working for 'research' department according to their salaries.**

SQL> select \*from empgpt1,depgpt1 where dno=dnumber and dname='RESEARCH' order by salary;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

DNAME DNUMBER MGRSSN MGR\_START

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

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

RESEARCH 2 1233 15-FEB-14

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

RESEARCH 2 1233 15-FEB-14

**Query No :06**

**List the employees working for 'technical' department according to their salaries in descending order.**

SQL> select \*from empgpt1,depgpt1 where dno=dnumber and dname='TECHNICAL' order by salary desc;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

DNAME DNUMBER MGRSSN MGR\_START

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

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

TECHNICAL 3 1237 15-MAR-14

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

TECHNICAL 3 1237 15-MAR-14

**Query No :07**

**List the employees working for 'administration' department according to their salaries.**

SQL> select \*from empgpt1,depgpt1 where dno=dnumber and dname='ADMINISTRATION' order by salary;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

DNAME DNUMBER MGRSSN MGR\_START

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

ADMINISTRATION 1 1232 15-JAN-14

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

ADMINISTRATION 1 1232 15-JAN-14

**Query No :08**

**List all employees according to their birth date.**

**SQL> select \*from empgpt1 order by bdate;**

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 15000 1231 1

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

8 rows selected.

**Expt No: - 7. AGGREGATE FUNCTIONS IN SQL:**

* Introducing the concept of aggregate function as a relational operator is because grouping of attributes is required in many database applications.
* Many built-in functions exist, they are,

**Functions Meaning**

* COUNT Determines the number of rows or non NULL column values
* SUM Determines the arithmetic sum of all selected columns
* MAX Determines the largest of all selected values of a columns
* MIN Determines the smallest of all selected values of a columns
* AVG Determines the average of all selected values of a columns

In all the above NULL values are ignored.

**Query No :01**

**Compute the total salaries of all the employees.**

SQL> select sum(salary) from empgpt1;

SUM(SALARY)

-----------

198000

**Query No :02**

**Compute the total salaries of all the who were working in department number 1.**

SQL> select sum(salary) from empgpt1 where dno=1;

SUM(SALARY)

-----------

45000

**Query No :03**

**Compute the average salary of all the employees.**

SQL> select avg(salary) from empgpt1;

AVG(SALARY)

-----------

24750

**Query No :04**

**select the maximum and minimum salaries of the employees who working in department number 1**.

SQL> select max(salary),min(salary) from empgpt1 where dno=1;

MAX(SALARY) MIN(SALARY)

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

30000 15000

**Query No :05**

**Count the number of employees in employee table.**

SQL> select count(\*) from empgpt1;

COUNT(\*)

---------

8

**Query No :06**

**Count the number of employees working in department number 1.**

SQL> select count(\*) from empgpt1 where dno=1;

COUNT(\*)

---------

2

**Query No :07**

**Count the number of departments in department table.**

SQL> select count(\*) from depgpt1;

COUNT(\*)

---------

4

**Expt No: - 8. GROUPING THE RESULT OF THE QUERY- GROUP BY AND HAVING CLAUSE:**

* To subgroups of tuples in a relation we apply aggregate functions, where subgroups are based on some attribute values.
* For example, if we want to find the average salary of al the employees in each department separately. In this case we need to partition the relation into groups, specifying the average salary of employees in each department. This process is called as grouping of attribute.
* Each grouped attribute consists of tuples that have the same value of some attribute.
* It is achieved in SQL by using GROUP BY clause.
* The grouping attribute specified in the GROUP BY clause should also appear in the SELECT statement.
* Grouping can be applied to that particular attribute to which we can apply aggregate function.
* Sometimes we need to group the attributes values that satisfies some condition, where we apply HAVING clause.
* HAVING clause is applied to specify condition for those values of attributes specified in GROUP BY clause. Where the grouping of those values of attributes are done which satisfy the condition specified in HAVING clause.

**Query No :01**

**Compute Average salary for each department.**

SQL> select dno,avg(salary) from empgpt1 group by dno;

DNO AVG(SALARY)

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

1 22500

2 26500

3 24000

4 26000

**Query No :02**

**Find the maximum and minimum salary of each department.**

SQL> select dno, max(salary),min(salary) from empgpt1 group by dno;

DNO MAX(SALARY) MIN(SALARY)

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

1 30000 15000

2 35000 18000

3 28000 20000

4 27000 25000

**Query No :03**

**List the maximum salary of those departments in which more than one employee works.**

SQL> select dno,max(salary) from empgpt1 group by dno having count(\*)>1;

DNO MAX(SALARY)

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

1 30000

2 35000

3 28000

4 27000

**Query No :04**

**Find the department whose minimum salary is less than 22000.**

SQL> select dno,min(salary) from empgpt1 group by dno having min(salary)<22000;

DNO MIN(SALARY)

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

1 15000

2 18000

3 20000

**Query No :05**

**Find the department whose maximum salary is greater than 20000.**

SQL> select dno,max(salary) from empgpt1 group by dno having max(salary)>20000;

DNO MAX(SALARY)

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

1 30000

2 35000

3 28000

4 27000

**Query No :06**

**Find the department whose maximum salary is greater than 29000.**

SQL> select dno,max(salary) from empgpt1 group by dno having max(salary)>29000;

DNO MAX(SALARY)

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

1 30000

2 35000

**Expt. No 09. Perform UPDATE, ALTER, DELETE, DROP operations on tables**

1. **UPDATE:**

* UPDATE command is used to modify attribute values of one or more selected tuples.
* **General Syntax:**

UPDATE <table\_name>

SET Column name=Expression

WHERE Column name=Expression

* **Update** followed by the table name, is mandatory. Specifying to which table the update has to be done.
* **Set** followed by one or more columns you want to change. This is mandatory.
* **Where** clause is optional, selects the tuple to be modified from a single relation.
* Although the update of primary key should make a necessary update to all the keys which it references. This can be achieved by specifying **on update cascade** in the process of creation of table.

Example:

QUERY 1 : Modify the salary of employee 'VINAYAK' to 22000.

**SQL> update empgpt1 set salary=22000 where fname='VINAYAK';**

1 row updated.

SQL> select \*from empgpt1;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 16000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 22000 1233 3

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

8 rows selected.

Query 2- Modify the address of the employee ssn=1232 to ' VIJAYPUR'

SQL> update empgpt1 set address='VIJAYPUR' where

ssn=1232;

1 row updated.

SQL> select \*from empgpt1;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 16000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 VIJAYPUR M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 22000 1233 3

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

8 rows selected.

Query 3 - Modify DEPARTMENT LOCATION to mangalore from department 1

SQL> update dept\_locationgpt1 set dlocation='mangalore' where dnumber=1;

1 row updated.

SQL> select \* from dept\_locationgpt1;

DNUMBER DLOCATION

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

1 mangalore

2 Mumbai

3 Athani

4 Belgaum

2 Pune

3 Bijapur

6 rows selected.

Query 4 – Update the salary of employee ssn 1231 to 16000.

SQL> UPDATE EMPGPT1 SET SALARY=16000 WHERE SSN=1231;

1 row updated.

SQL> SELECT \*FROM EMPGPT1;

FNAME M LNAME SSN BDATE ADDRESS S SALARY SUPERSSN DNO

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

SHRADDHA M NAIK 1231 18-OCT-96 KRISHNA KITTUR F 16000 1231 1

VENKATESH D SANNAKKI 1232 02-JUL-96 ATHANI M 18000 1232 2

VINAYAK A SOUDHAGAR 1233 03-APR-97 ATHANI M 20000 1233 3

ANAND C KAMMAR 1234 09-JAN-93 UGAR KHURD M 25000 1234 4

GEETA G MATHAD 1235 16-APR-95 SATTI F 30000 1235 1

MAHANTESH S MUCHANDI 1236 10-JUL-95 ATHANI M 35000 1236 2

MAHANTESH M PATRI 1237 15-DEC-95 ATHANI M 28000 1237 3

MANJUNATH A MORDI 1238 12-NOV-95 KUDACHI M 27000 1238 4

8 rows selected.

1. **ALTER:**

* ALTER command is used to change the definition of base table.
* The possible actions of ALTER command is

i. Adding or Dropping a column.

ii. Changing the column definition.

iii. Renaming the base table or attribute

**i. Adding a column:**

Adding can be achieved by using ALTER command, with **add** keyword.

General Syntax:

**ALTER TABLE <table\_name> add(<attribute name> <data type> <constraint>);**

***Note:*** it is not possible to set NOT NULL constraint to the new attribute we are trying to add using ALTER command.

**ii. Changing the column definition:**

* The definition of column can be changed by using keyword **modify**.
* The changing includes adding constraint, removing constraint, changing the data type of the attribute.
* General syntax:
* **ALTER TABLE <table\_name> MODIFY(<attribute\_name> <new\_data\_type> <new\_constraint>);**

Query : Modify the datatype of the column plocation to varchar2(25)

SQL> ALTER TABLE PROJECTGPT1 MODIFY(PLOCATION VARCHAR2(25));

Table altered.

SQL> DESC PROJECTGPT1;

Name Null? Type

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

PNUMBER NOT NULL NUMBER(5)

PNAME NOT NULL VARCHAR2(15)

PLOCATION VARCHAR2(25)

DNUM NOT NULL NUMBER(5)

**iii. Renaming a base table:**

* The **rename** keyword along with ALTER command is used to rename the base table.
* General syntax:

**ALTER TABLE <table\_name> RENAME TO <new\_name\_for\_table>);**

Query : Rename the the projectgpt1 table to Project01;

SQL> ALTER TABLE PROJECTGPT1 RENAME TO PROJECT01;

Table altered.

SQL> DESC PROJECTGPT1;

ERROR:

ORA-04043: object PROJECTGPT1 does not exist

**3. DELETE:**

* The DELETE command removes tuples from a relation.
* General syntax:

DELETE FROM <table\_name>

WHERE <condition>;

* Where clause select the tuples to be deleted. It is optional.
* At a time the records are deleted from only one table.

DELETE without any condition (Unspecified WHERE clause) will cause to delete all the records from the table, preserving the definition of base table

1. **DROP:**

* DROP is a command used to delete the table created.
* Along with the records of the table it deletes even the definition of the base table.
* Dropping becomes difficult if reference constraints are not properly specified for the column which acts as a referential key in the base table.
* General syntax:

**DROP TABLE <table\_name>;**

Query: Drop the table student.

TNAME TABTYPE CLUSTERID

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

SHASHI TABLE

ST TABLE

STUDENT TABLE

STUDENT2 TABLE

STUDENT3 TABLE

WORKS\_ONGPT1 TABLE

31 rows selected.

SQL> drop table student;

Table dropped.

SQL> desc student;

ERROR:

ORA-04043: object student does not exist

SQL>

**Expt. No. 10 QUERY MULTIPLE TABLES USING JOIN OPERATION:**

* JOIN operation is used to join two tables, i.e., combines the columns of two different tables.
* The relationship between two tables achieved by primary key and foreign keys are used to join the tables.
* The list of tables joined are specified in the FROM clause of a query.
* General syntax:

**<table1\_name> JOIN <table2\_name> ON <condition>**

is specified in FROM clause.

* From the joined table, based on the condition specified in WHERE clause, the list of attributes mentioned in the SELECT statement is retrieved.

Query 1: List the employees and the department names in which they are working

SQL> select fname,lname,salary,dname from empgpt1,depgpt1

where empgpt1.dno=depgpt1.dnumber;

FNAME LNAME SALARY DNAME

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

SHRADDHA NAIK 16000 ADMINISTRATION

VENKATESH SANNAKKI 15000 RESEARCH

VINAYAK SOUDHAGAR 22000 TECHNICAL

ANAND KAMMAR 25000 HEADQUARTER

GEETA MATHAD 30000 ADMINISTRATION

MAHANTESH MUCHANDI 35000 RESEARCH

MAHANTESH PATRI 28000 TECHNICAL

MANJUNATH MORDI 27000 HEADQUARTER

8 rows selected.

Query 2: List the name and Salary of employees working in 'RESEARCH' department

SQL> select fname,lname,salary,dname from empgpt1,depgpt1

where empgpt1.dno=depgpt1.dnumber and dname='RESEARCH';

FNAME LNAME SALARY DNAME

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

VENKATESH SANNAKKI 15000 RESEARCH

MAHANTESH MUCHANDI 35000 RESEARCH

Query 3: List the employees who are working on project 'Production X'

SQL> select fname,lname,salary from empgpt1,works\_ongpt1,project01 where

empgpt1.ssn=works\_ongpt1.essn and works\_ongpt1.pno=project01.pnumber

and pname='Product X';

FNAME LNAME SALARY

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

SHRADDHA NAIK 16000

Query 4:List the employees who are assigned to a department as well as employees who are not assigned to a department

SQL> select e.ssn,e.fname,e.dno,d.dname from empgpt1 e,depgpt1 d

where e.dno(+)=d.dnumber;

SSN FNAME DNO DNAME

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

1231 SHRADDHA 1 ADMINISTRATION

1235 GEETA 1 ADMINISTRATION

1232 VENKATESH 2 RESEARCH

1236 MAHANTESH 2 RESEARCH

1233 VINAYAK 3 TECHNICAL

1237 MAHANTESH 3 TECHNICAL

1234 ANAND 4 HEADQUARTER

1238 MANJUNATH 4 HEADQUARTER

8 rows selected.

SQL>

Query 5: list the employees with their managers

SQL> select e.ssn,e.fname,m.ssn,m.fname from empgpt1 e,empgpt1 m

where e.superssn=m.ssn;

SSN FNAME SSN FNAME

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

1231 SHRADDHA 1238 MANJUNATH

1232 VENKATESH 1237 MAHANTESH

1233 VINAYAK 1236 MAHANTESH

1234 ANAND 1235 GEETA

1235 GEETA 1234 ANAND

1236 MAHANTESH 1233 VINAYAK

1237 MAHANTESH 1232 VENKATESH

1238 MANJUNATH 1231 SHRADDHA

8 rows selected.

Query 6: List the Manager name for the employee 'GEETA'

SQL> select e.ssn,e.fname,m.ssn,m.fname from empgpt1 e,empgpt1 m

where e.superssn=m.ssn and e.fname='GEETA';

SSN FNAME SSN FNAME

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

1235 GEETA 1234 ANAND

Query 7: List the dependents with their relationship of employee 1232

SQL> select dependent\_name,relationship,essn from dependentgpt1,empgpt1

where empgpt1.ssn=dependentgpt1.essn and empgpt1.ssn=1232;

DEPENDENT\_NAME RELATIONSHIP ESSN

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

Guru Son 1232

Soumya Mother 1232

Vijay Son 1232

**Expt. No. 11 USE OF DIFFERENT GROUP FUNCTIONS:**

* Group functions are the in-built functions used to have a mathematical, character, date functions in SQL.
* This also includes Aggregate functions like AVG, SUM, MAX etcetera.
* Mathematical functions includes power, sin, cos, tan, log, sqrt, exp etcetera.
* Character functions include upper, lower etcetera.
* Date functions include sysdate etcetera
* General syntax:

SELECT <function\_list> FROM dual;

The dual is a in-built file name which includes description of all the group functions.

**1. MATHEMATICAL FUNCTIONS:**

**Functions Meaning**

1. Power(m,n) Finds power value
2. Sin(m) Finds Sin value of a given angle
3. Cos(m) Finds Cos value of a given angle
4. Tan(m) Finds Tan value of a given angle
5. Log(m,n) Finds Logarithmic value
6. Exp(m) Finds Exponential value
7. Sqrt(x) Finds Square root value
8. Round(x) Finds Rounds up to 1 value

**SQL> SELECT POWER(3,4) FROM DUAL;**

POWER(3,4)

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

81

**SQL> SELECT SIN(90) FROM DUAL;**

SIN(90)

----------

.893996664

**SQL> SELECT COS(0) FROM DUAL;**

COS(0)

----------

1

**SQL> SELECT TAN(0) FROM DUAL;**

TAN(0)

----------

0

**SQL> SELECT LOG(10,100) FROM DUAL;**

LOG(10,100)

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

2

**SQL> SELECT LOG(20,2) FROM DUAL;**

LOG(20,2)

----------

.231378213

**SQL> SELECT EXP(2) FROM DUAL;**

EXP(2)

----------

7.3890561

**SQL> SELECT SQRT(81) FROM DUAL;**

SQRT(81)

----------

9

**Round(n,m)** it returns result rounded to m places to the right of the decimal point

**SQL> SELECT ROUND(3456.678) FROM DUAL;**

ROUND(3456.678)

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

3457

**Ceil(n)** it returns nearest whole integer greater than or equal to number

**SQL> SELECT CEIL(34.56) FROM DUAL;**

CEIL(34.56)

-----------

35

**Mod(m,n)** it returns remainder of m divided by n. if n=0 then m is returned

**SQL> SELECT MOD(105,10) FROM DUAL;**

MOD(105,10)

-----------

5

**Sign(n)** if n=0 it returns 0, if n>0, returns 1,if n<0 returns -1

**SQL> SELECT SIGN(35) FROM DUAL;**

SIGN(35)

----------

1

**Q33. SQL> SELECT FNAME, SALARY, SALARY/15, ROUND(SALARY/15,2), ROUND(SALARY/15,0)**

**FROM EMPLOYEE;**

FNAME SALARY SALARY/15 ROUND(SALARY/15,2) ROUND(SALARY/15,0)

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

JOHN 30000 2000 2000 2000

FRANKLIN 40000 2666.66667 2666.67 2667

ALICIA 25000 1666.66667 1666.67 1667

JENNIFER 43000 2866.66667 2866.67 2867

RAMESH 38000 2533.33333 2533.33 2533

JOYCE 25000 1666.66667 1666.67 1667

AHMAD 25000 1666.66667 1666.67 1667

JAMES 55000 3666.66667 3666.67 3667

8 rows selected.

**2. CHARACTER FUNCTIONS:**

1. **Lower(string):** Converts the string to lower case and result of the query in lower case, if the records in the base table are in upper case.

Ex:

**SQL> SELECT LOWER('NAVATHE') FROM DUAL;**

LOWER('NAVATHE’)

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

navathe

**SQL> SELECT LOWER(DLOCATION) FROM DEPT\_LOCATIONS;**

LOWER(DLOCATION

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

houston

stafford

bellaire

houston

1. **Upper(string):** It converts the string to upper case and display the result of the query in upper case, if the records in base table are in lower case.

Ex:

**SQL> SELECT UPPER('navathe') FROM DUAL;**

UPPER('navathe’)

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

NAVATHE

**SQL> SELECT UPPER(DLOCATION) FROM DEPT\_LOCATIONS;**

UPPER(DLOCATION

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

HOUSTON

STAFFORD

BELLAIRE

HOUSTON

SUGARLAND

1. **Initcap(char)** it capitalizes the first character of each word in the string.

SQL> SELECT INITCAP('mr. Navathe') FROM DUAL;

INITCAP('MR.RAJE

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

Mr.Navathe

1. **Substr(String,m,n):** It returns a substring, starting with m, n characters long.

SQL> SELECT SUBSTR('NAVATHE', 1,4) FROM DUAL;

SUBS

-------

NAVA

1. **Length(string):** It returns the length of a string. Usually applicable only with varchar2 and not with char type columns.

SQL> SELECT LENGTH('NAVATHE') FROM DUAL;

LENGTH('NAVATHE')

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

7

1. **Instr(string,char)** It returns the position of first occurrence of **‘char’** in **‘string’**

SQL> SELECT INSTR('NAVATHE','T') FROM DUAL;

INSTR('NAVATHE','T')

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

5

**3. DATE FUNCTION:**

**Functions Meaning**

Sysdate Returns the current date.

Last\_day(d) Returns the date of the last day of the month specified

Next\_day(date,day) finds the date of the ‘day’ after the ‘date’

**SQL> SELECT SYSDATE FROM DUAL;**

SYSDATE

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

20-MAR-16

**SQL> SELECT LAST\_DAY(SYSDATE) FROM DUAL;**

LAST\_DAY(

---------

31-MAR-16

**SQL> SELECT NEXT\_DAY(SYSDATE,'THURSDAY') FROM DUAL;**

NEXT\_DAY(

---------

21-MAR-16

**Expt. No. 12 VIEWS**

* A view is a special database object that can be created to restrict access to certain columns or rows of data within a table.
* A view is a SQL \* PLUS query i.e., permanently stored in the database and assigned a name.
* A view is a **virtual table**, i.e., it is derived from an another table called a **base table**.
* So, view are the masks placed upon tables. Views are logical tables of data extracted from existing tables. Hence we can display data to the users according to their need.
* When several people are querying the database, different people want to look at data differently i.e., each group of people would like to see different fields of same table.
* By generating the views, Oracle makes the querying of table much easier.
* The DBA treats a view like a table. Hence, you can query a view in the same way like a table.
* The query field on a view runs faster than a query for a table because, it is the subset of the total number of columns in the table.

**Views may be created for following reasons**

* The DBA stores the view as a definition only. Hence there is no duplication of data.
* Simplifies queries
* Can be queried as a base table itself
* Provides data security
* Avoids data redundancy.

**Syntax:**

CREATE VIEW view\_name

AS SELECT column\_name, column\_name,….

FROM table\_name

WHERE column\_name=expression list;

**Updatable Views:**

* Users can perform Insert, Update and Delete operations on the view.
* The views on which data manipulation can be done are called **updatable views.**
* When you update a view, the modification of the data will be passed to its respective base table

**Deleting view:**

A view can be dropped by using the DROP VIEW command

**Syntax:**

DROP VIEW view\_name;

Query : Create a view which contains employee ssn, fname, lname, bdate

and salary of all employees.

SQL> create view emp\_view as select ssn,fname,lname,bdate,salary from empgpt1;

View created.

Query: Create a view of all employees working for the department 1

SQL> create view dep\_view as select dno,ssn,fname,lname,salary

from empgpt1 where dno=1;

View created.

SQL> select \*from dep\_view;

DNO SSN FNAME LNAME SALARY

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

1 1231 SHRADDHA NAIK 16000

1 1235 GEETA MATHAD 30000

SQL> select \*from emp\_view;

SSN FNAME LNAME BDATE SALARY

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

1231 SHRADDHA NAIK 18-OCT-96 16000

1232 VENKATESH SANNAKKI 02-JUL-96 15000

1233 VINAYAK SOUDHAGAR 03-APR-97 22000

1234 ANAND KAMMAR 09-JAN-93 25000

1235 GEETA MATHAD 16-APR-95 30000

1236 MAHANTESH MUCHANDI 10-JUL-95 35000

1237 MAHANTESH PATRI 15-DEC-95 28000

1238 MANJUNATH MORDI 12-NOV-95 27000

8 rows selected.

Query: Drop the view emp\_view

SQL> drop view emp\_view;

View dropped.

SQL> select \*from emp\_view;

select \*from emp\_view

\*

ERROR at line 1:

ORA-00942: table or view does not exist

Query: Drop the view dep\_view

SQL> drop view dep\_view;

View dropped.

SQL> select \*from dep\_view;

select \*from dep\_view

\*

ERROR at line 1:

ORA-00942: table or view does not exist

**13. USE COMMIT AND ROLL BACK COMMAND:**

**COMMIT:**

* A transaction is a sequence of operations in the database.
* The operations like INSERT, DELETE and UPDATE operations modify the state of the database every time.
* To make the changes permanent to the database when some manipulation operation is executed, a COMMIT command is used.
* The COMMIT command writes the contents of log file in the physical database.
* COMMIT command can be issued in two ways:

🡪 Explicit command and

🡪 Implicit command

* An implicit commit is issued by setting a system environment parameter AUTOCOMMIT ON.
* To give commit command explicitly, AUTOCOMMIT must be OFF.

**ROLL BACK:**

* ROLLBACK command is used for canceling the changes made to the database using INSERT, DELETE and UPDATE operations.
* The implicit ROLLBACK occurs when an error occurs during transaction.
* The explicit ROLLBACK can be given if we don’t want changes made to the database permanent (i.e., to cancel any changes made to the database).
* If we simply rollback, all changes made to the database after previous commit will be cancelled.
* The changes made can be cancelled only if COMMIT is not executed.

Ex:

SQL> select \*from dept\_locationgpt1;

DNUMBER DLOCATION

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

1 mangalorre

2 Mumbai

3 Athani

4 Belgaum

2 Pune

3 Bijapur

6 rows selected.

**SQL> COMMIT;**

Commit complete.

**SQL> DELETE**  \*from dept\_locationgpt1;

6 rows deleted.

**SQL>** select \*from dept\_locationgpt1;

no rows selected

SQL> ROLL BACK;

Rollback complete.

SQL> select \*from dept\_locationgpt1;

DNUMBER DLOCATION

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

1 mangalorre

2 Mumbai

3 Athani

4 Belgaum

2 Pune

3 Bijapur

6 rows selected.