

## LAB CYCLE 1

### Experiment No: 1

#### Familiarization of DDL Commands

Data Definition Language (DDL) - These SQL commands are used for creating, modifying, and dropping the structure of database objects. The commands are CREATE, ALTER, DROP, RENAME, and TRUNCATE.

Some common DDL commands:

- Create Database  
Syntax: create database database\_name;
- Drop Database  
Syntax: drop database [if exists] database\_name;
- Create Table  
Syntax: create table *table\_name* ( *column1 datatype*,....);
- Drop Table  
Syntax: drop table table\_name;
- Alter Table
  - Add Column  
Syntax: alter table *table\_name* add *column\_name datatype*;
  - Drop Column  
Syntax: alter table *table\_name* drop column *column\_name*;
  - Rename Column  
Syntax: alter table *table\_name* rename column *old\_name* to *new\_name*;
  - Modify Column properties  
Syntax: alter table *table\_name* modify *column\_name datatype*;

A. Consider the database for a college. Write SQL commands to implement the following:

1. Create a database

**SQL :** create database college;

**Output:** Database created

2. Select the current database

**SQL :** use college;

**Output:**

Database changed

3. Create the following tables:

a) Student (roll\_no integer, name varchar, dob date, address text, phone\_no varchar, blood\_grp varchar)

**SQL:** create table Student(roll\_no int,name varchar(20),dob date,address text(50),phone\_no varchar(10),blood\_grp varchar(5));

**Output:**

Query OK, 0 rows affected (0.38 sec)

```
mysql> describe Student;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| roll_no    | int           | YES  |     | NULL    |       |
| name       | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| phone_no   | varchar(10)   | YES  |     | NULL    |       |
| blood_grp  | varchar(5)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

b) Course (Course\_id integer, Course\_name varchar, course\_duration integer)

**SQL:** create table course(course\_id int,course\_name varchar(20),course\_duration int);

**Output:**

```
Query OK, 0 rows affected (0.38 sec)
```

```
mysql> describe course;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| course_id      | int           | YES  |     | NULL    |       |
| course_name    | varchar(20)   | YES  |     | NULL    |       |
| course_duration | int           | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

4. List all tables in the current database.

**SQL:** show tables;

**Output:**

```
+-----+
| Tables_in_22mca54 |
+-----+
| Student           |
| course            |
+-----+
2 rows in set (0.00 sec)
```

5. Display the structure of the Student table.

**SQL:** describe Student;

Output:

```
mysql> describe Student;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| roll_no    | int           | YES  |     | NULL    |       |
| name       | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| phone_no   | varchar(10)   | YES  |     | NULL    |       |
| blood_grp  | varchar(5)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

6. Drop the column blood\_grp from Student table.

**SQL:** alter table Student drop column blood\_grp;

Output:

```
Query OK, 0 rows affected (0.92 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| roll_no    | int           | YES  |     | NULL    |       |
| name       | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| phone_no   | varchar(10)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

7. Add a new column Adar\_no with domain number to the table Student.

**SQL:** alter table Student add adar\_no int;

Output:

```
Query OK, 0 rows affected (0.67 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| roll_no    | int           | YES  |     | NULL    |       |
| name       | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| phone_no   | varchar(10)   | YES  |     | NULL    |       |
| adar_no    | int           | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)
```

8. Change the datatype of phone\_no from varchar to int

**SQL:** alter table Student modify phone\_no int;

**Output:**

```
Query OK, 0 rows affected (2.36 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe Student;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| roll_no | int           | YES  |     | NULL    |       |
| name    | varchar(20)   | YES  |     | NULL    |       |
| dob     | date          | YES  |     | NULL    |       |
| address | tinytext      | YES  |     | NULL    |       |
| phone_no | int           | YES  |     | NULL    |       |
| adar_no | int           | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

9. Drop the tables.

**SQL:** drop table Student;  
drop table course;

**Output:**

```
mysql> drop table Student;
Query OK, 0 rows affected (0.42 sec)

mysql> drop table course;
Query OK, 0 rows affected (0.39 sec)
```

10. Delete the database.

**SQL:** drop database college;

**Output:**

```
Query OK, 0 rows affected (0.38 sec)
```

B. Consider the database for an organization. Write SQL commands to implement the following:

1. Create a database

**SQL:** create database organization

**Output:**

Database created

2. Select the current database

**SQL:** use organization

**Output:**

```
Database changed
```

3. Create the following tables:

- a) Employee (emp\_no varchar, emp\_name varchar, dob date, address text, mobile\_no integer, dept\_no varchar, salary integer)

**SQL:** create table Employee(emp\_no varchar(20),emp\_name varchar(20),dob date,address text(50),mobile\_no int,dept\_no varchar(20),salary int);

**Output:**

```
Query OK, 0 rows affected (0.96 sec)

mysql> desc Employee;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| emp_no     | varchar(20)   | YES  |     | NULL    |       |
| emp_name   | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| mobile_no  | int           | YES  |     | NULL    |       |
| dept_no    | varchar(20)   | YES  |     | NULL    |       |
| salary     | int           | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.01 sec)
```

- b) Department (dept\_no varchar, dept\_name varchar, location varchar)

**SQL:** create table department(dept\_no varchar(20),dept\_name varchar(20),location varchar(20));

**Output:**

```
Query OK, 0 rows affected (0.52 sec)

mysql> desc department;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| dept_no    | varchar(20)   | YES  |     | NULL    |       |
| dept_name  | varchar(20)   | YES  |     | NULL    |       |
| location   | varchar(20)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)
```

4. List all tables in the current database.

**SQL:** show tables;

**Output:**

```
+-----+
| Tables_in_22mca54 |
+-----+
| Employee           |
| department         |
+-----+
2 rows in set (0.01 sec)
```

5. Display the structure of the Employee and Department table.

**SQL:** desc Employee;  
desc department;

**Output:**

Field	Type	Null	Key	Default	Extra
emp_no	varchar(20)	YES		NULL	
emp_name	varchar(20)	YES		NULL	
dob	date	YES		NULL	
address	tinytext	YES		NULL	
mobile_no	int	YES		NULL	
dept_no	varchar(20)	YES		NULL	
salary	int	YES		NULL	

7 rows in set (0.01 sec)

Field	Type	Null	Key	Default	Extra
dept_no	varchar(20)	YES		NULL	
dept_name	varchar(20)	YES		NULL	
location	varchar(20)	YES		NULL	

3 rows in set (0.01 sec)

6. Add a new column 'Designation' to the table Employee.

**SQL:** alter table Employee add Designation varchar(20);

**Output:**

Query OK, 0 rows affected (0.48 sec)  
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc Employee;

Field	Type	Null	Key	Default	Extra
emp_no	varchar(20)	YES		NULL	
emp_name	varchar(20)	YES		NULL	
dob	date	YES		NULL	
address	tinytext	YES		NULL	
mobile_no	int	YES		NULL	
dept_no	varchar(20)	YES		NULL	
salary	int	YES		NULL	
Designation	varchar(20)	YES		NULL	

8 rows in set (0.01 sec)

7. Drop the column 'location' from Department table.

**SQL:** alter table department drop column location;

**Output:**

```
Query OK, 0 rows affected (0.51 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc department;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| dept_no    | varchar(20)   | YES  |     | NULL    |       |
| dept_name  | varchar(20)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.01 sec)
```

## Experiment No: 2

### Familiarization of SQL Constraints.

Constraints are used to specify rules for data in a table.

Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

➤ With Create Table

Syntax: create table table\_name ( column1 datatype constraint,...);

➤ With Alter Table

Syntax: alter table persons add constraint <name> column name

Different constraints

- NOT NULL - Ensures that a column cannot have a NULL value  
Syntax: create table table\_name ( column\_name datatype NOT NULL,...);
- UNIQUE - Ensures that all values in a column are different  
Syntax: create table table\_name ( column\_name datatype UNIQUE,...);
- PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table  
Syntax: create table table\_name ( column\_name datatype PRIMARY KEY,...);
- FOREIGN KEY - a foreign key is a field or a column that is used to establish a link between two tables.  
Syntax: create table table\_name (column\_list,..., foreign key (column\_list) references parent\_table(column\_list));
- CHECK - Ensures that all values in a column satisfies a specific condition  
Syntax: create table table\_name ( column\_name datatype check (expression),...);
- DEFAULT - Sets a default value for a column when no value is specified.  
Syntax: create table table\_name ( column\_name datatype default value,...);

1. Create new table Persons with attributes PersonID (integer, PRIMARY KEY), Name (varchar, NOT NULL), Aadhar (Number, NOT NULL, UNIQUE), Age (integer, CHECK>18).

**SQL:** create table Persons(PersonID int PRIMARY KEY,Name varchar(20) NOT NULL,Aadhar int NOT NULL UNIQUE,Age int CHECK (Age>18));

**Output:**

```
mysql> create table Persons(PersonID int PRIMARY KEY,Name varchar(20) NOT NULL,Aadhar int NOT NULL UNIQUE,Age int CHECK (Age>18));
Query OK, 0 rows affected (0.72 sec)
```

2. Create table Orders with attributes OrderID (PRIMARY KEY), OrderNumber(NOT NULL) and PersonID( set FOREIGN KEY on attribute PersonID referencing the column PersonId of Person table)

**SQL:** create table Orders(OrderID int PRIMARY KEY,Order\_Number int NOT NULL,PersonID int,FOREIGN KEY(PersonID) References Persons(PersonID));

**Output:**

```
mysql> create table Orders(OrderID int PRIMARY KEY,Order_Number int NOT NULL,PersonID int,FOREIGN KEY(PersonID) References Persons(PersonID));
Query OK, 0 rows affected (1.43 sec)
```

3. Display the structure of Persons tables.

**SQL:** desc Persons;



Output:

```
mysql> desc Persons;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| PersonID   | int           | NO   | PRI | NULL    |       |
| Name       | varchar(20)   | NO   |     | NULL    |       |
| Aadhar     | int           | NO   | UNI | NULL    |       |
| Age        | int           | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

4. Display the structure of Orders tables.

**SQL:** desc Orders;

Output:

```
mysql> desc Orders;
+-----+-----+-----+-----+-----+-----+
| Field          | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| OrderID       | int  | NO   | PRI | NULL    |       |
| Order_Number  | int  | NO   |     | NULL    |       |
| PersonID      | int  | YES  | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

5. Add emp\_no as the primary key of the table Employee.

**SQL:** alter table Employee ADD CONSTRAINT EMP\_FK PRIMARY KEY(emp\_no);

Output:

```
mysql> alter table Employee ADD CONSTRAINT EMP_FK PRIMARY KEY(emp_no);
Query OK, 0 rows affected (2.09 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc Employee;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| emp_no     | varchar(20)   | NO   | PRI | NULL    |       |
| emp_name   | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| mobile_no  | int           | YES  |     | NULL    |       |
| dept_no    | varchar(20)   | YES  |     | NULL    |       |
| salary     | int           | YES  |     | NULL    |       |
| Designation | varchar(20)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

6. Add dept\_no as the primary key of the table Department.

**SQL:** alter table department ADD CONSTRAINT DEPT\_FK PRIMARY KEY(dept\_no);

**Output:**

```
mysql> alter table department ADD CONSTRAINT DEPT_FK PRIMARY KEY(dept_no);
Query OK, 0 rows affected (1.99 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc department;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| dept_no    | varchar(20)   | NO   | PRI | NULL    |       |
| dept_name  | varchar(20)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.01 sec)
```

7. Add dept\_no in Employee table as the foreign key reference to the table Department with on delete cascade.

**SQL:** alter table Employee ADD CONSTRAINT EMP\_FK FOREIGN KEY(dept\_no) References department(dept\_no) ON DELETE CASCADE;

**Output:**

```
mysql> alter table Employee ADD CONSTRAINT EMP_FK FOREIGN KEY(dept_no) References department(dept_no) ON DELETE CASCADE;
Query OK, 0 rows affected (1.68 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc Employee;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| emp_no     | varchar(20)   | NO   | PRI | NULL    |       |
| emp_name   | varchar(20)   | YES  |     | NULL    |       |
| dob        | date          | YES  |     | NULL    |       |
| address    | tinytext      | YES  |     | NULL    |       |
| mobile_no  | int           | YES  |     | NULL    |       |
| dept_no    | varchar(20)   | YES  | MUL | NULL    |       |
| salary     | int           | YES  |     | NULL    |       |
| Designation | varchar(20)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

8. Drop the primary key of the table Orders.

**SQL:** alter table Orders drop PRIMARY KEY;

**Output:**

```
mysql> alter table Orders drop PRIMARY KEY;
Query OK, 0 rows affected (3.60 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc Orders;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| OrderID    | int           | NO   |     | NULL    |       |
| Order_Number | int           | NO   |     | NULL    |       |
| PersonID   | int           | YES  | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

### Experiment No: 3

#### Familiarization of DML Commands.

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database. Basically, DCL statements are grouped with DML statements.

- Insert command  
Syntax: insert into tablename (columnname1, columnname2, ...) values (column1\_value, column2\_value, ... );
- Update command  
Syntax: UPDATE tablename SET column1 = new\_value1, column2 = new\_value2,... WHERE search condition;
- Select command  
Syntax: select [distinct] <select-list> from <from-list> [where <qualification>]
- Delete command  
Syntax: delete from table\_name where some\_condition;

1. Add at least 10 rows into the table Employee and Department.

**SQL:** insert into department values('D02','CSE');

insert into department values('D01','MCA');

insert into department values('D03','Mech');

insert into department values('D04','ECE');

insert into department values('D05','EC');

insert into department values('D06','Civil');

insert into department values('D07','Arch');

insert into department values('D08','BCA');

insert into department values('D09','Bsc');

insert into department values('D10','Maths');

insert into Employee values('emp1','John','1985-01-01','ABC Avenue',9874563210,'D02',150000,'Manager');

insert into Employee values('emp2','Aby','1988-01-01','SH Avenue',9874563310,'D01',35000,'Computer');

insert into Employee values('emp3','Abraham','1983-01-01','SH Avenue',9874563310,'D03',180000,'Computer Assistant');

insert into Employee values('emp4','Aiswarya','1983-01-01','SH Avenue',9874563310,'D04',45000,'Computer Assistant');

insert into Employee values('emp5','Aisu','1983-01-01','SH Avenue',9874563310,'D05',45000,'Manager');

insert into Employee values('emp6','Ram','1983-01-01','SH Avenue',9874563310,'D06',7000,'Clerk');

insert into Employee values('emp7','Raj','1983-01-01','SH Avenue',9874563310,'D07',8000,'Clerk');

insert into Employee values('emp8','Rohan','1983-01-01','NY Avenue',9874563310,'D08',18000,'TL');

insert into Employee values('emp9','Rohith','1983-01-01','NY Avenue',9874563310,'D09',70000,'TL');

```
insert into Employee values('emp10','Renju','1983-01-01','NY
AVenue',9874563310,'D10',70000,'TL');\
insert into Employee values('emp11','Rohith','1983-01-01','NY
AVenue',9874563310,'D09',4000,'TL');
```

**Output:**

```
Query OK, 1 row affected (0.11 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

2. Display all the records from the above tables.

**SQL:** select \* from Employee;

Select \* from department;

**Output:**

```
mysql> select * from Employee;
+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC AVenue | 9874563210 | D02     | 150000 | Manager     |
| emp10  | Renju    | 1983-01-01 | NY AVenue | 9874563310 | D10     | 70000  | TL          |
| emp11  | Rohith   | 1983-01-01 | NY AVenue | 9874563310 | D09     | 4000   | TL          |
| emp2   | Aby      | 1988-01-01 | SH AVenue | 9874563310 | D01     | 35000  | Computer    |
| emp3   | Abraham  | 1983-01-01 | SH AVenue | 9874563310 | D03     | 180000 | Computer Assistant |
| emp4   | Aiswarya | 1983-01-01 | SH AVenue | 9874563310 | D04     | 45000  | Computer Assistant |
| emp5   | Aisu     | 1983-01-01 | SH AVenue | 9874563310 | D05     | 45000  | Manager     |
| emp6   | Ram      | 1983-01-01 | SH AVenue | 9874563310 | D06     | 7000   | Clerk       |
| emp7   | Raj      | 1983-01-01 | SH AVenue | 9874563310 | D07     | 8000   | Clerk       |
| emp8   | Rohan    | 1983-01-01 | NY AVenue | 9874563310 | D08     | 18000  | TL          |
| emp9   | Rohith   | 1983-01-01 | NY AVenue | 9874563310 | D09     | 70000  | TL          |
+-----+-----+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

```
mysql> select * from department;
+-----+-----+
| dept_no | dept_name |
+-----+-----+
| D01     | MCA       |
| D02     | CSE       |
| D03     | Mech      |
| D04     | ECE       |
| D05     | EC        |
| D06     | Civil     |
| D07     | Arch      |
| D08     | BCA       |
| D09     | Bsc       |
| D10     | Maths     |
+-----+-----+
10 rows in set (0.00 sec)
```

3. Display the emp\_no and name of employees from department no 'D02'.

**SQL:** select emp\_no,emp\_name from Employee where dept\_no='D02';

**Output:**

```
mysql> select emp_no,emp_name from Employee where dept_no='D02';
+-----+-----+
| emp_no | emp_name |
+-----+-----+
| emp1   | John     |
+-----+-----+
1 row in set (0.00 sec)
```

4. Display emp\_no, emp\_name , designation, deptno and salary of employees in the descending order of salary.

**SQL:** select emp\_no, emp\_name , Designation, dept\_no,salary from Employee order by salary desc;

**Output:**

```
mysql> select emp_no, emp_name , Designation, dept_no,salary from Employee order by salary desc;
+-----+-----+-----+-----+-----+
| emp_no | emp_name | Designation | dept_no | salary |
+-----+-----+-----+-----+-----+
| emp3   | Abraham | Computer Assistant | D03     | 180000 |
| emp1   | John   | Manager       | D02     | 150000 |
| emp10  | Renju  | TL            | D10     | 70000  |
| emp9   | Rohith | TL            | D09     | 70000  |
| emp4   | Aiswarya | Computer Assistant | D04     | 45000  |
| emp5   | Aisu   | Manager       | D05     | 45000  |
| emp2   | Aby    | Computer      | D01     | 35000  |
| emp8   | Rohan  | TL            | D08     | 18000  |
| emp7   | Raj    | Clerk         | D07     | 8000   |
| emp6   | Ram    | Clerk         | D06     | 7000   |
| emp11  | Rohith | TL            | D09     | 4000   |
+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

5. Display the emp\_no , name of employees whose salary is between 2000 and 5000

**SQL:** select emp\_no,emp\_name from Employee where salary Between 2000 and 5000;

**Output:**

```
mysql> select emp_no,emp_name from Employee where salary Between 2000 and 5000;
+-----+-----+
| emp_no | emp_name |
+-----+-----+
| emp11  | Rohith   |
+-----+-----+
1 row in set (0.00 sec)
```

6. Display the designations without duplicate values

**SQL:** select DISTINCT(designation) from Employee;

**Output:**

```
mysql> select DISTINCT(designation) from Employee;
+-----+
| designation |
+-----+
| Manager     |
| TL          |
| Computer    |
| Computer Assistant |
| Clerk       |
+-----+
5 rows in set (0.00 sec)
```

7. Change the salary of employees to 45000 whose designation is 'Manager'

**SQL:** update Employee set salary=45000 where designation='Manager';

**Output:**

```
mysql> update Employee set salary=45000 where designation='Manager';
Query OK, 1 row affected (0.10 sec)
Rows matched: 2  Changed: 1  Warnings: 0

mysql> select * from Employee;
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address    | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC AVenue | 9874563210 | D02     | 45000 | Manager     |
| emp10  | Renju    | 1983-01-01 | NY AVenue  | 9874563310 | D10     | 70000 | TL          |
| emp11  | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 4000  | TL          |
| emp2   | Aby      | 1988-01-01 | SH AVenue  | 9874563310 | D01     | 35000 | Computer    |
| emp3   | Abraham  | 1983-01-01 | SH AVenue  | 9874563310 | D03     | 180000 | Computer Assistant |
| emp4   | Aiswarya | 1983-01-01 | SH AVenue  | 9874563310 | D04     | 45000 | Computer Assistant |
| emp5   | Aisu     | 1983-01-01 | SH AVenue  | 9874563310 | D05     | 45000 | Manager     |
| emp6   | Ram      | 1983-01-01 | SH AVenue  | 9874563310 | D06     | 7000  | Clerk       |
| emp7   | Raj      | 1983-01-01 | SH AVenue  | 9874563310 | D07     | 8000  | Clerk       |
| emp8   | Rohan    | 1983-01-01 | NY AVenue  | 9874563310 | D08     | 18000 | TL          |
| emp9   | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 70000 | TL          |
+-----+-----+-----+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

8. Change the mobile number of employees named John

**SQL:** update Employee set mobile\_no=6547893210 where emp\_name='John';

**Output:**

```
mysql> update Employee set mobile_no=6547893210 where emp_name='John';
Query OK, 1 row affected (0.11 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Employee;
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address    | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC AVenue | 6547893210 | D02     | 45000 | Manager     |
| emp10  | Renju    | 1983-01-01 | NY AVenue  | 9874563310 | D10     | 70000 | TL          |
| emp11  | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 4000  | TL          |
| emp2   | Aby      | 1988-01-01 | SH AVenue  | 9874563310 | D01     | 35000 | Computer    |
| emp3   | Abraham  | 1983-01-01 | SH AVenue  | 9874563310 | D03     | 180000 | Computer Assistant |
| emp4   | Aiswarya | 1983-01-01 | SH AVenue  | 9874563310 | D04     | 45000 | Computer Assistant |
| emp5   | Aisu     | 1983-01-01 | SH AVenue  | 9874563310 | D05     | 45000 | Manager     |
| emp6   | Ram      | 1983-01-01 | SH AVenue  | 9874563310 | D06     | 7000  | Clerk       |
| emp7   | Raj      | 1983-01-01 | SH AVenue  | 9874563310 | D07     | 8000  | Clerk       |
| emp8   | Rohan    | 1983-01-01 | NY AVenue  | 9874563310 | D08     | 18000 | TL          |
| emp9   | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 70000 | TL          |
+-----+-----+-----+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

9. Delete all employees whose salary is equal to Rs.7000

**SQL:** delete from Employee where salary=7000;

**Output:**

```
mysql> delete from Employee where salary=7000;
Query OK, 1 row affected (0.10 sec)

mysql> select * from Employee;
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address    | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC AVenue | 6547893210 | D02     | 45000 | Manager     |
| emp10  | Renju    | 1983-01-01 | NY AVenue  | 9874563310 | D10     | 70000 | TL          |
| emp11  | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 4000  | TL          |
| emp2   | Aby      | 1988-01-01 | SH AVenue  | 9874563310 | D01     | 35000 | Computer    |
| emp3   | Abraham  | 1983-01-01 | SH AVenue  | 9874563310 | D03     | 180000 | Computer Assistant |
| emp4   | Aiswarya | 1983-01-01 | SH AVenue  | 9874563310 | D04     | 45000 | Computer Assistant |
| emp5   | Aisu     | 1983-01-01 | SH AVenue  | 9874563310 | D05     | 45000 | Manager     |
| emp7   | Raj      | 1983-01-01 | SH AVenue  | 9874563310 | D07     | 8000  | Clerk       |
| emp8   | Rohan    | 1983-01-01 | NY AVenue  | 9874563310 | D08     | 18000 | TL          |
| emp9   | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 70000 | TL          |
+-----+-----+-----+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

10. Retrieve the name, mobile number of all employees whose name start with “A”.

**SQL:** select emp\_name,mobile\_no from Employee where emp\_name like 'A%';

**Output:**

```
mysql> select emp_name,mobile_no from Employee where emp_name like 'A%';
+-----+-----+
| emp_name | mobile_no |
+-----+-----+
| Aby      | 9874563310 |
| Abraham  | 9874563310 |
| Aiswarya | 9874563310 |
| Aisu     | 9874563310 |
+-----+-----+
4 rows in set (0.00 sec)
```

11. Display the details of the employee whose name has at least three characters and salary greater than 20000.

**SQL:** select \* from Employee where emp\_name like '\_\_\_%' and salary>20000;

**Output:**

```
mysql> select * from Employee where emp_name like '___%' and salary>20000;
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC AVenue | 6547893210 | D02     | 45000 | Manager     |
| emp10  | Renju    | 1983-01-01 | NY AVenue  | 9874563310 | D10     | 70000 | TL          |
| emp2   | Aby      | 1988-01-01 | SH AVenue  | 9874563310 | D01     | 35000 | Computer    |
| emp3   | Abraham  | 1983-01-01 | SH AVenue  | 9874563310 | D03     | 180000 | Computer Assistant |
| emp4   | Aiswarya | 1983-01-01 | SH AVenue  | 9874563310 | D04     | 45000 | Computer Assistant |
| emp5   | Aisu     | 1983-01-01 | SH AVenue  | 9874563310 | D05     | 45000 | Manager     |
| emp9   | Rohith   | 1983-01-01 | NY AVenue  | 9874563310 | D09     | 70000 | TL          |
+-----+-----+-----+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)
```

12. Display the details of employees with empid ‘emp1’, ‘emp2’ and ‘emp6’.

**SQL:** select \* from Employee where emp\_no in ('emp1','emp2','emp6');

**Output:**

```
mysql> select * from Employee where emp_no in ('emp1','emp2','emp6');
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC AVenue | 6547893210 | D02     | 45000 | Manager     |
| emp2   | Aby      | 1988-01-01 | SH AVenue  | 9874563310 | D01     | 35000 | Computer    |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

13. Display employee name and employee id of those who have salary between 120000 and 300000.

**SQL:** select emp\_no,emp\_name from Employee where salary between 120000 and 300000;

**Output:**

```
mysql> select emp_no,emp_name from Employee where salary between 120000 and 300000;
+-----+-----+
| emp_no | emp_name |
+-----+-----+
| emp3   | Abraham  |
+-----+-----+
1 row in set (0.00 sec)
```



14. Display the details of employees whose designation is 'Manager' or 'Computer Assistant'.

**SQL:** select \* from Employee where designation in ('Manager','Computer Assistant');

**Output:**

```
mysql> select * from Employee where designation in ('Manager','Computer Assistant');
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_no | emp_name | dob       | address | mobile_no | dept_no | salary | Designation |
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp1   | John     | 1985-01-01 | ABC Avenue | 6547893210 | D02     | 45000 | Manager     |
| emp3   | Abraham  | 1983-01-01 | SH Avenue | 9874563310 | D03     | 180000 | Computer Assistant |
| emp4   | Aiswarya | 1983-01-01 | SH Avenue | 9874563310 | D04     | 45000 | Computer Assistant |
| emp5   | Aisu     | 1983-01-01 | SH Avenue | 9874563310 | D05     | 45000 | Manager     |
+-----+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

15. Displays how many employees work for each department.

**SQL:** select dept\_no,count(\*) from Employee Group by dept\_no;

**Output:**

```
mysql> select dept_no,count(*) from Employee Group by dept_no;
+-----+-----+
| dept_no | count(*) |
+-----+-----+
| D01     | 1        |
| D02     | 1        |
| D03     | 1        |
| D04     | 1        |
| D05     | 1        |
| D07     | 1        |
| D08     | 1        |
| D09     | 2        |
| D10     | 1        |
+-----+-----+
9 rows in set (0.00 sec)
```

16. Displays average salary of employees in each department.

**SQL:** select dept\_no,avg(salary) from Employee Group by dept\_no;

**Output:**

```
mysql> select dept_no,avg(salary) from Employee Group by dept_no;
+-----+-----+
| dept_no | avg(salary) |
+-----+-----+
| D01     | 35000.0000 |
| D02     | 45000.0000 |
| D03     | 180000.0000 |
| D04     | 45000.0000 |
| D05     | 45000.0000 |
| D07     | 8000.0000  |
| D08     | 18000.0000 |
| D09     | 37000.0000 |
| D10     | 70000.0000 |
+-----+-----+
9 rows in set (0.00 sec)
```



17. Displays total salary of employees in each department.

**SQL:** select dept\_no,sum(salary) from Employee Group by dept\_no;

**Output:**

```
mysql> select dept_no,sum(salary) from Employee Group by dept_no;
+-----+-----+
| dept_no | sum(salary) |
+-----+-----+
| D01     | 35000       |
| D02     | 45000       |
| D03     | 180000      |
| D04     | 45000       |
| D05     | 45000       |
| D07     | 8000        |
| D08     | 18000       |
| D09     | 74000       |
| D10     | 70000       |
+-----+-----+
9 rows in set (0.00 sec)
```

18. Displays top and lower salary of employees in each department.

**SQL:** select dept\_no,Min(salary) as 'MIN Salary',Max(Salary) as 'Max Salary' from Employee Group by dept\_no;

**Output:**

```
mysql> select dept_no,Min(salary) as 'MIN Salary',Max(Salary) as 'Max Salary' from Employee Group by dept_no;
+-----+-----+-----+
| dept_no | MIN Salary | Max Salary |
+-----+-----+-----+
| D01     | 35000      | 35000      |
| D02     | 45000      | 45000      |
| D03     | 180000     | 180000     |
| D04     | 45000      | 45000      |
| D05     | 45000      | 45000      |
| D07     | 8000       | 8000       |
| D08     | 18000      | 18000      |
| D09     | 4000       | 70000      |
| D10     | 70000      | 70000      |
+-----+-----+-----+
9 rows in set (0.00 sec)
```

19. Displays average salary of employees in all departments except department with department number 'D05'.

**SQL:** select dept\_no,avg(salary) from Employee where dept\_no<>'D05' Group by dept\_no;

**Output:**

```
mysql> select dept_no,avg(salary) from Employee where dept_no<>'D05' Group by dept_no;
+-----+-----+
| dept_no | avg(salary) |
+-----+-----+
| D01     | 35000.0000   |
| D02     | 45000.0000   |
| D03     | 180000.0000  |
| D04     | 45000.0000   |
| D07     | 8000.0000    |
| D08     | 18000.0000   |
| D09     | 37000.0000   |
| D10     | 70000.0000   |
+-----+-----+
8 rows in set (0.00 sec)
```

20. Displays average salary of employees in all departments except department with department number 'D01' and average salary greater than 20000 in the ascending order of average salary.

**SQL:** select dept\_no,avg(salary) from Employee where dept\_no<>'D01' group by dept\_no having avg(salary)>20000 order by avg(salary);

**Output:**

```
mysql> select dept_no,avg(salary) from Employee where dept_no<>'D01' group by dept_no having avg(salary)>20000 order by avg(salary);
+-----+-----+
| dept_no | avg(salary) |
+-----+-----+
| D09     | 37000.0000  |
| D02     | 45000.0000  |
| D04     | 45000.0000  |
| D05     | 45000.0000  |
| D10     | 70000.0000  |
| D03     | 180000.0000 |
+-----+-----+
6 rows in set (0.01 sec)
```

**Lab Cycle 2**  
**Date : 10/04/2023**  
**Experiment No: 4**

**AIM: Familiarization of subquery, joins, views and set operations.**

A relational database consists of multiple related tables linking together using common columns, which are known as foreign key columns. A join is a method of linking data between one (self-join) or more tables based on values of the common column between the tables.

MySQL supports the following types of joins:

1. Inner join : The inner join clause includes only matching rows from both tables.  
Syntax: select column\_list FROM table\_1 INNER JOIN table\_2 ON join\_condition;
2. Left join : The left join selects data starting from the left table.  
Syntax: select column\_list FROM table\_1 LEFT JOIN table\_2 ON join\_condition;
3. Right join: The right join selects data starting from the right table.  
Syntax: select column\_list FROM table\_1 RIGHT JOIN table\_2 ON join\_condition;
4. Full outer join  
Syntax: select column\_list FROM table\_1 FULL OUTER JOIN table\_2 ON join\_condition;
5. Self join: A self join is a join in which a table is joined with itself,  
Syntax: select column\_list FROM table1 a, table1 b WHERE a.common\_field = b.common\_field;
6. Natural join: A natural join creates an implicit join by combining tables based on columns with the same name and data type.  
Syntax: SELECT [column\_names | \*] FROM table\_name1 NATURAL JOIN table\_name2;
7. Cross join: The cross join makes a Cartesian product of rows from the joined tables.  
Syntax: select column\_list FROM table\_1 CROSS JOIN table\_2;

A **subquery** is a query within a query. A select statement is embedded in a clause of another select statement. The result of the subquery is used by the main query. Subquery can be placed in the where, having or from clause.

Syntax: SELECT column\_name FROM table\_name WHERE column\_name expression operator ( SELECT COLUMN\_NAME from TABLE\_NAME WHERE ... );

A **view** is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

- Create view  
Syntax: CREATE VIEW view\_name AS SELECT column1, column2, ... FROM table\_name WHERE condition;
- Drop view  
Syntax: DROP VIEW view\_name;
- Update view  
Syntax: CREATE OR REPLACE VIEW view\_name AS SELECT column1, column2, ... FROM table\_name WHERE condition;

1. Find all employees who locate in the location with the id 1700.

**SQL:** select \* from Employees where department\_id in (select department\_id from departments where location\_id=1700);

**Output:**

```
mysql> select * from Employees where department_id in (select department_id from departments where location_id=1700);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| employee_id | first_name | last_name | email          | phone_number | hire_date | job_id | salary | manager_id | department_id |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2           | Arya       | V         | arya@gmail.com | 9874563210   | 1996-03-03 | 2      | 80000  | 2          | 2             |
| 3           | Ram        | A         | ram@gmail.com  | 9874563210   | 1996-03-03 | 1      | 80000  | 2          | 2             |
| 4           | Raj        | A         | raj@gmail.com  | 9874563210   | 1996-03-03 | 1      | 50000  | 2          | 2             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

2. Find all employees who do not locate at the location 1700.

**SQL:** select \* from Employees where department\_id not in (select department\_id from departments where location\_id=1700);

**Output:**

```
mysql> select * from Employees where department_id not in (select department_id from departments where location_id=1700);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| employee_id | first_name | last_name | email          | phone_number | hire_date | job_id | salary | manager_id | department_id |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1           | Rohith     | V         | rohith@gmail.com | 9874563210   | 1996-03-03 | 2      | 80000  | 1          | 1             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

3. Finds the employees who have the highest salary.

**SQL:** select \* from Employees where salary= (select MAX(salary) from Employees);

**Output:**

```
mysql> select * from Employees where salary= (select MAX(salary) from Employees);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| employee_id | first_name | last_name | email          | phone_number | hire_date | job_id | salary | manager_id | department_id |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1           | Rohith     | V         | rohith@gmail.com | 9874563210   | 1996-03-03 | 2      | 80000  | 1          | 1             |
| 2           | Arya       | V         | arya@gmail.com  | 9874563210   | 1996-03-03 | 2      | 80000  | 2          | 2             |
| 3           | Ram        | A         | ram@gmail.com  | 9874563210   | 1996-03-03 | 1      | 80000  | 2          | 2             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

4. Finds all employees who salaries are greater than the average salary of all employees.

**SQL:** select \* from Employees where salary>(select avg(salary) from Employees);

**Output:**

```
mysql> select * from Employees where salary>(select avg(salary) from Employees);
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| employee_id | first_name | last_name | email          | phone_number | hire_date | job_id | salary | manager_id | department_id |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1           | Rohith     | V         | rohith@gmail.com | 9874563210   | 1996-03-03 | 2      | 80000  | 1          | 1             |
| 2           | Arya       | V         | arya@gmail.com  | 9874563210   | 1996-03-03 | 2      | 80000  | 2          | 2             |
| 3           | Ram        | A         | ram@gmail.com  | 9874563210   | 1996-03-03 | 1      | 80000  | 2          | 2             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

5. Finds all departments (Department Id, Name) which have at least one employee with the salary is greater than 10,000.

**SQL:** SELECT d.department\_id, d.department\_name FROM departments d WHERE EXISTS ( SELECT 1 FROM Employees e WHERE e.department\_id = d.department\_id AND e.salary > 10000 );

**Output:**

```
mysql> SELECT d.department_id, d.department_name FROM departments d WHERE EXISTS ( SELECT 1 FROM Employees e WHERE e.department_id = d.department_id AND e.salary > 10000 );
+-----+-----+
| department_id | department_name |
+-----+-----+
| 1             | MCA              |
| 2             | MCA              |
+-----+-----+
2 rows in set (0.04 sec)
```

6. Finds all departments (Department Id, Name) that do not have any employee with the salary greater than 10,000.

**SQL:** SELECT d.department\_id, d.department\_name FROM departments d WHERE NOT EXISTS ( SELECT 1 FROM Employees e WHERE e.department\_id = d.department\_id AND e.salary > 10000 );

**Output:**

```
mysql> SELECT d.department_id, d.department_name FROM departments d WHERE NOT EXISTS ( SELECT 1 FROM Employees e WHERE e.department_id = d.department_id AND e.salary > 10000 );
```

department_id	department_name
3	EC

1 row in set (0.00 sec)

7. Finds all employees whose salaries are greater than the lowest salary of every Department.

**SQL:** SELECT e.\* FROM Employees e JOIN ( SELECT department\_id, MIN(salary) as min\_salary FROM Employees GROUP BY department\_id) AS dept\_min\_salary ON e.department\_id = dept\_min\_salary.department\_id WHERE e.salary > dept\_min\_salary.min\_salary;

**Output:**

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
2	Arya	V	arya@gmail.com	9874563210	1996-03-03	2	80000	2	2
3	Ram	A	ram@gmail.com	9874563210	1996-03-03	1	80000	2	2

2 rows in set (0.00 sec)

8. Finds all employees whose salaries are greater than or equal to the highest salary of every department.

**SQL:** SELECT e.\* FROM Employees e JOIN ( SELECT department\_id, MAX(salary) as max\_salary FROM Employees GROUP BY department\_id) AS dept\_max\_salary ON e.department\_id = dept\_max\_salary.department\_id WHERE e.salary >= dept\_max\_salary.max\_salary;

**Output:**

```
mysql> SELECT e.* FROM Employees e JOIN ( SELECT department_id, MAX(salary) as max_salary FROM Employees GROUP BY department_id) AS dept_max_salary ON e.department_id = dept_max_salary.department_id WHERE e.salary >= dept_max_salary.max_salary;
```

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
1	Rohith	V	rohith@gmail.com	9874563210	1996-03-03	2	80000	1	1
2	Arya	V	arya@gmail.com	9874563210	1996-03-03	2	80000	2	2
3	Ram	A	ram@gmail.com	9874563210	1996-03-03	1	80000	2	2
5	John	S	john123@gmail.com	9874512630	1996-10-24	1	5000	1	3

4 rows in set (0.00 sec)

9. Calculate the average of average salary of departments. (Hint: SQL subquery in the FROM clause)

**SQL:** SELECT AVG(dept\_avg\_salary) AS overall\_avg\_salary FROM (SELECT department\_id, AVG(salary) AS dept\_avg\_salary FROM Employees GROUP BY department\_id) AS dept\_avg\_salary\_subquery;

**Output:**

```
mysql> SELECT AVG(dept_avg_salary) AS overall_avg_salary FROM (SELECT department_id, AVG(salary) AS dept_avg_salary FROM Employees GROUP BY department_id) AS dept_avg_salary_subquery;
```

overall_avg_salary
51666.66666667

1 row in set (0.00 sec)

10. Finds the salaries of all employees, their average salary, and the difference between the salary of each employee and the average salary. (Hint: SQL Subquery in the SELECT clause)

**SQL:** SELECT e.salary, AVG(e.salary) AS avg\_salary, e.salary - AVG(e.salary) AS salary\_difference FROM Employees e GROUP BY e.salary ORDER BY e.salary;

**Output:**

```
mysql> SELECT e.salary, AVG(e.salary) AS avg_salary, e.salary - AVG(e.salary) AS salary_difference FROM Employees e GROUP BY e.salary ORDER BY e.salary;
```

salary	avg_salary	salary_difference
5000	5000.0000	0.0000
50000	50000.0000	0.0000
80000	80000.0000	0.0000

3 rows in set (0.00 sec)

11. Finds all employees whose salary is higher than the average salary of the employees in their departments. (Hint : Use Correlated Subquery).

**SQL:** SELECT e.\* FROM Employees e WHERE e.salary > (SELECT AVG(e2.salary) FROM Employees e2 WHERE e2.department\_id = e.department\_id);

**Output:**

```
mysql> SELECT e.* FROM Employees e WHERE e.salary > (SELECT AVG(e2.salary) FROM Employees e2 WHERE e2.department_id = e.department_id);
```

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
2	Arya	V	arya@gmail.com	9874563210	1996-03-03	2	80000	2	2
3	Ram	A	ram@gmail.com	9874563210	1996-03-03	1	80000	2	2

2 rows in set (0.00 sec)

12. Returns all employees who have no dependents.

**SQL:** select \* from Employees where employee\_id not in (select employee\_id from dependents);

**Output:**

```
mysql> select * from Employees where employee_id not in (select employee_id from dependents);
```

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
2	Arya	V	arya@gmail.com	9874563210	1996-03-03	2	80000	2	2
3	Ram	A	ram@gmail.com	9874563210	1996-03-03	1	80000	2	2
4	Raj	A	raj@gmail.com	9874563210	1996-03-03	1	50000	2	2
5	John	S	john123@gmail.com	9874512630	1996-10-24	1	5000	1	3

4 rows in set (0.00 sec)

13. Display first name, last name, department name of employees of the Department with id 1, 2 and 3.

**SQL:** select e.first\_name,e.last\_name,d.department\_name from Employees e JOIN departments d ON e.department\_id=d.department\_id where e.department\_id IN ('1','2','3');

**Output:**

```
mysql> select e.first_name,e.last_name,d.department_name from Employees e JOIN departments d ON e.department_id=d.department_id where e.department_id IN ('1','2','3');
```

first_name	last_name	department_name
Rohith	V	MCA
Arya	V	MCA
Ram	A	MCA
Raj	A	MCA
John	S	EC

5 rows in set (0.00 sec)



14. Display the first name, last name, job title, and department name of employees who work in department with id 1, 2, and 3 and salary greater than 10000.

**SQL:** select e.first\_name,e.last\_name,d.department\_name,j.job\_title from jobs j  
JOIN Employees e ON j.job\_id=e.job\_id JOIN departments d ON  
e.department\_id=d.department\_id where e.department\_id IN ('1','2','3') AND  
e.salary>10000;

**Output:**

first_name	last_name	department_name	job_title
Ram	A	MCA	Professor
Raj	A	MCA	Professor
Rohith	V	MCA	HOD
Arya	V	MCA	HOD

4 rows in set (0.02 sec)

15. Display Department name, street address, postal code, country name and region name of all departments.

**SQL:** SELECT d.department\_name, l.street\_address, l.postal\_code,  
c.country\_name, r.region\_name FROM departments d JOIN locations l ON  
d.location\_id=l.location\_id JOIN countries c ON l.country\_id = c.country\_id JOIN  
regions r ON c.region\_id = r.region\_id;

**Output:**

department_name	street_address	postal_code	country_name	region_name
EC	Nedumkuzhy	679512	India	Asia
MCA	Nedumkuzhy	679512	India	Asia
MCA	Thrissur	679512	India	Asia

3 rows in set (0.00 sec)

16. Write a SQL query to find out which employees have or do not have a department. Return first name, last name, department ID, department name.

**SQL:** SELECT e.first\_name, e.last\_name, d.department\_id, d.department\_name  
FROM Employees e LEFT JOIN departments d ON e.department\_id =  
d.department\_id;

**Output:**

```
mysql> SELECT e.first_name, e.last_name, d.department_id, d.department_name
-> FROM Employees e
-> LEFT JOIN departments d ON e.department_id = d.department_id;
```

first_name	last_name	department_id	department_name
Rohith	V	1	MCA
Arya	V	2	MCA
Ram	A	2	MCA
Raj	A	2	MCA
John	S	3	EC

5 rows in set (0.00 sec)

17. Write a SQL query to find those employees whose first name contains the letter 'Z'. Return first name, last name, department, city, and state province.

**SQL:** SELECT e.first\_name, e.last\_name, d.department\_name, l.city, l.state\_province FROM Employees e JOIN departments d ON e.department\_id = d.department\_id JOIN location l ON d.location\_id=l.location\_id WHERE e.first\_name LIKE '%Z%';

**Output:**

first_name	last_name	department_name	city	state_province
Zakheer	Taylor	MCA	Thrissur	Kerala

1 row in set (0.00 sec)

18. Write a SQL query to find all departments, including those without employees Return first name, last name, department ID, department name

**SQL:** SELECT e.first\_name, e.last\_name, d.department\_id, d.department\_name FROM departments d LEFT JOIN Employees e ON d.department\_id = e.department\_id;

**Output:**

first_name	last_name	department_id	department_name
Rohith	V	1	MCA
Zakheer	Taylor	2	MCA
Arya	V	2	MCA
akheer	Taylor	2	MCA
Ram	A	2	MCA
Raj	A	2	MCA
Smith	Taylor	2	MCA
John	S	3	EC

8 rows in set (0.01 sec)

19. Write a SQL query to find the employees and their managers. . Those managers do not work under any manager also appear in the list. Return the first name of the employee and manager.

**SQL:** SELECT e.first\_name AS employee\_first\_name, m.first\_name AS manager\_first\_name FROM Employees e LEFT JOIN Employees m ON e.manager\_id = m.employee\_id;

**Output:**

employee_first_name	manager_first_name
Rohith	Rohith
Zakheer	Smith
Arya	Arya
akheer	Zakheer
Ram	Arya
Raj	Arya
John	Rohith
Smith	Raj

8 rows in set (0.00 sec)



20. Write a SQL query to find the employees who work in the same department as the employee with the last name Taylor. Return first name, last name and department ID.

**SQL:** SELECT e.first\_name, e.last\_name, e.department\_id FROM Employees e  
INNER JOIN Employees t ON e.department\_id = t.department\_id WHERE  
t.last\_name = 'Taylor';

**Output:**

first_name	last_name	department_id
Zakheer	SV	2
Arya	V	2
akheer	SV	2
Ram	A	2
Raj	A	2
Smith	Taylor	2

6 rows in set (0.00 sec)

21. Write a SQL query to calculate the difference between the maximum salary of the job and the employee's salary. Return job title, employee name, and salary difference.

**SQL:** SELECT j.job\_title, CONCAT(e.first\_name, ' ', e.last\_name) AS  
employee\_name, (j.max\_salary - e.salary) AS salary\_difference FROM Employees e  
INNER JOIN jobs j ON e.job\_id = j.job\_id;

**Output:**

job_title	employee_name	salary_difference
Professor	Ram A	220000
Professor	Raj A	250000
Professor	John S	295000
HOD	Rohith V	2920000
HOD	Zakheer SV	2921459
HOD	Arya V	2920000
HOD	akheer SV	2921459
HOD	Smith Taylor	2921459

8 rows in set (0.00 sec)

22. Write a SQL query to calculate the average salary, the number of employees receiving commissions in that department. Return department name, average salary and number of employees of all departments.

**SQL:**

**Output:**

23. Create a view which contains employee name, employee id, phone number, job title, department name, manager name of employees belongs to department whose location is in 'Delhi' and display the details,

**SQL:** CREATE VIEW v\_employee\_details AS SELECT e.first\_name, e.last\_name,  
e.employee\_id, e.phone\_number, j.job\_title, d.department\_name,  
CONCAT(m.first\_name, ' ', m.last\_name) AS manager\_name FROM Employees e  
INNER JOIN jobs j ON e.job\_id = j.job\_id INNER JOIN departments d ON  
e.department\_id = d.department\_id LEFT JOIN Employees m ON e.manager\_id =  
m.employee\_id WHERE d.location\_id IN (SELECT location\_id FROM locations  
WHERE city = 'DELHI');

### Output:

```
mysql> CREATE VIEW v_employee_details AS
-> SELECT e.first_name, e.last_name, e.employee_id, e.phone_number, j.job_title, d.department_name, CONCAT(m.first_name, ' ', m.last_name) AS manager_name
-> FROM Employees e
-> INNER JOIN jobs j ON e.job_id = j.job_id
-> INNER JOIN departments d ON e.department_id = d.department_id
-> LEFT JOIN Employees m ON e.manager_id = m.employee_id
-> WHERE d.location_id IN (
->   SELECT location_id
->   FROM locations
->   WHERE city = 'DELHI'
-> );
Query OK, 0 rows affected (0.19 sec)
```

24. Use the above created view to obtain the names of employees whose job title is 'Manager' and department is 'Finance'.

**SQL:** SELECT first\_name, last\_name FROM v\_employee\_details WHERE job\_title = 'Manager' AND department\_name = 'Finance';

### Output:

```
+-----+-----+
| first_name | last_name |
+-----+-----+
| ABU        | Salim     |
+-----+-----+
1 row in set (0.00 sec)
```

25. Check whether it is possible to update the phone number of employee whose first name is 'Smith' by using the above created view.

**SQL:** UPDATE v\_employee\_details SET phone\_number = 6985741230 WHERE first\_name = 'Smith';

### Output:

```
mysql> UPDATE v_employee_details SET phone_number = 6985741230 WHERE first_name = 'Smith';
ERROR 1288 (HY000): The target table v_employee_details of the UPDATE is not updatable
mysql> █
```

26. Display the details of employee who have no dependents.

**SQL:** SELECT e.employee\_id, e.first\_name, e.last\_name, e.phone\_number FROM Employees e LEFT JOIN dependents dep ON e.employee\_id = dep.employee\_id WHERE dep.employee\_id IS NULL;

### Output:

```
+-----+-----+-----+-----+
| employee_id | first_name | last_name | phone_number |
+-----+-----+-----+-----+
| 101          | Zakheer   | SV        | 9874521470   |
| 102          | ABU       | Salim     | 9874521470   |
| 2            | Arya      | V         | 9874563210   |
| 201          | akheer    | SV        | 9874521470   |
| 3            | Ram       | A         | 9874563210   |
| 4            | Raj       | A         | 9874563210   |
| 5            | John      | S         | 9874512630   |
| 8            | Smith     | Taylor    | 9874521470   |
+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

27. Display the details of employee whose manager id is 101 or 201. (Use Union Clause)

**SQL:** select \* from Employees where manager\_id = '101' UNION select \* from Employees where manager\_id='201';

### Output:

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| employee_id | first_name | last_name | email          | phone_number | hire_date | job_id | salary | manager_id | department_id |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 201         | akheer    | SV        | taylor@gmail.com | 9874521470   | 2002-10-24 | 2      | 78541 | 101        | 2             |
| 104         | ABDU      | Sm        | salim@gmail.com  | 9874521470   | 2002-10-24 | 4      | 78541 | 201        | 6             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

```

28. Display the details of employees who have at least one dependent.

**SQL:** select \* from Employees where employee\_id in (select distinct(employee\_id) from dependents);

**Output:**

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| employee_id | first_name | last_name | email          | phone_number | hire_date | job_id | salary | manager_id | department_id |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1           | Rohith     | V         | rohith@gmail.com | 9874563210   | 1996-03-03 | 2      | 80000 | 1          | 1             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

```