NAME:- NABEEL MOHAMMAD RIZWAN

SUBJECT:- DATABASE MANAGEMENT SYSTEM

II-YEAR, IV-SEM

DBMS LAB FILE

Computer Engineering

Faculty of Engineering & Technology

Jamia Millia Islamia

INDEX

S.No.	Date	Assignment No.
1	31-Jan-2022	Assignment-1
2	7-Feb-2022	Assignment-2
3	14-Feb-2022	Assignment-3
4	24-Feb-2022	Assignment-4
5	13-March-2022	Assignment-5
6	13-March-2022	Assignment-6
7	21-March-2022	Assignment-7
8	3-April-2022	Assignment-8
9	9-April-2022	Assignment-9
10	24-April-2022	Assignment-10
11	1-May-2022	Assignment-11

Assignment-1

```
CREATE TABLE EMPLOYEE
        ( Fname          VARCHAR(15)        NOT NULL,
          Minit          CHAR,
          Lname          VARCHAR(15)        NOT NULL,
          Ssn            CHAR(9)            NOT NULL,
          Bdate          DATE,
          Address        VARCHAR(30),
          Sex            CHAR,
          Salary         DECIMAL(10,2),
          Super_ssn      CHAR(9),
          Dno            INT                NOT NULL,
        PRIMARY KEY (Ssn),
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) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber        INT                NOT NULL,
          Dlocation      VARCHAR(15)        NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
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) );
CREATE TABLE WORKS_ON
        ( Essn           CHAR(9)            NOT NULL,
          Pno            INT                NOT NULL,
          Hours          DECIMAL(3,1)       NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn           CHAR(9)            NOT NULL,
          Dependent_name VARCHAR(15)        NOT NULL,
          Sex            CHAR,
```

Figure 6.1 SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 5.7.

NAME:- NABEEL MOHAMMAD RIZWAN
ROLL NO:- 20BCS087
DATABASE MANAGEMENT LAB
ASSIGNMENT 1

```
mysql> USE 20BCS087_NABEEL_MOHD_RIZWAN;
mysql> create table employee(     ->
Fname VARCHAR(15) NOT NULL,
    -> Minit CHAR,
    -> Lname VARCHAR(15) NOT NULL,
    -> Ssn CHAR(9) NOT NULL,
    -> Bdate DATE,
    -> Address VARCHAR(30),
    ->
    -> Sex CHAR,
    -> Salary DECIMAL(10,2),
    -> Super_ssn VARCHAR(9),
    -> Dno INT NOT NULL,
    -> PRIMARY KEY(Ssn)
    -> );

mysql>  create table department(
-> Dname VARCHAR(15) NOT NULL,
    ->  Dnumber INT NOT NULL,
    ->  Mgr_ssn VARCHAR(9) NOT NULL,
    -> Mgr_start_date DATE,
    -> PRIMARY KEY(Dnumber),
    -> UNIQUE(Dname),
    -> FOREIGN KEY(Mgr_ssn) REFERENCES employee(Ssn)
    -> );

mysql> INSERT INTO employee values('John','B','Smith',123456789,'1965-01-09','731
Fondren,Houston,TX','M',30000,333445555,5);

mysql> insert into employee values('Franklin','T','Wong',333445555,'1955-12-08','638
Voss,Houston,TX','M',40000,888665555,5),
    -> ('Alicia','J','Zelaya',999887777,'1968-01-19','3321 Castle,Spring,TX','F',25000,987654321,4),
    -> ('Jennifer','S','Wallace',987654321,'1941-06-20','291 Berry,Bellaire,TX','F',43000,888665555,4),
    -> ('Ramesh','K','Narayan',666884444,'1962-09-15','975 Fire Oak,Humble,TX','M',38000,333445555,5);

mysql> insert into employee values('Joyce','A','English',453453453,'1972-07-31','5631
Rice,Houston,TX','F',25000,333445555,5);

mysql> insert into employee values('Ahmad','V','Jabbar',987987987,'1969-03-29','980
Dallas,Houston,TX','M',25000,987654321,4);

mysql> insert into employee(Fname,Minit,Lname,Ssn,Bdate,Address,Sex,Salary,Dno)
values('James','E','Borg',888665555,'1937-1110','450 Stone,Houston,TX','M',55000,1);

mysql> insert into department values('Research',5,333445555,'1988-05-22');

mysql> insert into department values('Administration',4,987654321,'1995-01-01'),
    ->
    -> ('Headquaters',1,888665555,'1981-06-19');
```

QUERIES:

```
mysql> select * from employee;
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+
| Fname    | Minit | Lname   | Ssn       | Bdate      | Address                | Sex  | Salary   | Super_ssn | Dno |
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+
| John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren,Houston,TX | M    | 30000.00 | 333445555 |   5 |
| Franklin | T     | Wong    | 333445555 | 1955-12-08 | 638 Voss,Houston,TX    | M    | 40000.00 | 888665555 |   5 |
| Joyce    | A     | English | 453453453 | 1972-07-31 | 5631 Rice,Houston,TX   | F    | 25000.00 | 333445555 |   5 |
| Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak,Humble,TX | M    | 38000.00 | 333445555 |   5 |
| James    | E     | Borg    | 888665555 | 1937-11-10 | 450 Stone,Houston,TX   | M    | 55000.00 | NULL      |   1 |
| Jennifer | S     | Wallace | 987654321 | 1941-06-20 | 291 Berry,Bellaire,TX  | F    | 43000.00 | 888665555 |   4 |
| Ahmad    | V     | Jabbar  | 987987987 | 1969-03-29 | 980 Dallas,Houston,TX  | M    | 25000.00 | 987654321 |   4 |
| Alicia   | J     | Zelaya  | 999887777 | 1968-01-19 | 3321 Castle,Spring,TX  | F    | 25000.00 | 987654321 |   4 |
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+

mysql> select * from department;
+----------------+---------+-----------+----------------+
| Dname          | Dnumber | Mgr_ssn   | Mgr_start_date |
+----------------+---------+-----------+----------------+
| Headquaters    |       1 | 888665555 | 1981-06-19     |
| Administration |       4 | 987654321 | 1995-01-01     |
| Research       |       5 | 333445555 | 1988-05-22     |
+----------------+---------+-----------+----------------+

mysql> select Bdate,Address
    -> from employee
    -> where Fname = 'John' and Minit='B' AND Lname = 'Smith';
+------------+------------------------+
| Bdate      | Address                |
+------------+------------------------+
| 1965-01-09 | 731 Fondren,Houston,TX |
+------------+------------------------+

mysql> select Fname,Lname,Address
    -> from employee,department
    -> where Dname = 'Research' AND Dnumber =Dno;
+----------+---------+------------------------+
| Fname    | Lname   | Address                |
+----------+---------+------------------------+
| John     | Smith   | 731 Fondren,Houston,TX |
| Franklin | Wong    | 638 Voss,Houston,TX    |
| Joyce    | English | 5631 Rice,Houston,TX   |
| Ramesh   | Narayan | 975 Fire Oak,Humble,TX |
+----------+---------+------------------------+

mysql> select employee.Fname, employee.Lname,employee.Address
    -> from employee,department
    -> where department.Dname = 'Research' AND department.Dnumber = employee.Dno;
+----------+---------+------------------------+
| Fname    | Lname   | Address                |
+----------+---------+------------------------+
| John     | Smith   | 731 Fondren,Houston,TX |
| Franklin | Wong    | 638 Voss,Houston,TX    |
| Joyce    | English | 5631 Rice,Houston,TX   |
| Ramesh   | Narayan | 975 Fire Oak,Humble,TX |
+----------+---------+------------------------+
```

```
mysql> select  all salary
    -> from employee;
+----------+
| salary   |
+----------+
| 30000.00 |
| 40000.00 |
| 25000.00 |
| 38000.00 |
| 55000.00 |
| 43000.00 |
| 25000.00 |
| 25000.00 |
```

```
mysql> select Fname, Lname
    -> from employee
    -> where address like '%Houston,TX%';
+----------+---------+
| Fname    | Lname   |
+----------+---------+
| John     | Smith   |
| Franklin | Wong    |
| Joyce    | English |
| James    | Borg    |
| Ahmad    | Jabbar  |
+----------+---------+
```

```
mysql> select Fname,Lname
    -> from employee
    -> where Bdate like '__7_______';
+-------+---------+
| Fname | Lname   |
+-------+---------+
| Joyce | English |
+-------+---------+
```

```
mysql> select *
-> from employee
    -> where(salary between 30000 AND 40000) and Dno = 5;
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+
| Fname    | Minit | Lname   | Ssn       | Bdate      | Address                | Sex  | Salary   | Super_ssn | Dno |
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+
| John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren,Houston,TX | M    | 30000.00 | 333445555 |   5 |
| Franklin | T     | Wong    | 333445555 | 1955-12-08 | 638 Voss,Houston,TX    | M    | 40000.00 | 888665555 |   5 |
| Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak,Humble,TX | M    | 38000.00 | 333445555 |   5 |
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+
```

```
mysql> select distinct salary
    -> from employee;
+----------+
| salary   |
+----------+
| 30000.00 |
| 40000.00 |
| 25000.00 |
| 38000.00 |
| 55000.00 |
| 43000.00 |
+----------+
mysql> select * from employee
    -> order by Fname;
+----------+-------+---------+-----------+------------+------------------------+------+----------+-----------+-----+
```

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas,Houston,TX	M	25000.00	987654321	4
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle,Spring,TX	F	25000.00	987654321	4
Franklin	T	Wong	333445555	1955-12-08	638 Voss,Houston,TX	M	40000.00	888665555	5
James	E	Borg	888665555	1937-11-10	450 Stone,Houston,TX	M	55000.00	NULL	1
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000.00	888665555	4
John	B	Smith	123456789	1965-01-09	731 Fondren,Houston,TX	M	30000.00	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice,Houston,TX	F	25000.00	333445555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak,Humble,TX	M	38000.00	333445555	5

```
mysql> select * from employee;
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
| Fname    | Minit | Lname   | Ssn       | Bdate      | Address               | Sex  | Salary   | Super_ssn | Dno |
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
| John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren,Houston,TX | M    | 30000.00 | 333445555 |   5 |
| Franklin | T     | Wong    | 333445555 | 1955-12-08 | 638 Voss,Houston,TX    | M    | 40000.00 | 888665555 |   5 |
| Joyce    | A     | English | 453453453 | 1972-07-31 | 5631 Rice,Houston,TX   | F    | 25000.00 | 333445555 |   5 |
| Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak,Humble,TX | M    | 38000.00 | 333445555 |   5 |
| James    | E     | Borg    | 888665555 | 1937-11-10 | 450 Stone,Houston,TX   | M    | 55000.00 | NULL      |   1 |
| Jennifer | S     | Wallace | 987654321 | 1941-06-20 | 291 Berry,Bellaire,TX  | F    | 43000.00 | 888665555 |   4 |
| Ahmad    | V     | Jabbar  | 987987987 | 1969-03-29 | 980 Dallas,Houston,TX  | M    | 25000.00 | 987654321 |   4 |
| Alicia   | J     | Zelaya  | 999887777 | 1968-01-19 | 3321 Castle,Spring,TX  | F    | 25000.00 | 987654321 |   4 |
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
8 rows in set (0.00 sec)
```

```
mysql> insert into department values('Administration',4,987654321,'1995-01-01'),
    ->
    -> ('Headquaters',1,888665555,'1981-06-19');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from department;
+---------------+---------+-----------+---------------+
| Dname         | Dnumber | Mgr_ssn   | Mgr_start_date |
+---------------+---------+-----------+---------------+
| Headquaters   |       1 | 888665555 | 1981-06-19    |
| Administration |       4 | 987654321 | 1995-01-01    |
| Research      |       5 | 333445555 | 1988-05-22    |
+---------------+---------+-----------+---------------+
3 rows in set (0.00 sec)

mysql> select Bdate,Address
    -> from employee
    -> where Fname = 'John' and Minit='B' AND Lname = 'Smith';
+------------+-----------------------+
| Bdate      | Address               |
+------------+-----------------------+
| 1965-01-09 | 731 Fondren,Houston,TX |
+------------+-----------------------+
1 row in set (0.00 sec)

mysql> select Fname,Lname,Address
    -> from employee,department
    -> where Dname = 'Research' AND Dnumber =Dno;
+----------+---------+-----------------------+
| Fname    | Lname   | Address               |
+----------+---------+-----------------------+
| John     | Smith   | 731 Fondren,Houston,TX |
| Franklin | Wong    | 638 Voss,Houston,TX   |
| Joyce    | English | 5631 Rice,Houston,TX  |
| Ramesh   | Narayan | 975 Fire Oak,Humble,TX |
+----------+---------+-----------------------+
4 rows in set (0.00 sec)
```

```
mysql> select employee.Fname, employee.Lname,employee.Address
    -> from employee,department
    -> where department.Dname = 'Research' AND department.Dnumber = employee.Dno;
+----------+---------+-----------------------+
| Fname    | Lname   | Address               |
+----------+---------+-----------------------+
| John     | Smith   | 731 Fondren,Houston,TX |
| Franklin | Wong    | 638 Voss,Houston,TX    |
| Joyce    | English | 5631 Rice,Houston,TX   |
| Ramesh   | Narayan | 975 Fire Oak,Humble,TX |
+----------+---------+-----------------------+
4 rows in set (0.00 sec)

mysql> select  all salary
    -> from employee;
+----------+
| salary   |
+----------+
| 30000.00 |
| 40000.00 |
| 25000.00 |
| 38000.00 |
| 55000.00 |
| 43000.00 |
| 25000.00 |
| 25000.00 |
+----------+
8 rows in set (0.00 sec)

mysql> select Fname, Lname
    -> from employee
    -> where address like '%Houston,TX%';
+----------+---------+
| Fname    | Lname   |
+----------+---------+
| John     | Smith   |
| Franklin | Wong    |
| Joyce    | English |
| James    | Borg    |
| Ahmad    | Jabbar  |
+----------+---------+
5 rows in set (0.00 sec)
```

```
+-------+---------+
| Fname | Lname   |
+-------+---------+
| Joyce | English |
+-------+---------+
1 row in set (0.00 sec)

mysql> select *
    -> from employee
    -> where(salary between 30000 AND 40000) and Dno = 5;
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
| Fname    | Minit | Lname   | Ssn       | Bdate      | Address               | Sex  | Salary   | Super_ssn | Dno |
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
| John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren,Houston,TX | M    | 30000.00 | 333445555 |   5 |
| Franklin | T     | Wong    | 333445555 | 1955-12-08 | 638 Voss,Houston,TX    | M    | 40000.00 | 888665555 |   5 |
| Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak,Humble,TX | M    | 38000.00 | 333445555 |   5 |
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
3 rows in set (0.00 sec)

mysql> select distinct salary
    -> from employee;
+----------+
| salary   |
+----------+
| 30000.00 |
| 40000.00 |
| 25000.00 |
| 38000.00 |
| 55000.00 |
| 43000.00 |
+----------+
6 rows in set (0.00 sec)

mysql> select * from employee
    -> order by Fname;
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
| Fname    | Minit | Lname   | Ssn       | Bdate      | Address               | Sex  | Salary   | Super_ssn | Dno |
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
| Ahmad    | V     | Jabbar  | 987987987 | 1969-03-29 | 980 Dallas,Houston,TX  | M    | 25000.00 | 987654321 |   4 |
| Alicia   | J     | Zelaya  | 999887777 | 1968-01-19 | 3321 Castle,Spring,TX  | F    | 25000.00 | 987654321 |   4 |
| Franklin | T     | Wong    | 333445555 | 1955-12-08 | 638 Voss,Houston,TX    | M    | 40000.00 | 888665555 |   5 |
| James    | E     | Borg    | 888665555 | 1937-11-10 | 450 Stone,Houston,TX   | M    | 55000.00 | NULL      |   1 |
| Jennifer | S     | Wallace | 987654321 | 1941-06-20 | 291 Berry,Bellaire,TX  | F    | 43000.00 | 888665555 |   4 |
| John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren,Houston,TX | M    | 30000.00 | 333445555 |   5 |
| Joyce    | A     | English | 453453453 | 1972-07-31 | 5631 Rice,Houston,TX   | F    | 25000.00 | 333445555 |   5 |
| Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak,Humble,TX | M    | 38000.00 | 333445555 |   5 |
+----------+-------+---------+-----------+------------+-----------------------+------+----------+-----------+-----+
```

Assignment - 2

Exercise 1: Create the tables described below

a) Table Name: Client_Master
Description: used to store information about clients.

Column_Name	Data_Type	Size
ClientNo	varchar2	6
Name	varchar2	20
Address1	varchar2	30
Address2	varchar2	30
City	varchar2	15
PinCode	Number	8
State	varchar2	15
BalDue	Number	10

b) Table Name: Product_Master
Description: Used to store information about products.

Column_Name	Data_Type	Size
ProductNo	varchar2	6
Description	varchar2	15
ProfitPercent	varchar2	4
UnitMeasure	varchar2	10
QtyOnHand	Number	8
RecorderLvi	Number	8
SellPrice	Number	8
CostPrice	Number	8

Exercise 2: Insert the following data into the tables:

a) Data for Client_Master Table.

ClientNo	Name	Address1	Address2	City	PinCode	State	BalDue
C00001	Ivan Bayross	Address1	Address2	Mumbai	400001	Maharashtra	15000
C00002	Mamta Muzumdar	Address1	Address2	Madras	780001	Tamil Nadu	0
C00003	Chhaya Bankar	Address1	Address2	Mumbai	400057	Maharashtra	5000
C00004	Ashwini Joshi	Address1	Address2	Bangalore	560001	Karnataka	0
C00005	Hansei Colaco	Address1	Address2	Mumbai	400060	Maharashtra	2000
C00006	Deepak Sharma	Address1	Address2	Mangalore	560050	Karnataka	0

b) Data for Product_Master Table.

ProductNo	ProductName	Quantity	Piece	QtyOnHand	Tax	CostPrice	SalePrice
P00001	T-Shirts	5.00	Piece	200	50	350	250
P0345	Shirts	6.00	Piece	150	50	500	350
P06734	Cotton Jeans	5.00	Piece	100	20	600	450
P07865	Jeans	5.00	Piece	100	20	750	500
P07868	Trousers	2.00	Piece	150	50	850	550
P07885	Pull Overs	2.50	Piece	150	50	850	550
P07868	Denim Shirts	4.00	Piece	100	40	350	250
P07868	Lycra Tops	5.00	Piece	70	30	300	175
P07868	Skirts	5.00	Piece	75	30	450	300

c) Data for Salesman_Master Table.

[illegible]	[illegible]	Address1	Address2	City	PinCode	State	SalAmt	TgtToGet	YtdSales	Remarks
[illegible]	Aman	A/14	Worli	Mumbai	400002	Maharashtra	3000	100	50	Good
[illegible]	Jawad	65	Nariman	Mumbai	400001	Maharashtra	3000	200	100	Good
[illegible]	Raj	P-7	Bandra	Mumbai	400032	Maharashtra	3000	200	100	Good
[illegible]	Ashish	A/5	Juhu	Mumbai	400044	Maharashtra	3500	200	150	Good

Exercise 3: Exercise on retreiving the data.

a) Find out the name of all the clients.

b) Retreive the entire contents of the Client_Master table.

c) Retreive the list of names, city and the state of al the clients.

d) List the various products available from the Product_Master teble.

e) List all the clients who live in Mumbai.

f) Find the names of salesman who have a salary equal to Rs.3000.

Exercise 4: Exercise on updating the records in the table.

a) Change the city of ClientNo 'C00005' to 'Bangalore'.

b) Change the BalDue of ClientNo 'C00001' to Rs.1000.

c) Change the cost price of 'Trousers' to Rs.950.00.

d) Change the city of the salesman to Pune.

alter user hr identified
> connect hr/hr;

c) Data for Salesman_Master Table.

SalesmanNo	SalesmanName	Address1	Address2	City	PinCode	State	SalAmt	TgtToGet	YtdSales	Remarks
S00001	Aman	A/14	Worli	Mumbai	400002	Maharashtra	3000	100	50	Good
S00002	Omkar	65	Nariman	Mumbai	400001	Maharashtra	3000	200	100	Good
S00003	Raj	P-7	Bandra	Mumbai	400032	Maharashtra	3000	200	100	Good
S00004	Ashish	A/5	Juhu	Mumbai	400044	Maharashtra	3500	200	150	Good

Name - Nabeel Mohammad Rizwan
Roll No.- 20BCS087
Program 2
DBMS

INPUT:

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;
```

Exercise 1:

```
mysql> create table client_master(
    -> ClientNo varchar(6),
    -> Name varchar(20),
    -> Address1 varchar(30),
    -> Address2 varchar(30),
    -> City varchar(15),
    -> Pincode int,
    -> State varchar(15),
    -> BalDue int
    -> );

mysql> create table product_master(
    -> ProductNo varchar(6),
    -> ProductName varchar(15),
    -> Quantity decimal(3,2),
    -> Piece varchar(15),
    -> QtyOnHand int,
    -> Tax int,
    -> CostPrice int,
    -> SalePrice int
    -> );
```

Exercise 2:

```
mysql> insert into client_master values('C00001','Ivan
Bayross','Address1','Address2','Mumbai',400001,'Maharashtra',35000);

mysql> insert into client_master values('C00002','Mamta Muzumdar','Address1','Address2','Madras',780001,'Tamil
Nadu',0),
    -> ('C00003','Chhaya Bankar','Address1','Address2','Mumbai',400057,'Maharashtra',5000),
    -> ('C00004','Ashwini Joshi','Address1','Address2','Banglore',560001,'Karnataka',0),
    -> ('C00005','Hansei Colaco','Address1','Address2','Mumbai',400060,'Maharashtra',2000),
    -> ('C00006','Deepak Sharma','Address1','Address2','Manglore',560050,'Karnataka',0);

mysql> select * from client_master;
+----------+----------------+----------+----------+----------+---------+-------------+--------+
| ClientNo | Name           | Address1 | Address2 | City     | Pincode | State       | BalDue |
+----------+----------------+----------+----------+----------+---------+-------------+--------+
| C00001   | Ivan Bayross   | Address1 | Address2 | Mumbai   |  400001 | Maharashtra |  35000 |
| C00002   | Mamta Muzumdar | Address1 | Address2 | Madras   |  780001 | Tamil Nadu  |      0 |
| C00003   | Chhaya Bankar  | Address1 | Address2 | Mumbai   |  400057 | Maharashtra |   5000 |
| C00004   | Ashwini Joshi  | Address1 | Address2 | Banglore |  560001 | Karnataka   |      0 |
| C00005   | Hansei Colaco  | Address1 | Address2 | Mumbai   |  400060 | Maharashtra |   2000 |
| C00006   | Deepak Sharma  | Address1 | Address2 | Manglore |  560050 | Karnataka   |      0 |
+----------+----------------+----------+----------+----------+---------+-------------+--------+

mysql> insert into product_master values('P00001','T-shirts',5.00,'piece',200,50,350,250);

mysql> insert into product_master values('P0345','Shirts',6.00,'piece',150,50,500,350),
    -> ('P06734','Cotton Jeans',5.00,'piece',100,20,600,450),
    -> ('P07865','Jeans',5.00,'piece',100,20,750,500),
    -> ('P07865','Trousers',2.00,'piece',150,50,850,550),
    -> ('P07885','Pull Overs',2.50,'piece',150,50,850,550),
    -> ('P07868','Denim Shirts',4.00,'piece',100,40,350,250),
    -> ('P07868','Lycra Tops',5.00,'piece',70,30,300,175),
    -> ('P07868','Skirts',5.00,'piece',75,30,450,300);
```

```
mysql> select * from product_master;
+-----------+--------------+----------+-------+-----------+------+-----------+-----------+
| ProductNo | ProductName  | Quantity | Piece | QtyOnHand | Tax  | CostPrice | SalePrice |
+-----------+--------------+----------+-------+-----------+------+-----------+-----------+
| P00001    | T-shirts     |     5.00 | piece |       200 |   50 |       350 |       250 |
| P0345     | Shirts       |     6.00 | piece |       150 |   50 |       500 |       350 |
| P06734    | Cotton Jeans |     5.00 | piece |       100 |   20 |       600 |       450 |
| P07865    | Jeans        |     5.00 | piece |       100 |   20 |       750 |       500 |
| P07865    | Trousers     |     2.00 | piece |       150 |   50 |       850 |       550 |
| P07885    | Pull Overs   |     2.50 | piece |       150 |   50 |       850 |       550 |
| P07868    | Denim Shirts |     4.00 | piece |       100 |   40 |       350 |       250 |
| P07868    | Lycra Tops   |     5.00 | piece |        70 |   30 |       300 |       175 |
| P07868    | Skirts       |     5.00 | piece |        75 |   30 |       450 |       300 |
+-----------+--------------+----------+-------+-----------+------+-----------+-----------+
```

QUERIES:

Exercise 3:

a)Find out the names of all the clients.

```
mysql> select Name from client_master;
+----------------+
| Name           |
+----------------+
| Ivan Bayross   |
| Mamta Muzumdar |
| Chhaya Bankar  |
| Ashwini Joshi  |
| Hansei Colaco  |
| Deepak Sharma  |
+----------------+
```

b)Retreive the entire contents of the client_master table.

```
mysql> select * from client_master;
+----------+----------------+----------+----------+----------+---------+-------------+--------+
| ClientNo | Name           | Address1 | Address2 | City     | Pincode | State       | BalDue |
+----------+----------------+----------+----------+----------+---------+-------------+--------+
| C00001   | Ivan Bayross   | Address1 | Address2 | Mumbai   |  400001 | Maharashtra |  35000 |
| C00002   | Mamta Muzumdar | Address1 | Address2 | Madras   |  780001 | Tamil Nadu  |      0 |
| C00003   | Chhaya Bankar  | Address1 | Address2 | Mumbai   |  400057 | Maharashtra |   5000 |
| C00004   | Ashwini Joshi  | Address1 | Address2 | Banglore |  560001 | Karnataka   |      0 |
| C00005   | Hansei Colaco  | Address1 | Address2 | Mumbai   |  400060 | Maharashtra |   2000 |
| C00006   | Deepak Sharma  | Address1 | Address2 | Manglore |  560050 | Karnataka   |      0 |
+----------+----------------+----------+----------+----------+---------+-------------+--------+
```

c)Retreive the list of names, city and the states of all the clients.

```
mysql> select Name, City, State
    -> from client_master;
+----------------+----------+-------------+
| Name           | City     | State       |
+----------------+----------+-------------+
| Ivan Bayross   | Mumbai   | Maharashtra |
| Mamta Muzumdar | Madras   | Tamil Nadu  |
| Chhaya Bankar  | Mumbai   | Maharashtra |
| Ashwini Joshi  | Banglore | Karnataka   |
| Hansei Colaco  | Mumbai   | Maharashtra |
| Deepak Sharma  | Manglore | Karnataka   |
+----------------+----------+-------------+
```

d)List the various products available from the product_master table.

```
mysql> select ProductName
    -> from product_master;
+--------------+
| ProductName  |
+--------------+
| T-shirts     |
| Shirts       |
| Cotton Jeans |
| Jeans        |
| Trousers     |
| Pull Overs   |
| Denim Shirts |
| Lycra Tops   |
| Skirts       |
+--------------+
```

e)List all the clients who live in Mumbai.

```
mysql> select Name
    -> from client_master
    -> where City = 'Mumbai';
+---------------+
| Name          |
+---------------+
| Ivan Bayross  |
| Chhaya Bankar |
| Hansei Colaco |
+---------------+
```

Exercise 4:

a) Change the city of clientNo ‘C00005’ to ‘Banglore’.

```
mysql> update client_master
    -> set city = 'Banglore'
    -> where ClientNo = 'C00005';
```

b)Change the BalDue of ClientNo ‘C00001’ to Rs. 1000.

```
mysql> update client_master
    ->
    -> set BalDue = 1000
    -> where ClientNo = 'C00001';
```

c)Change the cost price of ‘Trousers’ to Rs.950.00.

```
mysql> update product_master
    -> set CostPrice = 950
    -> where ProductName = 'Trousers';
```

Assignment-3

Exercise 1: Create the tables described below

a) Table Name: Client_Master
Description: used to store information about clients.

Column_Name	Data_Type	Size
ClientNo	varchar2	6
Name	varchar2	20
Address1	varchar2	30
Address2	varchar2	30
City	varchar2	15
PinCode	Number	8
State	varchar2	15
BalDue	Number	10

b) Table Name: Product_Master
Description: Used to store information about products.

Column_Name	Data_Type	Size
ProductNo	varchar2	6
Description	varchar2	15
ProfitPercent	varchar2	4
UnitMeasure	varchar2	10
QtyOnHand	Number	8
RecorderLvi	Number	8
SellPrice	Number	8
CostPrice	Number	8

c) Table Name: Salesman_Master
Description: Used to store information about salesman working in the company.

Column_Name	Data_Type	Size
SalesmanNo	varchar2	6
SalesmanName	varchar2	20
Address1	varchar2	30
Address2	varchar2	30
City	varchar2	20
PinCode	Number	8
State	varchar2	20
SalAmt	Number	8
TgttoGet	Number	6
YtdSales	Number	6
Remarks	varchar2	60

Exercise 2: Insert the following data into the tables:

a) Data for Client_Master Table.

ClientNo	Name	Address1	Address2	City	PinCode	State	BailDue
C00001	Ivan Bayross	Address1	Address2	Mumbai	400001	Maharashtra	15000
C00002	Mamta Muzumdar	Address1	Address2	Madras	780001	Tamil Nadu	0
C00003	Chhaya Bankar	Address1	Address2	Mumbai	400057	Maharashtra	5000
C00004	Ashwini Joshi	Address1	Address2	Bangalore	560001	Karnataka	0
C00005	Hansei Colaco	Address1	Address2	Mumbai	400060	Maharashtra	2000
C00006	Deepak Sharma	Address1	Address2	Mangalore	560050	Karnataka	0

b) Data for Product_Master Table.

ProductNo	ProductName	Quantity	Piece	QtyOnHand	Tax	CostPrice	SalePrice
P00001	T-Shirts	5.00	Piece	200	50	350	250
P0345	Shirts	6.00	Piece	150	50	500	350
P06734	Cotton Jeans	5.00	Piece	100	20	600	450
P07865	Jeans	5.00	Piece	100	20	750	500
P07868	Trousers	2.00	Piece	150	50	850	550
P07885	Pull Overs	2.50	Piece	150	50	850	550
P07868	Denim Shirts	4.00	Piece	100	40	350	250
P07868	Lycra Tops	5.00	Piece	70	30	300	175
P07868	Skirts	5.00	Piece	75	30	450	300

b) Data for Salesman_Master Table.

SalesmanNo	SalesmanName	Address1	Address2	City	PinCode	State	SalAmt	TgtToGet	YtdSales	Remarks
S00001	Aman	A/14	Worli	Mumbai	400002	Maharashtra	3000	100	50	Good
S00002	Omkar	65	Nariman	Mumbai	400001	Maharashtra	3000	200	100	Good
S00003	Raj	P-7	Bandra	Mumbai	400032	Maharashtra	3000	200	100	Good
S00004	Ashish	A/5	Juhu	Mumbai	400044	Maharashtra	3500	200	150	Good

Exercise 3: Exercise on retreiving the data.

a) Find out the name of all the clients.

b) Retreive the entire contents of the Client_Master table.

c) Retreive the list of names,city and the state of al the clients.

d) List the various products available from the Product_Master teble.

e) List all the clients who live in Mumbai.

f) Find the names of salesman who have a salary equal to Rs.3000.

Exercise 4: Exercise on updating the records in the table.

a) Change the city of ClientNo ‘C00005’ to ‘Bangalore’.

b) Change the BalDue of ClientNo ‘C00001’ to Rs.1000.

c) Change the cost price of ‘Trousers’ to Rs.950.00.

d) Change the city of the salesman to Pune.

Exercise 5: Exercise on deleting the records in the table.

a) Delete all the salesman from the Salesman_Master whose salaries are equal
to 3500.

b) Delete all products from Product_Master where the QtyOnHand is equal to

100.

c) Delete from Client where the column state =’Maharashtra’.

Exercise 6: Exercise on altering the table structure.

d) Add a column called ‘Telephone’ of data type ‘number’ and size ‘10’ to the
Client_Master table.

e) Change the size ODF SellPrice column Product_Master to 10,2.

Exercise 7: Exercise on deleting the table structure.

f) Destroy table Client_Master along with its data.

Exercise 8: Exercise on renaming the table.

g) Change the name of the Salesman_Master to sman_mast.

NAME- NABEEL MOHAMMAD RIZWAN
ROLL NO:- 20BCS087
ASG-3
DBMS LAB

Assignment-3

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;

mysql> select * from client_master;
+----------+----------------+----------+----------+----------+---------+-------------+--------+
| ClientNo | Name           | Address1 | Address2 | City     | Pincode | State       | BalDue |
+----------+----------------+----------+----------+----------+---------+-------------+--------+
| C00001   | Ivan Bayross   | Address1 | Address2 | Mumbai   |  400001 | Maharashtra |   1000 |
| C00002   | Mamta Muzumdar | Address1 | Address2 | Madras   |  780001 | Tamil Nadu  |      0 |
| C00003   | Chhaya Bankar  | Address1 | Address2 | Mumbai   |  400057 | Maharashtra |   5000 |
| C00004   | Ashwini Joshi  | Address1 | Address2 | Banglore |  560001 | Karnataka   |      0 |
| C00005   | Hansei Colaco  | Address1 | Address2 | Banglore |  400060 | Maharashtra |   2000 |
| C00006   | Deepak Sharma  | Address1 | Address2 | Manglore |  560050 | Karnataka   |      0 |
+----------+----------------+----------+----------+----------+---------+-------------+--------+

mysql> select * from product_master
    -> ;
+-----------+--------------+----------+-------+-----------+------+-----------+-----------+
| ProductNo | ProductName  | Quantity | Piece | QtyOnHand | Tax  | CostPrice | SalePrice |
+-----------+--------------+----------+-------+-----------+------+-----------+-----------+
| P00001    | T-shirts     |     5.00 | piece |       200 |   50 |       350 |       250 |
| P0345     | Shirts       |     6.00 | piece |       150 |   50 |       500 |       350 |
| P06734    | Cotton Jeans |     5.00 | piece |       100 |   20 |       600 |       450 |
| P07865    | Jeans        |     5.00 | piece |       100 |   20 |       750 |       500 |
| P07865    | Trousers     |     2.00 | piece |       150 |   50 |       950 |       550 |
| P07885    | Pull Overs   |     2.50 | piece |       150 |   50 |       850 |       550 |
| P07868    | Denim Shirts |     4.00 | piece |       100 |   40 |       350 |       250 |
| P07868    | Lycra Tops   |     5.00 | piece |        70 |   30 |       300 |       175 |
| P07868    | Skirts       |     5.00 | piece |        75 |   30 |       450 |       300 |
+-----------+--------------+----------+-------+-----------+------+-----------+-----------+

mysql> select * from salesman_master;
+------------+-------------+---------+---------+--------+---------+-------------+--------+----------+----------+---------+
| SalesmanNo | SalesmanName | Address1 | Address2 | City   | Pincode | state       | SalAmt | TgttoGet | YtdSales | Remarks |
+------------+-------------+---------+---------+--------+---------+-------------+--------+----------+----------+---------+
| S00001     | Aman        | A/14    | Worli   | Mumbai |  400002 | Maharashtra |   3000 |      100 |       50 | Good    |
| S00002     | Omkar       | 65      | Nariman | Mumbai |  400001 | Maharashtra |   3000 |      200 |      100 | Good    |
| S00003     | Raj         | P-7     | Bandra  | Mumbai |  400032 | Maharashtra |   3000 |      200 |      100 | Good    |
| S00004     | Ashish      | A/5     | Juhu    | Mumbai |  400044 | Maharashtra |   3500 |      200 |      150 | Good    |
+------------+-------------+---------+---------+--------+---------+-------------+--------+----------+----------+---------+
```

Exercise 5:

Q)Delete all the salesman from salesman_master whose salaries are equal to 3500

```
mysql> delete from salesman_master
    -> where SalAmt = 3500;
```

Q)Delete all products from product_master where the QtyOnHand is equal to 100

```
mysql> delete from product_master
    -> where QtyOnHand = 100;
```

Q)Delete from client where the column state=’Maharashtra’

```
mysql> delete from client_master
    -> where State = 'Maharashtra';
```

Exercise 6:

Q)Add a column called ‘Telephone of data type ‘number’ and size ‘10’ to the client_master table.

```
mysql> alter table client_master add telephone int;
```

Q)Change the size ODF SalePrice column product_master to 10,2

```
mysql> alter table product_master
    -> modify SalePrice
    -> decimal(10,2);
```

Exercise 7:

Q)Destroy table client_master along with its data

```
mysql> drop table client_master;
```

Exercise 8:

Q)Change the name of the salesman_master to sman_mast

```
mysql> alter table salesman_master
    -> rename to sman_mast;
```

Assignment-4

Create a relational database that contains the following tables and insert the following data into these tables.

department:

DeptID	DeptName
1	Information Technology
2	Electrical
3	Civil
4	Mechanical
5	Chemical

stud_member:

RollNo	FName	MName	SName	DeptID	Semester	ContactNo	Gender
1	Ankur	Samir	Kahar	1	1	272121	M
2	Dhaval	Dhiren	Joshi	1	1	232122	M
3	Ankita	Biren	Shah	1	1	112121	F
10	Komal	MaheshKumar	Pandya	2	3	123123	F
13	Amit	Jitenkumar	Mehta	3	3	453667	M
23	Jinal	Ashish	Gandhi	2	1	323232	M
22	Ganesh	Asha	Patel	2	3	124244	M
4	Shweta	Mihir	Patel	3	1	646341	F
7	Pooja	Mayaank	Desai	3	3	328656	F
8	Komal	Krishnaraj	Bhatia	2	3	257422	F
43	Kiran	Viraj	Shah	1	1	754124	F

Now, solve the following SQL Queries:

1. Display the names and contact numbers of all student members.

2. Give the names and roll number of all students of Information Technology who are the members.

3. Display names of Departments whose students are members.

4. Display names of Departments for which no student are members.

5. Display names of all Departments.

6. Find the number of students of Electrical Department who are members.

7. Display information of student members whose name begins with the letter 'A'.

8. Display all details of Male members only.

9. Display data of student members who are currently in semester 3.

10. Display data of student female members in alphabetical order.

NAME- NABEEL MOHAMMAD RIZWAN
ROLL NO:- 20BCS087
DBMS LAB
ASSIGNMENT

Assignment-4

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;

mysql> select * from department;
+--------+------------------------+
| DeptID | DeptName               |
+--------+------------------------+
|      1 | Information Technology |
|      2 | Electrical             |
|      3 | Civil                  |
|      4 | Mechanical             |
|      5 | Chemical               |
+--------+------------------------+

mysql> select * from stud_member;
+--------+--------+-------------+--------+--------+----------+-----------+--------+
| RollNo | Fname  | Mname       | Sname  | DeptID | semester | ContactNo | Gender |
+--------+--------+-------------+--------+--------+----------+-----------+--------+
|      1 | Ankur  | Samir       | Kahar  |      1 |        1 |    272121 | M      |
|      2 | Dhaval | Dhiren      | Joshi  |      1 |        1 |    232122 | M      |
|      3 | Ankita | Biren       | Shah   |      1 |        1 |    112121 | F      |
|     10 | Komal  | MaheshKumar | Pandya |      2 |        3 |    123123 | F      |
|     13 | Amit   | JitenKumar  | Mehta  |      3 |        3 |    453667 | M      |
|     23 | Jinal  | Ashish      | Gandhi |      2 |        1 |    323232 | M      |
|     22 | Ganesh | Asha        | Patel  |      2 |        3 |    124244 | M      |
|      4 | Shweta | Mihir       | Patel  |      3 |        1 |    646341 | F      |
|      7 | Pooja  | Mayaank     | Desai  |      3 |        3 |    328656 | F      |
|      8 | Komal  | Krishnaraj  | Bhatia |      2 |        3 |    257422 | F      |
|     43 | Kiran  | Viraj       | Shah   |      1 |        1 |    754124 | F      |
+--------+--------+-------------+--------+--------+----------+-----------+--------+
```

1)Display the names and contact numbers of all student members.

```
mysql> select Fname,Mname,Sname,ContactNo
    -> from stud_member;
+--------+-------------+--------+-----------+
| Fname  | Mname       | Sname  | ContactNo |
+--------+-------------+--------+-----------+
| Ankur  | Samir       | Kahar  |    272121 |
| Dhaval | Dhiren      | Joshi  |    232122 |
| Ankita | Biren       | Shah   |    112121 |
| Komal  | MaheshKumar | Pandya |    123123 |
| Amit   | JitenKumar  | Mehta  |    453667 |
| Jinal  | Ashish      | Gandhi |    323232 |
| Ganesh | Asha        | Patel  |    124244 |
| Shweta | Mihir       | Patel  |    646341 |
| Pooja  | Mayaank     | Desai  |    328656 |
| Komal  | Krishnaraj  | Bhatia |    257422 |
| Kiran  | Viraj       | Shah   |    754124 |
+--------+-------------+--------+-----------+
```

2)Give the names and roll number of all students of information technology who are the members

```
mysql> select Fname,Mname,Sname,RollNo from stud_member a natural join department
    -> where department.DeptName='Information Technology';
+--------+--------+-------+--------+
| Fname  | Mname  | Sname | RollNo |
+--------+--------+-------+--------+
| Ankur  | Samir  | Kahar |      1 |
| Dhaval | Dhiren | Joshi |      2 |
| Ankita | Biren  | Shah  |      3 |
| Kiran  | Viraj  | Shah  |     43 |
+--------+--------+-------+--------+
```

3)Display names of departments whose students are members.

```
mysql> select DeptID ,DeptName
    -> from stud_member a natural join department;
+--------+------------------------+
| DeptID | DeptName               |
+--------+------------------------+
|      1 | Information Technology |
|      1 | Information Technology |
|      1 | Information Technology |
|      2 | Electrical             |
|      3 | Civil                  |
|      2 | Electrical             |
|      2 | Electrical             |
|      3 | Civil                  |
|      3 | Civil                  |
|      2 | Electrical             |
|      1 | Information Technology |
+--------+------------------------+
```

4)Display names of departments for which no student are members.

```
mysql> select * from department
    -> where department.DeptID not in(select DeptID from stud_member);
+--------+------------+
| DeptID | DeptName   |
+--------+------------+
|      4 | Mechanical |
|      5 | Chemical   |
+--------+------------+
```

5)Display names of all departments.

```
mysql> select * from department;
+--------+------------------------+
| DeptID | DeptName               |
+--------+------------------------+
|      1 | Information Technology |
|      2 | Electrical             |
|      3 | Civil                  |
|      4 | Mechanical             |
|      5 | Chemical               |
+--------+------------------------+

mysql> select DeptName from department;
+------------------------+
| DeptName               |
+------------------------+
| Information Technology |
| Electrical             |
| Civil                  |
| Mechanical             |
| Chemical               |
+------------------------+
```

6)Find the number of students of Electrical Department who are members.

```
mysql> select count(DeptID)
    -> from stud_member
    -> where DeptID=2;
+---------------+
| count(DeptID) |
+---------------+
|             4 |
+---------------+
```

7)Display information of student members whose name begins with the letter ‘A’.

```
mysql> select * from stud_member
    -> where Fname like 'A%';
+--------+--------+------------+--------+--------+----------+-----------+--------+
| RollNo | Fname  | Mname      | Sname  | DeptID | semester | ContactNo | Gender |
+--------+--------+------------+--------+--------+----------+-----------+--------+
|      1 | Ankur  | Samir      | Kahar  |      1 |        1 |    272121 | M      |
|      3 | Ankita | Biren      | Shah   |      1 |        1 |    112121 | F      |
|     13 | Amit   | JitenKumar | Mehta  |      3 |        3 |    453667 | M      |
+--------+--------+------------+--------+--------+----------+-----------+--------+
```

8)Display all details of Male members only.

```
mysql> select * from stud_member
    -> where Gender='M';
+--------+--------+------------+--------+--------+----------+-----------+--------+
| RollNo | Fname  | Mname      | Sname  | DeptID | semester | ContactNo | Gender |
+--------+--------+------------+--------+--------+----------+-----------+--------+
|      1 | Ankur  | Samir      | Kahar  |      1 |        1 |    272121 | M      |
|      2 | Dhaval | Dhiren     | Joshi  |      1 |        1 |    232122 | M      |
|     13 | Amit   | JitenKumar | Mehta  |      3 |        3 |    453667 | M      |
|     23 | Jinal  | Ashish     | Gandhi |      2 |        1 |    323232 | M      |
|     22 | Ganesh | Asha       | Patel  |      2 |        3 |    124244 | M      |
+--------+--------+------------+--------+--------+----------+-----------+--------+
```

9)Display data of student members who are currently in semester 3.

```
mysql> select * from stud_member
    -> where semester=3;
+--------+--------+-------------+--------+--------+----------+-----------+--------+
| RollNo | Fname  | Mname       | Sname  | DeptID | semester | ContactNo | Gender |
+--------+--------+-------------+--------+--------+----------+-----------+--------+
|     10 | Komal  | MaheshKumar | Pandya |      2 |        3 |    123123 | F      |
|     13 | Amit   | JitenKumar  | Mehta  |      3 |        3 |    453667 | M      |
|     22 | Ganesh | Asha        | Patel  |      2 |        3 |    124244 | M      |
|      7 | Pooja  | Mayaank     | Desai  |      3 |        3 |    328656 | F      |
|      8 | Komal  | Krishnaraj  | Bhatia |      2 |        3 |    257422 | F      |
+--------+--------+-------------+--------+--------+----------+-----------+--------+
```

10)Display data of student female members in alphabetical order.

```
mysql> select * from stud_member
    -> where Gender='F'
    -> order by Fname;
+--------+--------+-------------+--------+--------+----------+-----------+--------+
| RollNo | Fname  | Mname       | Sname  | DeptID | semester | ContactNo | Gender |
+--------+--------+-------------+--------+--------+----------+-----------+--------+
|      3 | Ankita | Biren       | Shah   |      1 |        1 |    112121 | F      |
|     43 | Kiran  | Viraj       | Shah   |      1 |        1 |    754124 | F      |
|     10 | Komal  | MaheshKumar | Pandya |      2 |        3 |    123123 | F      |
|      8 | Komal  | Krishnaraj  | Bhatia |      2 |        3 |    257422 | F      |
|      7 | Pooja  | Mayaank     | Desai  |      3 |        3 |    328656 | F      |
|      4 | Shweta | Mihir       | Patel  |      3 |        1 |    646341 | F      |
+--------+--------+-------------+--------+--------+----------+-----------+--------+
```

Assignment-5

Questions for Lab Assignment-5

Create the tables and solve the following queries:

Table 1: Employee

Emp_id	Emp_name	Salary	DNo
101	Amit	25000	D1001
102	Sunil	20000	D1002
103	Rakesh	18000	D1003
104	Ajay	16000	D1001
105	Suhail	20000	D1002
106	Arif	18000	D1004
107	Suresh	24000	D1002
108	Vijay	22000	D1003

Table 2: Department

DNo	Dept_name
D1001	IT
D1002	Sales
D1003	Marketing
D1004	HR

1. Display total sum required to pay the salary of all employees.

2. Display the average salary, minimum salary and maximum salary of the Company.

3. Display the sum of salary department-wise.

4. Display the maximum salary department-wise.

5 Display the details of the employee who earns the maximum salary.

6. Display details of every employee having maximum salary in his Department.

7. Display the details of the employee who earns more salary than the average salary of his department.

8. Display total number of employees in each department along with the department name.

Name- Nabeel Mohammad Rizwan
Roll No:- 20BCS087

Assignment-5

DBMS Lab

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;

mysql> create table employee(
    -> Emp_id int primary key,
    -> Emp_name varchar(20),
    -> Salary int,
    -> Dno varchar(20)
    -> );

mysql> insert into employee values(102,'Sunil',20000,'D1002'),
    -> (103,'Rakesh',18000,'D1003'),
    -> (104,'Ajay',16000,'D1001'),
    -> (105,'Suhail',20000,'D1002'),
    -> (106,'Arif',18000,'D1004'),
    -> (107,'Suresh',24000,'D1002'),
    -> (108,'Vijay',22000,'D1003');

mysql> select * from employee;
+--------+----------+--------+-------+
| Emp_id | Emp_name | Salary | Dno   |
+--------+----------+--------+-------+
|    101 | Amit     |  25000 | D1001 |
|    102 | Sunil    |  20000 | D1002 |
|    103 | Rakesh   |  18000 | D1003 |
|    104 | Ajay     |  16000 | D1001 |
|    105 | Suhail   |  20000 | D1002 |
|    106 | Arif     |  18000 | D1004 |
|    107 | Suresh   |  24000 | D1002 |
|    108 | Vijay    |  22000 | D1003 |
+--------+----------+--------+-------+


mysql> create table department(
    -> Dno varchar(20),
    -> Dept_name varchar(20)
    -> );

mysql> insert into department values('D1001','IT'),
    -> ('D1002','Sales'),
    -> ('D1003','Marketing'),
    -> ('D1004','HR');

mysql> select * from department;
+-------+-----------+
| Dno   | Dept_name |
+-------+-----------+
| D1001 | IT        |
| D1002 | Sales     |
| D1003 | Marketing |
| D1004 | HR        |
+-------+-----------+
```

1) Display total sum required to pay salary of all employees.

```
mysql> select sum(salary)
    -> from employee;
+-------------+
| sum(salary) |
+-------------+
|      163000 |
+-------------+
```

2) Display the average salary, minimum salary and maximum salary of the company.

```
mysql> select avg(salary), min(salary), max(salary)
    -> from employee;
+-------------+-------------+-------------+
| avg(salary) | min(salary) | max(salary) |
+-------------+-------------+-------------+
|  20375.0000 |       16000 |       25000 |
+-------------+-------------+-------------+
```

3) Display the sum of salary department-wise.

```
mysql> select sum(salary)
    -> from employee
    -> group by Dno;
+-------------+
| sum(salary) |
+-------------+
|       41000 |
|       64000 |
|       40000 |
|       18000 |
+-------------+
```

4) Display the maximum salary department-wise.

```
mysql> select department.Dept_name, max(employee.salary)
    -> from department
    -> join employee
    -> on department.Dno = employee.Dno
    -> group by Dept_name;
+-----------+----------------------+
| Dept_name | max(employee.salary) |
+-----------+----------------------+
| Sales     |                24000 |
| Marketing |                22000 |
| IT        |                16000 |
| HR        |                18000 |
+-----------+----------------------+
```

5) Display the details of the employee who earns the maximum salary.

```
mysql> select * from employee
    -> where salary = (select max(salary) from employee);
+--------+----------+--------+-------+
| Emp_id | Emp_name | Salary | Dno   |
+--------+----------+--------+-------+
|    101 | Amit     |  25000 | D1001 |
+--------+----------+--------+-------+
```

6) Display details of every employee having maximum salary in his department.

```
mysql> select * from employee
    -> where salary in (select max(salary) from employee group by Dno);
+--------+----------+--------+-------+
| Emp_id | Emp_name | Salary | Dno   |
+--------+----------+--------+-------+
|    101 | Amit     |  25000 | D1001 |
|    106 | Arif     |  18000 | D1004 |
|    107 | Suresh   |  24000 | D1002 |
|    108 | Vijay    |  22000 | D1003 |
+--------+----------+--------+-------+
```

7) Display the details of the employee who earns more salary than the average salary of the department.

```
mysql> select employee.*, department.Dept_name
    -> from employee, department
    -> where employee.Dno = department.Dno and salary > (select avg(salary) from employee);
+--------+----------+--------+-------+-----------+
| Emp_id | Emp_name | Salary | Dno   | Dept_name |
+--------+----------+--------+-------+-----------+
|    107 | Suresh   |  24000 | D1002 | Sales     |
|    105 | Suhail   |  20000 | D1002 | Sales     |
|    102 | Sunil    |  20000 | D1002 | Sales     |
|    108 | Vijay    |  22000 | D1003 | Marketing |
+--------+----------+--------+-------+-----------+
```

8) Display total number of employees in each department along with the depertment name.

```
mysql> select department.Dept_name, count(employee.Emp_id)
    -> from department
    -> join employee
    -> on department.Dno = employee.Dno
    -> group by dept_name;
+-----------+------------------------+
| Dept_name | count(employee.Emp_id) |
+-----------+------------------------+
| Sales     |                      3 |
| Marketing |                      2 |
| IT        |                      1 |
| HR        |                      1 |
+-----------+------------------------+
```

Assignment-6

Questions for Lab Assignment-6

Table: Sales

OrderID	Date	Price	Quantity	CustomerName
1	2005/12/22	160	2	Smith
2	2005/08/10	190	3	Johnson
3	2005/07/13	500	5	Baldwin
4	2005/07/15	420	2	Smith
5	2005/12/22	1000	4	Wood
6	2005/10/02	820	4	Smith
7	2005/11/03	2000	2	Baldwin

Create the table Sales and Write SQL queries for the following:

1. Count how many orders have made a customer with CustomerName of Smith.

2. Find number of unique customers that have ordered from the store.

3. Find out total no. of items ordered by all the customers.

4. Find out average number of items per order.

5. Find out the average Quantity for all orders with Price greater than 200.

6. Find out what was the minimum price paid for any of the orders.

7. Find out the highest Price form the given sales table.

8. List out unique customers name only from the table.

9. List out name of the customers who have given order in the month of December.

10. Find out the total amount of money spent for each of the customers.

11. Select all unique customers who have spent more than 1200 in the store.

12. Select all customers that have ordered more than 5 items in total from all their orders.

13. Select all customers who have spent more than 1000, after 10/01/2005.

14. Select orders in increasing order of order price.

15. Select orders in decreasing order of order price.

Name:- Nabeel Mohammad Rizwan
Roll No:- 20BCS087
Assignment-6
DBMS LAB

```
mysql> create table sales(
    -> OrderID int,
    -> Date date,
    -> Price int,
    -> Quantity int,
    -> CustomerName varchar(30)
    -> );

mysql>  insert into sales values(1,'2005-12-22',160,2,'Smith'),
    -> (2,'2005-08-10',190,3,'Johnson'),
    -> (3,'2005-07-13',500,5,'Baldwin'),
    -> (4,'2005-07-15',420,2,'Smith'),
    -> (5,'2005-12-22',1000,4,'Wood'),
    -> (6,'2005-10-02',820,4,'Smith'),
    -> (7,'2005-11-03',2000,2,'Baldwin');

mysql> select * from sales;
+---------+------------+-------+----------+--------------+
| OrderID | Date       | Price | Quantity | CustomerName |
+---------+------------+-------+----------+--------------+
|       1 | 2005-12-22 |   160 |        2 | Smith        |
|       2 | 2005-08-10 |   190 |        3 | Johnson      |
|       3 | 2005-07-13 |   500 |        5 | Baldwin      |
|       4 | 2005-07-15 |   420 |        2 | Smith        |
|       5 | 2005-12-22 |  1000 |        4 | Wood         |
|       6 | 2005-10-02 |   820 |        4 | Smith        |
|       7 | 2005-11-03 |  2000 |        2 | Baldwin      |
+---------+------------+-------+----------+--------------+
```

Exercise:

1. Count how many orders have made a customer with CustomerName of Smith.

```
mysql> select count(quantity)
    -> from sales
    -> where customername = 'Smith';
+-----------------+
| count(quantity) |
+-----------------+
|               3 |
+-----------------+
```

2. Find number of unique customers that have ordered from the store.

```
mysql> select count(distinct CustomerName)
    -> from sales;
+------------------------------+
| count(distinct CustomerName) |
+------------------------------+
|                            4 |
+------------------------------+
```

3. Find out total no. of items ordered by all the customers.

```
mysql> select sum(quantity)
    -> from sales;
+---------------+
| sum(quantity) |
+---------------+
|            22 |
+---------------+
```

4. Find out average number of items per order.

```
mysql> select avg(Quantity)
    -> from sales;
+---------------+
| avg(Quantity) |
+---------------+
|        3.1429 |
+---------------+
```

5. Find out the average Quantity for all orders with Price greater than 200.

```
mysql> select avg(Quantity)
    -> from sales
    -> where Price > 200;
+---------------+
| avg(Quantity) |
+---------------+
|        3.4000 |
+---------------+
```

6. Find out what was the minimum price paid for any of the orders.

```
mysql> select min(Price)
    -> from sales;
+------------+
| min(Price) |
+------------+
|        160 |
+------------+
```

7. Find out the highest Price form the given sales table.

```
mysql> select max(Price)
    -> from Sales;
+------------+
| max(Price) |
+------------+
|       2000 |
+------------+
```

8. List out unique customers name only from the table.

```
mysql> select distinct CustomerName
    -> from Sales;
+--------------+
| CustomerName |
+--------------+
| Smith        |
| Johnson      |
| Baldwin      |
| Wood         |
+--------------+
```

9. List out name of the customers who have given order in the month of December.

```
mysql> select CustomerName
    -> from Sales
    -> where date like '%-12-%';
+--------------+
| CustomerName |
+--------------+
| Smith        |
| Wood         |
+--------------+
```

10. Find out the total amount of money spent for each of the customers.

```
mysql> select CustomerName, sum(price)
    -> from sales
    -> group by CustomerName;
+--------------+------------+
| CustomerName | sum(price) |
+--------------+------------+
| Smith        |       1400 |
| Johnson      |        190 |
| Baldwin      |       2500 |
| Wood         |       1000 |
+--------------+------------+
```

11. Select all unique customers who have spent more than 1200 in the store.

```
mysql> Select distinct(CustomerName)
    -> from sales
    -> where price > 1200;
+--------------+
| CustomerName |
+--------------+
| Baldwin      |
+--------------+
```

12. Select all customers that have ordered more than 5 items in total from all their orders.

```
mysql> select CustomerName, sum(quantity) as sum
    -> from sales
    -> group by CustomerName
    -> having sum > 5;
+--------------+------+
| CustomerName | sum  |
+--------------+------+
| Smith        |    8 |
| Baldwin      |    7 |
+--------------+------+
```

13. Select all customers who have spent more than 1000, after 10/01/2005.

```
mysql> select CustomerName
    -> from sales
    -> where price > 1000 and date > '2005-01-10';
+--------------+
| CustomerName |
+--------------+
| Baldwin      |
+--------------+
```

14. Select orders in increasing order of order price.

```
mysql> select *
    -> from sales
    -> order by price;
+---------+------------+-------+----------+--------------+
| OrderID | Date       | Price | Quantity | CustomerName |
+---------+------------+-------+----------+--------------+
|       1 | 2005-12-22 |   160 |        2 | Smith        |
|       2 | 2005-08-10 |   190 |        3 | Johnson      |
|       4 | 2005-07-15 |   420 |        2 | Smith        |
|       3 | 2005-07-13 |   500 |        5 | Baldwin      |
|       6 | 2005-10-02 |   820 |        4 | Smith        |
|       5 | 2005-12-22 |  1000 |        4 | Wood         |
|       7 | 2005-11-03 |  2000 |        2 | Baldwin      |
+---------+------------+-------+----------+--------------+
```

15. Select orders in decreasing order of order price.

```
mysql> select *
    -> from sales
    -> order by price desc;
+---------+------------+-------+----------+--------------+
| OrderID | Date       | Price | Quantity | CustomerName |
+---------+------------+-------+----------+--------------+
|       7 | 2005-11-03 |  2000 |        2 | Baldwin      |
|       5 | 2005-12-22 |  1000 |        4 | Wood         |
|       6 | 2005-10-02 |   820 |        4 | Smith        |
|       3 | 2005-07-13 |   500 |        5 | Baldwin      |
|       4 | 2005-07-15 |   420 |        2 | Smith        |
|       2 | 2005-08-10 |   190 |        3 | Johnson      |
|       1 | 2005-12-22 |   160 |        2 | Smith        |
+---------+------------+-------+----------+--------------+
```

Assignment-7

Questions for Lab Assignment-7

Table sales:

OrderId	OrderDate	OrderPrice	OrderQuantity	CustomerName
1	2005-12-22	160	2	Smith
2	2005-08-10	190	2	Johnson
3	2005-07-13	500	5	Baldwin
4	2005-07-15	420	2	Smith
5	2005-12-22	1000	4	Wood
6	2005-10-02	820	4	Smith
7	2005-11-03	2000	2	Baldwin
8	2002-12-22	1000	4	Wood
9	2004-12-29	5000	4	Smith

Table products:

Product_id	OrderId	Manufacture_Date	Raw_Material	Vender_id
AZ145	2	2005-12-23	Steel	1
AZ147	6	2002-08-15	Steel	3
CS435	5	2001-11-04	Steel	1
CS783	1	2004-11-03	Plastic	2
CS784	4	2005-11-28	Plastic	2
FD123	2	2005-10-03	Milk	2
FD267	5	2002-21-03	Bread	4
FD333	9	2001-12-12	Milk	1
FD344	3	2005-11-03	Milk	1
GR233	3	2005-11-30	Pulses	2
GR567	6	2005-09-03	Pulses	2

Table vender_info:

Vender_id	Vender_name
1	Smith
2	Wills
3	Johnson
4	Roger

Table venders:

Raw_Material	Venders	Vender_id
Steel	Smith	1
Plastic	Wills	2
Steel	Johnson	3
Milk	Smith	1
Pulses	Wills	2
Bread	Roger	4
Bread	Wills	2
Milk	Wills	3

Solve the following queries for above tables:

1. Display product information which are ordered in the same year of its manufacturing year.

2. Display product information which are ordered in the same year of its manufacturing year where vender is ‘Smith’.

3. Display total number of orders placed in each year.

4. Display total number of orders placed in each year by vender Wills.

5. Display the name of all those persons who are venders and customers both.

6. Display total number of food items ordered every year.

7. Display total number of food items ordered every year made from bread.

8. Display list of product_id whose vender and customer is different.

9. Display all those customers who are ordering products of milk by smith.

10. Display total number of orders by each vender every year.

11. Display name of those venders whose products are sold more than 2000 Rs.Every year.

Name- Nabeel Mohammad Rizwan
Roll No:- 20BCS087
DBMS LAB Assignment-7

Assignment-7

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;

mysql> select * from sales;
+---------+------------+-------+----------+--------------+
| OrderID | Date       | Price | Quantity | CustomerName |
+---------+------------+-------+----------+--------------+
|       1 | 2005-12-22 |   160 |        2 | Smith        |
|       2 | 2005-08-10 |   190 |        3 | Johnson      |
|       3 | 2005-07-13 |   500 |        5 | Baldwin      |
|       4 | 2005-07-15 |   420 |        2 | Smith        |
|       5 | 2005-12-22 |  1000 |        4 | Wood         |
|       6 | 2005-10-02 |   820 |        4 | Smith        |
|       7 | 2005-11-03 |  2000 |        2 | Baldwin      |
|       8 | 2002-12-22 |  1000 |        4 | Wood         |
|       9 | 2004-12-29 |  5000 |        4 | Smith        |
+---------+------------+-------+----------+--------------+

mysql> select * from products;
+------------+---------+------------------+--------------+-----------+
| product_id | OrderID | manufacture_date | raw_material | vender_id |
+------------+---------+------------------+--------------+-----------+
| AZ145      |       2 | 2005-12-23       | Steel        |         1 |
| AZ147      |       6 | 2005-08-15       | Steel        |         3 |
| CS435      |       5 | 2001-11-04       | Steel        |         1 |
| CS783      |       1 | 2004-11-03       | Plastic      |         2 |
| CS784      |       4 | 2005-11-28       | Plastic      |         2 |
| FD123      |       2 | 2005-10-03       | Milk         |         2 |
| FD333      |       9 | 2001-12-12       | Milk         |         1 |
| FD344      |       3 | 2005-11-03       | Milk         |         1 |
| GR233      |       3 | 2005-11-30       | Pulses       |         2 |
| GR567      |       6 | 2005-09-03       | Pulses       |         2 |
| FD267      |       5 | 2002-12-03       | Bread        |         4 |
+------------+---------+------------------+--------------+-----------+

mysql> select * from vender_info;
+-----------+-------------+
| vender_id | vender_name |
+-----------+-------------+
|         1 | Smith       |
|         2 | Wills       |
|         3 | Johnson     |
|         4 | Roger       |
+-----------+-------------+

mysql> select * from venders;
+--------------+---------+-----------+
| Raw_Material | venders | vender_id |
+--------------+---------+-----------+
| Steel        | Smith   |         1 |
| Plastic      | Wills   |         2 |
| Steel        | Johnson |         3 |
| Milk         | Smith   |         1 |
| Pulses       | Wills   |         2 |
| Bread        | Roger   |         4 |
| Bread        | Wills   |         2 |
| Milk         | Wills   |         3 |
+--------------+---------+-----------+
```

Exercise:

1. Display product information which is ordered in the same year of its manufacturing year.

```
mysql> select p.* from products p, sales s where year(p.Manufacture_date)
= year(s.Date) and p.OrderId = s.OrderId;

+------------+---------+------------------+--------------+-----------+
| product_id | OrderID | manufacture_date | raw_material | vender_id |
+------------+---------+------------------+--------------+-----------+
| AZ145      |       2 | 2005-12-23       | Steel        |         1 |
| AZ147      |       6 | 2005-08-15       | Steel        |         3 |
| CS784      |       4 | 2005-11-28       | Plastic      |         2 |
| FD123      |       2 | 2005-10-03       | Milk         |         2 |
| FD344      |       3 | 2005-11-03       | Milk         |         1 |
| GR233      |       3 | 2005-11-30       | Pulses       |         2 |
| GR567      |       6 | 2005-09-03       | Pulses       |         2 |
+------------+---------+------------------+--------------+-----------+
```

2. Display product information which is ordered in the same year of its manufacturing year where the vender is ‘smith’.

```
mysql> select p.* from products p, sales s where year(p.Manufacture_date)
= year(s.Date) and p.OrderId = s.OrderId and p.Vender_id = (select
Vender_id from vender_info where Vender_name = 'Smith');

+------------+---------+------------------+--------------+-----------+
| product_id | OrderID | manufacture_date | raw_material | vender_id |
+------------+---------+------------------+--------------+-----------+
| AZ145      |       2 | 2005-12-23       | Steel        |         1 |
| FD344      |       3 | 2005-11-03       | Milk         |         1 |
+------------+---------+------------------+--------------+-----------+
```

3. Display the total number of orders placed in each year.

```
mysql> select sum(quantity), year(date)
    -> from sales
    -> group by year(date);

+---------------+------------+
| sum(quantity) | year(date) |
+---------------+------------+
|            22 |       2005 |
|             4 |       2002 |
|             4 |       2004 |
+---------------+------------+
```

4. Display the total number of orders placed in each year by vender Wills.

```
mysql> select count(*),a.date,c.Vender_name
    -> from sales a natural join products
```

```
    ->  b natural join vender_info c where c.Vender_name = 'Wills' group
by (a.date);

+----------+------------+-------------+
| count(*) | date       | Vender_name |
+----------+------------+-------------+
|        1 | 2005-12-22 | Wills       |
|        1 | 2005-08-10 | Wills       |
|        1 | 2005-07-13 | Wills       |
|        1 | 2005-07-15 | Wills       |
|        1 | 2005-10-02 | Wills       |
+----------+------------+-------------+
```

5. Display the name of all those persons who are vendors and customers both.

```
mysql>  select Vender_name from vender_info where Vender_name in (select
CustomerName from sales);

+-------------+
| Vender_name |
+-------------+
| Smith       |
| Johnson     |
+-------------+
```

6. Display the total number of food items ordered every year.

```
mysql> select year(date),sum(Quantity)
    -> from sales
    -> group by year(date);

+------------+---------------+
| year(date) | sum(Quantity) |
+------------+---------------+
|       2005 |            22 |
|       2002 |             4 |
|       2004 |             4 |
+------------+---------------+
```

7. Display the total number of food items ordered every year made from bread.

```
mysql> select year(date),sum(Quantity)
    -> from sales
    ->  where OrderId in (select OrderId from products where Raw_Material
= 'Bread')
    ->  group by (date);



+------------+---------------+
| year(date) | sum(Quantity) |
+------------+---------------+
|       2005 |             4 |
+------------+---------------+
```

8. Display list of product_id whose vendor and customer is different.

```
mysql> select a.Product_id
    -> from products a natural join vender_info b natural join
    ->  sales c where b.Vender_name != c.CustomerName;
+------------+
| Product_id |
+------------+
| CS783      |
| FD123      |
| AZ145      |
| GR233      |
| FD344      |
| CS784      |
| FD267      |
| CS435      |
| GR567      |
| AZ147      |
+------------+
```

9. Display all those customers who are ordering products of milk by smith.

```
mysql> select c.CustomerName
    -> from products
    -> a natural join vender_info b natural join sales c where
b.Vender_name = 'Smith' and a.Raw_Material = 'Milk';

+--------------+
| CustomerName |
+--------------+
| Baldwin      |
| Smith        |
+--------------+
```

10. Display the total number of orders by each vender every year.

```
mysql> select sum(c.Quantity), b.Vender_name, year(c.Date)
    -> from products a natural join vender_info b natural join sales c
    -> group by Vender_name, year(date);

+-----------------+-------------+--------------+
| sum(c.Quantity) | Vender_name | year(c.Date) |
+-----------------+-------------+--------------+
|              16 | Wills       |         2005 |
|              12 | Smith       |         2005 |
|               4 | Roger       |         2005 |
|               4 | Johnson     |         2005 |
|               4 | Smith       |         2004 |
+-----------------+-------------+--------------+
```

11. Display name of those vendors whose products are sold more than 2000 Rs. Every year.

```
mysql> select Vender_name, year(date), sum(Price*Quantity) as TotalAmount
from products a natural join
```

```
vender_info b natural join sales c group by b.Vender_name, year(date)
having sum(c.price*c.quantity) > 2000;
```

Vender_name	year(date)	TotalAmount
Wills	2005	7510
Smith	2005	7070
Roger	2005	4000
Johnson	2005	3280
Smith	2004	20000

Assignment-8

DBMS LAB Assignment No. 8

Department of Computer Engineering
Jamia Millia Islamia

- **Select the right option for each of the question by creating the required tables and inserting the required data only.**
- **Suitable data can be assumed if necessary.**

GATE-2010

1. A relational schema for a train reservation database is given below.

Passenger (pid, pname, age)
Reservation (pid, class, tid)

Table: Passenger

pid	pname	age
0	Sachin	65
1	Rahul	66
2	Sourav	67
3	Anil	69

Table : Reservation

pid	class	tid
0	AC	8200
1	AC	8201
2	SC	8201
5	AC	8203
1	SC	8204
3	AC	8202

What pids are returned by the following SQL query for the above instance of the tables?

```
SELECT pid

FROM Reservation

WHERE class ‘AC’ AND

    EXISTS (SELECT *

       FROM Passenger

       WHERE age > 65 AND

       Passenger. pid = Reservation.pid)
```

(a) 1, 0 (b) 1, 2 (c) 1, 5 (d) 1, 3

GATE-2009
Common Data for Questions 2 and 3

2. Consider the following relational schema:
Suppliers(sid:integer, sname:string, city:string,
street:string) Parts(pid:integer, pname:string, color:string)
Catalog(sid:integer, pid:integer, cost:real)
Consider the following relational query on the above database:

```
SELECT S.sname

    FROM Suppliers S

        WHERE S.sid NOT IN (SELECT C.sid FROM Catalog C

                            WHERE C.pid NOT IN (SELECT P.pid FROM Parts P
WHERE P.color<> 'blue'))
```

Assume that relations corresponding to the above schema are not empty. Which one of the following is the correct interpretation of the above query?
(a) Find the names of all suppliers who have not supplied a non-blue part.
(b) Find the names of all suppliers who have supplied only blue parts.
(c) Find the names of all suppliers who have not supplied only blue parts.
(d) Find the names of all suppliers who have supplied a non-blue part.

GATE-2004

3. The employee information in a company is stored in the relation
Employee (name, sex, salary, deptName)
Consider the following SQL query

```
   Select deptName

         From Employee

         Where sex = ‘M’

    Group by deptName

    Having avg(salary) >

         (select avg (salary) from Employee)
```

It returns the names of the department in which

(a) the average salary of male employees is more than the average salary in the company
(b) the average salary is more than the average salary in the company
(c) the average salary of male employees is more than the average salary of all male employees in the company
(d) the average salary of male employees is more than the average salary of employees in the same department.

GATE-2005

4. The relation book (title, price) contains the titles and prices of different books. Assuming that no two books have the same price, what does the following SQL query list?

```
Select title

From book as B

Where (Select count(*)

   from book as T

   Where T.price > B.price) < 5
```

(a) Titles of the four most expensive books (b) Title of the fifth most inexpensive book
(c) Title of the fifth most expensive book
(d) Titles of the five most expensive books

GATE – 2006

5. Consider the relation enrolled (student, course) in which (student, course) is the primary key, and the relation paid (student, amount) where student is the primary key. Assume no null values and no foreign keys or integrity constraints. Given the following four queries:

Query1: select student from enrolled where student in (select student from paid)
Query2: select student from paid where student in (select student from enrolled)
Query3: select E.student from enrolled E, paid P where E.student = P.student
Query4: select student from paid where exists
(select * from enrolled where enrolled.student = paid.student)

Which one of the following statements is correct?

(a) All queries return identical row sets for any database

(b) Query2 and Query4 return identical row sets for all databases but there exist databases for which Query1 and Query2 return different row sets.

(c) There exist databases for which Query3 returns strictly fewer rows than Query2.

(d) There exist databases for which Query4 will encounter an integrity violation at runtime.

GATE-2006

6. Consider the relation account (customer, balance) where customer is a primary key and there are no null values. We would like to rank customers according to decreasing balance. The customer with the largest balance gets rank 1. Ties are not broke but ranks are skipped: if exactly two customers have the largest balance they each get rank 1 and rank 2 is not assigned.

Query1:
```
select A.customer, count(B.customer)
from account A, account B
where A.balance <=B.balance
group by A.customer
```

Query2:
```
select A.customer, 1+count(B.customer)
from account A, account B
where A.balance < B.balance
group by A.customer
```

Consider these statements about Query1 and Query2.
1. Query1 will produce the same row set as Query2 for some but not all databases.
2. Both Query1 and Query2 are correct implementation of the specification
3. Query1 is a correct implementation of the specification but Query2 is not
4. Neither Query1 nor Query2 is a correct implementation of the specification
5. Assigning rank with a pure relational query takes less time than scanning in decreasing balance order assigning ranks using ODBC.

Which two of the above statements are correct?
(a) 2 and 5
(b) 1 and 3
(c) 1 and 4
(d) 3 and 5

GATE-2011

7. Database table by name Loan_Records is given below.

```
---------------------------------------
BorrowerBank_Manager   Loan_Amount
---------------------------------------
RameshSunderajan10000.00
SureshRamgopal5000.00
MaheshSunderajan7000.00
---------------------------------------
```

What is the output of the following SQL query?

```
SELECT Count(*)

FROM  ( (SELECT Borrower, Bank_Manager

       FROM   Loan_Records) AS S

        NATURAL JOIN (SELECT Bank_Manager,

                             Loan_Amount

                      FROM   Loan_Records) AS T );
```

(a) 3 (b) 5 (c) 9 (d) 6

GATE-2007

8. Consider the table employee(empId, name, department, salary) and the two queries Q1, Q2 below. Assuming that department 5 has more than one employee, and we want to find the employees who get higher salary than anyone in the department 5, which one of the statements is TRUE for any arbitrary employee table?

Q1 : Select e.empId
From employee e
Where not exists
(Select * From employee s where s.department = “5” and
s.salary >=e.salary)
Q2 : Select e.empId
From employee e
Where e.salary > Any
(Select distinct salary From employee s Where s.department = “5”)

(a) Q1 is the correct query
(b) Q2 is the correct query
(c) Both Q1 and Q2 produce the same answer.
(d) Neither Q1 nor Q2 is the correct query.

GATE-2000

9. Given relations r(w, x) and s(y, z), the result of

```
select distinct w, x
from r, s
```

is guaranteed to be same as r, provided
(a) r has no duplicates and s is non-empty
(b) r and s have no duplicates
(c) s has no duplicates and r is non-empty
(d) r and s have the same number of tuples

GATE – 2015

10. Consider the following relations:

Student	
Roll_No	Student_Name
1	Raj
2	Rohit
3	Raj

Performance		
Roll_No	Course	Marks
1	Math	80
1	English	70
2	Math	75
3	English	80
2	Physics	65
3	Math	80

Consider the following SQL query.

```
SELECT S. Student_Name, sum (P.Marks)

FROM Student S, Performance P

WHERE S. Roll_No =P.Roll_No

GROUP BY S.Student_Name
```

The number of rows that will be returned by the SQL query is _____.

GATE – 2015

11. Consider the following relation

```
Cinema (theater, address, capacity)
```

Which of the following options will be needed at the end of the SQL query

```
SELECT P1.address FROM Cinema P1
```

such that it always finds the addresses of theaters of theaters with maximum capacity?

(a) WHERE P1.capacity > = Any (select P2. capacity from Cinema P2)
(b) WHERE P1.capacity > All (select max (P2. capacity) from Cinema P2)
(c) WHERE P1.capacity >Any (select max (P2. capacity) from Cinema P2)
(d) WHERE P1.capacity > = All (select P2. capacity from Cinema P2)

Name:- Nabeel Mohammad Rizwan
Roll No:- 20BCS087
DBMS ASSIGNMENT - 8

1)

d) (1,3)

```
mysql> CREATE DATABASE RailwayReservation;

mysql> USE RailwayReservation;

mysql> CREATE TABLE Passenger(pid INT,pname VARCHAR(20),age
INT);

mysql> CREATE TABLE Reservation(pid INT,class VARCHAR(20),tid
INT);

mysql> INSERT INTO Passenger VALUES(0,"Sachin",65);

mysql> INSERT INTO Passenger VALUES(1,"Rahul",66);

mysql> INSERT INTO Passenger VALUES(2,"Sourav",67);

mysql> INSERT INTO Passenger VALUES(3,"Anil",69);

mysql> INSERT INTO Reservation VALUES(0,"AC",8200);

mysql> INSERT INTO Reservation VALUES(1,"AC",8201);

mysql> INSERT INTO Reservation VALUES(2,"SC",8201);

mysql> INSERT INTO Reservation VALUES(5,"AC",8203);

mysql> INSERT INTO Reservation VALUES(1,"SC",8204);

mysql> INSERT INTO Reservation
VALUES(3,"AC",8202);

mysql> SELECT * FROM Passenger;
+------+--------+------+
| pid  | pname  | age  |
+------+--------+------+
|    0 | Sachin |   65 |
|    1 | Rahul  |   66 |
```

```
|    2 | Sourav |   67 |
|    3 | Anil   |   69 |
+------+--------+------+

mysql> SELECT * FROM Reservation;
+------+-------+------+
| pid  | class | tid  |
+------+-------+------+
|    0 | AC    | 8200 |
|    1 | AC    | 8201 |
|    2 | SC    | 8201 |
|    5 | AC    | 8203 |
|    1 | SC    | 8204 |
|    3 | AC    | 8202 |
+------+-------+------+

mysql> SELECT pid FROM Reservation WHERE class="AC" AND EXISTS
(SELECT * FROM Passenger WHERE age > 65 AND Passenger. pid =
Reservation.pid);
+------+
| pid  |
+------+
|    1 |
|    3 |
+------+
```

2)

Ans)

Find the names of all suppliers who have supplied only blue parts.

```
mysql> CREATE TABLE Suppliers(sid INT,sname VARCHAR(20),city
VARCHAR(20),street VARCHAR(20));

mysql> CREATE TABLE Parts(pid INT,pname VARCHAR(20),color
VARCHAR(20));

mysql> CREATE TABLE Catalog(sid INT,pid INT,cost REAL);

mysql> INSERT INTO Suppliers
VALUES(1,"Amit","Bangalore","Bellandur");

mysql> INSERT INTO Suppliers VALUES(2,"Aadil","Kolkata","New
Town");
```

```
mysql> INSERT INTO Suppliers
VALUES(3,"Faizan","Mumbai","Dadar");

mysql> INSERT INTO Suppliers VALUES(4,"Jatin","Delhi","Lodhi
Colony");

mysql> INSERT INTO Parts VALUES(1,"Sunmica","White");

mysql> INSERT INTO Parts VALUES(2,"Sofa Cover","Blue");

mysql> INSERT INTO Parts VALUES(3,"Bedsheet","Green");

mysql> INSERT INTO Parts VALUES(4,"Curtains","Red");

mysql> INSERT INTO Catalog VALUES(1,3,1200);

mysql> INSERT INTO Catalog VALUES(4,1,500);

mysql> INSERT INTO Catalog VALUES(2,3,500);

mysql> INSERT INTO Catalog VALUES(3,4,900);

mysql> SELECT S.sname FROM Suppliers S WHERE S.sid NOT IN
(SELECT C.sid FROM Catalog C WHERE C.pid NOT IN (SELECT P.pid
FROM Parts P WHERE P.color<> 'blue'));
+--------+
| sname  |
+--------+
| Amit   |
| Aadil  |
| Faizan |
| Jatin  |
+--------+

mysql> SELECT * FROM Suppliers;
+------+--------+-----------+--------------+
| sid  | sname  | city      | street       |
+------+--------+-----------+--------------+
|    1 | Amit   | Bangalore | Bellandur    |
|    2 | Aadil  | Kolkata   | New Town     |
|    3 | Faizan | Mumbai    | Dadar        |
|    4 | Jatin  | Delhi     | Lodhi Colony |
+------+--------+-----------+--------------+

mysql> SELECT * FROM Parts;
+------+------------+-------+
| pid  | pname      | color |
```

```
+------+------------+-------+
|    1 | Sunmica    | White |
|    2 | Sofa Cover | Blue  |
|    3 | Bedsheet   | Green |
|    4 | Curtains   | Red   |
+------+------------+-------+

mysql> SELECT * FROM Catalog;
+------+------+------+
| sid  | pid  | cost |
+------+------+------+
|    1 |    3 | 1200 |
|    4 |    1 |  500 |
|    2 |    3 |  500 |
|    3 |    4 |  900 |
+------+------+------+
```

3)

Ans)

The average salary of male employees is more than the average salary in the company.

```
mysql> CREATE TABLE Employee(name VARCHAR(20),sex CHAR,salary
INT,deptName VARCHAR(20),PRIMARY KEY(name));

mysql> INSERT INTO Employee
VALUES("Sudarshan","M",15000,"Mathematics");

mysql> INSERT INTO Employee VALUES("Anzal","M",17000,"Computer
Science");

mysql> INSERT INTO Employee VALUES("Kanika","F",12000,"Arts");

mysql> INSERT INTO Employee
VALUES("Aftab","M",13000,"Electrical");

mysql> Select deptName From Employee Where sex = 'M' Group by
deptName Having avg(salary) > (select avg (salary) from
Employee);
+------------------+
| deptName         |
+------------------+
| Computer Science |
| Mathematics      |
+------------------+
```

```
mysql> SELECT AVG(salary) FROM Employee WHERE sex="M";
+-------------+
| AVG(salary) |
+-------------+
|  15000.0000 |
+-------------+

mysql> SELECT AVG(salary) FROM Employee;
+-------------+
| AVG(salary) |
+-------------+
|  14250.0000 |
+-------------+
```

4)
Ans)

```
(d) Title of 5 most expensive books. mysql> SELECT title FROM book
as B WHERE (SELECT COUNT(*) FROM book
as T WHERE T.price > B.price) < 5;
+-------+
| title |
+-------+
| C     |
| D     |
| E     |
| F     |
| G     |
+-------+
```

5)

Ans)

```
(a) All queries return identical row sets for any database

mysql> CREATE TABLE enrolled(student VARCHAR(20),course
VARCHAR(20));

CREATE TABLE paid(student VARCHAR(20),amount INT,PRIMARY
KEY(student));
```

```
mysql> INSERT INTO enrolled VALUES("xyz","CSE");

mysql> INSERT INTO enrolled VALUES("abc","ECE");

mysql> INSERT INTO enrolled VALUES("pqr","CSE");

mysql> INSERT INTO paid VALUES("abc",20000);

mysql> INSERT INTO paid VALUES("xyz",10000);

mysql> INSERT INTO paid VALUES("rst",10000);

mysql> SELECT * FROM paid;
+---------+--------+
| student | amount |
+---------+--------+
| abc     |  20000 |
| rst     |  10000 |
| xyz     |  10000 |
+---------+--------+

mysql> SELECT * FROM enrolled;
+---------+--------+
| student | course |
+---------+--------+
| xyz     | CSE    |
| abc     | ECE    |
| pqr     | CSE    |
+---------+--------+
```

Query1. mysql> SELECT student FROM enrolled WHERE student in (SELECT student FROM paid);

```
+---------+
| student |
+---------+
| xyz     |
| abc     |
+---------+
```

Query2. mysql> SELECT student FROM paid WHERE student in (SELECT student FROM enrolled);

```
+---------+
| student |
+---------+
| xyz     |
| abc     |
+---------+
```

Query3. mysql> SELECT E.student FROM enrolled E,paid P WHERE E.student = P.student;

```
+---------+
| student |
+---------+
| xyz     |
| abc     |
+---------+
```

Query4. mysql> SELECT student FROM paid WHERE EXISTS (SELECT * FROM enrolled WHERE enrolled.student = paid.student);

```
+---------+
| student |
+---------+
| xyz     |
| abc     |
+---------+
```

6)
Ans)

(d) 3 and 5
Query 1 is a correct implementation of the specification but Query 2 is not.
Assigning rank with a pure relational query takes less time than scanning in decreasing balance order assigning ranks using ODBC.

```
mysql> CREATE TABLE account(customer VARCHAR(20),balance INT,PRIMARY KEY(customer));

mysql> INSERT INTO account VALUES("abc",4000);

mysql> INSERT INTO account VALUES("def",3000);

mysql> INSERT INTO account VALUES("ghi",2000);

mysql> INSERT INTO account VALUES("xyz",1000);
```

Query1: mysql> SELECT A.customer,count(B.customer) from account A,account B WHERE A.balance<=B.balance GROUP BY A.customer;

```
+----------+-------------------+
| customer | count(B.customer) |
+----------+-------------------+
| xyz      |                 4 |
| ghi      |                 3 |
| def      |                 2 |
| abc      |                 1 |
+----------+-------------------+
```

Query2: mysql> SELECT A.customer,1+count(B.customer) from account A,account B WHERE A.balance<B.balance GROUP BY A.customer;

```
+----------+---------------------+
| customer | 1+count(B.customer) |
+----------+---------------------+
| xyz      |                   4 |
| ghi      |                   3 |
| def      |                   2 |
+----------+---------------------+
```

7)
Ans)

```
mysql> CREATE TABLE Loan_Records(Borrower
VARCHAR(30),Bank_Manager VARCHAR(30),Loan_Amount INT);

mysql> INSERT INTO Loan_Records
VALUES("Ramesh","Sunderajan",10000);

mysql> INSERT INTO Loan_Records
VALUES("Mahesh","Sunderajan",7000);

mysql> INSERT INTO Loan_Records
VALUES("Suresh","Ramgopal",5000);

mysql> SELECT Count(*) FROM ( (SELECT Borrower, Bank_Manager
FROM Loan_Records) AS S NATURAL JOIN (SELECT Bank_Manager,
Loan_Amount FROM Loan_Records) AS T );
+----------+
| Count(*) |
+----------+
|        5 |
```

```
+----------+
```

8)
Ans)

-> Both Q1 and Q2 produce the same answer

```
mysql> CREATE TABLE employees(empId INT,name
VARCHAR(20),department INT,salary INT);

mysql> INSERT INTO employees VALUES(1001,"Sudarshan",1,12000);

mysql> INSERT INTO employees VALUES(1002,"Anzal",3,15000);

mysql> INSERT INTO employees VALUES(1003,"Naveen",5,16000);

mysql> INSERT INTO employees VALUES(1004,"Ijlal",4,19000);

mysql> SELECT * FROM employees;
+-------+-----------+------------+--------+
| empId | name      | department | salary |
+-------+-----------+------------+--------+
|  1001 | Sudarshan |          1 |  12000 |
|  1002 | Anzal     |          3 |  15000 |
|  1003 | Naveen    |          5 |  16000 |
|  1004 | Ijlal     |          4 |  19000 |
+-------+-----------+------------+--------+
```

Query1. mysql> Select e.empId From employees e Where not exists (Select * From employees s where s.department = 5 and s.salary >=e.salary);

```
+-------+
| empId |
+-------+
|  1004 |
+-------+
```

Query2. mysql> Select e.empId From employees e Where e.salary > Any (Select distinct salary From employees s Where s.department = 5);

```
+-------+
| empId |
+-------+
|  1004 |
+-------+
```

9)

Ans-> r and s have the same number of tuples.

10)

```
mysql> CREATE TABLE Performance(Roll_No INT,Course
VARCHAR(20),Marks INT);

mysql> INSERT INTO Performance VALUES(1,"Math",80);

mysql> INSERT INTO Performance VALUES(1,"English",70);

mysql> INSERT INTO Performance VALUES(2,"Math",75);

mysql> INSERT INTO Performance
VALUES(3,"English",80);

mysql> INSERT INTO Performance VALUES(2,"Physics",65);

mysql> INSERT INTO Performance VALUES(3,"Math",80);

mysql> SELECT * FROM Performance;
+---------+---------+-------+
| Roll_No | Course  | Marks |
+---------+---------+-------+
|       1 | Math    |    80 |
|       1 | English |    70 |
|       2 | Math    |    75 |
|       3 | English |    80 |
|       2 | Physics |    65 |
|       3 | Math    |    80 |
+---------+---------+-------+

mysql> SELECT S.Student_Name,SUM(P.marks) FROM Student
S,Performance P WHERE S.Roll_No = P.Roll_No GROUP BY
S.Student_Name;
+--------------+--------------+
| Student_Name | SUM(P.marks) |
+--------------+--------------+
| Raj          |          310 |
| Rohit        |          140 |
+--------------+--------------+
```

11)

Ans)

```
mysql> CREATE TABLE Cinema(theater VARCHAR(20),address
VARCHAR(20),Capacity INT);

mysql> INSERT INTO Cinema VALUES("PVR","Gurgaon",100);

mysql> INSERT INTO Cinema VALUES("XYZ","Delhi",70);

mysql> INSERT INTO Cinema VALUES("ABC","Mumbai",90);

mysql> INSERT INTO Cinema VALUES("DEF","Kolkata",60);

(D) mysql> SELECT P1.address FROM Cinema P1 WHERE
P1.capacity>=All(SELECT P2.capacity FROM Cinema P2);
+---------+
| address |
+---------+
| Gurgaon |
+---------+
```

Lab Assignment-9

Consider the following tables:

Table Student:

snum	sname	major	level	age
101	Jhon	CS	SR	19
102	Smith	CS	JR	20
103	Jacob	ECE	SR	20
104	Tom	CS	JR	20
105	Sid	CS	JR	20
106	Harry	History	SR	21
107	Hellen	CS	JR	21
108	Bob	English	SR	22
109	Andy	ECE	JR	21
110	Charles	History	SR	23

Table Class:

cname	meets_at	room	fid
CSC342	Morning	R128	201
CSC343	Noon	R128	203
CSC345	Night	R154	204
ECE300	Morning	R111	202
ECE301	Noon	R111	203
ENG366	Morning	R154	203
ENG367	Evening	R111	205
HIS320	Evening	R128	205

Table Enrolled:

snum	cname
101	CSC342
101	CSC343
101	CSC345
101	ECE300
101	ENG366
102	CSC343
102	CSC345
102	ECE301
103	ECE300
103	ECE301
104	CSC342
104	ECE301
105	CSC345
105	ECE300
106	ENG366
106	HIS320
107	CSC342
107	ENG366
108	ENG367
108	HIS320
109	ECE300
109	ECE301
110	ENG366
110	HIS320

Table Faculty:

fid	fname	deptid
201	S. Jackson	301
202	M. Shanks	302
203	I. Teach	302
204	A. Zobrah	303
205	M. Jensen	303

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class.

Write the SQL statements required to create these relations, including appropriate versions of all primary and foreign key integrity constraints.

Read all questions first and insert values accordingly.

No duplicates should be printed in any of the

answers. Write the following queries in SQL:

1. Find the names of all Juniors(Level = JR) who are enrolled in a class taught by I. Teach.

2. Find the age of the oldest student who is either a History major or enrolled in a course taught by I. Teach.

3. Find the names of all classes that either meet in room R128 or have five or more students enrolled.

4. Find the names of all students who are enrolled in two class that meet at the same time.

5. Find the names of faculty members who teach in every room in which some class is taught.

6. Find the names of faculty members for whom the combined enrollment of the course that they teach is less than five.

7. For each level, print the level and the average age of students for that level.

8. For all levels except JR, print the level and the average age of students for that level.

9. For each faculty member that has taught class only in room R128 print the faculty member's name and the total number of classes he or she has taught.

10. Find the names of students enrolled in the maximum number of classes.

Name :- Nabeel Mohammad Rizwan
Roll No :- 20BCS087
DBMS Assignment :- 9

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;
```

```
mysql> select * from student;
+------+---------+---------+-------+------+
| snum | sname   | major   | level | age  |
+------+---------+---------+-------+------+
|  101 | Jhon    | CS      | SR    |   19 |
|  102 | Smith   | CS      | JR    |   20 |
|  103 | Jacob   | ECE     | SR    |   20 |
|  104 | Tom     | CS      | JR    |   20 |
|  105 | Sid     | CS      | JR    |   20 |
|  106 | Harry   | History | SR    |   21 |
|  107 | Hellen  | CS      | JR    |   21 |
|  108 | Bob     | English | SR    |   22 |
|  109 | Andy    | ECE     | JR    |   21 |
|  110 | Charles | History | SR    |   23 |
+------+---------+---------+-------+------+
```

```
mysql> select * from class;
+--------+----------+------+------+
| cname  | meets_at | room | fid  |
+--------+----------+------+------+
| CSC342 | Morning  | R128 |  201 |
| CSC343 | Noon     | R128 |  203 |
| CSC345 | Night    | R154 |  204 |
| ECE300 | Morning  | R111 |  202 |
| ECE301 | Noon     | R111 |  203 |
| ENG366 | Morning  | R154 |  203 |
| ENG367 | Evening  | R111 |  205 |
| HIS320 | Evening  | R128 |  205 |
+--------+----------+------+------+
```

```
mysql> select * from enrolled;
+------+--------+
| snum | cname  |
+------+--------+
|  101 | CSC342 |
|  101 | CSC343 |
|  101 | CSC345 |
|  101 | ECE300 |
|  101 | ENG366 |
|  102 | CSC343 |
|  102 | CSC345 |
|  102 | ECE301 |
|  103 | ECE300 |
|  103 | ECE301 |
|  104 | CSC342 |
|  104 | ECE301 |
|  105 | CSC345 |
|  105 | ECE300 |
|  106 | ENG366 |
|  106 | HIS320 |
|  107 | CSC342 |
|  107 | ENG366 |
|  108 | ENG367 |
|  108 | HIS320 |
|  109 | ECE300 |
|  109 | ECE301 |
|  110 | ENG366 |
|  110 | HIS320 |
+------+--------+
```

```
mysql> select * from faculty;
+------+------------+--------+
| fid  | fname      | deptid |
+------+------------+--------+
|  201 | S. Jackson |    301 |
|  202 | M. Shanks  |    302 |
|  203 | I. Teach   |    302 |
|  204 | A. Zobrah  |    303 |
|  205 | M. Jensen  |    303 |
+------+------------+--------+
```

Exercise:

1) Find the names of all Juniours(Level = JR) who are enrolled in a class taught by I. Teach.

```
mysql> select distinct a.sname from Student a natural join Class b natural join Enrolled c
natural join Faculty d where a.level = 'JR' and d.fname = 'I. Teach';
+--------+
| sname  |
+--------+
| Smith  |
| Tom    |
| Hellen |
| Andy   |
+--------+
```

2) Find the age of the oldest student who is either a History major or enrolled in a course taught by I. Teach.

```
mysql>  select a.sname, a.age from Student a natural join Class b natural join
Enrolled c natural join Faculty d where d.fname = 'I. Teach' and a.major =
'History' and a.age = (select max(age) from Student a natural join Class b natural
join Enrolled c natural join Faculty d where d.fname = 'I. Teach' and a.major =
'History');
+---------+------+
| sname   | age  |
+---------+------+
| Charles |   23 |
+---------+------+
```

3) Find the names of all classes that either meet in room R128 or have five or more students enrolled.

```
mysql> select b.cname,count(*) from Student a natural join Class b natural join
Enrolled c natural join Faculty d where b.room = 'R128' or b.cname in (select cname
from Enrolled group by cname having count(*)>= 5) group by b.cname;
+--------+----------+
| cname  | count(*) |
+--------+----------+
| CSC342 |        3 |
| CSC343 |        2 |
| HIS320 |        3 |
+--------+----------+
```

4) Find the names of all students who are enrolled in two-class that meet at the same time.

```
mysql>  select a.sname from Student a natural join Class b natural join Enrolled c
group by a.sname, b.meets_at having count(*) >= 2;
+--------+
| sname  |
+--------+
| Jhon   |
| Smith  |
| Hellen |
| Bob    |
+--------+
```

5) Find the names of faculty members who teach in every room in which some class is taught.

```
mysql>  select * from Faculty where fid in (select fid from Class group by fid
having count(*) = (select count(distinct room) from Class));
+------+----------+--------+
| fid  | fname    | deptid |
+------+----------+--------+
|  203 | I. Teach |    302 |
+------+----------+--------+
```

6) Find the names of faculty members for whom the combined enrollment of the course that they teach is less than five.

```
mysql>  select * from Faculty where fid in (select fid from Class where cname in
(select cname from Enrolled group by cname having count(*)<5));
+------+------------+--------+
| fid  | fname      | deptid |
+------+------------+--------+
|  201 | S. Jackson |    301 |
|  202 | M. Shanks  |    302 |
|  203 | I. Teach   |    302 |
|  204 | A. Zobrah  |    303 |
|  205 | M. Jensen  |    303 |
+------+------------+--------+
```

7) For each level, print the level and the average age of students for that level.

```
mysql>  select level,avg(age) from Student group by level;
+-------+----------+
| level | avg(age) |
+-------+----------+
| SR    |  21.0000 |
| JR    |  20.4000 |
+-------+----------+
```

8) For all levels except JR, print the level and the average age of students for that level.

```
mysql>  select level,avg(age) from Student where level != 'JR' group by level;
+-------+----------+
| level | avg(age) |
+-------+----------+
| SR    |  21.0000 |
+-------+----------+
```

9) For each faculty member that has taught class only in room R128 print the faculty member's name and the total number of classes he or she has taught.

```
mysql>  select f.fname,count(*) from Faculty f natural join Class c natural join
Enrolled e where c.room = 'R128' group by fname;
+------------+----------+
| fname      | count(*) |
+------------+----------+
| S. Jackson |        3 |
| I. Teach   |        2 |
| M. Jensen  |        3 |
+------------+----------+
```

10) Find the names of students enrolled in the maximum number of classes.

```
select * from Enrolled e natural join Student s group by e.snum order by count(*)
desc limit 1;
 +------+--------+---------+-------+-------+------+
 | snum | cname  | sname   | major | level | age  |
 +------+--------+---------+-------+-------+------+
 |  101 | CSC342 | Charlie | CS    | SR    |   19 |
 +------+--------+---------+-------+-------+------+
```

Questions

Assignment-10

1. The following relations keep track of airline flight information:

   Flights(*flno:* **integer**, *from:* **string**, *to:* **string**, *distance:* **integer**, *departs:* **time**,
   arrives: **time**, *price:* **integer**)
   Aircraft(*aid:* **integer**, *aname:* **string**, *cruisingrange:* **integer**)
   Certified(eid: **integer**, aid: **integer**)
   Employees(*eid:* **integer**, *ename:* **string**, *salary:* **integer**)

   Note that the Employees relation describes pilots and other kinds of employees aswell; every pilot is certified for some aircraft, and only pilots are certified to fly.Write each of the following queries in SQL.

   a. Find the names of aircraft such that all pilots certified to operate them earn more than $80,000.
   b. For each pilot who is certified for more than three aircraft, find the *eid* and the maximum *cruisingrange* of the aircraft for which she or he is certified.
   c. Find the names of pilots whose *salary* is less than the price of the cheapest route from Los Angeles to Honolulu.
   d. For all aircraft with *cruisingrange* over 1000 miles, find the name of the aircraft and the average salary of all pilots certified for this aircraft.
   e. Find the names of pilots certified for some Boeing aircraft.
   f. Find the *aids* of all aircraft that can be used on routes from Los Angeles to Chicago.
   g. Identify the routes that can be piloted by every pilot who makes more than $100,000.
   h. Print the *enames* of pilots who can operate planes with *cruisingrange* greater than 3000 miles but are not certified on any Boeing aircraft.
   i. A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.
   j. Compute the difference between the average salary of a pilot and the average salary of all employees (including pilots).
   k. Print the name and salary of every nonpilot whose salary is more than the average salary for pilots.

---

   l. Print the names of employees who are certified only on aircrafts with cruising range longer than 1000 miles.
   m. Print the names of employees who are certified only on aircrafts with cruising range longer than 1000 miles, but on at least two such aircrafts.
   n. Print the names of employees who are certified only on aircrafts with cruising range longer than 1000 miles and who are certified on some Boeing aircraft.

Name:- Nabeel Mohammad Rizwan
Assignment:- 10
DBMS Lab

```
mysql> use 20BCS087_NABEEL_MOHD_RIZWAN;

mysql> CREATE TABLE Flights(
FLNO VARCHAR(10),
Origin VARCHAR(30),
Destination VARCHAR(30),
DISTANCE INT,
DEPARTS TIME,
ARRIVES TIME,
PRICE INT);

mysql> INSERT INTO Flights values("AI-101","Delhi","New
York",7302,"1:45:00","6:50:00",30000);

mysql> INSERT INTO Flights values("UA-121","Los
Angeles","Honolulu",2558,"2:45:00","8:00:00",15000);

mysql> INSERT INTO Flights values("DL-541","Los
Angeles","Chicago",1745,"3:45:00","7:55:00",25000);

mysql> INSERT INTO Flights values("UA-925","Madison","New
York",809,"1:45:00","3:55:00",14000);

mysql> INSERT INTO Flights values("AI-
121","Delhi","Frankfurt",3800,"1:25:00","5:50:00",35000);

mysql> INSERT INTO Flights values("AI-
20","Delhi","Kolkata",869,"2:00:00","4:10:00",4650);

mysql> INSERT INTO Flights values("UK-
21","Delhi","Paris",4084,"2:30:00","8:55:00",14650);
```

```
mysql> SELECT * FROM Flights;
+--------+-------------+-------------+----------+----------+----------+-------+
| FLNO   | Origin      | Destination | DISTANCE | DEPARTS  | ARRIVES  | PRICE |
+--------+-------------+-------------+----------+----------+----------+-------+
| AI-101 | Delhi       | New York    |     7302 | 01:45:00 | 06:50:00 | 30000 |
| UA-121 | Los Angeles | Honolulu    |     2558 | 02:45:00 | 08:00:00 | 15000 |
| DL-541 | Los Angeles | Chicago     |     1745 | 03:45:00 | 07:55:00 | 25000 |
| UA-925 | Madison     | New York    |      809 | 01:45:00 | 03:55:00 | 14000 |
| AI-121 | Delhi       | Frankfurt   |     3800 | 01:25:00 | 05:50:00 | 35000 |
| AI-20  | Delhi       | Kolkata     |      869 | 02:00:00 | 04:10:00 |  4650 |
| UK-21  | Delhi       | Paris       |     4084 | 02:30:00 | 08:55:00 | 14650 |
+--------+-------------+-------------+----------+----------+----------+-------+
```

```
mysql> CREATE TABLE Aircraft(AID INT,AName VARCHAR(20),CruisingRange
INT);

mysql> INSERT INTO Aircraft VALUES(112,"Boeing787-8",9000);

mysql> INSERT INTO Aircraft VALUES(151,"Boeing777-200LR",9500);

mysql> INSERT INTO Aircraft VALUES(135,"Airbus A330-300",8300);
mysql> INSERT INTO Aircraft VALUES(135,"Airbus A320NEO",4500);

mysql> INSERT INTO Aircraft VALUES(189,"Boeing787-8",9000);

mysql> INSERT INTO Aircraft VALUES(191,"Boeing777-300ER",9300);

mysql> INSERT INTO Aircraft VALUES(131,"Boeing787-9",9360);

mysql> SELECT * FROM Aircraft;
+------+-----------------+---------------+
| AID  | AName           | CruisingRange |
+------+-----------------+---------------+
|  112 | Boeing787-8     |          9000 |
|  151 | Boeing777-200LR |          9500 |
|  135 | Airbus A330-300 |          8300 |
|  144 | Airbus A320NEO  |          4500 |
|  189 | Boeing787-8     |          9000 |
|  191 | Boeing777-300ER |          9300 |
|  131 | Boeing787-9     |          9360 |
+------+-----------------+---------------+

mysql> CREATE TABLE Certified(EID INT,AID INT);

mysql> INSERT INTO Certified VALUES(591,112);

mysql> INSERT INTO Certified VALUES(601,112);

mysql> INSERT INTO Certified VALUES(621,151);

mysql> INSERT INTO Certified VALUES(641,135);

mysql> INSERT INTO Certified VALUES(661,144);

mysql> INSERT INTO Certified VALUES(681,144);

mysql> INSERT INTO Certified VALUES(701,144);

mysql> INSERT INTO Certified VALUES(721,144);
```

```
mysql> SELECT * FROM Certified;
+------+------+
| EID  | AID  |
+------+------+
|  591 |  112 |
|  601 |  112 |
|  621 |  151 |
|  641 |  135 |
|  661 |  144 |
|  681 |  144 |
|  701 |  144 |
|  721 |  144 |
+------+------+


mysql> CREATE TABLE Employees(EID INT,Ename VARCHAR(20),Salary INT);

mysql> INSERT INTO Employees VALUES(591,"Devi Sharan",12000);

mysql> INSERT INTO Employees VALUES(601,"Aditya",12900);

mysql> INSERT INTO Employees VALUES(621,"Deepak",13900);

mysql> INSERT INTO Employees VALUES(641,"Vasant",15000);

mysql> INSERT INTO Employees VALUES(661,"Abhishek",15000);

mysql> INSERT INTO Employees VALUES(681,"Devendra",16000);

mysql> INSERT INTO Employees VALUES(701,"Sudarshan",16000);

mysql> INSERT INTO Employees VALUES(721,"Sharad",16000);

mysql> SELECT * FROM Employees;
+------+-------------+--------+
| EID  | Ename       | Salary |
+------+-------------+--------+
|  591 | Devi Sharan |  12000 |
|  601 | Aditya      |  12900 |
|  621 | Deepak      |  13900 |
|  641 | Vasant      |  15000 |
|  661 | Abhishek    |  15000 |
|  681 | Devendra    |  16000 |
|  701 | Sudarshan   |  16000 |
|  721 | Sharad      |  16000 |
+------+-------------+--------+
```

Exercise:

Ques1: Find the names of aircraft such that all pilots certified to operate them earn more than $80000.

Ans:

```
mysql> SELECT DISTINCT a.AName FROM Aircraft a WHERE a.Aid IN(SELECT
C.aid FROM Certified C,Employees E WHERE C.EID = E.EID AND NOT
EXISTS (SELECT * FROM Employees E1 WHERE E1.EID = E.EID AND
Salary<80000));
+-----------------+
| AName           |
+-----------------+
| Boeing787-8     |
| Boeing777-200LR |
| Airbus A330-300 |
| Airbus A320NEO  |
+-----------------+
```

Ques2: For each pilot who is certified for more than 3 aircrafts, find EID and Maximum Cruising Range of the aircraft for which he or she is certified.

Ans:

```
mysql> SELECT C.EID,MAX(A.CruisingRange) FROM Certified C,Aircraft A
WHERE C.AID = A.AID GROUP BY C.EID;
+------+----------------------+
| EID  | MAX(A.CruisingRange) |
+------+----------------------+
|  591 |                 9000 |
|  601 |                 9000 |
|  621 |                 9500 |
|  641 |                 8300 |
|  661 |                 4500 |
|  681 |                 4500 |
|  701 |                 4500 |
|  721 |                 4500 |
+------+----------------------+
```

Ques3: Find the names of pilot whose salary is less than the price of the cheapest route from Los Angeles to Honolulu.

Ans:

```
mysql> SELECT DISTINCT E.Ename FROM Employees E WHERE
E.Salary<(SELECT MIN(F.Price) FROM Flights F WHERE Origin="Los
Angeles" AND Destination="Honolulu");
```

Ques4: For all aircraft with cruising range over 1000 miles, find name of the aircraft and the average salary of all the pilots certified for this aircraft.

Ans:

```
mysql> SELECT A.AName,AVG(E.Salary) FROM Aircraft A,Employees E
Group by A.AName;
+-----------------+---------------+
| AName           | AVG(E.Salary) |
+-----------------+---------------+
| Boeing787-9     |    55737.5000 |
| Boeing777-300ER |    55737.5000 |
| Boeing787-8     |    55737.5000 |
| Airbus A320NEO  |    55737.5000 |
| Airbus A330-300 |    55737.5000 |
| Boeing777-200LR |    55737.5000 |
+-----------------+---------------+
```

Ques5: Find the names of pilots certified for some Boeing aircraft.

Ans:

```
mysql> SELECT DISTINCT E.EName FROM Employees E,Certified C,Aircraft
A WHERE E.EID = C.EID AND C.AID = A.AID AND A.AName = "Boeing777-
200LR";
+--------+
| EName  |
+--------+
| Deepak |
+--------+

mysql> SELECT DISTINCT E.EName FROM Employees E,Certified C,Aircraft
A WHERE E.EID = C.EID AND C.AID = A.AID AND A.AName = "Boeing787-8";
+-------------+
| EName       |
+-------------+
| Devi Sharan |
| Aditya      |
+-------------+
```

Ques6: Find the AID’S of all aircrafts that can be used on routes from Los Angeles to Chicago.

Ans:

```
mysql> SELECT AName FROM Aircraft;
+-----------------+
| AName           |
+-----------------+
| Boeing787-8     |
| Boeing777-200LR |
| Airbus A330-300 |
| Airbus A320NEO  |
| Boeing787-8     |
| Boeing777-300ER |
| Boeing787-9     |
+-----------------+
```

Ques7: Identify the routes that can be piloted by every pilot who makes more than $100,000.

Ans:
```
SELECT DISTINCT F.Origin, F.Destination FROM Flights F WHERE NOT
EXISTS (SELECT * FROM Employees E WHERE E.Salary>100000 AND NOT
EXISTS (SELECT * FROM Aircraft A, Certified C WHERE A.CruisingRange
> F.Distance AND E.EID = C.EID AND A.AID = C.AID))
```

Ques8: Find the ENames of pilots who can operate planes with CruisingRange greater than 3000 miles but are not certified on any Boeing Aircraft.

Ans:
```
mysql> SELECT DISTINCT E.EName FROM Employees E,Certified C,Aircraft
A WHERE E.EID = C.EID AND C.AID = A.AID AND CruisingRange>3000;
+-------------+
| EName       |
+-------------+
| Devi Sharan |
| Aditya      |
| Deepak      |
| Vasant      |
| Abhishek    |
| Devendra    |
| Sudarshan   |
| Sharad      |
+-------------+
```

Ques9: A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.

Ans:
```
SELECT F.DEPARTS FROM Flights F WHERE F.FLNO IN ((SELECT F0.FLNO
FROM Flights F0 WHERE F0.Origin ="Madison" AND F0.Destination ="New
York" AND F0.Arrives < '18:00')
UNION
(SELECT F0.FLNO FROM Flights F0,Flights F1 WHERE F0.Origin="Madison"
AND F0.Destination <> 'New York' AND F1.Departs>F0.Arrives AND
F1.Arrives < '18:00')
UNION
(SELECT F0.FLNO FROM Flights F0, Flights F1, Flights F2 WHERE
F0.Origin="Madison"
AND F0.Destination=F1.Origin
AND F1.Destination=F2.Origin
AND F2.Destination='New York'
AND F0.Destination<>'New York'
AND F1.Destination<>'New York'
AND F1.Departs>F0.Arrives
AND F2.Departs>F1.Arrives
AND F1.Arrives<'18:00'))
```

Ques10: Compute the difference between the average salary of a pilot and the average salary of all employees (including pilots).

Ans:
SELECT Temp1.Avg – Temp2.Avg FROM (SELECT AVG(E.Salary) AS Avg FROM Employees E WHERE E.EID IN (SELECT DISTINCT C.EID FROM Certified C)) AS Temp1, (SELECT AVG(E1.Salary) AS Avg FROM Employees E1)AS Temp2

Ques11: Print the name and salary of every nonpilot whose salary is more than the average salary for pilots.

Ans:
SELECT E.EName, E.Salary FROM Employees E WHERE E.EID NOT IN (SELECT DISTINCT C.EID FROM Certified C) AND E.Salary > (SELECT AVG (E1.Salary) FROM Employees E1 WHERE E1 WHERE E1.EID IN (SELECT DISTINCT C1.EID FROM Certified C1))

Ques12: Print the names of employees who are certified only on aircrafts with cruising range longer than 1000 miles.

Ans:
Select E.EName from Employees E,Certified C,Aircraft A where C.AID=A.AID AND E.EID=C.EID Group By E.EID,E.EName HAVING Every (A.CruisingRange>1000);

Ques13: Print the names of employees who are certified only on aircrafts with cruising range longer than 1000 miles, but on at least two suc aircrafts.

Ans: Select E.EName from Employees E,Certified C,Aircraft A where C.AID=A.AID AND E.EID=C.EID Group By E.EID,E.EName HAVING Every (A.CruisingRange>1000) AND Count(*)>1;

Ques14: Print the names of employees who are certified only on aircrafts with cruising range longer than 1000 miles and who are certified on some Boeing aircraft.

Ans: Select E.EName from Employees E,Certified C,Aircraft A where C.AID=A.AID AND E.EID=C.EID Group By E.EID,E.EName HAVING Every (A.CruisingRange>1000) AND ANY (A.AName='Boeing');

2. Stored Procedures and Functions

1) Write a SQL function and stored procedure for average of three numbers.
Function:

```
create function 19MTavg3no(a int,b int,c int) returns int
begin
        declare sum,avg int;
        set sum = a + b + c;
        set avg = sum/3;
        return avg;
end]
```

```
mysql> create function 19MTavg3no(a int,b int,c int) returns int
    -> begin
    -> declare sum,avg int;
    -> set sum = a + b + c;
    -> set avg = sum/3;
    -> return avg;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> select 19MTavg3no(4,5,6)]
+--------------------+
| 19MTavg3no(4,5,6) |
+--------------------+
|                  5 |
+--------------------+
1 row in set (0.01 sec)
```

Stored Procedure:

```
create procedure 19MTavg3no(In a int,In b int,In c int,Out t int)
begin
        declare sum int;
        set sum = a + b + c;
        set t = sum/3;
end]
```

```
mysql> create procedure 19MTavg3no(In a int,In b int,In c int,Out t int)
    -> begin
    -> declare sum int;
    -> set sum = a + b + c;
    -> set t = sum/3;
    -> end]
Query OK, 0 rows affected (0.01 sec)

mysql> call 19MTavg3no(4,5,6,@avg)]
Query OK, 0 rows affected (0.00 sec)

mysql> select @avg]
+-------+
| @avg |
+-------+
|     5 |
+-------+
1 row in set (0.00 sec)
```

2) Write a SQL function and stored procedure to calculate factorial.

Function:

```
create function 19MTfactorial(n int) returns int
begin
        declare f,i int default 1;
        myloop:loop
        if i > n then
        leave myloop;
        else
        set f = f * i;
        set i = i + 1;
        iterate myloop;
        end if;
        end loop;
        return f;
end]
```

```
mysql> create function 19MTfactorial(n int) returns int
    -> begin
    -> declare f,i int default 1;
    -> myloop:loop
    -> if i > n then
    -> leave myloop;
    -> else
    -> set f = f * i;
    -> set i = i + 1;
    -> iterate myloop;
    -> end if;
    -> end loop;
    -> return f;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> select 19MTfactorial(5)]
+------------------+
| 19MTfactorial(5) |
+------------------+
|              120 |
+------------------+
1 row in set (0.00 sec)
```

Stored Procedure:

```
create procedure 19MTfactorial(In n int,Out fact int)
begin
        declare f,i int default 1;
        myloop:loop
        if i > n then
        leave myloop;
        else
        set f = f * i;
        set i = i + 1;
        iterate myloop;
        end if;
        end loop;
        set fact = f;
end]
```

```
mysql> create procedure 19MTfactorial(In n int,Out fact int)
    -> begin
    -> declare f,i int default 1;
    -> myloop:loop
    -> if i > n then
    -> leave myloop;
    -> else
    -> set f = f * i;
    -> set i = i + 1;
    -> iterate myloop;
    -> end if;
    -> end loop;
    -> set fact = f;
    -> end]
Query OK, 0 rows affected (0.01 sec)

mysql> call 19MTfactorial(5, @factorial)]
Query OK, 0 rows affected (0.01 sec)

mysql> select @factorial]
+-------------+
| @factorial |
+-------------+
|         120 |
+-------------+
1 row in set (0.00 sec)
```

3) Write a SQL function and stored procedure to print fibonacci series upto n terms and its sum.

Function:

```
create function 19MTfibonacci(n int) returns varchar(1000)
begin
        declare i int default 3;
        declare a,temp int default 0;
        declare b,sum int default 1;
        declare str varchar(1000);
        set str = CAST(a as char(2));
        set str = CONCAT(str, " ");
        myloop:loop
        if i > n then
        leave myloop;
        else
        set temp = a + b;
        set a = b;
        set b = temp;
        set i = i+1;
        set sum = sum + temp;
        set str = CONCAT(str, CAST(a as char(2)));
        set str = CONCAT(str, " ");
        end if;
        end loop;
        set str = CONCAT(str, CAST(b as char(2)));
        set str = CONCAT(str, " and sum = ");
        set str = CONCAT(str, CAST(sum as char(3)));
        return str;
end]
```

```
mysql> create function 19MTfibonacci(n int) returns varchar(1000)
    -> begin
    -> declare i int default 3;
    -> declare a,temp int default 0;
    -> declare b,sum int default 1;
    -> declare str varchar(1000);
    -> set str = CAST(a as char(2));
    -> set str = CONCAT(str, " ");
    -> myloop:loop
    -> if i > n then
    -> leave myloop;
    -> else
    -> set temp = a + b;
    -> set a = b;
    -> set b = temp;
    -> set i = i+1;
    -> set sum = sum + temp;
    -> set str = CONCAT(str, CAST(a as char(2)));
    -> set str = CONCAT(str, " ");
    -> end if;
    -> end loop;
    -> set str = CONCAT(str, CAST(b as char(2)));
    -> set str = CONCAT(str, " and sum = ");
    -> set str = CONCAT(str, CAST(sum as char(3)));
    -> return str;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> select 19MTfibonacci(6)]
+-------------------------+
| 19MTfibonacci(6)        |
+-------------------------+
| 0 1 1 2 3 5 and sum = 12 |
+-------------------------+
1 row in set (0.01 sec)
```

Stored Procedure:

```
create procedure 19MTsprofibonacci(In n int, Out retStr varchar(1000))
begin
        declare i int default 3;
        declare a,temp int default 0;
        declare b,sum int default 1;
        declare str varchar(1000);
        set str = CAST(a as char(2));
        set str = CONCAT(str, " ");
        myloop:loop
        if i > n then
        leave myloop;
        else
        set temp = a + b;
        set a = b;
        set b = temp;
        set i = i+1;
        set sum = sum + temp;
        set str = CONCAT(str, CAST(a as char(2)));
        set str = CONCAT(str, " ");
        end if;
        end loop;
        set str = CONCAT(str, CAST(b as char(2)));
        set str = CONCAT(str, " and sum = ");
        set str = CONCAT(str, CAST(sum as char(3)));
        set retStr = str;
end]
```

```
mysql> create procedure 19MTsprofibonacci(In n int, Out retStr varchar(1000))
    -> begin
    -> declare i int default 3;
    -> declare a,temp int default 0;
    -> declare b,sum int default 1;
    -> declare str varchar(1000);
    -> set str = CAST(a as char(2));
    -> set str = CONCAT(str, " ");
    -> myloop:loop
    -> if i > n then
    -> leave myloop;
    -> else
    -> set temp = a + b;
    -> set a = b;
    -> set b = temp;
    -> set i = i+1;
    -> set sum = sum + temp;
    -> set str = CONCAT(str, CAST(a as char(2)));
    -> set str = CONCAT(str, " ");
    -> end if;
    -> end loop;
    -> set str = CONCAT(str, CAST(b as char(2)));
    -> set str = CONCAT(str, " and sum = ");
    -> set str = CONCAT(str, CAST(sum as char(3)));
    -> set retStr = str;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> call 19MTsprofibonacci(6, @str)]
Query OK, 0 rows affected (0.00 sec)

mysql> select @str]
+-------------------------+
| @str                    |
+-------------------------+
| 0 1 1 2 3 5 and sum = 12 |
+-------------------------+
1 row in set (0.00 sec)
```

4) Write a SQL function and stored procedure to calculate age.

Function:

```
create function 19MTcalcAge(dat date) returns varchar(25)
begin
        declare curDate date default CURRENT_DATE();
        declare tempDate date;
        declare year,month,date int default 0;
        declare str varchar(25) default "";
        set year = TIMESTAMPDIFF(YEAR, dat, curDate);
        set month = TIMESTAMPDIFF(MONTH, dat, curDate);
        set month = month - (year * 12);
        set tempDate = DATE_ADD(dat, INTERVAL year YEAR);
        set tempDate = DATE_ADD(tempDate, INTERVAL month MONTH);
        set date = DATEDIFF(curDate, tempDate) + 1;
        set str = CONCAT(str, CAST(year as char(2)));
        set str = CONCAT(str, "Y ");
        set str = CONCAT(str, CAST(month as char(2)));
        set str = CONCAT(str, "M ");
        set str = CONCAT(str, CAST(date as char(2)));
        set str = CONCAT(str, "D");
        return str;
end]
```

```
mysql> create function 19MTcalcAge(dat date) returns varchar(25)
    -> begin
    -> declare curDate date default CURRENT_DATE();
    -> declare tempDate date;
    -> declare year,month,date int default 0;
    -> declare str varchar(25) default "";
    -> set year = TIMESTAMPDIFF(YEAR, dat, curDate);
    -> set month = TIMESTAMPDIFF(MONTH, dat, curDate);
    -> set month = month - (year * 12);
    -> set tempDate = DATE_ADD(dat, INTERVAL year YEAR);
    -> set tempDate = DATE_ADD(tempDate, INTERVAL month MONTH);
    -> set date = DATEDIFF(curDate, tempDate) + 1;
    -> set str = CONCAT(str, CAST(year as char(2)));
    -> set str = CONCAT(str, "Y ");
    -> set str = CONCAT(str, CAST(month as char(2)));
    -> set str = CONCAT(str, "M ");
    -> set str = CONCAT(str, CAST(date as char(2)));
    -> set str = CONCAT(str, "D");
    -> return str;
    -> end]
Query OK, 0 rows affected (0.02 sec)

mysql> select 19MTcalcAge('1992-05-11')]
+---------------------------+
| 19MTcalcAge('1992-05-11') |
+---------------------------+
| 27Y 4M 26D                |
+---------------------------+
1 row in set (0.00 sec)
```

Stored Procedure:

```
create procedure 19MTsprocalcAge(In dat date, Out retStr varchar(25))
begin
        declare curDate date default CURRENT_DATE();
        declare tempDate date;
        declare year,month,date int default 0;
        declare str varchar(25) default "";
        set year = TIMESTAMPDIFF(YEAR, dat, curDate);
        set month = TIMESTAMPDIFF(MONTH, dat, curDate);
        set month = month - (year * 12);
        set tempDate = DATE_ADD(dat, INTERVAL year YEAR);
        set tempDate = DATE_ADD(tempDate, INTERVAL month MONTH);
        set date = DATEDIFF(curDate, tempDate) + 1;
        set str = CONCAT(str, CAST(year as char(2)));
        set str = CONCAT(str, "Y ");
        set str = CONCAT(str, CAST(month as char(2)));
        set str = CONCAT(str, "M ");
        set str = CONCAT(str, CAST(date as char(2)));
        set str = CONCAT(str, "D");
        set retStr = str;
end]
```

```
mysql> create procedure 19MTsprocalcAge(In dat date, Out retStr varchar(25))
    -> begin
    -> declare curDate date default CURRENT_DATE();
    -> declare tempDate date;
    -> declare year,month,date int default 0;
    -> declare str varchar(25) default "";
    -> set year = TIMESTAMPDIFF(YEAR, dat, curDate);
    -> set month = TIMESTAMPDIFF(MONTH, dat, curDate);
    -> set month = month - (year * 12);
    -> set tempDate = DATE_ADD(dat, INTERVAL year YEAR);
    -> set tempDate = DATE_ADD(tempDate, INTERVAL month MONTH);
    -> set date = DATEDIFF(curDate, tempDate) + 1;
    -> set str = CONCAT(str, CAST(year as char(2)));
    -> set str = CONCAT(str, "Y ");
    -> set str = CONCAT(str, CAST(month as char(2)));
    -> set str = CONCAT(str, "M ");
    -> set str = CONCAT(str, CAST(date as char(2)));
    -> set str = CONCAT(str, "D");
    -> set retStr = str;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> call 19MTsprocalcAge('1992-05-11',@age)]
Query OK, 0 rows affected (0.00 sec)

mysql> select @age]
+------------+
| @age       |
+------------+
| 27Y 4M 26D |
+------------+
1 row in set (0.00 sec)
```

5) Write a SQL function and stored procedure to count the total number of employees present in the employee table.

Function:

```
create function 19MTtotalNoEmployees() returns int
begin
        declare s int;
        select count(*) from EMPLOYEE into s;
        return s;
end]
```

```
mysql> create function 19MTtotalNoEmployees() returns int
    -> begin
    -> declare s int;
    -> select count(*) from EMPLOYEE into s;
    -> return s;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> select 19MTtotalNoEmployees()]
+-------------------------+
| 19MTtotalNoEmployees() |
+-------------------------+
|                      12 |
+-------------------------+
1 row in set (0.00 sec)
```

Stored Procedure:

```
create procedure 19MTtotalNoEmployees(Out count int)
begin
        declare s int;
        select count(*) from EMPLOYEE into s;
        set count = s;
end]
```

```
mysql> create procedure 19MTtotalNoEmployees(Out count int)
    -> begin
    -> declare s int;
    -> select count(*) from EMPLOYEE into s;
    -> set count = s;
    -> end]
Query OK, 0 rows affected (0.01 sec)

mysql> call 19MTtotalNoEmployees(@res)]
Query OK, 1 row affected (0.00 sec)

mysql> select @res]
+-------+
| @res |
+-------+
|    12 |
+-------+
1 row in set (0.00 sec)
```

6) Write a SQL function and stored procedure to calculate the budget of the department.

Function:

```
create function 19MTcalcBudget(dept varchar(30)) returns int
begin
        declare deptnumber varchar(5);
        declare budget int default 0;
        select deptno from DEPARTMENT where dname = dept into deptnumber;
        select sum(sal) from EMPLOYEE where deptno = deptnumber into budget;
        return budget;
end]
```

```
mysql> create function 19MTcalcBudget(dept varchar(30)) returns int
    -> begin
    -> declare deptnumber varchar(5);
    -> declare budget int default 0;
    -> select deptno from DEPARTMENT where dname = dept into deptnumber;
    -> select sum(sal) from EMPLOYEE where deptno = deptnumber into budget;
    -> return budget;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> select 19MTcalcBudget('Accounting')]
+------------------------------+
| 19MTcalcBudget('Accounting') |
+------------------------------+
|                        67000 |
+------------------------------+
1 row in set (0.02 sec)
```

Stored Procedure:

```
create procedure 19MTcalcBudget(In dept varchar(30), Out budget int)
begin
        declare deptnumber varchar(5);
        declare sumSal int default 0;
        select deptno from DEPARTMENT where dname = dept into deptnumber;
        select sum(sal) from EMPLOYEE where deptno = deptnumber into sumSal;
        set budget = sumSal;
end]
```

```
mysql> create procedure 19MTcalcBudget(In dept varchar(30), Out budget int)
    -> begin
    -> declare deptnumber varchar(5);
    -> declare sumSal int default 0;
    -> select deptno from DEPARTMENT where dname = dept into deptnumber;
    -> select sum(sal) from EMPLOYEE where deptno = deptnumber into sumSal;
    -> set budget = sumSal;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> call 19MTcalcBudget('Accounting',@res)]
Query OK, 1 row affected (0.14 sec)

mysql> select @res]
+--------+
| @res   |
+--------+
| 67000  |
+--------+
1 row in set (0.01 sec)
```

7) Write a SQL function and stored procedure to print the following message:

"Hello <name> How are you".

Function:

```
create function 19MTprintMsg(name varchar(50)) returns varchar(100)
begin
        declare msg varchar(100) default "Hello ";
        set msg = CONCAT(msg, name);
        set msg = CONCAT(msg, " How are you?");
        return msg;
end]
```

```
mysql> create function 19MTprintMsg(name varchar(50)) returns varchar(100)
    -> begin
    -> declare msg varchar(100) default "Hello ";
    -> set msg = CONCAT(msg, name);
    -> set msg = CONCAT(msg, " How are you?");
    -> return msg;
    -> end]
Query OK, 0 rows affected (0.03 sec)

mysql> select 19MTprintMsg('Salman')]
+---------------------------+
| 19MTprintMsg('Salman')    |
+---------------------------+
| Hello Salman How are you? |
+---------------------------+
1 row in set (0.00 sec)
```

Stored Procedure:

```
create procedure 19MTprintMsg(In name varchar(50),Out message varchar(100))
begin
        declare msg varchar(100) default "Hello ";
        set msg = CONCAT(msg, name);
        set msg = CONCAT(msg, " How are you?");
        set message = msg;
end]
```

```
mysql> create procedure 19MTprintMsg(In name varchar(50),Out message varchar(100))
    -> begin
    -> declare msg varchar(100) default "Hello ";
    -> set msg = CONCAT(msg, name);
    -> set msg = CONCAT(msg, " How are you?");
    -> set message = msg;
    -> end]
Query OK, 0 rows affected (0.01 sec)

mysql> call 19MTprintMsg('Salman', @message)]
Query OK, 0 rows affected (0.00 sec)

mysql> select @message]
+---------------------------+
| @message                  |
+---------------------------+
| Hello Salman How are you? |
+---------------------------+
1 row in set (0.00 sec)
```

3. Triggers

Creating tables:

1. Employee

```
create table Employee (
Eid varchar(5) primary key,
Ename varchar(50),
Esal varchar(6)
);
```

```
mysql> create table Employee (
    -> Eid varchar(5) primary key,
    -> Ename varchar(50),
    -> Esal varchar(6)
    -> );
Query OK, 0 rows affected (0.87 sec)
```

2. LogTable

```
create table LogTable (
User varchar(50),
Operation varchar(20),
Time varchar(20),
Peid varchar(5),
Pename varchar(50),
Pesal varchar(6),
Neid varchar(5),
Nename varchar(50),
Nesal varchar(6)
);
```

```
mysql> create table LogTable (
    -> User varchar(50),
    -> Operation varchar(20),
    -> Time varchar(20),
    -> Peid varchar(5),
    -> Pename varchar(50),
    -> Pesal varchar(6),
    -> Neid varchar(5),
    -> Nename varchar(50),
    -> Nesal varchar(6)
    -> );
Query OK, 0 rows affected (0.29 sec)
```

1) Insert Trigger

```
create trigger insertTrig after insert on Employee for each row
begin
insert into LogTable values (user(),'Insert',now(),'-','-','-',new.Eid,new.Ename,new.Esal);
end]
```

```
mysql> create trigger insertTrig after insert on Employee for each row
    -> begin
    -> insert into LogTable values (user(),'Insert',now(),'-','-','-',new.Eid,new.Ename,new.Esal);
    -> end]
Query OK, 0 rows affected (0.19 sec)

mysql> insert into Employee values ('E0001','Salman Zafar','79350')]
Query OK, 1 row affected (0.40 sec)

mysql> select * from LogTable]
+------------------+-----------+---------------------+------+--------+-------+-------+--------------+-------+
| User             | Operation | Time                | Peid | Pename | Pesal | Neid  | Nename       | Nesal |
+------------------+-----------+---------------------+------+--------+-------+-------+--------------+-------+
| Salman@localhost | Insert    | 2019-10-16 22:04:01 | -    | -      | -     | E0001 | Salman Zafar | 79350 |
+------------------+-----------+---------------------+------+--------+-------+-------+--------------+-------+
1 row in set (0.02 sec)
```

2) Update Trigger

```
create trigger updateTrig after update on Employee for each row
begin
insert into LogTable values
(user(),'Update',now(),old.Eid,old.Ename,old.Esal,new.Eid,new.Ename,new.Esal);
end]
```

```
mysql> create trigger updateTrig after update on Employee for each row
    -> begin
    -> insert into LogTable values (user(),'Update',now(),old.Eid,old.Ename,old.Esal,new.Eid,new.Ename,new.Esal);
    -> end]
Query OK, 0 rows affected (0.13 sec)

mysql> update Employee set Esal = '100000' where Eid = 'E0001']
Query OK, 1 row affected (0.28 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from LogTable]
+------------------+-----------+---------------------+-------+--------------+-------+-------+--------------+--------+
| User             | Operation | Time                | Peid  | Pename       | Pesal | Neid  | Nename       | Nesal  |
+------------------+-----------+---------------------+-------+--------------+-------+-------+--------------+--------+
| Salman@localhost | Insert    | 2019-10-16 22:04:01 | -     | -            | -     | E0001 | Salman Zafar | 79350  |
| Salman@localhost | Update    | 2019-10-16 22:36:16 | E0001 | Salman Zafar | 79350 | E0001 | Salman Zafar | 100000 |
+------------------+-----------+---------------------+-------+--------------+-------+-------+--------------+--------+
2 rows in set (0.00 sec)
```

3) Delete Trigger

```
create trigger deleteTrig after delete on Employee for each row
begin
insert into LogTable values (user(),'Delete',now(),old.Eid,old.Ename,old.Esal,'-','-','-');
end]
```

```
mysql> create trigger deleteTrig after delete on Employee for each row
    -> begin
    -> insert into LogTable values (user(),'Delete',now(),old.Eid,old.Ename,old.Esal,'-','-','-');
    -> end]
Query OK, 0 rows affected (0.07 sec)

mysql> delete from Employee where Eid = 'E0001']
Query OK, 1 row affected (0.08 sec)

mysql> select * from LogTable]
+-------------------+-----------+---------------------+-------+--------------+--------+-------+--------------+--------+
| User              | Operation | Time                | Peid  | Pename       | Pesal  | Neid  | Nename       | Nesal  |
+-------------------+-----------+---------------------+-------+--------------+--------+-------+--------------+--------+
| Salman@localhost  | Insert    | 2019-10-16 22:04:01 | -     | -            | -      | E0001 | Salman Zafar | 79350  |
| Salman@localhost  | Update    | 2019-10-16 22:36:16 | E0001 | Salman Zafar | 79350  | E0001 | Salman Zafar | 100000 |
| Salman@localhost  | Delete    | 2019-10-16 22:41:42 | E0001 | Salman Zafar | 100000 | -     | -            | -      |
+-------------------+-----------+---------------------+-------+--------------+--------+-------+--------------+--------+
3 rows in set (0.00 sec)
```

4. Cursor

i) Write a cursor to output salary of all employees in a string.

```
create procedure mypro(out s varchar(6))
begin
declare f int default 1;
declare str longtext default "";
declare cur cursor for select Esal from Employee;
declare continue handler for not found set f=0;
open cur;
myloop:loop
fetch cur into s;
if f=0 then
leave myloop;
else
set str = CONCAT(str," ",s);
end if;
end loop;
close cur;
select str;
end]
```

```
mysql> select * from Employee]
+-------+-------+-------+
| Eid   | Ename | Esal  |
+-------+-------+-------+
| E0002 | ABC   | 30000 |
| E0003 | XYZ   | 40000 |
| E0004 | DEF   | 50000 |
+-------+-------+-------+
3 rows in set (0.00 sec)

mysql> create procedure mypro(out s varchar(6))
    -> begin
    -> declare f int default 1;
    -> declare str longtext default "";
    -> declare cur cursor for select Esal from Employee;
    -> declare continue handler for not found set f=0;
    -> open cur;
    -> myloop:loop
    -> fetch cur into s;
    -> if f=0 then
    -> leave myloop;
    -> else
    -> set str = CONCAT(str," ",s);
    -> end if;
    -> end loop;
    -> close cur;
    -> select str;
    -> end]
Query OK, 0 rows affected (0.00 sec)

mysql> call mypro(@s)]
+--------------------+
| str                |
+--------------------+
|  30000 40000 50000 |
+--------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Name:- Nabeel Mohammad Rizwan
Roll No.- 20BCS087
DBMS lab assignment - 11

Assignment-11

use 20BCS087;

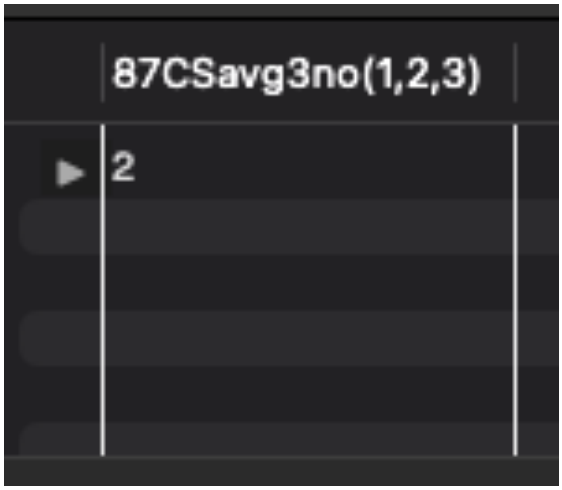
Functions and Stored procedures:

1)

Function:

```
delimiter //
create function 87CSavg3no(a int,b int,c int)returns int
reads sql data deterministic
begin
declare avg,sum int;
set sum := a+b+c;
set avg := sum/3;
return avg;
end
//
```

select 87CSavg3no(1,2,3);

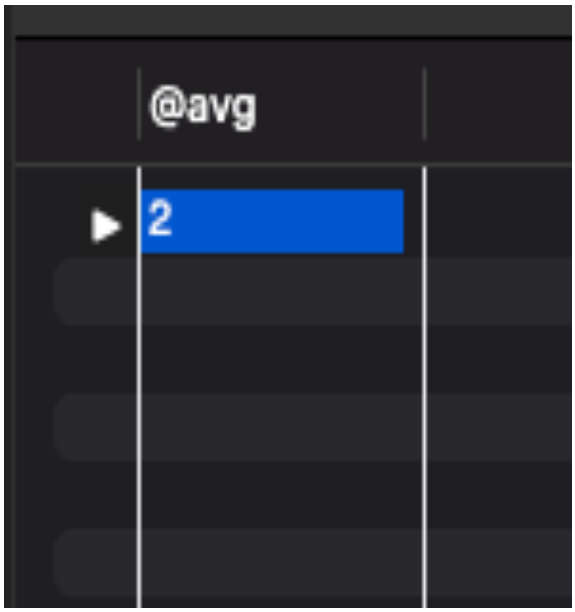


87CSavg3no(1,2,3)
2

Stored procedure:

```
delimiter //
create procedure 87CSavg3no(In a int,In b int,In c int,Out t int)
begin
declare sum int;
set sum = a+b+c;
set t = sum/3;
```

```
end
//

call 87CSavg3no(1,2,3,@avg);

select @avg;
```



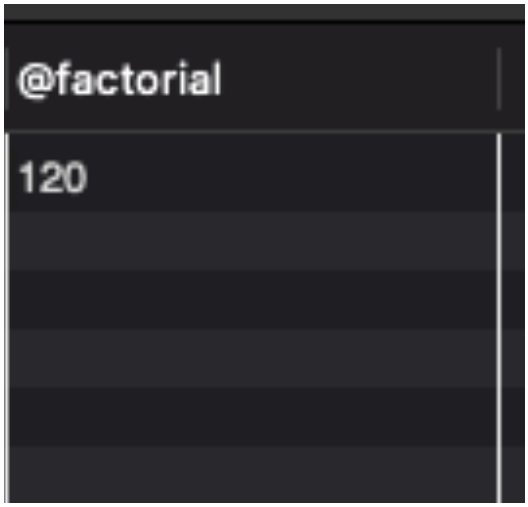
2)

Function:

```
delimiter //
create function 87CSfactorial(n int)returns int
reads sql data deterministic
begin
declare f,i int default 1;
myloop:loop
if i>n then
leave myloop;
else
set f = f*i;
set i = i+1;
iterate myloop;
end if;
end loop;
return f;
end
//

select 87CSfactorial(5);
```

Stored Procedure:

```
delimiter //
create procedure 87CSfactorial(In n int,Out fact int)
reads sql data deterministic
begin
declare f,i int default 1;
myloop:loop
if i>n then
leave myloop;
else
set f=f*i;
set i=i+1;
iterate myloop;
end if;
end loop;
set fact=f;
end
//

call 87CSfactorial(5,@factorial);

select @factorial;
```

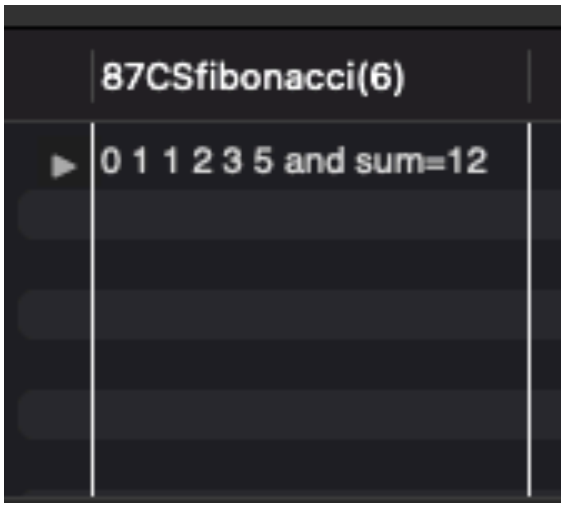


3)

Function:

```
delimiter //
create function 87CSfibonacci(n int)returns varchar(1000)
reads sql data deterministic
begin
declare i int default 3;
declare a,temp int default 0;
declare b,sum int default 1;
declare str varchar(1000);
set str = CAST(a as char(2));
set str = CONCAT(str," ");
myloop:loop
if i>n then
leave myloop;
else
set temp = a+b;
set a=b;
set b=temp;
set i=i+1;
set sum=sum+temp;
set str=CONCAT(str,CAST(a as char(2)));
set str=CONCAT(str," ");
end if;
end loop;
set str=CONCAT(str,CAST(b as char (2)));
set str=CONCAT(str, " and sum=");
set str=CONCAT(str,CAST(sum as char(3)));
return str;
end
//
```

select 87CSfibonacci(6);



Stored procedure:

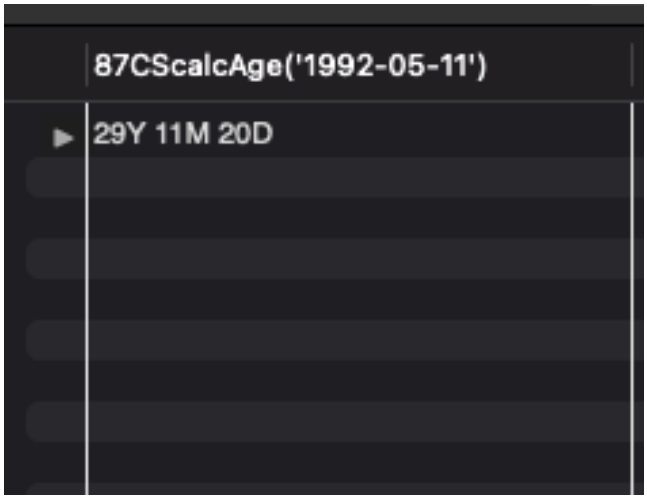
```
delimiter //
create procedure 87CSsproffibonacci(In n int,Out retstr varchar(1000))
reads sql data deterministic
begin
declare i int default 3;
declare a,temp int default 0;
declare b,sum int default 1;
declare str varchar(1000);
set str=CAST(a as char(2));
set str=CONCAT(str," ");
myloop:loop
if i>n then
leave myloop;
else
set temp=a+b;
set a=b;
set b=temp;
set i=i+1;
set sum=sum+temp;
set str=CONCAT(str,CAST(a as char(2)));
set str=CONCAT(str," ");
end if;
end loop;
set str=CONCAT(str,CAST(b as char(2)));
set str=CONCAT(str, "and sum = ");
set str=CONCAT(str,CAST(sum as char(3)));
set retstr=str;
end
//

call 87CSsproffibonacci(6,@str);
```

select @str;

4)

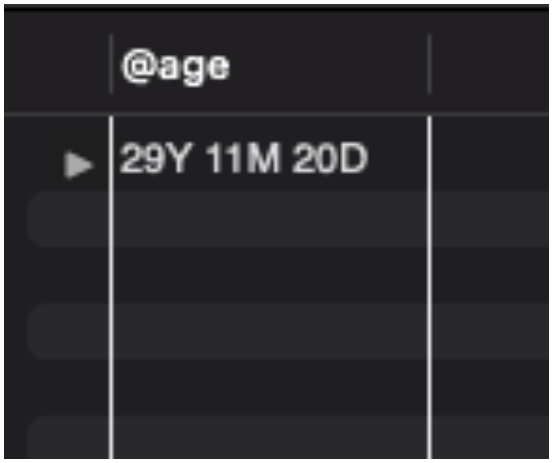
Function:

```
delimiter //
create function 87CScalcAge(dat date)returns varchar(25)
reads sql data deterministic
begin
declare curDate date default CURRENT_DATE();
declare tempDate date;
declare year,month,date int default 0;
declare str varchar(25) default "";
set year = TIMESTAMPDIFF(YEAR,dat,curDate);
set month = TIMESTAMPDIFF(MONTH,dat,curDate);
set month = month - (year*12);
set tempDate = DATE_ADD(dat,INTERVAL year YEAR);
set tempDate = DATE_ADD(tempDate,INTERVAL month MONTH);
set date = DATEDIFF(curDATE,tempDate)+1;
set str=CONCAT(str,CAST(year as char(2)));
set str=CONCAT(str,"Y ");
set str=CONCAT(str,CAST(month as char(2)));
set str=CONCAT(str,"M ");
set str=CONCAT(str,CAST(date as char(2)));
set str=CONCAT(str,"D ");
return str;
end
//

select 87CScalcAge('1992-05-11');
```



Stored procedure:

```
delimiter //
create procedure 87CSsprocalcAge(In dat date,Out retstr varchar(25))
reads sql data deterministic
begin
declare curDate date default CURRENT_DATE();
declare tempDate date;
declare year,month,date int default 0;
declare str varchar(25) default "";
set year = TIMESTAMPDIFF(YEAR,dat,curDate);
set month = TIMESTAMPDIFF(MONTH,dat,curDate);
set month = month-(year*12);
set tempDate=DATE_ADD(dat, INTERVAL year YEAR);
set tempDate=DATE_ADD(tempDate,INTERVAL month MONTH);
set date=DATEDIFF(curDate,tempDate)+1;
set str=CONCAT(str,CAST(year as char(2)));
set str=CONCAT(str,"Y ");
set str=CONCAT(str,CAST(month as char(2)));
set str=CONCAT(str,"M ");
set str=CONCAT(str,CAST(date as char(2)));
set str=CONCAT(str,"D ");
set retStr=str;
end
//

call 87CSsprocalcAge('1992-05-11',@age);

select @age;
```



5)

```
create table employee(
S_No int,
employee varchar(30)
);

insert into employee values(1,'sample employee 1'),
(2,'sample employee 2'),
(3,'sample employee 3');
```

Function:

```
delimiter //
create function 87CSTotalNoEmployees()returns int
reads sql data deterministic
begin
declare s int;
select count(*) from employee into s;
return s;
end
//

select 87CSTotalNoEmployees();
```

87CSTotalNoEmployee...
3

Stored procedure:

```
delimiter //
create procedure 87CStotalNoEmployees(Out count int)
reads sql data deterministic
begin
declare s int;
select count(*) from employee into s;
set count=s;
end
//

call 87CStotalNoEmployees(@res);

select @res;
```

@res
3

6)

Function:

```
create function 87CSCalcBudget(dept varchar(30)) returns int  begin
 declare deptnumber varchar(5);
 declare budget int default 0;
 select deptno from DEPARTMENT where

dname=dept into deptnumber;
 select sum(sal) from Employee where

deptno=deptnumber into budget;

return budget;
end ]

select 87CSCalcBudget('Accounting') ]
```

Stored Procedure:

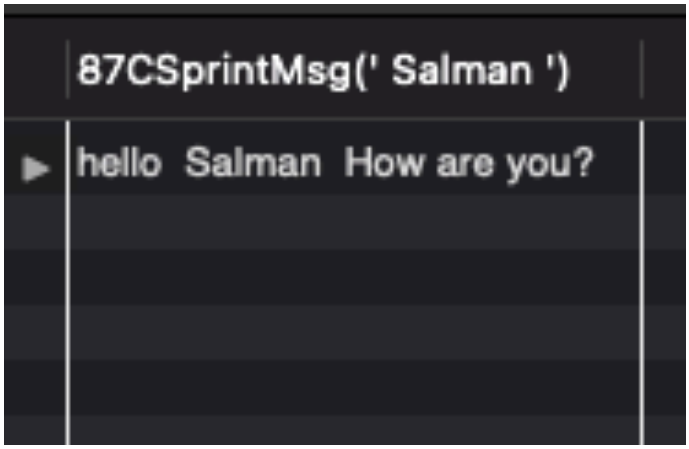
```
Create procedure 87CSCalcBudget(In dept varchar(30),Out budget int)

 begin
 declare deptnumber varchar(5);

 declare budget int default 0;

select deptno from DEPARTMENT where dname=dept into deptnumber;

select sum(sal) from Employee where deptno=deptnumber into budget;

set budget = sumSal;

end ]


call 87CSCalcBudget('Accounting',@res) ]

select @res ]
```

```
+------+
| @res |
+------+
|0     |
+------+
```

7)

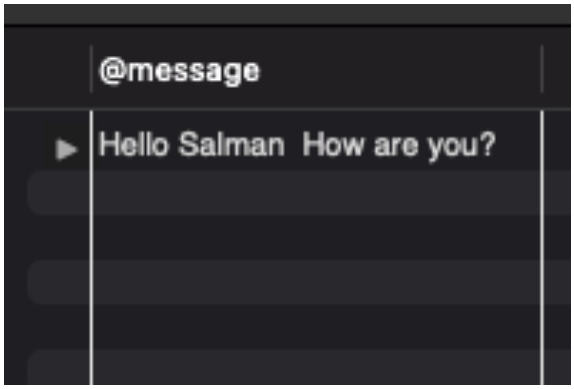
Function:

```
delimiter //
create function 87CSprintMsg(name varchar(20))returns varchar(100)
reads sql data deterministic
begin
declare msg varchar(100) default "hello ";
set msg=CONCAT(msg,name);
set msg=CONCAT(msg," How are you?");
return msg;
end
//

select 87CSprintMsg(' Salman ');
```



Stored procedure:

```
delimiter //
create procedure 87CSprintMsg(In name varchar(50),Out message varchar(100))
reads sql data deterministic
begin
declare msg varchar(100) default "Hello";
set msg=CONCAT(msg,name);
set msg=CONCAT(msg," How are you? ");
set message=msg;
end
//

call 87CSprintMsg(' Salman ', @message);

select @message;
```



Triggers:

```
create table employee(
Eid varchar(5) primary key,
Ename varchar(50),
Esal varchar(6)
);

create table LogTable(
User varchar(50),
operation varchar(50),
Time varchar(20),
Peid varchar(5),
Pename varchar(50),
Pesal varchar(6),
Neid varchar(5),
Nename varchar(50),
Nesal varchar(6)
);
```

1) Insert Trigger

```
delimiter //
create trigger inserting after insert on Employee for each row
begin
insert into LogTable values (user(),'Insert',now(),'-','-','-',new.Eid,new.Ename,new.Esal);
end
//

insert into employee values('E0001','Salman Zafar','79350');

select * from logtable;
```

User	operation	Time	Peid	Pename	Pesal	Neid	Nename	Nesal
root@localhost	Insert	2022-04-30 06:17:38	-	-	-	E0001	Salman Zafar	79350

2) Update Trigger

```
delimiter //
create trigger updateTrig after update on Employee for each row
begin
insert into LogTable values
(user(),'update',now(),old.Eid,old.Ename,old.Esal,new.Eid,new.Ename,new.Esal);
end
//

update employee set esal='100000' where Eid='E0001';
select * from logtable;
```

User	operation	Time	Peid	Pename	Pesal	Neid	Nename	Nesal
root@localhost	Insert	2022-04-30 06:17:38	-	-	-	E0001	Salman Zafar	79350
root@localhost	update	2022-04-30 06:25:36	E0001	Salman Zafar	100000	E0001	Salman Zafar	100000

Delete Trigger:

```
delimiter //
create trigger deleteTrig after delete on Employee for each row
begin
insert into LogTable values (user(),'Delete',now(),old.Eid,old.Ename,old.Esal,'-','-','-');
end
//

delete from employee where Eid='E0001';
select * from logtable;
```

User	operation	Time	Peid	Pename	Pesal	Neid	Nename	Nesal
root@localhost	Insert	2022-04-30 06:17:38	-	-	-	E0001	Salman Zafar	79350
root@localhost	update	2022-04-30 06:25:36	E0001	Salman Zafar	100000	E0001	Salman Zafar	100000
root@localhost	Delete	2022-04-30 06:30:16	E0001	Salman Zafar	100000	-	-	-

Cursor:

1)

```
create table employee(
Eid varchar(10),
Ename varchar(30),
Esal int
);

insert into employee values('E0002','ABC',30000),
('E0003','XYZ',40000),
('E0004','DEF',50000);

select * from employee;
```

Eid	Ename	Esal
E0002	ABC	30000
E0003	XYZ	40000
E0004	DEF	50000

```
delimiter //
create procedure mypro(out s varchar(6))
reads sql data deterministic
begin
declare f int default 1;
declare str longtext default "";
declare cur cursor for select Esal from employee;
declare continue handler for not found set f=0;
open cur;
myloop:loop
fetch cur into s;
if f=0 then
leave myloop;
else
set str=CONCAT(str," ",s);
end if;
end loop;
close cur;
select str;
end
//

call mypro(@s);
```

str
30000 40000 50000