SQL Queries

1. **Create Table Name: Student and Exam.**

➔ **Student table:**

Rollno	Name	Branch
1	Jay	Computer Science
2	Suhani	Electric and Com
3	Kriti	Electric and Com
NULL	NULL	NULL

➔ **Student table query:** create database Pro1;

```
use Pro1;
create table student (Rollno int primary key auto_increment,
Name varchar (30) not null,
Branch varchar (30) not null);

insert into student (Name, Branch)
values ('Jay', 'Computer Science'),
('Suhani', 'Electric and Com'),
('Kriti', 'Electric and Com');

select * from student;
```

➔ **Exam table:**

Rollno	S_code	Marks	P_code
1	CS11	50	CS
1	CS12	60	CS
2	EC101	66	EC
2	EC102	70	EC
3	CS101	45	EC
3	CS102	50	EC

➔ **Exam table query:** create table exam (

```
Rollno int,
foreign key (Rollno) references student (Rollno),
S_code text not null,
Marks int not null,
P_code text null);

insert into exam (Rollno, S_code, Marks, P_code)
values(1,'CS11',50,'CS'),
(1,'CS12',60,'CS'),
(2,'EC101',66,'EC'),
(2,'EC102',70,'EC'),
(3,'CS101',45,'EC'),
(3,'CS102',50,'EC');

select * from exam;
```

2. **Create table given below: Employee and Incentive Table.**

➔ **Employee table:**

Employee_id	First_Name	Last_Name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	John	Abraham	1000000	2013-01-01 12:00:00	Banking
3	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
4	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
5	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
6	Jerry	Pinto	650000	2013-01-01 12:00:00	Insurance
7	Philip	Mathew	750000	2013-01-01 12:00:00	Services
8	TestName1	123	650000	2013-01-01 12:00:00	Services
9	TestName1	Lname%	60000	2013-02-01 12:00:00	Insurance
NULL	NULL	NULL	NULL	NULL	NULL

➔ **Employee table query:** create database Pro2;

```
use Pro2;

create table Employee (
 Employee_id int primary key auto_increment,
 First_Name varchar(30) not null,
 Last_Name varchar(30) not null,
 Salary int not null,
 Joining_date datetime not null,
 Department varchar(30));

INSERT INTO Employee (First_Name, Last_Name, Salary, Joining_date, Department)
VALUES ('Michael', 'Clarke', 800000, '2013-01-01 12:00:00', 'Insurance'),
('Roy', 'Thomas', 700000, '2013-02-01 12:00:00', 'Banking'),
('Tom', 'Jose', 600000, '2013-02-01 12:00:00', 'Insurance'),
('Jerry', 'Pinto', 650000, '2013-01-01 12:00:00', 'Insurance'),
('Philip', 'Mathew', 750000, '2013-01-01 12:00:00', 'Services'),
('TestName1', '123', 650000, '2013-01-01 12:00:00', 'Services'),
('TestName1', 'Lname%', 60000, '2013-02-01 12:00:00', 'Insurance');

select * from Employee;
```

➔ **Incentive table:**

Employee_ref_id	Incentive_date	Incentive_amount
1	2013-02-01	5000
2	2013-02-01	3000
3	2013-02-01	4000
1	2013-01-01	4500
3	2013-01-01	3500

➔ **Incentive table query:** CREATE TABLE Incentive (

```
Employee_ref_id INT,
Incentive_date DATE NOT NULL,
Incentive_amount INT NOT NULL);

 insert into Incentive (Employee_ref_id, Incentive_date, Incentive_amount)
 values (1,'2013-02-01',5000),
 (2,'2013-02-01',3000),
 (3,'2013-02-01',4000),
 (1,'2013-01-01',4500),
 (3,'2013-01-01',3500);

 select * from Incentive;
```

3. **Get First_Name from employee table using Tom name “Employee Name”.**

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

as employee_name
Tom

➔ **Query:** select First_Name as employee_name from Employee where First_Name = 'Tom';

4. **Get FIRST_NAME, Joining Date, and Salary from employee table.**

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

First_Name	Joining_date	Salary
John	2013-01-01 12:00:00	1000000
John	2013-01-01 12:00:00	1000000
Michael	2013-01-01 12:00:00	800000
Roy	2013-02-01 12:00:00	700000
Tom	2013-02-01 12:00:00	600000
Jerry	2013-01-01 12:00:00	650000
Philip	2013-01-01 12:00:00	750000
TestName1	2013-01-01 12:00:00	650000
TestName1	2013-02-01 12:00:00	60000

➔ **Query:** select First_Name, Joining_date, Salary from Employee;

5. **Get all employee details from the employee table order by First_Name Ascending and Salary descending?**

➔ **First_Name Ascending:**

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Employee_id	First_Name	Last_Name	Salary	Joining_date	Department
6	Jerry	Pinto	650000	2013-01-01 12:00:00	Insurance
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	John	Abraham	1000000	2013-01-01 12:00:00	Banking
3	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
7	Philip	Mathew	750000	2013-01-01 12:00:00	Services
4	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
8	TestName1	123	650000	2013-01-01 12:00:00	Services
9	TestName1	Lname%	60000	2013-02-01 12:00:00	Insurance
5	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
NULL	NULL	NULL	NULL	NULL	NULL

➔ **First_Name Ascending query:** SELECT * FROM Employee ORDER BY First_Name ASC;

➔ **Salary descending:**

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Employee_id	First_Name	Last_Name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	John	Abraham	1000000	2013-01-01 12:00:00	Banking
3	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
7	Philip	Mathew	750000	2013-01-01 12:00:00	Services
4	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
6	Jerry	Pinto	650000	2013-01-01 12:00:00	Insurance
8	TestName1	123	650000	2013-01-01 12:00:00	Services
5	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
9	TestName1	Lname%	60000	2013-02-01 12:00:00	Insurance
NULL	NULL	NULL	NULL	NULL	NULL

➔ **Salary descending query:** select * from employee order by Salary DESC;

6. **Get employee details from employee table whose first name contains 'J'.**

➔ **first name contains 'J':**

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Employee_id	First_Name	Last_Name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	John	Abraham	1000000	2013-01-01 12:00:00	Banking
6	Jerry	Pinto	650000	2013-01-01 12:00:00	Insurance
NULL	NULL	NULL	NULL	NULL	NULL

➔ **Query:** select * from Employee where First_Name like 'j%';

7. **&**
8. **Get department wise maximum salary from employee table order by salary ascending?**

➔ **Max salary:**

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Department	Max_salary
Services	750000
Insurance	800000
Banking	1000000

➔ **Query:** select Department, max(Salary) as Max_salary from employee
group by Department
order by Max_salary asc;

9. **Select first_name, incentive amount from employee and incentives table forthose employees who have incentives and incentive amount greater than 3000.**

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

First_Name	Incentive_amount
John	5000
Michael	4000
John	4500
Michael	3500

➔ **Query:** SELECT e.First_Name, i.Incentive_amount
FROM Employee e
JOIN Incentive i ON e.Employee_id = i.Employee_ref_id
WHERE i.Incentive_amount > 3000;

10. **Create After Insert trigger on Employee table which insert records in view table.**

➔ **Trigger:**

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Employee_id	First_Name	Last_Name	Salary	Joining_date	Department
1	John	Abraham	1000000	2013-01-01 12:00:00	Banking
2	John	Abraham	1000000	2013-01-01 12:00:00	Banking
3	Michael	Clarke	800000	2013-01-01 12:00:00	Insurance
4	Roy	Thomas	700000	2013-02-01 12:00:00	Banking
5	Tom	Jose	600000	2013-02-01 12:00:00	Insurance
6	Jerry	Pinto	650000	2013-01-01 12:00:00	Insurance
7	Philip	Mathew	750000	2013-01-01 12:00:00	Services
8	TestName1	123	650000	2013-01-01 12:00:00	Services
9	TestName1	Lname%	60000	2013-02-01 12:00:00	Insurance
10	Darshan	Tank	100000	2023-04-28 12:00:00	Insurance
11	Dharam	Chavda	90000	2022-01-02 12:00:00	Banking
12	Dipesh	Chavda	95000	2023-02-02 12:00:00	Insurance
NULL	NULL	NULL	NULL	NULL	NULL

➔ **Query:** create trigger demotable
after insert
on employee for each row
begin
INSERT INTO Employee (First_Name, Last_Name, Salary, Joining_date, Department, record)
VALUES (NEW.First_Name, NEW.Last_Name, NEW.Salary, NEW.Joining_date, NEW.Department)

```
end//
delimiter ;

select * from employee;
```

11. **&**
12. **Create table given below: Salesperson and Customer.**

➔ **Salesperson table:**

SNo	SNAME	CITY	COMM
1001	PEEL	LONDON	0.12
1002	SERRES	SAN JOSE	0.13
1003	MOTIKA	LONDON	0.11
1004	RAFKIN	BARCELONA	0.15
1005	AXELROD	NEW YORK	0.1
NULL	NULL	NULL	NULL

➔ **Salesperson table query:** create database Pro3;

```
use Pro3;

create table Salesperson (
 SNo int primary key,
 SNAME varchar(30) not null,
 CITY varchar(30) not null,
 COMM float not null);

INSERT INTO Salesperson (SNo, SNAME, CITY, COMM)
VALUES (1001,'PEEL', 'LONDON', .12),
(1002,'SERRES', 'SAN JOSE', .13),
(1004,'MOTIKA', 'LONDON', .11),
(1007,'RAFKIN', 'BARCELONA', .15),
(1003,'AXELROD', 'NEW YORK', .1);

select * from Salesperson;
```

➔ **CUSTOMER TABLE:**

COMM	CNAME	CITY	RATING	SNo
201	Hoffman	London	100	1001
202	Giovanne	Roe	200	1003
203	Liu	san jose	300	1002
204	grass	Barselona	100	1002
206	Clemens	London	300	1007
207	Pereira	Roe	100	1004
NULL	NULL	NULL	NULL	NULL

➔ **CUSTOMER TABLE QUERY:** CREATE TABLE CUSTOMER (

```
COMM int primary key auto_increment,
CNAME varchar(30) NOT NULL,
CITY varchar(30) NOT NULL,
RATING INT,
SNo int ,
foreign key (SNo) references Salesperson (SNo));

 insert into CUSTOMER (COMM,CNAME,CITY,RATING,SNo)
 values (201, 'Hoffman','London',100 , 1001),
 (202,'Giovanne', 'Roe', 200, 1003),
 (203, 'Liu', 'san jose', 300, 1002),
```

(204,'grass', 'Barselona' , 100, 1002),
(206,'Clemens', 'London', 300, 1007),
(207,'Pereira', 'Roe', 100, 1004);

select * from CUSTOMER;

13. All orders for more than $1000.

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

max(RATING)
NULL

➔ **Query:** select max(RATING) from CUSTOMER where RATING > 1000;

14. Names and cities of all salespeople in London with commission above 0.12.

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

SNAME	city

➔ **Query:** select SNAME, city from Salesperson where city = 'London' AND comm > 0.12;

15. All salespeople either in Barcelona or in London.

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

SNAME	city
PEEL	LONDON
MOTIKA	LONDON
RAFKIN	BARCELONA

➔ **Query:** select SNAME, city from Salesperson where city = 'London' OR city = 'Barcelona';

16. All salespeople with commission between 0.10 and 0.12.

➔ .

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

SNAME	COMM
PEEL	0.12
AXELROD	0.1
MOTIKA	0.11

➔ **Query:** select SNAME, COMM from Salesperson where COMM > 0.10 AND COMM < 0.12;

17. All customers excluding those with rating <= 100 unless they are located in Rome.

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CNAME	RATING	city
Hoffman	100	London
Giovanne	200	Roe
grass	100	Barselona
Pereira	100	Roe

➔ **Query:** select CNAME, RATING, city from CUSTOMER where city = 'Roe' or RATING <= 100;

18. Write a SQL statement that displays all the information about all salespeople.

➔ **Salesman table:**

salesman_id	name	CITY	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5003	Lauson Hen	San Jose	0.12
5005	Pit Alex	LONDON	0.11
5006	Mc Lyon	Paris	0.14
5007	Paul Adam	Rome	0.13
NULL	NULL	NULL	NULL

➔ **Salesman table query:** create database Pro4;

```
use Pro4;

create table salesman (
 salesman_id int primary key,
 name varchar(30) not null,
 CITY varchar(30) not null,
 commission float not null);

INSERT INTO salesman (salesman_id, name, CITY, commission)
VALUES (5001,'James Hoog', ' New York', 0.15),
(5002,'Nail Knite', 'Paris', 0.13),
(5005,'Pit Alex', 'LONDON', 0.11),
(5006,'Mc Lyon', 'Paris', 0.14),
(5007,'Paul Adam', 'Rome', 0.13),
(5003,'Lauson Hen', 'San Jose', 0.12);

select * from salesman;
```

19. From the following table, write a SQL query to find orders that are delivered by a salesperson with ID. 5001. Return ord_no, ord_date, purch_amt.

➔ **Order table:**

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.5	2012-10-05	3005	5002
70002	65.26	2012-10-05	3002	5001
70003	2480.4	2012-10-10	3009	5003
70004	110.5	2012-08-17	3009	5003
70005	2400.6	2012-07-27	3007	5001
70007	948.5	2012-09-10	3005	5002
70008	5760	2012-09-10	3002	5001
70009	270.65	2012-09-10	3001	5005
70010	1983.43	2012-10-10	3004	5006
70011	75.29	2012-08-17	3003	5007
70012	250.45	2012-06-27	3008	5002
70013	3045.6	2012-04-25	3002	5001
NULL	NULL	NULL	NULL	NULL

➔ **Order table query:** select * from salesman;

```
CREATE TABLE orders (
ord_no int primary key not null,
purch_amt float NOT NULL,
ord_date date NOT NULL,
customer_id INT,
```

```
salesman_id int ,
foreign key (salesman_id) references salesman (salesman_id));

insert into orders (ord_no,purch_amt,ord_date,customer_id,salesman_id)
values (70001, 150.5,'2012-10-05', 3005, 5002),
(70009, 270.65, '2012-09-10', 3001, 5005),
(70002, 65.26, '2012-10-05', 3002, 5001),
(70004, 110.5, '2012-08-17', 3009, 5003),
(70007, 948.5, '2012-09-10', 3005, 5002),
(70005, 2400.6, '2012-07-27', 3007, 5001),
(70008, 5760, '2012-09-10', 3002, 5001),
(70010, 1983.43, '2012-10-10', 3004, 5006),
(70003, 2480.4, '2012-10-10', 3009, 5003),
(70012, 250.45, '2012-06-27', 3008, 5002),
(70011, 75.29, '2012-08-17', 3003, 5007),
(70013, 3045.6, '2012-04-25', 3002, 5001);

select * from orders;
```

➔ **Return ord_no, ord_date, purch_amt:**

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

salesman_id	ord_no	purch_amt
5001	70002	65.26
5001	70005	2400.6
5001	70008	5760
5001	70013	3045.6
NULL	NULL	NULL

➔ **Query:** select salesman_id, ord_no, purch_amt from orders where salesman_id = 5001;

20. From the following table, write a SQL query to select a range of products whose price is in the range Rs.200 to Rs.600. Begin and end values are included. Return pro_id, pro_name, pro_price, and pro_com.

➔ **item_mast table:**

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

PRO_ID	PRO_NAME	PRO_PRICE	PRO_COM
101	Mother Board	3200	15
102	Key Board	450	16
103	ZIP drive	250	14
104	Speaker	550	16
105	Monitor	5000	11
106	DVD drive	900	12
107	CD drive	800	12
108	Printer	2600	13
109	Refill cartridge	350	13
110	Mouse	250	12
NULL	NULL	NULL	NULL

➔ **item_mast query:** create database Pro5;
use Pro5;

```
create table item_mast (
 PRO_ID int primary key auto_increment,
 PRO_NAME varchar(30) not null,
 PRO_PRICE float not null,
 PRO_COM int not null);

 alter table `item_mast` auto_increment = 101;

INSERT INTO item_mast (PRO_NAME, PRO_PRICE, PRO_COM)
VALUES ('Mother Board', 3200.00, 15),
('Key Board', 450.00, 16),
('ZIP drive', 250.00, 14),
('Speaker', 550.00, 16),
('Monitor', 5000.00, 11),
('DVD drive', 900.00, 12),
('CD drive', 800.00, 12),
('Printer', 2600.00, 13),
('Refill cartridge', 350.00, 13),
('Mouse', 250.00, 12);

select * from item_mast;
```

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PRO_NAME	PRO_PRICE
Key Board	450
ZIP drive	250
Speaker	550
Refill cartridge	350
Mouse	250

➔ **Query:** select PRO_NAME, PRO_PRICE from item_mast where PRO_PRICE > 200 and PRO_PRICE < 600;

21. From the following table, write a SQL query to calculate the average price for a manufacturer code of 16. Return avg.

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

avg_price
500

➔ **Query: select avg (PRO_PRICE) as avg_price from item_mast where PRO_COM = 16;**

22. From the following table, write a SQL query to display the pro_name as 'Item Name' and pro_priceas 'Price in Rs.'

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Item Name	Price in Rs.
Mother Board	3200
Key Board	450
ZIP drive	250
Speaker	550
Monitor	5000
DVD drive	900
CD drive	800
Printer	2600
Refill cartridge	350
Mouse	250

➔ **Query:** select PRO_NAME as 'Item Name', PRO_PRICE as 'Price in Rs.' from item_mast;

23. From the following table, write a SQL query to find the items whose prices are higher than or equal to $250. Order the result by product price in descending, then product name in ascending. Return pro_name and pro_price.

➔

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PRO_NAME	PRO_PRICE
Mother Board	3200
Key Board	450
ZIP drive	250
Speaker	550
Monitor	5000
DVD drive	900
CD drive	800
Printer	2600
Refill cartridge	350
Mouse	250

➔ **Query:** select PRO_NAME, PRO_PRICE from item_mast where PRO_PRICE >= 250;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PRO_NAME	PRO_PRICE
Monitor	5000
Mothe Monitor	3200
Printer	2600
DVD drive	900
CD drive	800
Speaker	550
Key Board	450
Refill cartridge	350
ZIP drive	250
Mouse	250

➔

➔ **Query:** select PRO_NAME, PRO_PRICE from item_mast order by PRO_PRICE desc;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PRO_NAME	PRO_PRICE
CD drive	800
DVD drive	900
Key Board	450
Monitor	5000
Mother Board	3200
Mouse	250
Printer	2600
Refill cartridge	350
Speaker	550
ZIP drive	250

➔

➔ **Query:** select PRO_NAME, PRO_PRICE from item_mast order by PRO_NAME asc;

24. **From the following table, write a SQL query to calculate average price of the items for each company. Return average price and company code**.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

compney_cod	average_price
15	3200
16	500
14	250
11	5000
12	650
13	1475

➔

➔ **Query:** select PRO_COM as compney_cod, avg(PRO_PRICE) as average_price
from item_mast
group by PRO_COM;