1.Create table employee,dept with following column and insert given data(3 Marks)

```
create table employee(emp_id integer primary key,
Name character varying not null ,
Age Integer not null,
hobbies character varying not null ,
salary integer not null check ( salary > 0 ),
address character varying not Null ,
zip integer unique);
```

```
create table dept(dept_id integer primary key,
dept_name character varying ,
e_id integer references employee (emp_id),
manager character varying);
```

2. Q2 INSERT FOLLOWING DATA TO EMPLOYEE (3 Marks

```
insert into employee values (1,'mohit',23,'dancing', 10000, 'Mumbai',500049),
(2,'aniket',27,'painting', 20000, 'mumbai',500149),
(3,'ajay',31,'singing', 35000, 'delhi',273008),
(4,'priyanka',42,'dancing', 55000, 'delhi',123876),
(5,'deepika',26,'dancing', 10000, 'delhi',500786),
(6,'saloni',28,'singing', 50000, 'Mumbai',400149),
(7,'yash',34,'photography', 40000, 'Mumbai',450049),
(8,'vinay',45,'painting', 70000, 'Mumbai',273006);
```

```
insert into dept values (1 ,'ec',8, 'virat'),
(2,'cs',7, 'sachin'),
(3,'it',6, 'rahul'),
(4,'it',5, 'rahul'),
(5,'cs',4, 'sachin'),
(6,'ec',3, 'virat'),
(7,'ec',2, 'virat'),
(8,'ec',1, 'virat');
```

Q. Write a Query to count No. of employees (2 Marks)

```
>>select count (emp_id) from employee;
```

Q.Write a Query to get unique department of employees (2 Marks)

```
>>select distinct(dept_name) from dept;
```

Q.Write a Query to get min,max,avg,sum of salary for all employees (2 Marks)

```
>>select max(salary),min(salary), avg(salary), sum(salary) from employee;
```

Q.Write a Query to get highest salary of an individual based on hobbies (2 Marks)

```
>>select max(salary),hobbies from employee group by hobbies;
```

Q.Write a Query for sum of salary where address starts with 'M' or 'd' (2 Marks)

```
>>select sum(salary) from employee where address like 'M%' or address like 'd%';
```

Q.Write a Query to Get all employee details with their department details(2 Marks)

```
>>select * from employee e join dept d on e.emp_id=d.e_id ;
```

Q.Write a QUERY TO FIND employees age between 20 and 30 (2 Marks)

```
>>select name,age from employee where age between 20 and 30;
```

Q. Write a function to return name,emp_id,dept_name,hobbies,age by passing manager name (10 Marks)

```
>>create or replace function emp(man character varying) returns table
(name character varying ,emp_id integer ,dept_name character varying ,
 hobbies character varying ,age integer )
language plpgsql
as $$
begin
return query
select e.name,e.emp_id,d.dept_name,e.hobbies,e.age from employee e join dept d
on e.emp_id=d.e_id where manager=man;
end;
```

```
$$;
```

Q.CREATE MONGO DB COLLECTIONS with following details and insert data --DB = mongo exam --Collection = assignment,inventory

```
Use mongoexam
db.createCollection('assignment')
db.createCollection('inventory')
```

```
db.assignment.insertMany( [{ item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } }, { item: "mat", qty: 85, tags: ["gray"],size: { h: 27.9, w: 35.5, uom: "cm" } }, { item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85,uom: "cm" } }])
```

```
db.inventory.insertMany([{ item: "journal", qty: 25, tags: ["blank", "red"], dim_cm: [ 14, 21 ] }, { item: "notebook", qty: 50, tags: ["red", "blank"],dim_cm: [ 14, 21 ] }, { item: "paper", qty: 100, tags: ["red", "blank", "plain"], dim_cm: [ 14, 21 ] }, { item: "planner", qty:75, tags: ["blank", "red"], dim_cm: [ 22.85, 30 ] }, { item: "postcard", qty: 45, tags: ["blue"], dim_cm: [ 10, 15.25 ] } ])
```

Q.get assignment documents having tags = gray (2 Marks)

```
>>db.assignment.find({tags:'gray'})
```

Q.Get inventory details whose dim_cm > 10 , sorted by qty descending order and print only 3 dcouments.(2 Marks)

```
>>db.inventory.find({dim_cm:{$gt:10}}).sort({qty:-1}).limit(3)
```

Q.Create index on inventory in descending order of qty .(2 Marks)

```
>>db.inventory.createIndex({"qty":-1})
```

Q.Querty to aggregare sum of qty in inventory collection

```
>>db.inventory.aggregate({$sum:"qty"})
```

Q.query to update inventory collection item name where qty:75 and dim_cm > 22. (2 Marks

```
>>db.inventory.updateMany({$and:[{qty:75},{dim_cm:{$gt:22}}]},{$set:{item:"name"}})
```