**CRUD Operations:**

- **Create a table employee having id,name,salary,designation(designation examples A,B & C)**

```
practise=# create table employee(id int,name varchar(10),salary float,designation varchar(10));
CREATE TABLE
```

○ **Insert valid data into the Employee table**

```
practise=# insert into employee values(1,'Karthik',11000,'A'),(2,'Ankith',9000,'A'),
(3,'Rahul',5000,'B'),(4,'Rohan',12000,'C'),(5,'Chaitanya',2000,'C'),(6,'Sowmya',9000,'B');
INSERT 0 6
```

○ **Update the salary of id=1**

```
practise=# update employee set salary=13000 where id=1;
UPDATE 1
practise=# select*from employee;
 id |   name    | salary | designation
----+-----------+--------+-------------
  2 | Ankith    |   9000 | A
  3 | Rahul     |   5000 | B
  4 | Rohan     |  12000 | C
  5 | Chaitanya |   2000 | C
  6 | Sowmya    |   9000 | B
  1 | Karthik   |  13000 | A
(6 rows)
```

○ **Select the employees who are having salary > 8000 and name starts with the letter ‘A’**

```
practise=# select*from employee where salary>8000 and name like 'A%';
 id |  name  | salary | designation
----+--------+--------+-------------
  2 | Ankith |   9000 | A
(1 row)
```

○ **Delete the employees belongs to Designation “A & B”**

```
practise=# delete from employee where designation in('A','B');
DELETE 4
practise=# select*from employee;
 id |   name    | salary | designation
----+-----------+--------+------------
  4 | Rohan     |  12000 | C
  5 | Chaitanya |   2000 | C
(2 rows)
```

**Constraints**:

● **Update employee ‘id’ to primary key ,should not be null & Unique**

```
practise=# alter table employee add primary key(id) ;
ALTER TABLE
```

**● Add new column joining date to employee table which should not be null and date should be <= current date**

```
practise=# alter table employee add joining_date date check(joining_date<=current_date);
ALTER TABLE
practise=# alter table employee alter joining_date set not null;
ALTER TABLE
```

**● Update 'designation' to check constraint 'A/B/C' and the length of column should be 2**

```
practise=# alter table employee add check(designation in('A','B','C') and length(designation)=2);
ALTER TABLE
```

**DB Relationships:**

**● Create a table designation with id,name(A,B & C) having "ManyToMany" relation with employee table**

```
practise=# create table designation(d_id int primary key,d_name varchar(10));
CREATE TABLE
practise=# create table emp_designation(e_id int not null unique,foreign key(e_id) references
employee on delete cascade,d_id int not null ,foreign key(d_id) references designation(d_id));
CREATE TABLE
```

**Operations**

**● Select all employees who are having designation 'C'**

```
practise=# select*from employee a inner join emp_designation b on a.id=b.e_id inner join
designation c on b.d_id=c.d_id where a.designation='C';

 id |   name    | salary | designation | joining_date | e_id | d_id | d_id | d_name
----+-----------+--------+-------------+--------------+------+------+------+--------
  4 | Rohan     |  12000 | C           | 2022-08-06   |    4 |    3 |    3 | C
  5 | Chaitanya |   2000 | C           | 2022-08-06   |    5 |    3 |    3 | C
(2 rows)
```

**● Select all designations for the employees whose salary < 10000**

```
practise=#  select * from employee a inner join emp_designation b on a.id=b.e_id inner join
designation c on  b.d_id=c.d_id where salary<10000;
 id |   name    | salary | designation | joining_date | e_id | d_id | d_id | d_name
----+-----------+--------+-------------+--------------+------+------+------+--------
  2 | Ankith    |   9000 | A           | 2022-08-06   |    2 |    1 |    1 | A
  3 | Rahul     |   5000 | B           | 2022-08-06   |    3 |    2 |    2 | B
  6 | Sowmya    |   9000 | B           | 2022-08-05   |    6 |    2 |    2 | B
  5 | Chaitanya |   2000 | C           | 2022-08-06   |    5 |    3 |    3 | C
  (4 rows)
```

**● Insert a new employee with designation ‘A & B’**

```
practise=# insert into employee values(7,'sameera',9000,'A','2022-08-06',1);
INSERT 0 1
practise=# insert into employee values(8,'sailaja',8000,'B','2022-08-05',2);
INSERT 0 1
```

**● Delete the Designation ‘C’ which should update the existing employees with this designation**

**● Delete the employee whose id = 2(Should update the designation dependency)**

**● Create a table Address(id,name) having “OneToOne” relation with Employee table**

```
practise=# create table address(a_id int primary key,name varchar(10),e_id int unique not
null ,foreign key(e_id) references employee(id));
CREATE TABLE
```

**Operations**

**● Insert address(Hyderabad) for employee whose id=3**

```
practise=# insert into address values(1,'Hyderabad',3);
INSERT 0 1

practise=# insert into address(a_id,name,e_id) values (1,'Hyderabad',3) on conflict (e_id) do update
set  name=excluded.name;
INSERT 0 1
```

**● Update the address (Mumbai) of Employee whose id=3**

```
ractise=# update address set a_name='Mumbai' where e_id=3;
UPDATE 1
```

**● Select all employee names & Designations whose address is equals (ignorecase) to”Hyderabad”**

```
practise=# select*from employee join address on employee.id=address.e_id where
a_name='Hyderabad';
 id |  name   | salary | designation | joining_date | d_id | a_id |  a_name   | e_id
----+---------+--------+-------------+--------------+------+------+-----------+------
  1 | Karthik |  13000 | A           | 2022-08-05   |    1 |    2 | Hyderabad |    1
  7 | sameera |   9000 | A           | 2022-08-06   |    1 |    6 | Hyderabad |    7
(2 rows)
```

**● Delete the employees whose address is ‘MUMBAI’**

**Functions**:

● **Get the number of employees existed in Employee table**

```
practise=# select count(*) from employee ;
 count
-------
     8
(1 row)
```

● **Get Maximum , Minimum Salary, Name, Address & Designation of the employees from Employee table,**

```
practise=# select max(salary),min(salary),name,a_name,designation from employee join address on
employee.id=address.e_id group by name,a_name,designation;
  max  |  min  |   name    |  a_name   | designation
-------+-------+-----------+-----------+-------------
 12000 | 12000 | Rohan     | Pune      | C
 13000 | 13000 | Karthik   | Hyderabad | A
  2000 |  2000 | Chaitanya | Delhi     | C
  9000 |  9000 | sameera   | Hyderabad | A
  8000 |  8000 | sailaja   | Pune      | B
  9000 |  9000 | Ankith    | Mumbai    | A
  9000 |  9000 | Sowmya    | Delhi     | B
  5000 |  5000 | Rahul     | Mumbai    | B
(8 rows)
```

● **Get the employee Name, Address & Designation whose name length is greater among the employees**

● **Get all employee names in alphabetical order and in name should converted to Uppercase**

```
practise=# select upper(name) name from employee order by name asc;
   name
-----------
 ANKITH
 CHAITANYA
 KARTHIK
 RAHUL
 ROHAN
 SAILAJA
 SAMEERA
 SOWMYA
(8 rows)
```

● **Get sum of employee salaries whose designation is ‘A’ & Address is ‘Hyderabad’**

```
practise=# select sum(salary),designation,a_name address from employee join address on
employee.id=address.e_id where designation ='A' and a_name='Hyderabad' group by
designation,a_name;
  sum  | designation |  address
-------+-------------+-----------
 22000 | A           | Hyderabad
(1 row)
```

**2.Read the following and get the DB structure as per the use case(Create DB structure with correct tables, columns, relationships and constraints)**

**Relationship: one to many**

```
practise=# create table bus_reservation_system(b_id int primary key,b_name
varchar(20),arrival_time time);
CREATE TABLE

practise=# create table bus_tickets_reservations(r_id int primary
key,ticket_no int,date_of_journey date,b_id int not null,foreign key(b_id)
references bus_reservation_system(b_id));
CREATE TABLE


practise=# create table users(u_id int primary key,user_name varchar(20),r_id
int ,foreign key(r_id) references bus_tickets_reservations(r_id));
CREATE TABLE
```

**a.Bus Ticket reservation system. Users will have ticket reservations.**
**i.Users should be able to see all their reservations details**

```
practise=# select*from users  natural join bus_tickets_reservations  natural
join bus_reservation_system;
 b_id | r_id | u_id | user_name | ticket_no | date_of_journey | arrival_time |
b_name
------+------+------+-----------+-----------+-----------------+--------------+----------------
    1 |    1 |    1 | Ram mohan |        12 | 2022-08-05      | 19:30:00     | Bala
gangadhar
    2 |    3 |    2 | Prakash   |        14 | 2022-08-06      | 16:25:00     | Diwakar
    1 |    2 |    3 | Ajith     |        10 | 2022-08-05      | 19:30:00     | Bala
gangadhar
    2 |    4 |    4 | Naveen    |         2 | 2022-08-06      | 16:25:00     | Diwakar
(4 rows)
```

**ii.Every reservation should get details of the bus, user , date and time of journey**

```
practise=# select*from users  natural join bus_tickets_reservations  natural
join bus_reservation_system;
 b_id | r_id | u_id | user_name | ticket_no | date_of_journey | arrival_time |
b_name
------+------+------+-----------+-----------+-----------------+--------------+----------------
    1 |    1 |    1 | Ram mohan |        12 | 2022-08-05      | 19:30:00     | Bala
gangadhar
    2 |    3 |    2 | Prakash   |        14 | 2022-08-06      | 16:25:00     | Diwakar
    1 |    2 |    3 | Ajith     |        10 | 2022-08-05      | 19:30:00     | Bala
gangadhar
    2 |    4 |    4 | Naveen    |         2 | 2022-08-06      | 16:25:00     | Diwakar
```
**(4 rows)**

**iii.User should be able to see all the reservations of a bus between 2 dates**

**iv.Get total number of reservations for a particular user and bus(combination of user & bus)**

```
practise=# select*from bus_tickets_reservations a inner join users b on
a.r_id=b.r_id where b.u_id=1;
 r_id | ticket_no | b_id | date_of_journey | u_id | user_name | r_id
------+-----------+------+-----------------+------+-----------+------
    1 |        12 |    1 | 2022-08-05      |    1 | Ram mohan |    1
(1 row)
```

**v.Find all the reservations on a particular date**

```
practise=# select*from bus_tickets_reservations where
date_of_journey='2022-08-05';
 r_id | ticket_no | b_id | date_of_journey
------+-----------+------+-----------------
    1 |        12 |    1 | 2022-08-05
    2 |        10 |    1 | 2022-08-05
(2 rows)
```