```
create table Employees(employee_id int primary
key,first_name varchar(50),last_name varchar(50),
hire_date DATE, department_id int, salary decimal(10,2));

describe Employees;

insert into employees (employee_id,first_name,last_name,
hire_date,department_id,salary) values
(1,’Bipul’,’Kumar’,’2022-01-15’,1,60000),
(2,’Rohit’,’Sharma’,’2023-01-17’,1,55000),
(3,’Ajay’,’Devgan’,’2020-01-10’,2,70000),
(4,’Rahul’,’Dravid’,’2021-01-12’,2,75000),
(5,’Pavan’,’Kumar’,’2022-01-10’,3,50000),
(6,’Rajesh’,’Kumar’,’2023-01-19’,3,80000);
```

DQL Statements

```
select ∗ from employees;

select first_name,hire_date,salary from employees;

select ∗ from employees where department_id=1;

select ∗ from employees order by last_name;

select ∗ from employees order by last_name,first_name;

select ∗ from employees order by salary;

select ∗ from employees order by salary desc;
```

SQL Functions

```
select department_id,COUNT(∗) from employees group by
department_id;

select department_id,avg(salary) from employees group by
department_id;

select avg(salary) from employees;

select sum(salary) from employees;

select count(∗) from employees;
select min(salary) from employees;

select max(salary) from employees;

select first_name,last_name from employees;

select concat(first_name,' ',last_name) from employees;

select concat(first_name,' ',last_name) as full_name from
employees;

select first_name,length(first_name) from employees;

select upper(first_name) from employees;

select lower(last_name) from employees;

select substring(last_name,1,3) from employees;
```

Date - Time Functions

```
select now();

select date_format(hire_date , '%Y-%m-%d') from
employees;

select date_format(hire_date , '%y-%m-%d') from
employees;

select date_format(hire_date , '%y-%M-%d') from
employees;

select date_format(hire_date , '%y-%M-%D') from
employees;

select date_format(hire_date , '%d-%m-%Y') from
employees;

select datediff(now(),hire_date) as days_since_hire from
employees;

select * from Employees where salary > (select
avg(salary) from employees);
```

Logical Functions

Case when - perform conditional logic in query

```
select first_name,salary,
    -> case
    -> when salary > 60000 then 'High'
    -> when salary > 40000 then 'Medium'
    -> else 'Low'
    -> end as salary_category
    -> from employees;

select round(salary,2) from employees;
select abs(salary) from employees;
```

DML Statements

```
select * from Employees;

update employees set salary = 78000 where employee_id=3;

delete from employees where employee_id = 5;

update employees set full_name = concat(first_name,' ' ,
last_name),
    -> location_id=107
    -> where employee_id=7;


*********************************************************
**--Concat first_name & last_name into full_name add this
full_name into a new column full_name-**

#add a column full_name into employees
alter table employees add column full_name varchar(100);

#update the column values with full name data
update employees set full_name = concat(first_name , '
' , last_name);

Select * from employees;
```

```
update employees set location_id = case when employee_id
= 1 then 101 when employee_id = 2 then 102 when
employee_id = 3 then 103 when employee_id = 4 then 104
when employee_id = 5 then 105 when employee_id = 6 then
106 end;


update employees set location_id = v.location_id
from (values
       (1,201),
       (2,202),
       (3,303),
       (4,404),
       (5,505),
       (6,606)) as v(employee_id, location_id)
Where employees.employee_id = v.employee_id
```

Create a new data table after ordering existing one based on salary

```
create table ordered_employee as select * from employees
order by salary;
```

Create a new data table after assigning High, Medium or Low to each sample based on salary

```
create table salary_cat_table as select
first_name,salary,
    -> case
    -> when salary > 60000 then 'High'
    -> when salary > 40000 then 'Medium'
    -> else 'Low'
    -> end as salary_category
    -> from employees;
```