**Assignment Questions**

**Theoretical Questions**

1. What is the purpose of aggregate functions in SQL? Provide examples.

Aggregate functions perform calculations on a group of values and return a single result.

Sum(),max(),min(),avg()

1. Explain the difference between INNER JOIN and FULL OUTER JOIN.

INNER JOIN returns only matching records from both tables.

FULL OUTER JOIN returns all records from both tables, including unmatched ones (with NULL where no match exists).

1. What are scalar functions, and how are they different from aggregate functions?

Scalar functions: operate on single value and returns single value (len(),upper())

Aggregate functions: operates on multiple value and returns single result (sum(),avg())

1. How does a Primary Key differ from a Foreign Key?

Primary Key: Uniquely identifies a row in a table.

Foreign Key: Refers to the primary key in another table to establish a relationship.

1. Why would you use LEFT JOIN in a query?

To retrieve all the rows from left table and matching rows from right table

1. Explain the purpose of the ROUND() and ABS() functions with examples.

Round() 🡺 Rounds a number to a specified number of decimal places.

Abs() 🡺 Returns the absolute value of a number.

1. What is the use of SUBSTRING() in SQL?

Substring(name,start,stop)🡺 used to extract the portion of string

1. Describe the differences between CURDATE() and NOW().

Curdate() 🡺 returns only date (dd-mm-yyyy)

Now()🡺 returns both date and time (yyyy-mm-dd hh:mm:ss)

1. What is the significance of relationships in databases?

Relationships in databases **connect tables** to avoid duplicate data and keep information organized. They help in **retrieving related data easily** and **maintain consistency**.

10.How does CROSS JOIN differ from other types of joins?

CROSS JOIN creates all possible combinations of rows from both tables (cartesian product), while other joins match rows based on a common condition.

**Practical Questions**

1. Write a query to find the total salary of all employees in the Employees table.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int default 50000,

joindate date default (current\_date)

);

Insert into Employees values(1,"alice",30,"HR",default,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",default,"2022-05-20");

insert into Employees values(3,"charlie",28,"Finance",default,"2021-10-10");

select sum(sal) from Employees;

Output:

+----------+

| sum(sal) |

+----------+

| 150000 |

+----------+

1. Retrieve the names of employees who belong to the Sales department using an INNER JOIN.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR","2023-01-15");

insert into Employees values(2,"bob",25,"IT","2022-05-20");

insert into Employees values(3,"charlie",28,"sales","2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select Employees.empid,Employees.name,Departments.departmentname from Employees inner join Departments on Employees.empid=Departments.departmentid where Departments.departmentname="sales";

Output:

+-------+---------+----------------+

| empid | name | departmentname |

+-------+---------+----------------+

| 3 | charlie | sales |

+-------+---------+----------------+

1. Add a new column Bonus to the Employees table and update it with 10% of each employee’s salary.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int default 50000,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",default,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",default,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",default,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

alter table Employees add (bonus float);

update Employees set bonus=sal\*0.10;

select \* from Employees;

Output:

+-------+---------+------+-------+-------+------------+-------+

| empid | name | age | dept | sal | joindate | bonus |

+-------+---------+------+-------+-------+------------+-------+

| 1 | alice | 30 | HR | 50000 | 2023-01-15 | 5000 |

| 2 | bob | 25 | IT | 50000 | 2022-05-20 | 5000 |

| 3 | charlie | 28 | sales | 50000 | 2021-10-10 | 5000 |

+-------+---------+------+-------+-------+------------+-------+

1. Find the highest and lowest salaries from the Employees table.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",50000,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",70000,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",20000,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select max(sal) as max\_sal,min(sal) as min\_sal from Employees;

Output:

+---------+---------+

| max\_sal | min\_sal |

+---------+---------+

| 70000 | 20000 |

+---------+---------+

1. Write a query to extract the first three characters of each employee’s name.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",50000,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",70000,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",20000,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select empid,name,substring(name,1,3) as first\_threechar from Employees;

Output:

+-------+---------+-----------------+

| empid | name | first\_threechar |

+-------+---------+-----------------+

| 1 | alice | ali |

| 2 | bob | bob |

| 3 | charlie | cha |

+-------+---------+-----------------+

1. Retrieve employees who joined after 2022-01-01 using the WHERE clause.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",50000,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",70000,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",20000,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select \* from Employees where joindate > "2022-01-01";

Output:

+-------+-------+------+------+-------+------------+

| empid | name | age | dept | sal | joindate |

+-------+-------+------+------+-------+------------+

| 1 | alice | 30 | HR | 50000 | 2023-01-15 |

| 2 | bob | 25 | IT | 70000 | 2022-05-20 |

+-------+-------+------+------+-------+------------+

1. Create a query to format the JoiningDate column as DD-MM-YYYY in the result set.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",50000,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",70000,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",20000,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select date\_format(joindate,'%d-%m-%y') as join\_date from Employees;

Output:

+-----------+

| join\_date |

+-----------+

| 15-01-23 |

| 20-05-22 |

| 10-10-21 |

+-----------+

1. Write a query to calculate the average age of employees in each department.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",50000,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",70000,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",20000,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select dept,avg(age) from Employees group by dept;

Output:

+-------+----------+

| dept | avg(age) |

+-------+----------+

| HR | 30.0000 |

| IT | 25.0000 |

| sales | 28.0000 |

+-------+----------+

1. List all employees and their corresponding departments, including employees without a department.

QUERY:

create table Employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

joindate date default (current\_date)

);

create table Departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into Employees values(1,"alice",30,"HR",50000,"2023-01-15");

insert into Employees values(2,"bob",25,"IT",70000,"2022-05-20");

insert into Employees values(3,"charlie",28,"sales",20000,"2021-10-10");

insert into Departments values(1,"HR");

insert into Departments values(2,"IT");

insert into Departments values(3,"sales");

select Employees.\* ,Departments.departmentname from Employees left join Departments on Employees.dept=Departments.departmentname;

Output:

+-------+---------+------+-------+-------+------------+----------------+

| empid | name | age | dept | sal | joindate | departmentname |

+-------+---------+------+-------+-------+------------+----------------+

| 1 | alice | 30 | HR | 50000 | 2023-01-15 | HR |

| 2 | bob | 25 | IT | 70000 | 2022-05-20 | IT |

| 3 | charlie | 28 | sales | 20000 | 2021-10-10 | sales |

+-------+---------+------+-------+-------+------------+----------------+

10.Create a SELF JOIN query to find all pairs of employees who report to the same manager.

QUERY:

create table employees(

empid int primary key,

name varchar(100) not null,

age int check(age > 18),

dept varchar(50),

sal int,

mgrid int,

joindate date default (current\_date)

);

create table departments (

departmentid int primary key,

departmentname varchar(50) unique

);

insert into employees values(1,"alice",30,"hr",50000,1,"2023-01-15");

insert into employees values(2,"bob",25,"it",70000,2,"2022-05-20");

insert into employees values(3,"charlie",28,"sales",20000,1,"2021-10-10");

insert into departments values(1,"hr");

insert into departments values(2,"it");

insert into departments values(3,"sales");

select e1.name as employee1, e2.name as employee2, e1.mgrid from employees e1 join employees e2 on e1.mgrid= e2.mgrid and e1.empid <> e2.empid;

Output:

+-----------+-----------+-------+

| Employee1 | Employee2 | mgrid |

+-----------+-----------+-------+

| charlie | alice | 1 |

| alice | charlie | 1 |

+-----------+-----------+-------+