PRATIXA MISTRY

MODULE 4_PART_2 DATABASE

1. What is an SQL alias?

SQL aliases are used to give a table, or a column in a table, a temporary name.

Aliases are often used to make column names more readable.

An alias only exists for the duration of that query.

An alias is created with the AS keyword.

2. Write a query to create the table in Structured Query Language.

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

```
Ex: CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

3. Write a query to insert data into table.

```
INSERT INTO table_name (column1, column2, column3,etc)
VALUES
    (value1, value2, value3, etc),
    (value1, value2, value3, etc),
```

(value1, value2, value3, etc);

4. **Write a query to update data into table with validations.**

UPDATE *table_name*
SET *column1 = value1*, *column2 = value2*, ...
WHERE *condition*;

5. **Write a query to delete data from table with validations.**

DELETE FROM *table_name* WHERE *condition*;

6. **Write a query to insert new column in existing table.**

ALTER TABLE *table_name*
ADD *column_name datatype*;

Ex: ALTER TABLE Customers
ADD Email varchar(255);

7. **Write a query to drop table and database.**

DROP TABLE *table_name*;

Ex: DROP TABLE Shippers;

DROP DATABASE *databasename*;

Ex: DROP DATABASE testDB;

8. **Write a query to find max and min value from table.**

SELECT MAX(*column_name*)
FROM *table_name*
WHERE *condition*;

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

9. **Create two tables named Seller and Product apply foreign key in product table Fetch data from both table using different joins.**

```
-- Creating the Seller table
CREATE TABLE Seller (
    SellerID INT PRIMARY KEY,
    SellerName VARCHAR(100) NOT NULL,
    ContactNumber VARCHAR(15),
    Email VARCHAR(100)
);

-- Creating the Product table with a foreign key referencing Seller
CREATE TABLE Product (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100) NOT NULL,
    Price DECIMAL(10, 2),
    SellerID INT,
    FOREIGN KEY (SellerID) REFERENCES Seller(SellerID)
);

-- Example data insertion into Seller table
INSERT INTO Seller (SellerID, SellerName, ContactNumber, Email)
```

```
VALUES
(1, 'Alice', '123-456-7890', 'alice@example.com'),
(2, 'Bob', '987-654-3210', 'bob@example.com'),
(3, 'Charlie', '555-666-7777', 'charlie@example.com');

-- Example data insertion into Product table
INSERT INTO Product (ProductID, ProductName, Price, SellerID)
VALUES
(101, 'Laptop', 1200.00, 1),
(102, 'Smartphone', 800.00, 2),
(103, 'Tablet', 400.00, 1),
(104, 'Headphones', 100.00, 3),
(105, 'Smartwatch', 200.00, NULL);

-- Fetching data using INNER JOIN
SELECT
    p.ProductID,
    p.ProductName,
    p.Price,
    s.SellerName,
    s.Email
FROM
    Product p
```

```
INNER JOIN
    Seller s
ON
    p.SellerID = s.SellerID;

-- Fetching data using LEFT JOIN
SELECT
    p.ProductID,
    p.ProductName,
    p.Price,
    s.SellerName,
    s.Email
FROM
    Product p
LEFT JOIN
    Seller s
ON
    p.SellerID = s.SellerID;

-- Fetching data using RIGHT JOIN
SELECT
    p.ProductID,
    p.ProductName,
```

```
    p.Price,
    s.SellerName,
    s.Email
FROM
    Product p
RIGHT JOIN
    Seller s
ON
    p.SellerID = s.SellerID;
```