ASHRAF SHAIKH ASSIGNMENT

Aim: Design and Develop SQL DDL statements which demonstrate the use of
SQL objects such as Table, View, Index, Sequence, Synonym
Problem Statement:

1. Create table Customers with schema (cust_id, cust_name, product, quantity, total_price)
->

```
 CREATE TABLE Customers(cust_id int primary key auto_increment ,cust_name varchar(40), product varchar(25), quantity int , total_price float);
```

2. Use sequence/ auto-increment for incrementing customer ID and Insert 5 customer records to the table Customers
->

```
INSERT INTO Customers(cust_name,product,quantity,total_price)
VALUES('Ashraf','Laptop',5,50000),('Shoaib','Mobile',10,10000),('Aman','Mouse',10,500),('Eoin','Keyboard',15,1000),('Alex','Speakers',10,1000);
```

3. Alter the table Customers by adding one column 'price_per_qnty'
->

```
ALTER TABLE Customers ADD price_per_quantity float;
```

4. Create view Cust_View' on Customers displaying customer ID, customer name
->

```
CREATE VIEW Cust_View as SELECT cust_id,cust_name FROM Customers;

 SELECT * FROM Cust_View;

+---------+-----------+
| cust_id | cust_name |
+---------+-----------+
|       1 | Ashraf    |
|       2 | Shoaib    |
|       3 | Aman      |
|       4 | Eoin      |
|       5 | Alex      |
+---------+-----------+
```

5. Update the view 'Cust_View' to display customer ID, product, total price

->

```
CREATE OR REPLACE VIEW Cust_View as SELECT cust_id,product,total_price from Customers;

SELECT * FROM Cust_View;
+---------+----------+-------------+
| cust_id | product  | total_price |
+---------+----------+-------------+
|       1 | Laptop   |       50000 |
|       2 | Mobile   |       10000 |
|       3 | Mouse    |         500 |
|       4 | Keyboard |        1000 |
|       5 | Speakers |        1000 |
+---------+----------+-------------+
```

6. Drop the view 'Cust_View

->

```
 DROP VIEW Cust_View;
```

7. Create index Cust_index' on customer name

->

```
CREATE INDEX Cust_index on Customers (cust_name);
```

8. Drop index 'Cust_index

->

```
 ALTER TABLE Customers DROP INDEX Cust_index;
```

9. Use sequence/ auto-increment for incrementing customer ID

->

```
CREATE TABLE Customers(cust_id int primary key auto_increment ,cust_name varchar(40), product
varchar(25), quantity int , total_price float);
```

10. Use the name alias for table Customers (rename the table in query)

->

```
SELECT * FROM Customers as Customer_Information;
```

11. Drop the table Customers

-> `DROP TABLE Customers;`

```
================================================================================
================================================================================
```

Aim: Design at least 10 SQL queries for suitable database application using SQL DML statements: all types of Join, Sub-Query.

Problem Statement:

1. Create table Customers with schema (ID, name, age, address, salary)

->

```
CREATE TABLE Customers(ID int primary key, name varchar(45), age int , address varchar(40), salary float);
```

2. Create table Orders with Schema(0_ID, o_date, customer_id, amount)

->

```
 CREATE TABLE Orders(O_ID int primary key, o_date date, customer_id int, amount float);
```

3. Insert 5 records to each table keeping few customer ids common to both the tables

->

```
INSERT INTO Customers
VALUES(1,'Ashraf',22,'Jalgaon',45000),(2,'Aman',21,'Jalgaon',40000),(3,'Eoin',23,'Pune',10000),(4,'Sam',22,'Mumbai',5000) ,(5,'Alex',20,'Pune',50000);

SELECT * FROM Customers;
+----+--------+------+---------+--------+
| ID | name   | age  | address | salary |
+----+--------+------+---------+--------+
|  1 | Ashraf |   24 | Jalgaon |  45000 |
|  2 | Aman   |   26 | Jalgaon |  40000 |
|  3 | Eoin   |   24 | Pune    |  10000 |
|  4 | Sam    |   24 | Mumbai  |   5000 |
|  5 | Alex   |   26 | Pune    |  50000 |
+----+--------+------+---------+--------+

INSERT INTO Orders
VALUES(101,'2022-08-08',1,1000),(102,'2022-08-08',2,500),(103,'2022-01-08',1,1000),(104,'2022-02-08',3,800),(105,'2022-03-08',4,400);

SELECT * FROM Orders;
+------+------------+-------------+--------+
| O_ID | o_date     | customer_id | amount |
+------+------------+-------------+--------+
|  101 | 2022-08-08 |           1 |   1000 |
|  102 | 2022-08-08 |           2 |    500 |
|  103 | 2022-01-08 |           1 |   1000 |
|  104 | 2022-02-08 |           3 |    800 |
|  105 | 2022-03-08 |           4 |    400 |
+------+------------+-------------+--------+
```

4. Perform the inner join on customers and orders table to enlist the id, name, amount and O_date

->

```
SELECT Orders.O_ID,Customers.name,Orders.amount,Orders.o_date FROM Orders INNER JOIN
Customers ON Orders.customer_id=Customers.ID;

+------+--------+--------+------------+
| O_ID | name   | amount | o_date     |
+------+--------+--------+------------+
|  101 | Ashraf |   1000 | 2022-08-08 |
|  102 | Aman   |    500 | 2022-08-08 |
|  103 | Ashraf |   1000 | 2022-01-08 |
|  104 | Eoin   |    800 | 2022-02-08 |
|  105 | Sam    |    400 | 2022-03-08 |
+------+--------+--------+------------+
```

5. Perform the left outer join on customers and orders table to enlist the id, name, amount and o_date

->

```
SELECT Orders.O_ID,Customers.name,Orders.amount,Orders.o_date FROM Orders LEFT JOIN
Customers ON Orders.customer_id=Customers.ID;
+------+--------+--------+------------+
| O_ID | name   | amount | o_date     |
+------+--------+--------+------------+
|  101 | Ashraf |   1000 | 2022-08-08 |
|  102 | Aman   |    500 | 2022-08-08 |
|  103 | Ashraf |   1000 | 2022-01-08 |
|  104 | Eoin   |    800 | 2022-02-08 |
|  105 | Sam    |    400 | 2022-03-08 |
+------+--------+--------+------------+
```

6. Perform the right outer join on customers and orders table to enlist the id, name, amount and o_date

->

```
 SELECT Orders.O_ID,Customers.name,Orders.amount,Orders.o_date FROM Orders RIGHT JOIN
Customers ON Orders.customer_id=Customers.ID;
+------+--------+--------+------------+
| O_ID | name   | amount | o_date     |
+------+--------+--------+------------+
|  103 | Ashraf |   1000 | 2022-01-08 |
|  101 | Ashraf |   1000 | 2022-08-08 |
|  102 | Aman   |    500 | 2022-08-08 |
|  104 | Eoin   |    800 | 2022-02-08 |
|  105 | Sam    |    400 | 2022-03-08 |
| NULL | Alex   |   NULL | NULL       |
+------+--------+--------+------------+
```

7. Perform the full outer join on customers and orders table to enlist the id, name, amount and o_date by using 'union all' set operation
->

```
SELECT Orders.O_ID,Customers.name,Orders.amount,Orders.o_date FROM Orders RIGHT JOIN
Customers ON Orders.customer_id=Customers.ID UNION ALL SELECT
Orders.O_ID,Customers.name,Orders.amount,Orders.o_date FROM Orders LEFT JOIN Customers ON
Orders.customer_id=Customers.ID;
```

```
+------+--------+--------+------------+
| O_ID | name   | amount | o_date     |
+------+--------+--------+------------+
|  103 | Ashraf |   1000 | 2022-01-08 |
|  101 | Ashraf |   1000 | 2022-08-08 |
|  102 | Aman   |    500 | 2022-08-08 |
|  104 | Eoin   |    800 | 2022-02-08 |
|  105 | Sam    |    400 | 2022-03-08 |
| NULL | Alex   |   NULL | NULL       |
|  101 | Ashraf |   1000 | 2022-08-08 |
|  102 | Aman   |    500 | 2022-08-08 |
|  103 | Ashraf |   1000 | 2022-01-08 |
|  104 | Eoin   |    800 | 2022-02-08 |
|  105 | Sam    |    400 | 2022-03-08 |
+------+--------+--------+------------+
```

8. Perform the self join on customers table to enlist the pair of customers belonging to same Address
->

```
SELECT A.name , B.name FROM Customers as A, Customers as B where A.address = B.address;
```

```
+--------+--------+
| name   | name   |
+--------+--------+
| Aman   | Ashraf |
| Ashraf | Ashraf |
| Aman   | Aman   |
| Ashraf | Aman   |
| Alex   | Eoin   |
| Eoin   | Eoin   |
| Sam    | Sam    |
| Alex   | Alex   |
| Eoin   | Alex   |
+--------+--------+
```

9. Perform the Cross/ Cartesian join on customers and orders table to enlist the id, name, amount and o_date
->

```
SELECT Orders.O_ID,Customers.name,Orders.amount,Orders.o_date FROM Customers CROSS JOIN
Orders;
+------+--------+--------+------------+
| O_ID | name   | amount | o_date     |
+------+--------+--------+------------+
|  101 | Alex   |   1000 | 2022-08-08 |
|  101 | Sam    |   1000 | 2022-08-08 |
|  101 | Eoin   |   1000 | 2022-08-08 |
|  101 | Aman   |   1000 | 2022-08-08 |
|  101 | Ashraf |   1000 | 2022-08-08 |
|  102 | Alex   |    500 | 2022-08-08 |
|  102 | Sam    |    500 | 2022-08-08 |
|  102 | Eoin   |    500 | 2022-08-08 |
|  102 | Aman   |    500 | 2022-08-08 |
|  102 | Ashraf |    500 | 2022-08-08 |
|  103 | Alex   |   1000 | 2022-01-08 |
|  103 | Sam    |   1000 | 2022-01-08 |
|  103 | Eoin   |   1000 | 2022-01-08 |
|  103 | Aman   |   1000 | 2022-01-08 |
|  103 | Ashraf |   1000 | 2022-01-08 |
|  104 | Alex   |    800 | 2022-02-08 |
|  104 | Sam    |    800 | 2022-02-08 |
|  104 | Eoin   |    800 | 2022-02-08 |
|  104 | Aman   |    800 | 2022-02-08 |
|  104 | Ashraf |    800 | 2022-02-08 |
|  105 | Alex   |    400 | 2022-03-08 |
|  105 | Sam    |    400 | 2022-03-08 |
|  105 | Eoin   |    400 | 2022-03-08 |
|  105 | Aman   |    400 | 2022-03-08 |
|  105 | Ashraf |    400 | 2022-03-08 |
+------+--------+--------+------------+
```

10. Design the sub query with select statement for displaying all the details of the customers having salary greater than 20000
->

```
SELECT * FROM Customers WHERE (salary>20000);
+----+--------+------+---------+--------+
| ID | name   | age  | address | salary |
+----+--------+------+---------+--------+
|  1 | Ashraf |   24 | Jalgaon |  45000 |
|  2 | Aman   |   26 | Jalgaon |  40000 |
|  5 | Alex   |   26 | Pune    |  50000 |
+----+--------+------+---------+--------+
```

11. Create a backup table- 'cust_bkp' of the table customers by using insert statement with the subquery

->

```
CREATE TABLE cust_bkp(ID int primary key, name varchar(45), age int , address varchar(40), salary float
);

INSERT INTO cust_bkp SELECT * FROM Customers;

SELECT * FROM cust_bkp;
+----+--------+------+---------+--------+
| ID | name   | age  | address | salary |
+----+--------+------+---------+--------+
|  1 | Ashraf |   24 | Jalgaon |  45000 |
|  2 | Aman   |   26 | Jalgaon |  40000 |
|  3 | Eoin   |   24 | Pune    |  10000 |
|  4 | Sam    |   24 | Mumbai  |   5000 |
|  5 | Alex   |   26 | Pune    |  50000 |
+----+--------+------+---------+--------+
```

12. Update the salaries by 10% of all the customers(in customers table) having age greater than or equals to 24 by using subquery with update clause(by using backup table cust_bkp)

->

```
UPDATE Customers SET salary = salary*1.1 WHERE ID IN (SELECT ID FROM Cust_bkp where age >= 24);

SELECT * FROM Customers;
+----+--------+------+---------+--------+
| ID | name   | age  | address | salary |
+----+--------+------+---------+--------+
|  1 | Ashraf |   24 | Jalgaon |  49500 |
|  2 | Aman   |   26 | Jalgaon |  44000 |
|  3 | Eoin   |   24 | Pune    |  11000 |
|  4 | Sam    |   24 | Mumbai  |   5500 |
|  5 | Alex   |   26 | Pune    |  55000 |
+----+--------+------+---------+--------+
```

13. Delete all the customers having age greater than 26 by using delete clause with the Subquery

->

```
DELETE FROM Customers WHERE age > 26;

 select * from Customers;
+----+--------+------+---------+--------+
| ID | name   | age  | address | salary |
+----+--------+------+---------+--------+
|  1 | Ashraf |   24 | Jalgaon |  49500 |
|  2 | Aman   |   26 | Jalgaon |  44000 |
|  3 | Eoin   |   24 | Pune    |  11000 |
|  4 | Sam    |   24 | Mumbai  |   5500 |
|  5 | Alex   |   26 | Pune    |  55000 |
+----+--------+------+---------+--------+
```