Name→ Garvit joshi Registration : 12106692
Course Code → INT 350
Subject Name→ SQL and Data Visualization
Teacher Name → Aman kumar
Set → 09
CA → 01 Roll · No → 21

Ques1 a) Create a database for sales record, the next step involved in creating three tables - "sales", "member", and "menu" - that align with provided ER D.

```
CREATE database sales ;
CREATE TABLE sales (
                customer-id INT,
                order-date DATE,
                product-id INT,
                price FLOAT
              );
CREATE TABLE member (
                  customer-id INT,
                  join-date DATE,
                  menu VARCHAR (255),
              );
CREATE TABLE menu (
                  product-id INT,
                  product-name VARCHAR (255),
                  price FLOAT
              );
```

b) Insert Values into Tables.

```
INSERI INTO sales (customer_id, order_date,
                   product_id, price)
VALUES (1, '2023-08-30', 10.99),
       (2, '2023-08-30', 15.99),
       (3, '2023-08-30', 20.99),
       (4, '2023-08-30', 25.99);

INSERT INTO members (customer_id,
                     join_date, menu)
VALUES  (1, '2022-01-01', 'Gold'),
        (2, '2022-02-01', 'Silver'),
        (3, '2022-03-01', 'Bronze'),
        (4, '2022-04-01', 'Platinum');

INSERT INTO menu (prodect_id, product_
                               name,
                  price)
VALUES  (1, 'Coffee', 2.99);
        (2, 'Tea', 1.99),
        (3, 'Sandwich', 5.99);
        (4, 'Salad', 7.99);
```

Ques 2 Perform the the following queries on above created tables:

(a) What is most purchased item on the menu and how many times it was purchased?

```
SELECT menu.product, COUNT (sales.product
                 _id) AS times_purchased
FROM sales
JOIN menu ON sales.product_id = menu.
                               product_id
```

```
GROUP BY sales.product_id
 ORDER BY time_purchased DESC
LIMIT 1;
```

b) Create a new Column named Status in member TABLE.

```
ALTER TABLE member ADD COLUMN status
 BOOLEAN;
```

c) Set flag bit 'false' in status for customer those have not purchased any product in last month.

```
UPDATE members
SET STATUS = FALSE
WHERE customer_id NOT IN (
          SELECT DISTINCT customer_id
          FROM sales
          WHERE order_date >= DATEADD
                 (month, -1, GETDATE())
          );
```

Ques-3 Write an SQL Query to CREATE the following table with specific requirement & insert five rows in each of the tables.

a) Worker (worker_id PRIMARY KEY, first_name, last name, salary, joining_date, worker_mail)

```
CREATE TABLE Worker (
          worker_id INT PRIMARY KEY,
          first_name VARCHAR (255),
        last_name VARCHAR (255),
        salary FLOAT,
        joining_date DATE,
        worker_mail VARCHAR (255)
);

INSERT INTO Worker (worker_id, first_name,
                    last_name, salary,
                    joining_date, worker_mail)
VALUES (1, ‘john’, ‘Doe”, 5000.00, ‘2022-01-
        01’, ‘johndoe@example.com’),
       (2, ‘jane’, ‘Doe’, 6000.0, ‘2022-02-01’,
        ‘janedeo@example.com’),
       (3, ‘Bob’, ‘Smith’, 5500.0, ‘2022-03-01’,
        ‘bobsmith@example.com’),
       (4, ‘Alice’, ‘Johnson’, 6500.0,
        ‘2022-04-01’, ‘alicejohnson@example
                               .com’),
       (5, ‘David’, ‘Lee’, 7000..0, “2022-05-01
        ‘davidlee@example.com’);
```

b) Reward (worker_id, reward_date, reward_amt)

```
CREATE TABLE Reward (
        worker_id INT,
        reward_date DATE,
        reward_amt FLOAT,
        FOREIGN KEY (worker_id)
           REFRENCES worker (worker_
                               id);
```

```
INSERT INTO Reward ( worker_id, reward_
                               date, reward_amt)
VALUES (1, '2023-01-01', 100.0),
       (2, '2023-02-01', 200.0),
       (3, '2023-03-01', 150.0),
       (4, '2023-04-01', 250.0),
       (5, '2023-05-01', 300.0);
```

c) Sample (worker_id, worker_title)

```
CREATE TABLE Sample (
              worker_id INT,
              worker_title VARCHAR (225),
              FOREIGN KEY (worker_id)
              REFRENCES Worker ( Worker_id))
```

```
INSERT INTO Sample ( worker_id, worker_title
Values (1, 'Manager'),
       ( 2, 'Supervisor'),
       ( 3, 'Associate'),
       ( 4, 'Assistant Manager'),
       ( 5, 'Director');
```

Ques-4 Manipulate above CREATED data :

a) CREate a new Column named Department in WorkerTable.

```
ALTER TABLE WORKER ADD COLUMN
   department VARCHAR (255);
```

b) Set worker_id as foreign key in Reward and Sample table citing Worker Table.

```
ALTER TABLE REWARD ADD FOREIGN KEY
 (worker-id). REFRENCES Worker ( worker_id);

ALTER TABLE SAMPLE ADD FOREIGN KEY
 (worker_id) REFRENCES Worker (worker_id);
```

(C) Add Column Department in Worker Table & ADD Values for Department Column in Worker Table.

```
UPDATE Worker
SET department = 'Sales'
WHERE worker_id IN(1,3,5):
UPDATE Worker
 SET department = 'Marketing'
WHERE worker_id (2,4);
```

(d) Remove worker_mail column from Worker table.

```
 ALTER TABLE Worker DROP COLUMN
   worker_mail;
```

(e) Display only the odd record from the table.

```
SELECT *
FROM Worker
WHERE worker_id % 2 = 1;
```

Ques-5 Write an SQL query to perform the following:

a) To fetch first-name from Worker table in block letter using the alias name as "WORKER_NAME".

```
SELECT UPPER(first_name) AS WORKER_NAME
FROM Worker;
```

b) To display unique value from department in Worker table.

```
SELECT DISTINCT Department
FROM Worker;
```

c) To print first three Character of first_name from Worker table.

```
SELECT SUSTRING (first_name, 1, 3)
from Worker;
```

d) To print the first_name and last_name from Worker table into a single Column Complete_name, separated by a space Character.

```
SELECT CONCAT (first_name, ' ', last_name)
AS complete_name from Workers;
```

e) To print all the details from Reward table arranging reward_date in asceding order.

```
SELECT *
FROM reward
ORDER BY reward_date ASC;
```

Ques-6 Perform function and retrieve the following information from the table :-

(a) To count the number of employee working in a particular department.

(department = "CSE")

```
SELECT COUNT (*)
FROM Worker
WHERE department = 'CSE';
```

b) To fetch number of workers for each department in descending order.

```
SELECT department, COUNT (*) AS
  num_workers
FROM Worker
GROUP BY department
 ORDER BY num_worker DESC;
```

c) Display the Current Date & time.

```
SELECT NOW();
```

d) To show the top n (say 3) records of a table.

```
SELECT *
FROM Worker
LIMIT 3;
```

e) Display the number of workers in each department and segregate them into groups.

```
SELECT department, COUNT (*) AS
num_workers.
FROM worker
GROUP BY department;
```