**1. Database Schema for a customer-sale scenario**

Customer(**Cust id : integer,** cust\_name: string)

Item(**item\_id: integer**, item\_name: string, price: integer)

Sale(**bill\_no: integer**, bill\_data: date, **cust\_id: integer, item\_id: integer**, qty\_sold: integer)

For the above schema, perform the following—

1. **Create Tables**: Write the SQL statements to create the customer, item, and sale tables as described above. Ensure that the sale table has foreign key constraints to reference the customer and item tables.

|  |
| --- |
| CREATE TABLE customer(  cust\_id INT,  cust\_name VARCHAR(255),  PRIMARY KEY(cust\_id)  );    CREATE TABLE item(  item\_id INT,  item\_name VARCHAR(255),  price INT,  PRIMARY KEY(item\_id)  );    CREATE TABLE sale(  bill\_no INT PRIMARY KEY,  bill\_date DATE,  cust\_id INT ,  item\_id INT ,  qty\_sold INT ,  FOREIGN KEY (cust\_id) REFERENCES customer (cust\_id),  FOREIGN KEY (item\_id) REFERENCES item(item\_id)); |

**2. Insert Data**: Insert 5 records into each of the tables with the given data.

**customer:**

|  |  |
| --- | --- |
| **cust\_id** | **cust\_name** |
| 1 | John Doe |
| 2 | Jane Smith |
| 3 | Alice Johnson |
| 4 | Bob Brown |
| 5 | Charlie Davis |

**item:**

|  |  |  |
| --- | --- | --- |
| **item\_id** | **item\_name** | **price** |
| 1 | Laptop | 1000 |
| 2 | Smartphone | 800 |
| 3 | Tablet | 600 |
| 4 | Headphones | 200 |
| 5 | Camera | 100 |

**sale:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **bill\_no** | **bill\_date** | **cust\_id** | **item\_id** | **qty\_sold** |
| 1 | 2023-07-01 | 1 | 1 | 1 |
| 2 | 2023-07-01 | 2 | 2 | 2 |
| 3 | 2023-07-02 | 3 | 3 | 1 |
| 4 | 2023-07-03 | 4 | 4 | 3 |
| 5 | 2023-07-10 | 5 | 5 | 1 |

|  |
| --- |
| INSERT INTO customer VALUES(1,'Jhon Doe');  INSERT INTO customer VALUES(2,'Jane Smith');  INSERT INTO customer VALUES(3,'Alice Jhonson');  INSERT INTO customer VALUES(4,'Bob Brown');  INSERT INTO customer VALUES(5,'Charlie Davis');    INSERT INTO item VALUES(1,'Laptop',1000);  INSERT INTO item VALUES(2,'Smartphone',800);  INSERT INTO item VALUES(3,'Tablet',600);  INSERT INTO item VALUES(4,'Headphones',200);  INSERT INTO item VALUES(5,'Camera',100);    INSERT INTO sale VALUES(1,'2023-07-01',1,1,1);  INSERT INTo sale VALUES(2,'2023-07-01',2,2,2);  INSERT INTO sale VALUES(3,'2023-07-02',3,3,1);  INSERT INTO sale VALUES(4,'2023-07-03',4,4,3);  INSERT INTO sale VALUES(5,'2023-07-10',5,5,1); |

**3.**List all the bill numbers for the date 2023-07-01 along with the customer names and item IDs. Ensure the column names are bill\_no, cust\_name, and item\_id.

|  |
| --- |
| SELECT s.bill\_no, c.cust\_name, s.item\_id  FROM sale s,customer c  WHERE s.cust\_id = c.cust\_id AND s.bill\_date = '2023-07-01'; |

4. List all the total bill details including bill\_no, bill\_date, cust\_name, item\_name, price, qty\_sold, and the final amount paid for each item in each transaction ( price \* qty\_sold ).

|  |
| --- |
| SELECT s.bill\_no, s.bill\_date, c.cust\_name, i.item\_name, i.price, s.qty\_sold, (i.price \* s.qty\_sold)  AS amount  FROM sale s,customer c,item i  WHERE s.cust\_id = c.cust\_id AND s.item\_id = i.item\_id; |

**5.** List the details (names) of the customers who have bought a product that has a price greater than 200. Ensure the column name is cust\_name.

|  |
| --- |
| SELECT DISTINCT c.cust\_name  FROM sale s, item i, customer c  WHERE s.item\_id = i.item\_id AND s.cust\_id = c.cust\_id AND i.price > 200; |

**6.** Display the total number of products (represented by quantity sold) purchased by each customer, aggregating the quantities for all transactions made by each customer. Ensure the column names are cust\_name and total\_products\_purchased.

|  |
| --- |
| SELECT c.cust\_name, SUM(s.qty\_sold)  AS total\_products\_purchased  FROM sale s,customer c  WHERE s.cust\_id = c.cust\_id  GROUP BY c.cust\_name; |

**7.**Give a list of item names bought by the customer with cust\_id as 5. Ensure the column name is item\_name.

|  |
| --- |
| SELECT i.item\_name  FROM sale s,item i  WHERE s.item\_id = i.item\_id AND s.cust\_id = 5; |

**8.**List the item names that were sold on the date 2023-07-02. Ensure the column name is item\_name.

|  |
| --- |
| SELECT i.item\_name  FROM sale s, item i  WHERE s.item\_id = i.item\_id AND s.bill\_date = '2023-07-02'; |

1. List the bill details including bill\_no, bill\_date, cust\_id, item\_id, price, qty\_sold, and the amount for each sale transaction.

|  |
| --- |
| SELECT s.bill\_no, s.bill\_date, s.cust\_id, s.item\_id, i.price, s.qty\_sold, (i.price \* s.qty\_sold)  AS amount  FROM sale s,item i  WHERE s.item\_id = i.item\_id; |

1. Create a view named bill\_details\_view that lists detailed bill information including bill\_no, bill\_date, cust\_id, item\_id, price, qty\_sold, and the total amount as amount for each sale transaction. Ensure the column names are in this exact order.

|  |
| --- |
| CREATE VIEW bill\_details\_view AS  SELECT s.bill\_no, s.bill\_date, s.cust\_id, s.item\_id, i.price, s.qty\_sold, (i.price \* s.qty\_sold)  AS amount  FROM sale s,item i  WHERE s.item\_id = i.item\_id;  SELECT \* FROM bill\_details\_view; |

1. Create a view named daily\_sales\_view that lists daily sales (total sales amount) from a start date (2023-07-01) to an end date (2023-07-03), grouped by date. Ensure the column names are bill\_date and total\_sales.

|  |
| --- |
| CREATE VIEW daily\_sales\_view AS  SELECT s.bill\_date, SUM(i.price \* s.qty\_sold) AS total\_sales  FROM sale s,item i  WHERE s.item\_id = i.item\_id and s.bill\_date BETWEEN '2023-07-01' AND '2023-07-03'  GROUP BY s.bill\_date;  SELECT \* FROM daily\_sales\_view; |