**FSDM / CPCM – 2023S   
Database Design & SQL   
Student ID:** 901142 **Student Name:** Roshan Shrestha **Practical Exercise 33\_34\_35\_36**   
---------------------------------------------------------------------------------------------------------------------

1. A screenshot of a computer

   Description automatically generated **Creation of Customers\_142 table:**To create the table as given in the question we can use the query below: **CREATE TABLE Customers\_142 (  
   CUSTOMERID NUMBER (10) PRIMARY KEY, FIRSTNAME VARCHAR (50),  
   LASTNAME VARCHAR (50)  
   );**The table structure after executing the table can be visualized as below:  
     
   To insert the data into the table we can use the query below:  
     
   **INSERT INTO Customers\_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (1, 'Sara', 'Davis');   
   INSERT INTO Customers\_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (2, 'Rumi', 'Shah');   
   INSERT INTO Customers\_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (3, 'Paul', 'Johnson');   
   INSERT INTO Customers\_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (4, 'Samuel', 'Martinez');  
   INSERT INTO Customers\_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (5, 'Roshan', 'Shrestha');**

A screenshot of a computer

Description automatically generatedThe output of the above query can be visualized as below:

1. **Creation of Orders\_142 table:**To create the table Orders\_142 we can use the query below:  
     
   **CREATE TABLE Orders\_142 (  
     
   ORDERID NUMBER (10) PRIMARY KEY,  
     
   CUSTOMERID NUMBER (10),  
     
   ORDERDATE DATE,  
     
   ORDERAMOUNT NUMBER (10, 2),  
     
   FOREIGN KEY (CUSTOMERID) REFERENCES Customers\_142(CUSTOMERID)  
     
   );**

The output after executing the above query with the table structure is shown below:

A screenshot of a computer

Description automatically generated  
To insert the data into the table Orders\_142 we can use the query as below:

**INSERT INTO Orders\_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (1, 1, TO\_DATE('01-SEP-16', 'DD-MON-YY'), 10);**

**INSERT INTO Orders\_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (2, 2, TO\_DATE('02-SEP-16', 'DD-MON-YY'), 12.5);**

**INSERT INTO Orders\_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (3, 2, TO\_DATE('03-SEP-16', 'DD-MON-YY'), 18);**

**INSERT INTO Orders\_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (4, 3, TO\_DATE('15-SEP-16', 'DD-MON-YY'), 20);**

**INSERT INTO Orders\_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (5, 4, TO\_DATE('21-NOV-20', 'DD-MON-YY'), 20);**

The data inside the table after executing the above query is as below:

A screenshot of a computer

Description automatically generated

1. **Creation of Refunds\_142 table:**

To create the table with name Refunds\_142 we can use the query below: **CREATE TABLE Refunds\_142 (**

**REFUNDID NUMBER (10) PRIMARY KEY,**

**ORDERID NUMBER (10),**

**REFUNDDATE DATE,**

**REFUNDAMOUNT NUMBER (10, 2),**

**FOREIGN KEY (ORDERID) REFERENCES Orders\_142(ORDERID)**

**);**

The created table structure for Refunds\_142 can be visualized as below:

A screenshot of a computer

Description automatically generatedTo insert the data into the table we can use the query below:

**INSERT INTO Refunds\_142 (REFUNDID, ORDERID, REFUNDDATE, REFUNDAMOUNT) VALUES (1, 1, TO\_DATE('02-SEP-16', 'DD-MON-YY'), 5);**

**INSERT INTO Refunds\_142 (REFUNDID, ORDERID, REFUNDDATE, REFUNDAMOUNT) VALUES (2, 3, TO\_DATE('18-SEP-16', 'DD-MON-YY'), 18);**

A screenshot of a computer

Description automatically generated The inserted data inside the table can be visualized as below:

1. **Find the Details of the Customers and their orders (only the Customers with orders)**To find the details of the customers and their orders we need to join both table, to join the table we can use the query below: **SELECT c.CUSTOMERID, c.FIRSTNAME,**

**c.LASTNAME, o.ORDERID, o.ORDERDATE, o.ORDERAMOUNT**

**FROM Customers\_142 c**

**INNER JOIN Orders\_142 o ON c.CUSTOMERID = o.CUSTOMERID;**

A screenshot of a computer

Description automatically generatedThe output from the above query is as below:

**Why Customer 2 appear twice?  
Answer:** Customer 2 is listed twice in the query output because they have made multiple orders in the Orders\_142 table. In the provided data, Customer 2 placed two distinct orders, each with its own unique ORDERID, ORDERDATE, and ORDERAMOUNT. Consequently, the INNER JOIN operation between the Customers\_142 and Orders\_142 tables, based on the CUSTOMERID column, generates separate rows for each of Customer 2's orders.

**Why Customer 5 not appear in the output?  
Answer:** The reason why Customer 5 is not present in the output is that they have not made any orders in the Orders\_142 table. The INNER JOIN operation considers only rows with matching CUSTOMERID values in both tables. As there are no orders linked to Customer 5 in the Orders\_142 table, no matches are found, leading to their exclusion from the result.

1. **Find the details of the customers with orders and refunds (join all 3 tables)**To find the details of the customers with orders and refunds by joining all tables we can use the query below:  
   **SELECT c.FIRSTNAME, c.LASTNAME,**

**o.ORDERDATE, o.ORDERAMOUNT, r.REFUNDDATE,**

**r.REFUNDAMOUNT FROM Customers\_142 c**

**INNER JOIN Orders\_142 o ON c.CUSTOMERID = o.CUSTOMERID INNER JOIN Refunds\_142 r ON o.ORDERID = r.ORDERID;**

A screenshot of a computer

Description automatically generatedThe output from execution of the above query is shown below:

**Why only Sara and Rumi appear in the output?  
Answer:** In light of the updated query, we execute an INNER JOIN between Customers\_142 and Orders\_142 by matching the CUSTOMERID column (ON c.CUSTOMERID = o.CUSTOMERID). This yields records for CustomerID 1, 2, 3, and 4. Subsequently, we further combine these results with OrderID 1 and 3 from Refunds\_142 through another INNER JOIN (ON o.ORDERID = r.ORDERID). Consequently, the output comprises only Sara and Rumi since they are the exclusive customers with both orders and refunds, and their details align with the specified criteria.

1. **Find the details of the Customers orders and refunds irrespective of whether they place orders or having refunds.**To exhibit the particulars of Customers and their orders, even if refunds are not present, it is necessary to utilize the LEFT JOIN for the Orders\_142 and Refunds\_142 tables. This approach guarantees the inclusion of all records from the Orders\_142 table, and for orders without any refunds, the refund columns will contain NULL values.  
   Below is the SQL query which we can use to get the required data:  
     
   **SELECT c.CUSTOMERID, c.FIRSTNAME, c.LASTNAME, o.ORDERID,  
   o.ORDERDATE,  
   o.ORDERAMOUNT,  
   r.REFUNDID,  
   r.REFUNDDATE,  
   r.REFUNDAMOUNT  
   FROM Customers\_142 c  
   LEFT JOIN Orders\_142 o ON c.CUSTOMERID = o.CUSTOMERID LEFT JOIN Refunds\_142 r ON o.ORDERID = r.ORDERID;**  
   The output from the above query is as below:**A screenshot of a computer

   Description automatically generated**

**Why we see NULL values in certain columns?  
Answer:** The occurrence of NULL values in specific columns can be attributed to the utilization of LEFT JOINs in the SQL query. LEFT JOIN retrieves all records from the left table and corresponding matches from the right table based on the specified condition. In situations where no match is found in the right table for a particular row in the left table, the columns from the right table will hold NULL values in the final output. This implies that customers who have not placed orders will exhibit NULL values in the order-related columns, while those without any refunds will display NULL values in the refund-related columns. Additionally, customers who neither have orders nor refunds will present NULL values in both the order and refund columns. The presence of these NULL values signifies the absence of related records in the right tables for those specific customers.