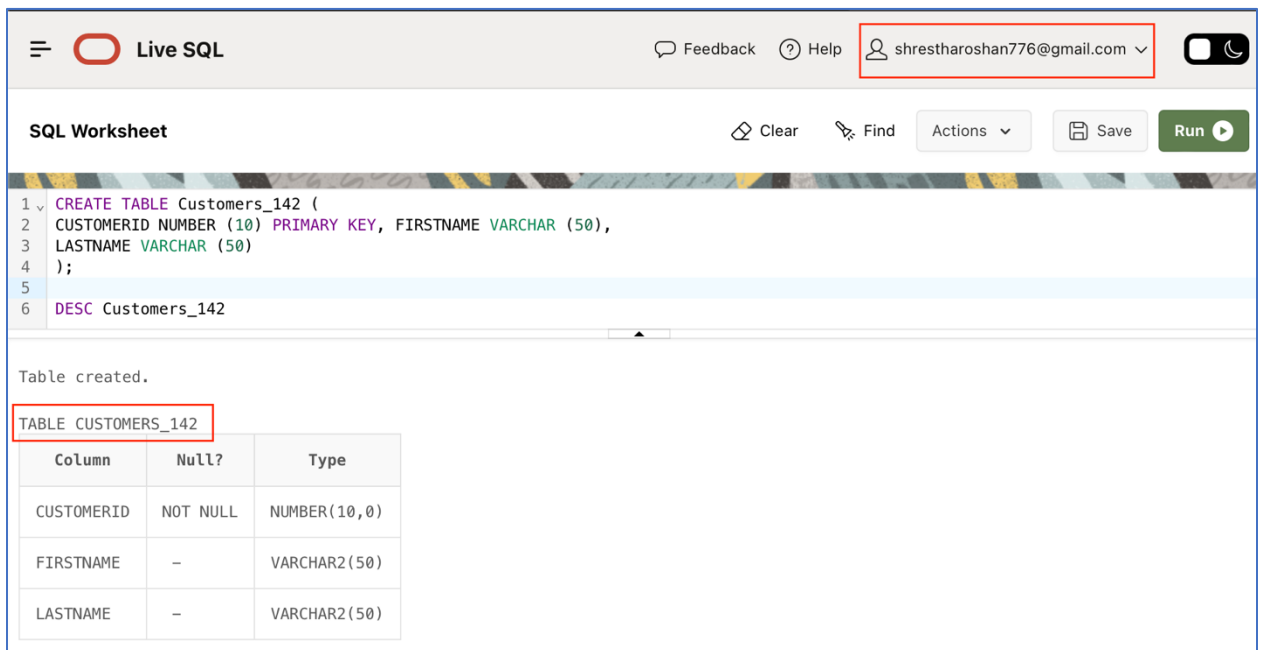


### 1. Creation of Customers\_142 table:

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 screenshot shows the Live SQL interface. The SQL Worksheet contains the following query:

```
1 CREATE TABLE Customers_142 (  
2 CUSTOMERID NUMBER (10) PRIMARY KEY, FIRSTNAME VARCHAR (50),  
3 LASTNAME VARCHAR (50)  
4 );  
5  
6 DESC Customers_142
```

Below the query, a message states "Table created." and a table structure is displayed:

Column	Null?	Type
CUSTOMERID	NOT NULL	NUMBER(10,0)
FIRSTNAME	-	VARCHAR2(50)
LASTNAME	-	VARCHAR2(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');
```

The output of the above query can be visualized as below:

The screenshot shows the Live SQL interface. The top bar includes a menu icon, the 'Live SQL' logo, a 'Feedback' button, a 'Help' button, a user profile dropdown with the email 'shrestharoshan776@gmail.com', and a dark mode toggle. Below the top bar is a 'SQL Worksheet' section with 'Clear', 'Find', 'Actions', 'Save', and 'Run' buttons. The main area contains SQL queries and their results.

```
1 INSERT INTO Customers_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (1, 'Sara', 'Davis');
2 INSERT INTO Customers_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (2, 'Rumi', 'Shah');
3 INSERT INTO Customers_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (3, 'Paul', 'Johnson');
4 INSERT INTO Customers_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (4, 'Samuel', 'Martinez');
5 INSERT INTO Customers_142 (CUSTOMERID, FIRSTNAME, LASTNAME) VALUES (5, 'Roshan', 'Shrestha');
6
7 SELECT * FROM Customers_142
```

The results section shows five '1 row(s) inserted.' messages. Below them is a table with the following data:

CUSTOMERID	FIRSTNAME	LASTNAME
1	Sara	Davis
2	Rumi	Shah
3	Paul	Johnson
4	Samuel	Martinez
5	Roshan	Shrestha

## 2. 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:

Live SQL

Feedback Help shrestharoshan776@gmail.com

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE Orders_142 (  
2 ORDERID NUMBER (10) PRIMARY KEY,  
3 CUSTOMERID NUMBER (10),  
4 ORDERDATE DATE,  
5 ORDERAMOUNT NUMBER (10, 2),  
6 FOREIGN KEY (CUSTOMERID) REFERENCES Customers_142(CUSTOMERID)  
7 );  
8 DESC Orders_142
```

Table created.

TABLE ORDERS\_142

Column	Null?	Type
ORDERID	NOT NULL	NUMBER(10,0)
CUSTOMERID	-	NUMBER(10,0)
ORDERDATE	-	DATE
ORDERAMOUNT	-	NUMBER(10,2)

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:

Feedback
Help
shrestharoshan776@gmail.com

SQL Worksheet
Clear
Find
Actions
Save
Run

```

1 INSERT INTO Orders_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (1, 1, TO_DATE('01-SEP-16', 'DD-MON-YY'), 10);
2 INSERT INTO Orders_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (2, 2, TO_DATE('02-SEP-16', 'DD-MON-YY'), 12.5);
3 INSERT INTO Orders_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (3, 2, TO_DATE('03-SEP-16', 'DD-MON-YY'), 18);
4 INSERT INTO Orders_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (4, 3, TO_DATE('15-SEP-16', 'DD-MON-YY'), 20);
5 INSERT INTO Orders_142 (ORDERID, CUSTOMERID, ORDERDATE, ORDERAMOUNT) VALUES (5, 4, TO_DATE('21-NOV-20', 'DD-MON-YY'), 20);
6 SELECT * FROM Orders_142

```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

ORDERID	CUSTOMERID	ORDERDATE	ORDERAMOUNT
1	1	01-SEP-16	10
2	2	02-SEP-16	12.5
3	2	03-SEP-16	18
4	3	15-SEP-16	20
5	4	21-NOV-20	20

### 3. 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:

Live SQL

Feedback Help shrestharoshan776@gmail.com

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE Refunds_142 (  
2   REFUNDID NUMBER (10) PRIMARY KEY,  
3   ORDERID NUMBER (10),  
4   REFUNDDATE DATE,  
5   REFUNDAMOUNT NUMBER (10, 2),  
6   FOREIGN KEY (ORDERID) REFERENCES Orders_142(ORDERID)  
7 );  
8 DESC Refunds_142
```

Table created.

TABLE REFUNDS\_142

Column	Null?	Type
REFUNDID	NOT NULL	NUMBER(10,0)
ORDERID	-	NUMBER(10,0)
REFUNDDATE	-	DATE
REFUNDAMOUNT	-	NUMBER(10,2)

To 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);**

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

Live SQL

Feedback Help shrestharoshan776@gmail.com

SQL Worksheet

Clear Find Actions Save Run

```
1 INSERT INTO Refunds_142 (REFUNDID, ORDERID, REFUNDDATE, REFUNDAMOUNT) VALUES (1, 1, TO_DATE('02-SEP-16', 'DD-MON-YY'), 5);  
2 INSERT INTO Refunds_142 (REFUNDID, ORDERID, REFUNDDATE, REFUNDAMOUNT) VALUES (2, 3, TO_DATE('18-SEP-16', 'DD-MON-YY'), 18);  
3 SELECT * FROM Refunds_142
```

1 row(s) inserted.  
1 row(s) inserted.

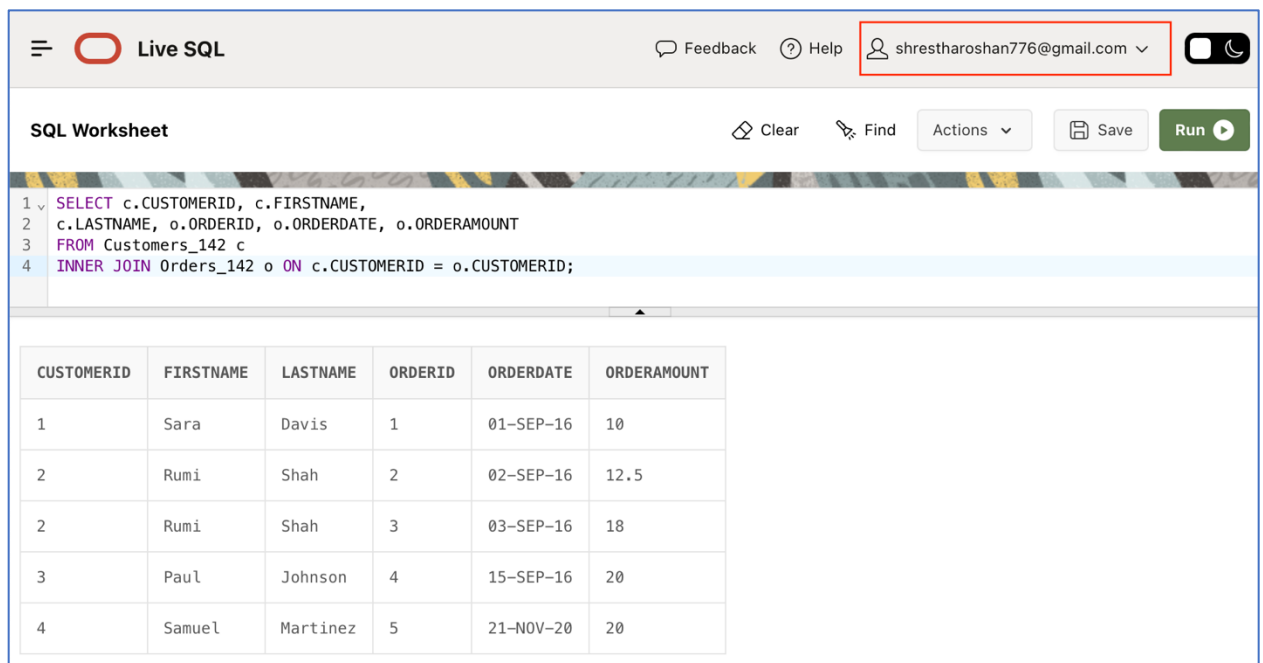
REFUNDID	ORDERID	REFUNDDATE	REFUNDAMOUNT
1	1	02-SEP-16	5
2	3	18-SEP-16	18

4. 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;
```

The output from the above query is as below:



The screenshot shows a web-based SQL editor interface. At the top, there's a header with a menu icon, the text 'Live SQL', and links for 'Feedback' and 'Help'. A user profile icon with the email 'shrestharoshan776@gmail.com' is also visible. Below the header, there's a 'SQL Worksheet' section with buttons for 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The SQL query is entered in a text area, and the results are displayed in a table below. The table has six columns: CUSTOMERID, FIRSTNAME, LASTNAME, ORDERID, ORDERDATE, and ORDERAMOUNT. The data shows five rows, with Customer 2 appearing twice because they have two orders.

CUSTOMERID	FIRSTNAME	LASTNAME	ORDERID	ORDERDATE	ORDERAMOUNT
1	Sara	Davis	1	01-SEP-16	10
2	Rumi	Shah	2	02-SEP-16	12.5
2	Rumi	Shah	3	03-SEP-16	18
3	Paul	Johnson	4	15-SEP-16	20
4	Samuel	Martinez	5	21-NOV-20	20

**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.

**5. 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:

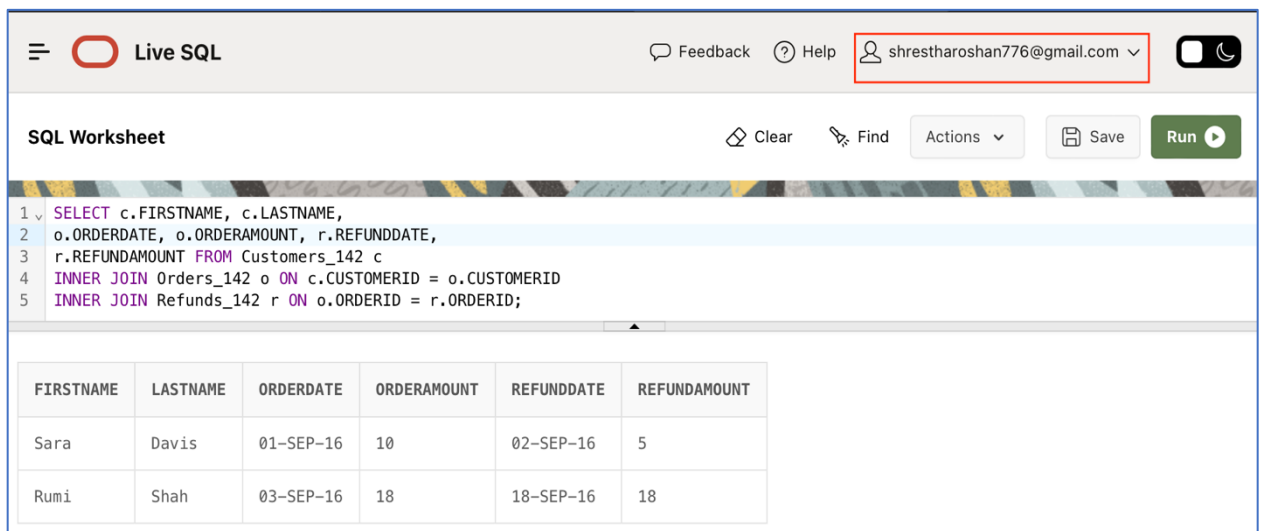
**SELECT c.FIRSTNAME, c.LASTNAME,**

**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;**

The output from execution of the above query is shown below:



The screenshot shows the 'Live SQL' interface. At the top, there's a navigation bar with a menu icon, the text 'Live SQL', and links for 'Feedback', 'Help', and a user profile 'shrestharoshan776@gmail.com'. Below this is a 'SQL Worksheet' section with buttons for 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The SQL query is entered in a text area, and the results are displayed in a table below.

```
1 SELECT c.FIRSTNAME, c.LASTNAME,
2 o.ORDERDATE, o.ORDERAMOUNT, r.REFUNDDATE,
3 r.REFUNDAMOUNT FROM Customers_142 c
4 INNER JOIN Orders_142 o ON c.CUSTOMERID = o.CUSTOMERID
5 INNER JOIN Refunds_142 r ON o.ORDERID = r.ORDERID;
```

FIRSTNAME	LASTNAME	ORDERDATE	ORDERAMOUNT	REFUNDDATE	REFUNDAMOUNT
Sara	Davis	01-SEP-16	10	02-SEP-16	5
Rumi	Shah	03-SEP-16	18	18-SEP-16	18

**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.

**6. 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.

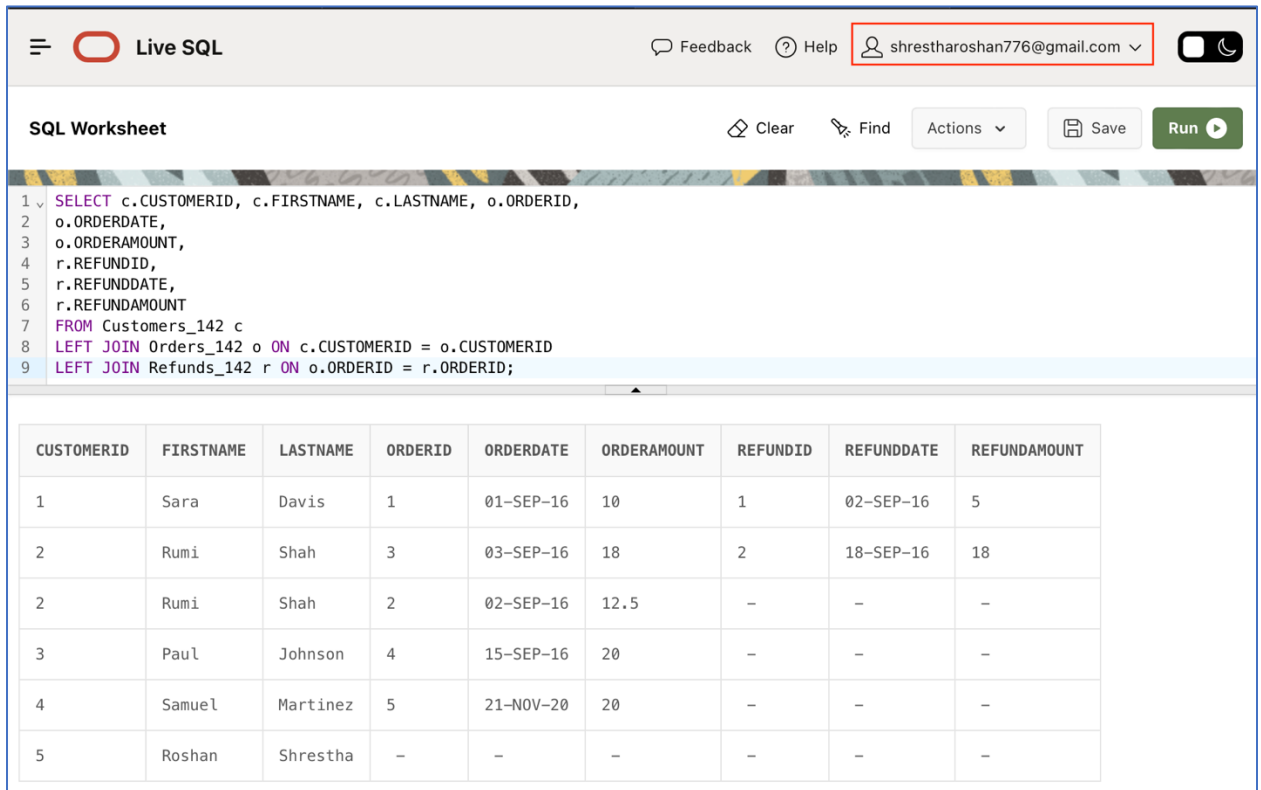
Below is the SQL query which we can use to get the required data:

```

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:



The screenshot shows the 'Live SQL' web application interface. At the top, there's a navigation bar with a menu icon, the 'Live SQL' logo, and a user profile dropdown showing 'shrestharoshan776@gmail.com'. Below the navigation bar is a toolbar with 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The main area displays the SQL query entered in the worksheet, which is the same query as shown in the previous block. Below the query, the results are displayed in a table with 9 columns: CUSTOMERID, FIRSTNAME, LASTNAME, ORDERID, ORDERDATE, ORDERAMOUNT, REFUNDID, REFUNDDATE, and REFUNDAMOUNT. The table contains 5 rows of data, showing customers and their associated orders and refunds. Some cells contain NULL values, represented by dashes in the screenshot.

CUSTOMERID	FIRSTNAME	LASTNAME	ORDERID	ORDERDATE	ORDERAMOUNT	REFUNDID	REFUNDDATE	REFUNDAMOUNT
1	Sara	Davis	1	01-SEP-16	10	1	02-SEP-16	5
2	Rumi	Shah	3	03-SEP-16	18	2	18-SEP-16	18
2	Rumi	Shah	2	02-SEP-16	12.5	-	-	-
3	Paul	Johnson	4	15-SEP-16	20	-	-	-
4	Samuel	Martinez	5	21-NOV-20	20	-	-	-
5	Roshan	Shrestha	-	-	-	-	-	-

### 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.