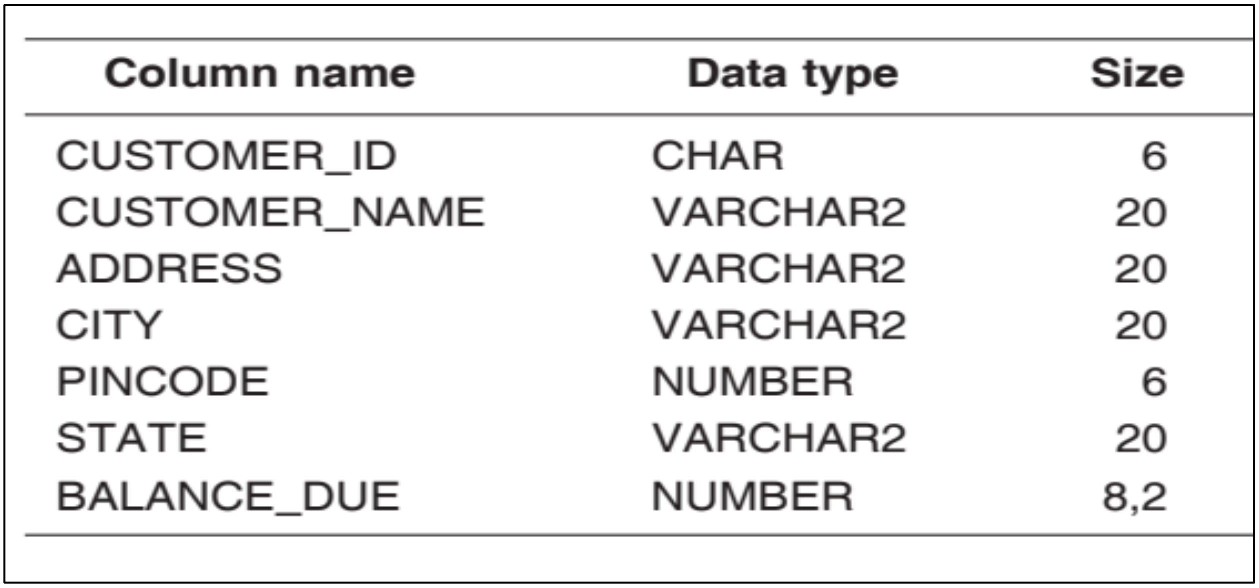
FSDM / CPCM – 2023S

Database Design & SQL

Student ID: 901142
Student Name: Roshan Shrestha
Practical Exercise 13_14_15_16

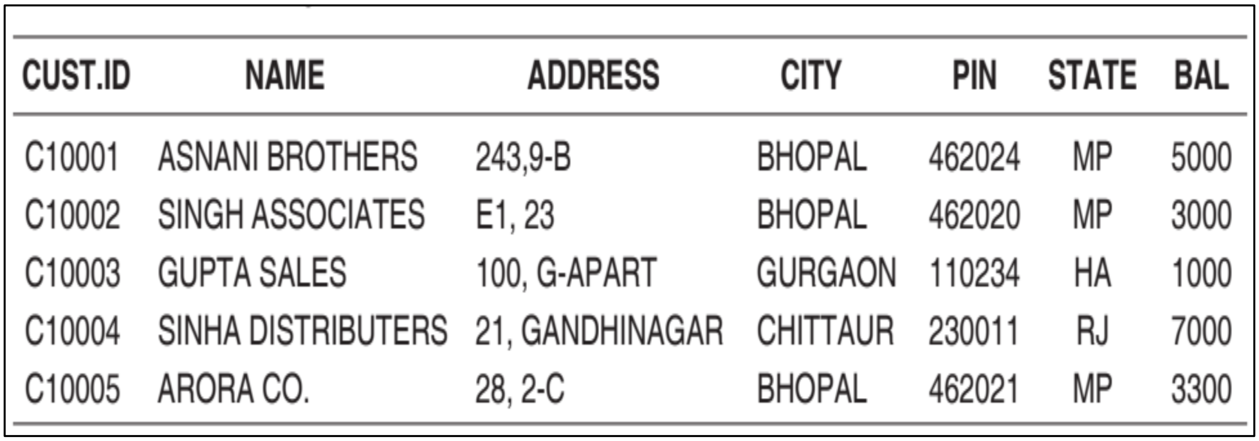
Fig 1



Column name	Data type	Size
CUSTOMER_ID	CHAR	6
CUSTOMER_NAME	VARCHAR2	20
ADDRESS	VARCHAR2	20
CITY	VARCHAR2	20
PINCODE	NUMBER	6
STATE	VARCHAR2	20
BALANCE_DUE	NUMBER	8,2

Figure 1: Table structure

Fig 2



CUST.ID	NAME	ADDRESS	CITY	PIN	STATE	BAL
C10001	ASNANI BROTHERS	243,9-B	BHOPAL	462024	MP	5000
C10002	SINGH ASSOCIATES	E1, 23	BHOPAL	462020	MP	3000
C10003	GUPTA SALES	100, G-APART	GURGAON	110234	HA	1000
C10004	SINHA DISTRIBUTERS	21, GANDHINAGAR	CHITTAUR	230011	RJ	7000
C10005	ARORA CO.	28, 2-C	BHOPAL	462021	MP	3300

Figure 2: Values inside table

Practical Activity #1

First, we need to execute the query to construct the **"CUSTOMERS_142"** table in the Oracle database using the column names and datatypes it contains, as shown in figure 1. The query is below:

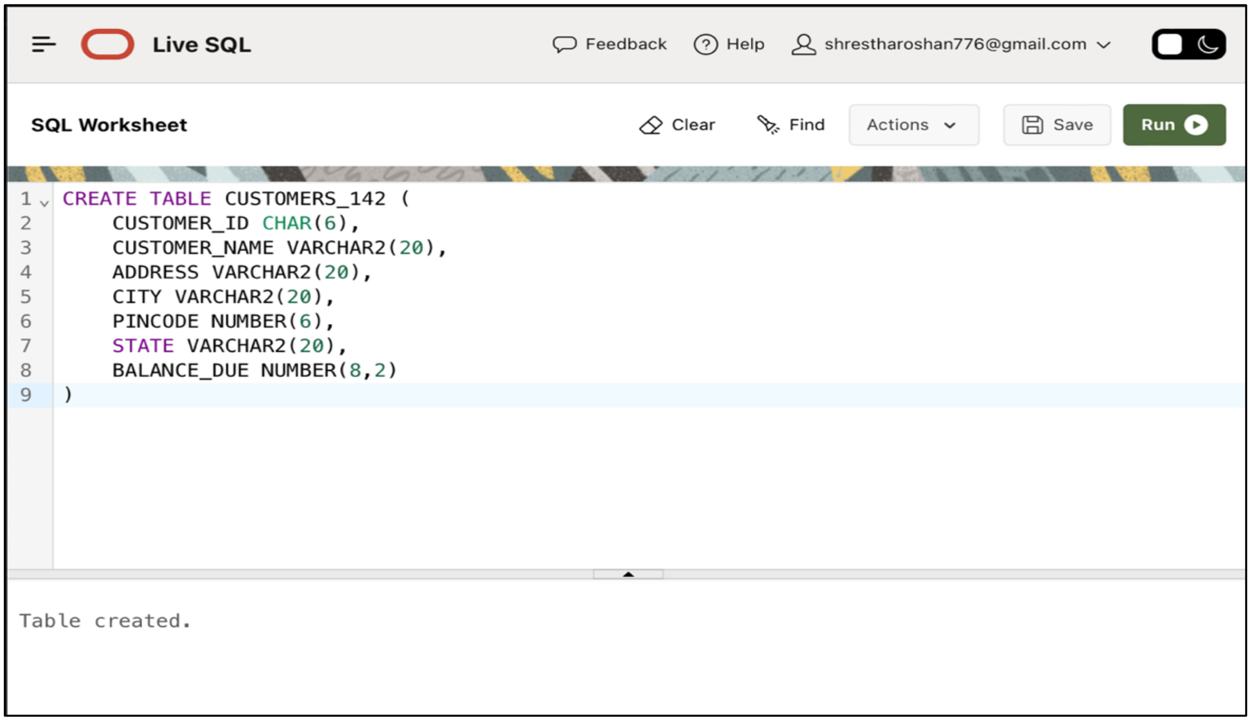
```
CREATE TABLE CUSTOMERS_142 (

   CUSTOMER_ID CHAR (6),

   CUSTOMER_NAME VARCHAR2(20),

   ADDRESS VARCHAR2(20),

   CITY VARCHAR2(20),

   PINCODE NUMBER (6),

   STATE VARCHAR2(20),

   BALANCE_DUE NUMBER (8,2))
```

Explanations of each line of table creation SQL query:

- To construct a new table with the name **"CUSTOMERS_142"** the query starts with **"CREATE TABLE CUSTOMERS_142"**.
- The following columns will be in the table:

    - The **"CUSTOMER_ID"** column, which has a **"CHAR (6)"** data type. It can hold as many as six characters.
    - **"VARCHAR2(20)"** data type for the "**CUSTOMER_NAME"** column. It can hold strings with a configurable length of up to 20 characters.
    - **"VARCHAR2(20)"** data type for the **"ADDRESS"** field. It can hold strings with a configurable length of up to 20 characters.
    - **"VARCHAR2(20)"** data type for the **"CITY"** column. It can hold strings with a configurable length of up to 20 characters.
    - **"PINCODE"** column with **"NUMBER (6)"** as the data type. It can store numbers with up to six digits.
    - A column named **"STATE"** with the data type **"VARCHAR2(20)"**. It can hold strings with a configurable length of up to 20 characters.
    - **"BALANCE_DUE"** column containing a data type of **"NUMBER (8,2)"**. It can store numerical values up to eight digits with two decimal places.

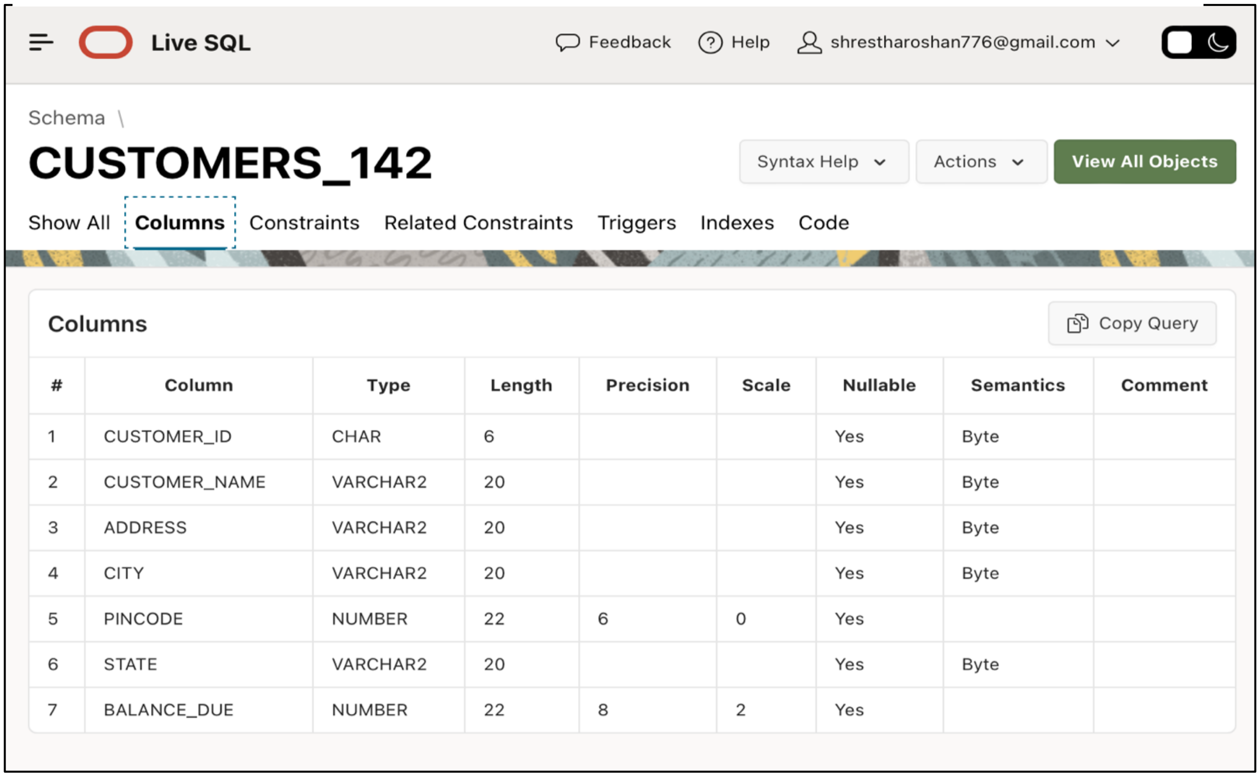
For the execution of the query and visualizing the output I have used live oracle from https://livesql.oracle.com/. The executed query and it's result output have been attached with screenshots below:



```
CREATE TABLE CUSTOMERS_142 (
    CUSTOMER_ID CHAR(6),
    CUSTOMER_NAME VARCHAR2(20),
    ADDRESS VARCHAR2(20),
    CITY VARCHAR2(20),
    PINCODE NUMBER(6),
    STATE VARCHAR2(20),
    BALANCE_DUE NUMBER(8,2)
)
```

```
Table created.
```

Figure 3: Query to create new table.



CUSTOMERS_142

Columns

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	CUSTOMER_ID	CHAR	6			Yes	Byte	
2	CUSTOMER_NAME	VARCHAR2	20			Yes	Byte	
3	ADDRESS	VARCHAR2	20			Yes	Byte	
4	CITY	VARCHAR2	20			Yes	Byte	
5	PINCODE	NUMBER	22	6	0	Yes		
6	STATE	VARCHAR2	20			Yes	Byte	
7	BALANCE_DUE	NUMBER	22	8	2	Yes		

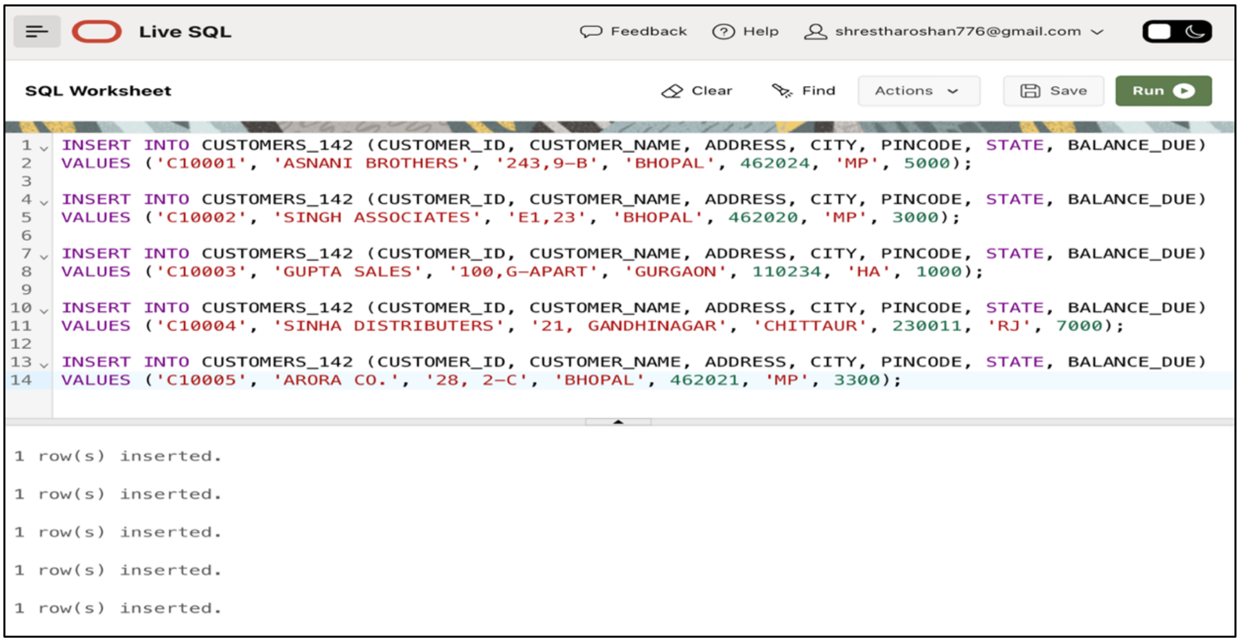
Figure 4: Created table structure.

Practical Activity #2

On the second step to insert the data into the **"CUSTOMERS_142"** table as shown in figure 2, we need to execute the **"INSERT"** query with appropriate values as below:

1. **INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE) VALUES ('C10001', 'ASNANI BROTHERS', '243,9-B', 'BHOPAL', 462024, 'MP', 5000);**
2. **INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE) VALUES ('C10002', 'SINGH ASSOCIATES', 'E1,23', 'BHOPAL', 462020, 'MP', 3000);**
3. **INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE) VALUES ('C10003', 'GUPTA SALES', '100, G-APART', 'GURGAON', 110234, 'HA', 1000);**
4. **INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE) VALUES ('C10004', 'SINHA DISTRIBUTERS', '21, GANDHINAGAR', 'CHITTAUR', 230011, 'RJ', 7000);**
5. **INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE) VALUES ('C10005', 'ARORA CO.', '28, 2-C', 'BHOPAL', 462021, 'MP', 3300);**

A single row of data is added to the table with each **INSERT** operation. The values offered in the **VALUES** clause match the columns in the table structure's defined order. The following figure illustrates the insert queries that were executed:

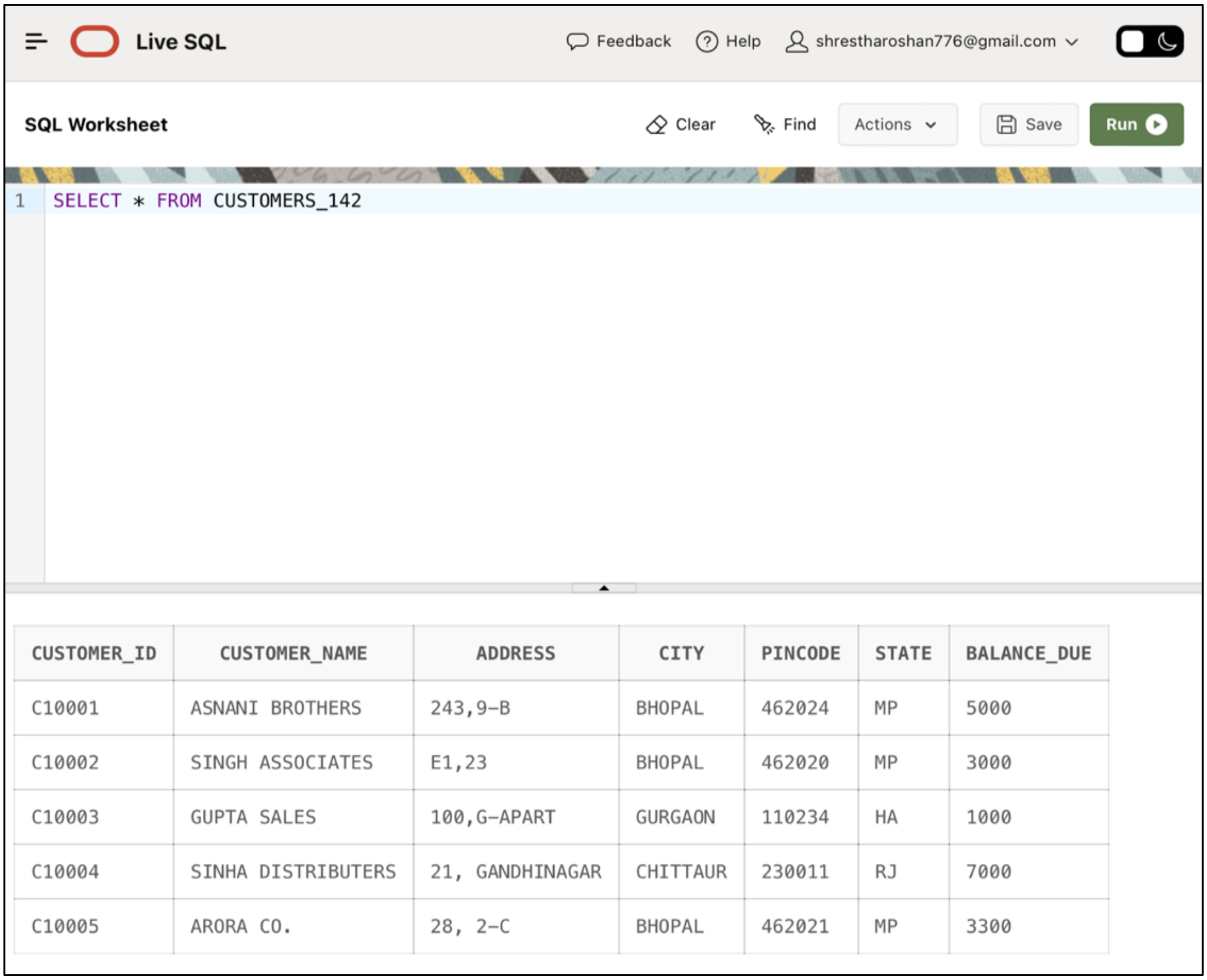


```
INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE)
VALUES ('C10001', 'ASNANI BROTHERS', '243,9-B', 'BHOPAL', 462024, 'MP', 5000);

INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE)
VALUES ('C10002', 'SINGH ASSOCIATES', 'E1,23', 'BHOPAL', 462020, 'MP', 3000);

INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE)
VALUES ('C10003', 'GUPTA SALES', '100,G-APART', 'GURGAON', 110234, 'HA', 1000);

INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE)
VALUES ('C10004', 'SINHA DISTRIBUTERS', '21, GANDHINAGAR', 'CHITTAUR', 230011, 'RJ', 7000);

INSERT INTO CUSTOMERS_142 (CUSTOMER_ID, CUSTOMER_NAME, ADDRESS, CITY, PINCODE, STATE, BALANCE_DUE)
VALUES ('C10005', 'ARORA CO.', '28, 2-C', 'BHOPAL', 462021, 'MP', 3300);
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

Figure 5: Query to insert values to table.

The data inserted into the table can be visualized in the picture below:



```
SELECT * FROM CUSTOMERS_142
```

CUSTOMER_ID	CUSTOMER_NAME	ADDRESS	CITY	PINCODE	STATE	BALANCE_DUE
C10001	ASNANI BROTHERS	243,9-B	BHOPAL	462024	MP	5000
C10002	SINGH ASSOCIATES	E1,23	BHOPAL	462020	MP	3000
C10003	GUPTA SALES	100,G-APART	GURGAON	110234	HA	1000
C10004	SINHA DISTRIBUTERS	21, GANDHINAGAR	CHITTAUR	230011	RJ	7000
C10005	ARORA CO.	28, 2-C	BHOPAL	462021	MP	3300

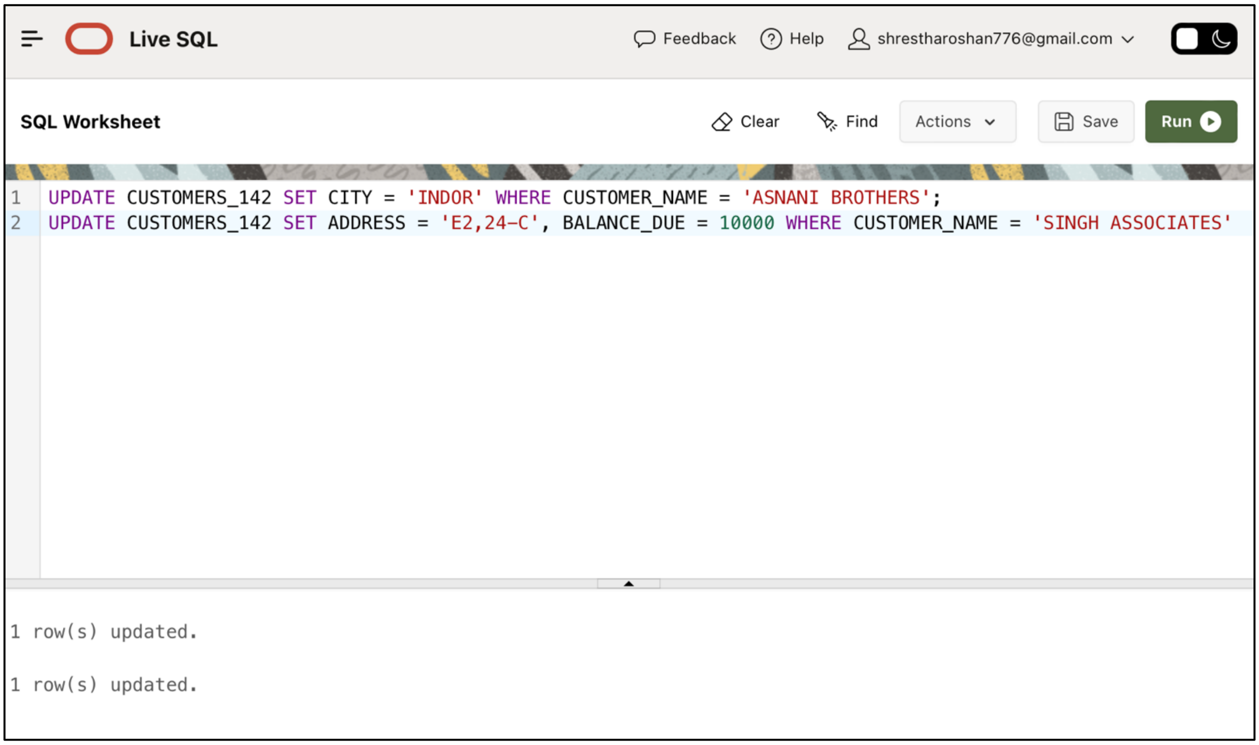
Figure 6: Inserted Values in the table

Practical Activity #3

To update the table with the given values we need to execute the following query along with the new values. The query is below:

UPDATE CUSTOMERS_142 SET CITY = 'INDOR' WHERE CUSTOMER_NAME = 'ASNANI BROTHERS';
UPDATE CUSTOMERS_142 SET ADDRESS = 'E2,24-C', BALANCE_DUE = 10000 WHERE CUSTOMER_NAME = 'SINGH ASSOCIATES'

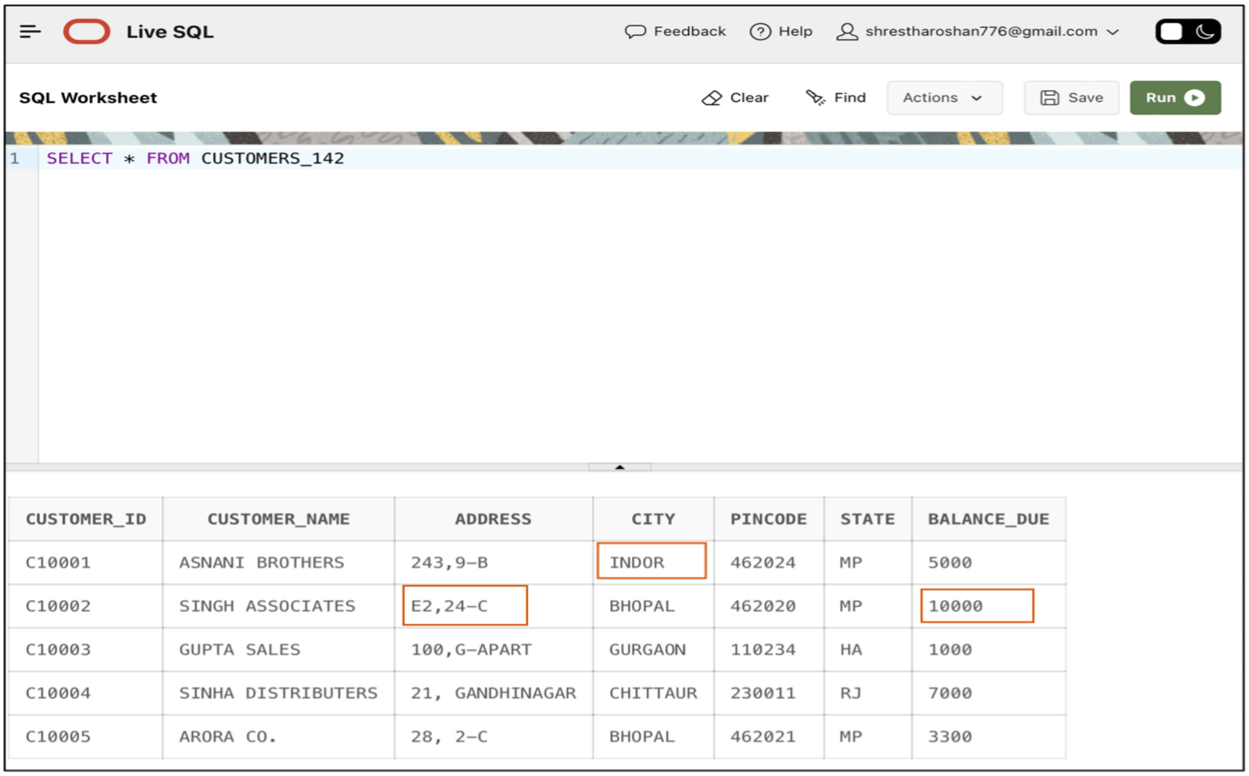
The first **UPDATE** query replaces **"BHOPAL"** with **"INDORE"** as the city of the client with the name **"ASNANI BROTHERS"**. In the second **UPDATE** query, the customer with the name **"SINGH ASSOCIATES"** has their address changed to **"E2, 24-C"** and their **BALANCE_DUE** value increased to **10,000**. The executed query and it's result can be visualized from the screenshot below:



```
UPDATE CUSTOMERS_142 SET CITY = 'INDOR' WHERE CUSTOMER_NAME = 'ASNANI BROTHERS';
UPDATE CUSTOMERS_142 SET ADDRESS = 'E2,24-C', BALANCE_DUE = 10000 WHERE CUSTOMER_NAME = 'SINGH ASSOCIATES'
```

```
1 row(s) updated.

1 row(s) updated.
```

Figure 7: Query to update columns in the table.



```
SELECT * FROM CUSTOMERS_142
```

CUSTOMER_ID	CUSTOMER_NAME	ADDRESS	CITY	PINCODE	STATE	BALANCE_DUE
C10001	ASNANI BROTHERS	243,9-B	INDOR	462024	MP	5000
C10002	SINGH ASSOCIATES	E2,24-C	BHOPAL	462020	MP	10000
C10003	GUPTA SALES	100,G-APART	GURGAON	110234	HA	1000
C10004	SINHA DISTRIBUTERS	21, GANDHINAGAR	CHITTAUR	230011	RJ	7000
C10005	ARORA CO.	28, 2-C	BHOPAL	462021	MP	3300

Figure 8: Updated columns with new values

Practical Activity #4

For deleting the records from the table, we can use **"DELETE"** query.

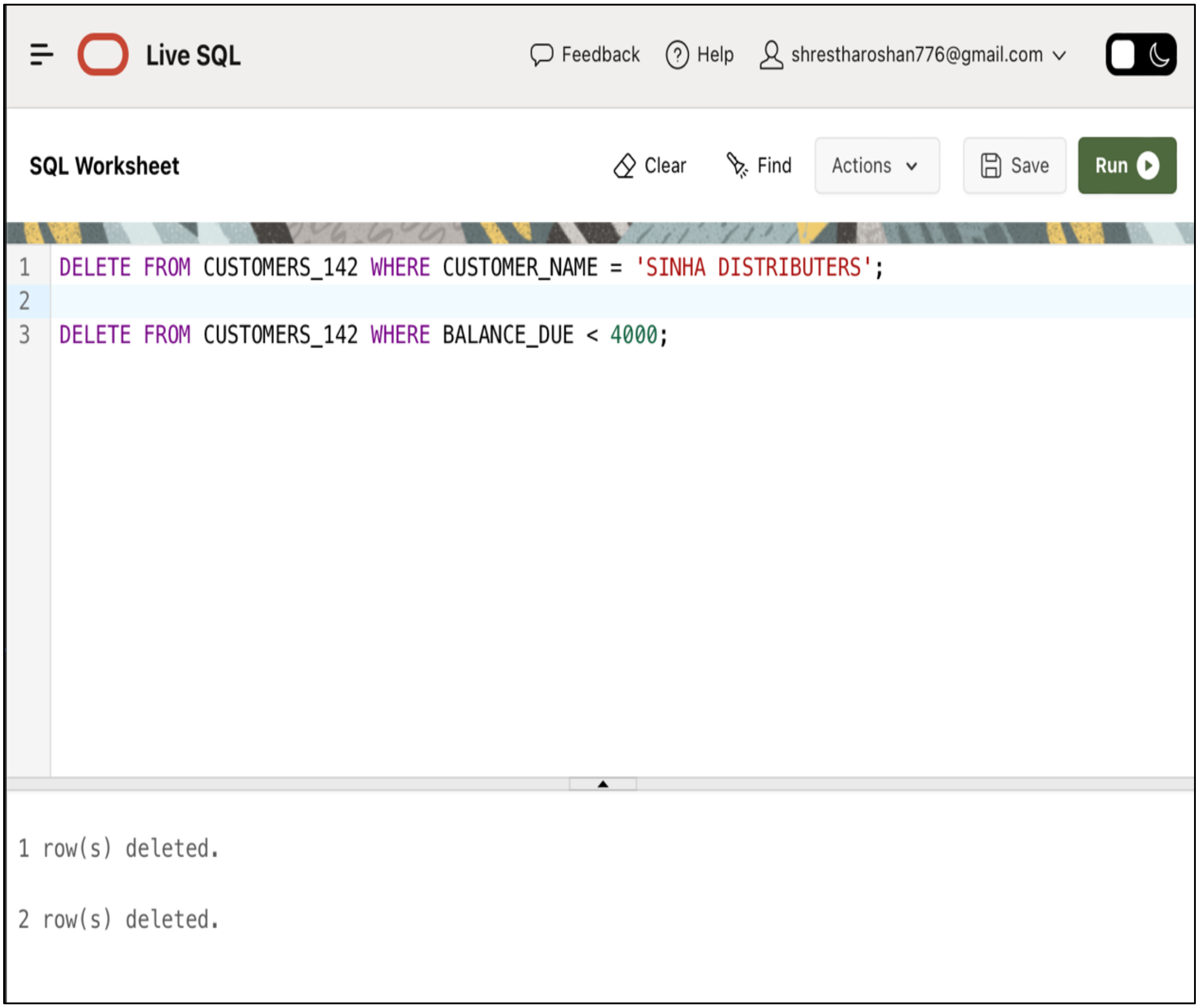
The required query to delete the records for **"SHINHA DISTRIBUTERS"** we can use the following query:

DELETE FROM CUSTOMERS_142 WHERE CUSTOMER_NAME = 'SINHA DISTRIBUTERS';

Similarly, to delete the customers with balance below 4000, we can use the following query:

DELETE FROM CUSTOMERS_142 WHERE BALANCE_DUE < 4000;

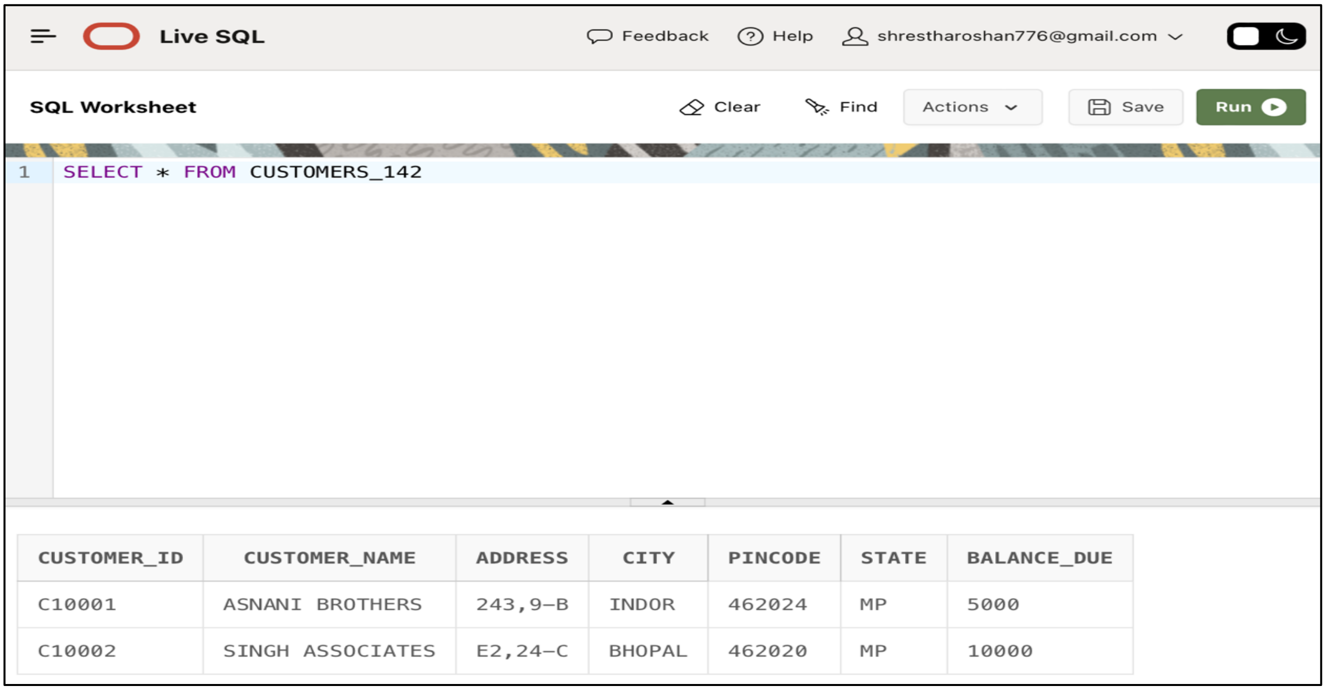
The executed query and the output are attached in below screenshots:



```
DELETE FROM CUSTOMERS_142 WHERE CUSTOMER_NAME = 'SINHA DISTRIBUTERS';

DELETE FROM CUSTOMERS_142 WHERE BALANCE_DUE < 4000;
```

```
1 row(s) deleted.

2 row(s) deleted.
```

Figure 9: Delete columns based on given condition.



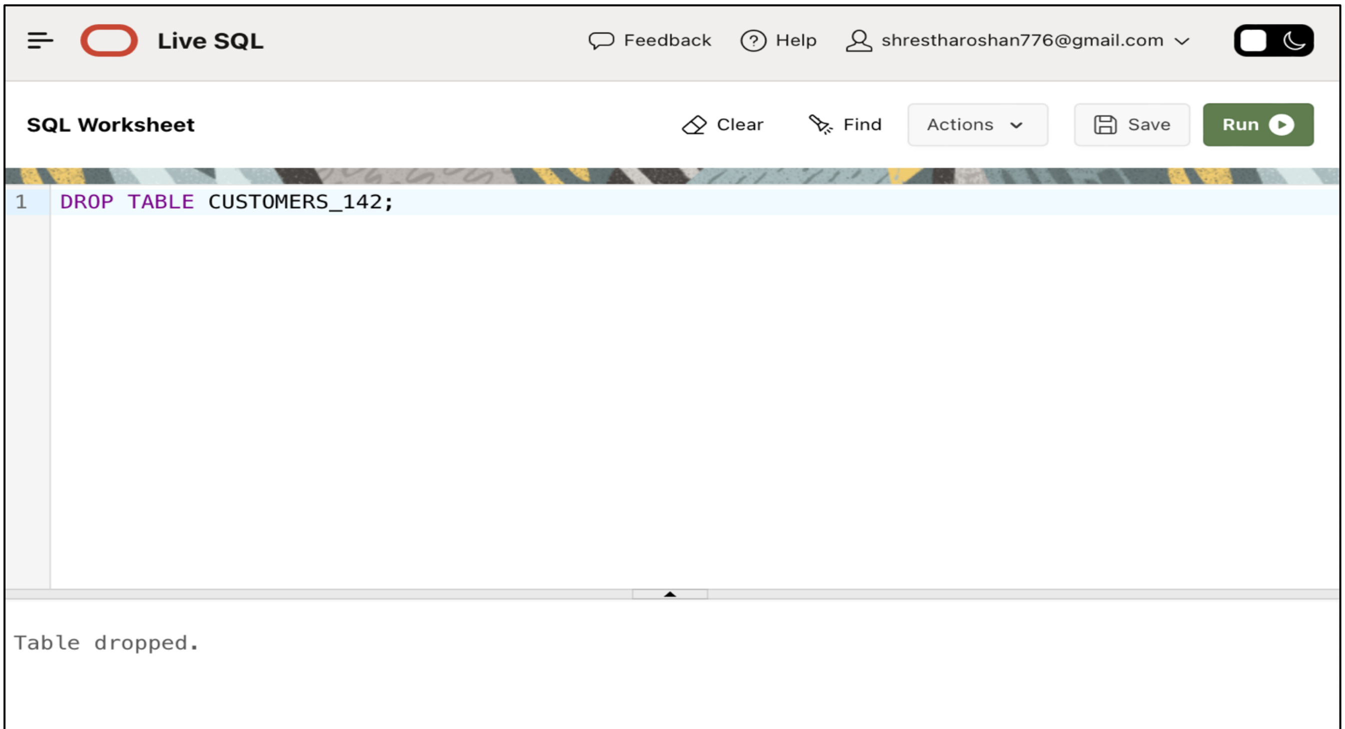
```
SELECT * FROM CUSTOMERS_142
```

CUSTOMER_ID	CUSTOMER_NAME	ADDRESS	CITY	PINCODE	STATE	BALANCE_DUE
C10001	ASNANI BROTHERS	243,9-B	INDOR	462024	MP	5000
C10002	SINGH ASSOCIATES	E2,24-C	BHOPAL	462020	MP	10000

Figure 10: Updated table after deletion of columns

Lastly, to delete the entire table we can use **"DROP"** query as shown below:

DROP TABLE CUSTOMERS_142;
The executed query can be visualized in screenshot below:

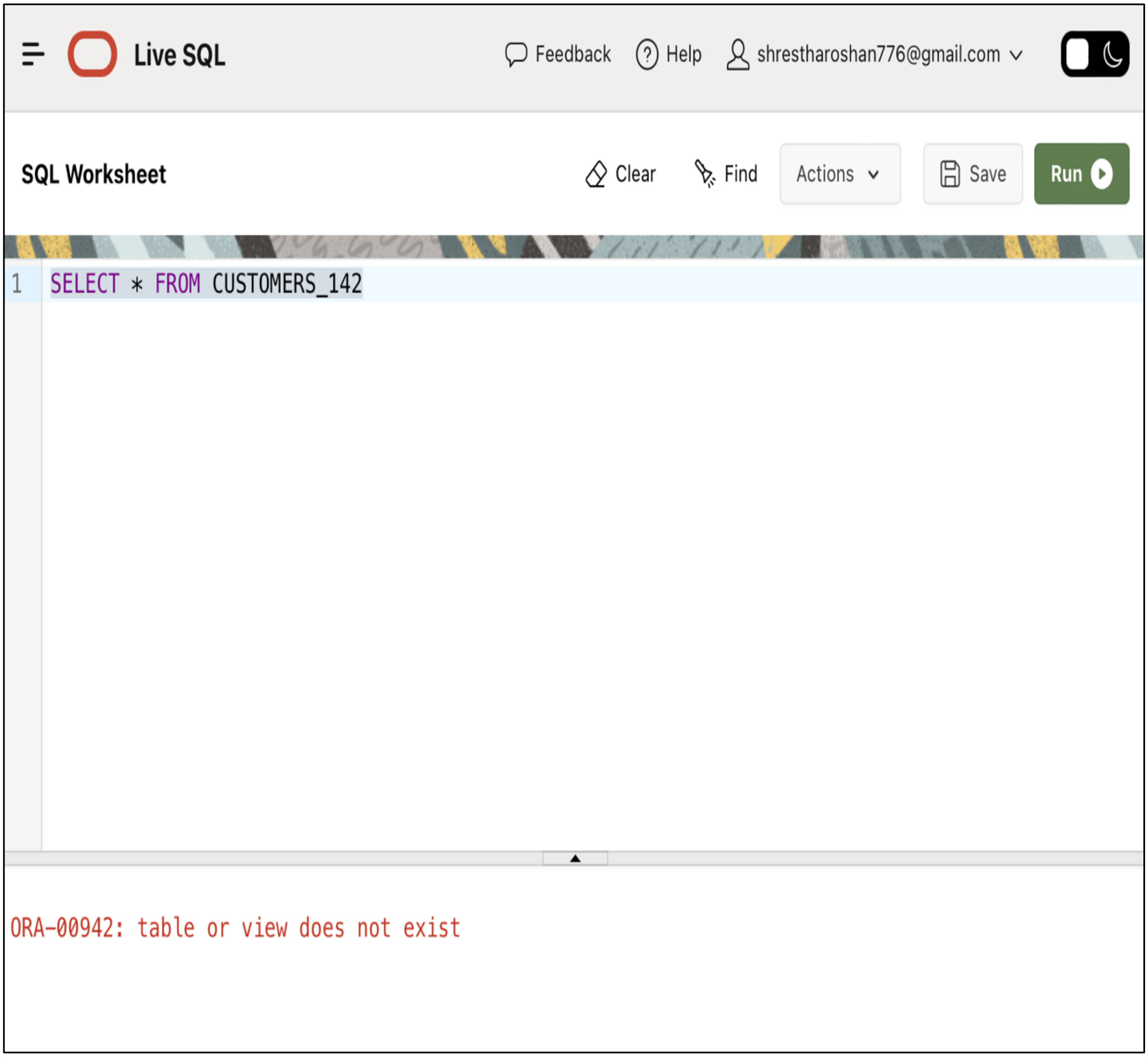


```
DROP TABLE CUSTOMERS_142;
```

```
Table dropped.
```

Figure 11: Drop the table

As we have erased the entire table along with its columns containing data, when we try to execute the query to retrieve the table, it gives us error explain that the requested table doesn't exist anymore as shown in below screenshot:



```
SELECT * FROM CUSTOMERS_142
```

```
ORA-00942: table or view does not exist
```

Figure 12: Retrieve the deleted table