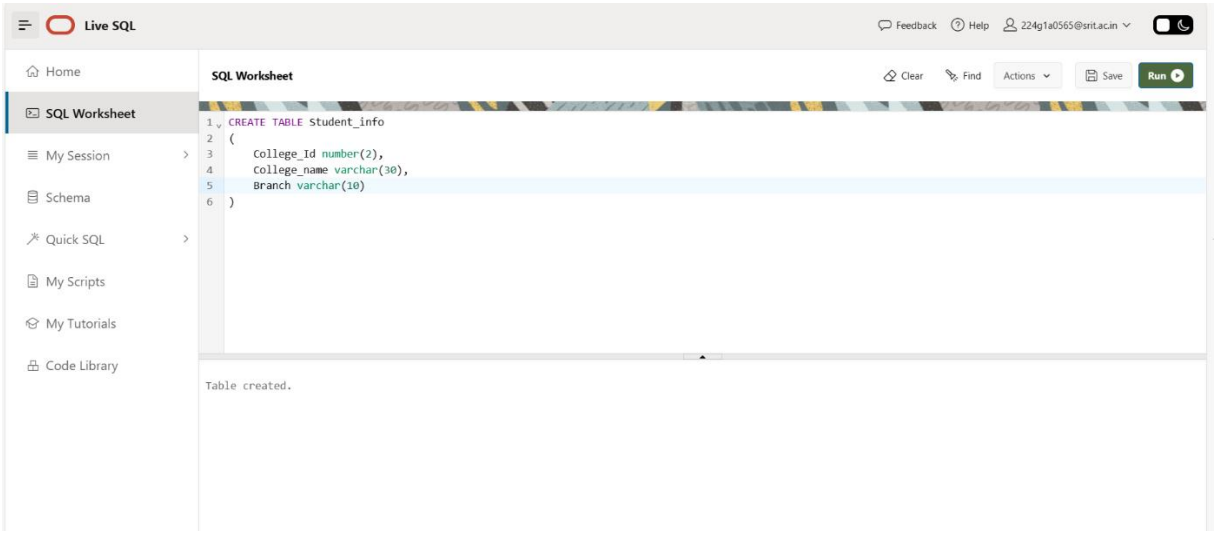
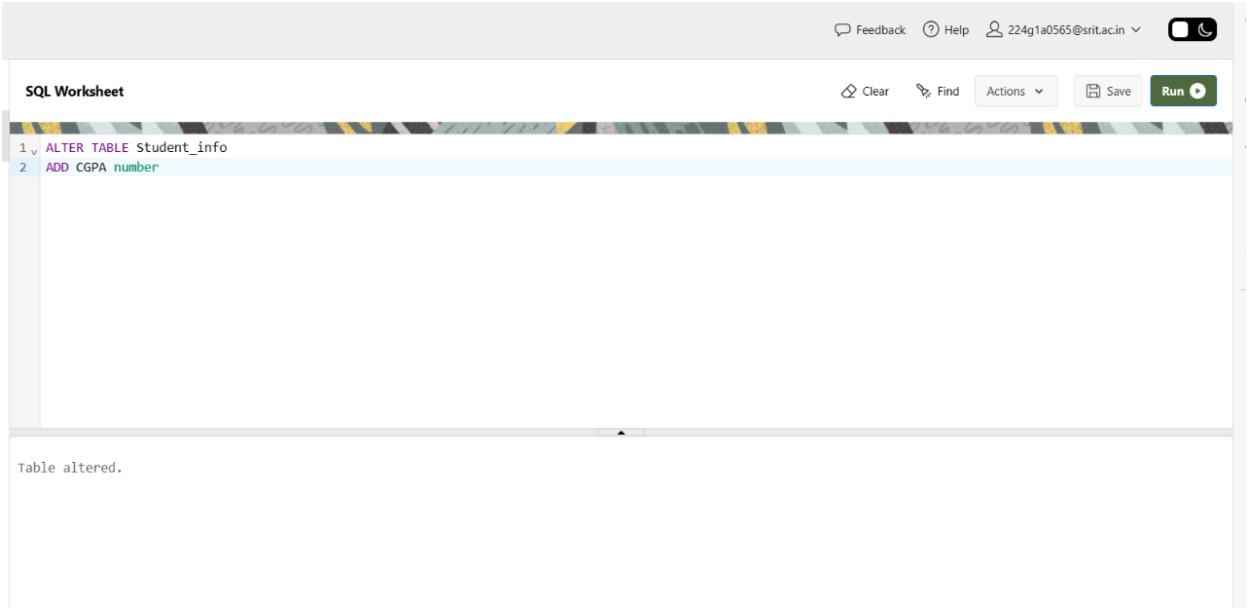
224g1a0565-k.pallavi

EXPERIMENT-1

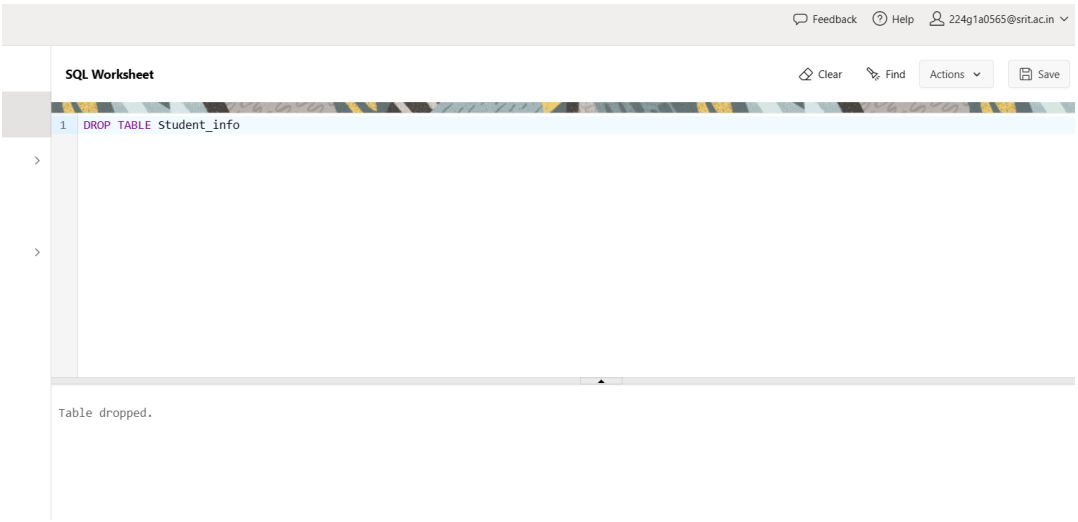
1.CREATE Table:



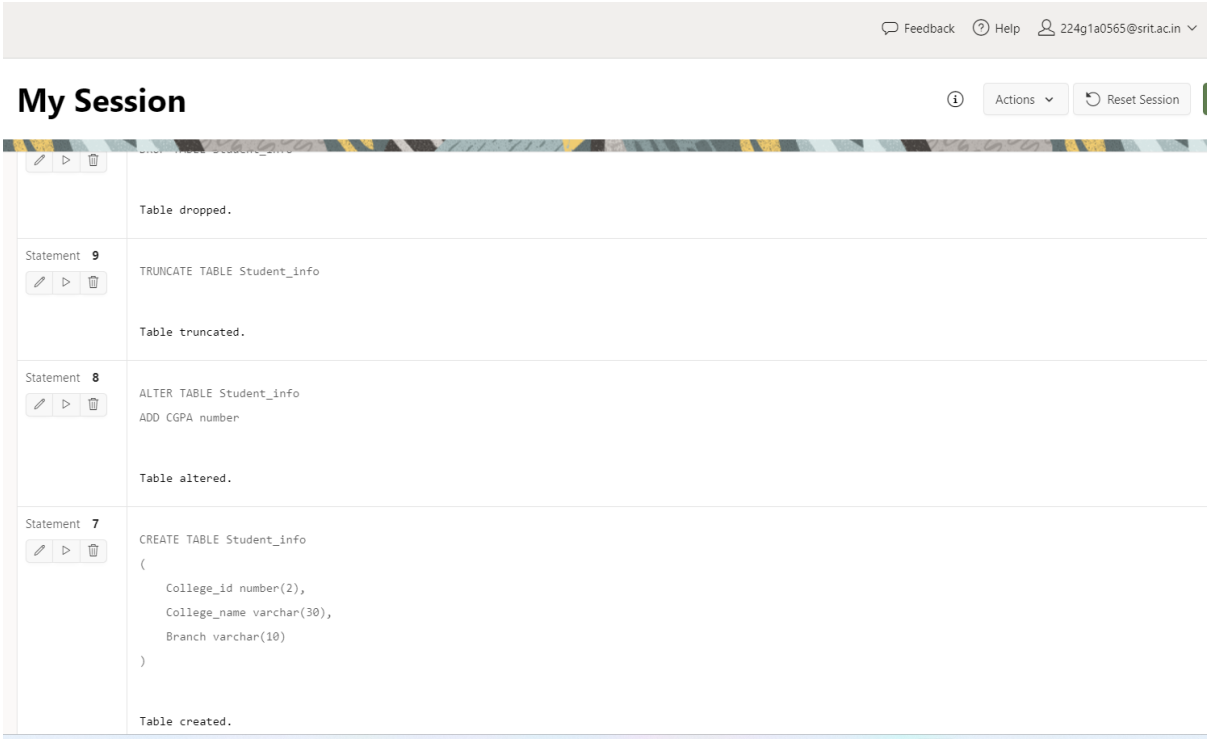
2.ALTER table:



3.Drop Table:



4.TRUNCATE Table:



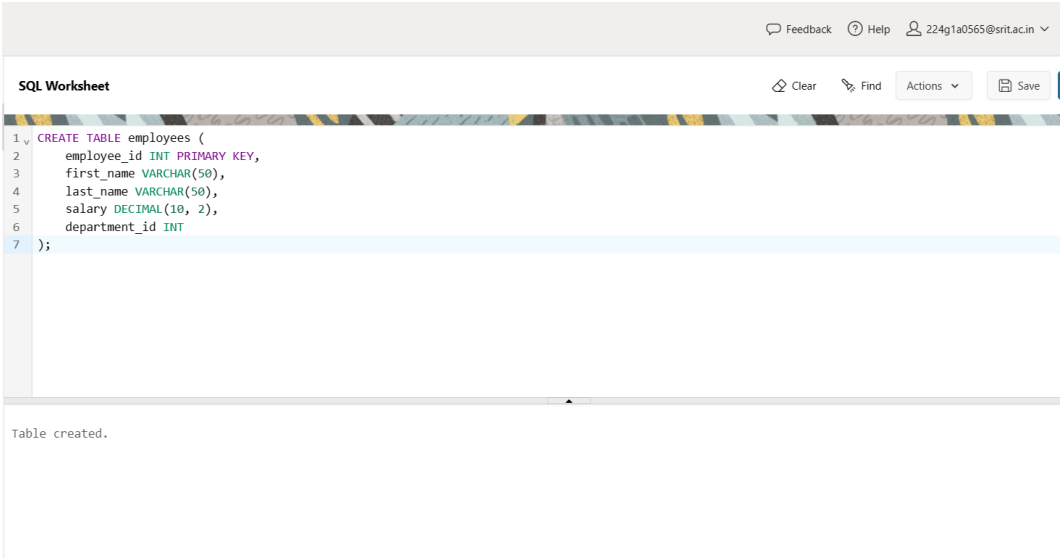
conclusion:

In this lab, we successfully practiced SQL queries to CREATE TABLES for various databases using DDL commands.

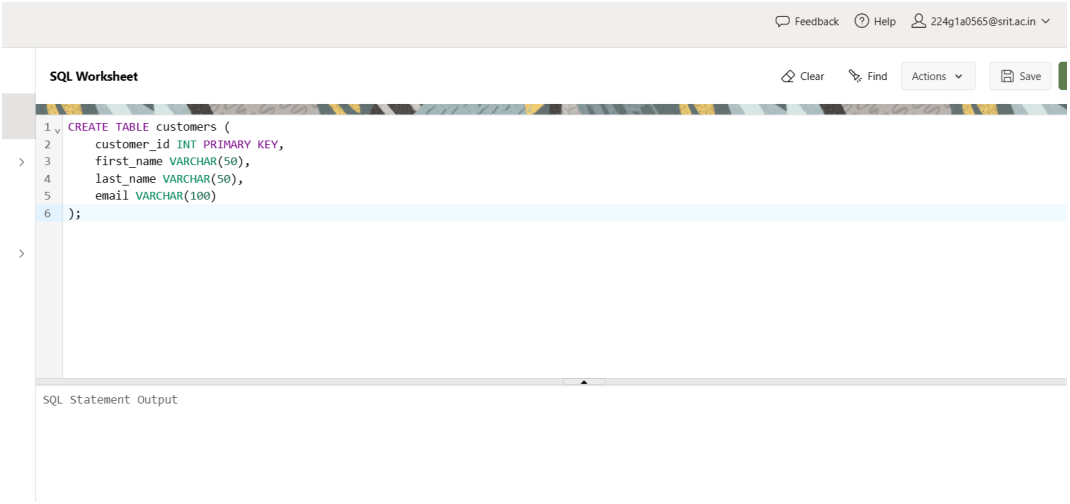
224G1A0565-k.pallavi

EXPERIMENT-2

1.CREATE TABLE for employees:

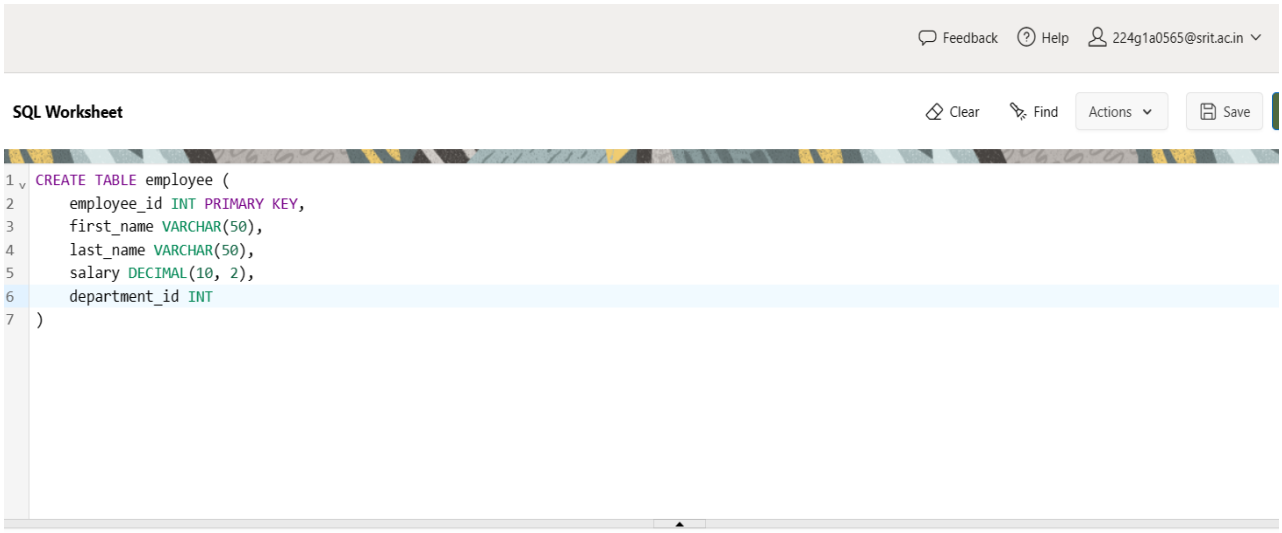


2.CREATE TABLE for customers:

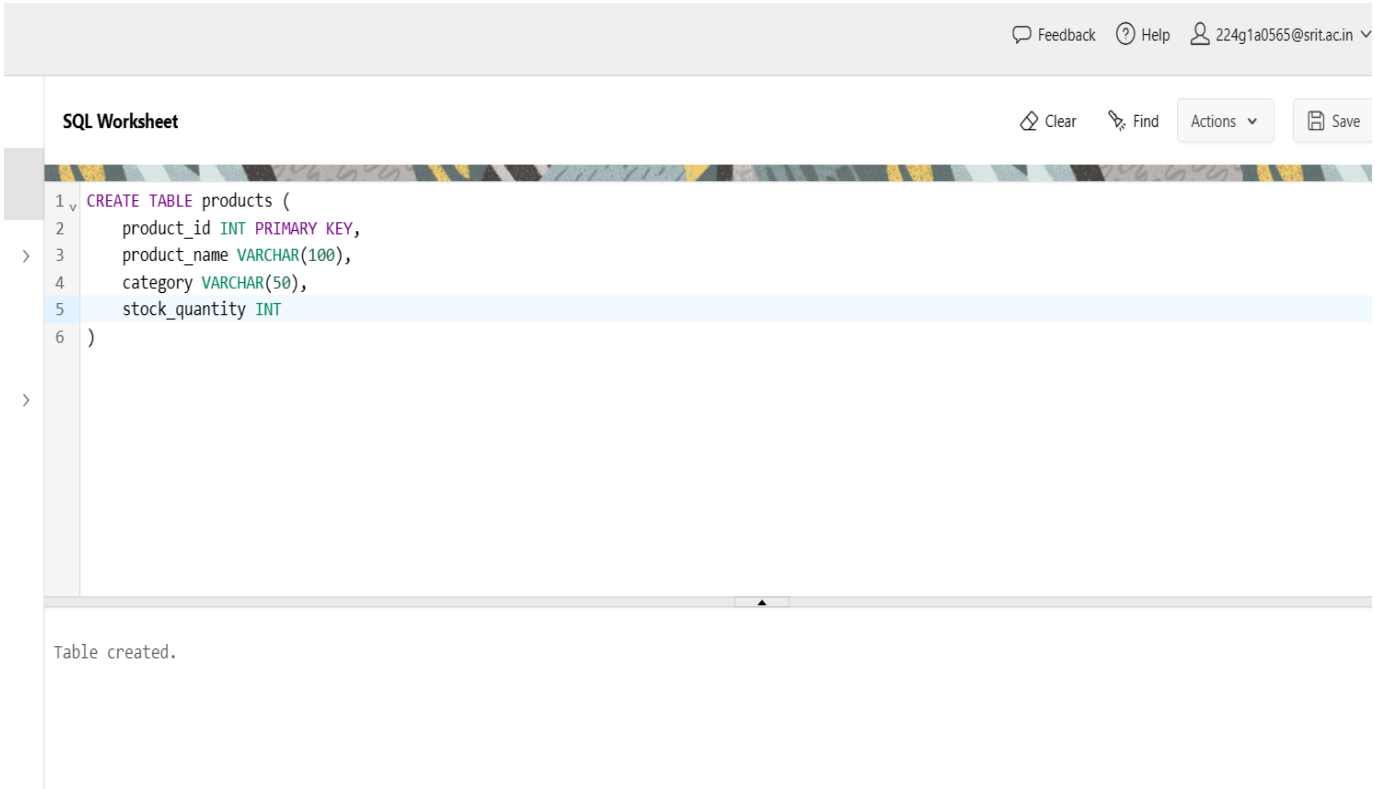


3.CREATE TABLE for products:

224G1A0565-k.pallavi

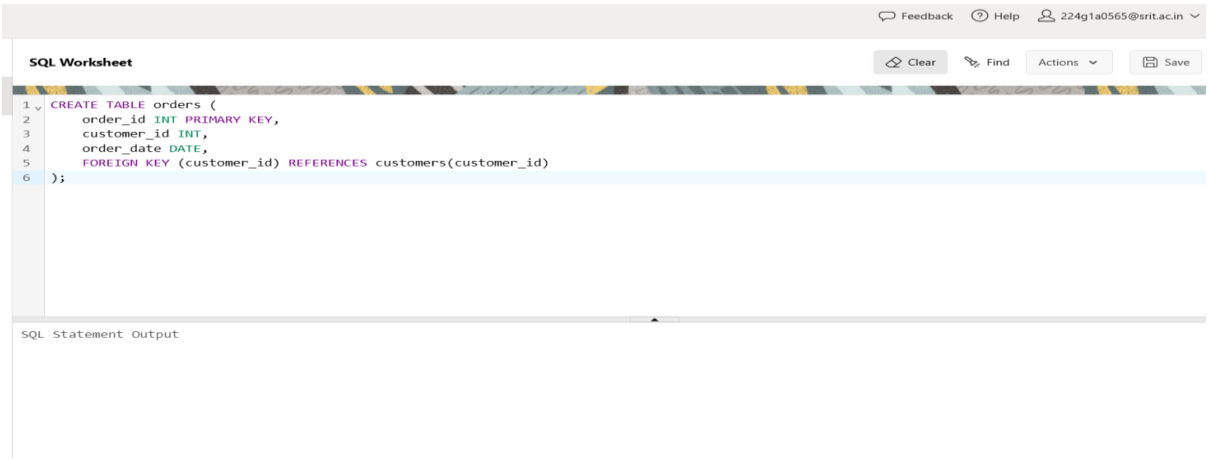



```
Table created.
```

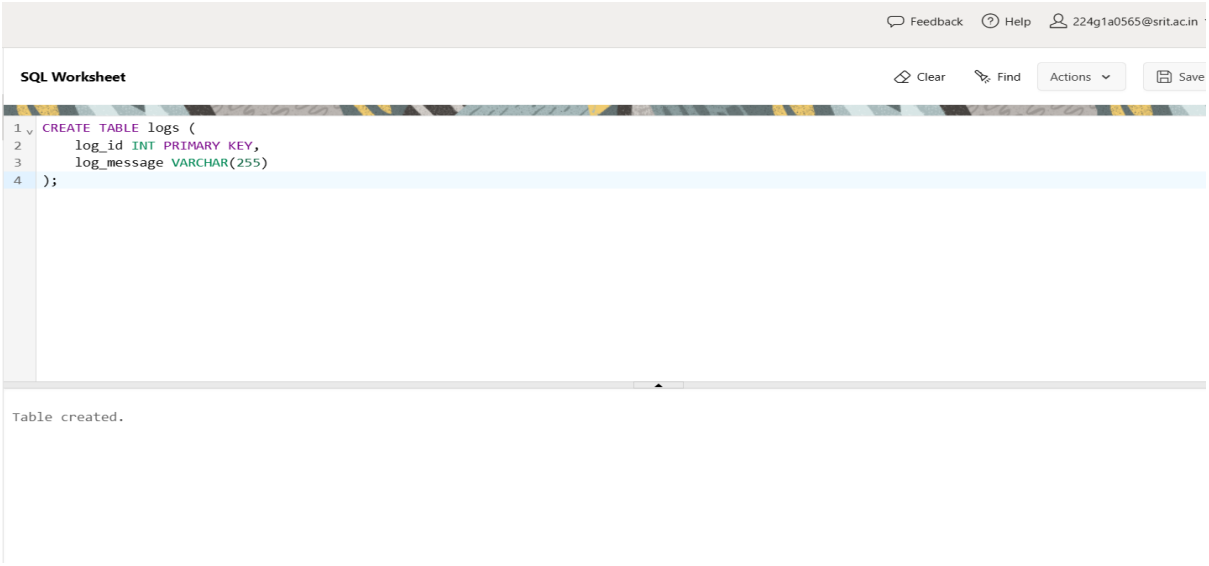


4.CREATE TABLE for orders:

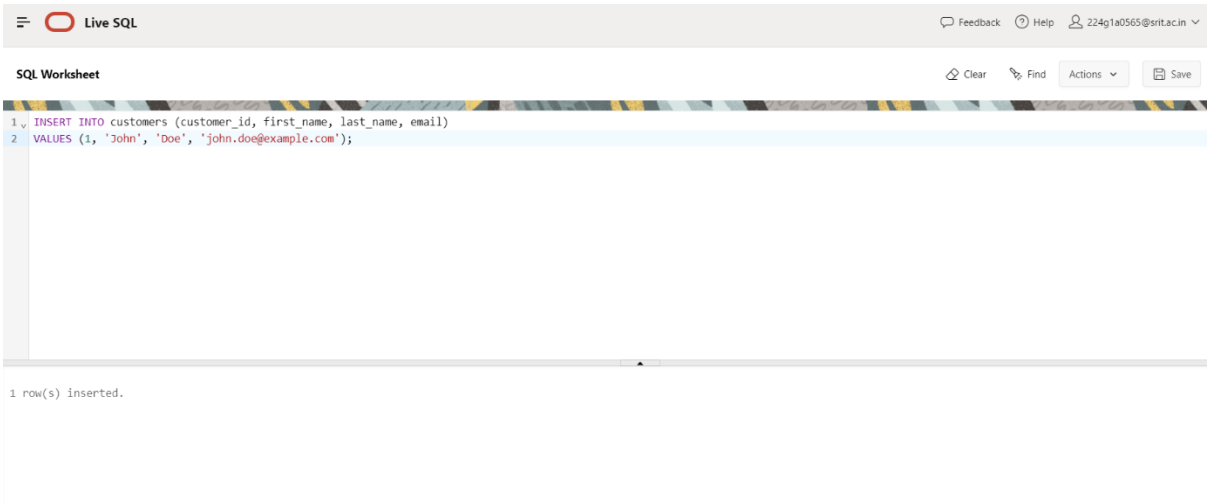
224G1A0565-k.pallavi



5.CREATE TABLE for logs:

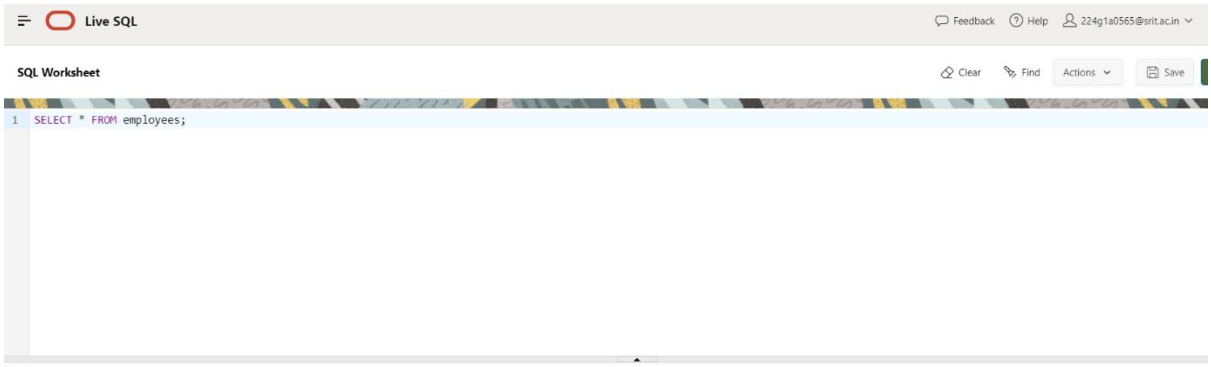


6.INSERT:



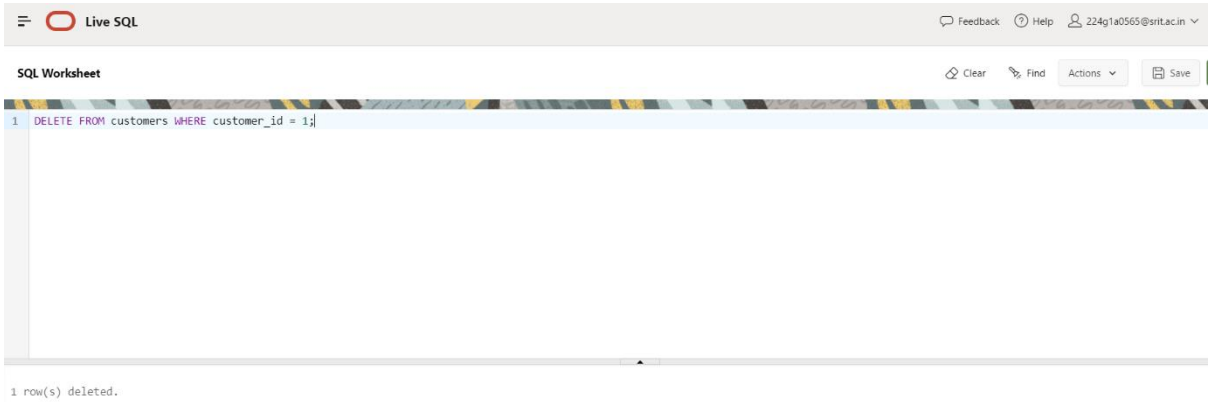
7.SELECT:

224G1A0565-k.pallavi



8.UPDATE:

9.DELETE:



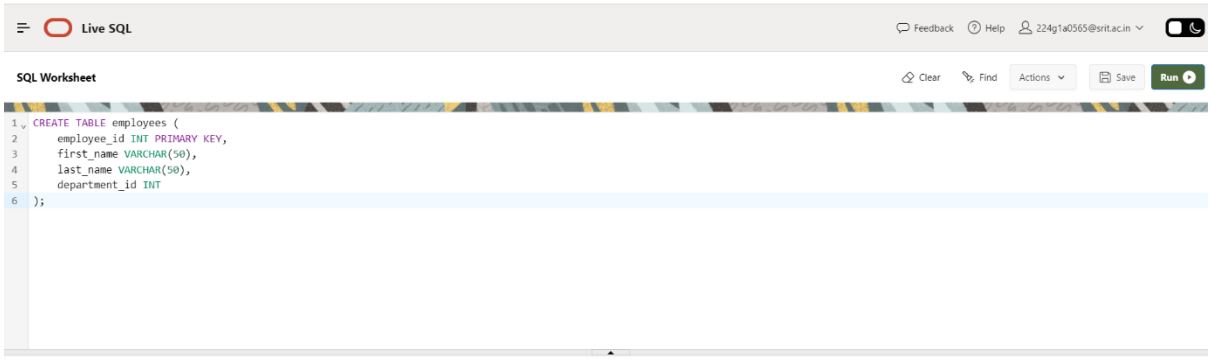
conclusion:

In this lab,we successfully practised SQL queries using DML commands (SELECT,UPDATE,INSERT,DELETE).

224G1A0565-K.pallavi

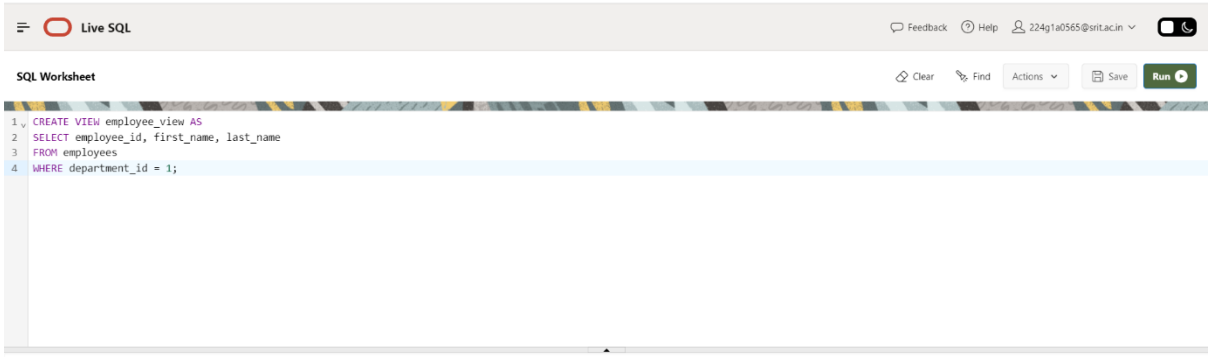
EXPERIMENT-3

1.CREATE TABLE:



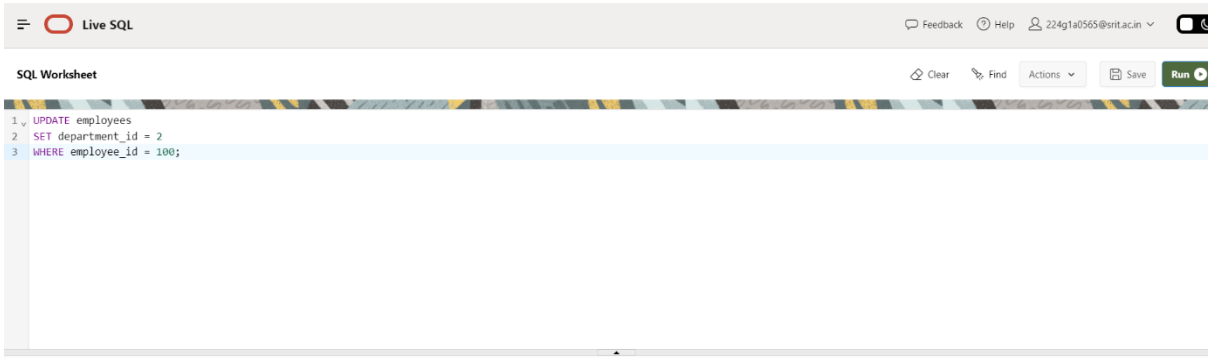

```
Table created.
```

2.CREATING A VIEW:



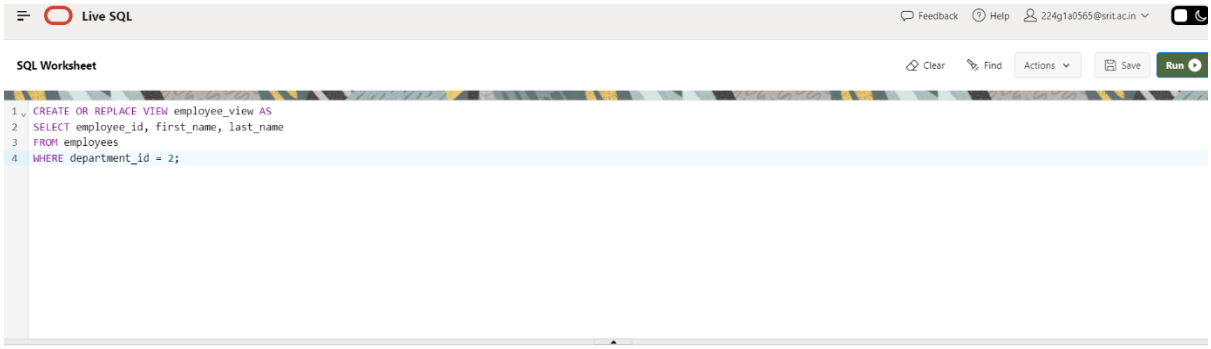

```
View created.
```

3.UPDATING A VIEW:

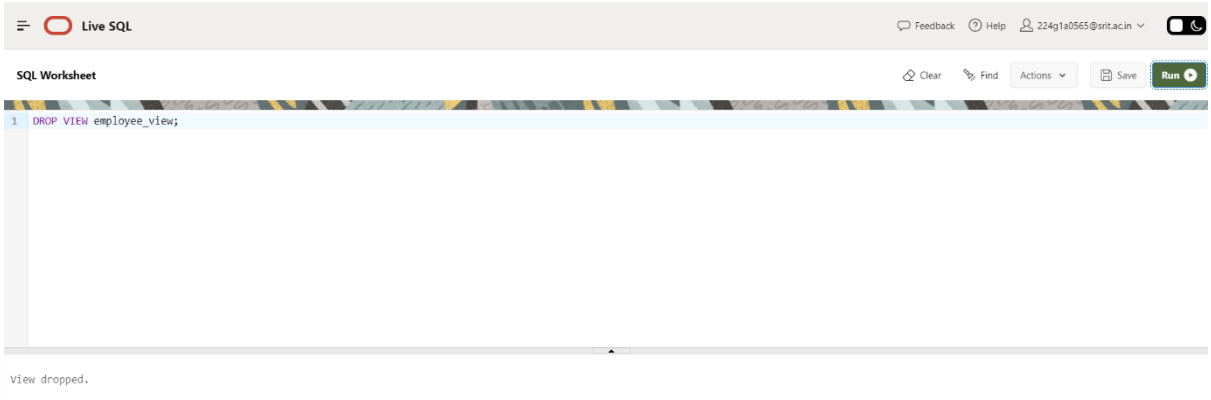



```
0 row(s) updated.
```

4.ALTERING A VIEW:



5.DELETING A VIEW:



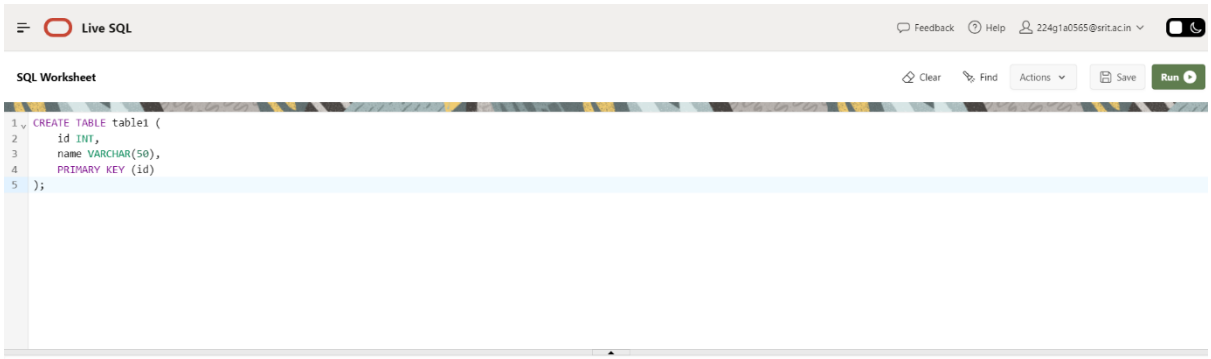
conclusion:

In the lab,we successfully practiced SQL Queries to create VIEWS for various data base (CREATE VIEW,UPDATE VIEW, ALTER VIEW,, DELETE VIEW).

224G1A0565-K.pallavi

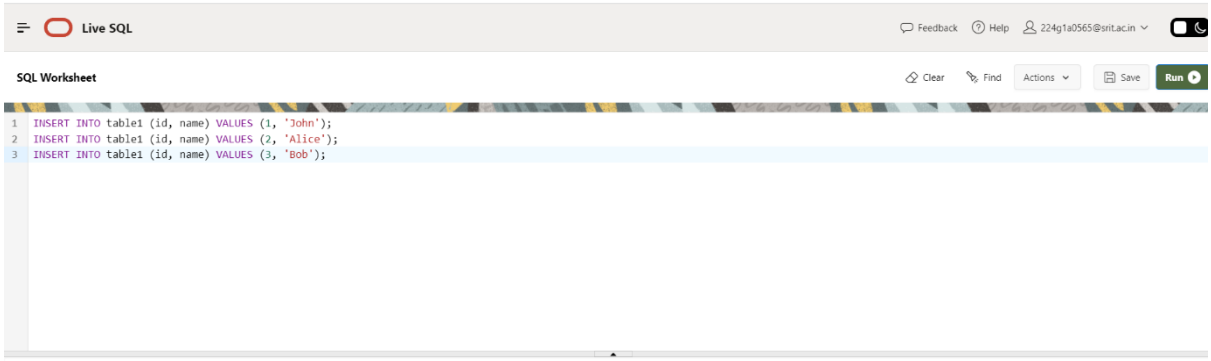
EXPERIMENT-4

1.creating a table1:




```
Table created.
```

2.inserting VALUES into table1:

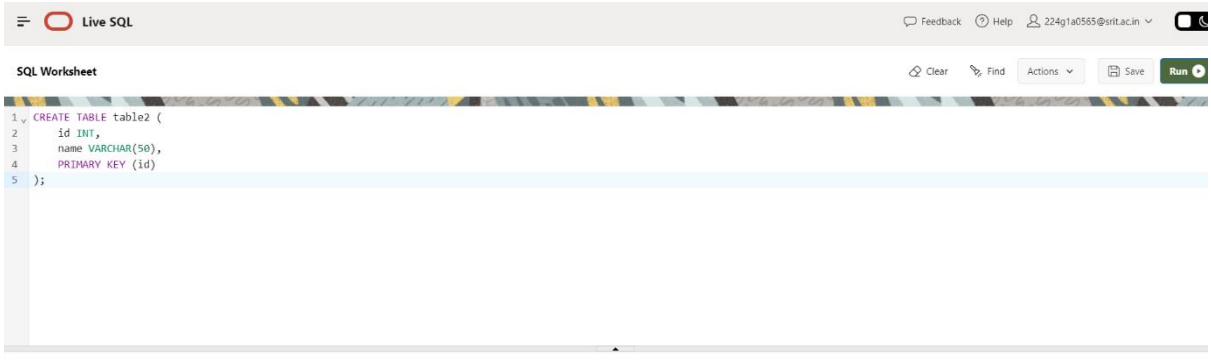



```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

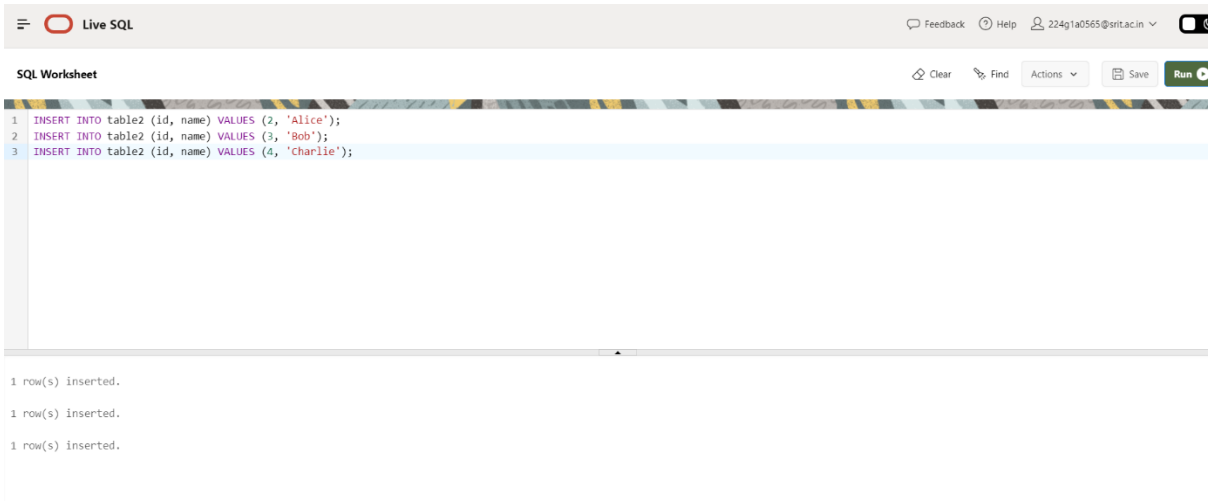
3.creating a table2:



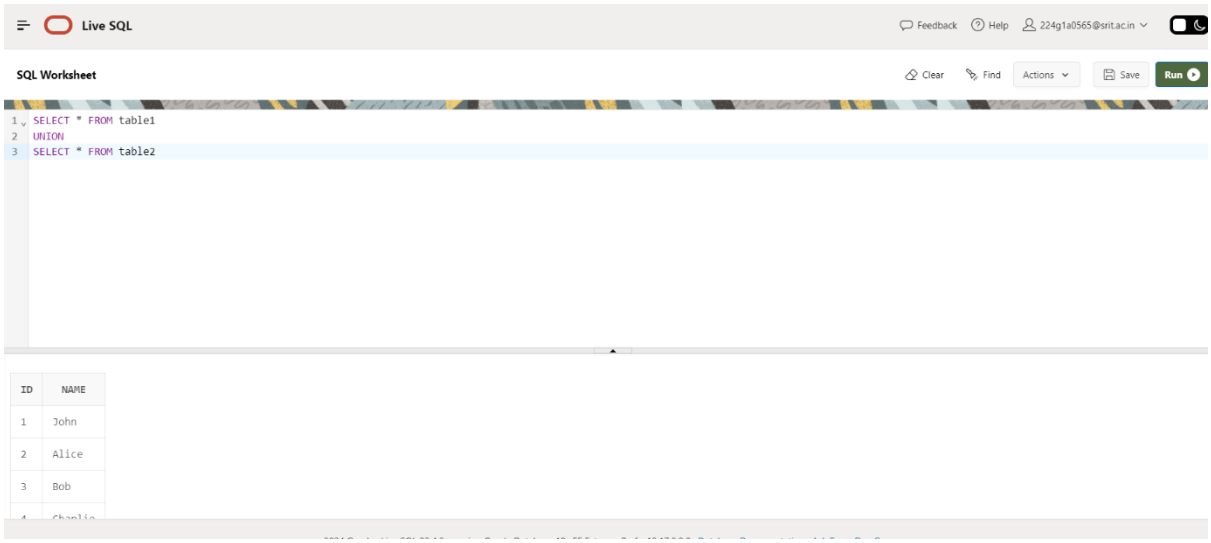

```
Table created.
```

4.Inserting VALUES into table2:

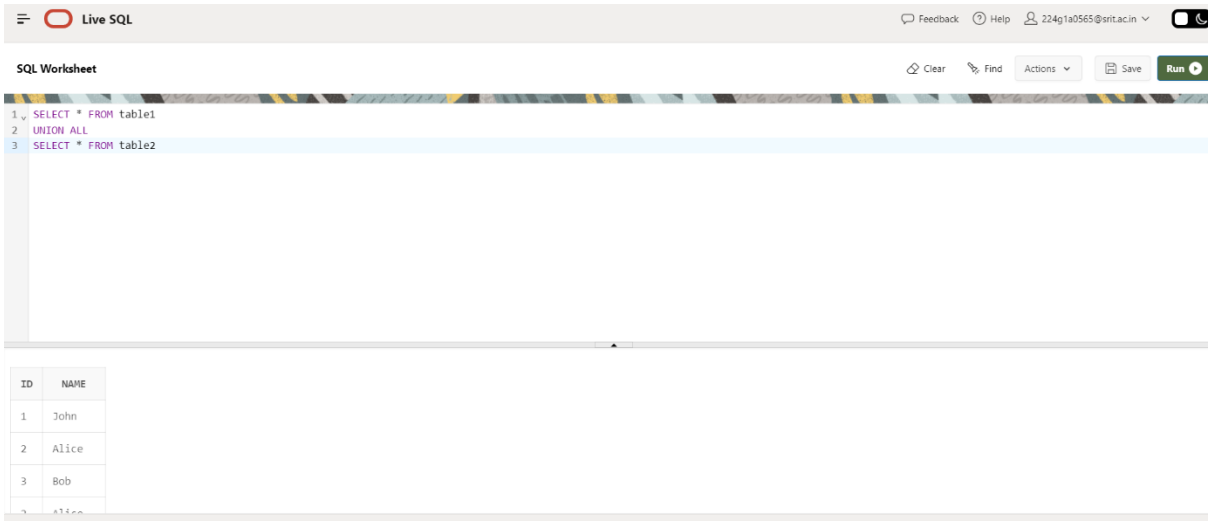
224G1A0565-K.pallavi



5.UNION:

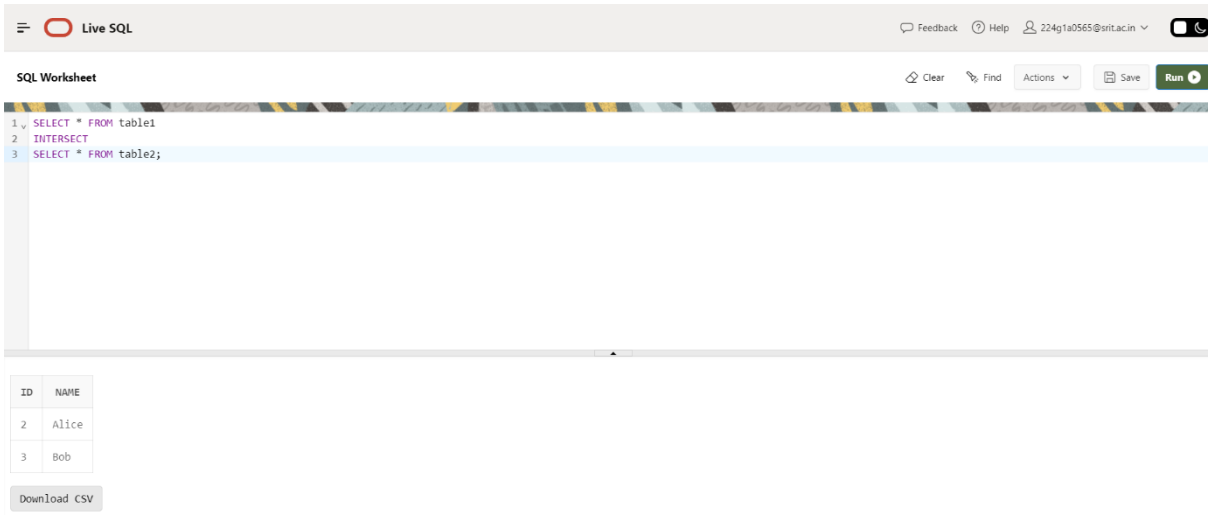


6.UNION ALL:

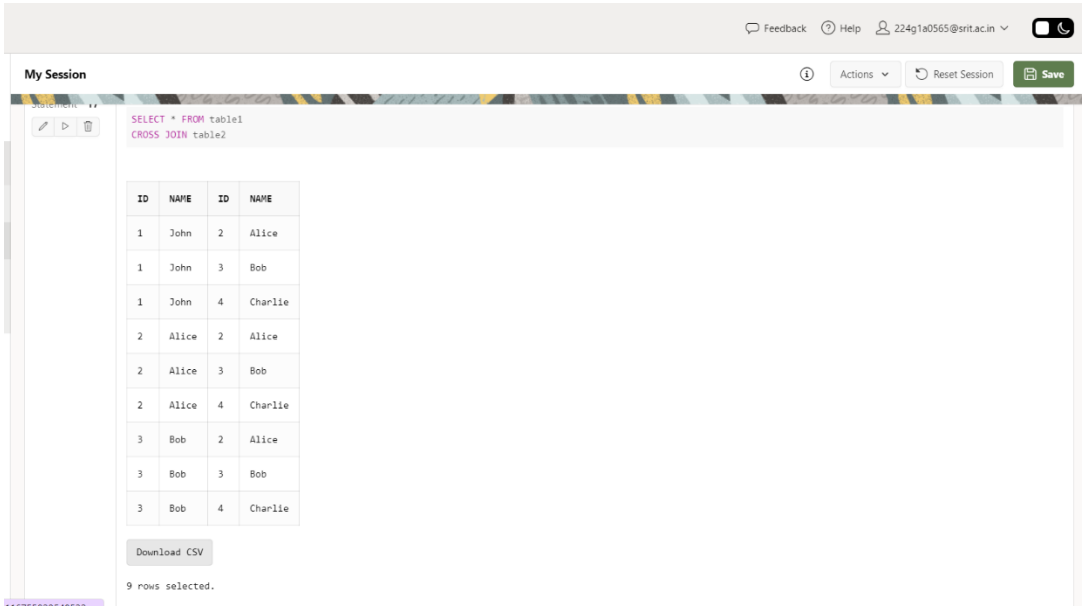


7.INTERSECT:

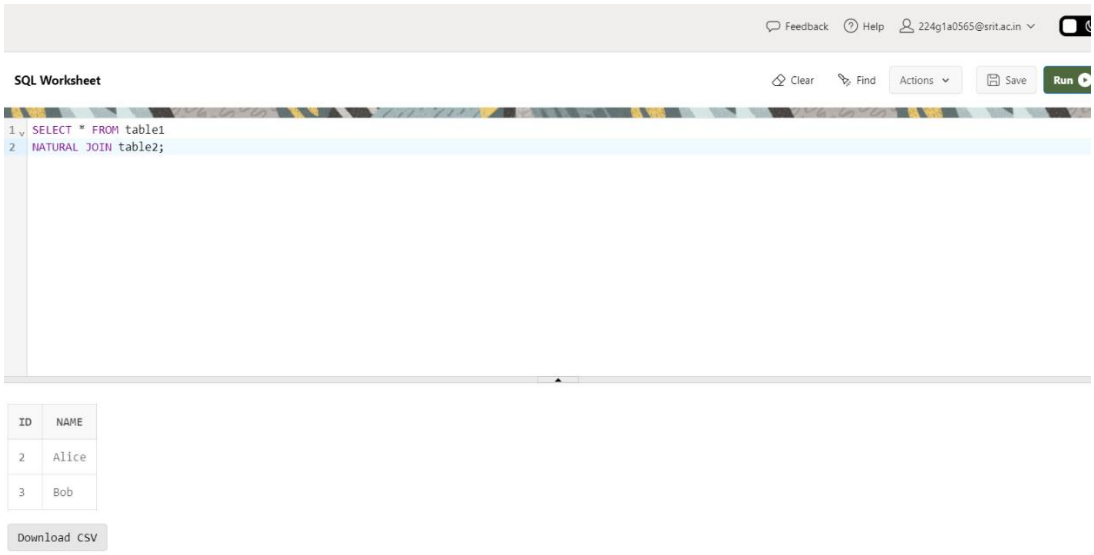
224G1A0565-K.pallavi



8.CROSS JOIN:



9.NATURAL JOIN:



conclusion:

In this lab,we successfully practiced how to perform SQL Queries on RELATIONAL OPERATIONS (UNION,UNION ALL,CROSS JOIN,NATURAL JOIN).

EXPERIMENT-5

WRITE SQL QUERIES TO PERFORM AGREGATE FUNCTIONS (count, sum, avg, min, max)

Aim:

To implement SQL QUERIES to perform Aggregate Functions (count, sum, avg, min, max)

Procedure:

```
C:\Users\kumma>sqlplus

SQL*Plus: Release 21.0.0.0.0 - Production on Thu Nov 9 11:53:35 2023
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle.  All rights reserved.
```

```
Enter user-name: system
Enter password:
Last Successful login time: Sat Oct 14 2023 10:11:50 +05:30

Connected to:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0
```

```
SQL> CREATE TABLE Emp1
  2  (eid int,
  3  eName VARCHAR(20),
  4  eSalary int);

Table created.
```

```
SQL> INSERT INTO Emp1 VALUES('1','Siddu','20000');

1 row created.

SQL> INSERT INTO Emp1 VALUES('2','shiva','30000');

1 row created.

SQL> INSERT INTO Emp1 VALUES('3','naveen','40000');

1 row created.

SQL> INSERT INTO Emp1 VALUES('3','Bhanu','50000');

1 row created.

SQL> INSERT INTO Emp1 VALUES('5','Uma','80000');

1 row created.
```

```
SQL> SELECT * from Emp1;

       EID ENAME                    ESALARY
---------- -------------------- ----------
         1 Siddu                     20000
         2 shiva                     30000
         3 naveen                    40000
         3 Bhanu                     50000
         5 Uma                       80000
```

```
SQL> SELECT avg(eid)
  2  from Emp1;

  AVG(EID)
----------
       2.8
```

```
SQL> SELECT min(eid)
  2  from Emp1;

  MIN(EID)
----------
         1
```

```
SQL> SELECT max(eid)
  2  from Emp1;

  MAX(EID)
----------
         5
```

```
SQL> SELECT count(*) eid
  2  from Emp1;

       EID
----------
         5
```

```
SQL> SELECT sum(eid)
  2  from Emp1;

  SUM(EID)
----------
        14
```

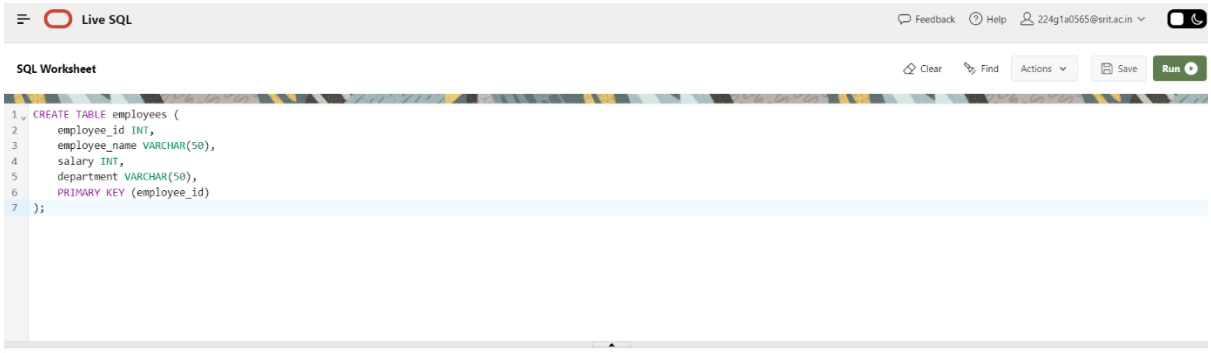
Conclusion:

In this lab, we implemented aggregate functions successfully.

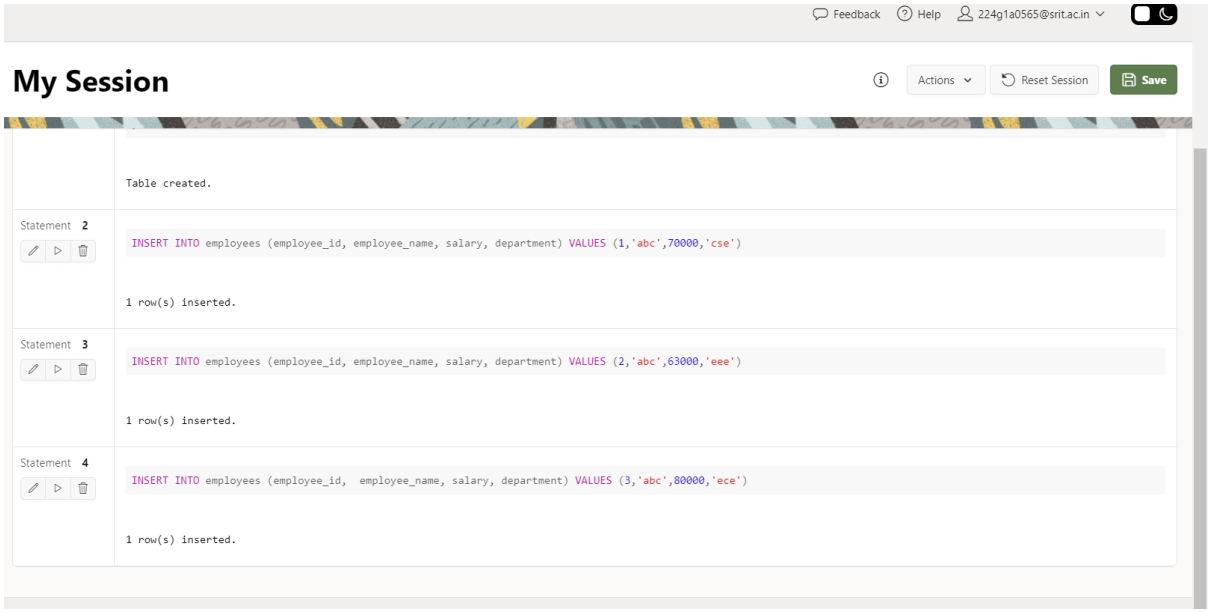
224g1a0565-k.pallavi

EXPERIMENT-6

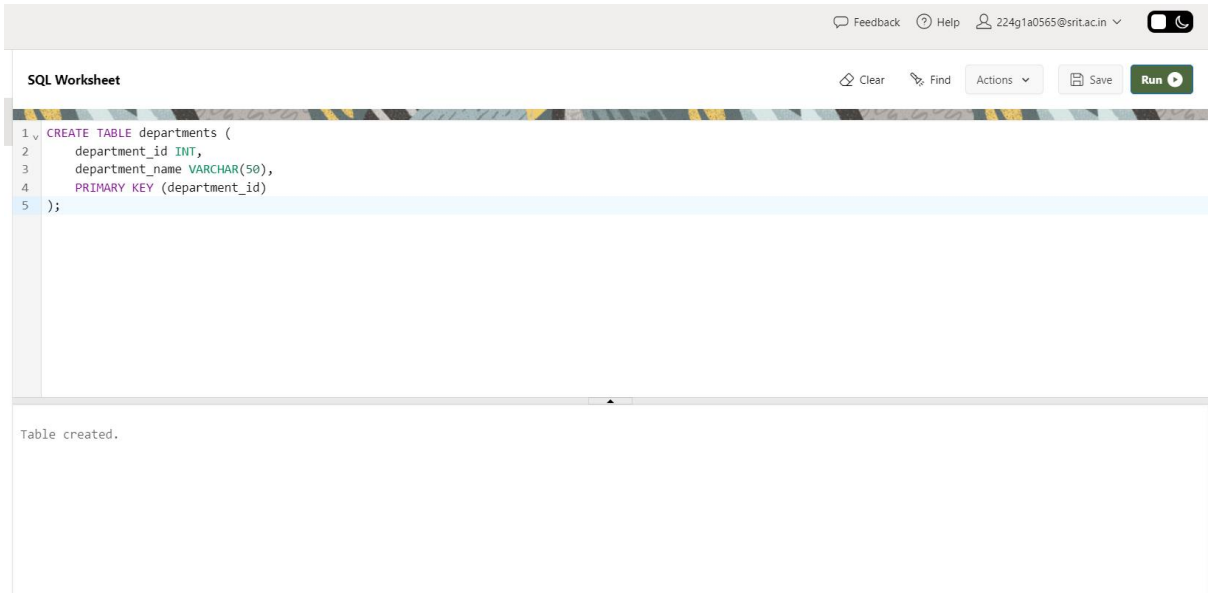
1.creating a employee table:



2.inserting VALUES into **employee**:



3.Creating a department table:



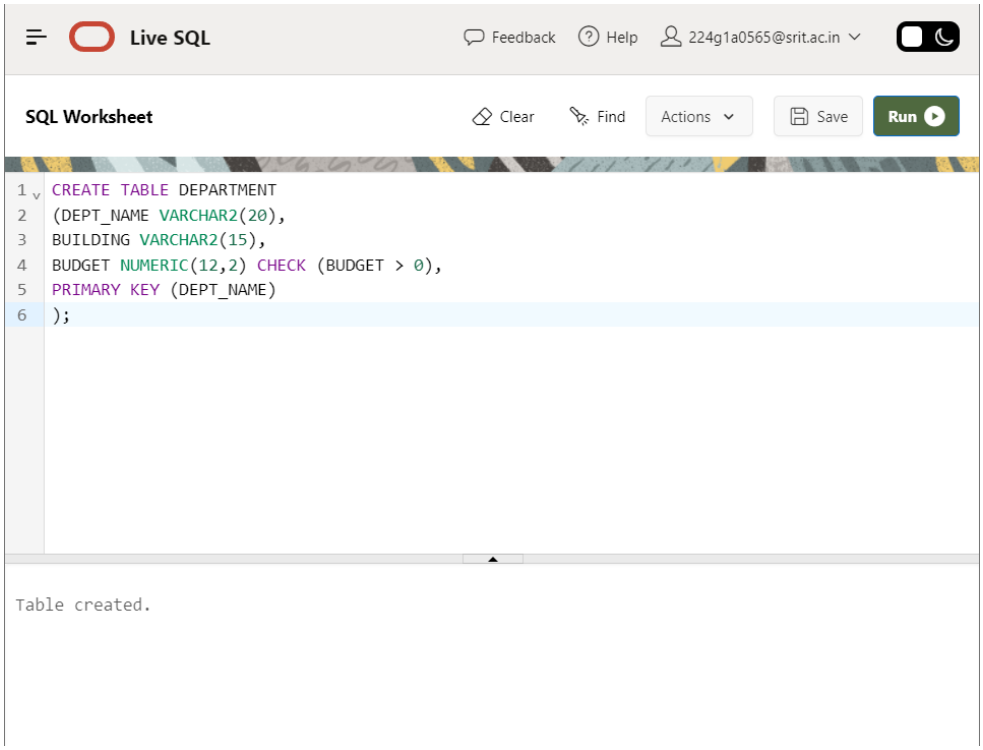
4.inserting VALUES into table :

conclusion:

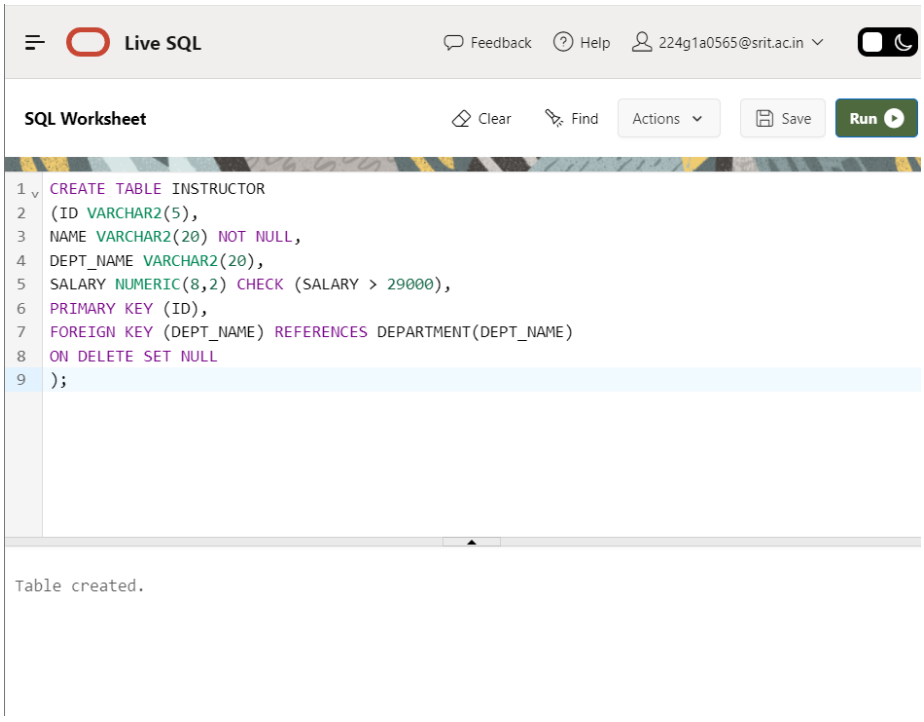
In this lab,we successfully practiced SQL Queries on JOIN OPERATIONS(CONDITIONAL JOIN,EQUI JOIN, LEFT OUTER JOIN,RIGHT OUTER JOIN).

EXPERIMENT-7

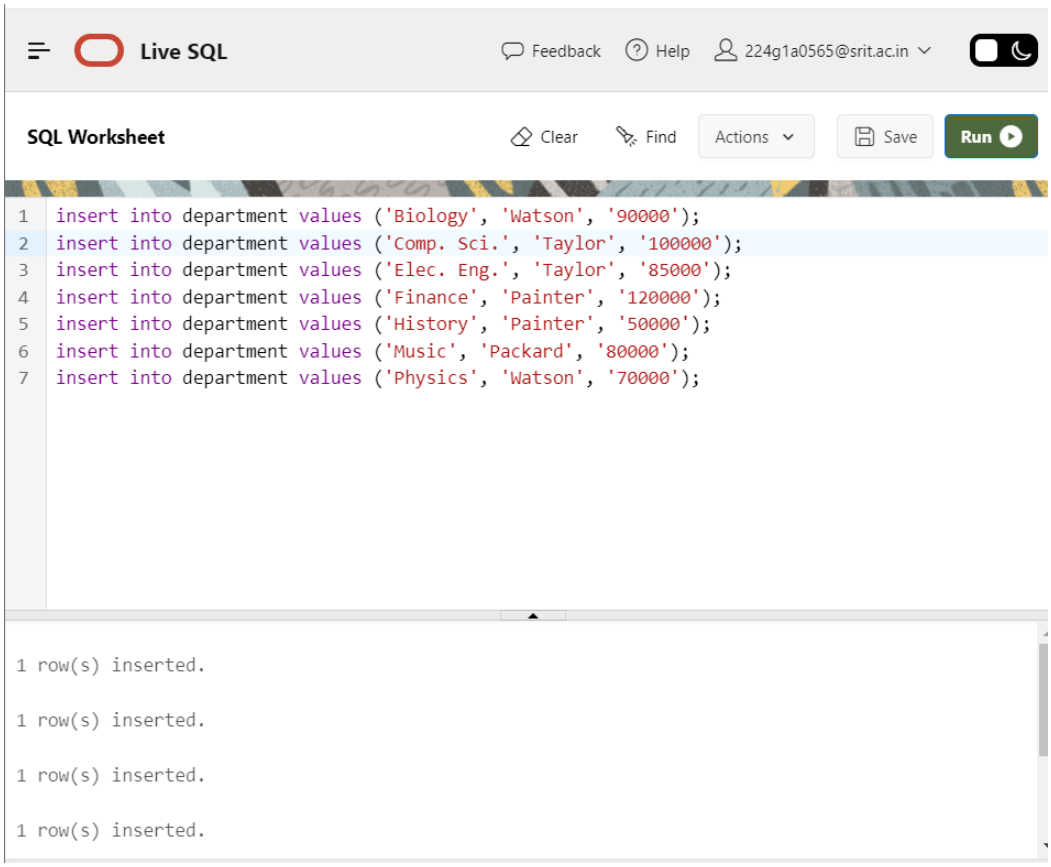
1.create department table:



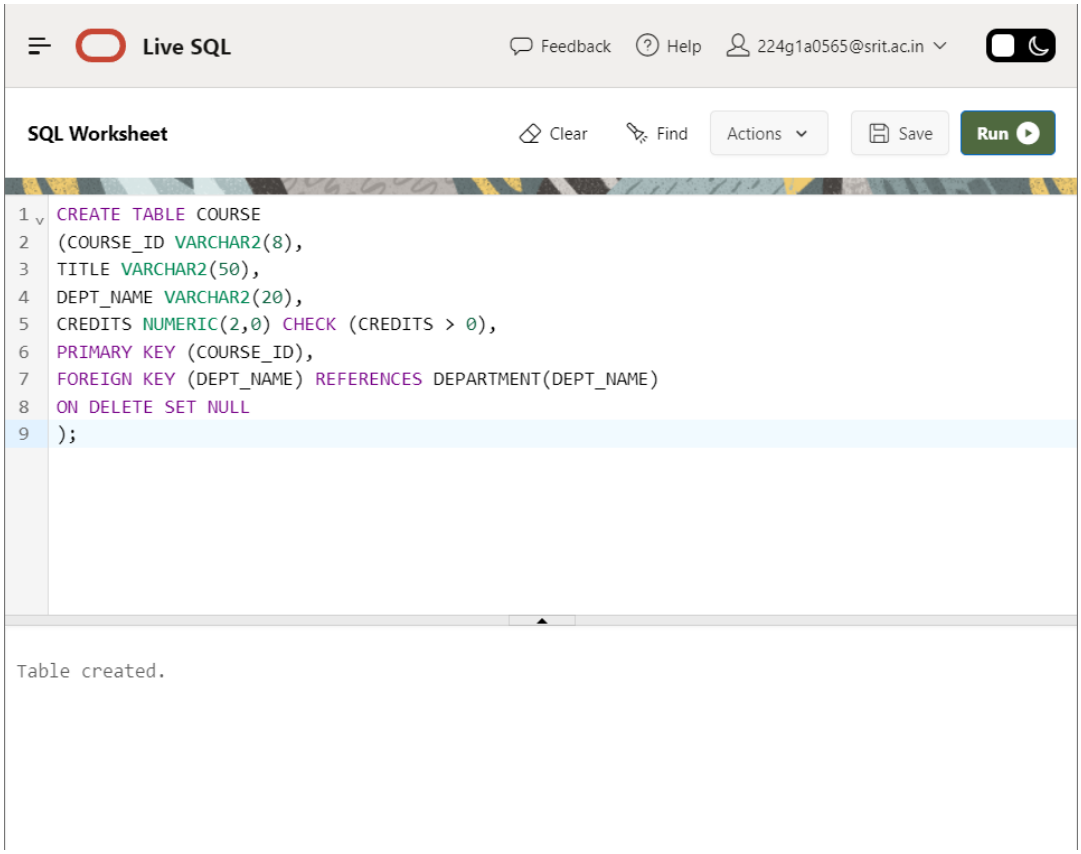
2.create instructor table:



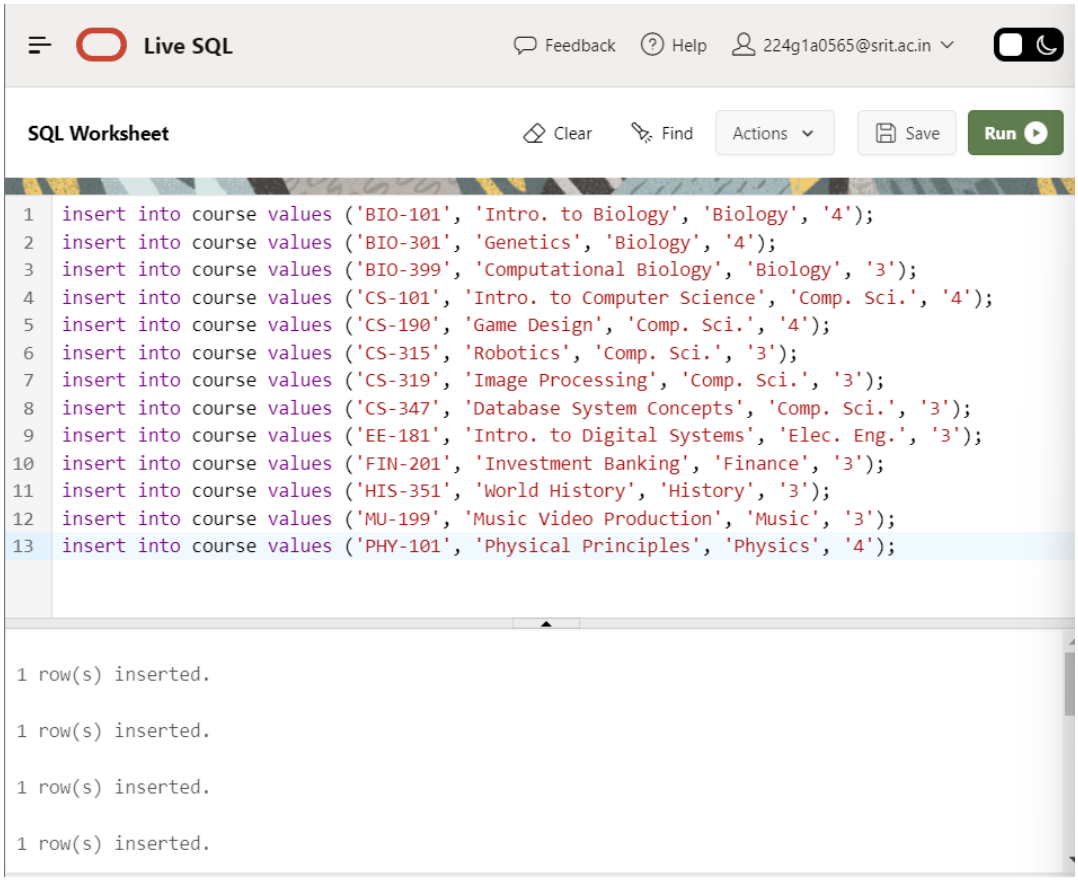
3.instances of instructor:



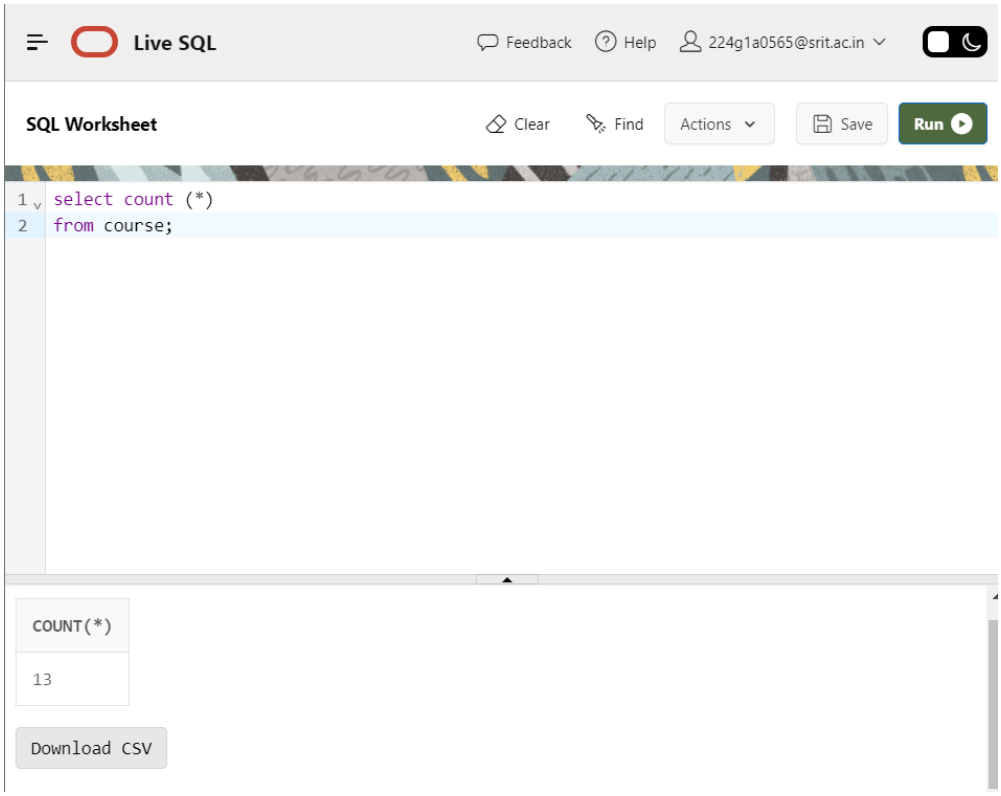
4.create course table:

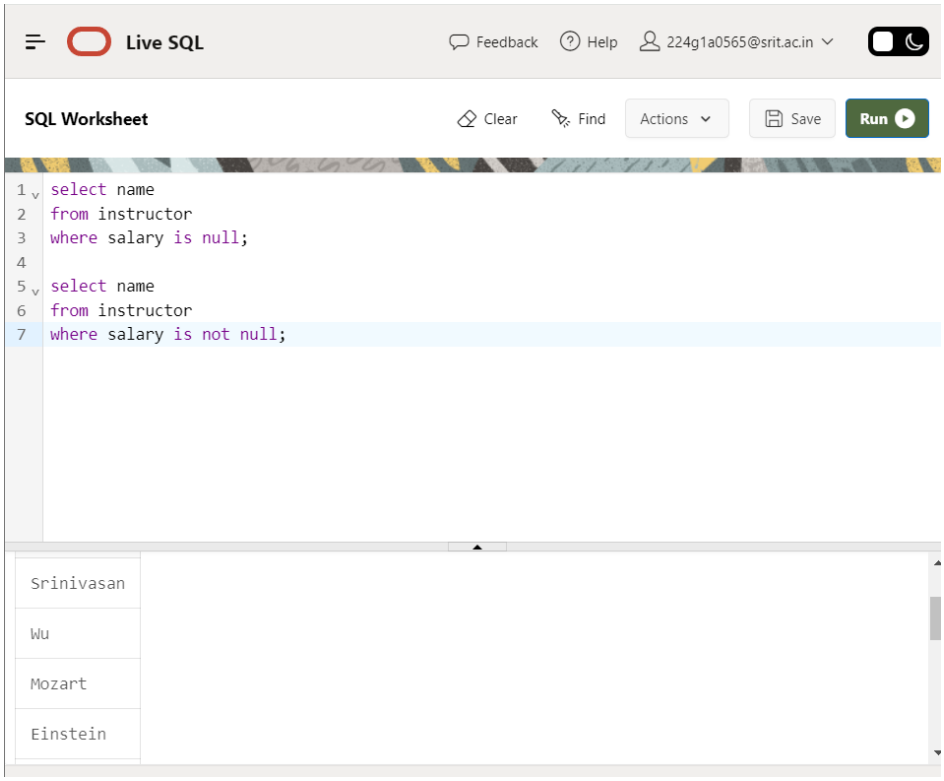


5.instances of COURSE table:



To find no.of course taught in the university:





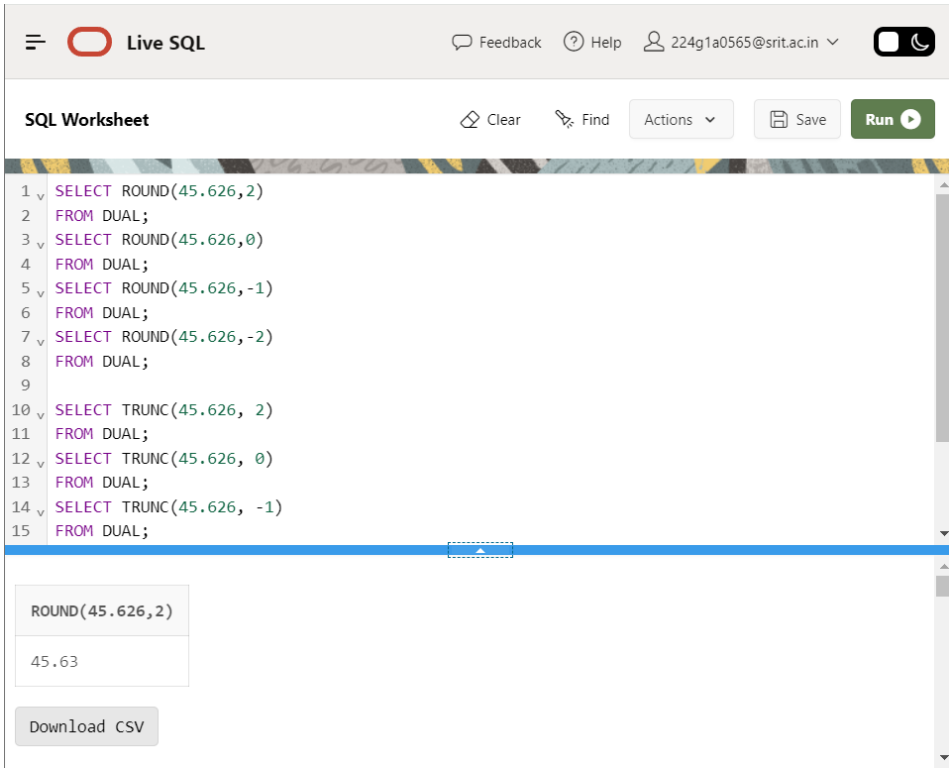
Conclusion:

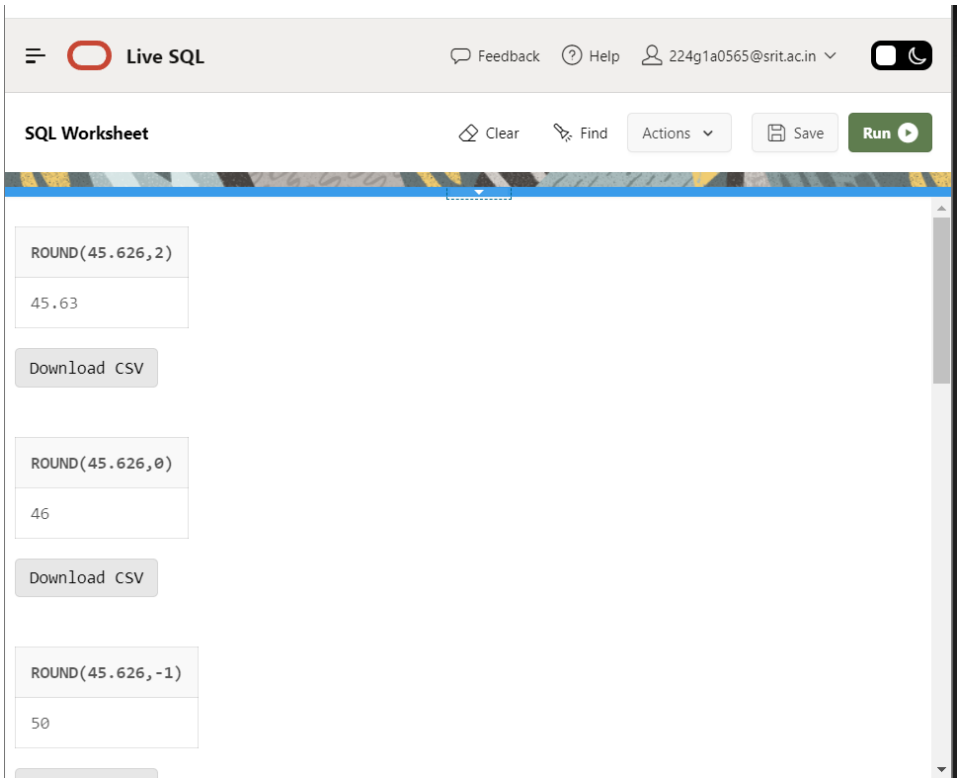
In this lab,we successfully practiced SQL Queries to perform AGGREGATE operations.

224g1a0565-k.pallavi

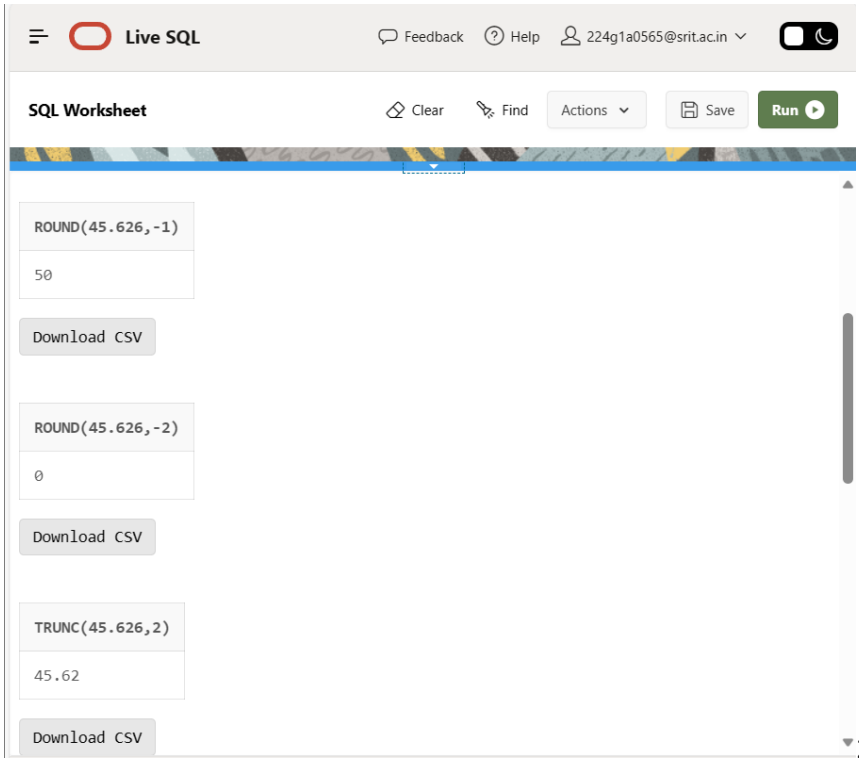
EXPERIMENT-8

1.NUMBER FUNCTIONS:

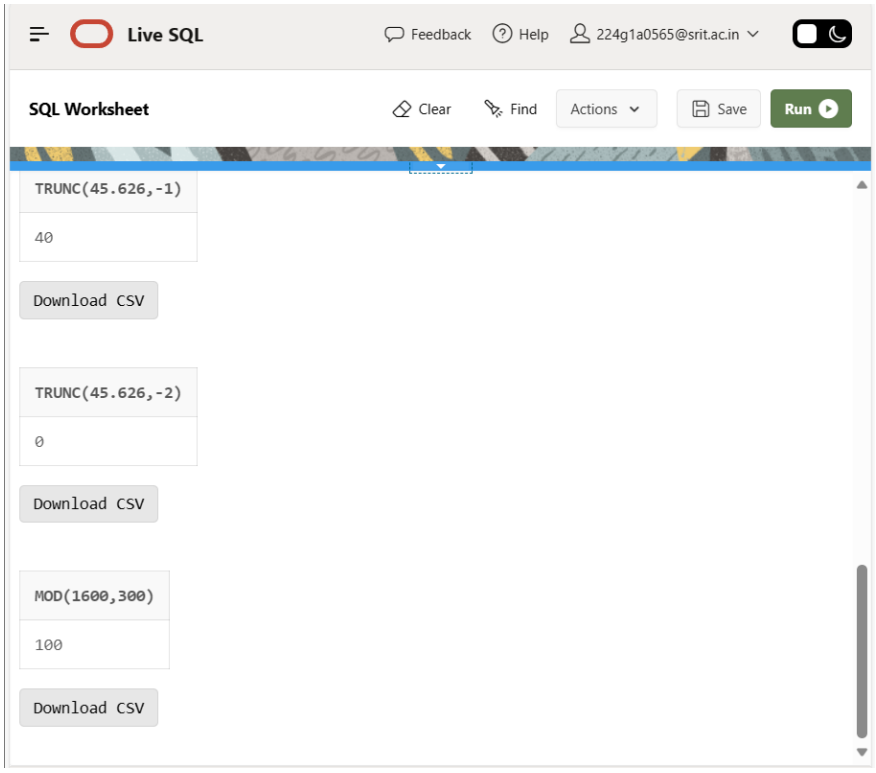




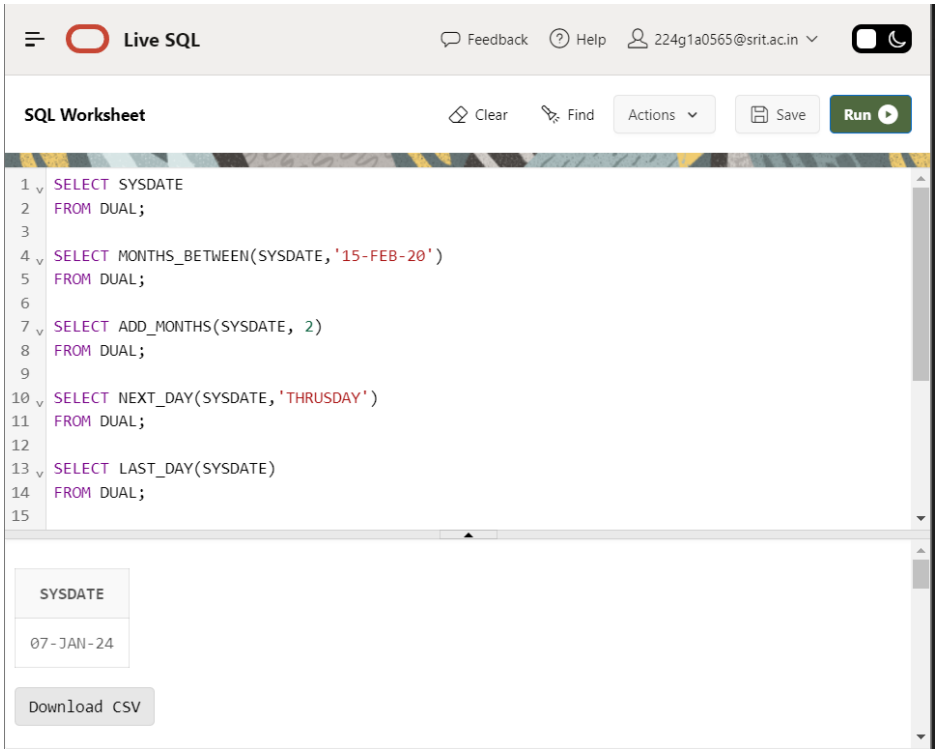
224g1a0565-k.pallavi

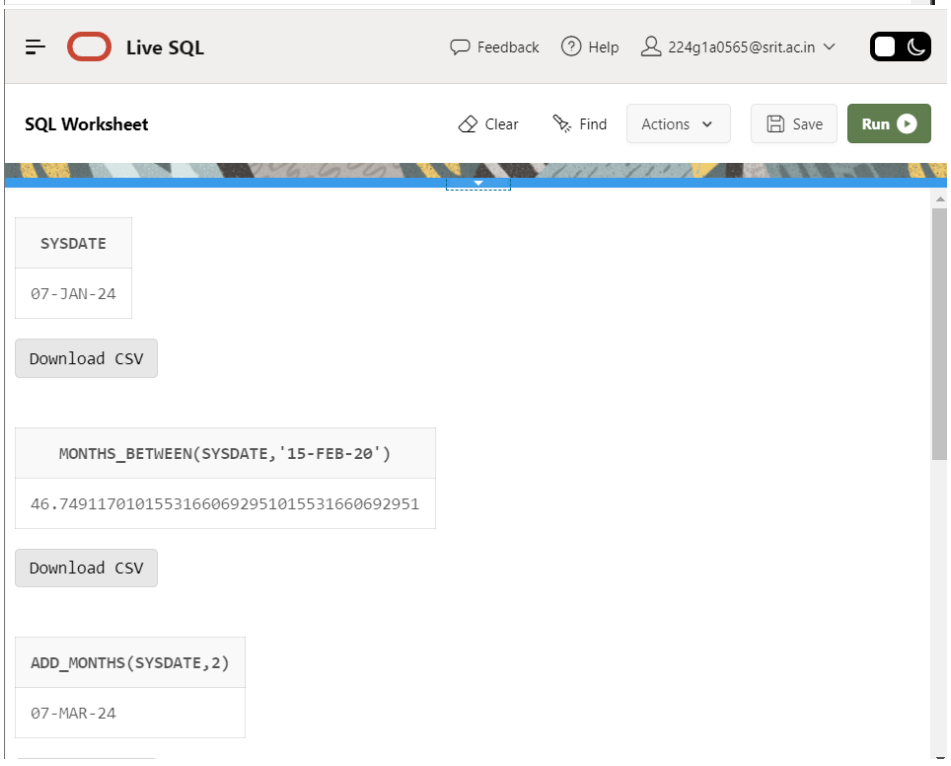


2.D



2.Date Functions:





Conclusion:

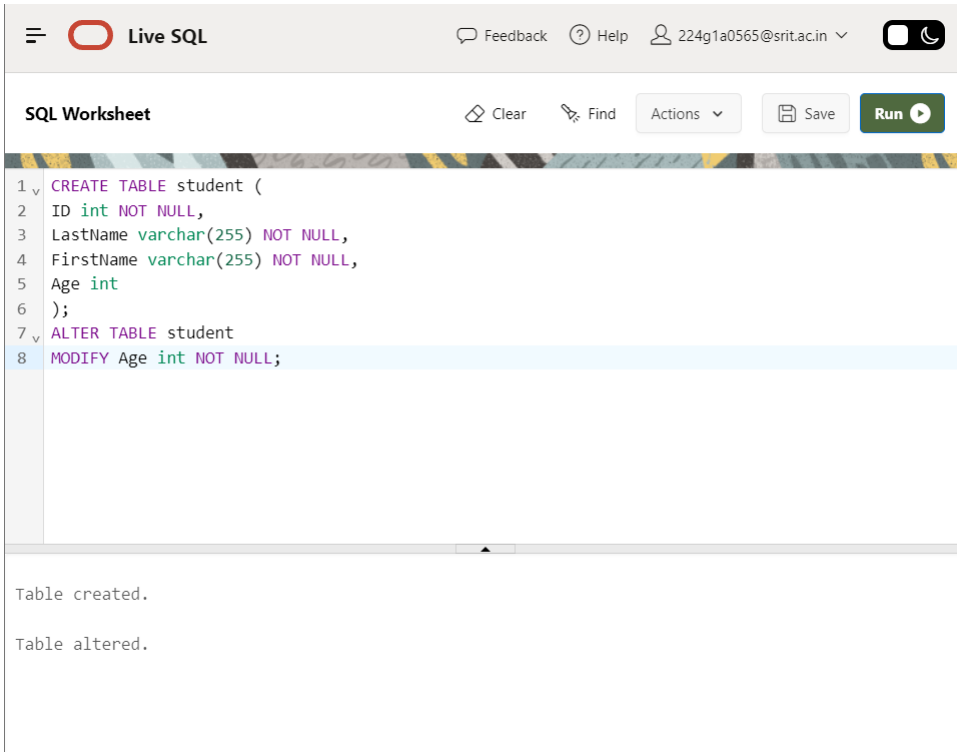
In this lab,we successfully practiced SQL Queries to perform ORACLE BUILT-IN Function.

224g1a0565-k.pallavi

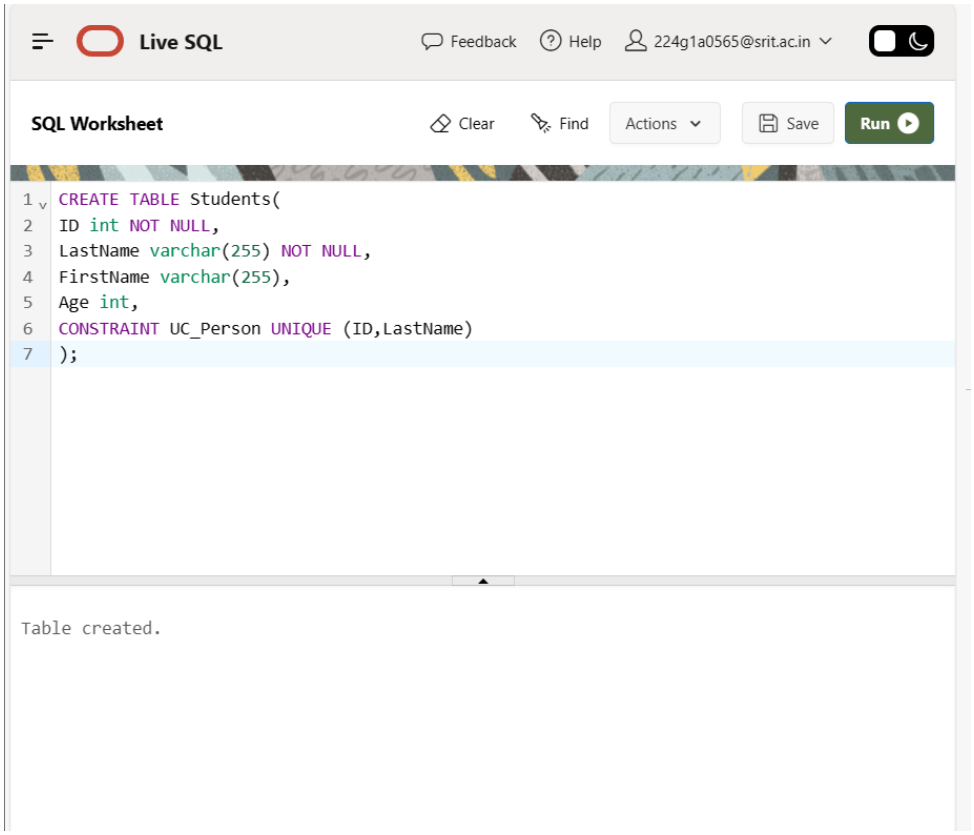
224g1a0565-k.pallavi

EXPERIMENT-9

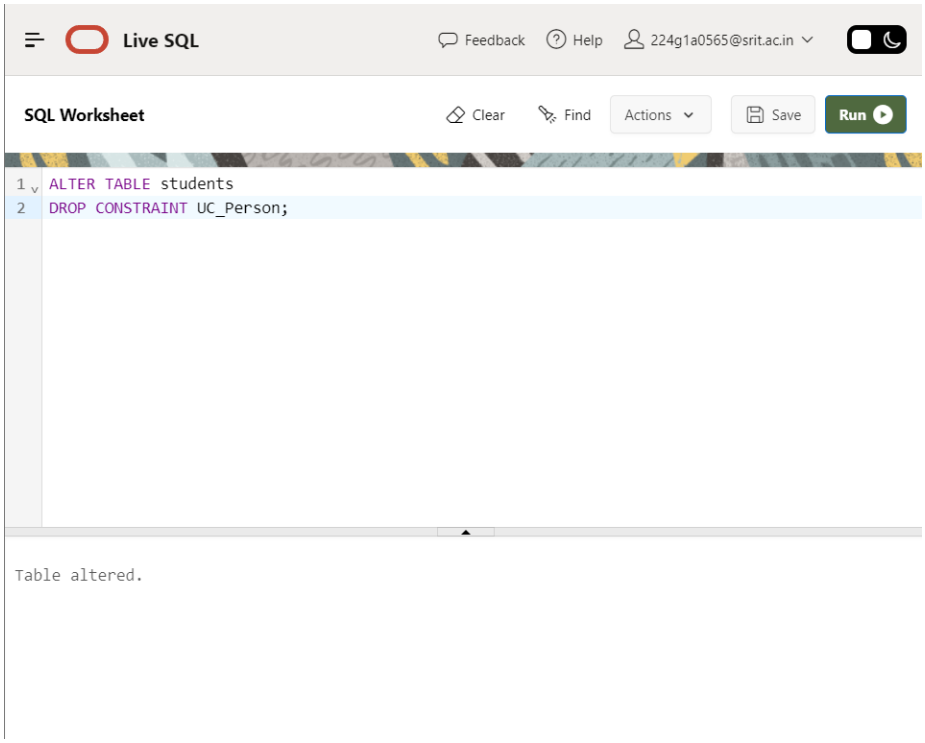
1.NOT NULL CONSTRAINT Example:

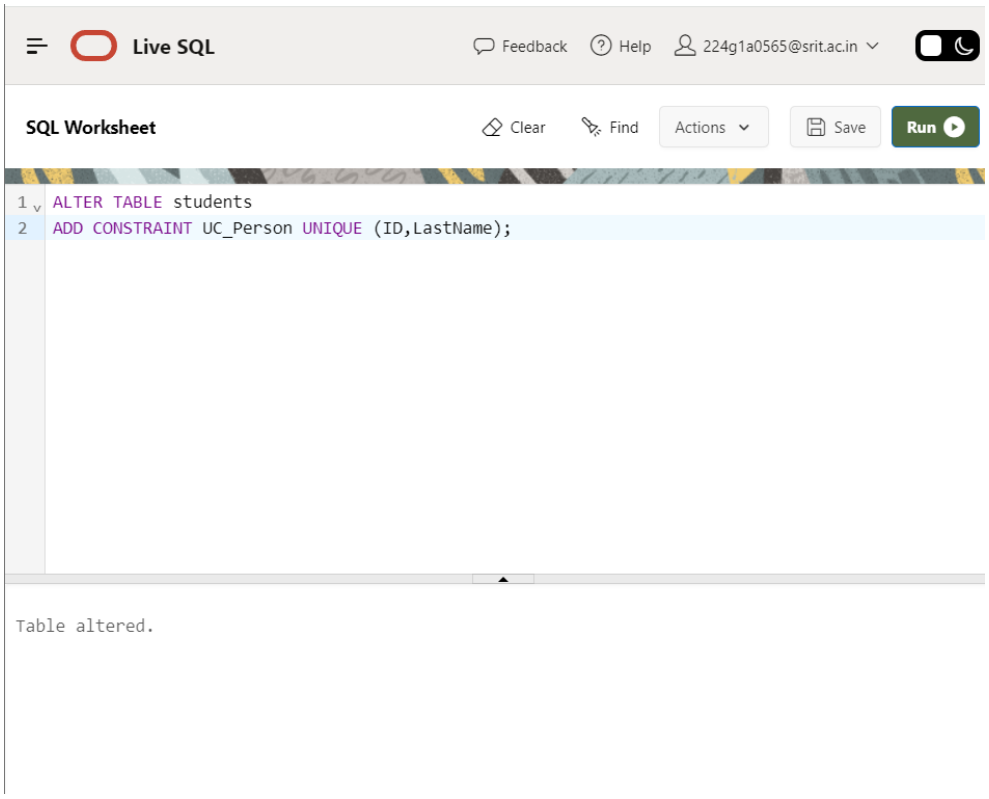


2.UNIQUE CONSTRAINT Example:



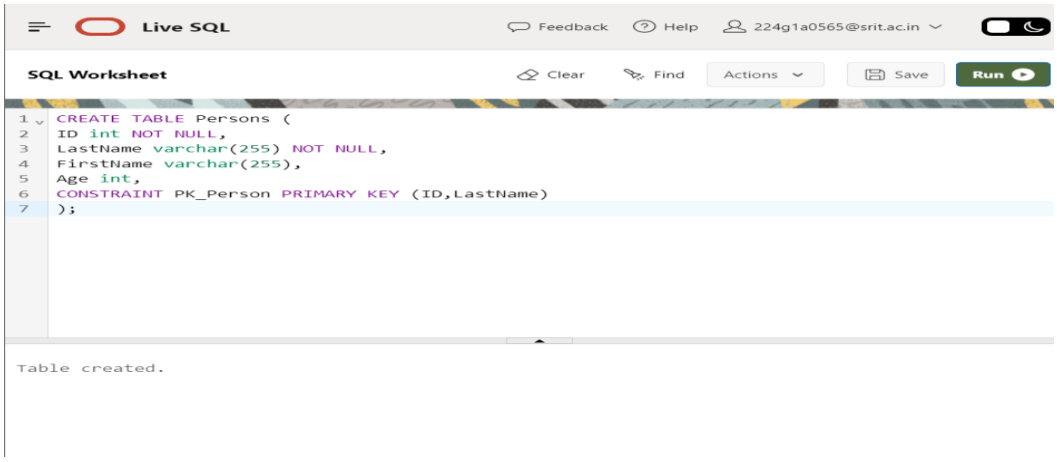
224g1a0565-k.pallavi

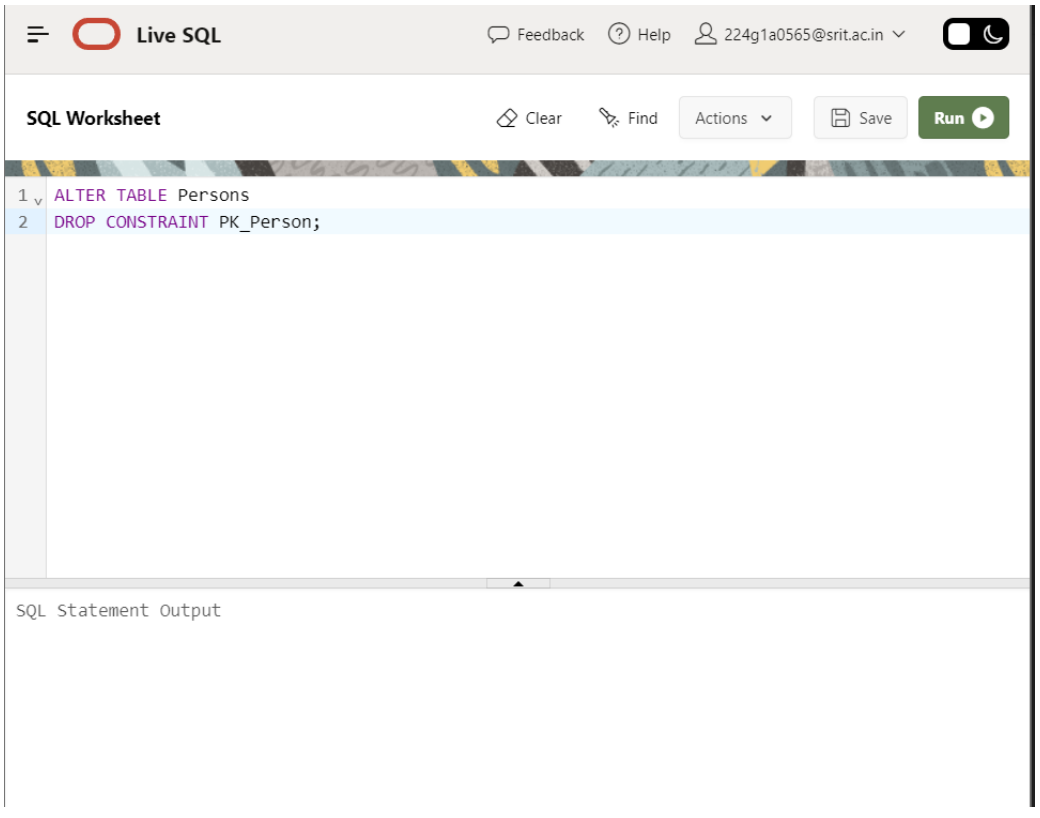




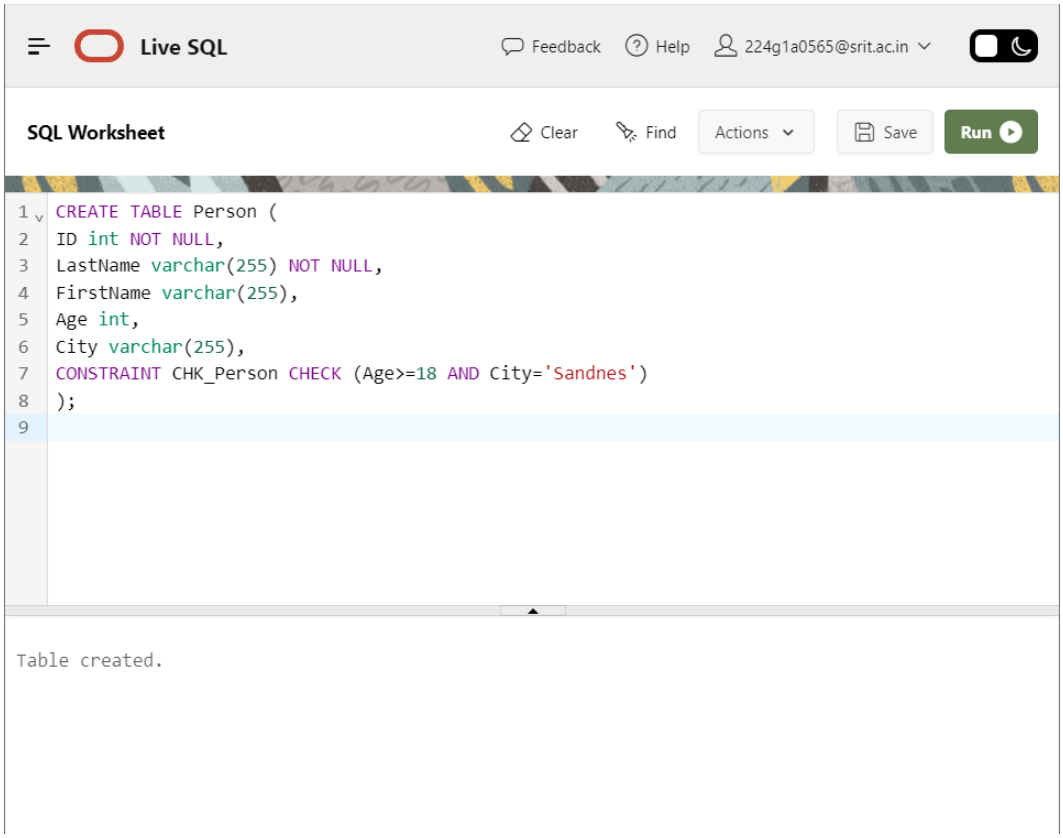
3.PRIMARY KEY CONSTRAINT Example:

224g1a0565-k.pallavi





4.DEFAULT CONSTRAINTS Example:

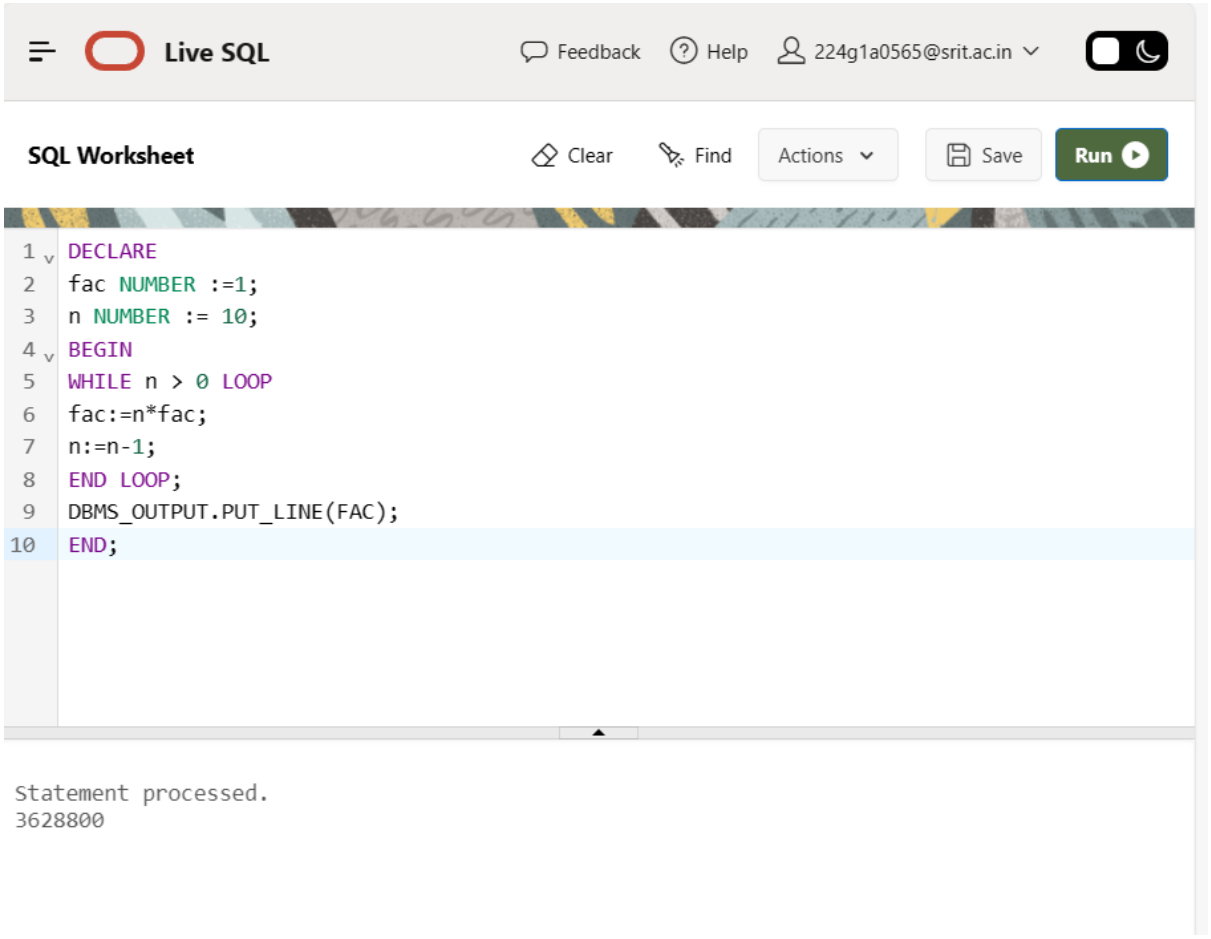


Conclusion:

In this lab, we successfully executed SQL Queries to perform KEY CONSTAINTS.

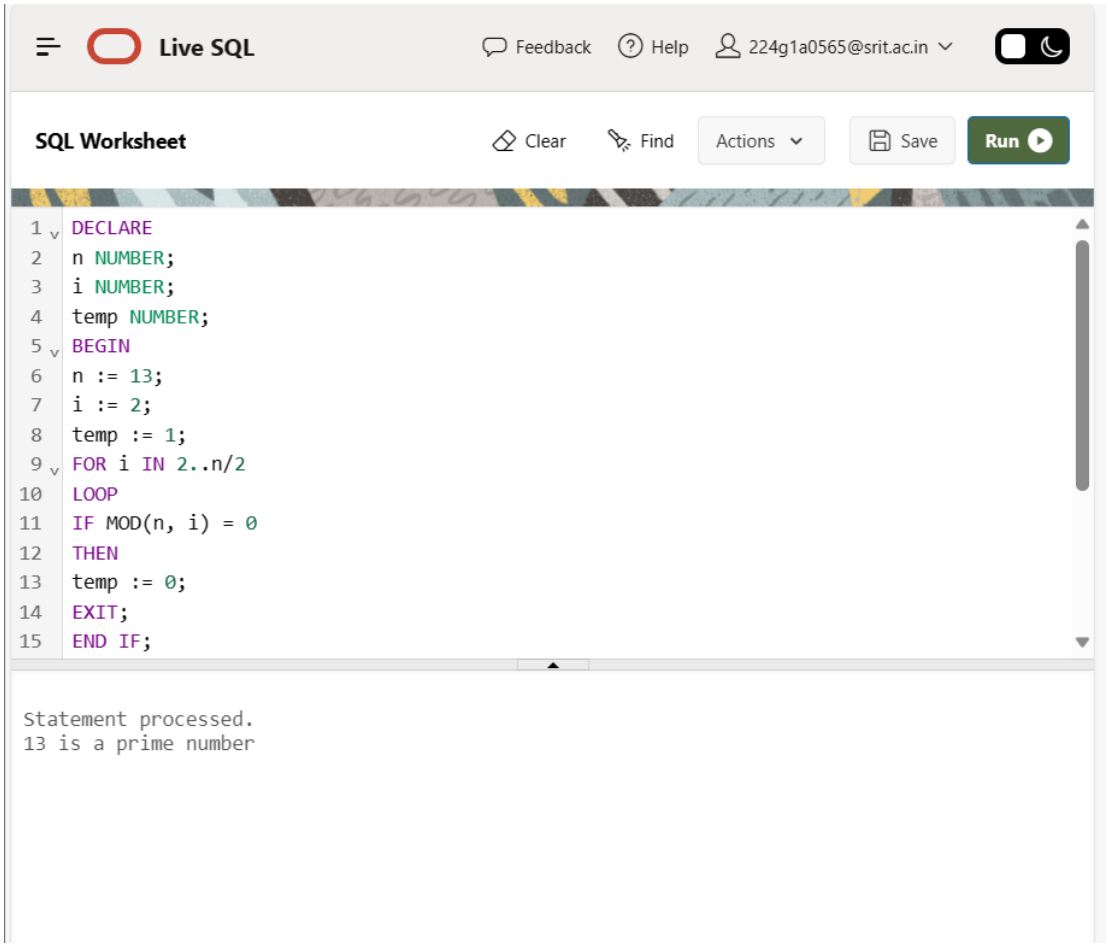
EXPERIMENT-10

1.Factorial Program



EXPERIMENT-11

1.Prime number or not Program:

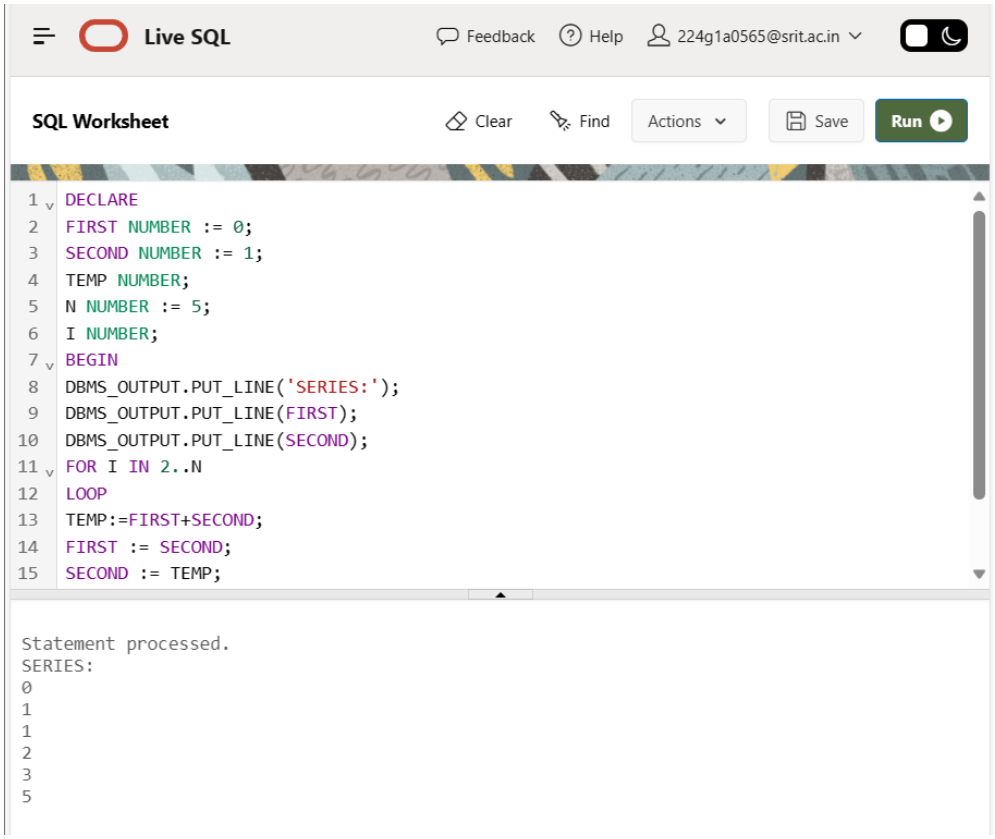


conclusion:

In this lab,we successfully practiced a program for Finding the given number is prime or not.

EXPERIMENT-12

1.Fibonacci Series Program:

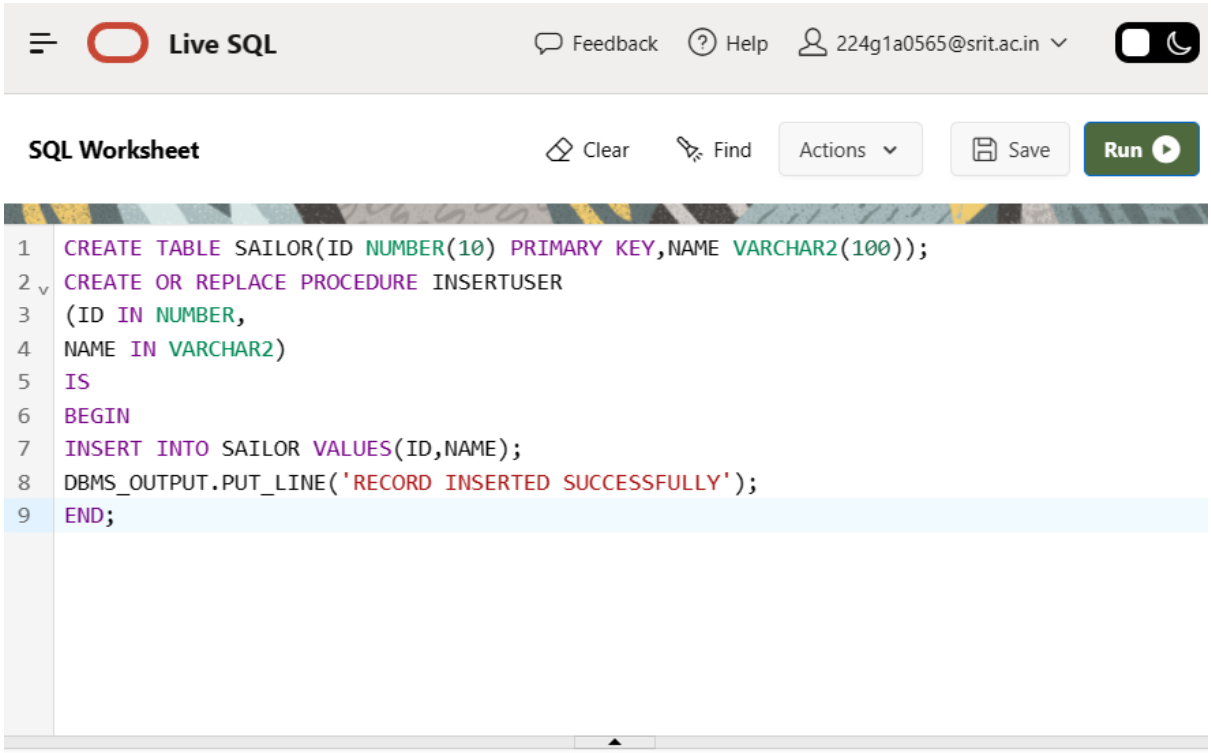


Conclusion :

In this lab,we Successfully practiced a program for displaying Fibonacci Series up to an integer.

Experiment-13

1.Create table:

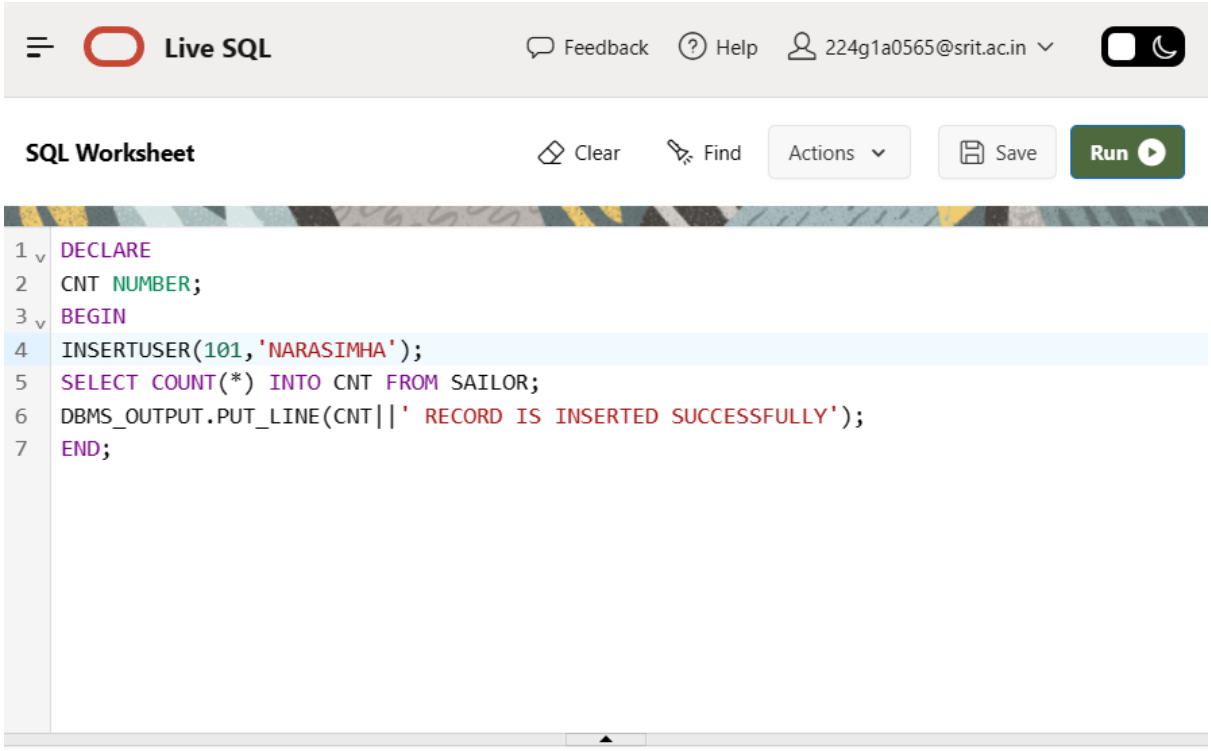



```
Table created.

Procedure created.
```

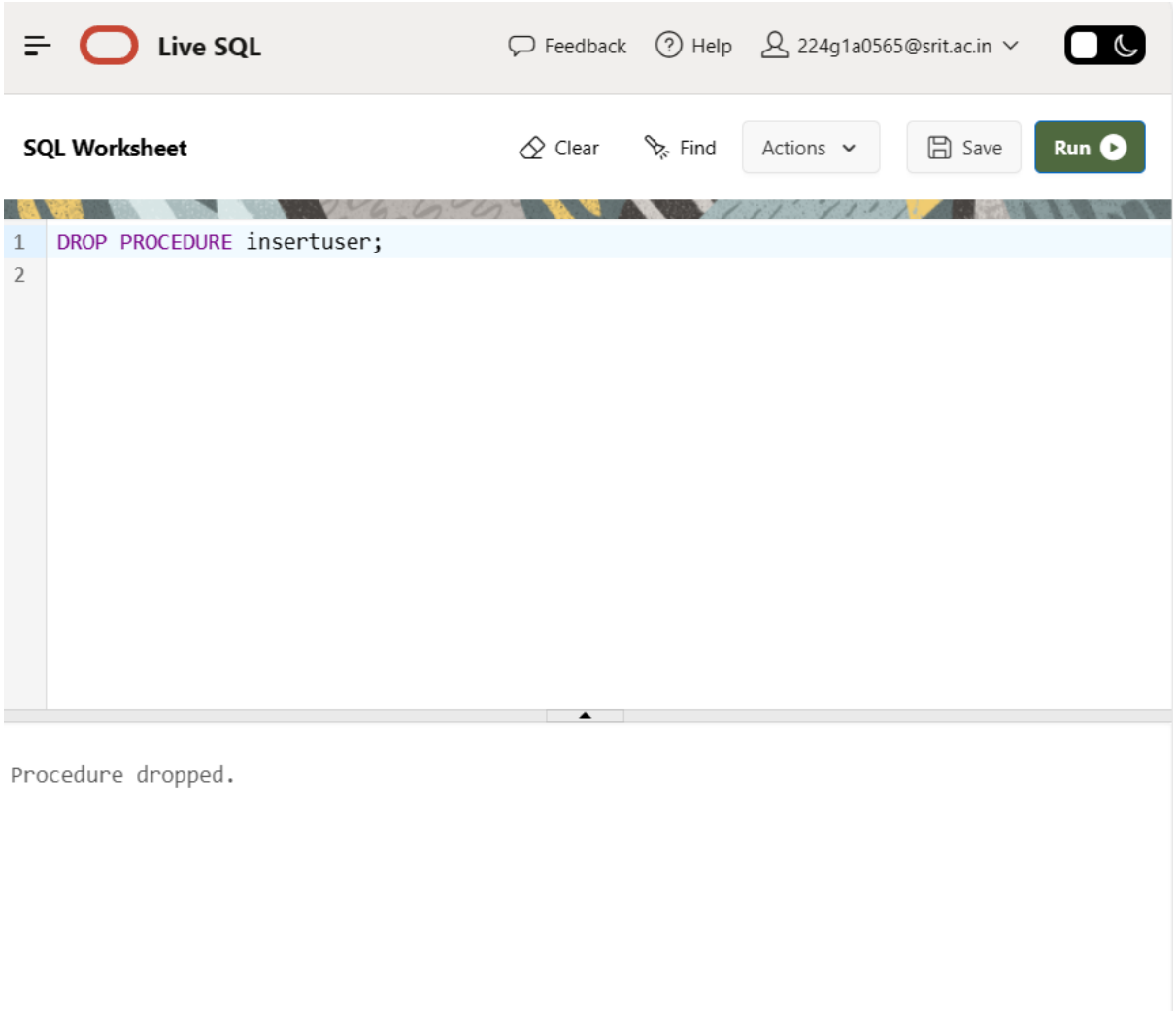
2.Execution Procedure:

224g1a0565-k.pallavi




```
Statement processed.
RECORD INSERTED SUCCESSFULLY
1 RECORD IS INSERTED SUCCESSFULLY
```

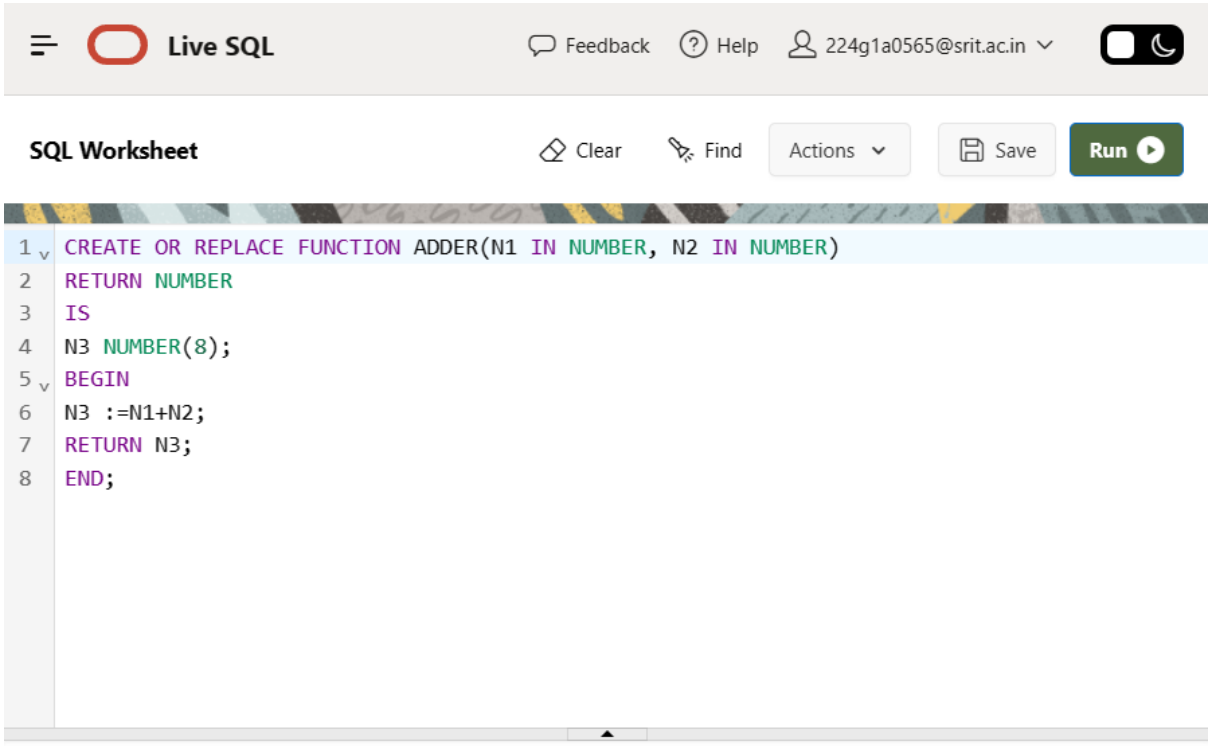
3.Drop Procedure:



Conclusion: In this lab, we successfully practiced How to write a program to implement Stored Procedure on table.

Experiment-14

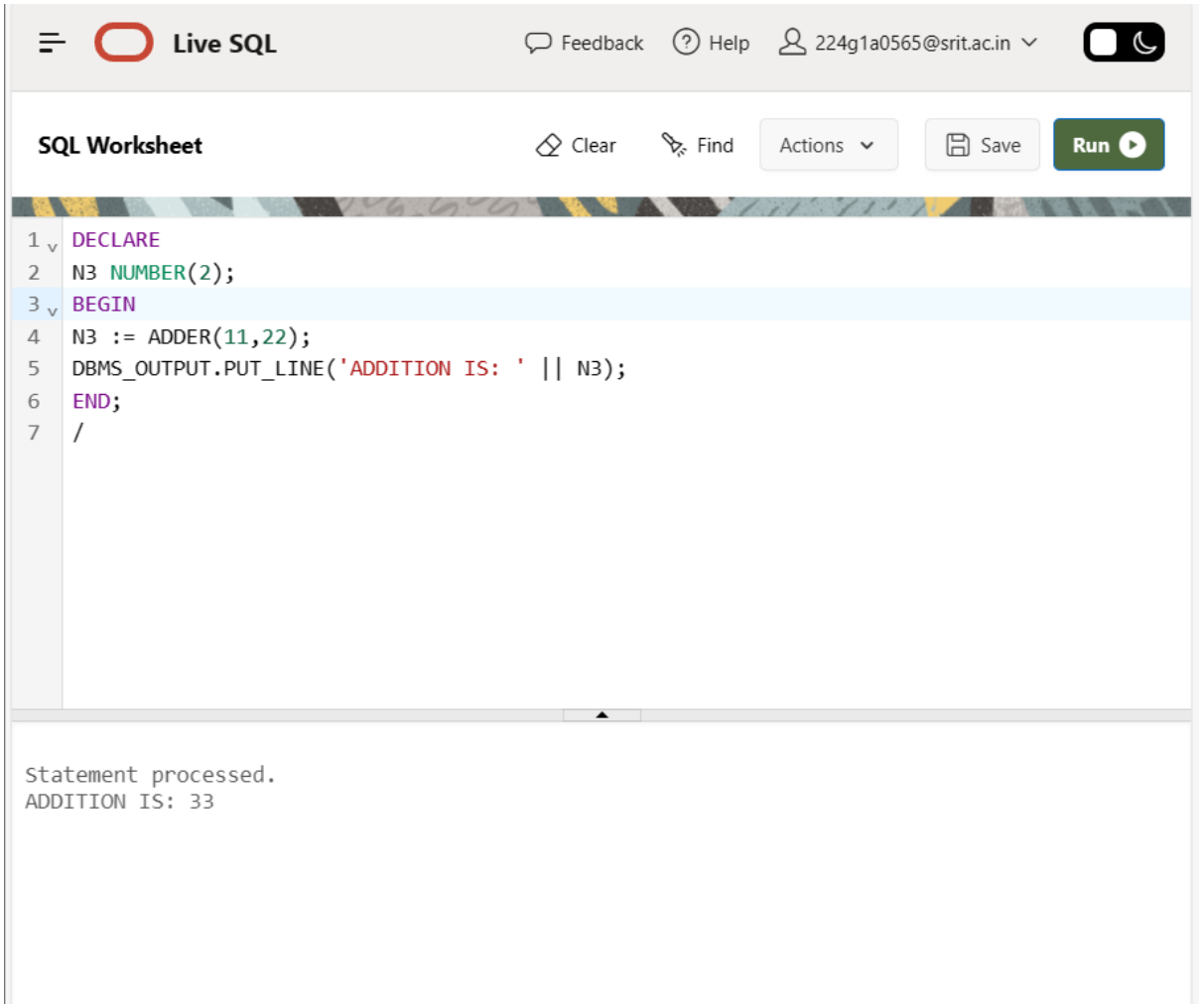
1.PL/SQL Function:




```
Function created.
```

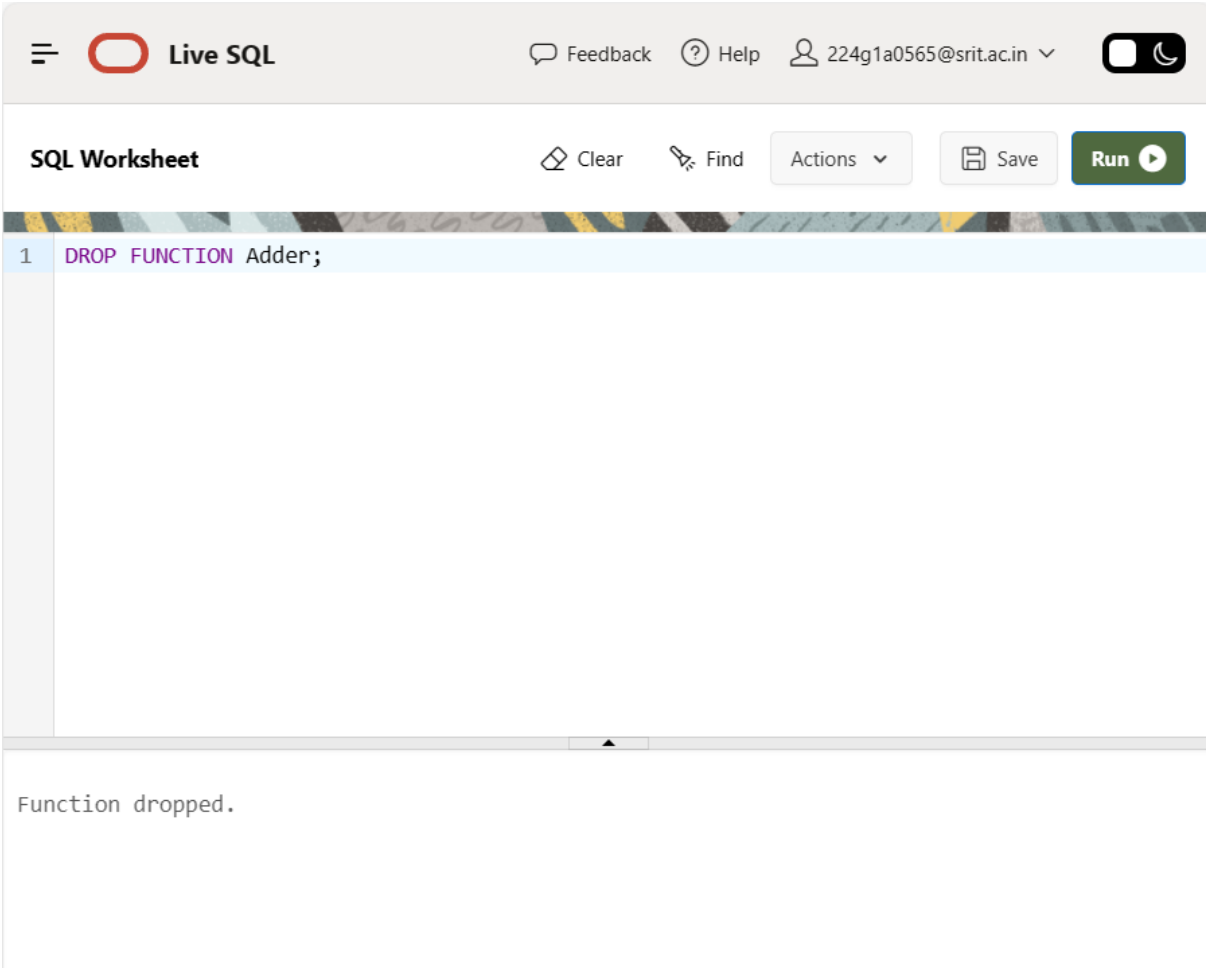
2.Execution Procedure:

224g1a0565-k.pallavi



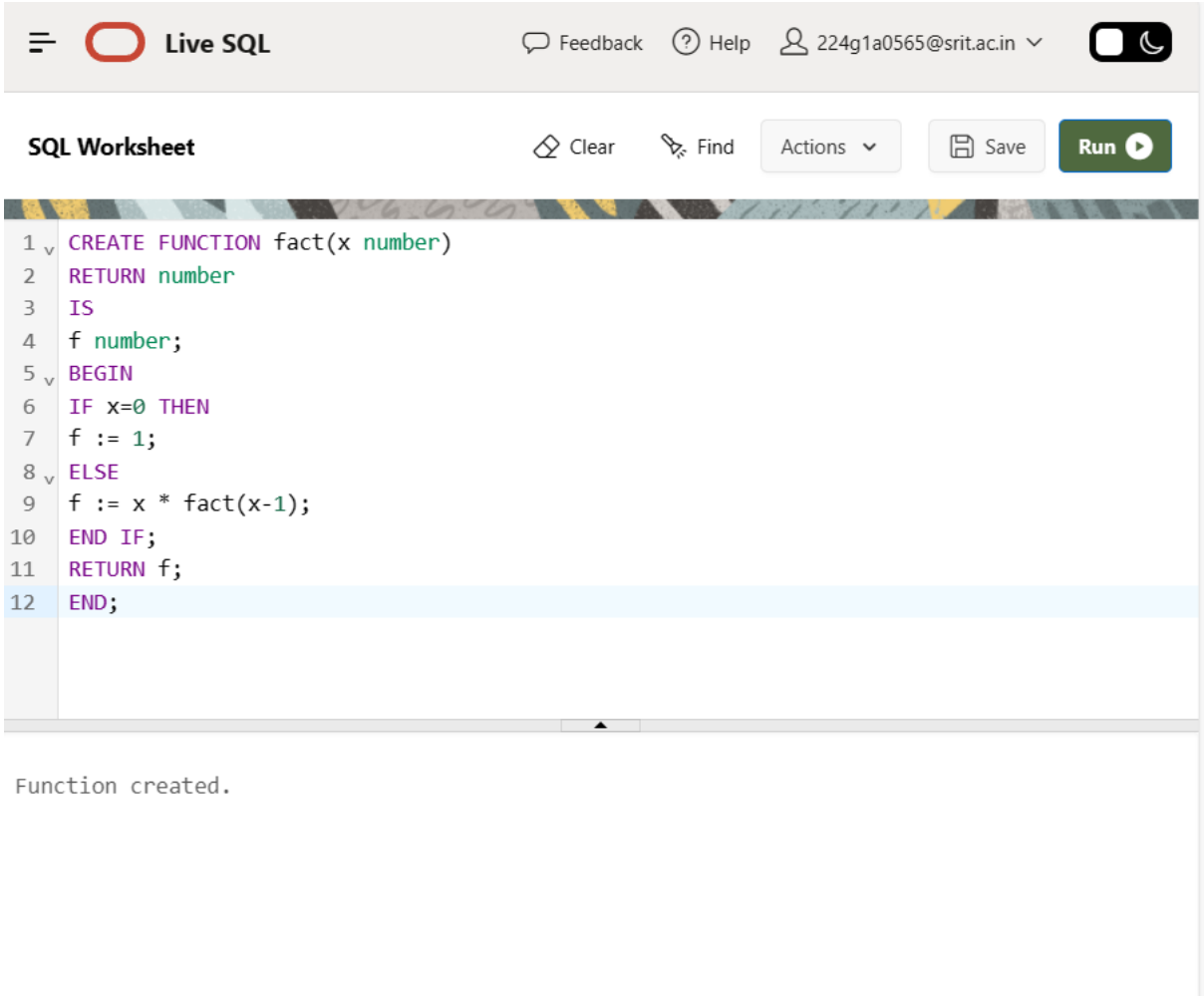
3.Drop Procedure:

224g1a0565-k.pallavi

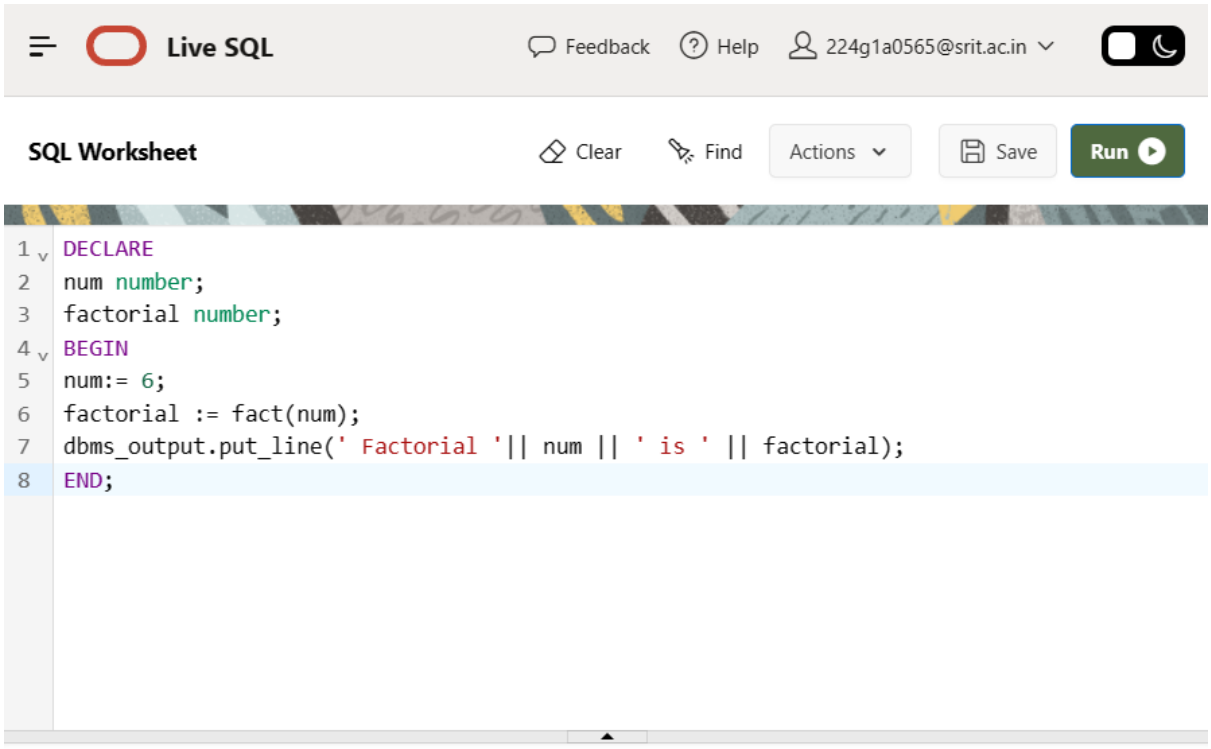


4.Recursive Function:

224g1a0565-k.pallavi

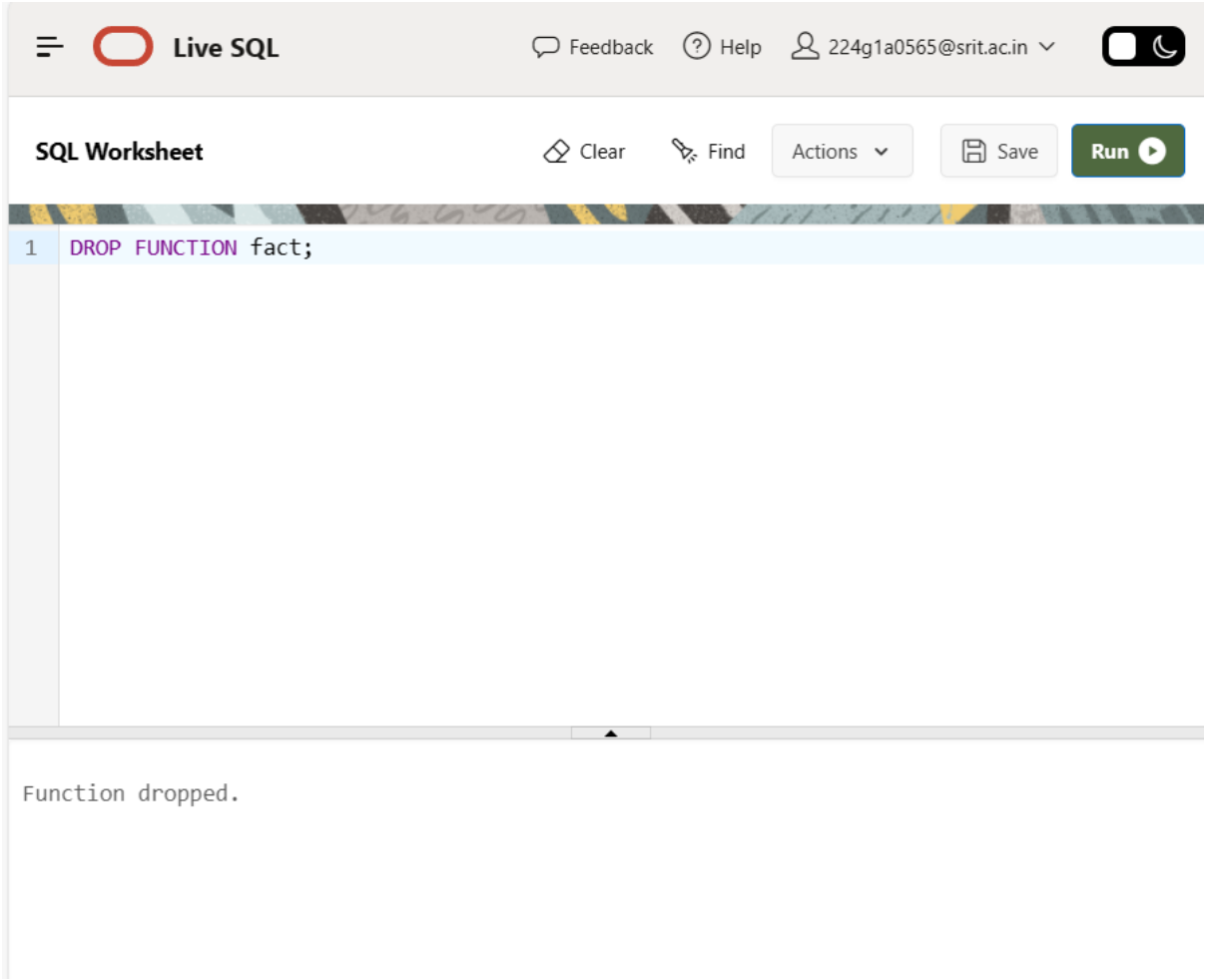


5.Execution Procedure:




```
Statement processed.
 Factorial 6 is 720
```

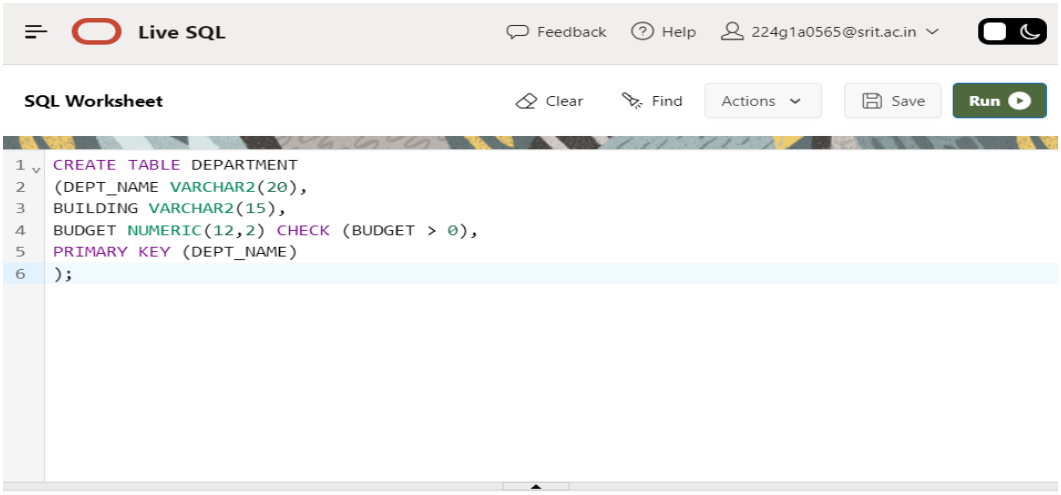
6.Drop Function:



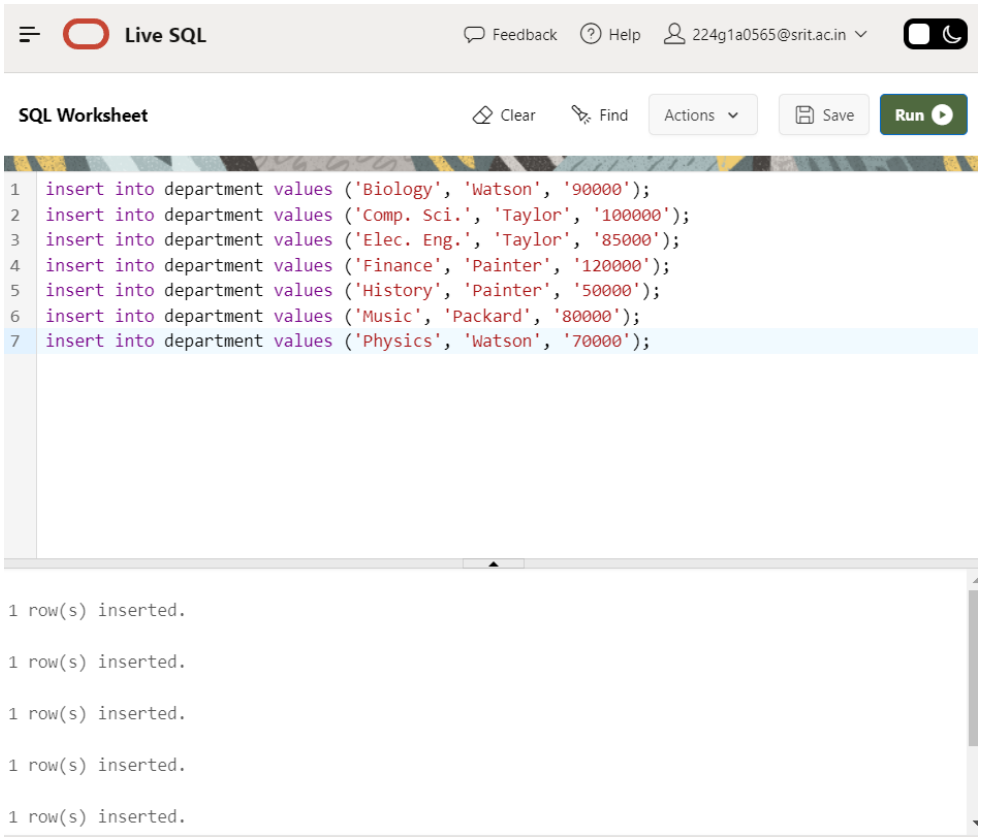
Conclusion: In This Lab, we successfully practiced PL/SQL Program to implement Stored Function on Table

Experiment-15

1.Create Department Table



2.Instances of Department Table



3.Create Instructor Table

224g1a0565-k.pallavi

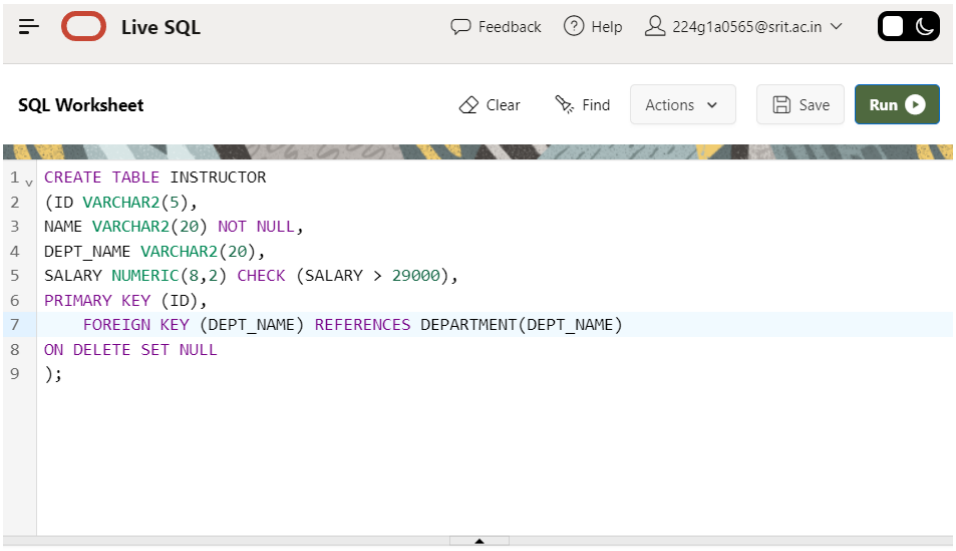
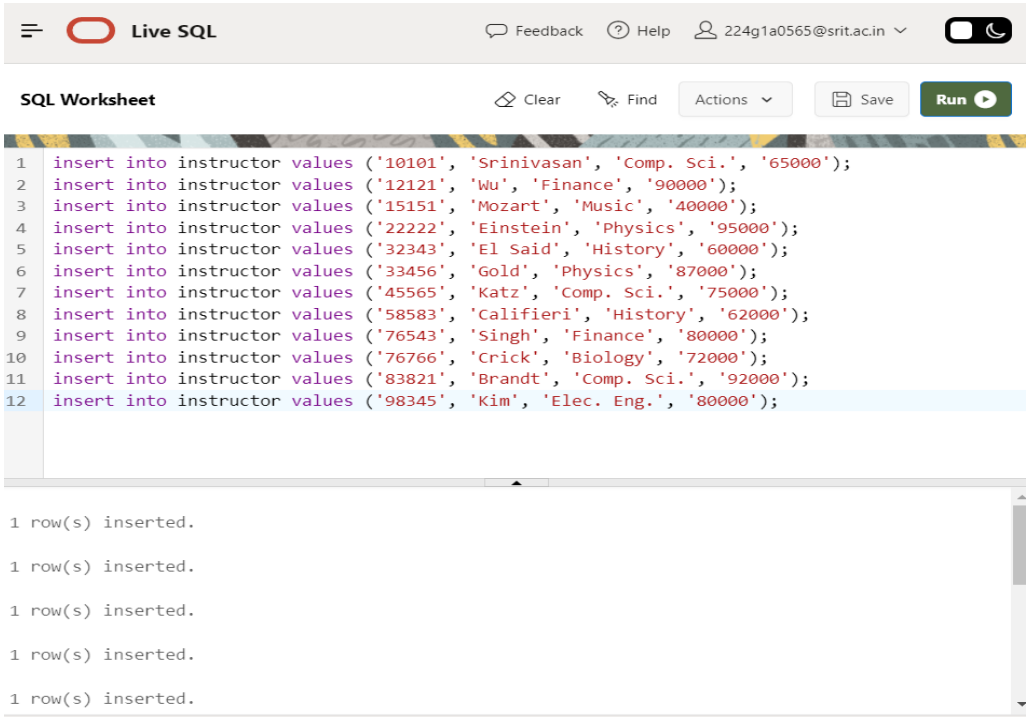
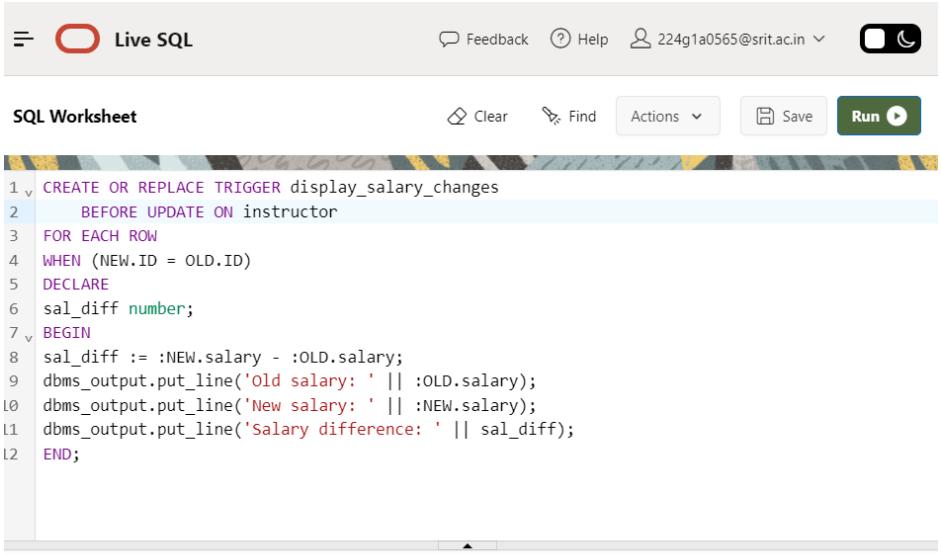


Table created.

4.Instances of Instructor Table



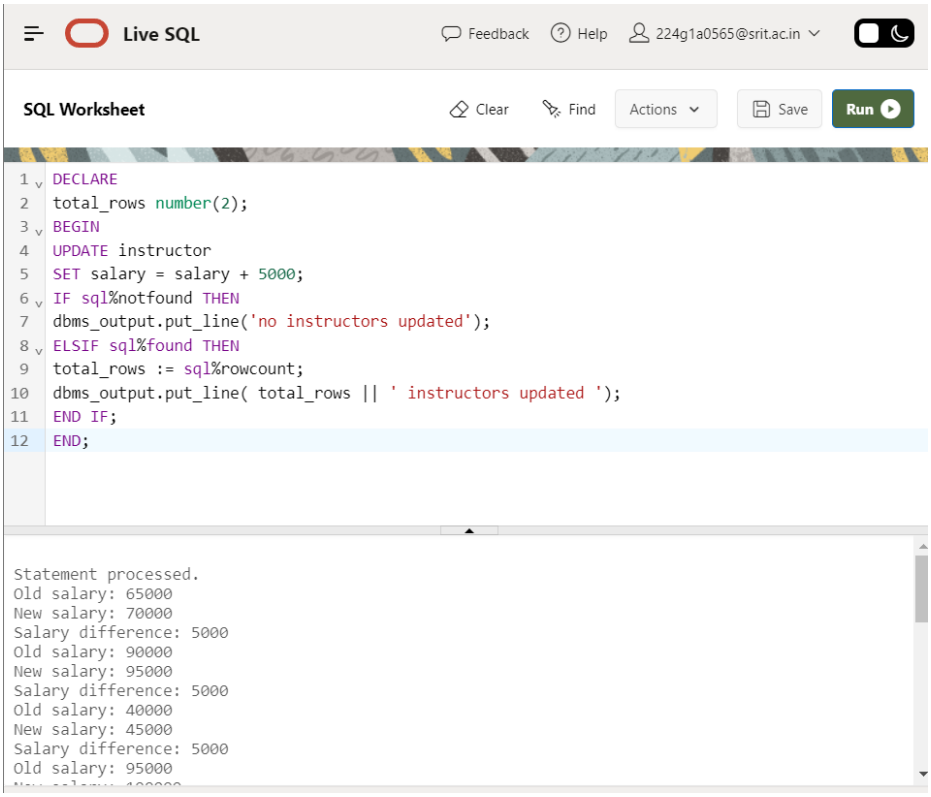
5.Example to Create Trigger

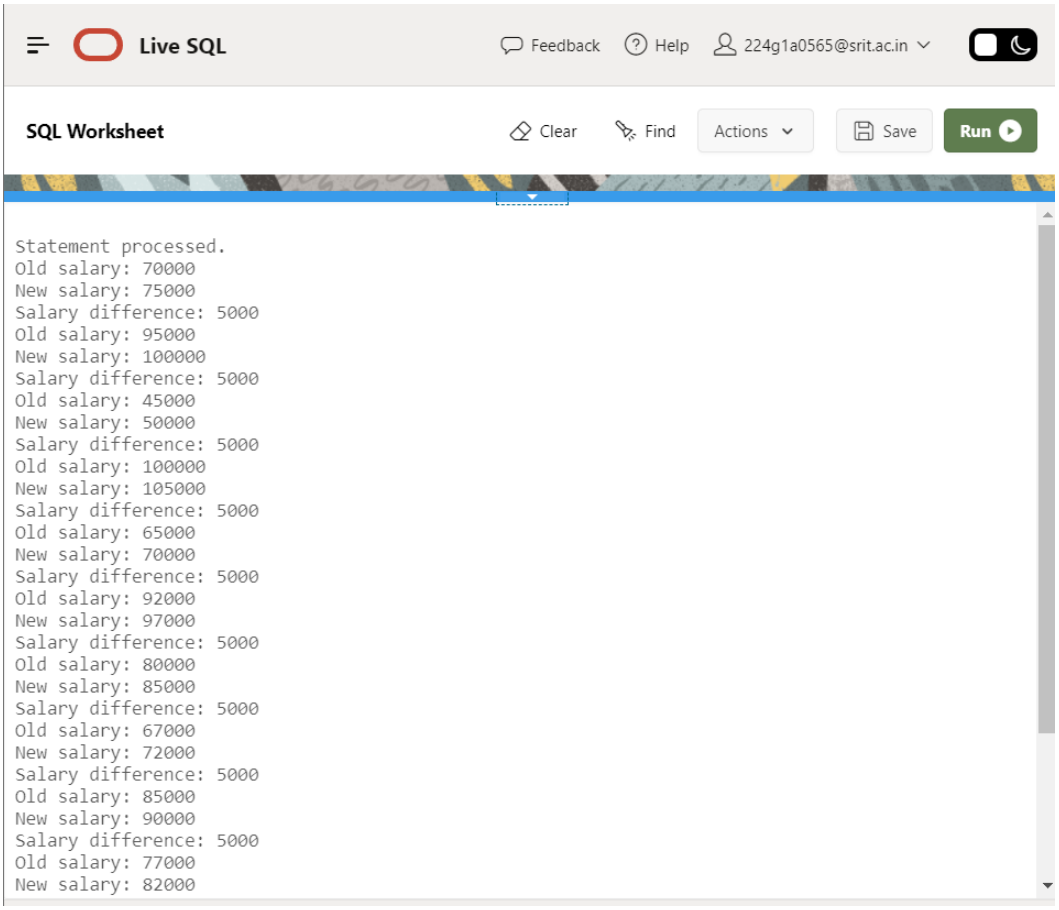


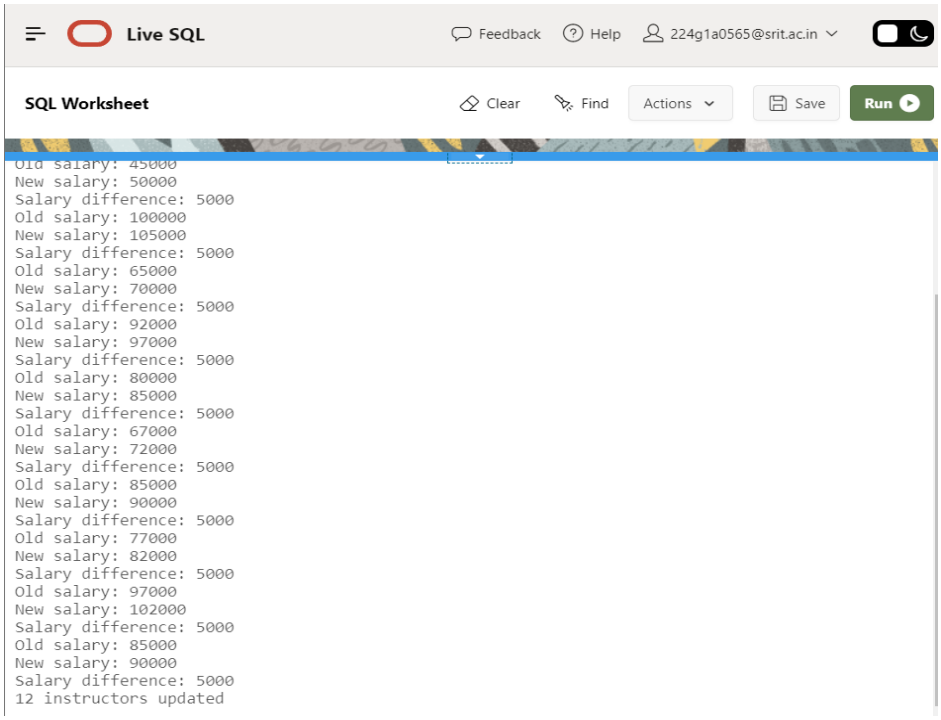

```
Trigger created.
```

6.PL/SQL Program to Execute a trigger

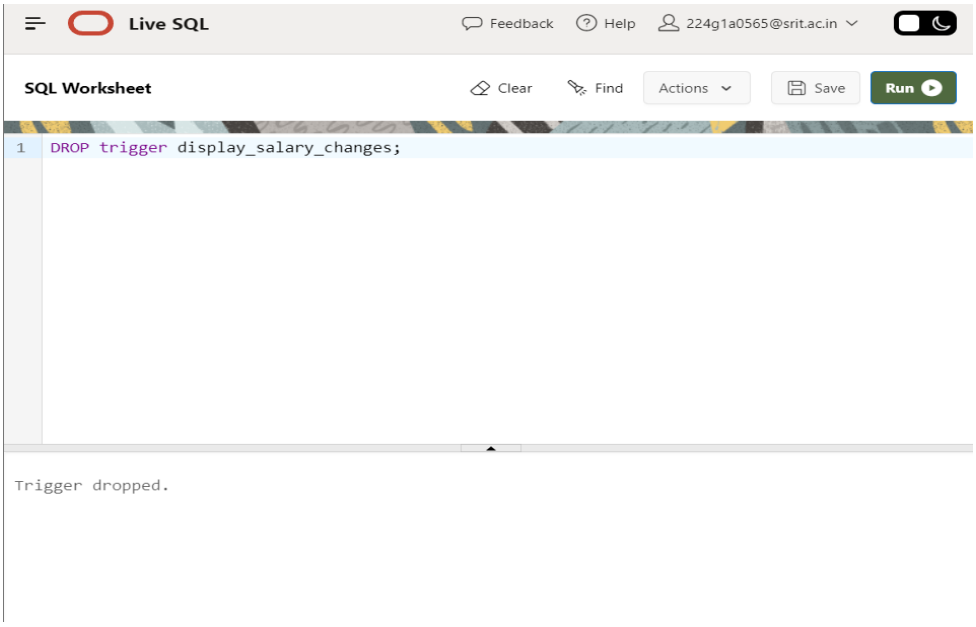
224g1a0565-k.pallavi







7.Drop the Trigger

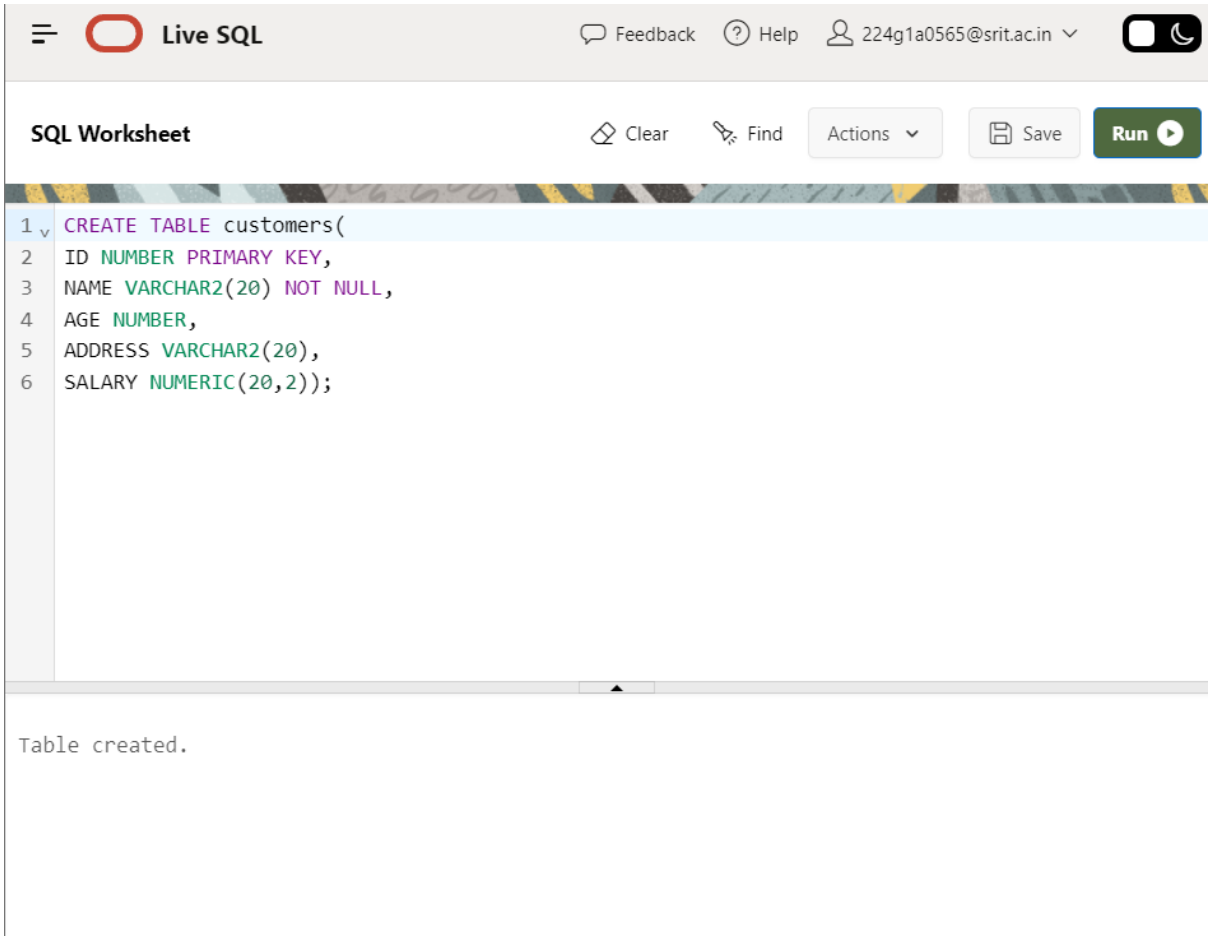


Conclusion:

In this lab,we successfully Executed PL/SQL program to implement Trigger on Table.

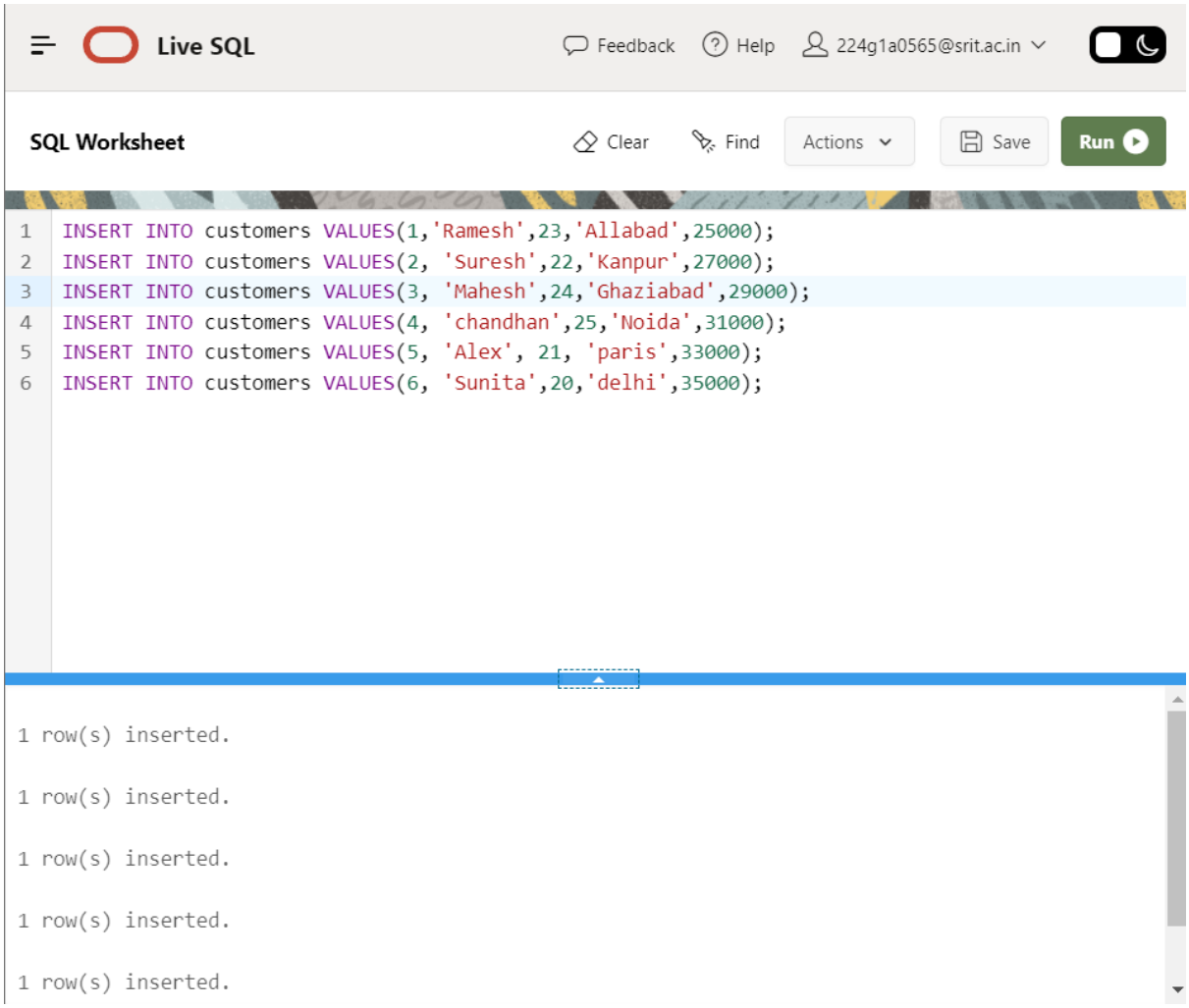
Experiment-16

1.create Table

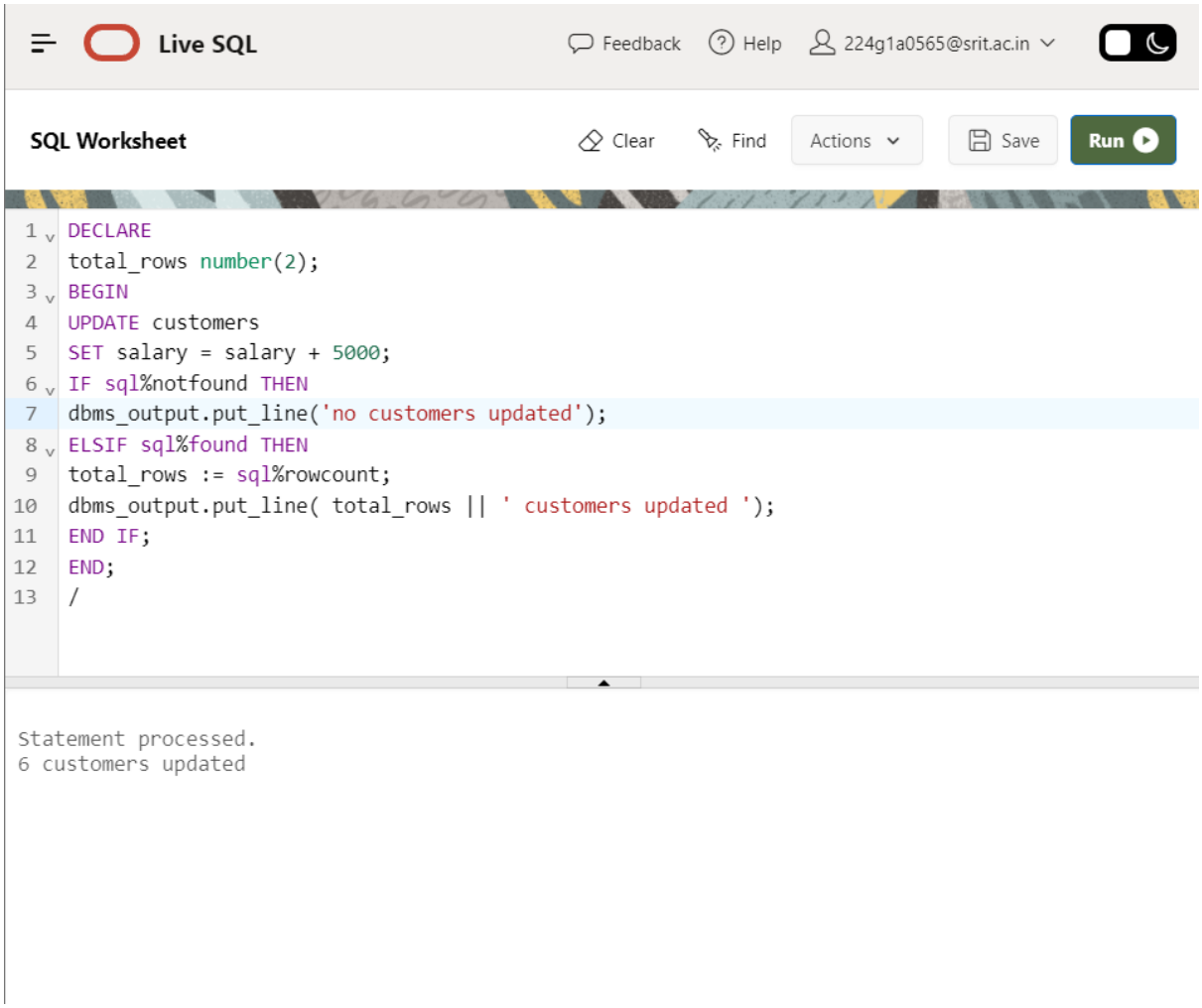


2.Instances of Customers

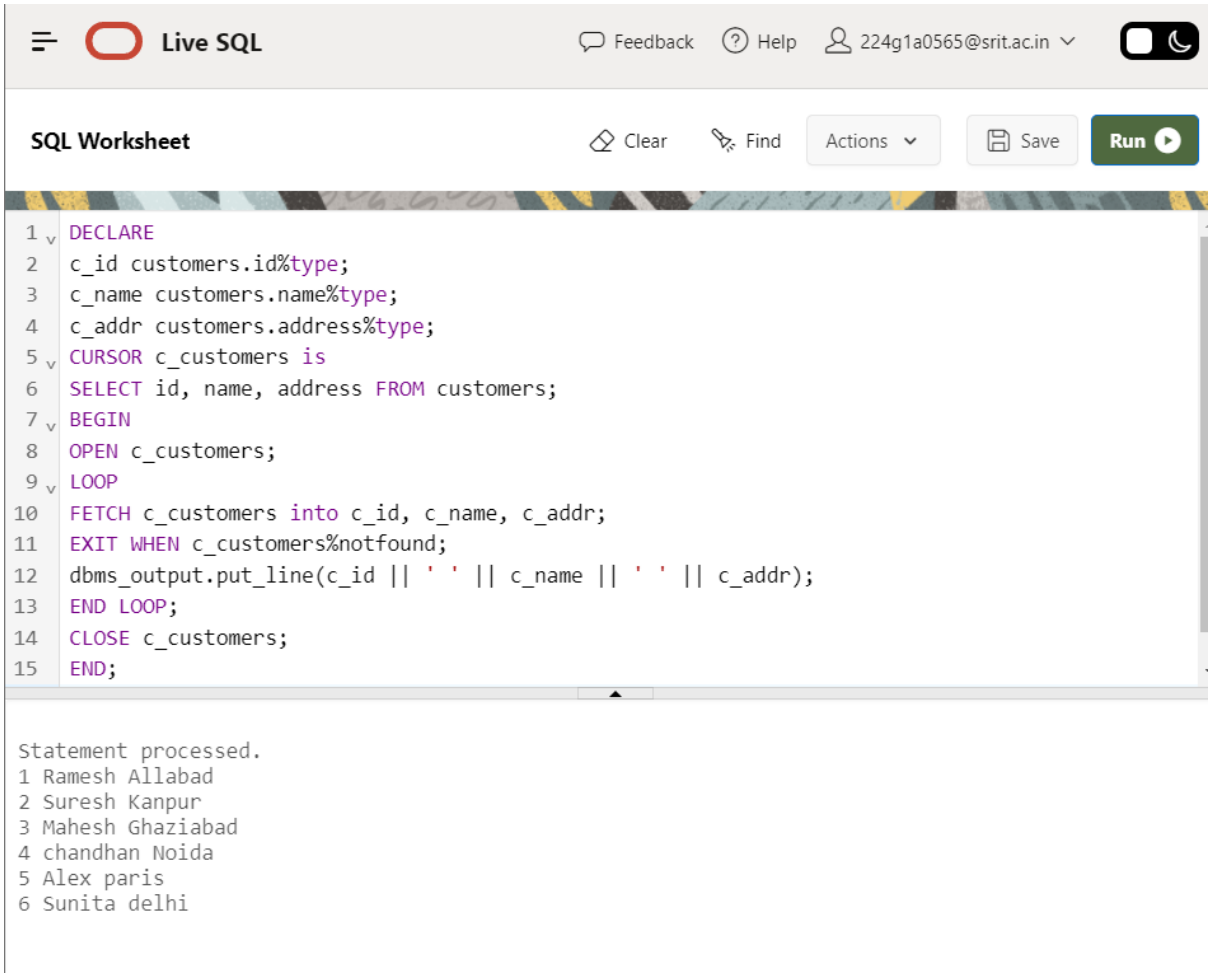
224G1A0565-k.pallavi



3.Create Update Procedure



4.PL/SQL Program using Explicit Cursors



Conclusion:

In this lab,we successfully Executed PL/SQL program to implement cursor on table.