EXPERIMENT-1

**AIM : Write SQL queries to CREATE TABLES for various databases using**  **DDL commands (i.e. CREATE, ALTER, DROP, TRUNCATE).**

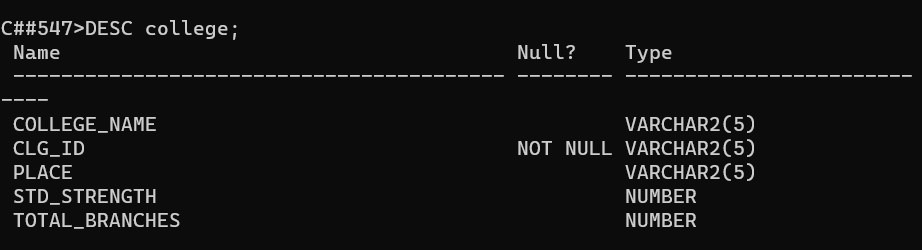
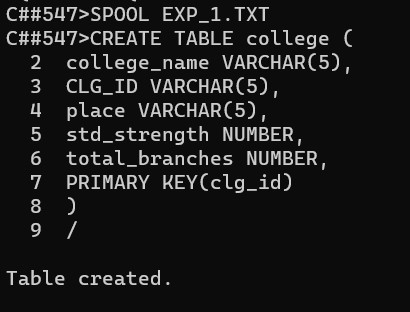
**CREATE TABLE:**

**Creates a table with specified constraints**

**SYNTAX:**

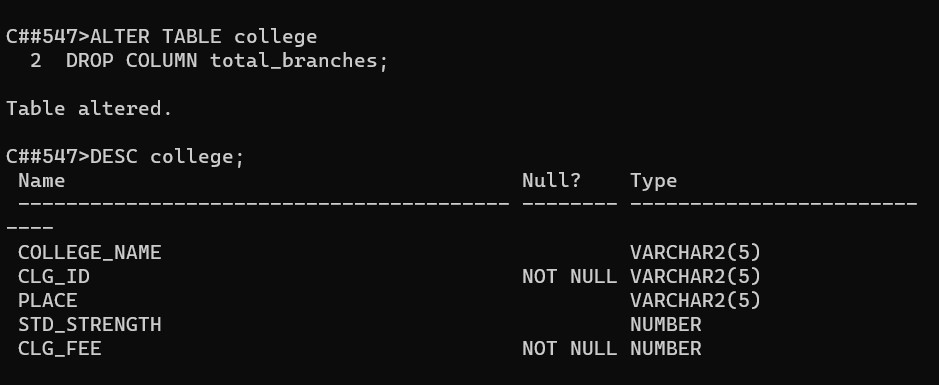
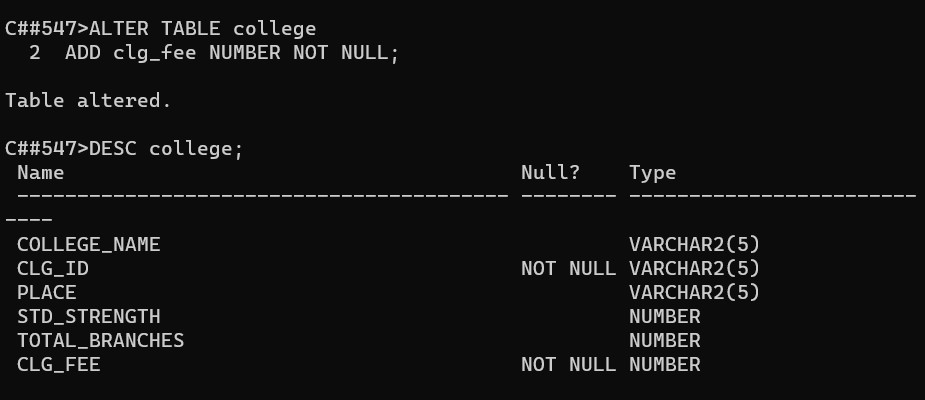
**CREATE TABLE tablename (**  **column1 data\_ type [constraint] [,**  **column2 data\_ type [constraint] ] [,**  **PRIMARY KEY (column1 [, column2]) ] [,**

**FOREIGN KEY (column1 [, column2]) REFERENCES tablename] [,CONSTRAINT**  **constraint]);**



**ALTER TABLE :**

**Used to add or modify table details like column names and data types,**  **column constraints.**

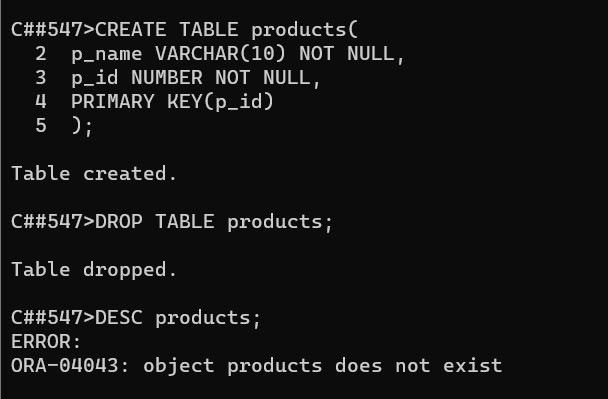


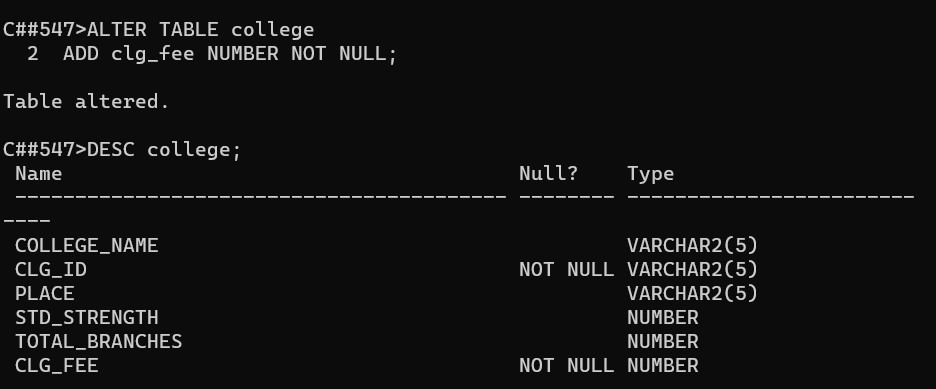
**DROP TABLE:**

Deletes the specified table.

**SYNTAX:**

DROP TABLE table\_name;

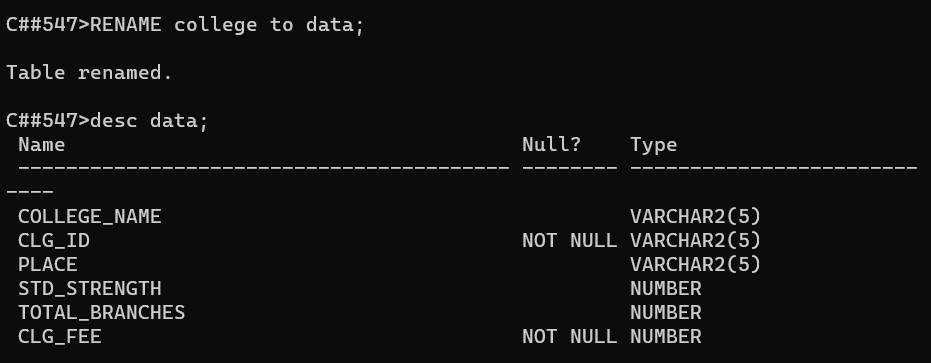




**RENAME TABLE:**

To rename table\_name, column\_name SYNTAXES:

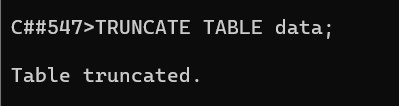
RENAME new\_table\_name TO old\_table\_name;



**TRUNCATE TABLE:**

**To remove all rows in a specified table.**  **SYNTAX:**

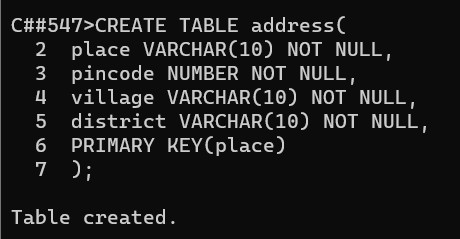
**TRUNCATE TABLE table\_name;**



# EXPERIMENT-2

**AIM : TO Write SQL queries to MANIPULATE TABLES for various databases using DML commands(i.e. INSERT, SELECT, UPDATE, DELETE,).**

**Creating table :**



**INSERT COMMAND:**

**It is used to add values to a table.**

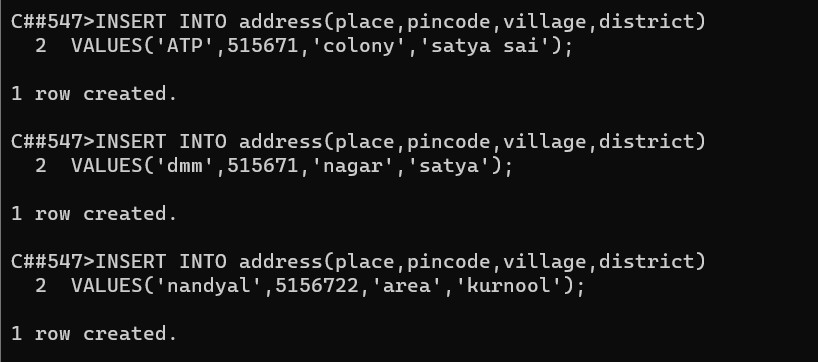
**SYNTAX:**

**INSERT INTO tablename**

**VALUES (value1,value2,...,valuen);**

**INSERT INTO tablename (column1, column2,...,column)**

**VALUES (value1, value2,...,valuen);**



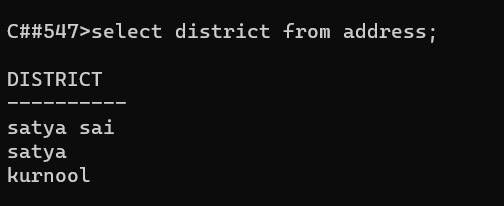
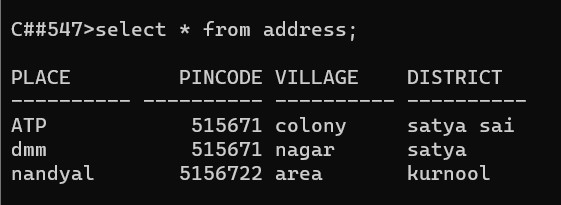
**SELECT COMMAND:**

**The SELECT command used to list the contents of a table.**

**SYNTAX:**

**Select \* from table\_name;**

**Select col\_name from table\_name;**



**UPDATE COMMAND:**

**The update command used to modify the contents of specified table.**

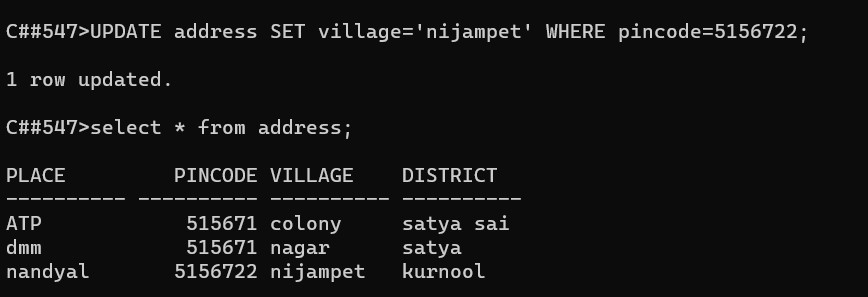
**SYNTAX:**

**UPDATE tablename**

**SET column\_name = value[,**

**Column\_name = value ]**

**[ WHERE condition\_lsit ];**

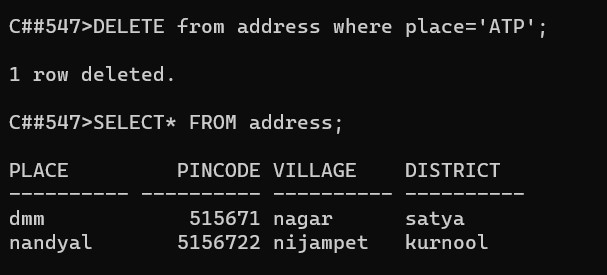


**DELETE COMMAND:**

**To delete all rows or specified rows in a table.**

**SYNTAX:**

**DELETE FROM tablename [ WHERE condition\_list];**

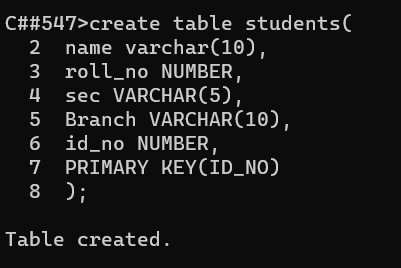


**Experiment-3**

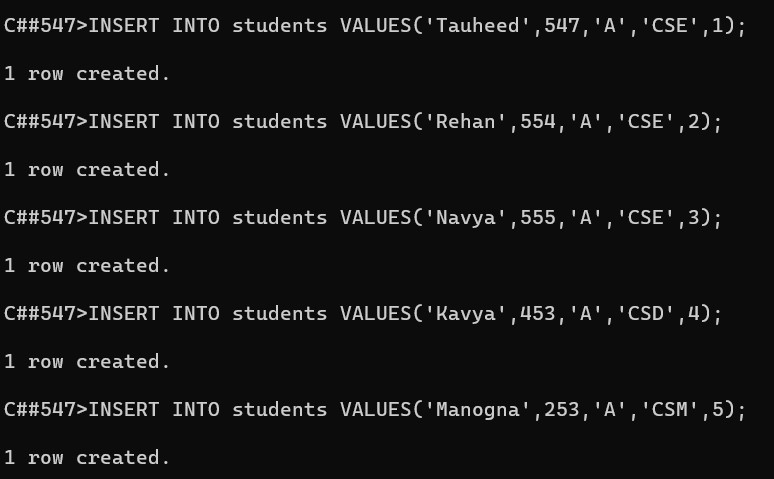
**DBMS**

**Aim:** To implement a view level design using CREATE VIEW,ALTER VIEW and DELETE VIEW ddl commands.

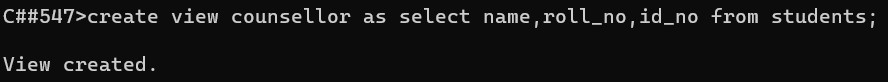
Creating a table:



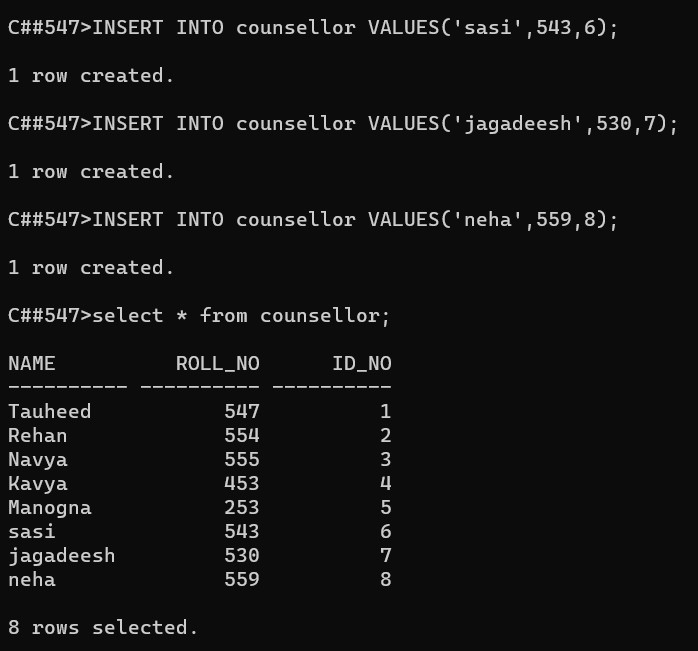
By using insert command we can insert values in a tables



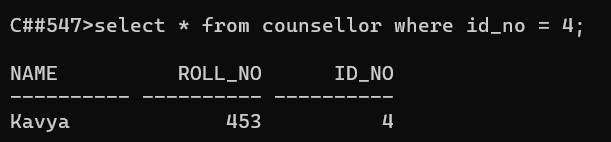
**Creating view councellor:**



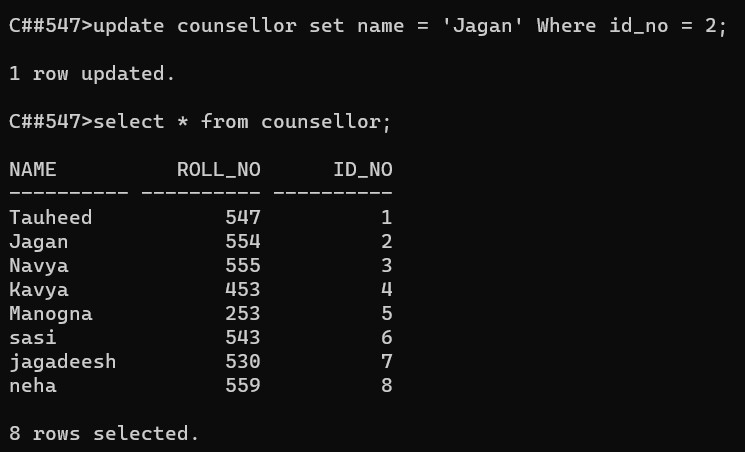
Inserting values into councellor:



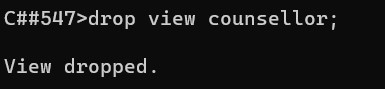
Selecting specific row :



**Update :**



**truncate or drop view:**



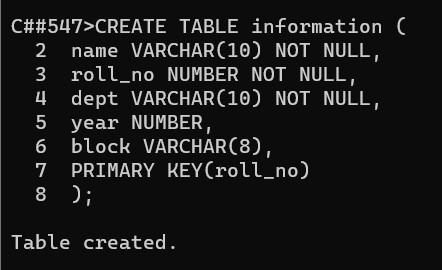
## EXPERIMENT-4

AIM : To create/perform relational set operations(i.e UNION

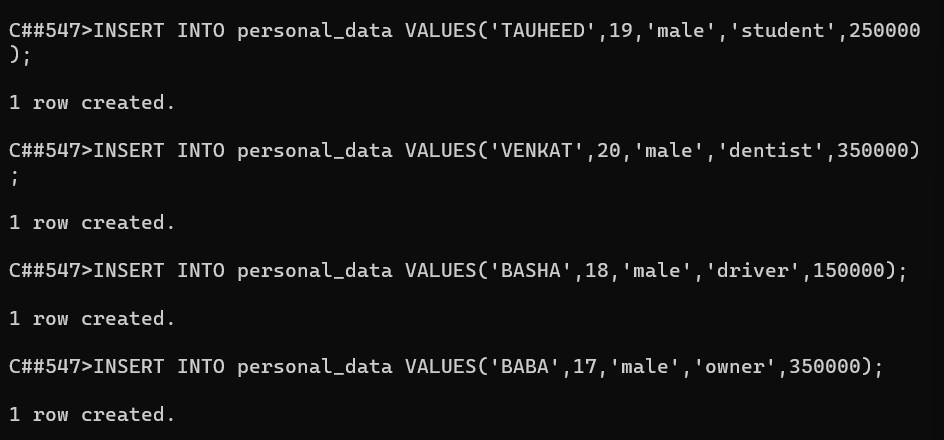
UNIONALL, INTERSECT, MINUS, CROSS JOIN, NATURAL

JOIN.)

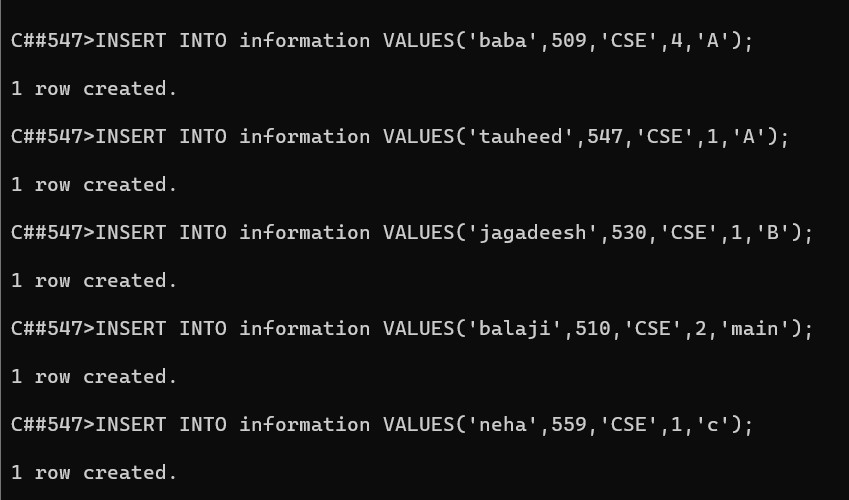
Creating tables:



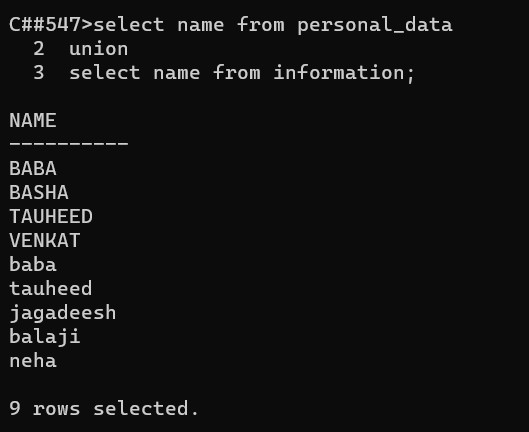
Inserting values into **personal\_data** table :



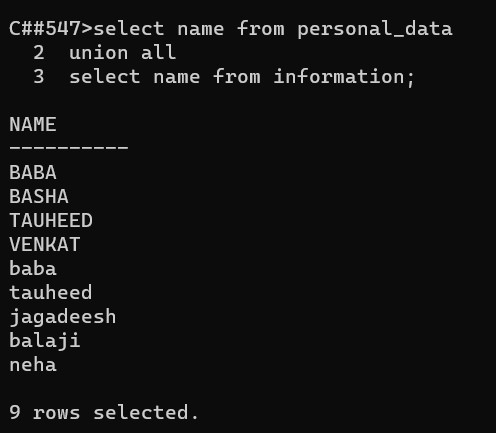
Inserting values into **information** table :



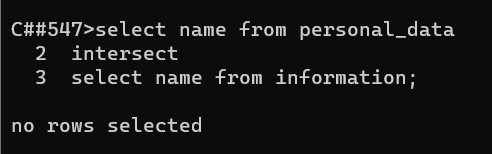
Union operation :



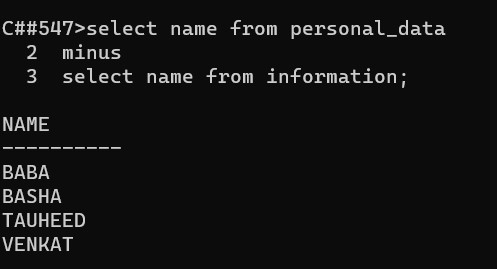
Union all operation :



Intersect operation :



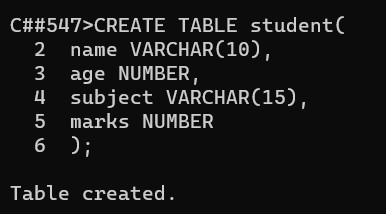
Minus operation :



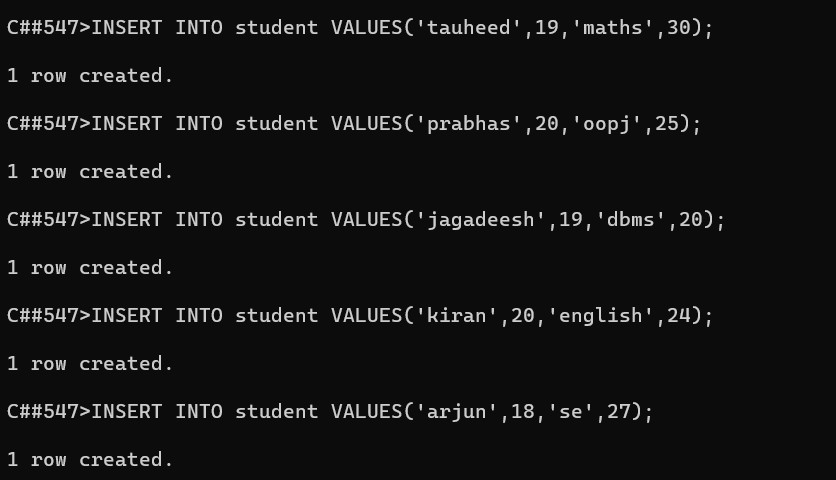
EXPERIMENT-5

Aim: write SQL queries for the aggregate functions(sum,count,min,max,avg)

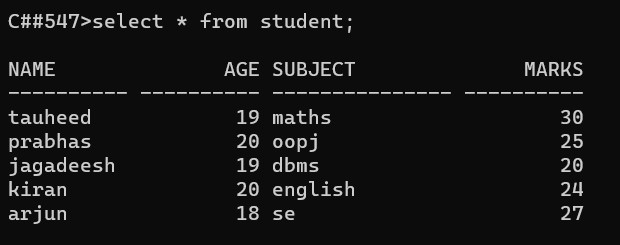
Creating a table:

 Inserting values

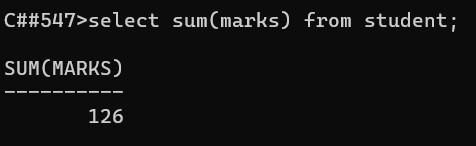
into table :



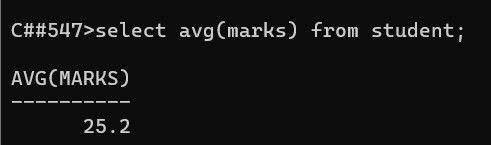
Selecting table :



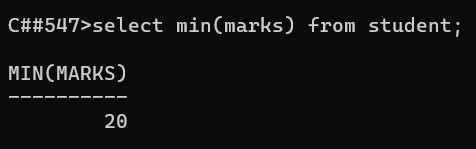
Sum();



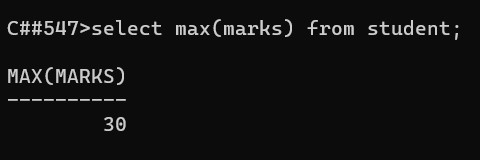
Avg();



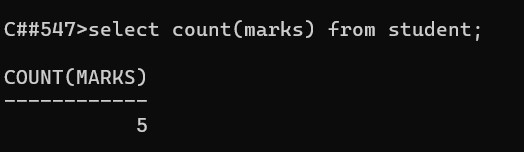
Min();



Max();



Count();



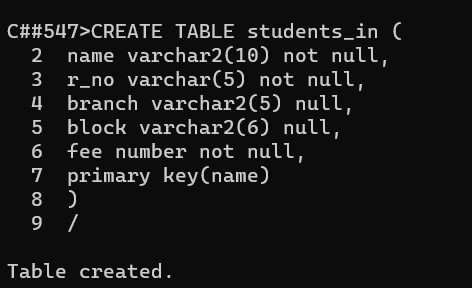
### EXPERIMENT-5

EXPERIMENT-6

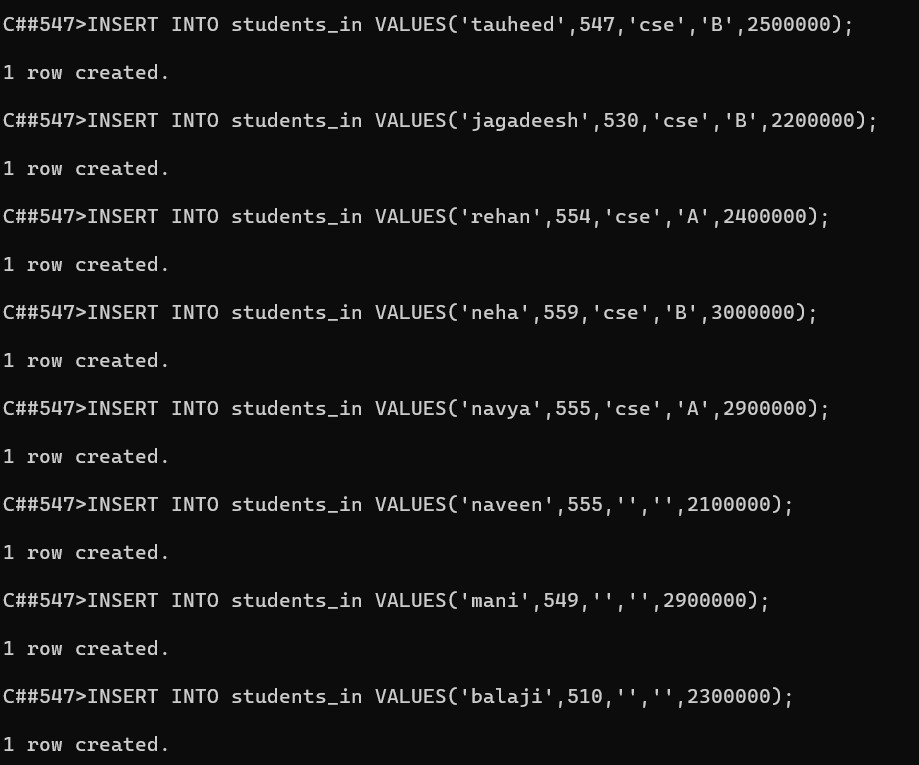
AIM: TO WRITE SQL QUERIES TO PERFORM SPECIAL OPERATIONS (i.e

LIKE, BETWEEN, ISNULL, ISNOTNULL)

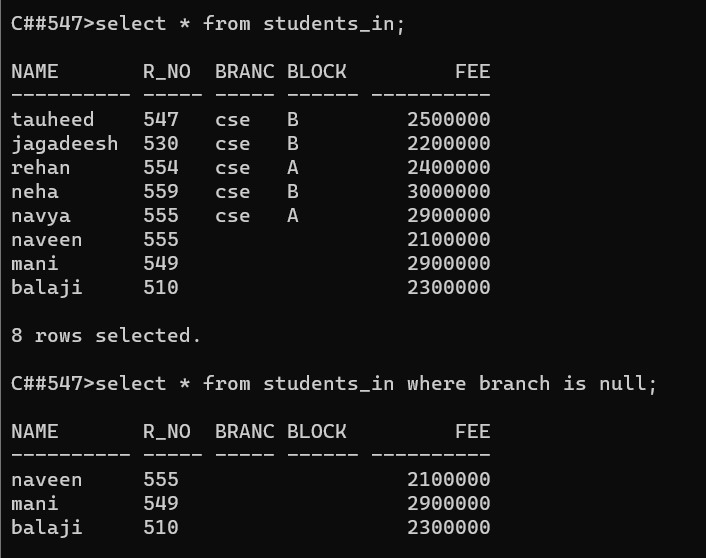
**Creating a table**



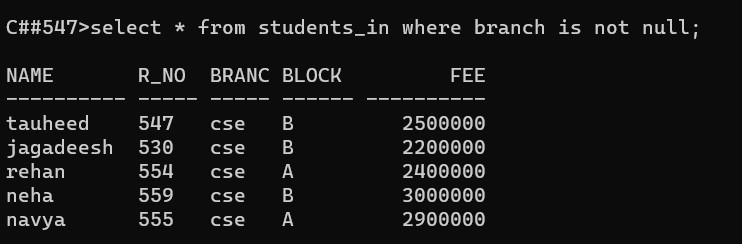
**Inserting values :**



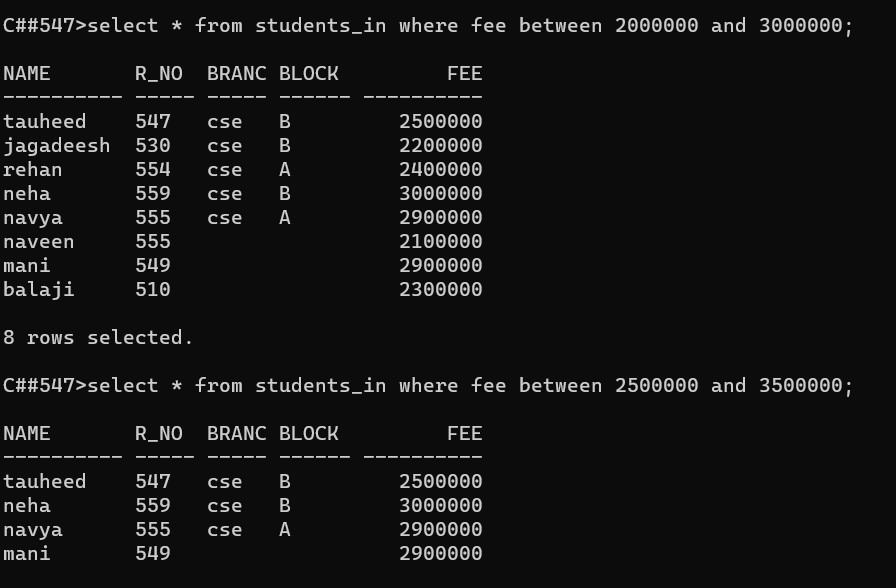
**Is Null operation :**



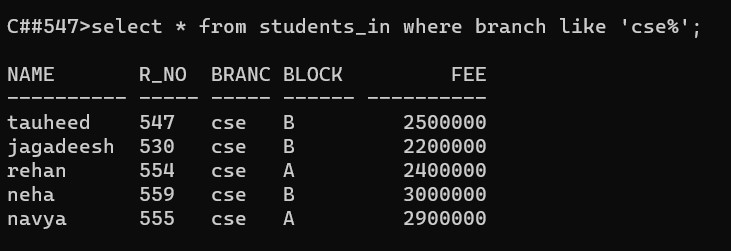
**Is not null operation :**

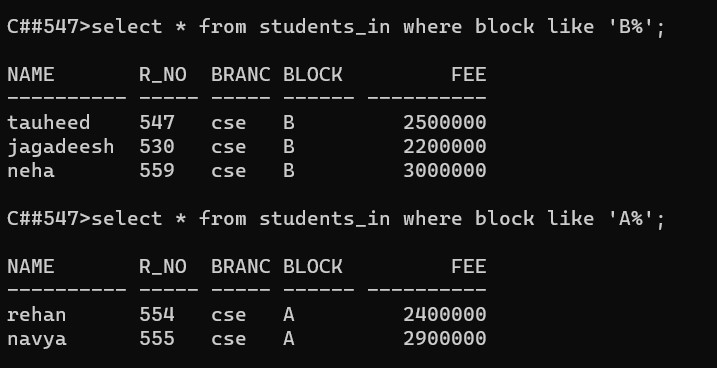


**Between operation :**

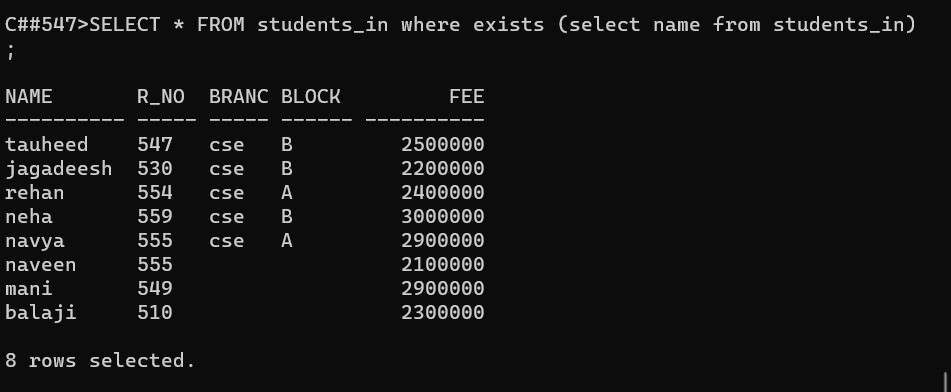


**Like operation:**





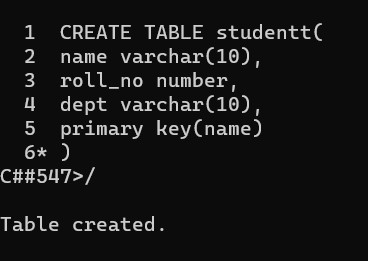
**Exists operation :**



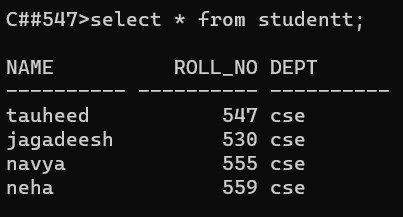
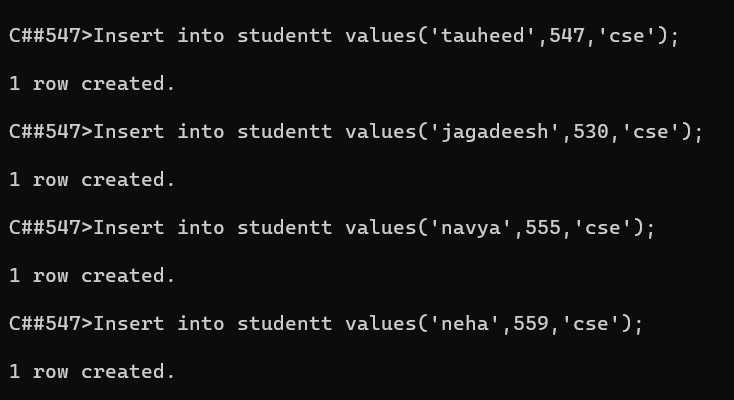
#### EXPERIMENT-7

AIM: Write SQL queries to perform JOIN OPERATIONS (i.e. CONDITIONAL JOIN, EQUI JOIN,LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTfER JOIN)

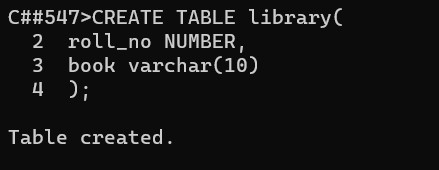
CREATING TABLE student :



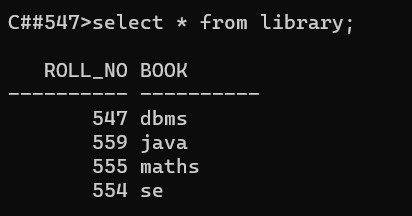
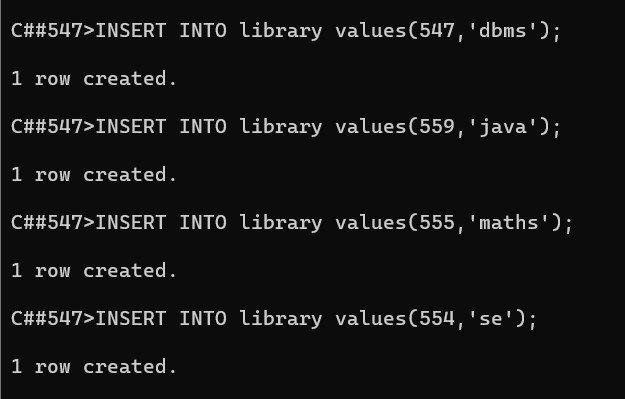
Inserting tables into student table :



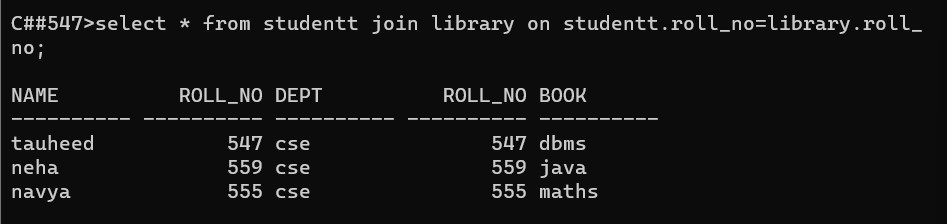
Creating table Library :



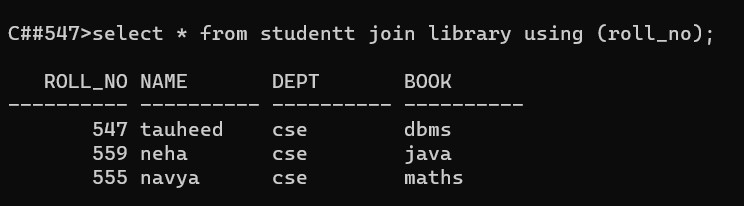
Inserting values into library table :



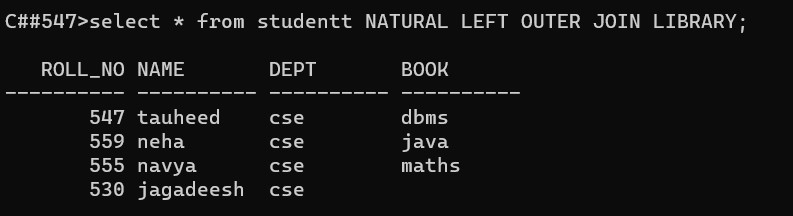
CONDITIONAL JOIN :



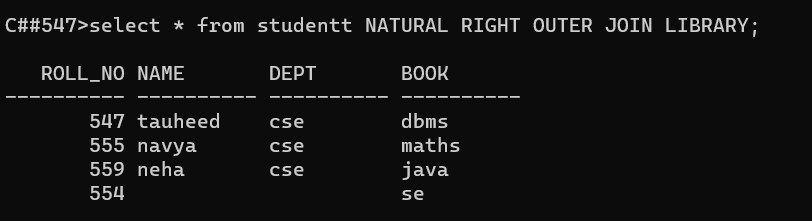
EQUI JOIN :



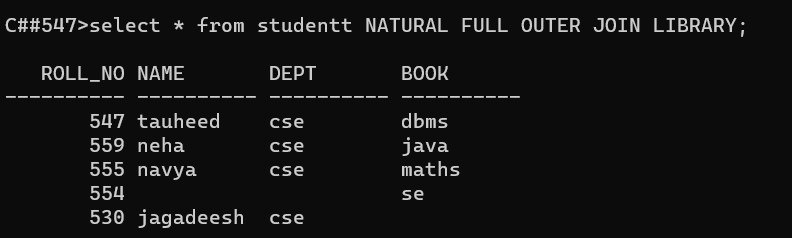
NATURAL LEFT OUTER JOIN :



NATURAL RIGHT OUTER JOIN :



NATURAL FULL OUTER JOIN :



#### EXPERIMENT-8

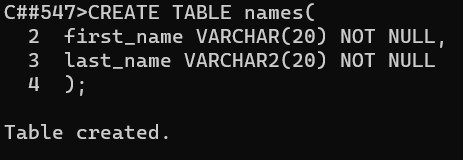
AIM : Write SQL queries to perform ORACLE BUILT-IN FUNCTIONS (i.e. DATE, TIME).

**Built-in Functions**

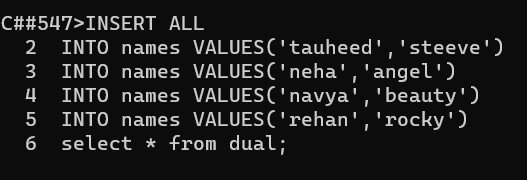
1. **Character Functions**  **I. Case-conversion functions**

**II. Character manipulation functions**

1. **Number Functions**
2. **DATE functions** CREATING TABLE :



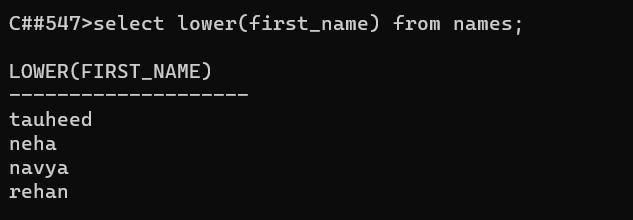
INSERTING VALUES :



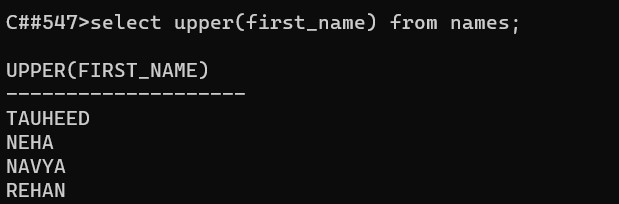
##### 1. Character Functions

**I. Case-conversion functions :**

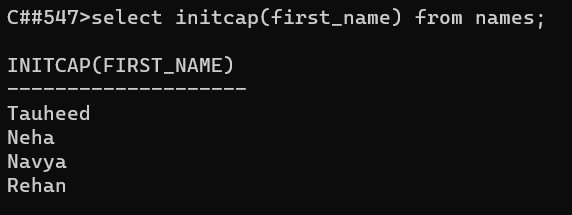
*LOWER ();*



*UPPER();*

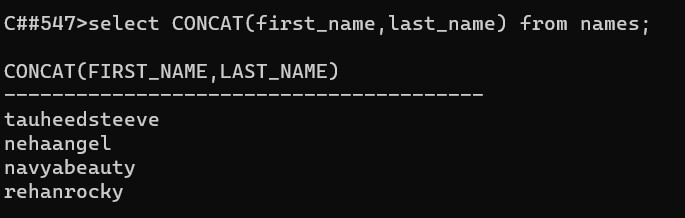


*INITCAP();*

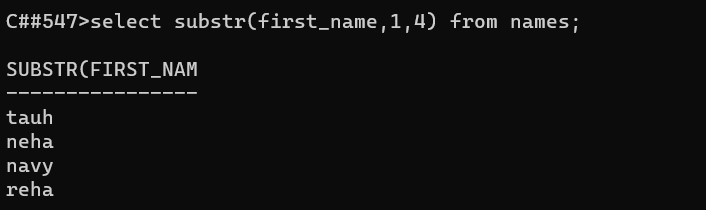


**Character manipulation functions:**

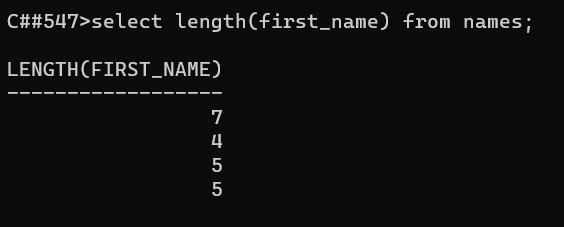
***CONCAT():***



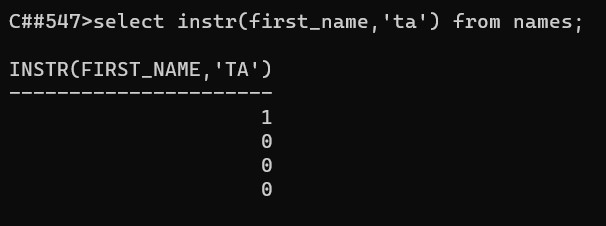
***SUBSTR():***



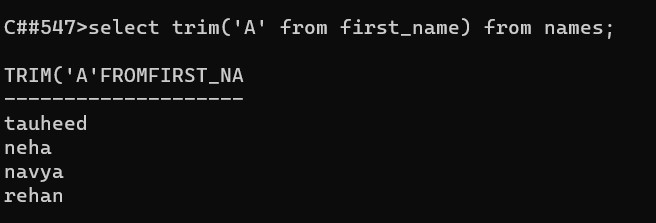
LENGTH() :



INSTR() :

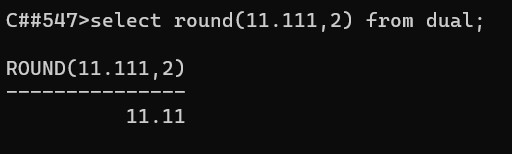


TRIM() :

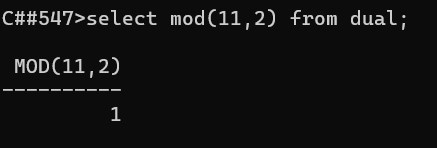


**2. Number Functions :**

ROUND() :

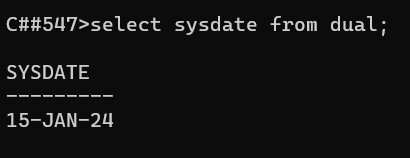


MOD() :

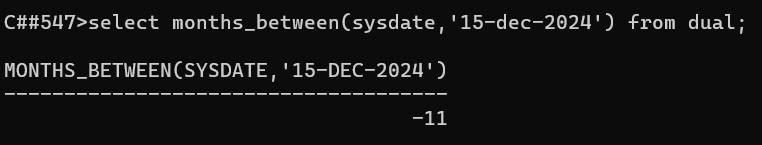


**2.DATE functions :**

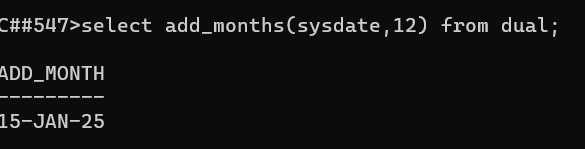
SYSDATE()



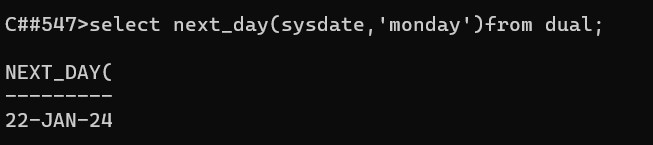
MONTHS-BETWEEN() :



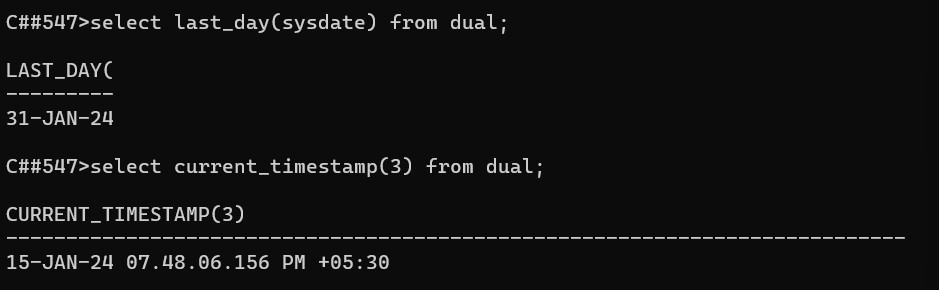
ADD\_MONTHS() :



NEXT\_DAY():



LAST\_DAY() :



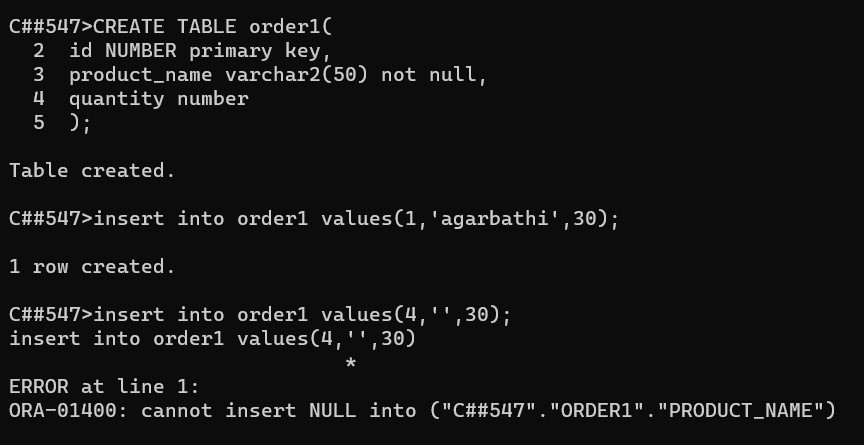
## **EXPERIMENT-9**

**AIM :** Write SQL queries to perform KEY CONSTRAINTS (i.e. PRIMARY KEY, FOREIGN KEY,UNIQUE NOT NULL, CHECK, DEFAULT).

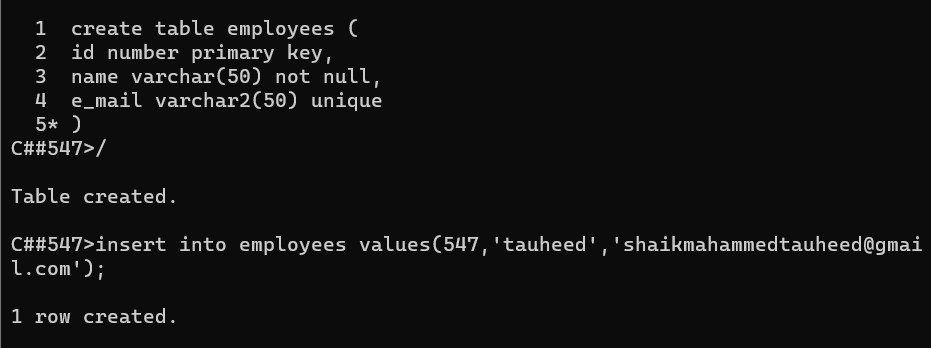
**Types of SQL Constraints.**

1. NOT NULL - Ensures that a column cannot have a NULL value
2. UNIQUE - Ensures that all values in a column are different
3. PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely I identifies each row in a table
4. FOREIGN KEY - Uniquely identifies a row/record in another table
5. CHECK - Ensures that all values in a column satisfies a specific condition
6. DEFAULT - Sets a default value for a column when no value is specified

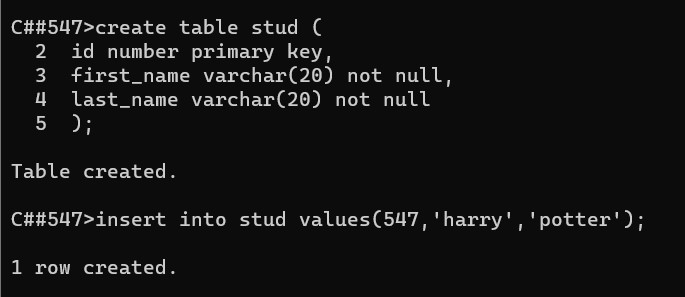
1.NOT NULL Constraint Example:



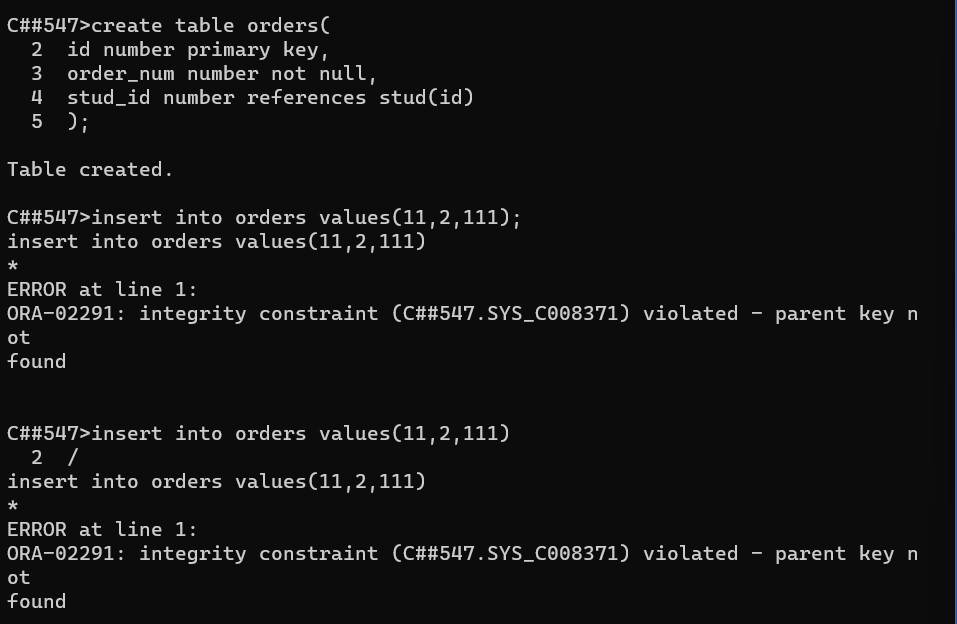
2.UNIQUE CONSTRAINT Example:



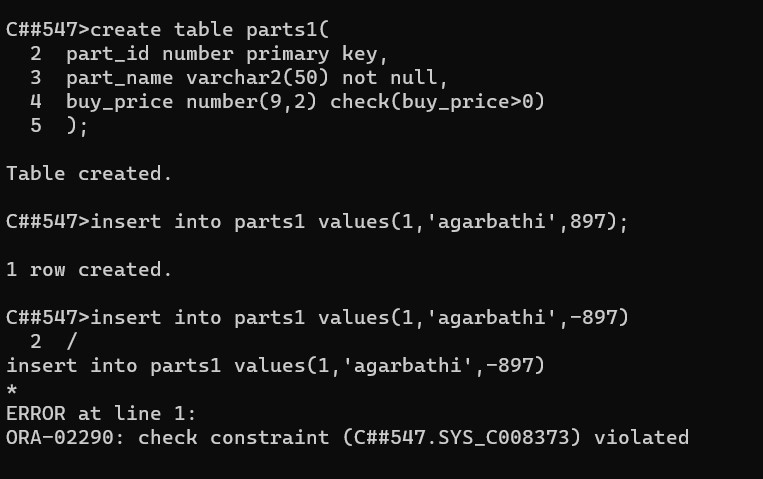
3.PRIMARY KEY CONSTRAINT Example:



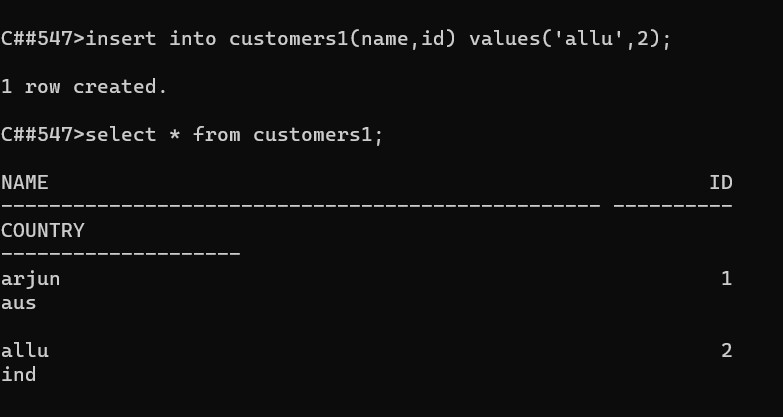
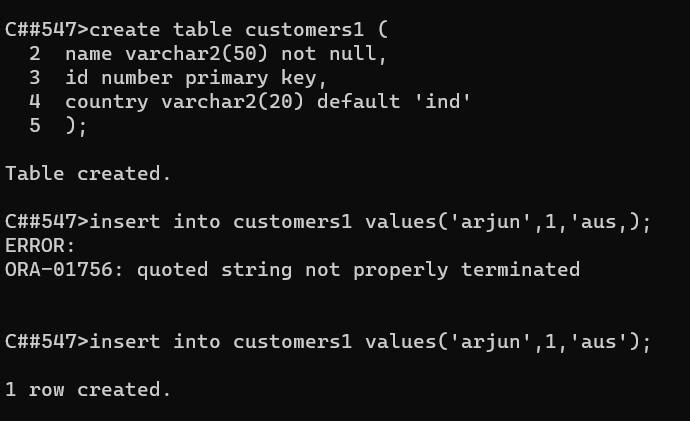
4.FORIEGN KEY CONSTRAINTS Example:



5.CHECK CONSTRAINTS Example:

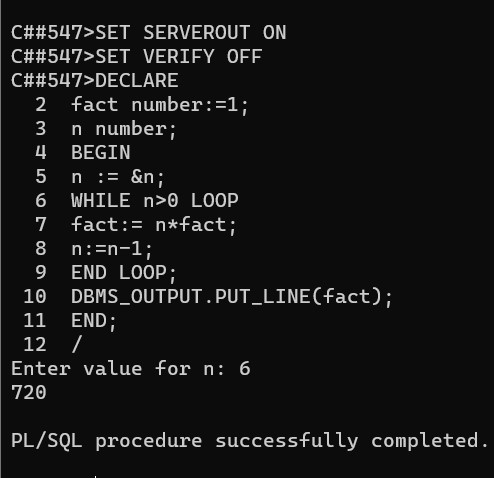


6.DEFAULT CONSTRAINTS Example:



AIM: To write a PL/SQL program for calculating the factorial of a given number**.**

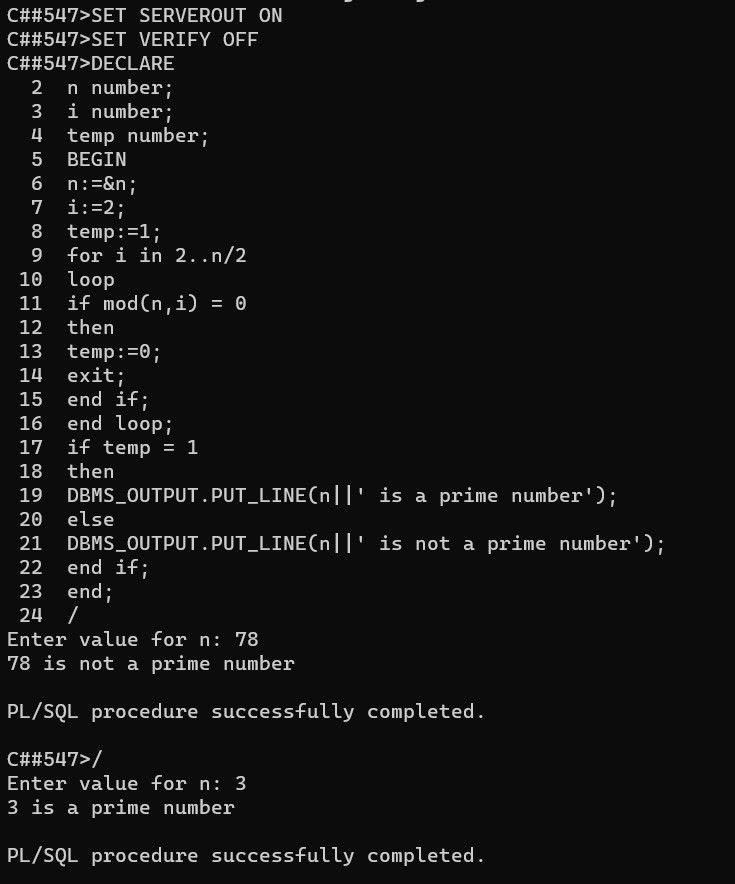
**Source code:**



Conclusion : The pl/sql program is successfully executed.

**AIM: Write a PL/SQL program for finding the given number is prime number or not.**

**SOURCE CODE:**

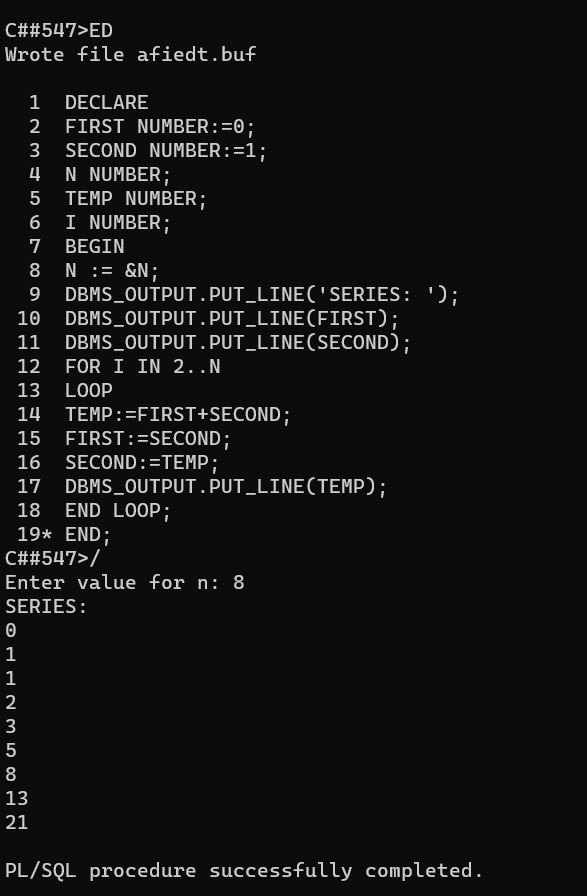


CONCLUSION: The pl/sql program is successfully executed.

**AIM:** **Write a PL/SQL program for displaying the Fibonacci series up**  **to an integer.**

**SOURCE CODE:**





CONCLUSION: The pl/sql program is successfully executed.

### EXPERIMENT-13

Write PL/SQL program to implement Stored Procedure on table.

AIM:

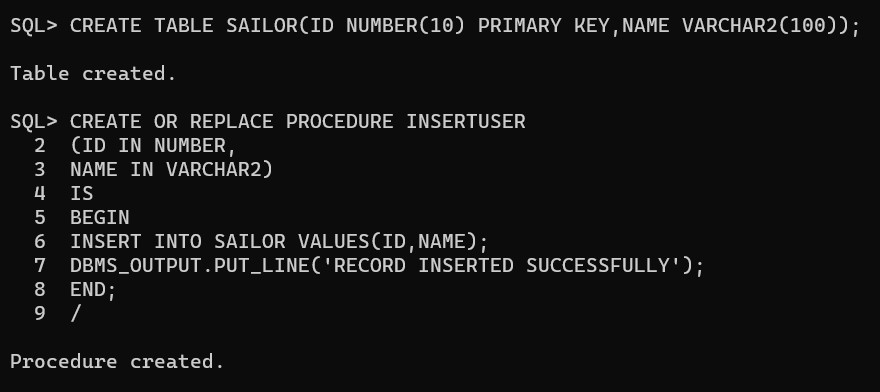
Write PL/SQL program to implement Stored Procedure on table.

**PL/SQL Procedure:**

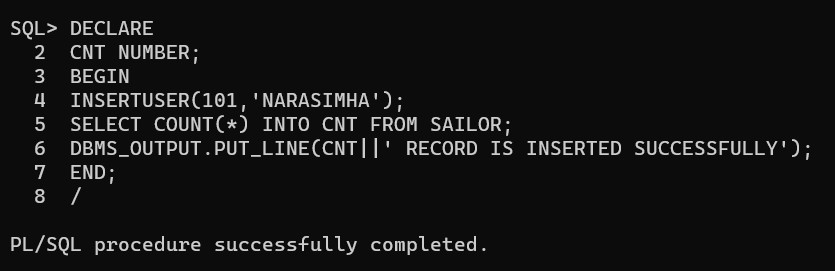
The PL/SQL stored procedure or simply a procedure is a PL/SQL block which performs one or more specific tasks. It is just like procedures in other programming languages.

The procedure contains a header and a body.

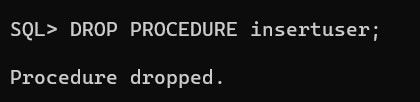
EXAMPLE :1



**Execution Procedure:**



**DROP PROCEDURE:**



**CONCLUSION :**

The pl/sql programs is successfully executed.

EXPERIMENT-14

AIM:

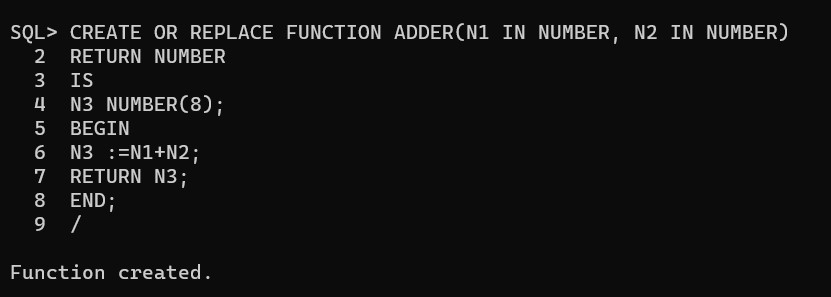
TO Write PL/SQL program to implement Stored Function on table.

PL/SQL Function:

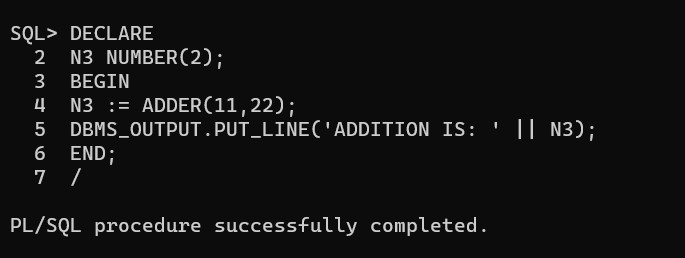
The PL/SQL Function is very similar to PL/SQL Procedure. The main difference between

procedure and a function is, a function must always return a value, and on the other hand a

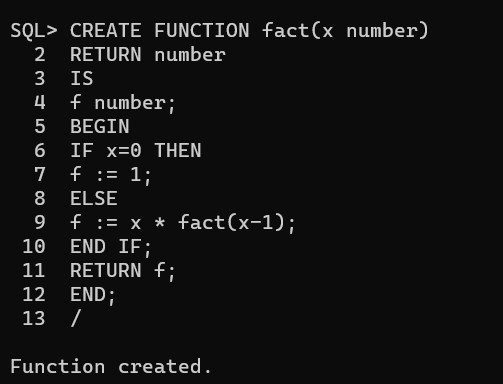
procedure may or may not return a value. Except this, all the other things of PL/SQL procedure are true for PL/SQL function too.



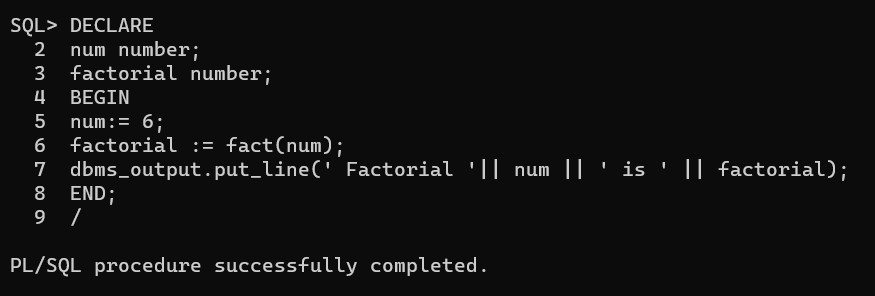
Execution Procedure:



EXAMPLE : 2



Execution Procedure:



Conclusion:

The pl/sql program is successfully executed.

### EXPERIMENT-15

AIM : TO Write PL/SQL program to implement Trigger on table.

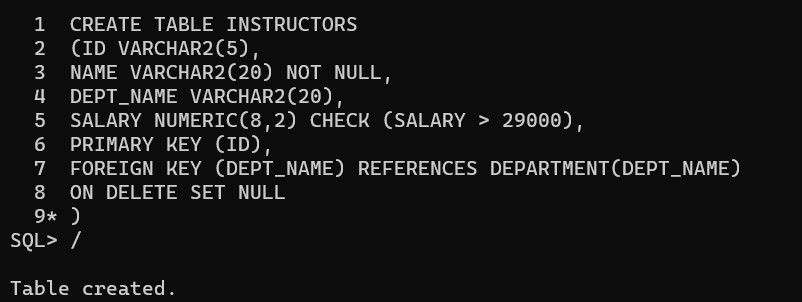
Trigger is invoked by Oracle engine automatically whenever a specified event occurs. Trigger is stored into database and invoked repeatedly, when specific condition match. Triggers are

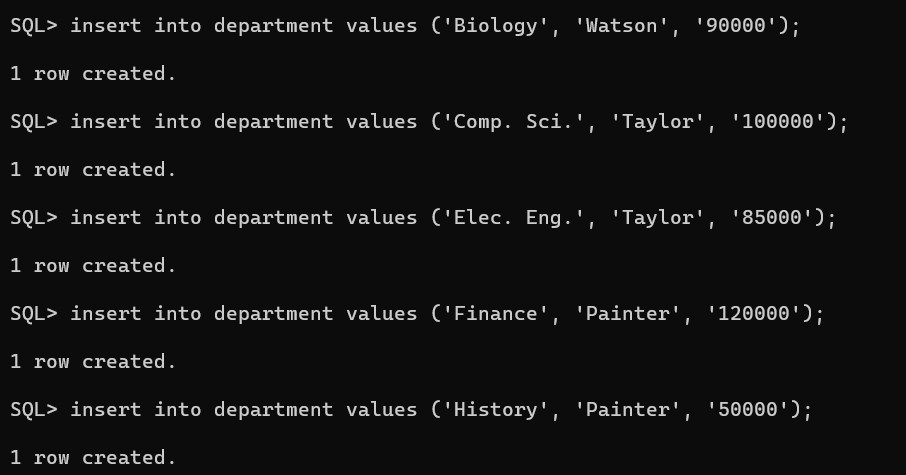
stored programs, which are automatically executed or fired when some event occurs. Triggers are written to be executed in response to any of the following events.

A database manipulation (DML) statement (DELETE, INSERT, or UPDATE).

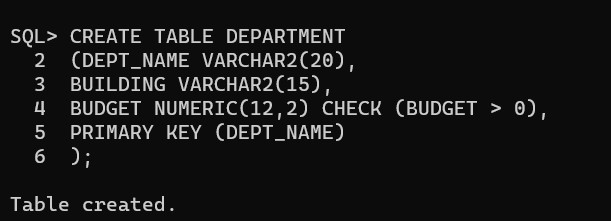
A database definition (DDL) statement (CREATE, ALTER, or DROP).

A database operation (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

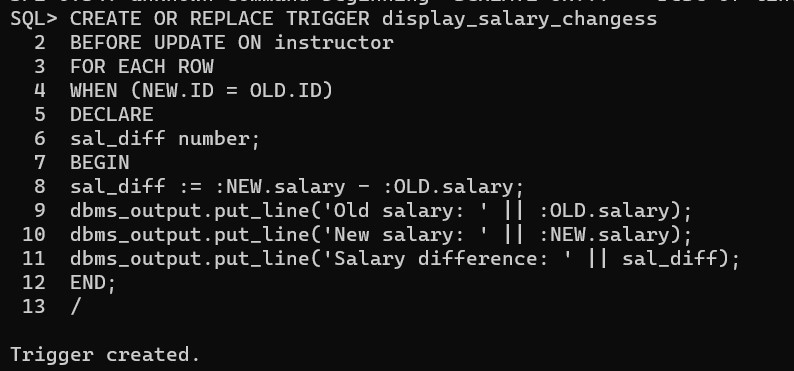




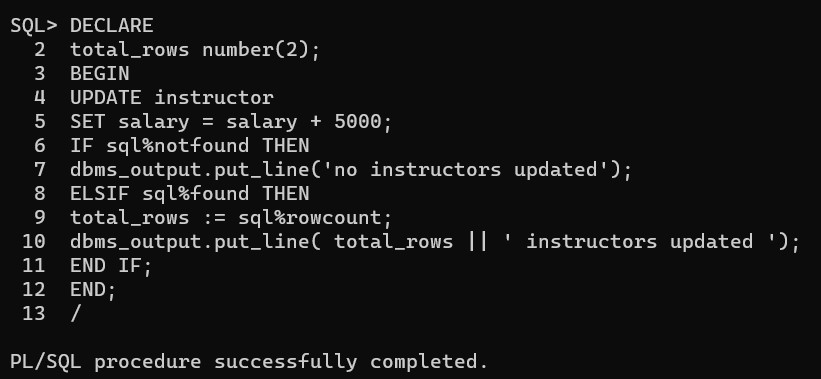
CREATING DEPARTMENT TABLE :



**An example to create Trigger :**



**A PL/SQL Procedure to execute a trigger:**



**Conclusion:**

**The pl/sql program is successfully executed.**

EXPERIMENT

-

16

AIM:

To write PL/SQL program to implement Cursor on table.

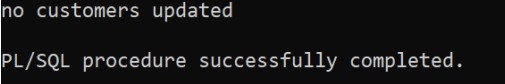
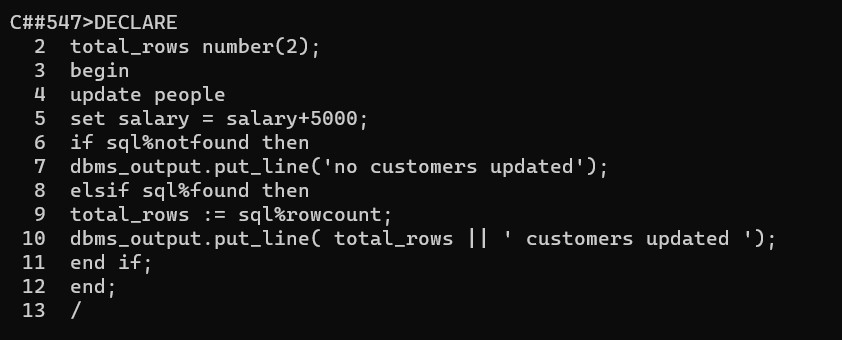
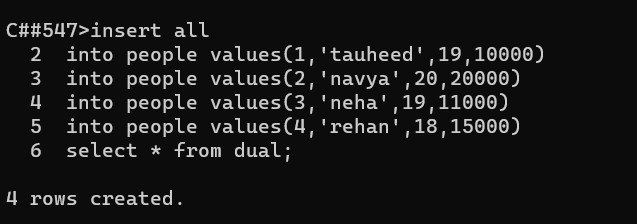
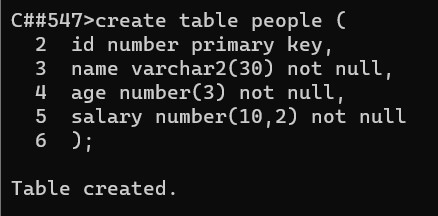
Source code:

Instances of people :

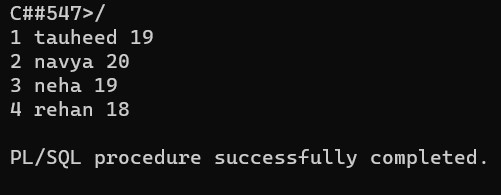
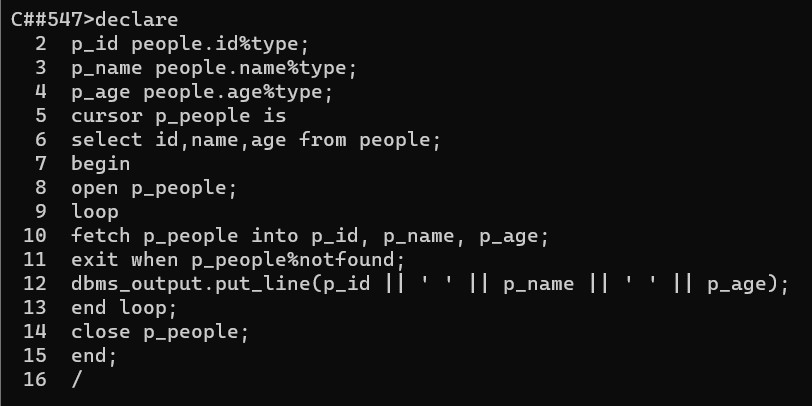
Create update procedure

Create procedure:

PL/SQL Program using Explicit Cursors :



CONCLUSION : The pl/sql program is successfully executed.



224G1A0547



MAHAMMED TAUHEED S



7



-



12



-



23



PL/SQL Program using Explicit Cursors :



CONCLUSION : The pl/sql program is success



fully executed.

