Name:- Atish Kumar

Roll No:- 120CS0173

Date:- 20 April,2022

DE Lab Assignment 08

Q 1: Write SQL queries to perform creation of table, insert values, select and drop for the following instruction.

1. **Create a table Employee having attribute Employee_ID, Name, Age, Salary, Department. Insert values in Employee table as shown & display all the values.**

Program:

```
CREATE DATABASE LAB08

USE LAB

CREATE TABLE Employes
(
Employee_ID VARCHAR(50),
Name VARCHAR(50),
Age INT,
Salary INT,
Department VARCHAR(50)
);

INSERT INTO Employes VALUES('1','Adiya',25,30000,'Executive');
INSERT INTO Employes VALUES('2','Aditya',28,31000,'Accountant');
INSERT INTO Employes VALUES('3','Priyanka',30,32000,'Salesman');
INSERT INTO Employes VALUES('4','Anmol',35,31000,'Executive');
INSERT INTO Employes VALUES('5','Rahul',31,29000,'Accountant');
INSERT INTO Employes VALUES('6','Shubham',29,27000,'Salesman');
INSERT INTO Employes VALUES('7','Sam',33,32000,'Salesman');
INSERT INTO Employes VALUES('8','Rohan',33,28000,'Salesman');
INSERT INTO Employes VALUES('9','Priya',23,35000,'Accountant');
INSERT INTO Employes VALUES('10','Kabir',29,26000,'Executive');

        SELECT*FROM Employes;
```

Output:

SQLQuery3.sql - UN...known Hector (57))*

```
INSERT INTO Employes VALUES('1','Adiya',25,30000,'Executive');
INSERT INTO Employes VALUES('2','Aditya',28,31000,'Accountant');
INSERT INTO Employes VALUES('3','Priyanka',30,32000,'Salesman');
INSERT INTO Employes VALUES('4','Anmol',35,31000,'Executive');
INSERT INTO Employes VALUES('5','Rahul',31,29000,'Accountant');
INSERT INTO Employes VALUES('6','Shubham',29,27000,'Salesman');
INSERT INTO Employes VALUES('7','Sam',33,32000,'Salesman');
INSERT INTO Employes VALUES('8','Rohan',33,28000,'Salesman');
INSERT INTO Employes VALUES('9','Priya',23,35000,'Accountant');
INSERT INTO Employes VALUES('10','Kabir',29,26000,'Executive');

SELECT*FROM Employes;
```

121 %

Results | Messages

	Employee_ID	Name	Age	Salary	Department
1	1	Adiya	25	30000	Executive
2	2	Aditya	28	31000	Accountant
3	3	Priyanka	30	32000	Salesman
4	4	Anmol	35	31000	Executive
5	5	Rahul	31	29000	Accountant
6	6	Shubham	29	27000	Salesman
7	7	Sam	33	32000	Salesman
8	8	Rohan	33	28000	Salesman
9	9	Priya	23	35000	Accountant
10	10	Kabir	29	26000	Executive

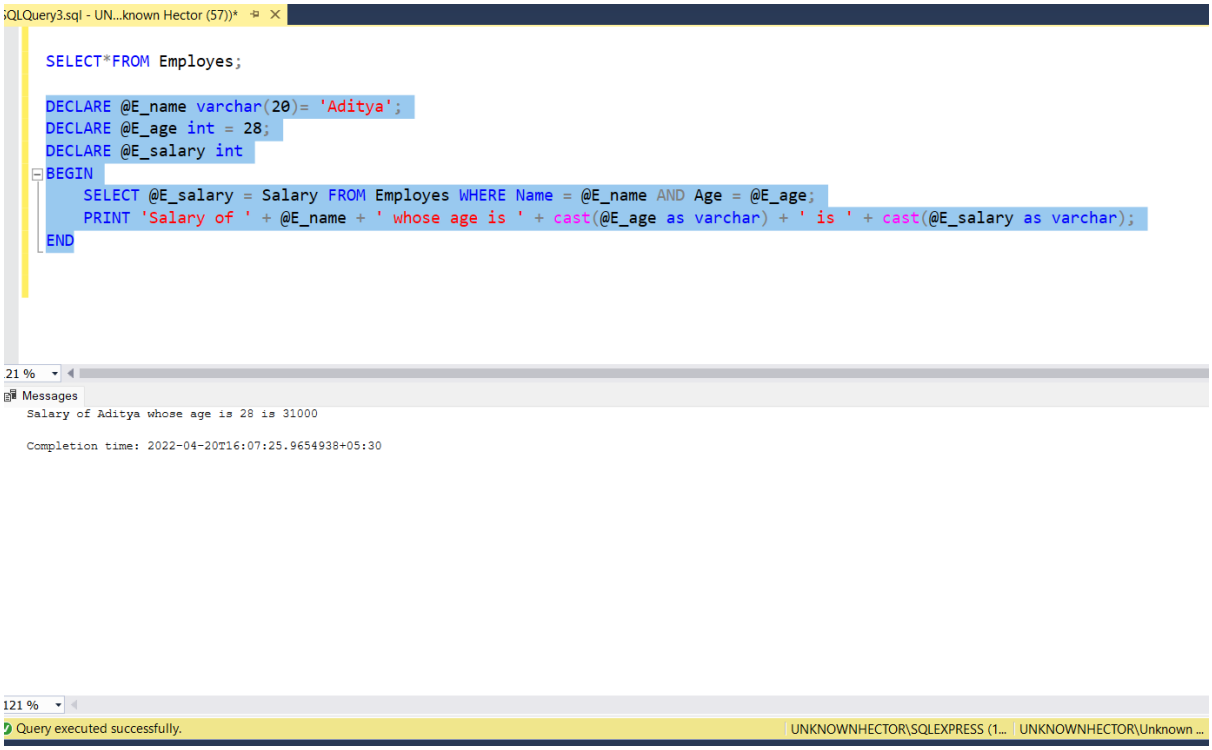
Query executed successfully. UNKNOWNHECTOR\SQLEXPRESS (1...

2. **Create a PL/SQL block which displays the salary earned by Employee.Name="Aditya" and Employee.Age="28".**

Program:

```
DECLARE @E_name varchar(20)= 'Aditya';
DECLARE @E_age int = 28;
DECLARE @E_salary int
BEGIN
       SELECT @E_salary = Salary FROM Employes WHERE Name = @E_name AND Age = @E_age;
       PRINT 'Salary of ' + @E_name + ' whose age is ' + cast(@E_age as varchar) + '
is ' + cast(@E_salary as varchar);
        END
```

Output:

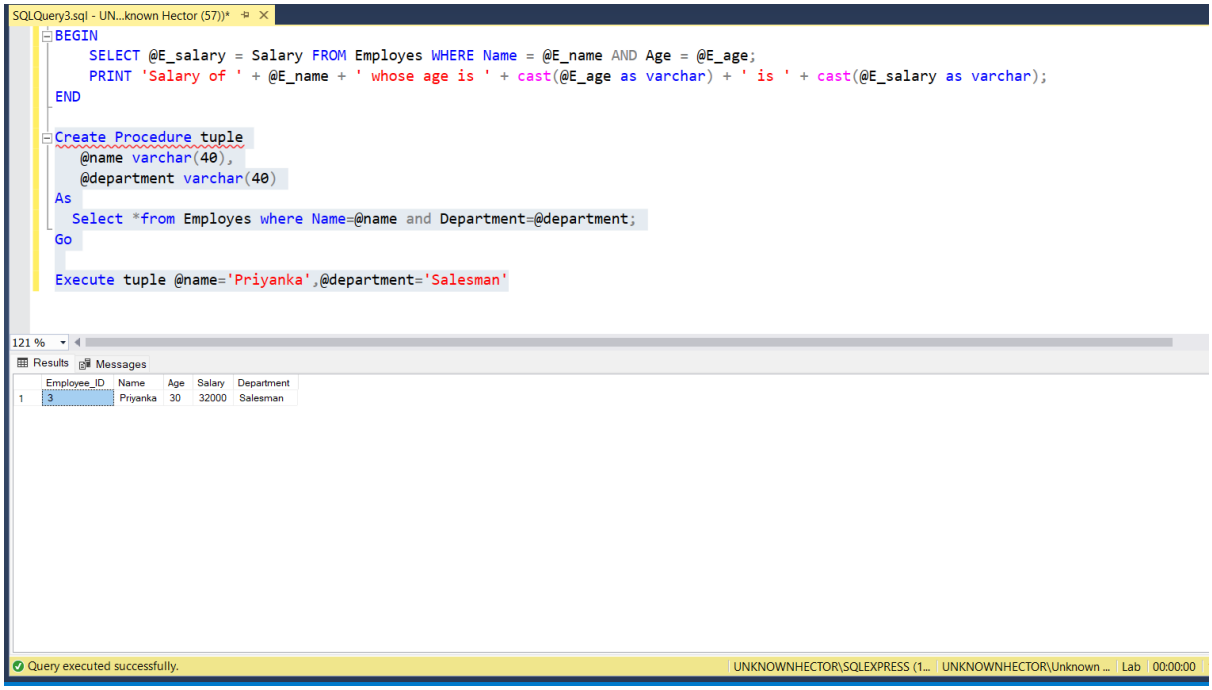


3. **Create a PL/SQL procedure that takes parameters as Employee.Name and Employee.Department and displays the tuple that has been passed as the parameter.**

Program:

```
Create Procedure tuple
   @name varchar(40),
   @department varchar(40)
As
  Select *from Employes where Name=@name and Department=@department;
Go

        Execute tuple @name='Priyanka',@department='Salesman'
```

Output:

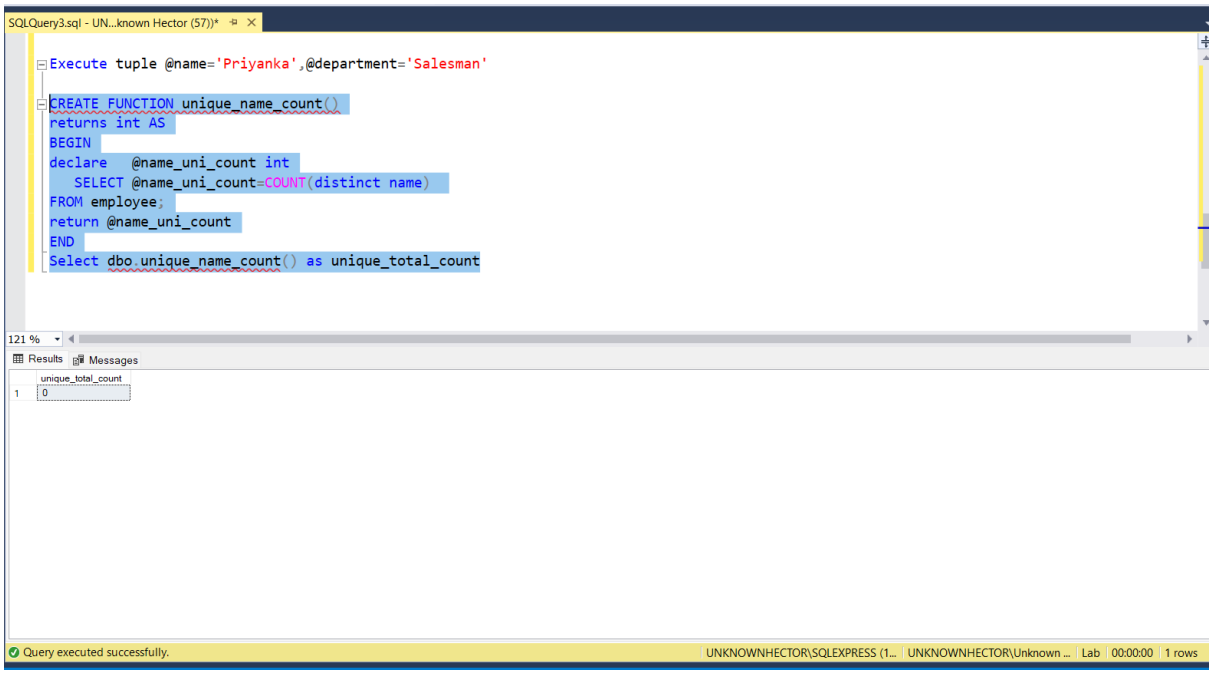


4. **Create a PL/SQL function that displays a unique count of attribute Employee.Name.**

Program:

```
CREATE FUNCTION unique_name_count()
returns int AS
BEGIN
declare   @name_uni_count int
   SELECT @name_uni_count=COUNT(distinct name)
FROM employee;
return @name_uni_count
END
        Select dbo.unique_name_count() as unique_total_count
```

Output:



5. **Create a PL/SQL cursor that displays Employee.Employee_ID, Employee.Name and Employee.Department using FETCH NEXT and deallocate it.**
   Program:

```
DECLARE my_cursor CURSOR FOR SELECT Employee_id, Name, Department FROM Employes;

OPEN my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

DEALLOCATE my_cursor;
```

Output:

SQLQuery3.sql - UN...known Hector (57))*

```sql
FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

DEALLOCATE my_cursor;
```

121 %

Results | Messages

	Employee_id	Name	Department
1	1	Adiya	Executive
1	2	Aditya	Accountant
1	3	Priyanka	Salesman
1	4	Anmol	Executive
1	5	Rahul	Accountant
1	6	Shubham	Salesman
1	7	Sam	Salesman
1	8	Rohan	Salesman
1	9	Priya	Accountant
1	10	Kabir	Executive

Query executed successfully. | UNKNOWNHECTOR\SQLEXPRESS (1... | UNKNOWNHECTOR\Unknown ... | Lab | 00:00:00 | 10 rows

SQLQuery3.sql - UN...known Hector (57))*

```sql
FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

DEALLOCATE my_cursor;
```

121 %

Messages

```
Commands completed successfully.

Completion time: 2022-04-20T16:31:18.3023347+05:30
```

121 %

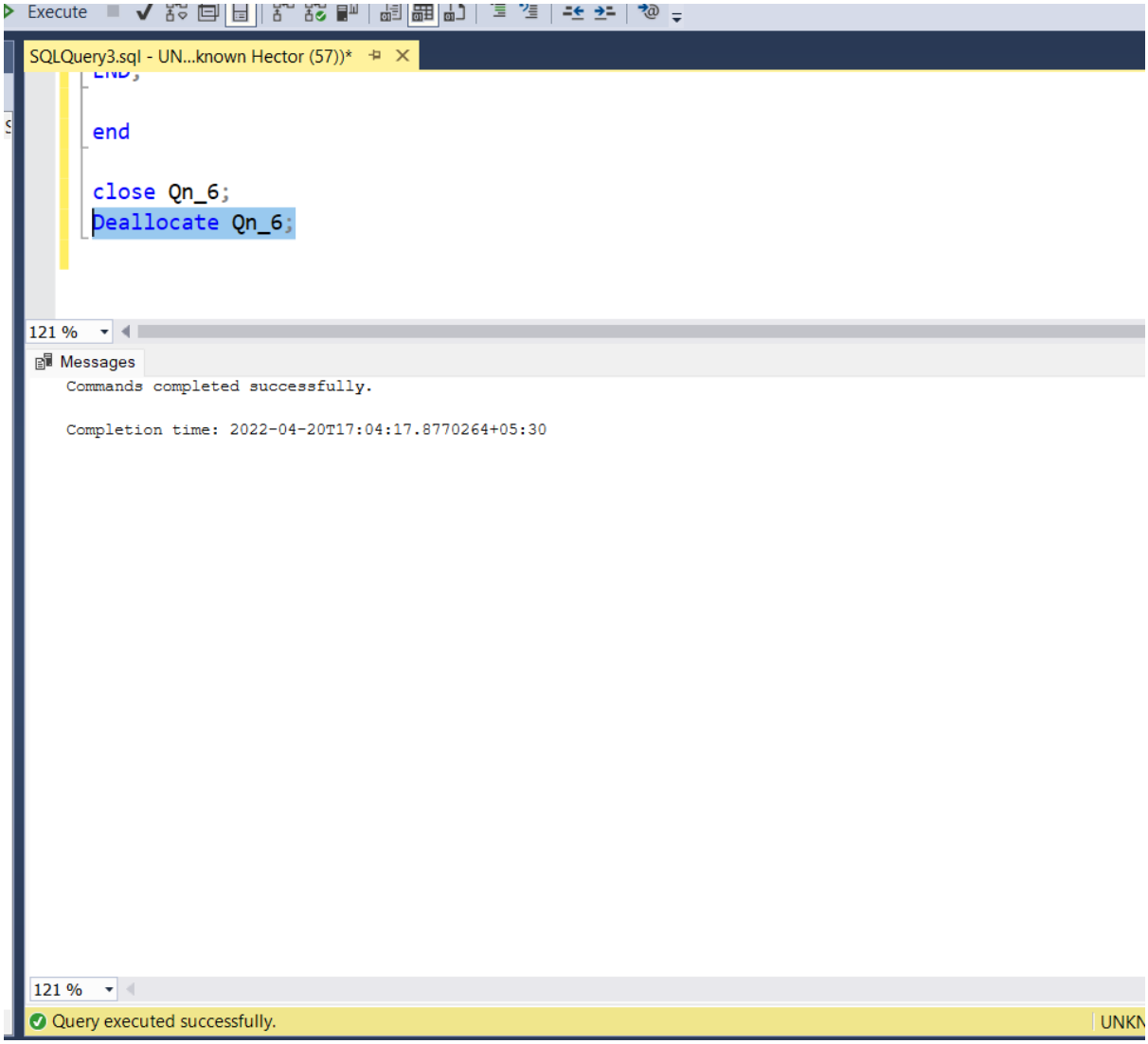
Query executed successfully. | UNKNOWNHECTOR\SQLEXPRESS (1... | UNKNOWNHECTOR\Unknown ...

6. Create a PL/SQL block that creates a cursor inside it , prints all tuples for attributes Employee.Employee_ID, Employee.Name and Employee.Salary , close it and later deallocate it.

Program:

```
Declare @v int
begin

DECLARE Qn_6 CURSOR FOR SELECT Employee_ID,Name,Salary FROM Employes;
OPEN Qn_6;
SET @v = 0;
WHILE @v < 10
BEGIN
FETCH NEXT FROM Qn_6;
SET @v = @v + 1;
END;

end

close Qn_6;

Deallocate Qn_6;
```

Output:

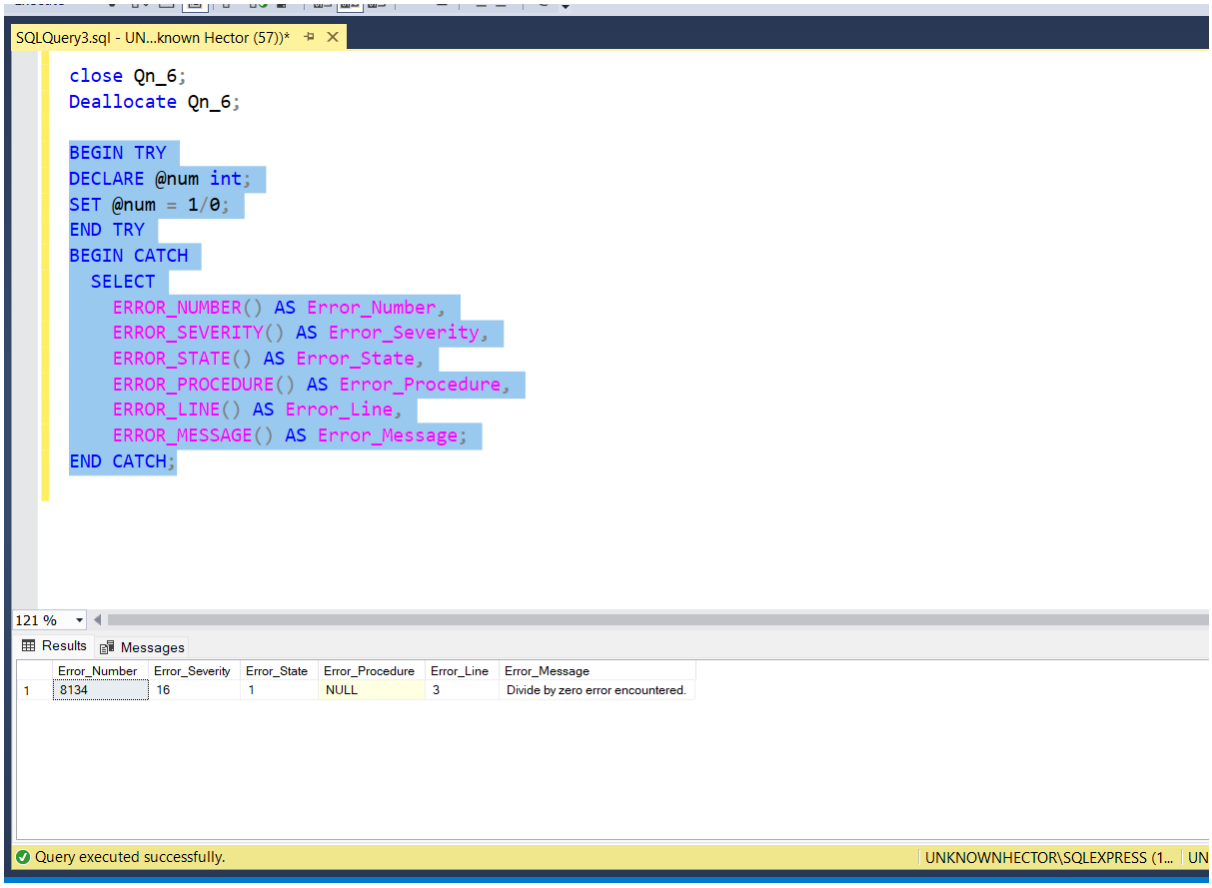


7. Create a PL/SQL try-catch exception block in which the try block is performing "divided by 0" operation and catch block displays Error_number, Error_Severity, Error_State, Error_Procedure, Error_Line and Error_Message.

Program:

```
BEGIN TRY
DECLARE @num int;
SET @num = 1/0;
END TRY
BEGIN CATCH
  SELECT
    ERROR_NUMBER() AS Error_Number,
       ERROR_SEVERITY() AS Error_Severity,
    ERROR_STATE() AS Error_State,
    ERROR_PROCEDURE() AS Error_Procedure,
    ERROR_LINE() AS Error_Line,
    ERROR_MESSAGE() AS Error_Message;
        END CATCH;
```

Output:

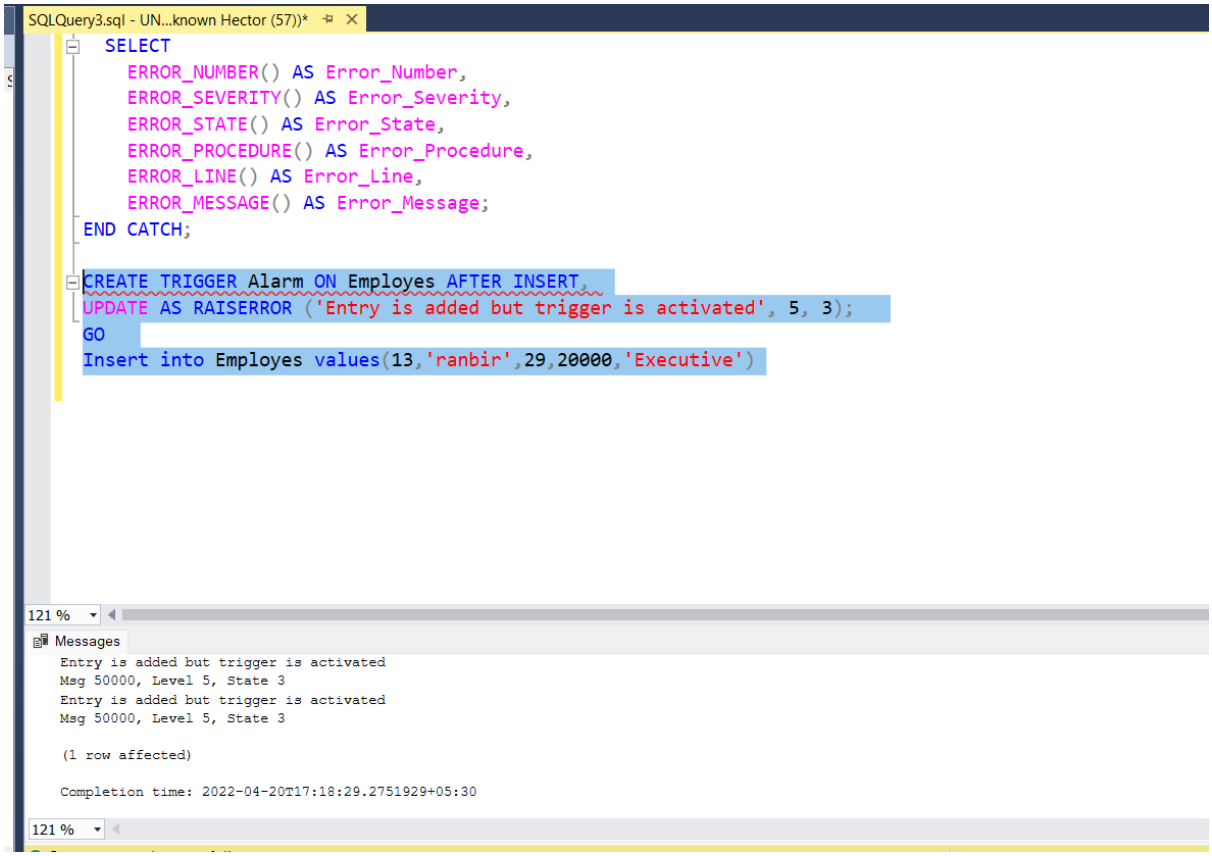


8. Create a PL/SQL trigger that Raise an Error on insert operation of level 5 , state 3 which displays a message "Entry is added but trigger is activated" after insert operation is performed.

Program:

```
CREATE TRIGGER Alarm ON Employes AFTER INSERT,
UPDATE AS RAISERROR ('Entry is added but trigger is activated', 5, 3);
GO
        Insert into Employes values(13,'ranbir',29,20000,'Executive')
```

Output:

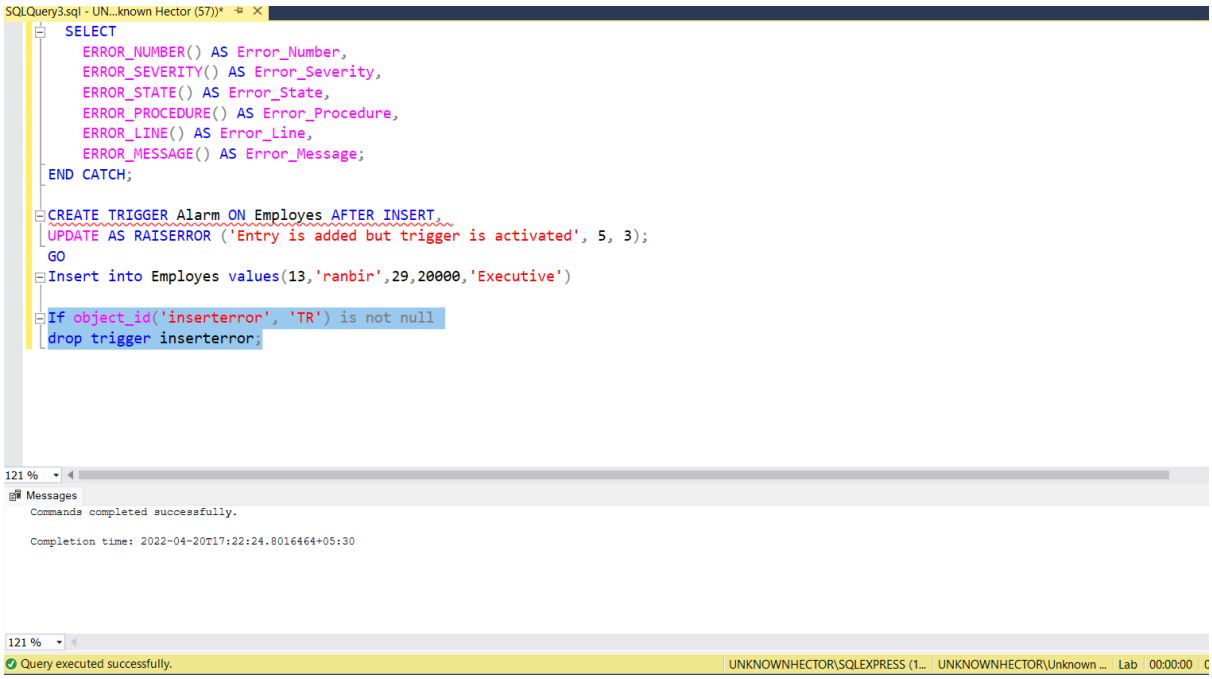


9. Delete the trigger created in question 8.

Program:

```
If object_id('inserterror', 'TR') is not null
        drop trigger inserterror;
```

Output:



10. Create a PL/SQL recursive function that takes a number "n" as parameter and displays the sum "n+(n-1(+(n-2)+.....+2+1

Program:

```
CREATE FUNCTION totalsum(@n int)
RETURNS INT
AS
BEGIN
        DECLARE @ans int;
        IF @n = 0 or @n = 1
        BEGIN
                SET @ans = 1;
        END
        ELSE
        BEGIN
                SET @ans = @n + dbo.totalsum(@n-1);
        END
        RETURN @ans;
END

        SELECT dbo.totalsum(10)
```

Output:

SQLQuery3.sql - UN...known Hector (57))*

```sql
CREATE FUNCTION totalsum(@n int)
RETURNS INT
AS
BEGIN
    DECLARE @ans int;
    IF @n = 0 or @n = 1
    BEGIN
        SET @ans = 1;
    END
    ELSE
    BEGIN
        SET @ans = @n + dbo.totalsum(@n-1);
    END
    RETURN @ans;
END

SELECT dbo.totalsum(10)
```

121 %

Results | Messages

	(No column name)
1	55

All Code:-

```
CREATE DATABASE LAB08

USE LAB

CREATE TABLE Employes
(
Employee_ID VARCHAR(50),
Name VARCHAR(50),
Age INT,
Salary INT,
Department VARCHAR(50)
);

INSERT INTO Employes VALUES('1','Adiya',25,30000,'Executive');
INSERT INTO Employes VALUES('2','Aditya',28,31000,'Accountant');
INSERT INTO Employes VALUES('3','Priyanka',30,32000,'Salesman');
INSERT INTO Employes VALUES('4','Anmol',35,31000,'Executive');
INSERT INTO Employes VALUES('5','Rahul',31,29000,'Accountant');
INSERT INTO Employes VALUES('6','Shubham',29,27000,'Salesman');
INSERT INTO Employes VALUES('7','Sam',33,32000,'Salesman');
INSERT INTO Employes VALUES('8','Rohan',33,28000,'Salesman');
INSERT INTO Employes VALUES('9','Priya',23,35000,'Accountant');
INSERT INTO Employes VALUES('10','Kabir',29,26000,'Executive');

SELECT*FROM Employes;

DECLARE @E_name varchar(20)= 'Aditya';
DECLARE @E_age int = 28;
DECLARE @E_salary int
BEGIN
       SELECT @E_salary = Salary FROM Employes WHERE Name = @E_name AND Age = @E_age;
       PRINT 'Salary of ' + @E_name + ' whose age is ' + cast(@E_age as varchar) + '
is ' + cast(@E_salary as varchar);
END

Create Procedure tuple
   @name varchar(40),
   @department varchar(40)
As
  Select *from Employes where Name=@name and Department=@department;
Go

Execute tuple @name='Priyanka',@department='Salesman'

CREATE FUNCTION unique_name_count()
returns int AS
BEGIN
declare   @name_uni_count int
   SELECT @name_uni_count=COUNT(distinct name)
FROM employee;
return @name_uni_count
END
Select dbo.unique_name_count() as unique_total_count

DECLARE my_cursor CURSOR FOR SELECT Employee_id, Name, Department FROM Employes;

OPEN my_cursor;
```

```sql
FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

FETCH NEXT FROM my_cursor;

DEALLOCATE my_cursor;

Declare @v int
begin

DECLARE Qn_6 CURSOR FOR SELECT Employee_ID,Name,Salary FROM Employes;
OPEN Qn_6;
SET @v = 0;
WHILE @v < 10
BEGIN
FETCH NEXT FROM Qn_6;
SET @v = @v + 1;
END;

end

close Qn_6;
Deallocate Qn_6;

BEGIN TRY
DECLARE @num int;
SET @num = 1/0;
END TRY
BEGIN CATCH
  SELECT
    ERROR_NUMBER() AS Error_Number,
       ERROR_SEVERITY() AS Error_Severity,
    ERROR_STATE() AS Error_State,
    ERROR_PROCEDURE() AS Error_Procedure,
    ERROR_LINE() AS Error_Line,
    ERROR_MESSAGE() AS Error_Message;
END CATCH;

CREATE TRIGGER Alarm ON Employes AFTER INSERT,
UPDATE AS RAISERROR ('Entry is added but trigger is activated', 5, 3);
GO
Insert into Employes values(13,'ranbir',29,20000,'Executive')

If object_id('inserterror', 'TR') is not null
       drop trigger inserterror;
```

```sql
CREATE FUNCTION totalsum(@n int)
RETURNS INT
AS
BEGIN
	DECLARE @ans int;
	IF @n = 0 or @n = 1
	BEGIN
		SET @ans = 1;
	END
	ELSE
	BEGIN
		SET @ans = @n + dbo.totalsum(@n-1);
	END
	RETURN @ans;
END

SELECT dbo.totalsum(10)
```