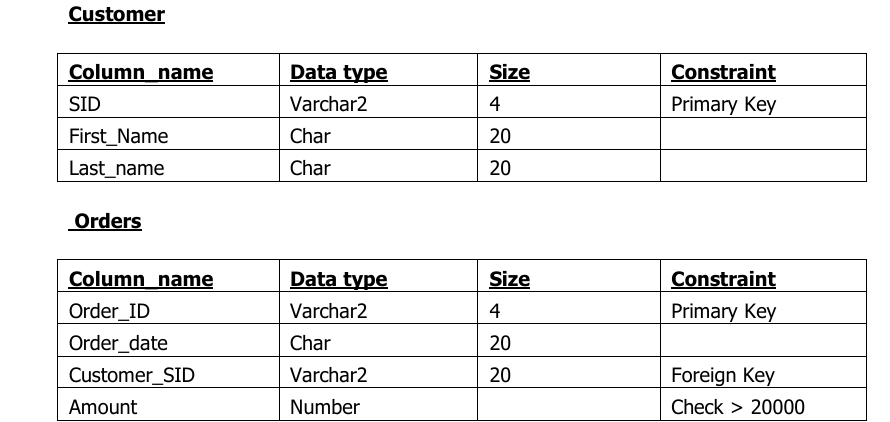
**Manav Rachna International Institute of Research and Studies**

**ORACLE LAB**

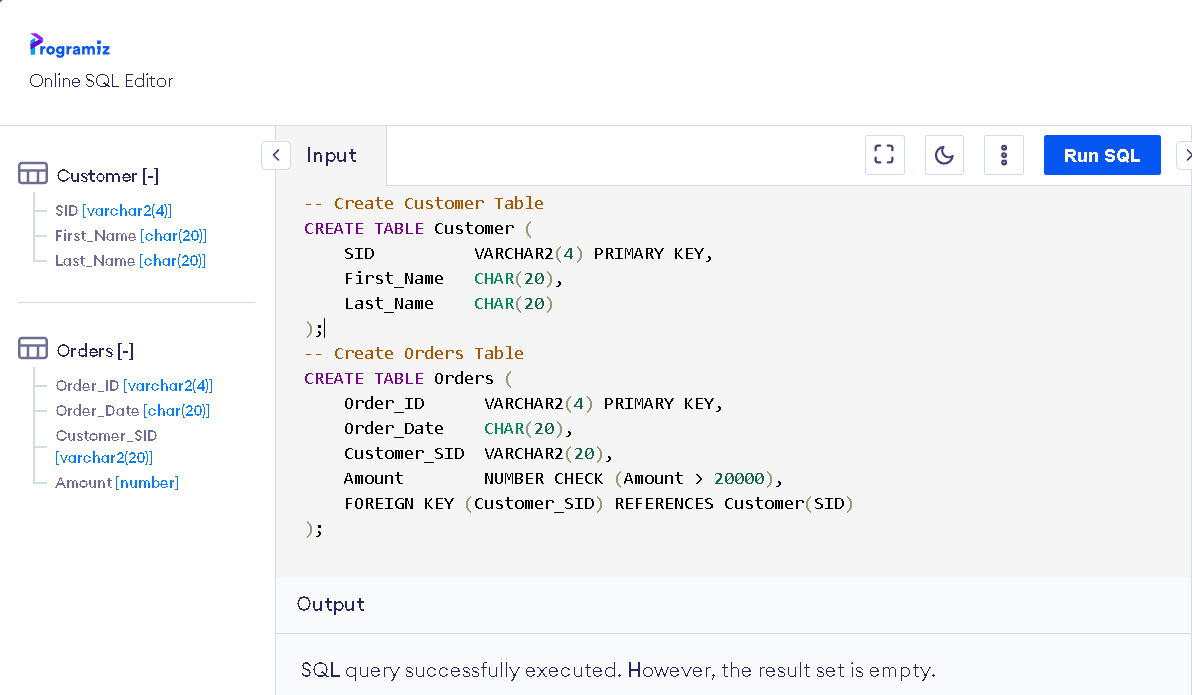
**Jasdeep Singh 22/FCA/BCA(CS)/019**

**EXERCISE 1**

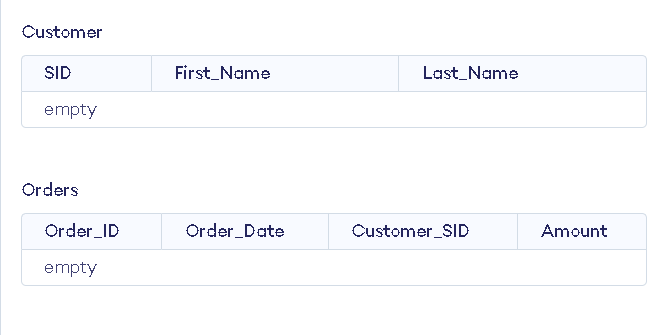
**AIM: Create the following table**

****

**Table Creation:**

****

**Output:**

****

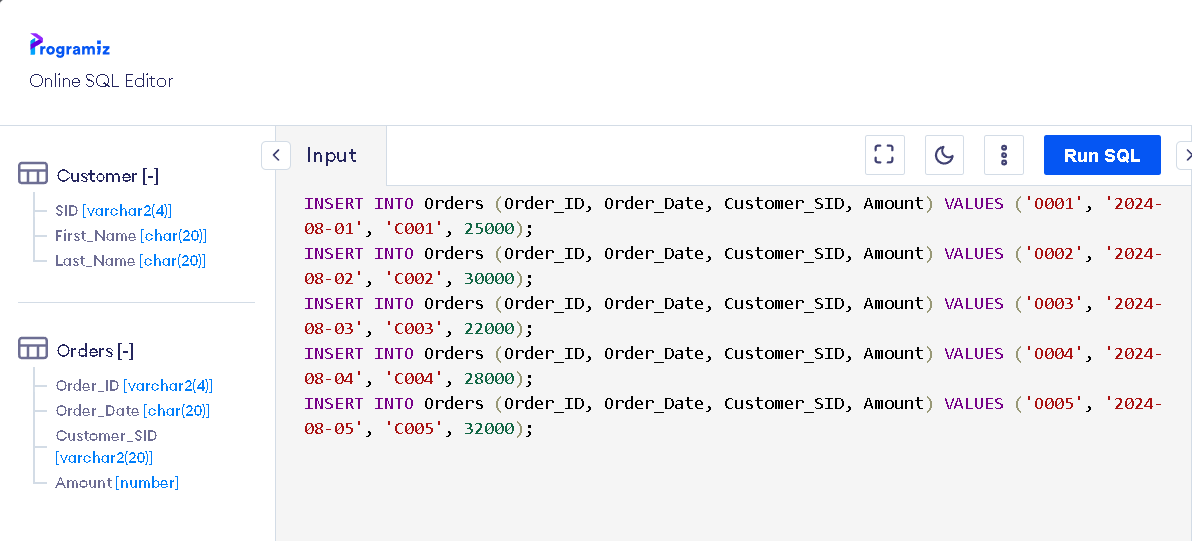
**EXERCISE 2**

**AIM: Insert 5 records for each table.**

**Inserting record in Customer ->**

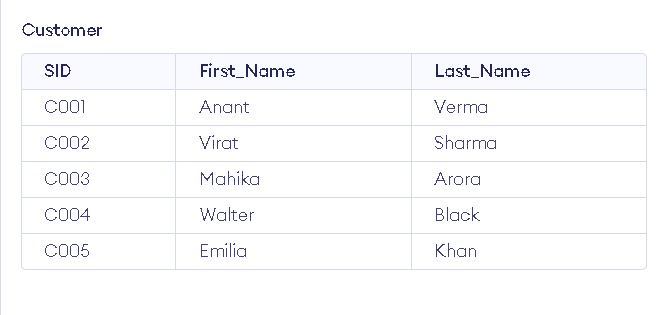
****

**Inserting records Order ->**

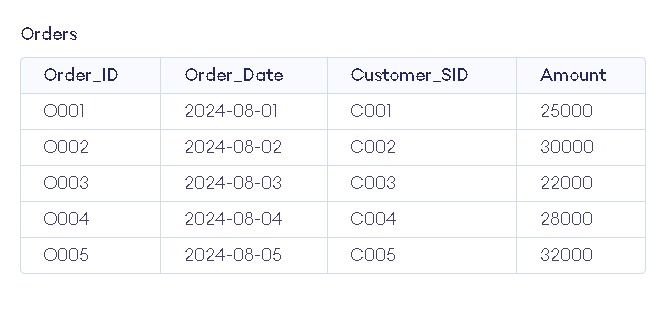
****

**Output:**

**Table output for Customer**

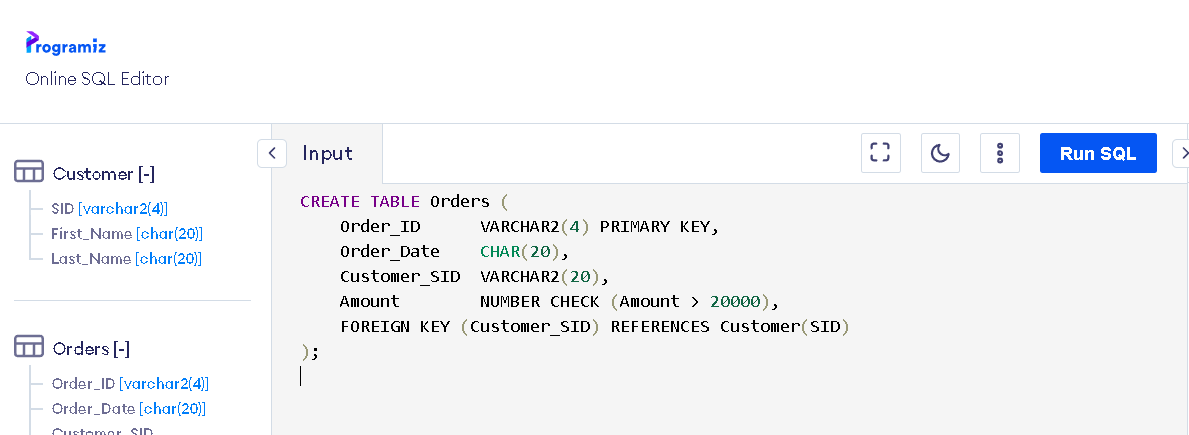
****

**Table output for Orders**

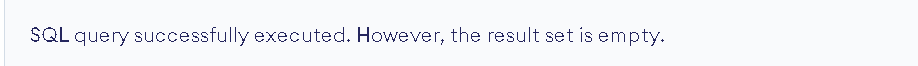
****

**EXERCISE 3**

**AIM: Customer SID column in the ORDERS table is a foreign key pointing to the SID column in the CUSTOMER table.**

****

**Output:**

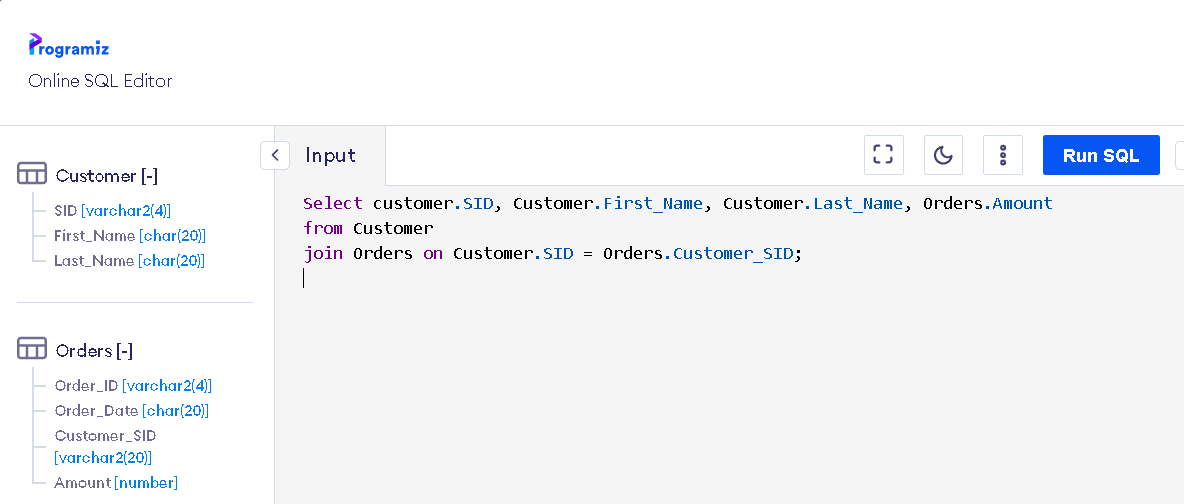
****

**In this code:**

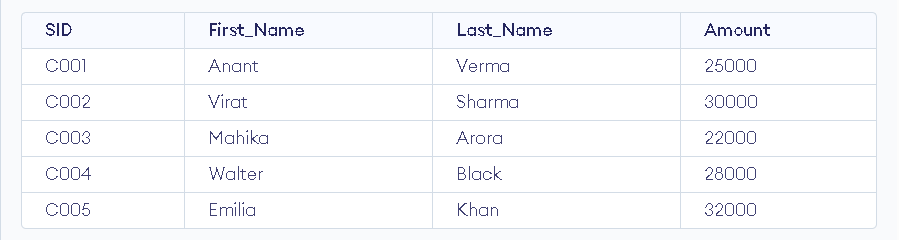
* **The ‘Customer\_SID’ column is a foreign key that links each order to a specific customer.**
* **The foreign key constraint ensures that any ‘Customer\_SID’ in the Orders table must match an existing SID in the Customer table. This prevents orders from being associated with non-existent customers.**

**EXERCISE 4**

**AIM: List the details of the customers along with the amount.**

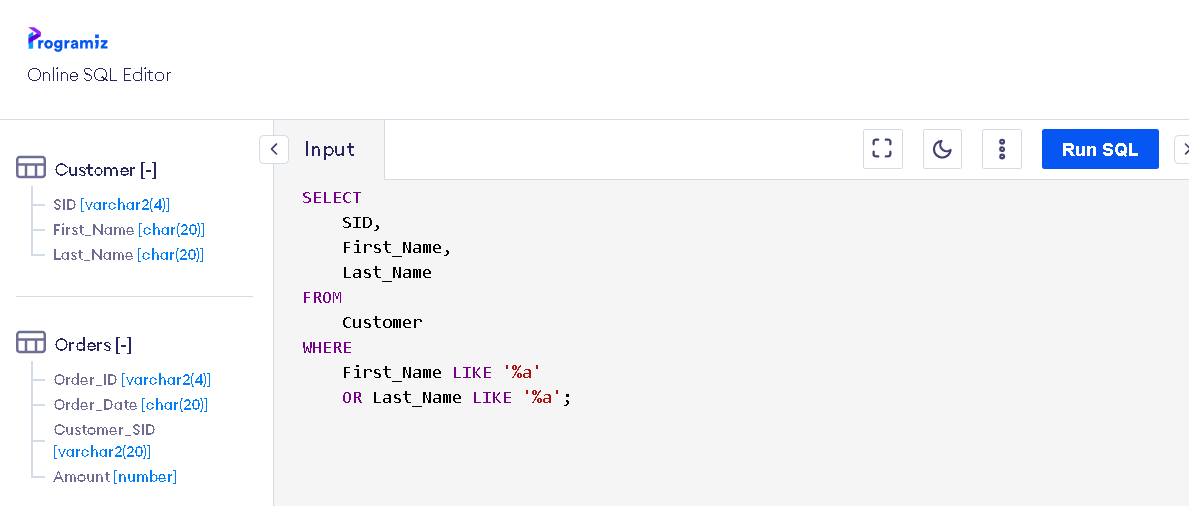
****

**Output:**

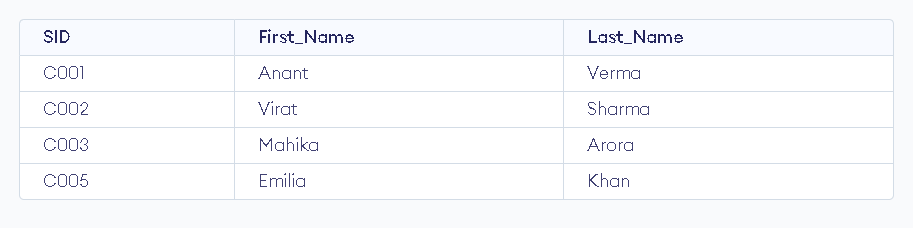
****

**EXERCISE 5**

**AIM: List the customers whose names end with “a”.**

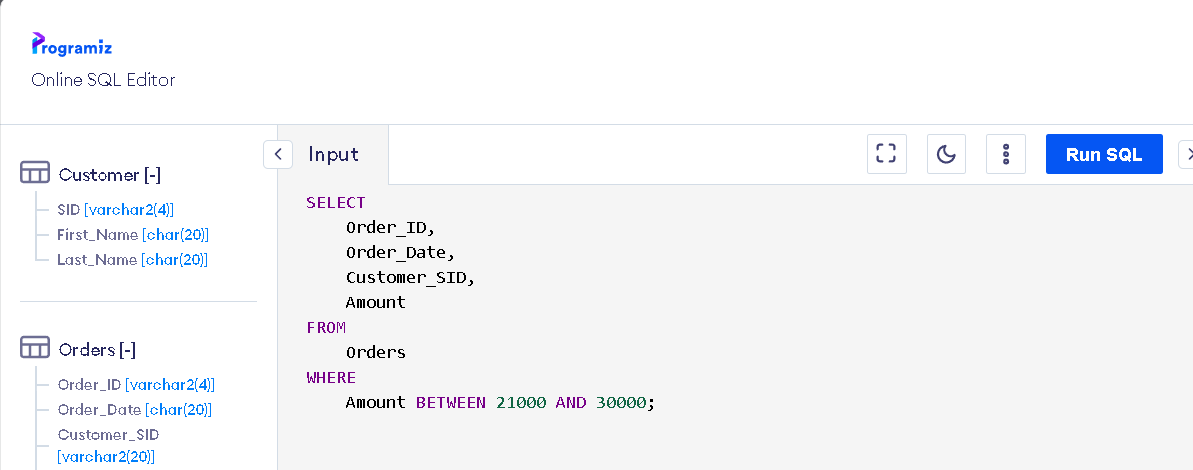
****

**Output:**

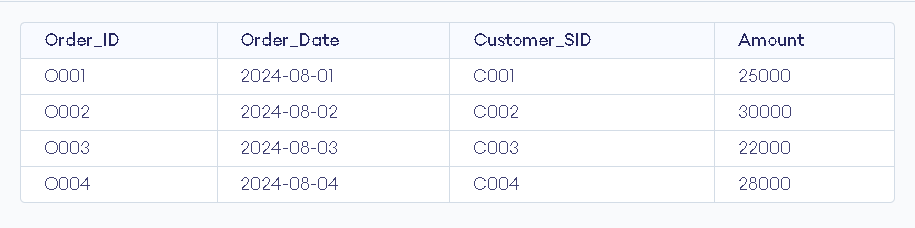
****

**EXERCISE 6**

**AIM: List the orders where amount is between 21000 and 30000**

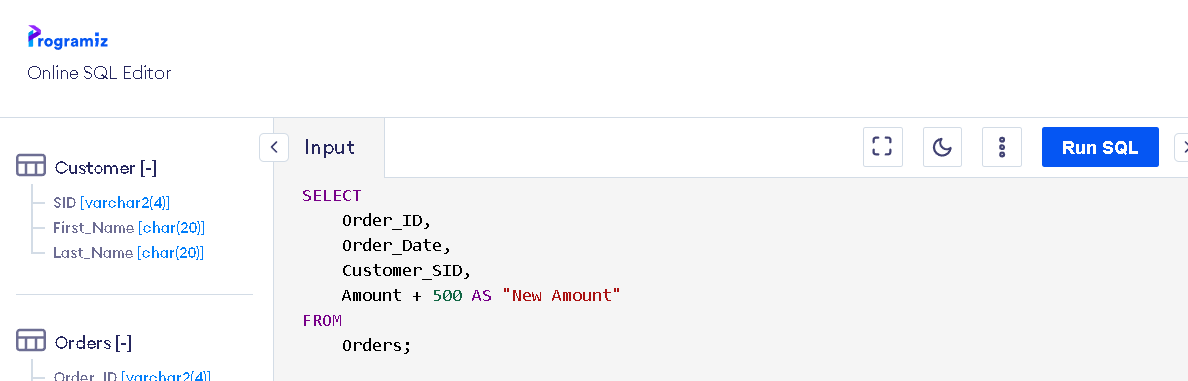
****

**Output:**

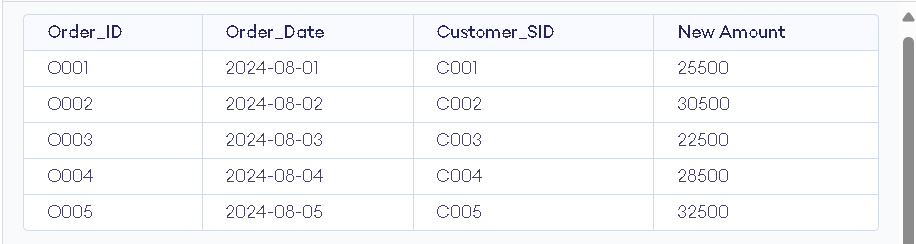
****

**EXERCISE 7**

**AIM: List the orders where amount is increased by 500 and replace with name “new amount”**

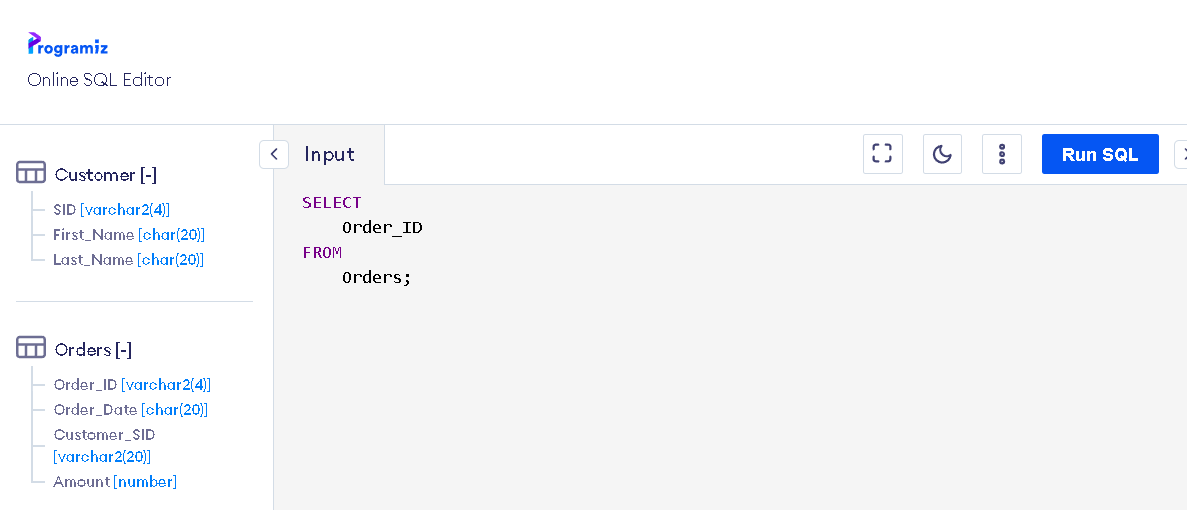
****

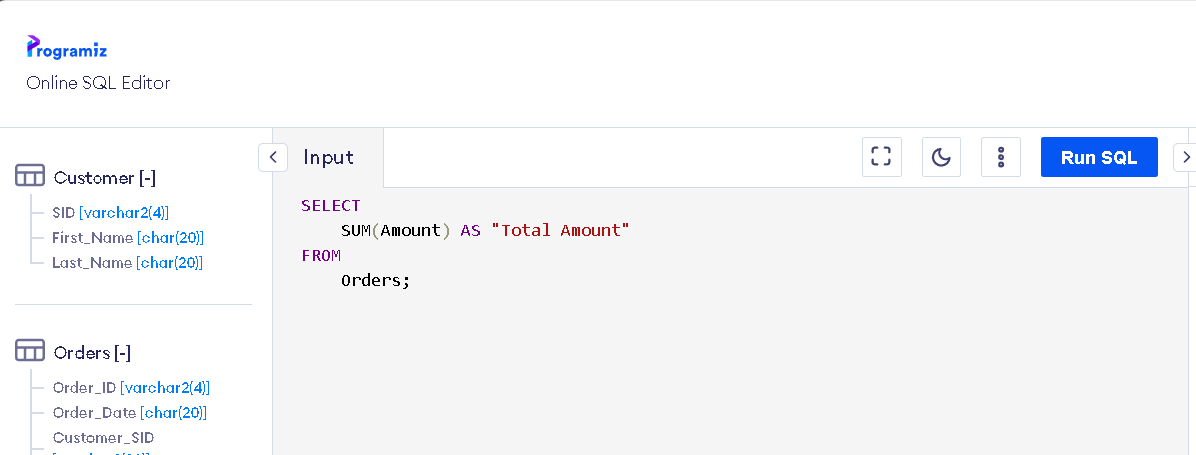
**Output:**

****

**EXERCISE 8**

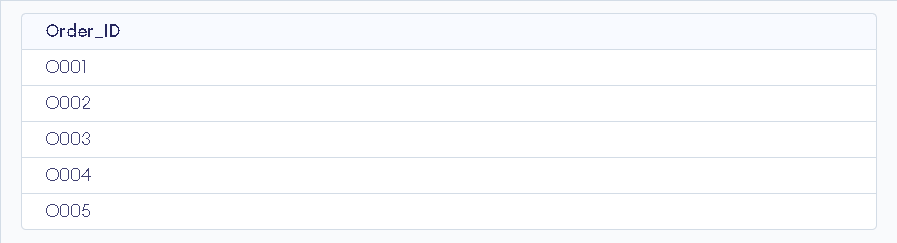
**AIM: Display the order\_id and total amount of orders.**

****

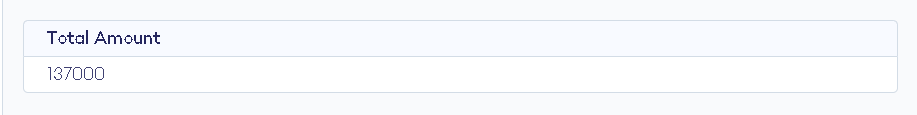
****

**Output:**

**Table output for Order\_ID from Orders ->**

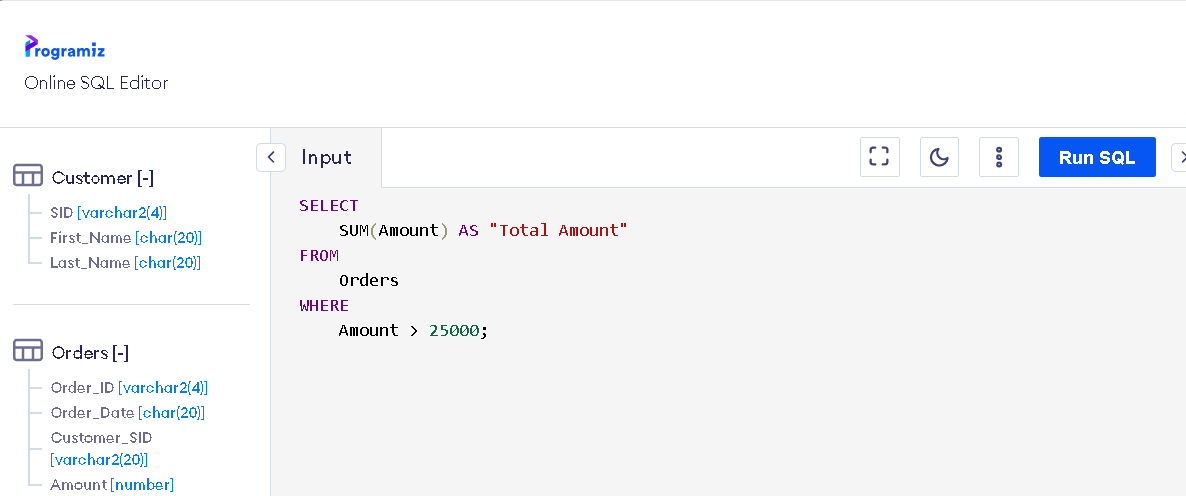
****

**Output for Total amount of all Orders ->**

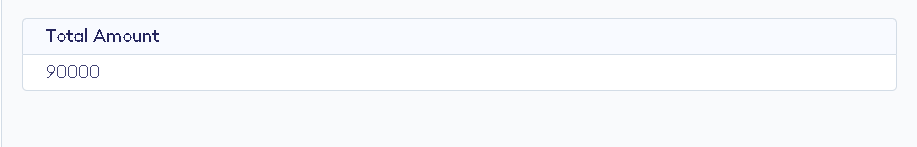
****

**EXERCISE 9**

**AIM: Calculate the total amount of orders that has more than 25000.**

****

**Output:**

****

**EXERCISE 10**

**AIM: Display all the string functions used in SQL.**

**1. LENGTH (string)**

* **Returns the length of a string.**

**2. LOWER (string)**

* **Converts all characters in a string to lowercase.**

**3. UPPER (string)**

* **Converts all characters in a string to uppercase.**

**4. SUBSTR (string, start\_position, length)**

* **Extracts a substring from a string starting at a given position for a specified length.**

**5. INSTR (string, substring)**

* **Returns the position of the first occurrence of a substring within a string.**

**6. TRIM ([LEADING | TRAILING | BOTH] trim\_character FROM string)**

* **Removes specified characters from the beginning (LEADING), end (TRAILING), or both ends (BOTH) of a string. By default, it removes spaces.**

**7. LTRIM (string)**

* **Removes leading spaces from a string.**

**8. RTRIM (string)**

* **Removes trailing spaces from a string.**

**9. REPLACE (string, search\_string, replace\_string)**

* **Replaces occurrences of a substring within a string with another substring.**

**10. CONCAT (string1, string2)**

* **Concatenates two or more strings together.**

**11. LPAD (string, length, pad\_string)**

* **Pads the left side of a string with a specified character up to a certain length.**

**12. RPAD (string, length, pad\_string)**

* **Pads the right side of a string with a specified character up to a certain length.**

**13. LEFT (string, number\_of\_characters)**

* **Returns the leftmost n characters from a string.**

**14. RIGHT (string, number\_of\_characters)**

* **Returns the rightmost n characters from a string.**

**15. ASCII (character)**

* **Returns the ASCII code of the first character in a string.**

**16. CHR (ascii\_code)**

* **Converts an ASCII code to its corresponding character.**

**17. INITCAP (string)**

* **Converts the first letter of each word in a string to uppercase and the rest to lowercase.**

**18. REVERSE (string) (Available in some SQL variants like SQL Server)**

* **Reverses the characters in a string.**

**19. POSITION (substring IN string)**

* **Returns the position of the first occurrence of a substring within a string.**

**20. SOUNDEX (string)**

* **Returns a string's phonetic representation, useful for comparing words that sound alike.**

**21. DIFFERENCE (string1, string2) (Available in SQL Server)**

* **Compares two strings and returns a value based on their phonetic similarity.**

**22. FORMAT (number, format) (Available in SQL Server)**

* **Returns a number formatted as a string, according to a specified format.**

**23. TRANSLATE (string, from\_chars, to\_chars) (Available in Oracle SQL)**

* **Replaces characters in a string with other characters based on their position.**

**24. REPEAT (string, number) (Available in MySQL)**

* **Repeats a string a specified number of times.**

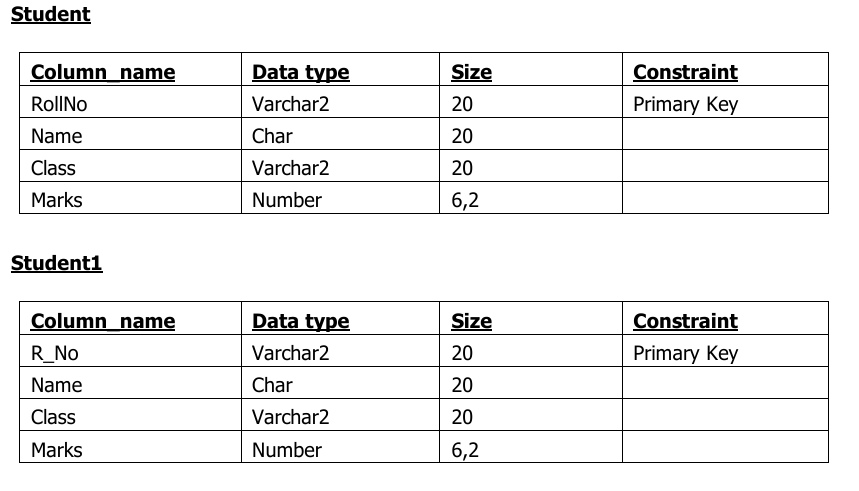
**25. SPACE (number) (Available in SQL Server)**

* **Returns a string of spaces with the specified length.**

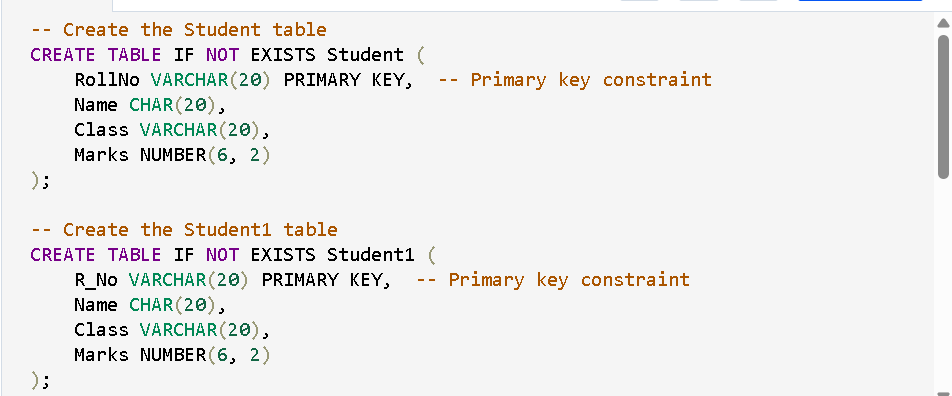
**These functions are commonly used for manipulating and querying string data across different SQL databases. The availability of these functions may vary depending on the specific SQL database you are using.**

**EXERCISE 11**

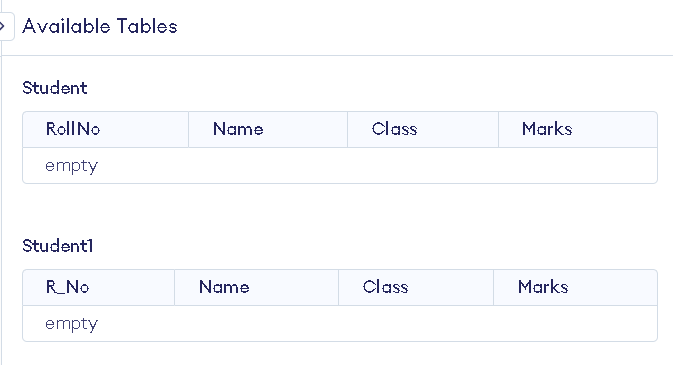
**AIM: Create the following table:**

****

**Table creation:**

****

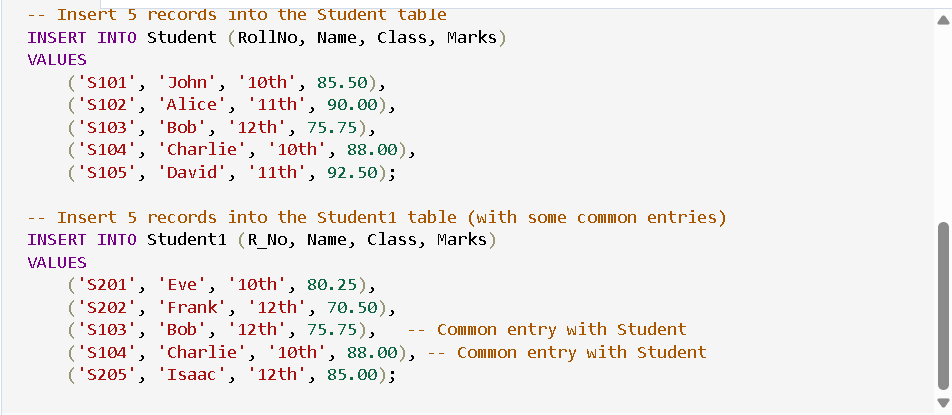
**Output:**

****

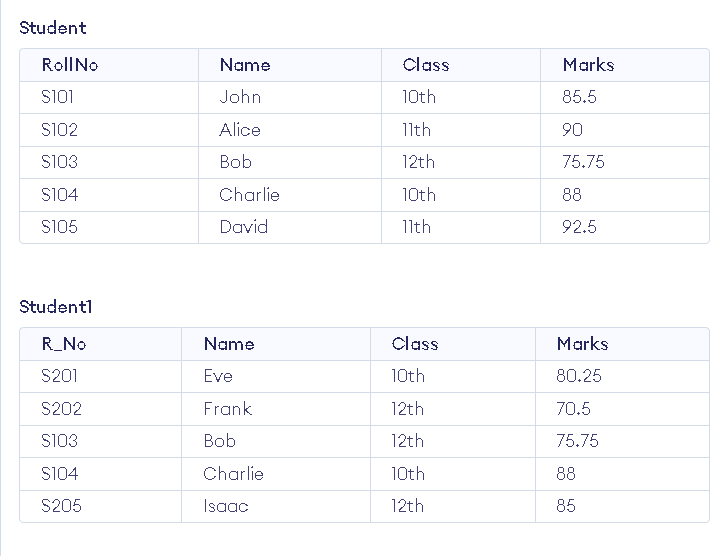
**EXERCISE 12**

**AIM: Display all the contents of student and student1 using union clause.**

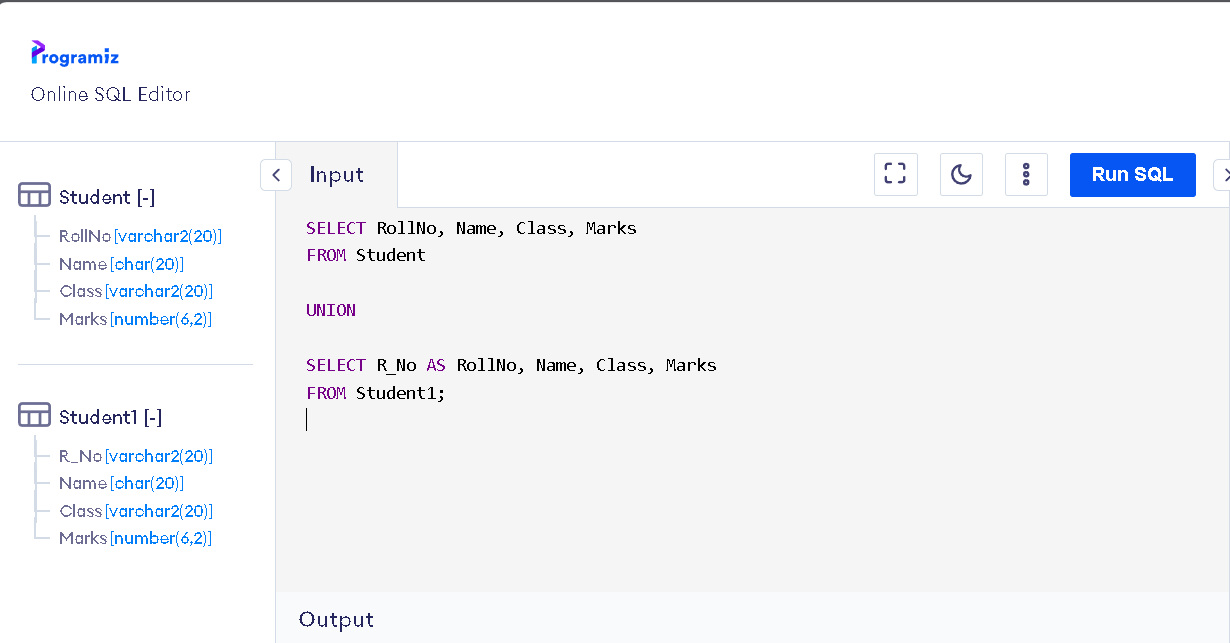
**First insert 5 records in each table i.e. Student and Student1**

****

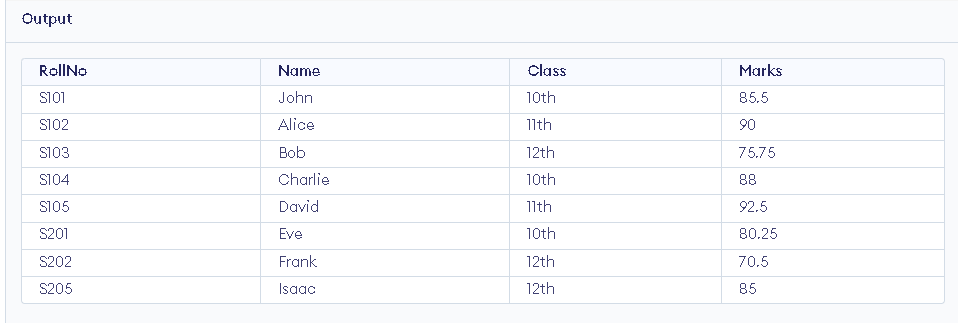
**Output:**

****

**Now union:**

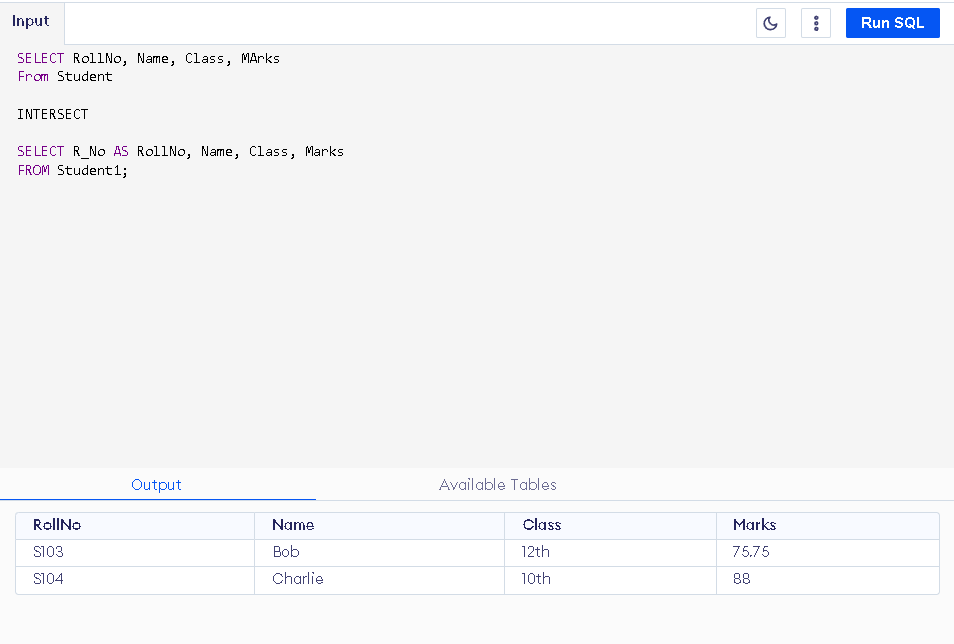
****

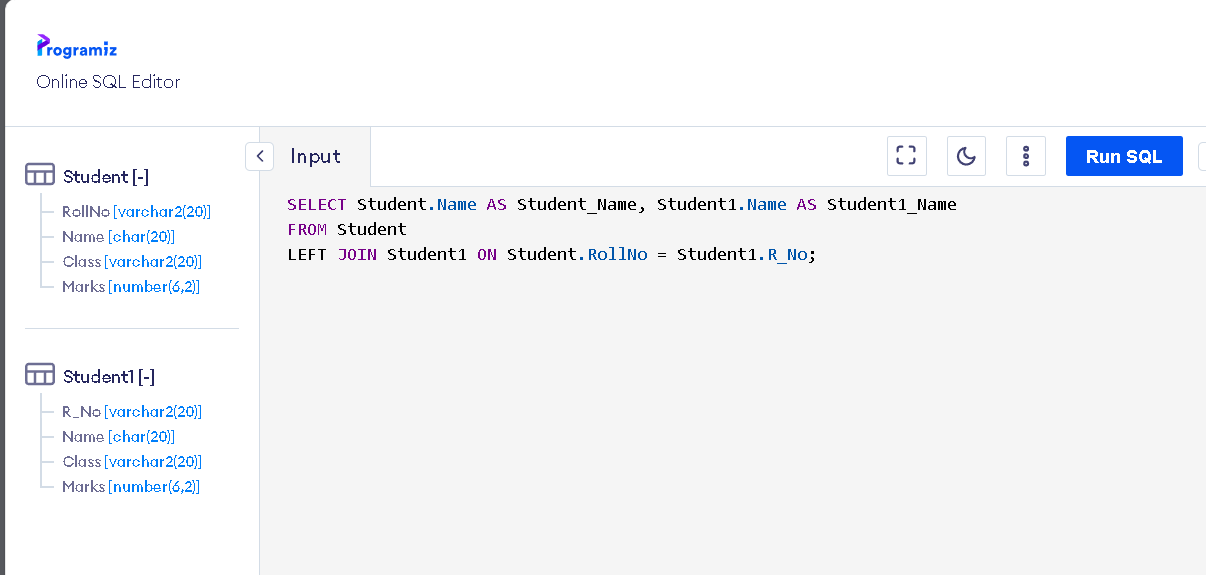
**Output:**

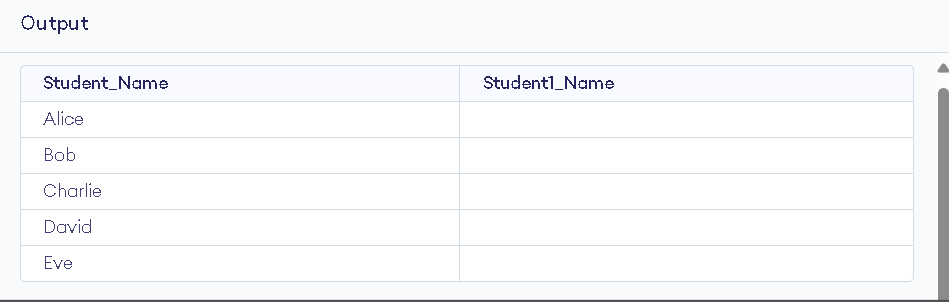
****

**EXERCISE 13**

**AIM: Find out the intersection of student and student1 tables.**

****

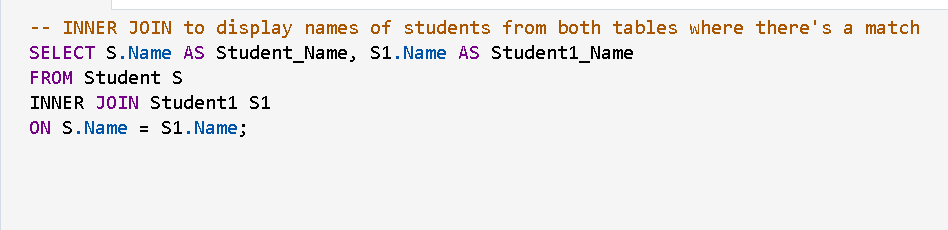
****

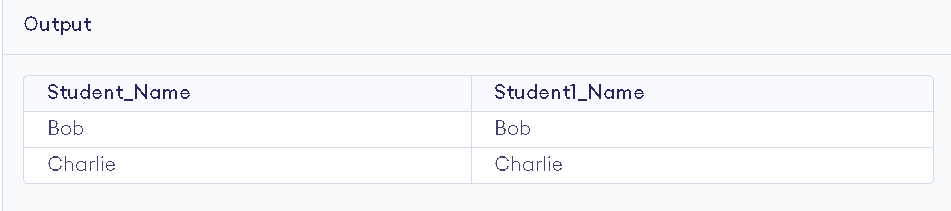
****

**EXERCISE 14**

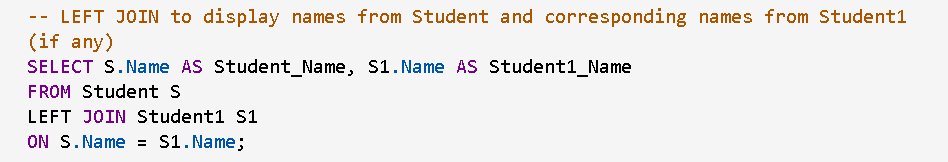
**AIM: Display the names of student and student1 tables using left, right, inner and full join.**

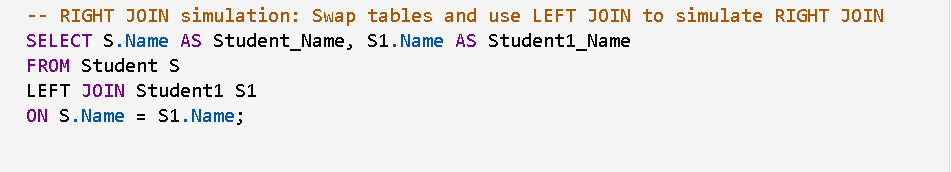
**INNER JOIN**

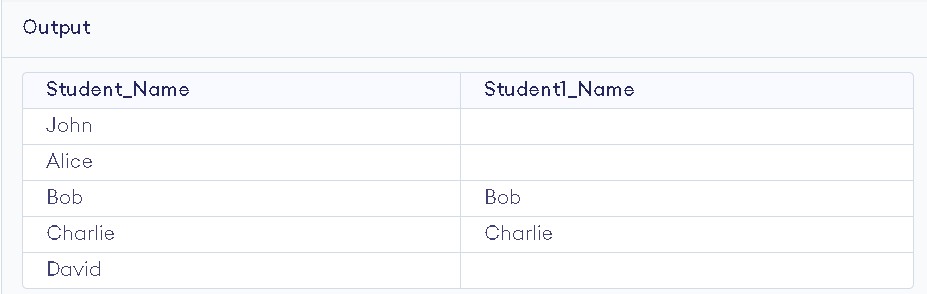
****

****

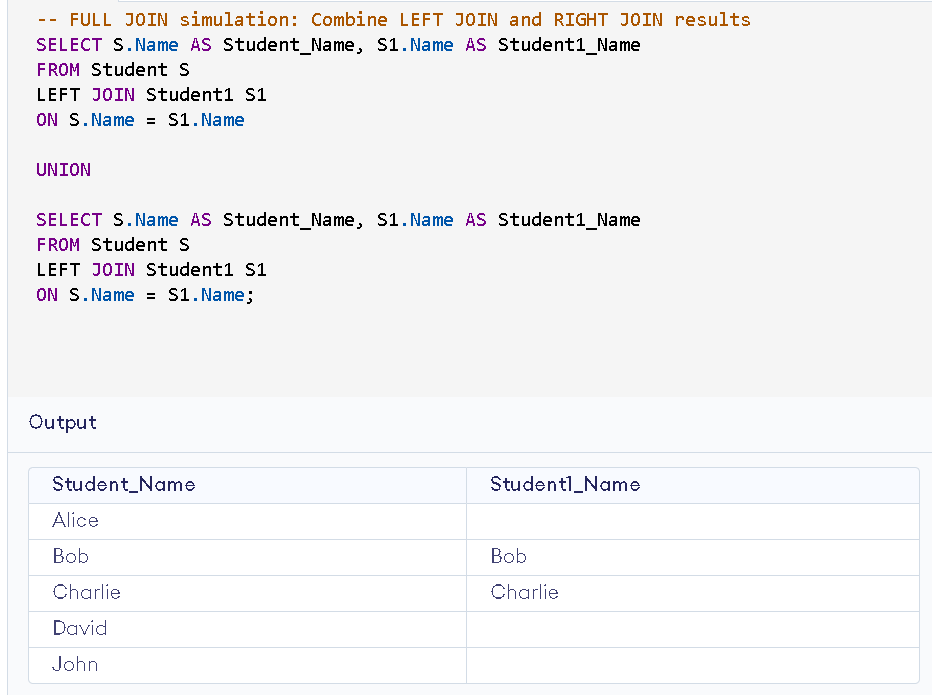
**LEFT JOIN AND RIGHT JOIN**

****

****

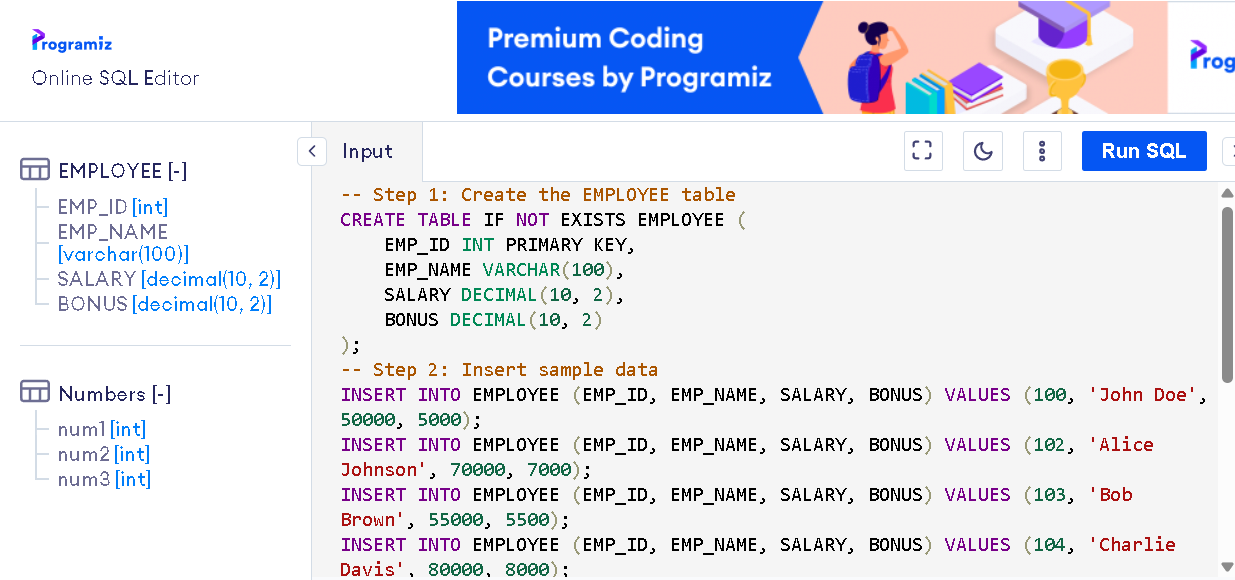
****

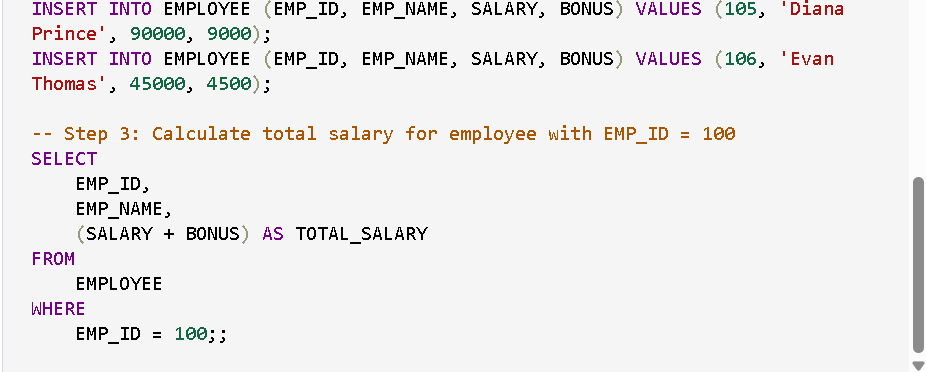
**FULL JOIN**

****

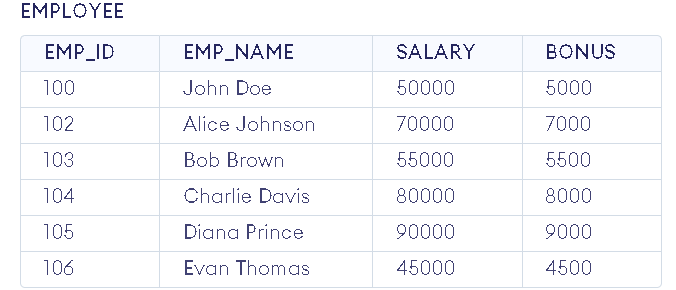
**Exercise 15**

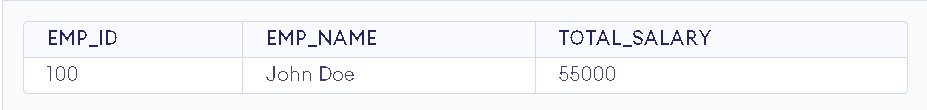
**AIM: To Write a PL/SQL block to calculate total salary of employee having employee number 100.**

****

****

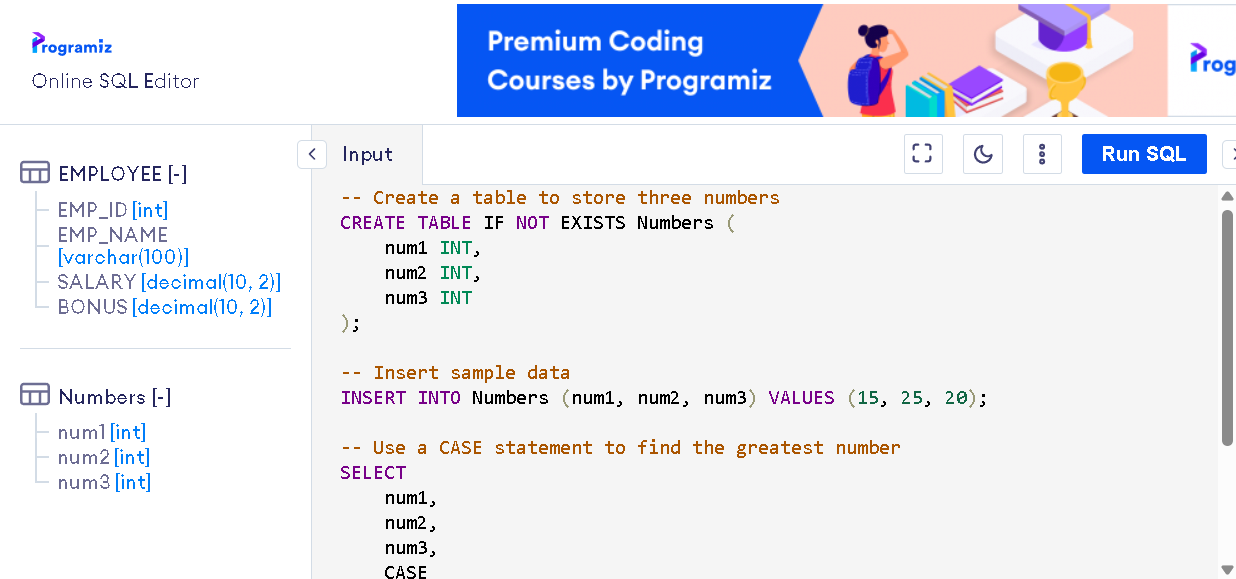
**OUTPUT:**

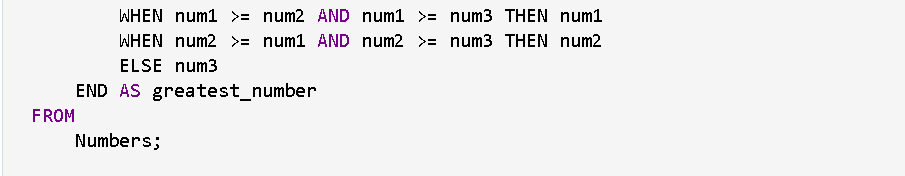
****

****

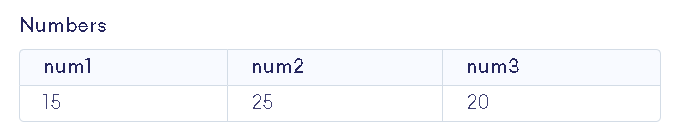
**EXERCISE 16**

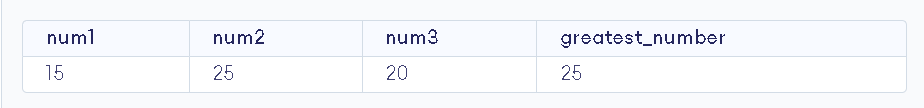
**AIM: To Write a PL/SQL code to find the greatest of three numbers.**

****

****

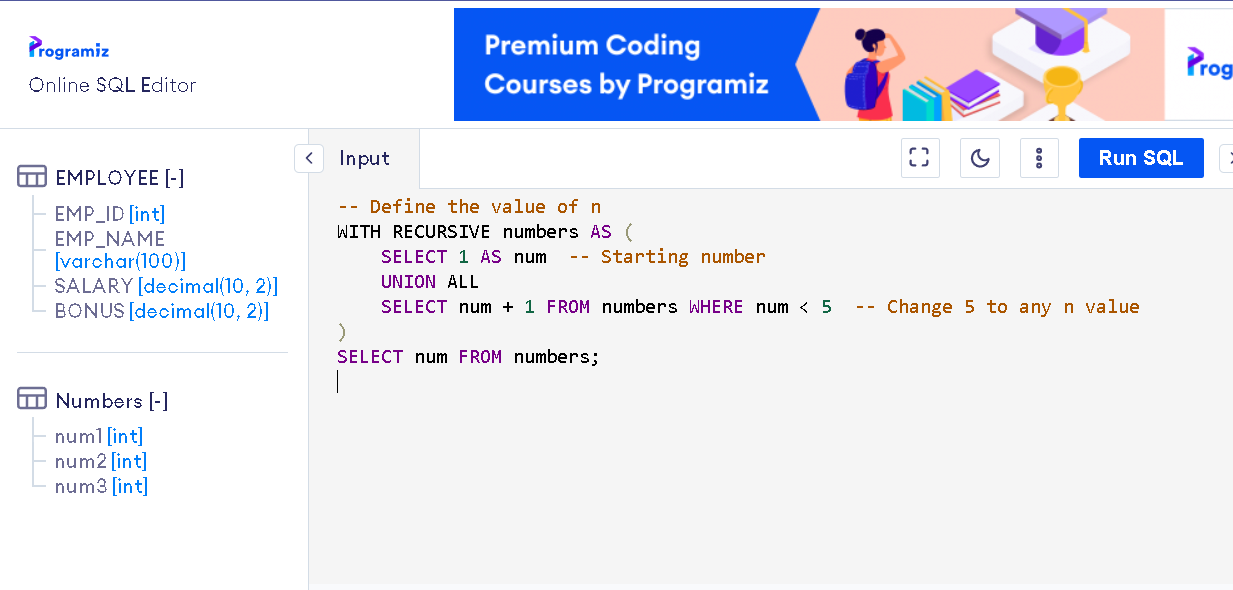
**OUTPUT:**

****

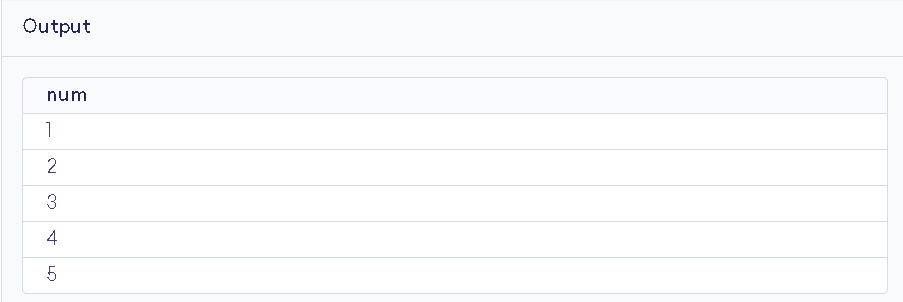
****

**EXERCISE 17**

**AIM: To Write a PL/SQL code to print the numbers from 1 to n.**

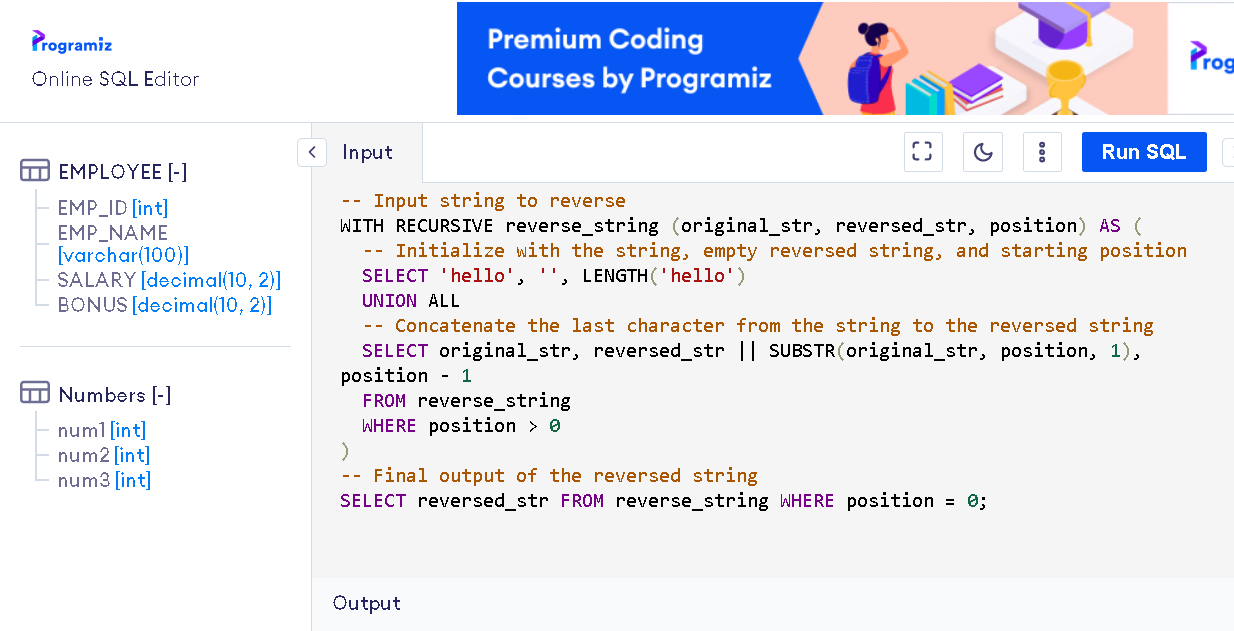
****

**OUTPUT:**

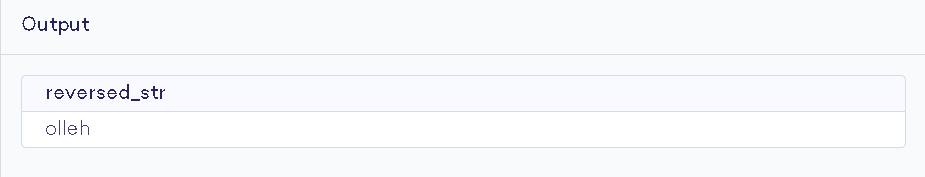
****

**EXERCISE 18**

**AIM: To Write a PL/SQL code to reverse a string using for loop.**

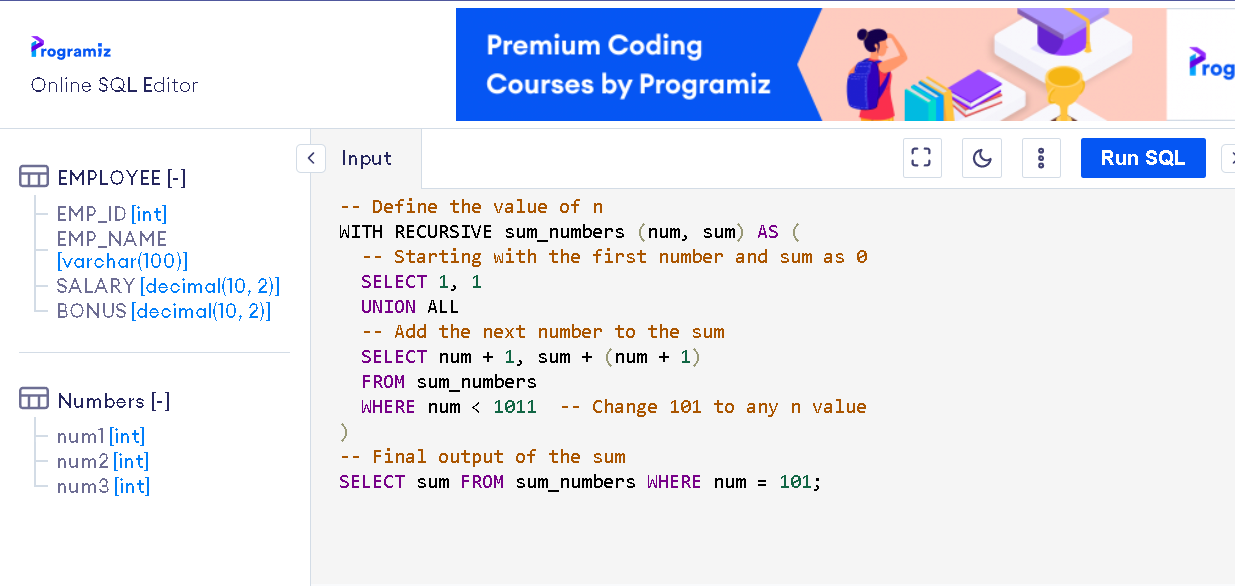
****

**OUTPUT:**

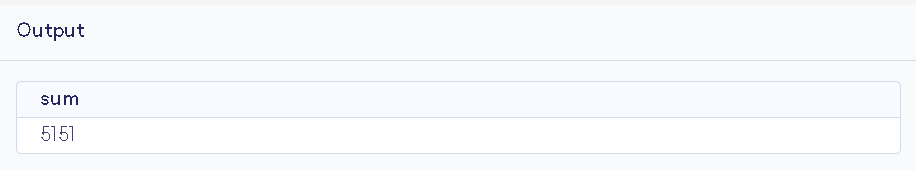
****

**EXERCISE 19**

**AIM: To Write a PL/SQL code to find the sum of n numbers.**

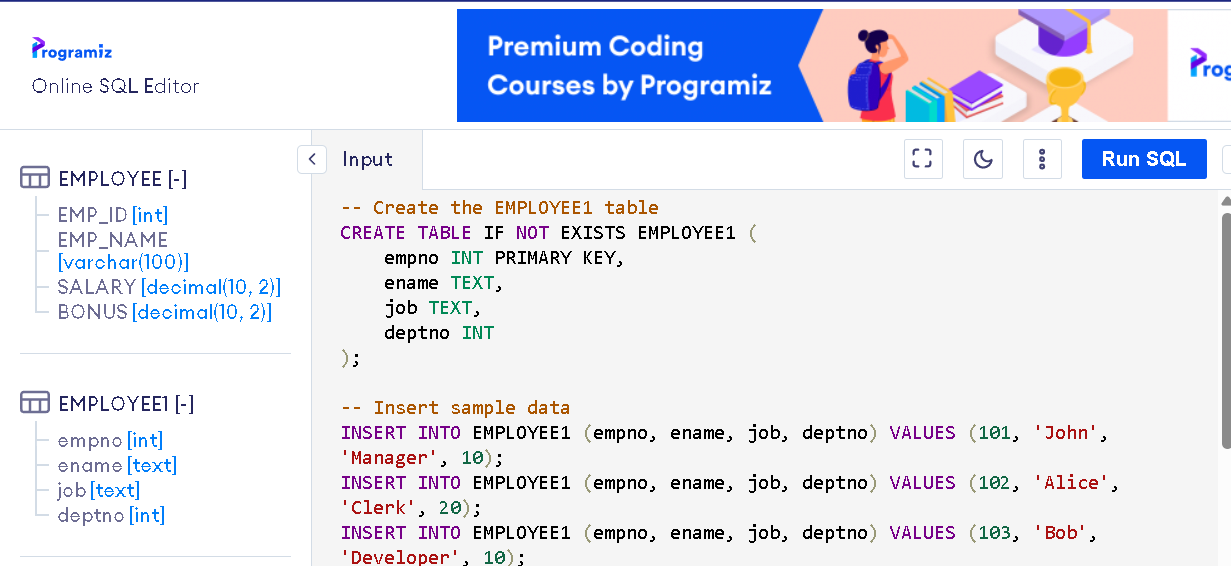
****

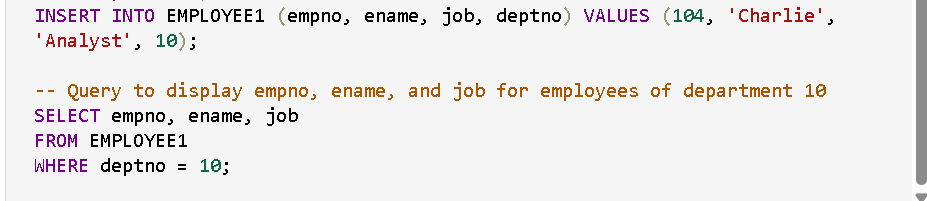
**OUTPUT:**

****

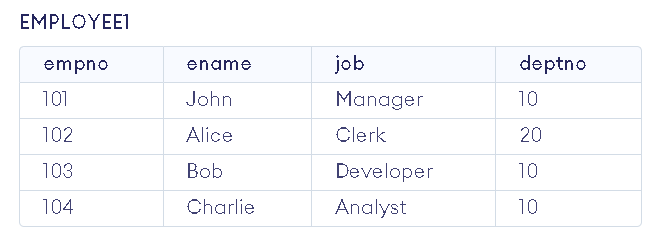
**EXERCISE 20**

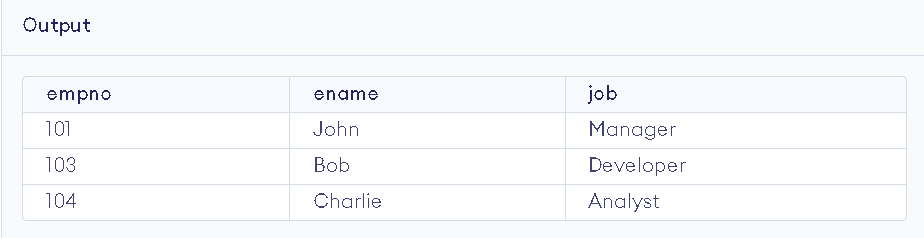
**AIM: To Consider a PL/SQL code to display the empno, ename, job of employees of department number 10.**

****

****

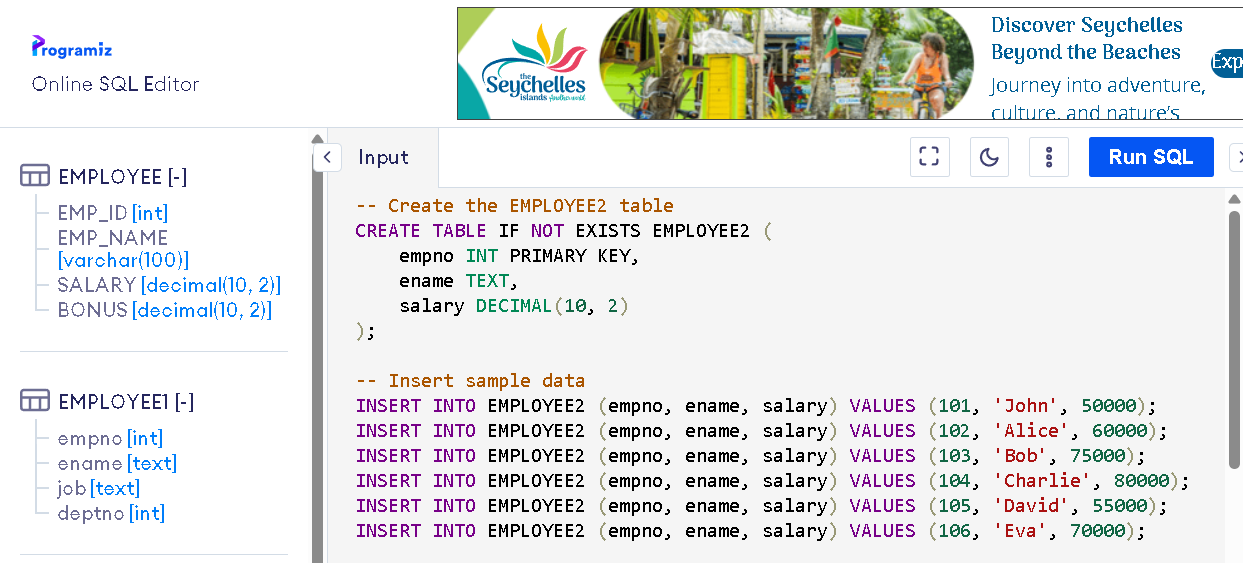
**OUTPUT:**

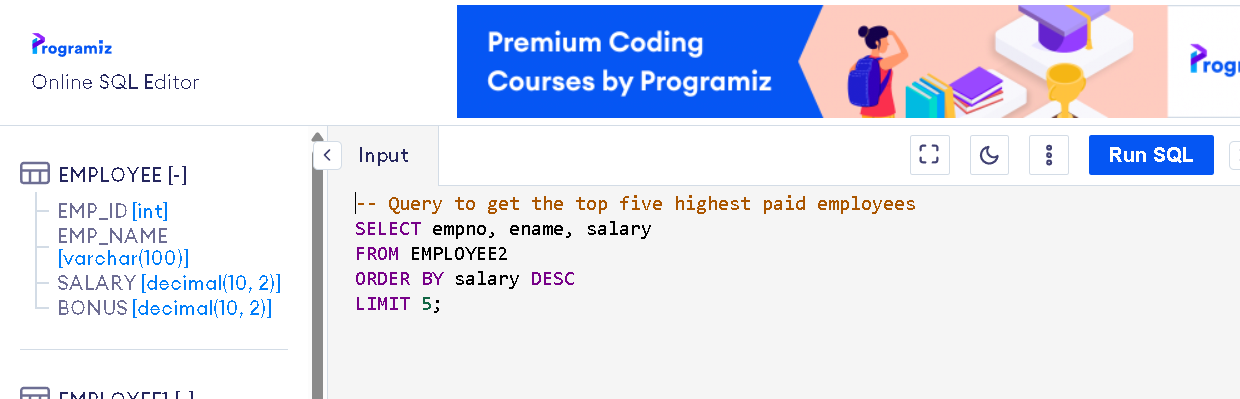
****

****

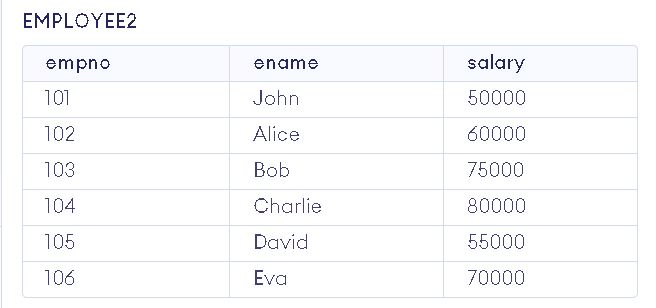
**EXERCISE 21**

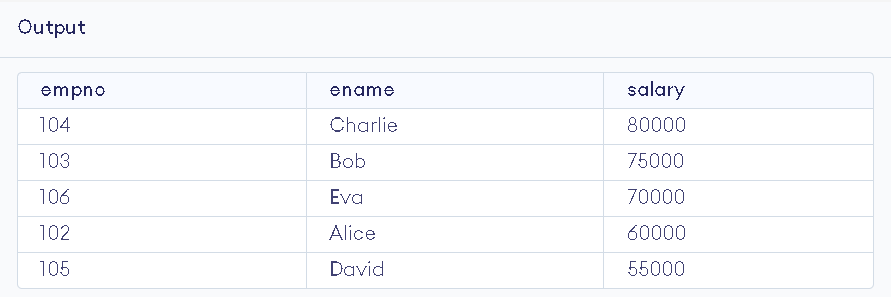
**AIM: To Consider a PL/SQL code to display the employee number & name of top five highest paid employees.**

****

****

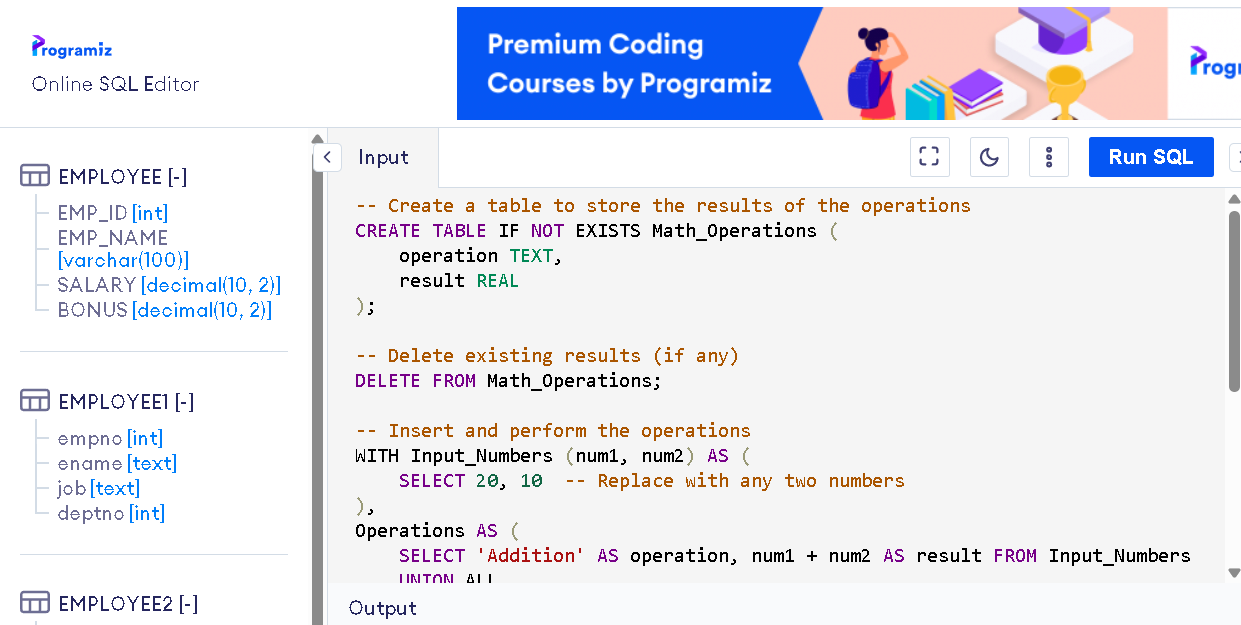
**OUTPUT:**

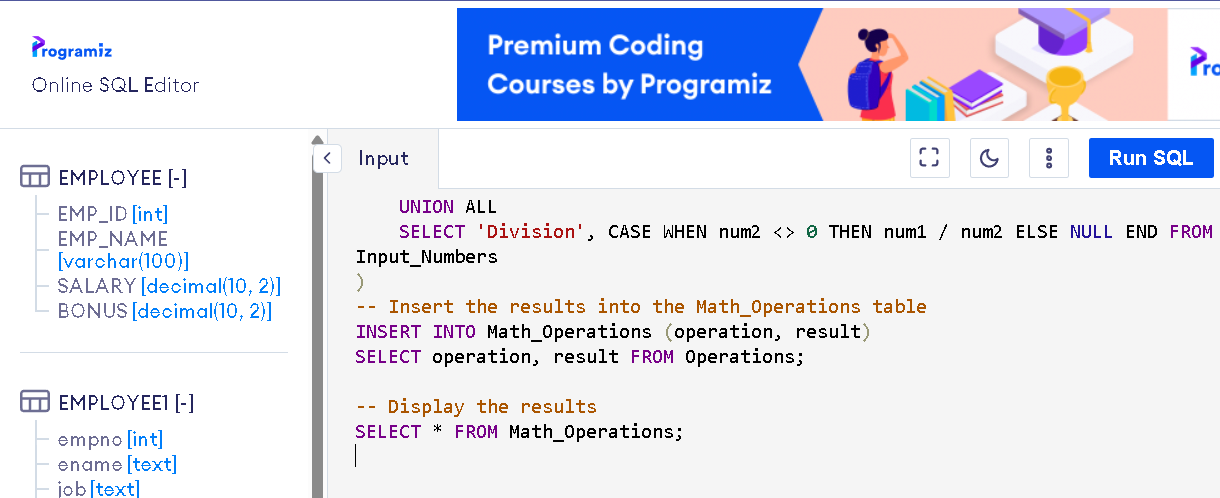
****

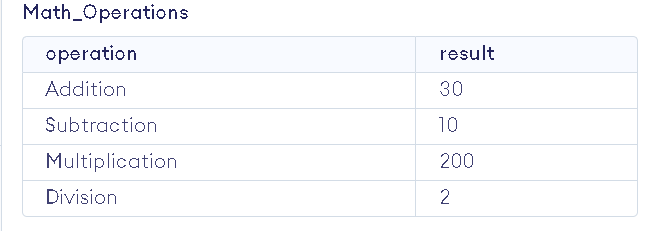
****

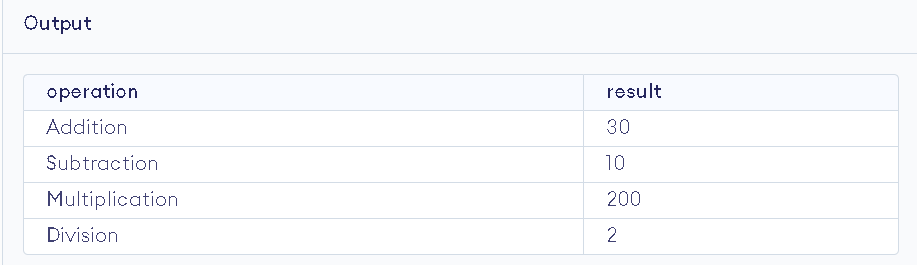
**EXERCISE 22**

**AIM: To Consider a PL/SQL procedure that accepts 2 numbers & return addition, subtraction, multiplication & division of two numbers using stored procedure AND local procedure.**

****

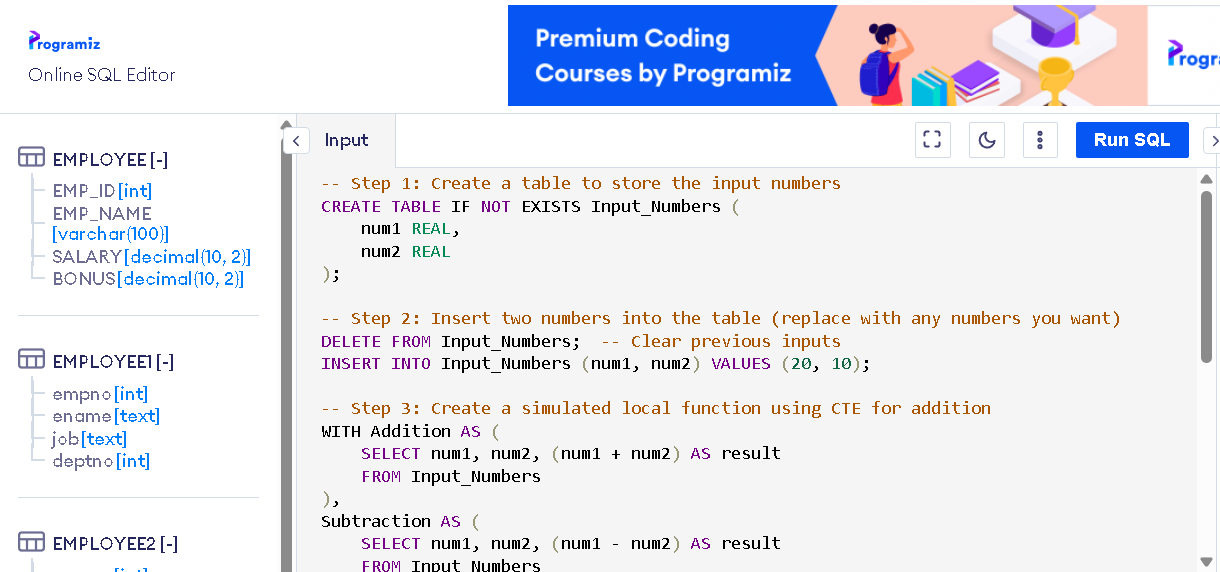
****

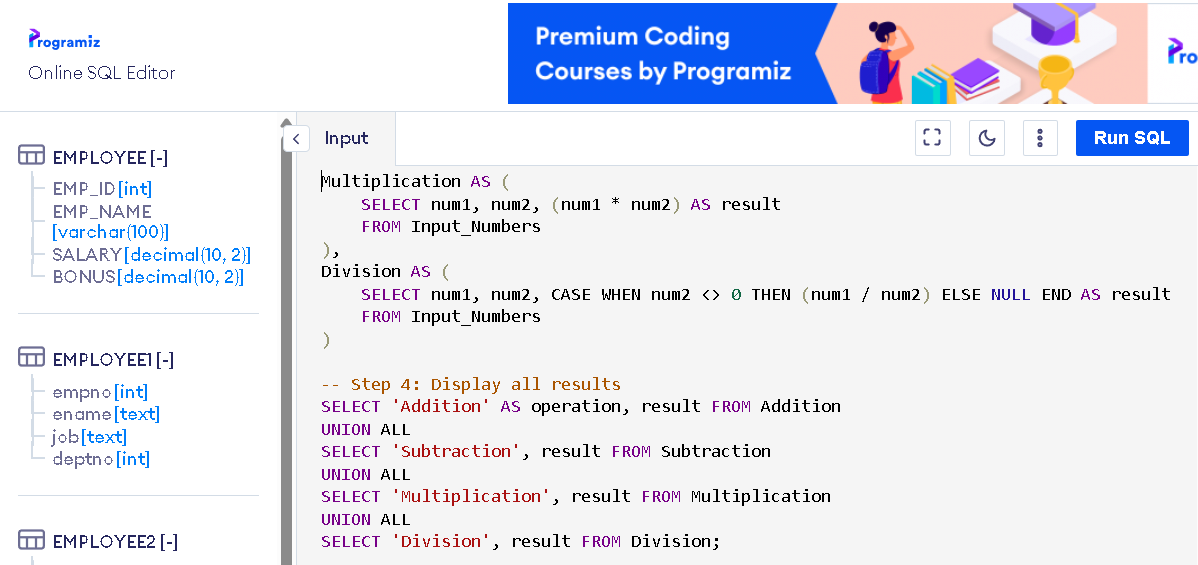
**OUTPUT:  
**

****

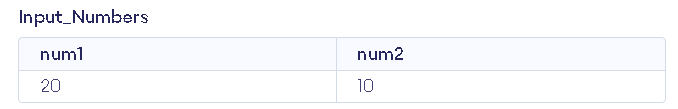
**EXERCISE 23**

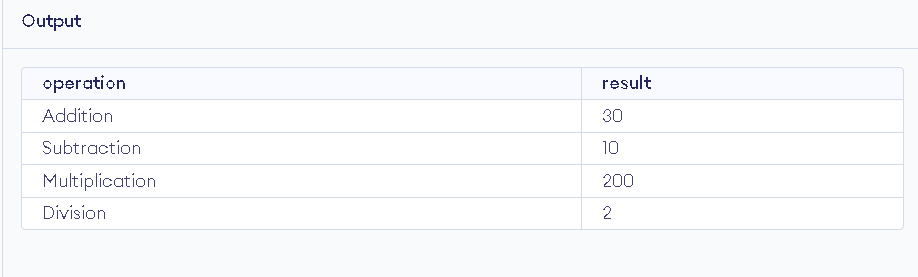
**AIM: To Consider a PL/SQL code that accepts 2 numbers & return addition, subtraction, multiplication & division of two numbers using stored functions and local function.**

****

****

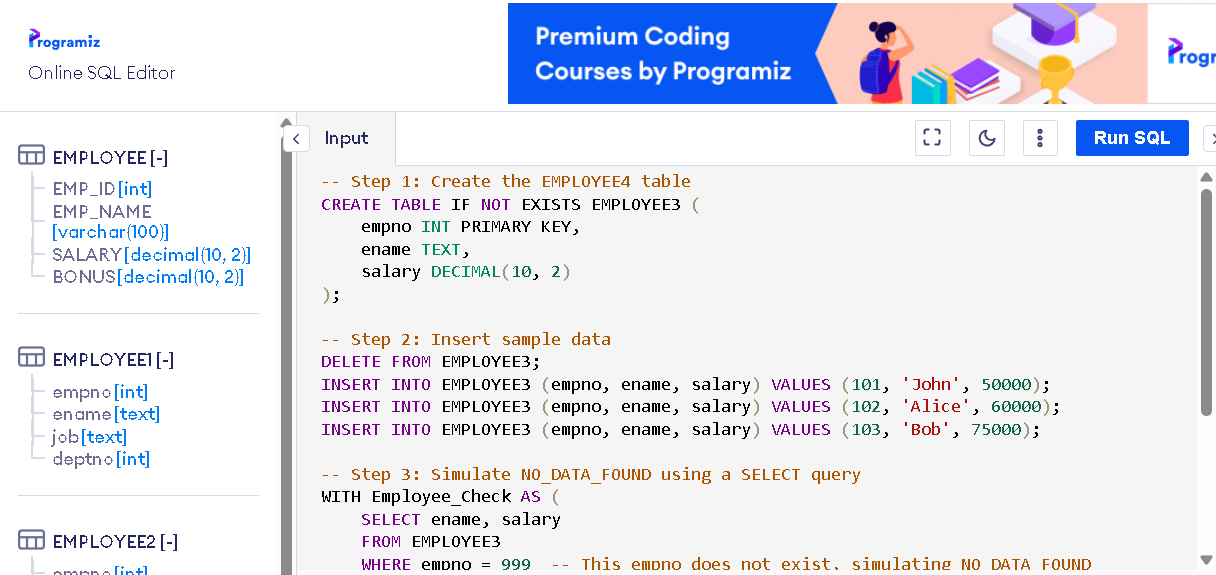
**OUTPUT:**

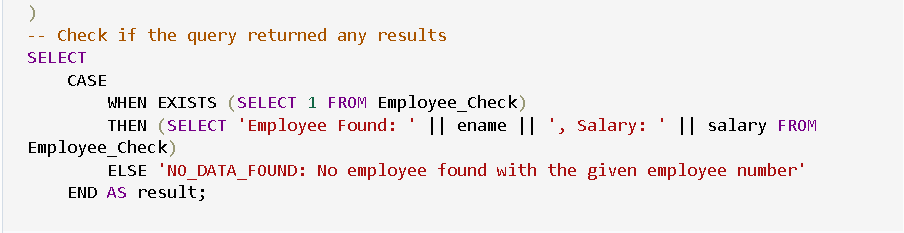
****

****

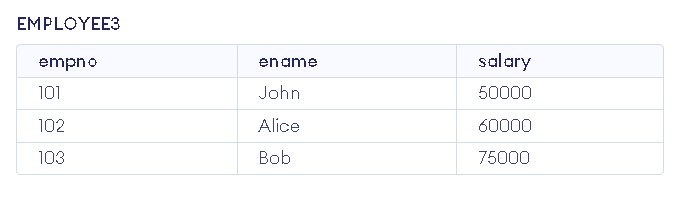
**EXERCISE 24**

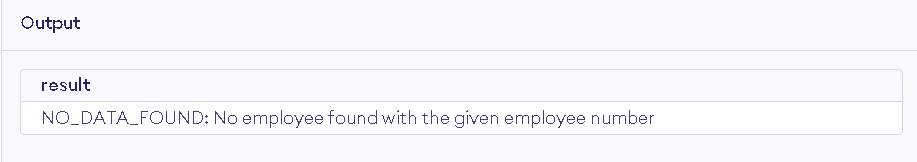
**AIM: To Write a PL/SQL block to show the use of NO\_DATA FOUND exception.**

****

****

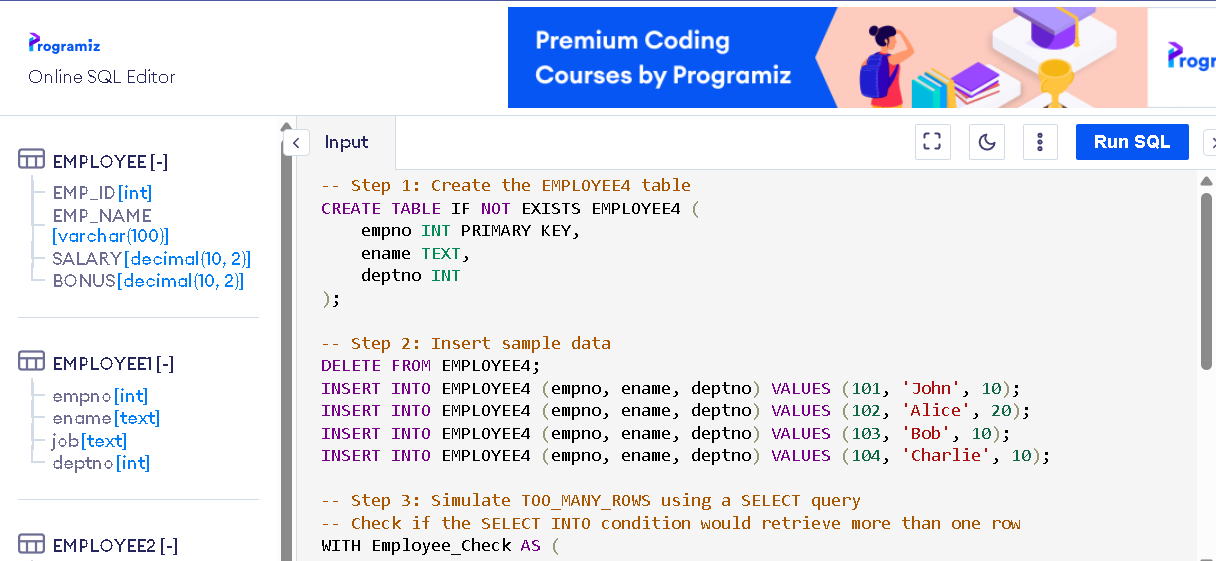
**OUTPUT:**

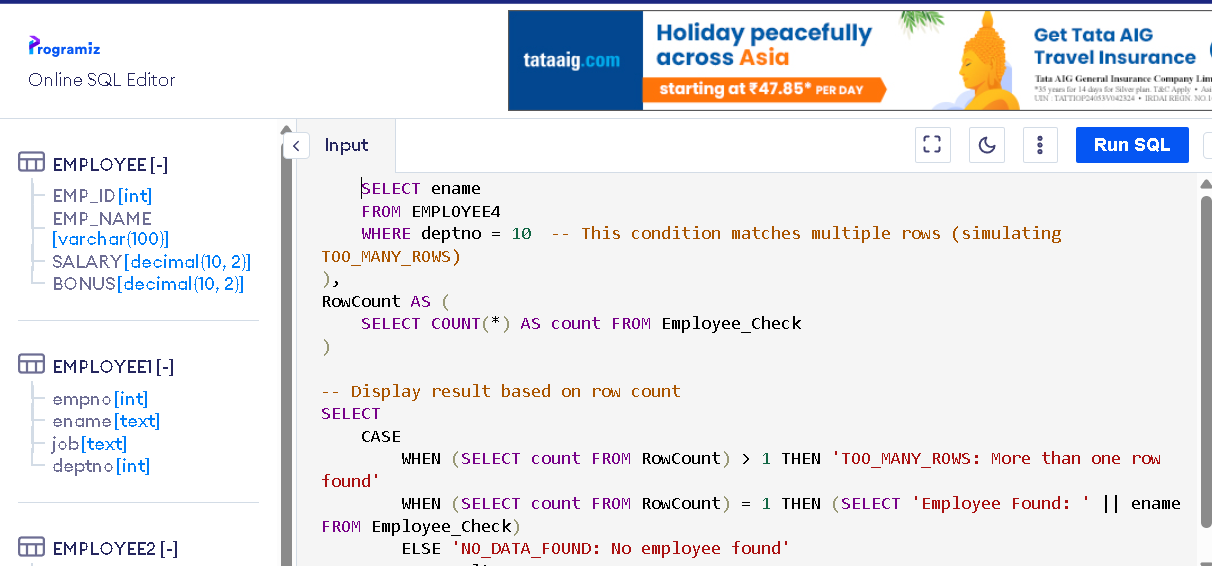
****

****

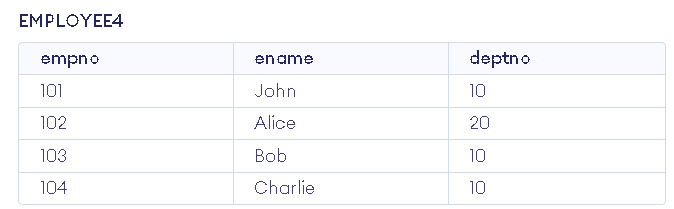
**EXERCISE 25**

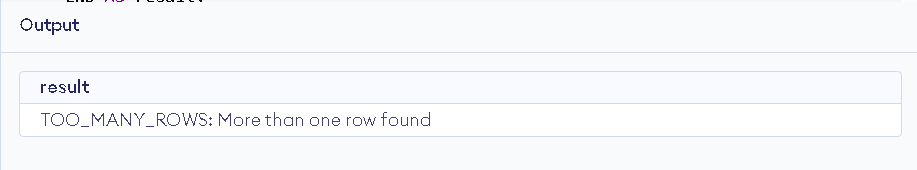
**AIM: To Write a PL/SQL block to show the use of TOO\_MANY ROWS exception.**

****

****

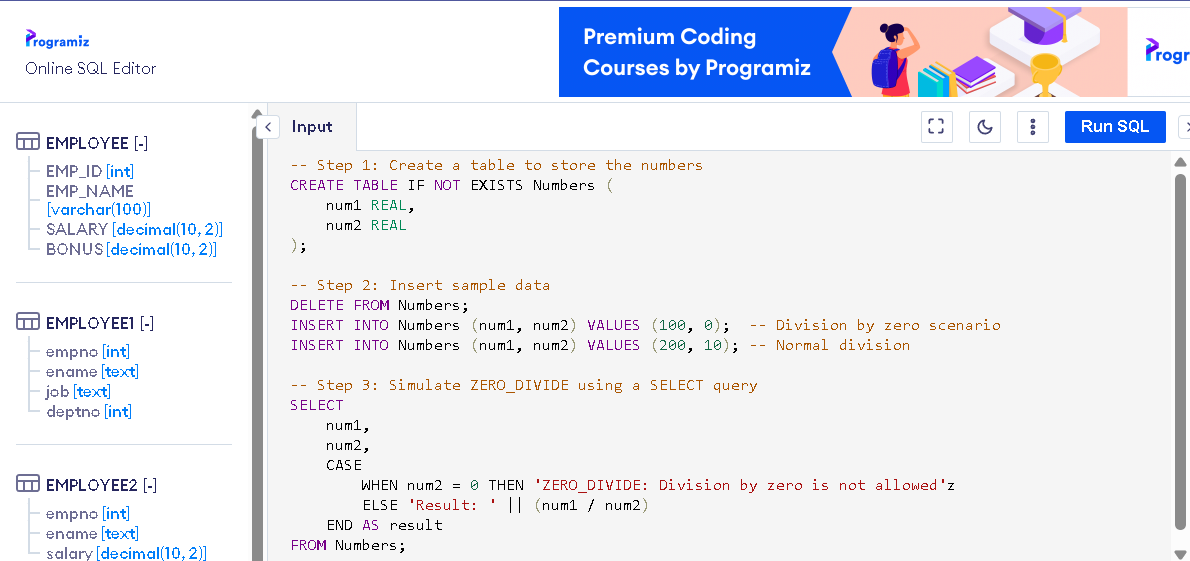
**OUTPUT:**

****

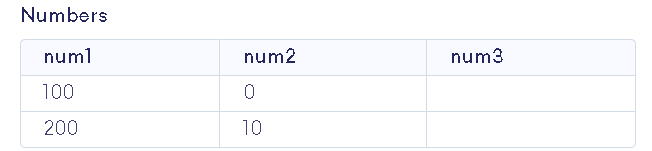
****

**EXERCISE 26**

**AIM: To Write a PL/SQL block to show the use of ZERO\_DIVIDE exception.**

****

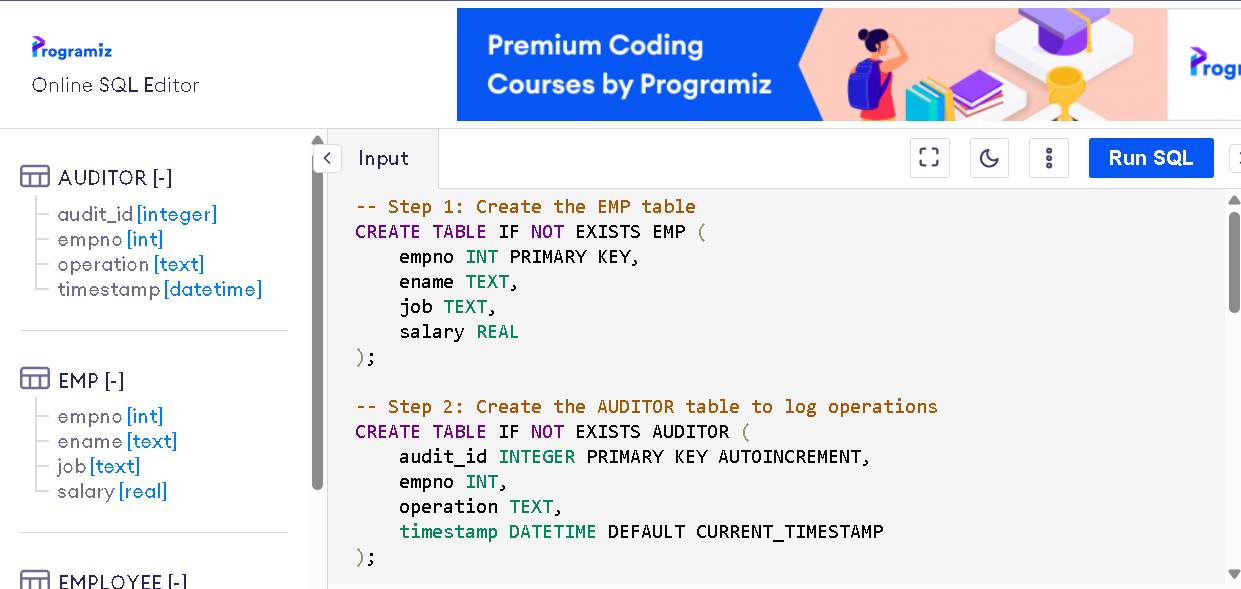
**OUTPUT:**

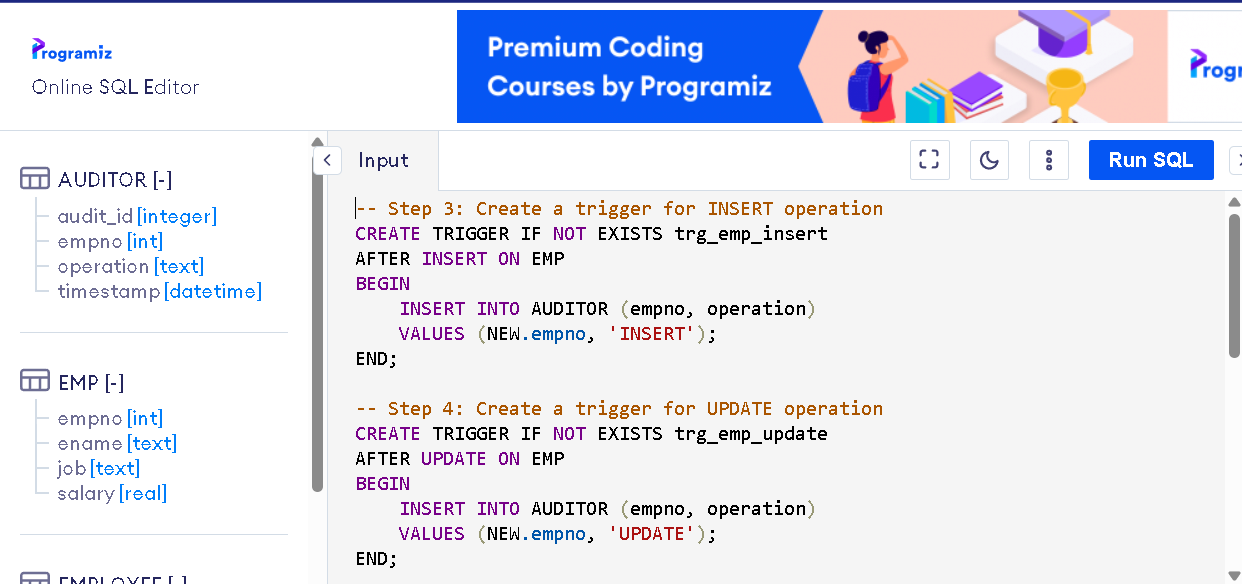
****

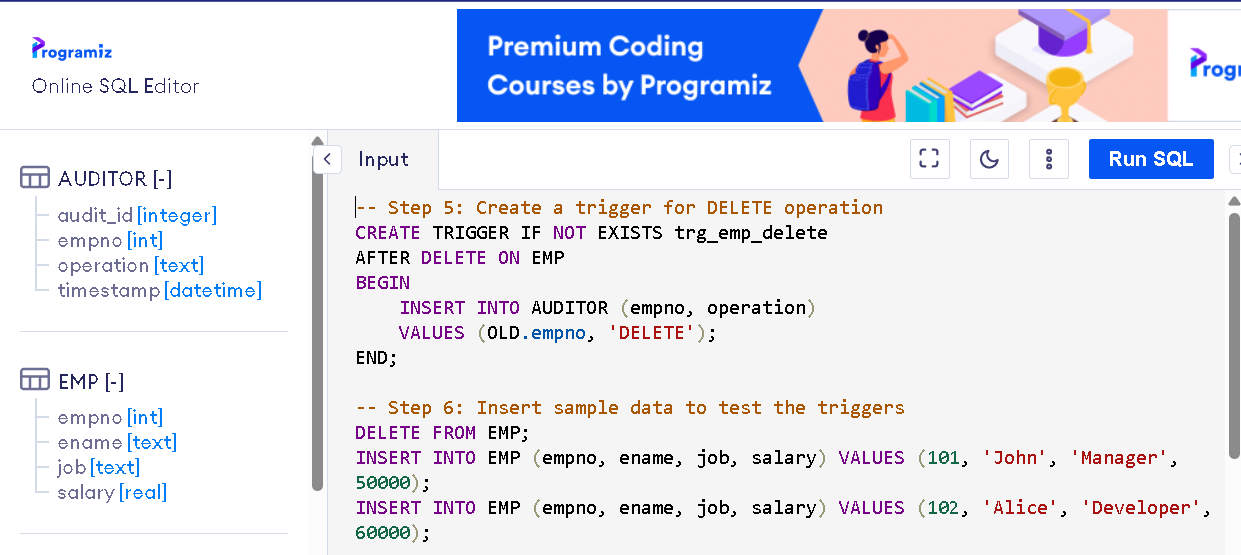
****

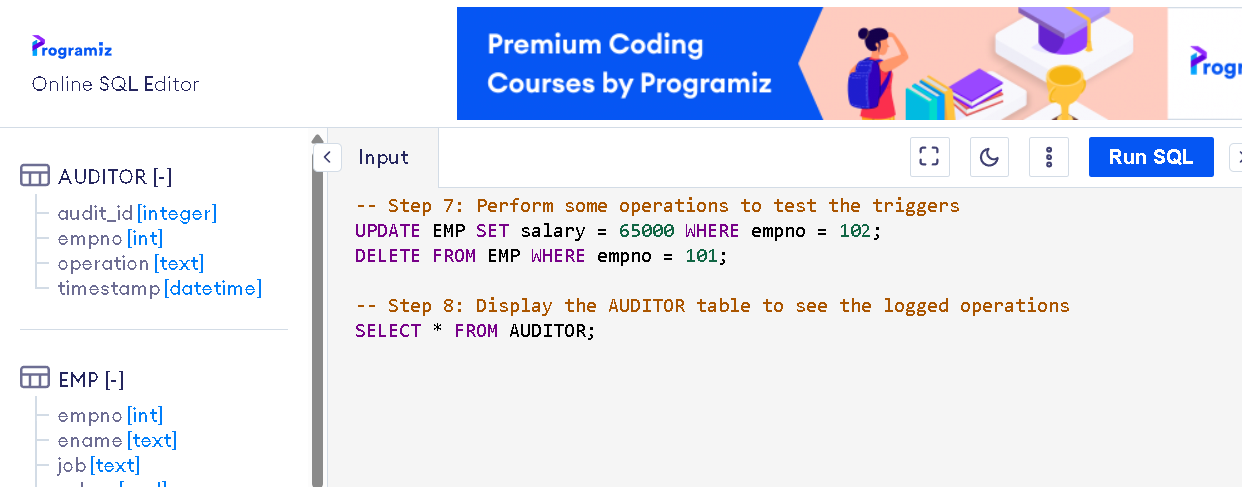
**EXERCISE 27**

**AIM: To create a trigger on the emp table, which store the empno& operation in the table auditor for each operation i.e. Insert, Update & Delete.**

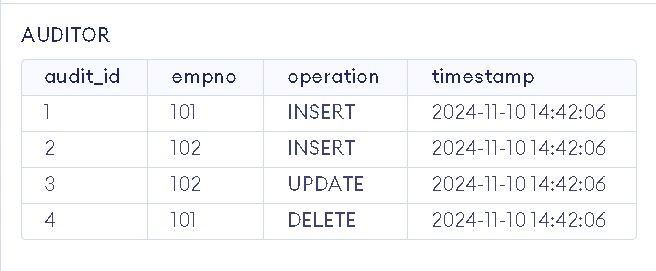
****

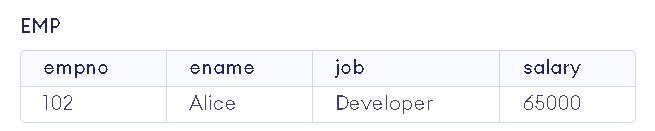
****

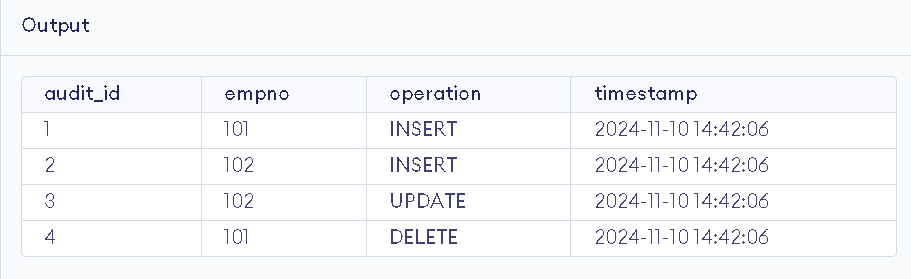
****

****

**OUTPUT:**

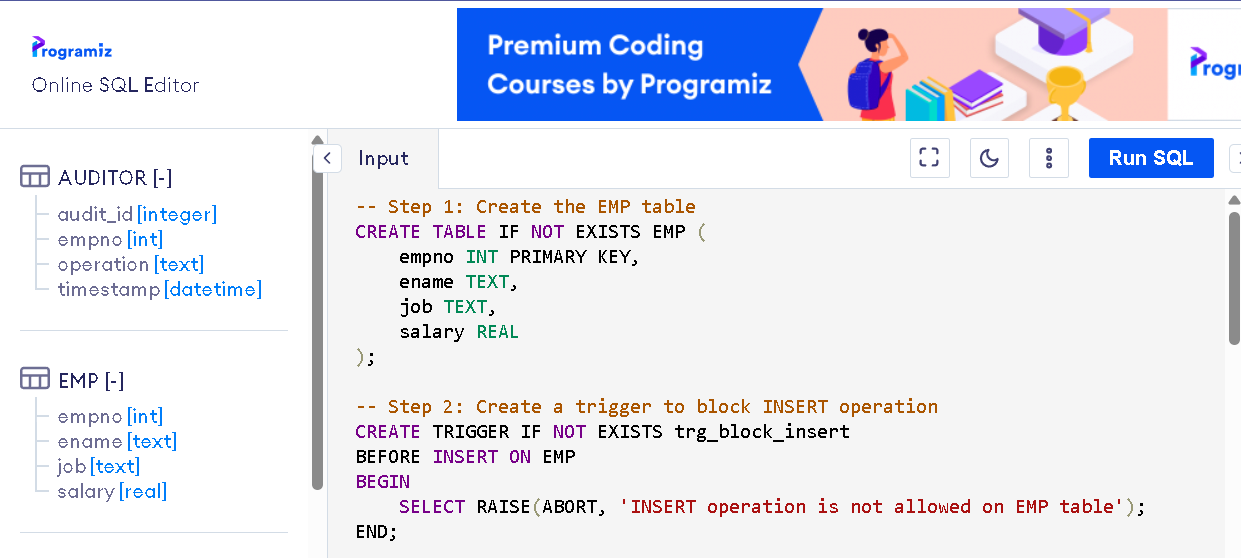
****

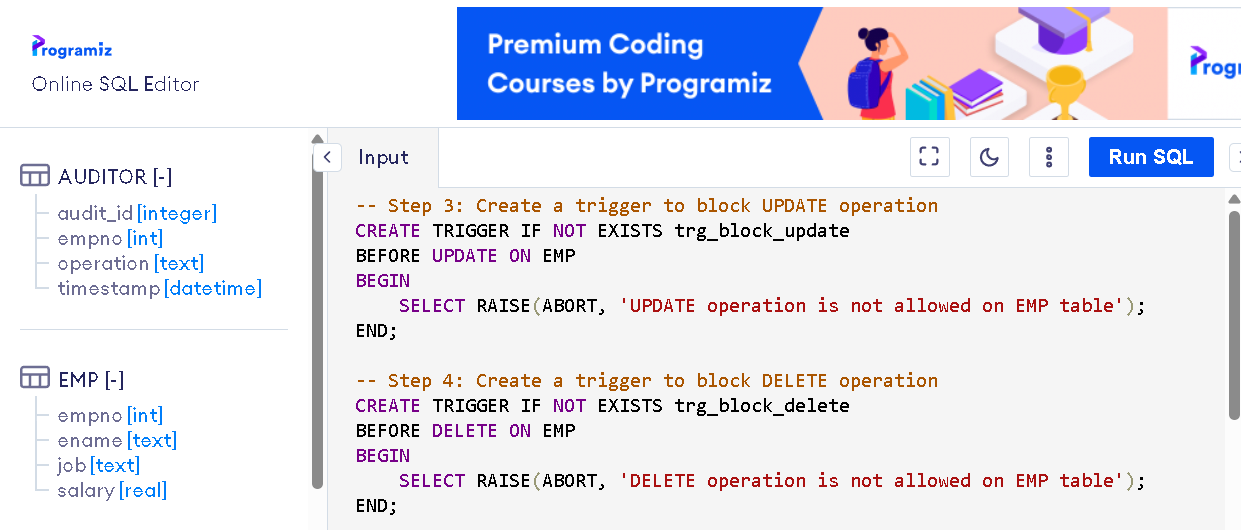
****

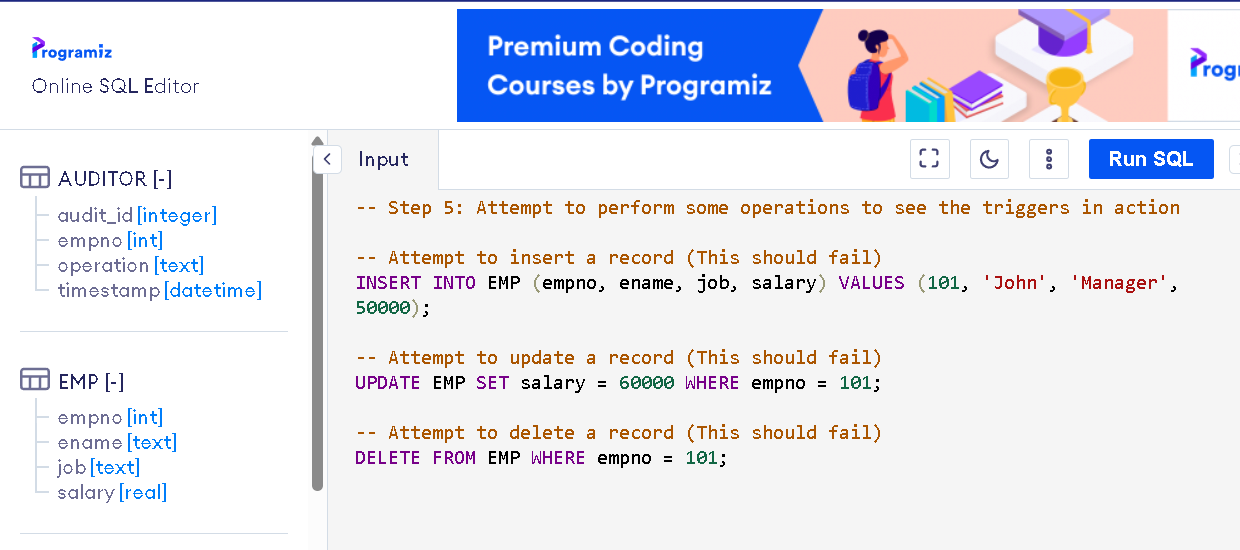
****

**EXERCISE 28**

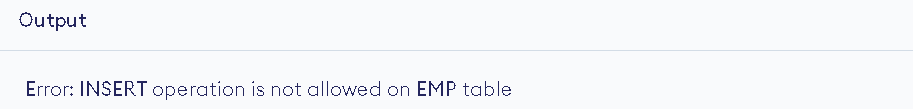
**AIM: To create a trigger so that no operation can be performed on emp table.**

****

****

****

**OUTPUT:**

****