**Student Name:** Jayaanth S R

**SuperSet Id :** 6420871

**EXERCISE 1: CONTROL STRUCTURES**

**Introduction:**

This PL/SQL script automates customer-based operations, including applying loan interest discounts and promoting customers to premium.

**Objective:**

* **Apply Interest Discounts:**  Identify customers who have made more than 5 orders and apply a 10% discount to them.
* **Promote Premium Status:** Identify customers whose total order value exceeds ₹50,000 and update their status to Premium.

**Implementation Breakdown:**

**Scenario 1:**

BEGIN

    FOR cus IN (

        SELECT c.CustomerID, c.Name, COUNT(o.OrderID) AS OrderCount

        FROM Customers c

        JOIN Orders o ON c.CustomerID = o.CustomerID

        GROUP BY c.CustomerID, c.Name

    ) LOOP

        IF cus.OrderCount > 5 THEN

            UPDATE Customers

            SET Discount = 10

            WHERE CustomerID = cus.CustomerID;

            DBMS\_OUTPUT.PUT\_LINE('10% discount applied to customer ' || cus.Name);

        END IF;

    END LOOP;

   END;

**Scenario 2:**

BEGIN

    FOR cus IN (

        SELECT c.CustomerID, c.Name, SUM(o.TotalAmount) AS TotalSpent, c.IsPremium

        FROM Customers c

        JOIN Orders o ON c.CustomerID = o.CustomerID

        GROUP BY c.CustomerID, c.Name, c.IsPremium

    ) LOOP

        IF cus.TotalSpent > 50000 AND cus.IsPremium = 'N' THEN

            UPDATE Customers

            SET IsPremium = 'Y'

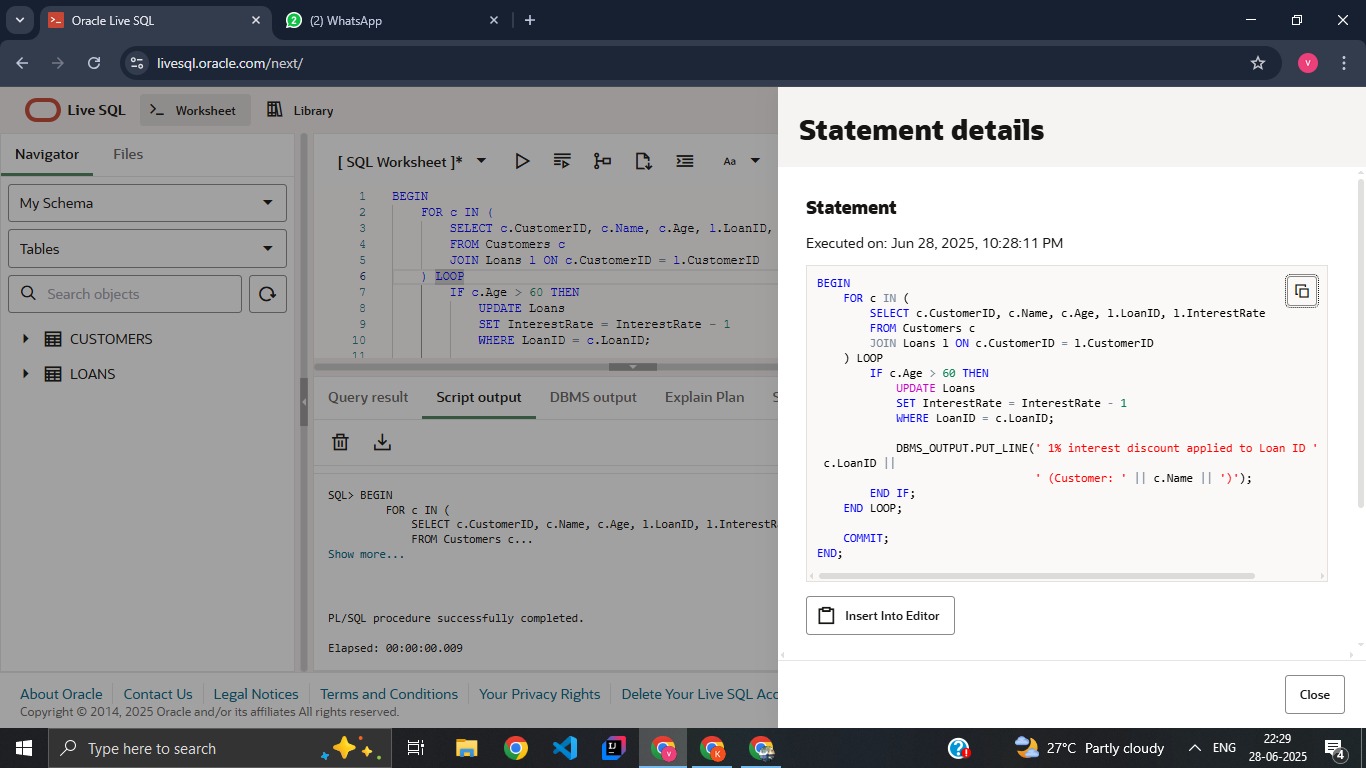
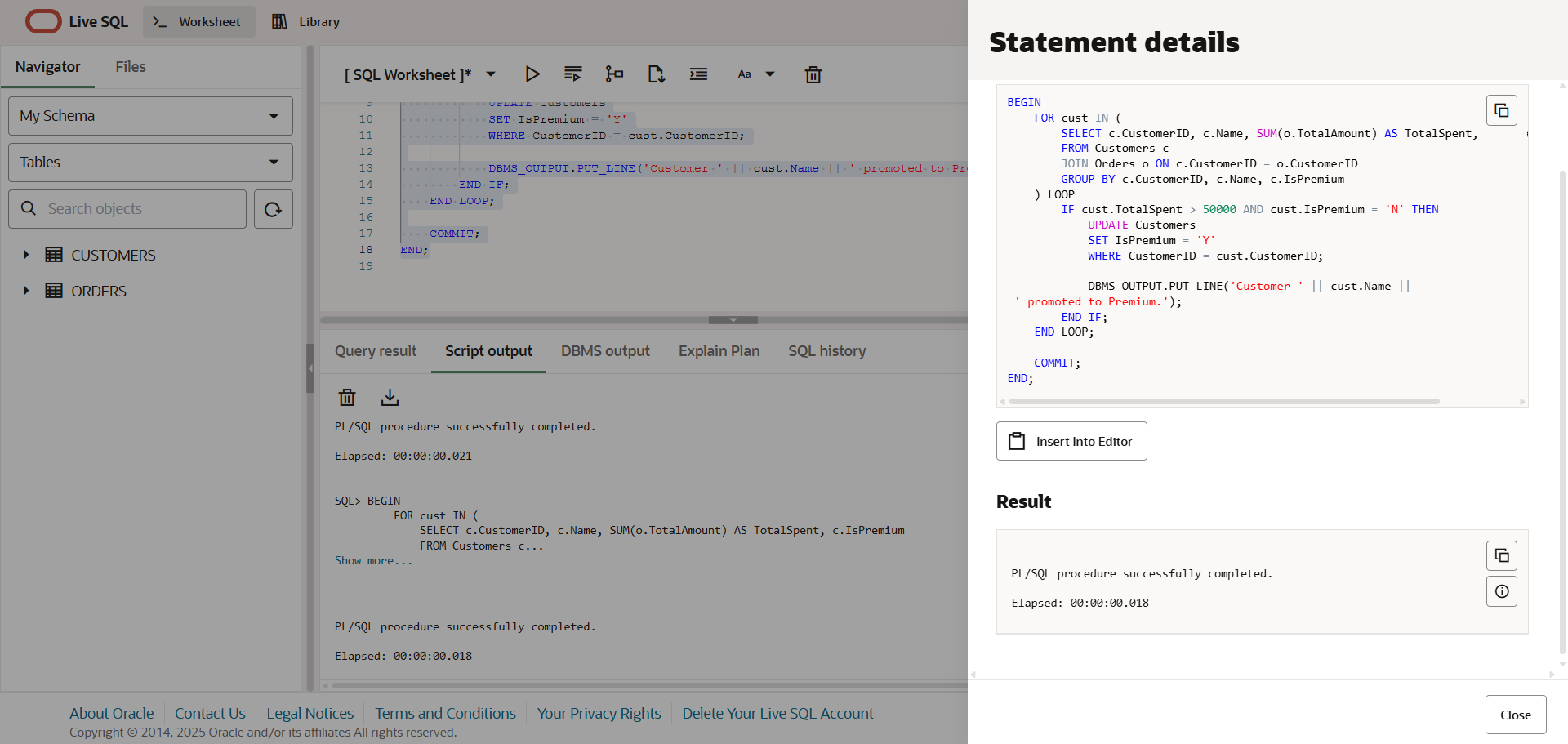
            WHERE CustomerID = cus.CustomerID;

            DBMS\_OUTPUT.PUT\_LINE('Customer ' || cus.Name || ' promoted to Premium.');

        END IF;

    END LOOP;

END;

**Output:** ****

**Conclusion:**

This PL/SQL block successfully automates the application of a 1% interest rate discount for customers above 60 years of age. It enhances efficiency by eliminating the need for manual updates and ensures that only eligible customers are affected. Logging with DBMS\_OUTPUT also improves traceability, making this a simple yet effective real-world database solution.

**EXERCISE 3: STORED STRUCTURES**

**Introduction:**

This PL/SQL program defines procedures to automate key e-commerce operations such as applying loyalty credit to premium customers, promoting customers based on their total purchase value, and securely transferring wallet balances between customer accounts.

**Objective:**

* **Apply Loyalty Credit:** Calculate and update 5% of total purchases as loyalty credit for Premium customers.
* **Promote to Premium:** Promote customers to Premium if their total purchase value exceeds ₹50,000.
* **Transfer Wallet Balance Securely:** Ensure safe transfer of wallet balance between two customers with sufficient balance checks.

**Implementation Breakdown:**

**Scenario 1:**

CREATE OR REPLACE PROCEDURE AddLoyaltyCredit AS

BEGIN

FOR cust IN (

SELECT c.CustomerID, SUM(o.TotalAmount) AS TotalSpent

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

WHERE c.IsPremium = 'Y'

GROUP BY c.CustomerID

) LOOP

UPDATE Customers

SET LoyaltyCredit = NVL(LoyaltyCredit, 0) + (cust.TotalSpent \* 0.05)

WHERE CustomerID = cust.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('5% loyalty credit added to Premium customer ' || cust.CustomerID);

END LOOP;

END;

**Scenario 2:**

CREATE OR REPLACE PROCEDURE PromotePremiumCustomers AS

BEGIN

FOR cust IN (

SELECT c.CustomerID, SUM(o.TotalAmount) AS TotalSpent, c.IsPremium

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.IsPremium

) LOOP

IF cust.TotalSpent > 50000 AND cust.IsPremium = 'N' THEN

UPDATE Customers

SET IsPremium = 'Y'

WHERE CustomerID = cust.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Customer ' || cust.CustomerID || ' promoted to Premium.');

END IF;

END LOOP;

END;

**Scenario 3:**

CREATE OR REPLACE PROCEDURE TransferWalletBalance (

p\_from\_customer IN NUMBER,

p\_to\_customer IN NUMBER,

p\_amount IN NUMBER

) AS

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance

FROM Customers

WHERE CustomerID = p\_from\_customer

FOR UPDATE;

IF v\_balance >= p\_amount THEN

UPDATE Customers

SET Balance = Balance - p\_amount

WHERE CustomerID = p\_from\_customer;

UPDATE Customers

SET Balance = Balance + p\_amount

WHERE CustomerID = p\_to\_customer;

DBMS\_OUTPUT.PUT\_LINE('Transfer of ₹' || p\_amount ||

' from Customer ' || p\_from\_customer ||

' to Customer ' || p\_to\_customer || ' successful.');

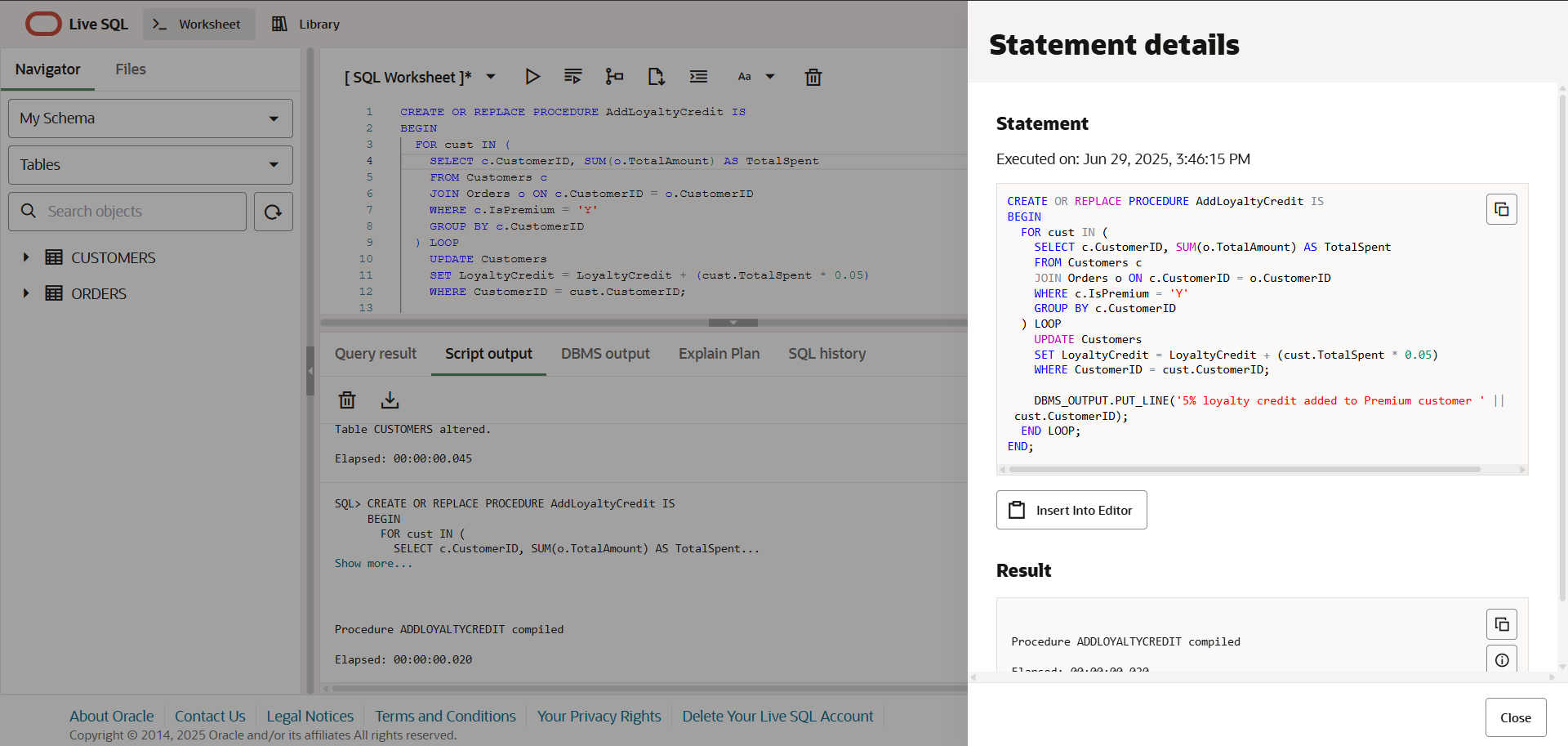
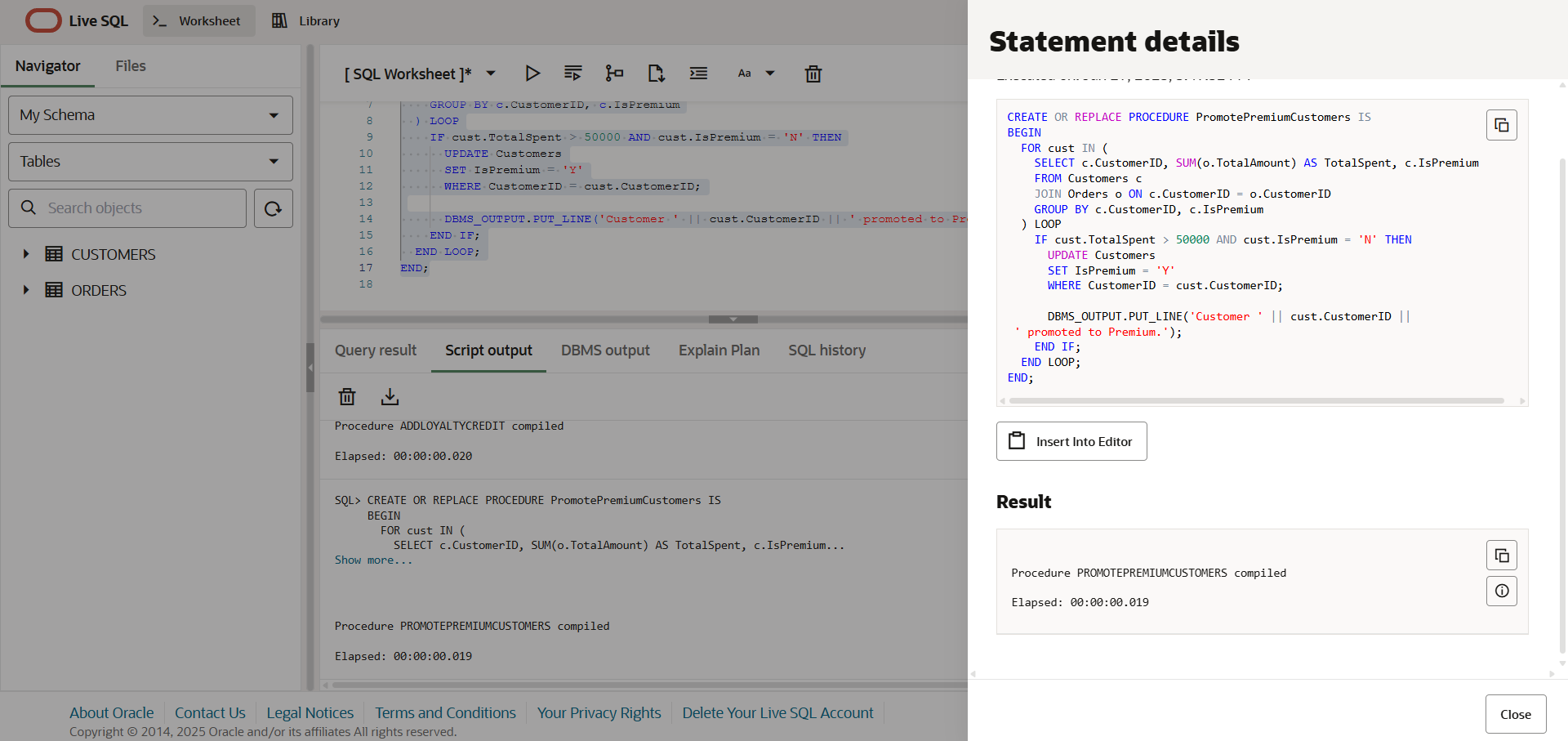
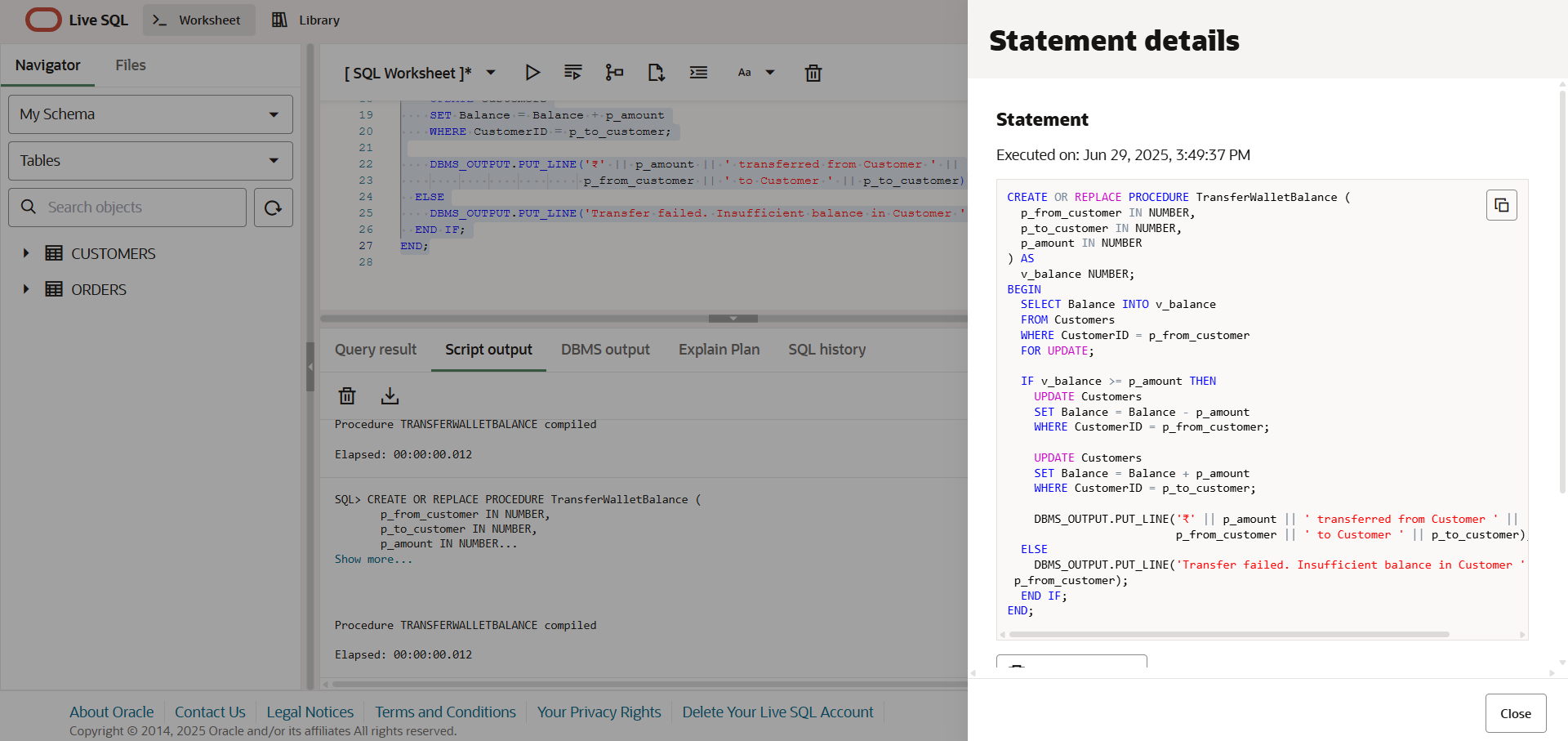
ELSE

DBMS\_OUTPUT.PUT\_LINE('Transfer failed. Insufficient balance in Customer ' || p\_from\_customer);

END IF;

END;

**Output:**

**** **** ****

**Conclusion:**

The procedures execute essential financial tasks efficiently, ensuring accurate balance updates, fair employee compensation, and secure fund transfers, thereby enhancing overall system reliability and user satisfaction.