**Triggers**

**Scenario 1:**

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

  :NEW.LastModified := SYSDATE;

END;

/

UPDATE Customers

SET Balance = Balance + 500

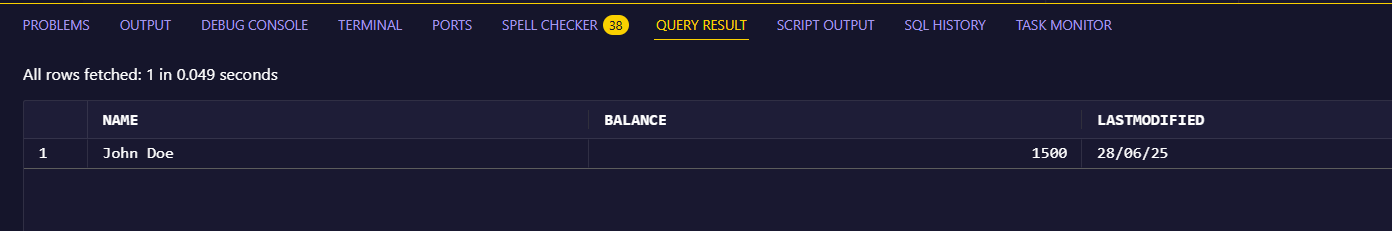
WHERE CustomerID = 1;

SELECT Name, Balance, LastModified

FROM Customers

WHERE CustomerID = 1;

**Output:**

****

**Scenario 2:**

CREATE TABLE AuditLog (

  LogID       NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,

  TransactionID NUMBER,

  ActionDate  DATE,

  Description VARCHAR2(100)

);

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

  INSERT INTO AuditLog (TransactionID, ActionDate, Description)

  VALUES (:NEW.TransactionID, SYSDATE, 'Transaction inserted');

END;

/

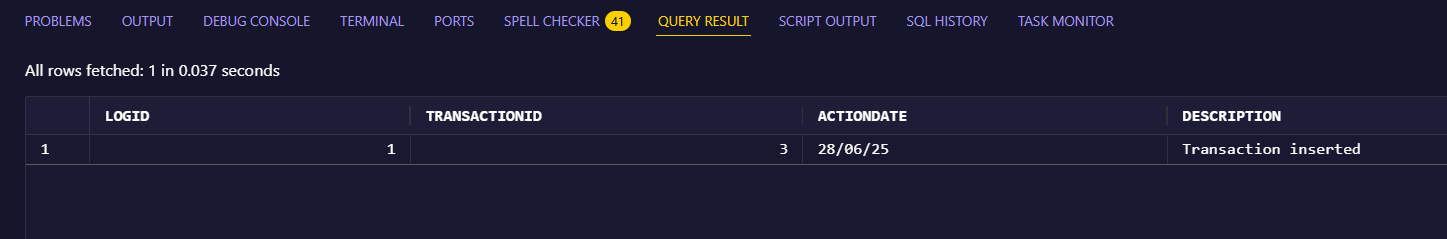
INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (3, 1, SYSDATE, 1000, 'Deposit');

SELECT \* FROM AuditLog

ORDER BY LogID DESC;

**Output:**

****

**Scenario 3:**

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON Transactions

FOR EACH ROW

DECLARE

  acc\_balance NUMBER;

BEGIN

  SELECT Balance INTO acc\_balance

  FROM Accounts

  WHERE AccountID = :NEW.AccountID;

  IF :NEW.TransactionType = 'Withdrawal' AND :NEW.Amount > acc\_balance THEN

    RAISE\_APPLICATION\_ERROR(-20001, 'Withdrawal exceeds current balance.');

  ELSIF :NEW.TransactionType = 'Deposit' AND :NEW.Amount <= 0 THEN

    RAISE\_APPLICATION\_ERROR(-20002, 'Deposit amount must be positive.');

  END IF;

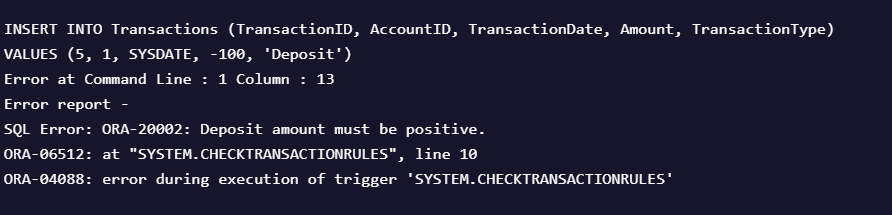
END;

/

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (5, 1, SYSDATE, -100, 'Deposit');

**Output:**

****