PL/SQL

1. Control Structures:

CREATE TABLE customers (

  customer\_id NUMBER,

  name VARCHAR2(50),

  age NUMBER,

  balance NUMBER,

  interest\_rate NUMBER,

  IsVIP VARCHAR2(5)

);

CREATE TABLE loans (

  loan\_id NUMBER,

  customer\_id NUMBER,

  due\_date DATE

);

INSERT INTO customers VALUES (101, 'Ravi', 65, 12000, 7.5, 'FALSE');

INSERT INTO customers VALUES (102, 'Asha', 45, 8000, 6.0, 'FALSE');

INSERT INTO customers VALUES (103, 'Meena', 68, 15000, 6.8, 'FALSE');

COMMIT;

INSERT INTO loans VALUES (1, 101, SYSDATE + 15);  -- Due in 15 days

INSERT INTO loans VALUES (2, 102, SYSDATE + 40);  -- Due in 40 days

INSERT INTO loans VALUES (3, 103, SYSDATE + 5);   -- Due in 5 days

COMMIT;

SELECT \* FROM customers;

SELECT \* FROM loans;

BEGIN

  1. Discount for seniors

  FOR cust IN (SELECT customer\_id, interest\_rate FROM customers WHERE age > 60) LOOP

    UPDATE customers

    SET interest\_rate = cust.interest\_rate - 1

    WHERE customer\_id = cust.customer\_id;

  END LOOP;

  2. Promote VIPs

  FOR vip IN (SELECT customer\_id FROM customers WHERE balance > 10000) LOOP

    UPDATE customers

    SET IsVIP = 'TRUE'

    WHERE customer\_id = vip.customer\_id;

  END LOOP;

  3. Print reminders

  FOR loan IN (

    SELECT customer\_id, due\_date FROM loans

    WHERE due\_date BETWEEN SYSDATE AND SYSDATE + 30

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE(

      'Reminder: Customer ID ' || loan.customer\_id ||

      ', your loan is due on ' || TO\_CHAR(loan.due\_date, 'DD-MON-YYYY')

    );

  END LOOP;

  COMMIT;

END;

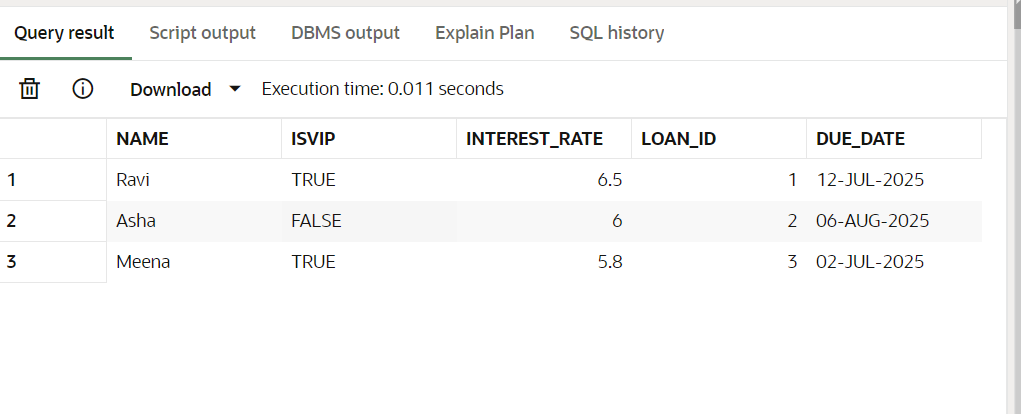
SELECT name, interest\_rate, IsVIP FROM customers;

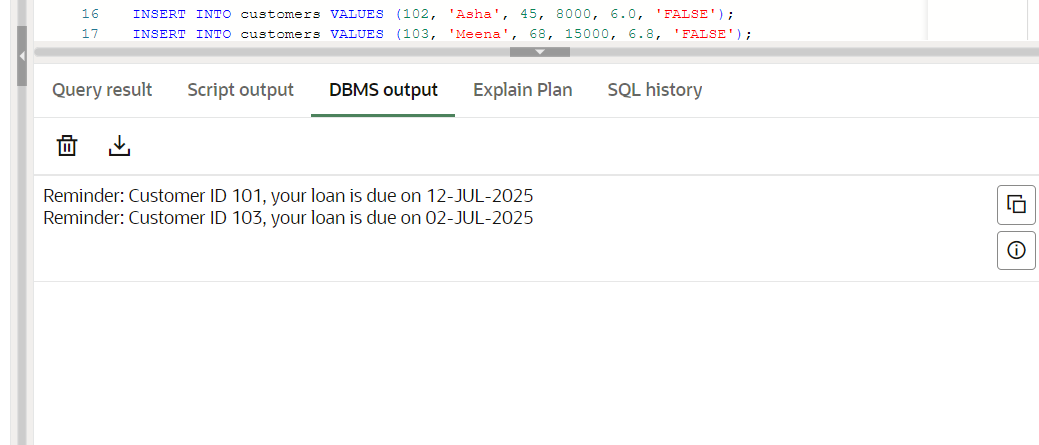
SELECT c.name, c.IsVIP, c.interest\_rate, l.loan\_id, TO\_CHAR(l.due\_date, 'DD-MON-YYYY') AS due\_date

FROM customers c

JOIN loans l ON c.customer\_id = l.customer\_id;

OUTPUT:





1. STORED PROCEDURE:

CREATE TABLE savings\_accounts (

  account\_id NUMBER,

  customer\_name VARCHAR2(50),

  balance NUMBER

);

INSERT INTO savings\_accounts VALUES (4001, 'Ravi', 12000);

INSERT INTO savings\_accounts VALUES (4002, 'Meena', 18000);

INSERT INTO savings\_accounts VALUES (4003, 'Keerthana', 9500);

COMMIT;

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

  UPDATE savings\_accounts

  SET balance = balance + (balance \* 0.01);

  COMMIT;

END;

BEGIN

  ProcessMonthlyInterest;

END;

CREATE TABLE employees (

  employee\_id NUMBER,

  department\_id NUMBER,

  name VARCHAR2(50),

  salary NUMBER

);

INSERT INTO employees VALUES (201, 10, 'Anil', 40000);

INSERT INTO employees VALUES (202, 10, 'Kavya', 45000);

INSERT INTO employees VALUES (203, 20, 'Rahul', 50000);

COMMIT;

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

  p\_department\_id IN NUMBER,

  p\_bonus\_percent IN NUMBER

) AS

BEGIN

  UPDATE employees

  SET salary = salary + (salary \* p\_bonus\_percent / 100)

  WHERE department\_id = p\_department\_id;

  COMMIT;

END;

BEGIN

  UpdateEmployeeBonus(10, 10);  -- 10% bonus to Dept 10

END;

CREATE TABLE accounts (

  account\_id NUMBER,

  customer\_name VARCHAR2(50),

  balance NUMBER

);

INSERT INTO accounts VALUES (3001, 'Keerthana', 10000);

INSERT INTO accounts VALUES (3002, 'Asha', 7000);

COMMIT;

CREATE OR REPLACE PROCEDURE TransferFunds (

  p\_from\_acct IN NUMBER,

  p\_to\_acct IN NUMBER,

  p\_amount IN NUMBER

) AS

  v\_balance NUMBER;

BEGIN

  SELECT balance INTO v\_balance

  FROM accounts

  WHERE account\_id = p\_from\_acct;

  IF v\_balance < p\_amount THEN

    RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance for transfer.');

  END IF;

  UPDATE accounts

  SET balance = balance - p\_amount

  WHERE account\_id = p\_from\_acct;

  UPDATE accounts

  SET balance = balance + p\_amount

  WHERE account\_id = p\_to\_acct;

  COMMIT;

END;

BEGIN

  TransferFunds(3001, 3002, 1500);  -- Transfer ₹1500

END;

SELECT \* FROM savings\_accounts;

SELECT \* FROM employees;

SELECT \* FROM accounts;

BEGIN

  DBMS\_OUTPUT.PUT\_LINE('Transfer complete!');

END;

/

OUTPUT:

