**Exercise 1: Control Structures**

--CUSTOMERS TABLE

CREATE TABLE customers (

customer\_id NUMBER PRIMARY KEY,

name VARCHAR2(50),

age NUMBER,

interest\_rate NUMBER(5,2),

balance NUMBER(10,2),

isvip VARCHAR2(5)

);

--LOANS TABLE

CREATE TABLE loans (

loan\_id NUMBER PRIMARY KEY,

customer\_id NUMBER,

due\_date DATE,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)

);

--Customers DATA

INSERT INTO customers VALUES (1, 'Gogulan', 65, 7.5, 15000, 'FALSE');

INSERT INTO customers VALUES (2, 'Arun', 58, 8.0, 9500, 'FALSE');

INSERT INTO customers VALUES (3, 'Sneha', 62, 9.0, 12000, 'FALSE');

INSERT INTO customers VALUES (4, 'Priya', 45, 8.5, 11000, 'FALSE');

INSERT INTO customers VALUES (5, 'Rahul', 70, 7.2, 8000, 'FALSE');

INSERT INTO customers VALUES (6, 'Deepa', 35, 8.8, 10500, 'FALSE');

INSERT INTO customers VALUES (7, 'Vikram', 61, 9.2, 13000, 'FALSE');

INSERT INTO customers VALUES (8, 'Kavya', 29, 8.6, 5000, 'FALSE');

INSERT INTO customers VALUES (9, 'Anil', 75, 6.9, 14000, 'FALSE');

INSERT INTO customers VALUES (10, 'Meena', 50, 7.8, 9100, 'FALSE');

select \* from customers;

--Loans data

INSERT INTO loans VALUES (101, 1, SYSDATE + 10);

INSERT INTO loans VALUES (102, 2, SYSDATE + 45);

INSERT INTO loans VALUES (103, 3, SYSDATE + 5);

INSERT INTO loans VALUES (104, 4, SYSDATE + 25);

INSERT INTO loans VALUES (105, 5, SYSDATE + 15);

INSERT INTO loans VALUES (106, 6, SYSDATE + 31);

INSERT INTO loans VALUES (107, 7, SYSDATE + 7);

INSERT INTO loans VALUES (108, 8, SYSDATE + 60);

INSERT INTO loans VALUES (109, 9, SYSDATE + 20);

INSERT INTO loans VALUES (110, 10, SYSDATE + 2);

select \* from loans;

commit;

SET SERVEROUTPUT ON;

**Scenario 1:**

DECLARE

  v\_discounted\_rate NUMBER(5,2);

BEGIN

  FOR cust IN (

    SELECT customer\_id, name, age, interest\_rate FROM customers

  ) LOOP

    IF cust.age > 60 THEN

      v\_discounted\_rate := cust.interest\_rate - (cust.interest\_rate \* 0.01);

      UPDATE customers

      SET interest\_rate = v\_discounted\_rate

      WHERE customer\_id = cust.customer\_id;

      DBMS\_OUTPUT.PUT\_LINE('Discount applied to ' || cust.name ||

                           ' (Age: ' || cust.age ||

                           ', New Interest Rate: ' || v\_discounted\_rate || '%)');

    END IF;

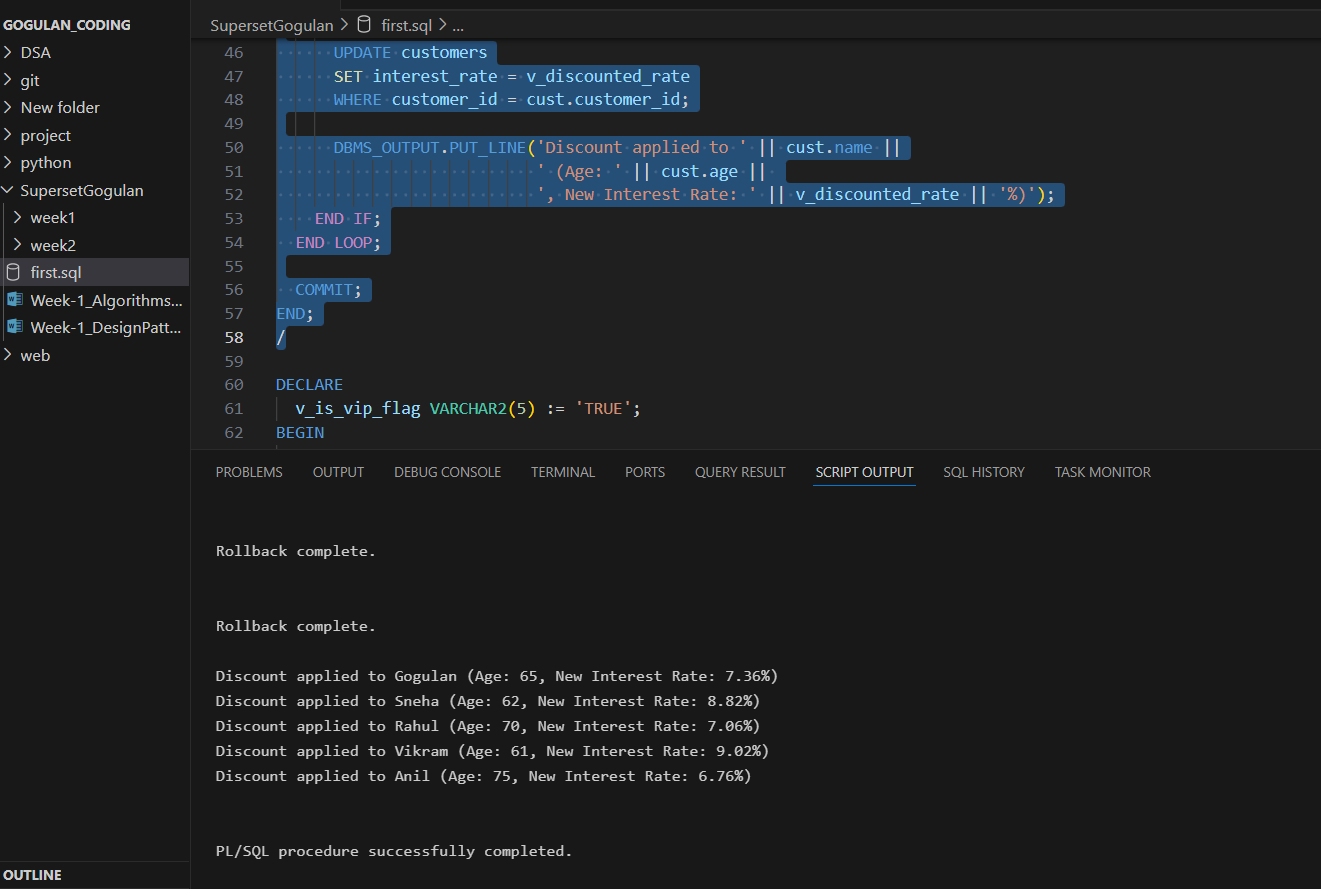
  END LOOP;

  COMMIT;

END;

/

**OUTPUT:**



**Scenario 2**

DECLARE

  v\_is\_vip\_flag VARCHAR2(5) := 'TRUE';

BEGIN

  FOR cust IN (

    SELECT customer\_id, name, balance FROM customers

  ) LOOP

    IF cust.balance > 10000 THEN

      UPDATE customers

      SET isvip = v\_is\_vip\_flag

      WHERE customer\_id = cust.customer\_id;

      DBMS\_OUTPUT.PUT\_LINE('Customer ' || cust.name ||

                           ' promoted to VIP (Balance: $' || cust.balance || ')');

    END IF;

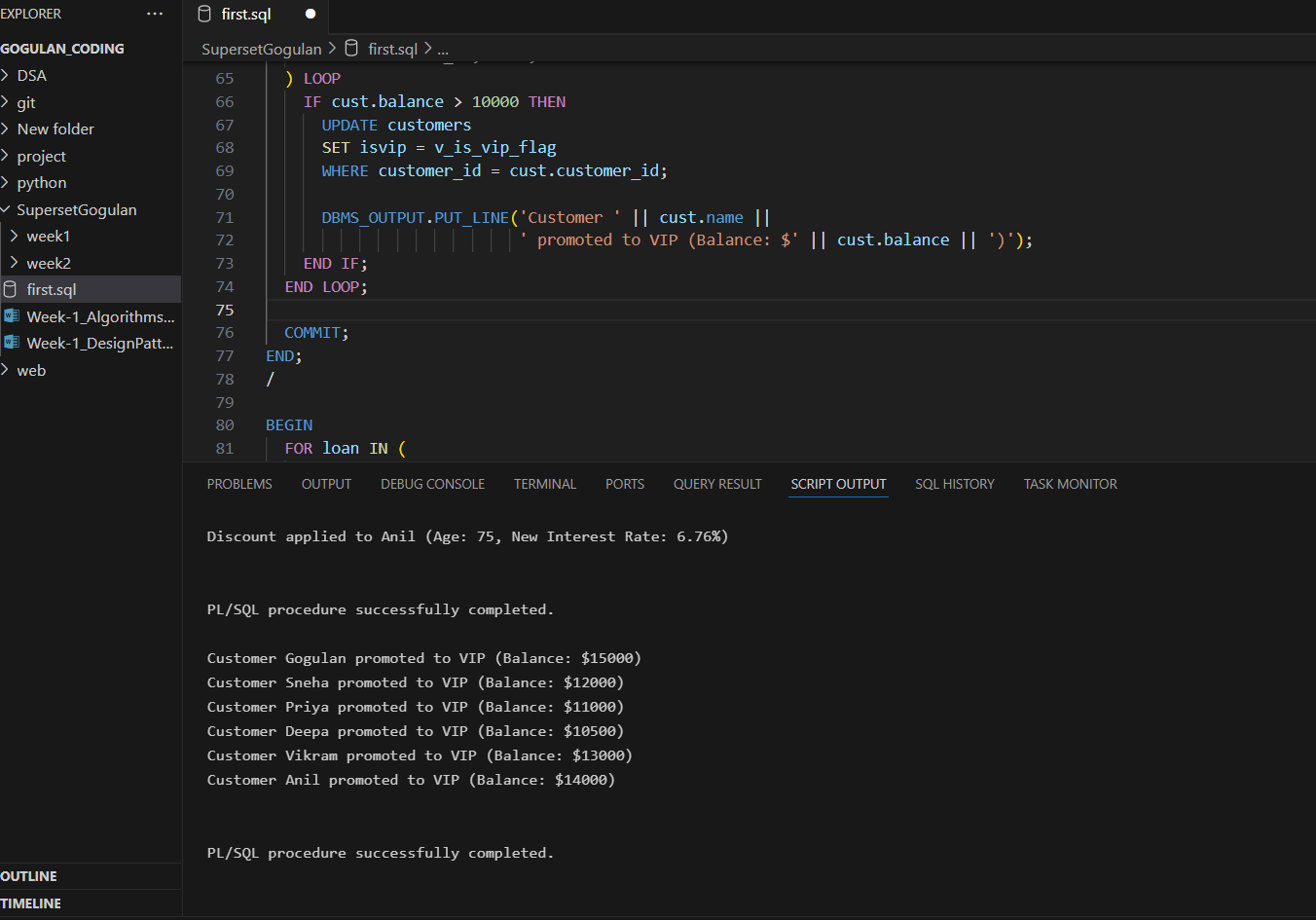
  END LOOP;

  COMMIT;

END;

/

**OUTPUT:**



**Scenario 3**

BEGIN

  FOR loan IN (

    SELECT loan\_id, customer\_id, due\_date FROM loans

    WHERE due\_date <= SYSDATE + 30

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan #' || loan.loan\_id ||

                         ' for Customer ID ' || loan.customer\_id ||

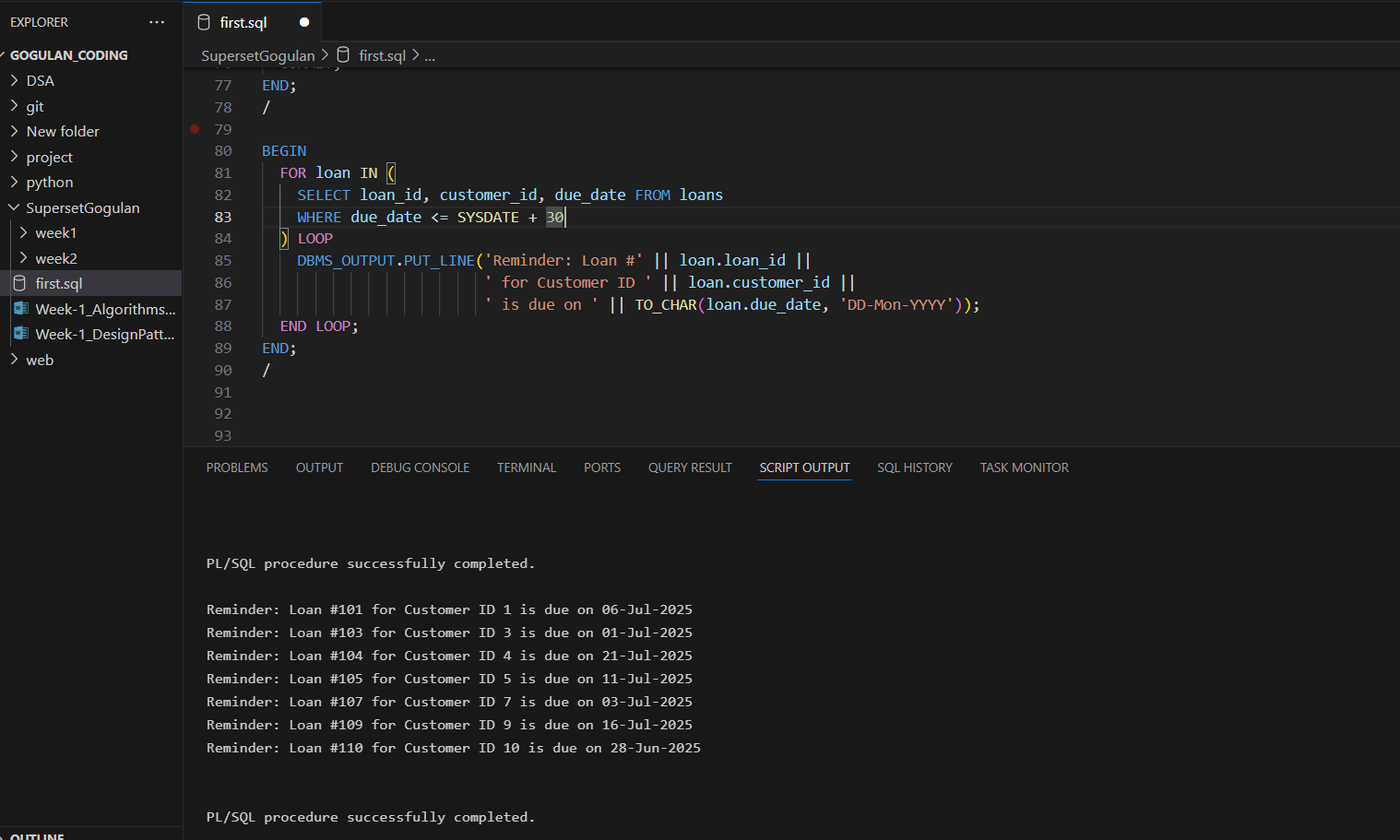
                         ' is due on ' || TO\_CHAR(loan.due\_date, 'DD-Mon-YYYY'));

  END LOOP;

END;

/

**OUTPUT:**



**Exercise 3: Stored Procedures**

-- Savings Accounts Table

CREATE TABLE savings\_accounts (

  account\_id NUMBER PRIMARY KEY,

  customer\_id NUMBER,

  balance NUMBER(12,2)

);

-- Employees Table

CREATE TABLE employees (

  emp\_id NUMBER PRIMARY KEY,

  name VARCHAR2(50),

  department\_id NUMBER,

  salary NUMBER(10,2)

);

-- General Accounts Table (used in TransferFunds)

CREATE TABLE accounts (

  account\_id NUMBER PRIMARY KEY,

  customer\_id NUMBER,

  balance NUMBER(12,2)

);

-- Insert into savings\_accounts

INSERT INTO savings\_accounts VALUES (101, 1, 5000);

INSERT INTO savings\_accounts VALUES (102, 2, 8000);

INSERT INTO savings\_accounts VALUES (103, 3, 12000);

-- Insert into employees

INSERT INTO employees VALUES (1, 'Gogulan', 201, 30000);

INSERT INTO employees VALUES (2, 'Arun', 202, 25000);

INSERT INTO employees VALUES (3, 'Sneha', 201, 28000);

-- Insert into accounts

INSERT INTO accounts VALUES (1001, 1, 15000);

INSERT INTO accounts VALUES (1002, 2, 10000);

INSERT INTO accounts VALUES (1003, 3, 5000);

COMMIT;

**Scenario 1:**

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

  v\_interest NUMBER(10,2);

BEGIN

  FOR acc IN (SELECT account\_id, balance FROM savings\_accounts) LOOP

    v\_interest := acc.balance \* 0.01;

    UPDATE savings\_accounts

    SET balance = balance + v\_interest

    WHERE account\_id = acc.account\_id;

    DBMS\_OUTPUT.PUT\_LINE('Interest ₹' || v\_interest || ' applied to Account ID: ' || acc.account\_id);

  END LOOP;

  COMMIT;

END;

/

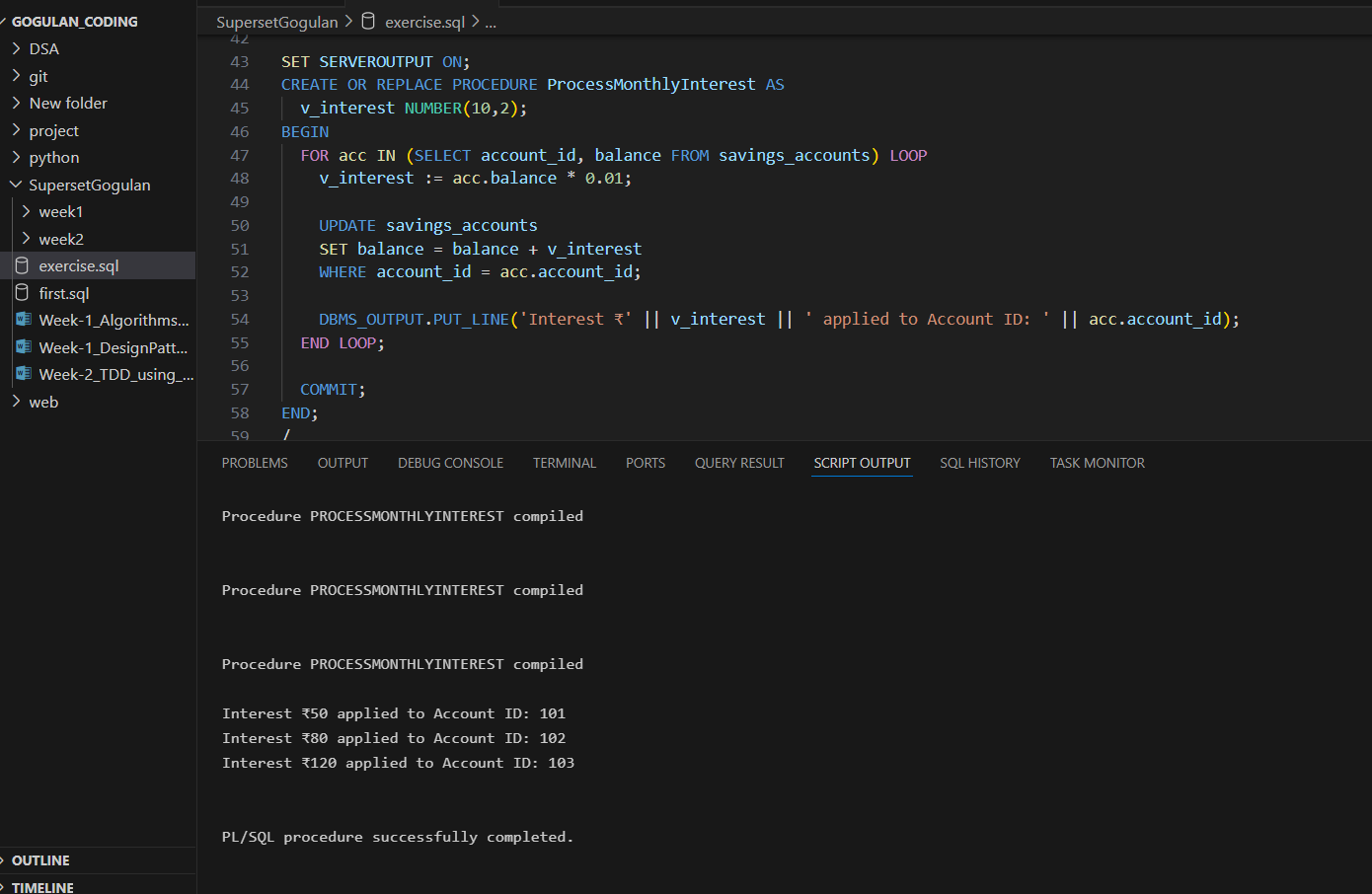
BEGIN

  ProcessMonthlyInterest;

END;

/

**OUTPUT:**



**Scenario 2:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

  p\_dept\_id IN NUMBER,

  p\_bonus\_pct IN NUMBER

) AS

  v\_bonus\_amt NUMBER(10,2);

BEGIN

  FOR emp IN (

    SELECT emp\_id, name, salary FROM employees WHERE department\_id = p\_dept\_id

  ) LOOP

    v\_bonus\_amt := emp.salary \* (p\_bonus\_pct / 100);

    UPDATE employees

    SET salary = salary + v\_bonus\_amt

    WHERE emp\_id = emp.emp\_id;

    DBMS\_OUTPUT.PUT\_LINE('Bonus ₹' || v\_bonus\_amt || ' added to ' || emp.name);

  END LOOP;

  COMMIT;

END;

/

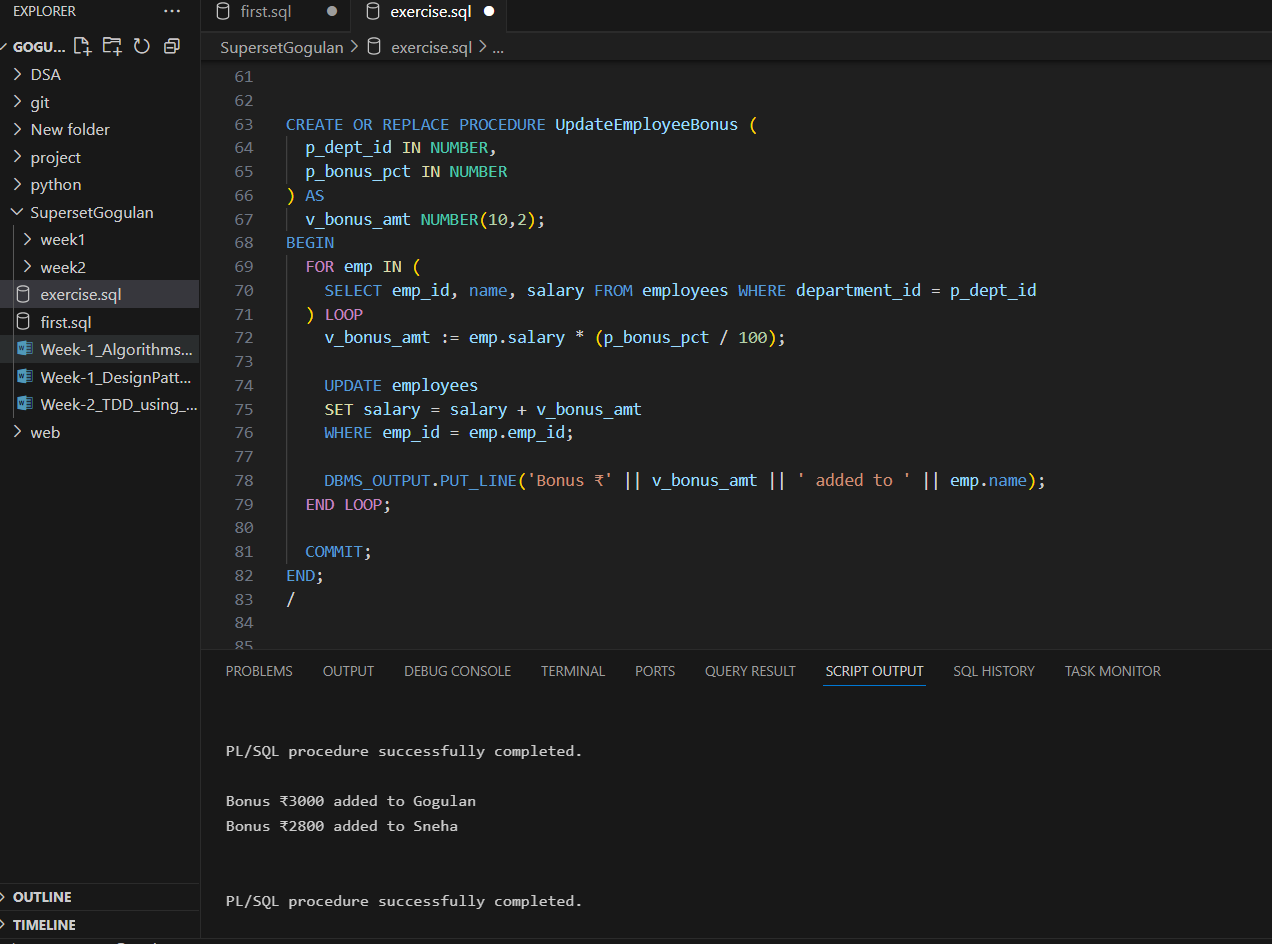
BEGIN

  UpdateEmployeeBonus(201, 10);

END;

/

**OUTPUT:**



**Scenario 3:**

CREATE OR REPLACE PROCEDURE TransferFunds (

  p\_from\_acc IN NUMBER,

  p\_to\_acc IN NUMBER,

  p\_amount IN NUMBER

) AS

  v\_from\_balance NUMBER(12,2);

BEGIN

  SELECT balance INTO v\_from\_balance FROM accounts WHERE account\_id = p\_from\_acc;

  IF v\_from\_balance < p\_amount THEN

    DBMS\_OUTPUT.PUT\_LINE('Transfer Failed: Insufficient balance in Account ' || p\_from\_acc);

  ELSE

    UPDATE accounts

    SET balance = balance - p\_amount

    WHERE account\_id = p\_from\_acc;

    UPDATE accounts

    SET balance = balance + p\_amount

    WHERE account\_id = p\_to\_acc;

    DBMS\_OUTPUT.PUT\_LINE('₹' || p\_amount || ' transferred from Account ' || p\_from\_acc ||

                         ' to Account ' || p\_to\_acc);

    COMMIT;

  END IF;

END;

/

BEGIN

  TransferFunds(1001, 1003, 2000);

END;

/

**OUTPUT:**

