**WEEK 2-PL/SQL PROGRAMMING**

**SUPERSET ID:6383615**

**EXERCISE 1-CONTROL STRUCTURES**

**SCENARIO 1**

CREATE TABLE bank\_customers (

   customer\_id     NUMBER PRIMARY KEY,

   name            VARCHAR2(50),

   age             NUMBER,

   balance         NUMBER,

   interest\_rate   NUMBER(5,2),

   isVIP           VARCHAR2(5)

);

INSERT INTO bank\_customers VALUES (1, 'Alice', 65, 12000, 7.5, 'FALSE');

INSERT INTO bank\_customers VALUES (2, 'Bob', 45, 8000, 6.0, 'FALSE');

INSERT INTO bank\_customers VALUES (3, 'Charlie', 70, 15000, 8.0, 'FALSE');

INSERT INTO bank\_customers VALUES (4, 'David', 30, 5000, 5.5, 'FALSE');

COMMIT;

/

select \* from bank\_customers;

**INITIAL TABLE**

A screenshot of a white background

AI-generated content may be incorrect.

BEGIN

   FOR customer\_rec IN (

      SELECT customer\_id

      FROM bank\_customers

      WHERE age > 60

   ) LOOP

      UPDATE bank\_customers

      SET interest\_rate = interest\_rate - 1

      WHERE customer\_id = customer\_rec.customer\_id;

   END LOOP;

   COMMIT;

END;

/

select \* from bank\_customers;

**OUTPUT FOR SCENARIO 1**

A screenshot of a computer

AI-generated content may be incorrect.

**SCENARIO 2**

BEGIN

   FOR customer\_rec IN (

      SELECT customer\_id

      FROM bank\_customers

      WHERE balance > 10000

   ) LOOP

      UPDATE bank\_customers

      SET isVIP = 'TRUE'

      WHERE customer\_id = customer\_rec.customer\_id;

   END LOOP;

   COMMIT;

END;

/

SELECT

   customer\_id, name, age,  balance,  interest\_rate, isVIP

FROM

   bank\_customers

ORDER BY

   customer\_id;

**OUTPUT FOR SCENARIO 2**

A screenshot of a computer

AI-generated content may be incorrect.

**SCENARIO 3**

ALTER TABLE bank\_customers ADD (due\_date DATE);

UPDATE bank\_customers SET due\_date = SYSDATE + 10 WHERE customer\_id = 1;

UPDATE bank\_customers SET due\_date = SYSDATE + 35 WHERE customer\_id = 2;

UPDATE bank\_customers SET due\_date = SYSDATE + 5  WHERE customer\_id = 3;

UPDATE bank\_customers SET due\_date = SYSDATE - 2  WHERE customer\_id = 4;

COMMIT;

BEGIN

   FOR cust\_rec IN (

      SELECT name, due\_date

      FROM bank\_customers

      WHERE due\_date BETWEEN SYSDATE AND SYSDATE + 30

   ) LOOP

      DBMS\_OUTPUT.PUT\_LINE(

         'Reminder: Loan for ' || cust\_rec.name ||

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

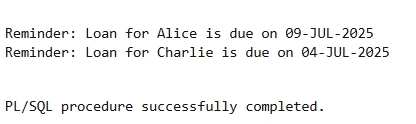
      );

   END LOOP;

END;

/

**OUTPUT FOR REMINDER**



SELECT

   customer\_id,

   name,

   age,

   balance,

   interest\_rate,

   isVIP,

   TO\_CHAR(due\_date, 'DD-MON-YYYY') AS due\_date

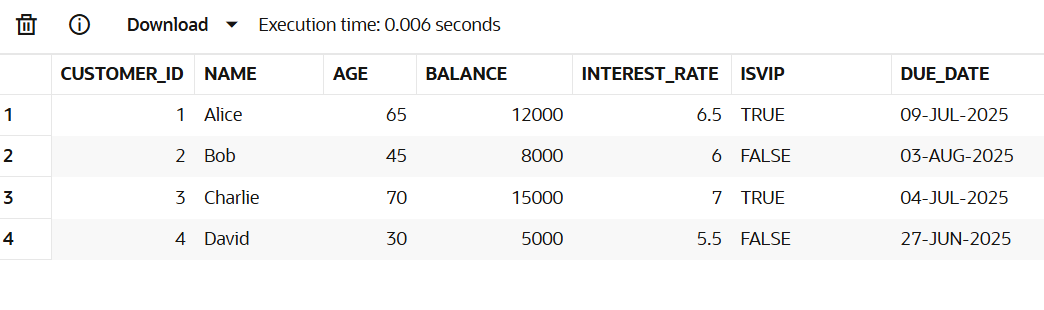
FROM

   bank\_customers

ORDER BY

   customer\_id;

**FINAL TABLE OUTPUT FOR SCENARIO 3**



**EXERCISE 3-STORED PROCEDURES**

**SCENARIO 1**

CREATE TABLE savings\_accounts (

   account\_id    NUMBER PRIMARY KEY,

   customer\_name VARCHAR2(50),

   balance       NUMBER(10, 2)

);

/

INSERT INTO savings\_accounts VALUES (101, 'Arun', 10000);

INSERT INTO savings\_accounts VALUES (102, 'Aravind', 7500);

INSERT INTO savings\_accounts VALUES (103, 'Ram', 15000);

COMMIT;

/

select \* from savings\_accounts;

**INITIAL TABLE**A screenshot of a computer

AI-generated content may be incorrect.

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

   UPDATE savings\_accounts

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

   COMMIT;

END;

/

EXEC ProcessMonthlyInterest;

**OUTPUT FOR SCENARIO 1**

A screenshot of a phone

AI-generated content may be incorrect.**SCENARIO 2**

CREATE TABLE employees (

   emp\_id      NUMBER PRIMARY KEY,

   name        VARCHAR2(50),

   department  VARCHAR2(30),

   salary      NUMBER(10, 2)

);

/

INSERT INTO employees VALUES (1, 'Arun', 'HR', 40000);

INSERT INTO employees VALUES (2, 'Aravind', 'IT', 50000);

INSERT INTO employees VALUES (3, 'Ram', 'IT', 60000);

COMMIT;

/

SELECT \* FROM employees ORDER BY emp\_id;

**INITIAL TABLE**

A screenshot of a computer

AI-generated content may be incorrect.

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

   dept\_name IN VARCHAR2,

   bonus\_percent IN NUMBER

) IS

BEGIN

   UPDATE employees

   SET salary = salary + (salary \* bonus\_percent / 100)

   WHERE department = dept\_name;

   COMMIT;

END;

/

EXEC UpdateEmployeeBonus('IT', 10);

SELECT \* FROM employees ORDER BY emp\_id;

**OUTPUT FOR SCENARIO 2**

A screenshot of a computer

AI-generated content may be incorrect.

**SCENARIO 3**

CREATE TABLE accounts (

   account\_number NUMBER PRIMARY KEY,

   customer\_name  VARCHAR2(50),

   balance        NUMBER(10, 2)

);

/

INSERT INTO accounts VALUES (201, 'Arun', 8000);

INSERT INTO accounts VALUES (202, 'Aravind', 5000);

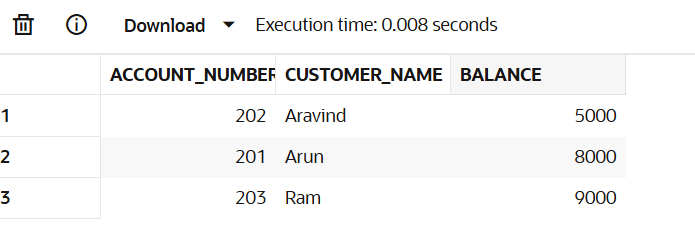
INSERT INTO accounts VALUES (203, 'Ram', 9000);

COMMIT;

/

SELECT \* FROM accounts;

**INITIAL TABLE**



CREATE OR REPLACE PROCEDURE TransferFunds (

   from\_account IN NUMBER,

   to\_account IN NUMBER,

   amount IN NUMBER

) IS

   insufficient\_balance EXCEPTION;

   curr\_balance NUMBER;

BEGIN

   SELECT balance INTO curr\_balance FROM accounts WHERE account\_number = from\_account;

   IF curr\_balance < amount THEN

      RAISE insufficient\_balance;

   END IF;

   UPDATE accounts

   SET balance = balance - amount

   WHERE account\_number = from\_account;

   UPDATE accounts

   SET balance = balance + amount

   WHERE account\_number = to\_account;

   COMMIT;

EXCEPTION

   WHEN insufficient\_balance THEN

      DBMS\_OUTPUT.PUT\_LINE('Error: Insufficient balance in source account.');

   WHEN NO\_DATA\_FOUND THEN

      DBMS\_OUTPUT.PUT\_LINE('Error: Account not found.');

   WHEN OTHERS THEN

      DBMS\_OUTPUT.PUT\_LINE('Unexpected error: ' || SQLERRM);

END;

/

EXEC TransferFunds(201, 202, 3000);

SELECT \* FROM accounts ORDER BY account\_number;

**OUTPUT FOR SCENARIO 3(AMOUNT TRANSFERRED FROM ARUN TO ARAVIND)**

A screenshot of a computer

AI-generated content may be incorrect.