PL SQL Programming

**Exercise 3: Stored Procedures**

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

* + **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

* + **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

**Scenario 3:** Customers should be able to transfer funds between their accounts.

* + **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

QUERY:

CREATE TABLE accounts (

account\_id NUMBER PRIMARY KEY,

customer\_id NUMBER,

account\_type VARCHAR2(20),

balance NUMBER

);

CREATE TABLE employees (

emp\_id NUMBER PRIMARY KEY,

emp\_name VARCHAR2(50),

salary NUMBER,

department\_id NUMBER

);

INSERT INTO accounts VALUES (101, 1, 'SAVINGS', 10000);

INSERT INTO accounts VALUES (102, 1, 'CURRENT', 5000);

INSERT INTO accounts VALUES (103, 2, 'SAVINGS', 20000);

INSERT INTO accounts VALUES (104, 3, 'CURRENT', 7000);

INSERT INTO employees VALUES (1, 'Alice', 30000, 10);

INSERT INTO employees VALUES (2, 'Bob', 25000, 20);

INSERT INTO employees VALUES (3, 'Charlie', 22000, 10);

INSERT INTO employees VALUES (4, 'Diana', 27000, 30);

COMMIT;

SCENARIO 1:

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

* + **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

QUERY:

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

UPDATE accounts

SET balance = balance + (balance \* 0.01)

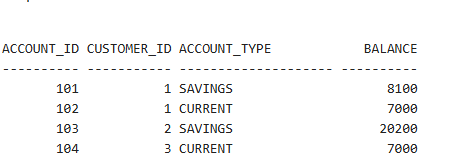
WHERE account\_type = 'SAVINGS';

COMMIT;

END;

/

OUTPUT:



SCENARIO 2:

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

* + **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

QUERY:

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_dept\_id IN NUMBER,

p\_bonus\_percent IN NUMBER

) AS

BEGIN

UPDATE employees

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

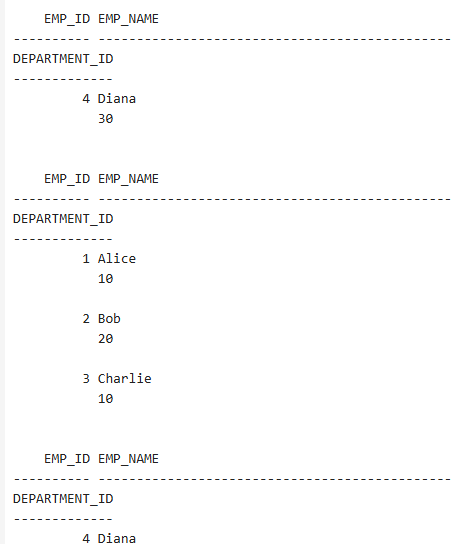
WHERE department\_id = p\_dept\_id;

COMMIT;

END;

/

OUTPUT:



SCENARIO 3:

**Scenario 3:** Customers should be able to transfer funds between their accounts.

* + **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

QUERY:

CREATE OR REPLACE PROCEDURE TransferFunds(

p\_from\_account IN NUMBER,

p\_to\_account IN NUMBER,

p\_amount IN NUMBER

) AS

v\_balance NUMBER;

BEGIN

SELECT balance INTO v\_balance FROM accounts WHERE account\_id = p\_from\_account FOR UPDATE;

IF v\_balance >= p\_amount THEN

UPDATE accounts

SET balance = balance - p\_amount

WHERE account\_id = p\_from\_account;

UPDATE accounts

SET balance = balance + p\_amount

WHERE account\_id = p\_to\_account;

COMMIT;

ELSE

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

END IF;

END;

/

BEGIN

ProcessMonthlyInterest;

END;

/

BEGIN

UpdateEmployeeBonus(10, 10);

END;

/

BEGIN

TransferFunds(101, 102, 2000);

END;

/

SELECT \* FROM accounts;

SELECT \* FROM employees;

OUTPUT:

