**Exercise 3: Stored Procedures  
  
Tables:**

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20), -- e.g., 'Savings', 'Checking'

Balance NUMBER

);

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

DepartmentID NUMBER,

Salary NUMBER

);

INSERT INTO Accounts VALUES (101, 1, 'Savings', 5000);

INSERT INTO Accounts VALUES (102, 1, 'Checking', 3000);

INSERT INTO Accounts VALUES (103, 2, 'Savings', 8000);

INSERT INTO Accounts VALUES (104, 2, 'Checking', 2000);

INSERT INTO Employees VALUES (1, 'Alice', 10, 40000);

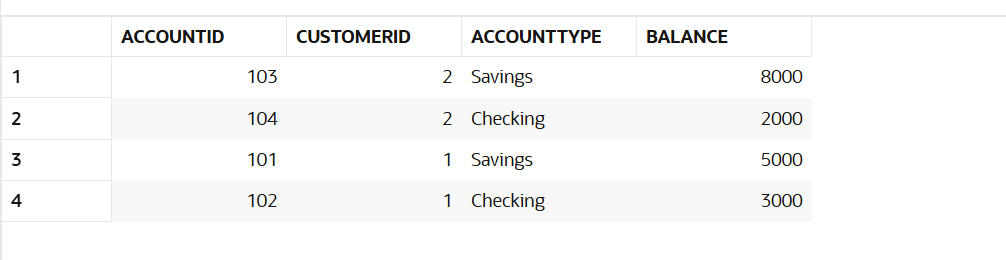
INSERT INTO Employees VALUES (2, 'Bob', 10, 45000);

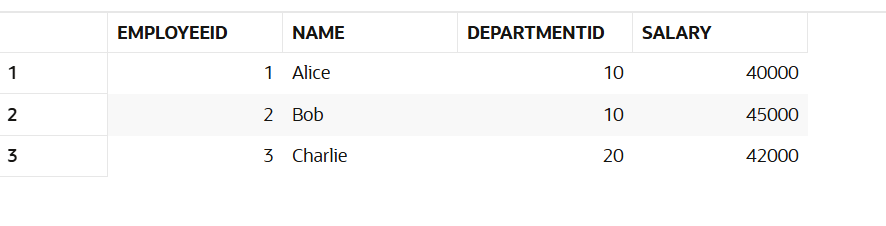
INSERT INTO Employees VALUES (3, 'Charlie', 20, 42000);

Select \* from Accounts;

Select \* from Employees;

**Output:**

****

****

**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.

**Answer:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountType = 'Savings';

COMMIT;

END;

/

//calling the procedure

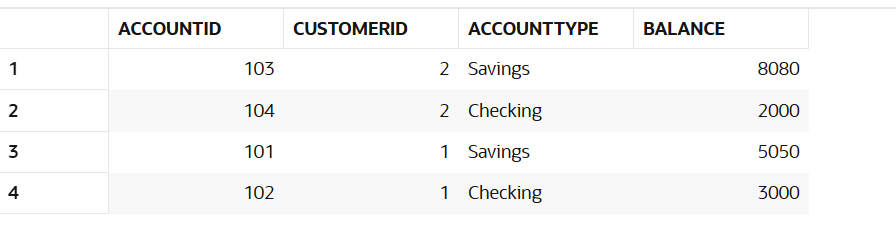
BEGIN

ProcessMonthlyInterest;

END;

/

**Output(Table after Scenario 1):**

****

**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.

**Answer:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_DepartmentID IN NUMBER,

p\_BonusPercent IN NUMBER

) IS

BEGIN

UPDATE Employees

SET Salary = Salary + (Salary \* p\_BonusPercent / 100)

WHERE DepartmentID = p\_DepartmentID;

COMMIT;

END;

//calling the procedure

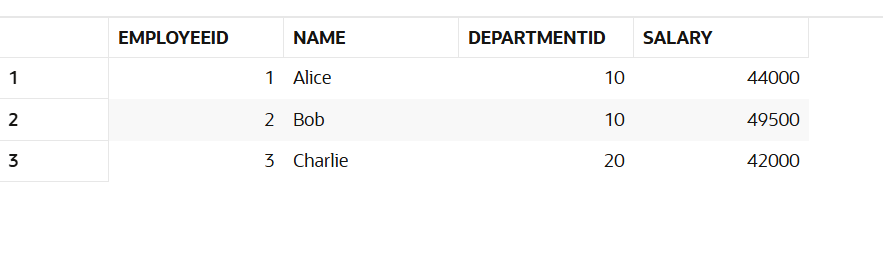
BEGIN

UpdateEmployeeBonus(10, 10);

END;

/

**Output(Table after Scenario 2):**



**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.

**Answer:**

CREATE OR REPLACE PROCEDURE TransferFunds(

p\_FromAccount IN NUMBER,

p\_ToAccount IN NUMBER,

p\_Amount IN NUMBER

) IS

v\_Balance NUMBER;

BEGIN

SELECT Balance INTO v\_Balance

FROM Accounts

WHERE AccountID = p\_FromAccount

FOR UPDATE;

IF v\_Balance >= p\_Amount THEN

UPDATE Accounts

SET Balance = Balance - p\_Amount

WHERE AccountID = p\_FromAccount;

UPDATE Accounts

SET Balance = Balance + p\_Amount

WHERE AccountID = p\_ToAccount;

COMMIT;

ELSE

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds in source account.');

END IF;

END;

/

//calling the procedure

BEGIN

TransferFunds(101, 102, 1000);

END;

/

**Output(Table after Scenario 3):**

