**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.

**Answer:**

SELECT \* FROM ACCOUNTS;

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE PROCESSMONTHLYINTEREST AS

BEGIN

UPDATE ACCOUNTS

SET BALANCE = BALANCE \* 1.01,

LASTMODIFIED = SYSDATE

WHERE ACCOUNTTYPE = 'Savings';

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed for all savings accounts.');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error processing monthly interest: ' || SQLERRM);

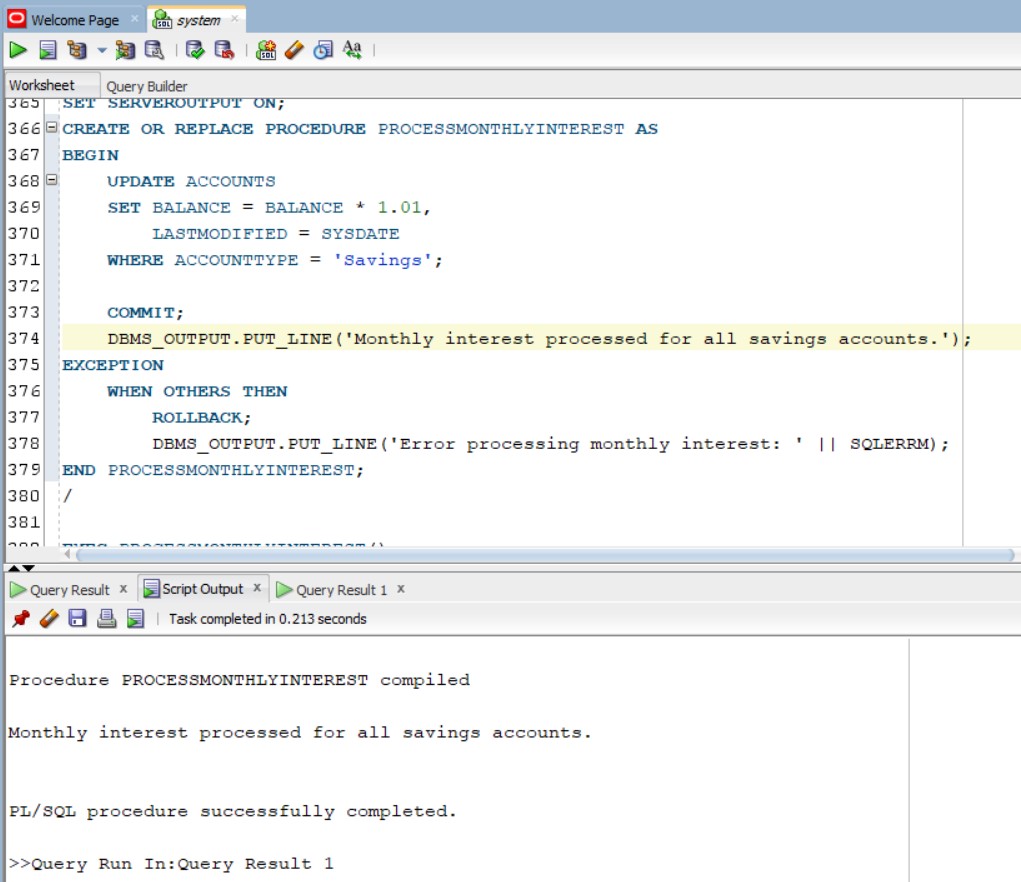
END PROCESSMONTHLYINTEREST;

/

EXEC PROCESSMONTHLYINTEREST();

SELECT \* FROM ACCOUNTS;

**Output:**



**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:**

SELECT \* FROM EMPLOYEES;

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE UPDATEEMPLOYEEBONUS(

P\_DEPARTMENT IN EMPLOYEES.DEPARTMENT%TYPE,

P\_BONUS\_PERCENTAGE IN NUMBER

) AS

BEGIN

UPDATE EMPLOYEES

SET SALARY = SALARY \* (1 + P\_BONUS\_PERCENTAGE / 100),

HIREDATE = SYSDATE

WHERE DEPARTMENT = P\_DEPARTMENT;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to employees in the ' || P\_DEPARTMENT || ' department.');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error updating employee bonuses: ' || SQLERRM);

END UPDATEEMPLOYEEBONUS;

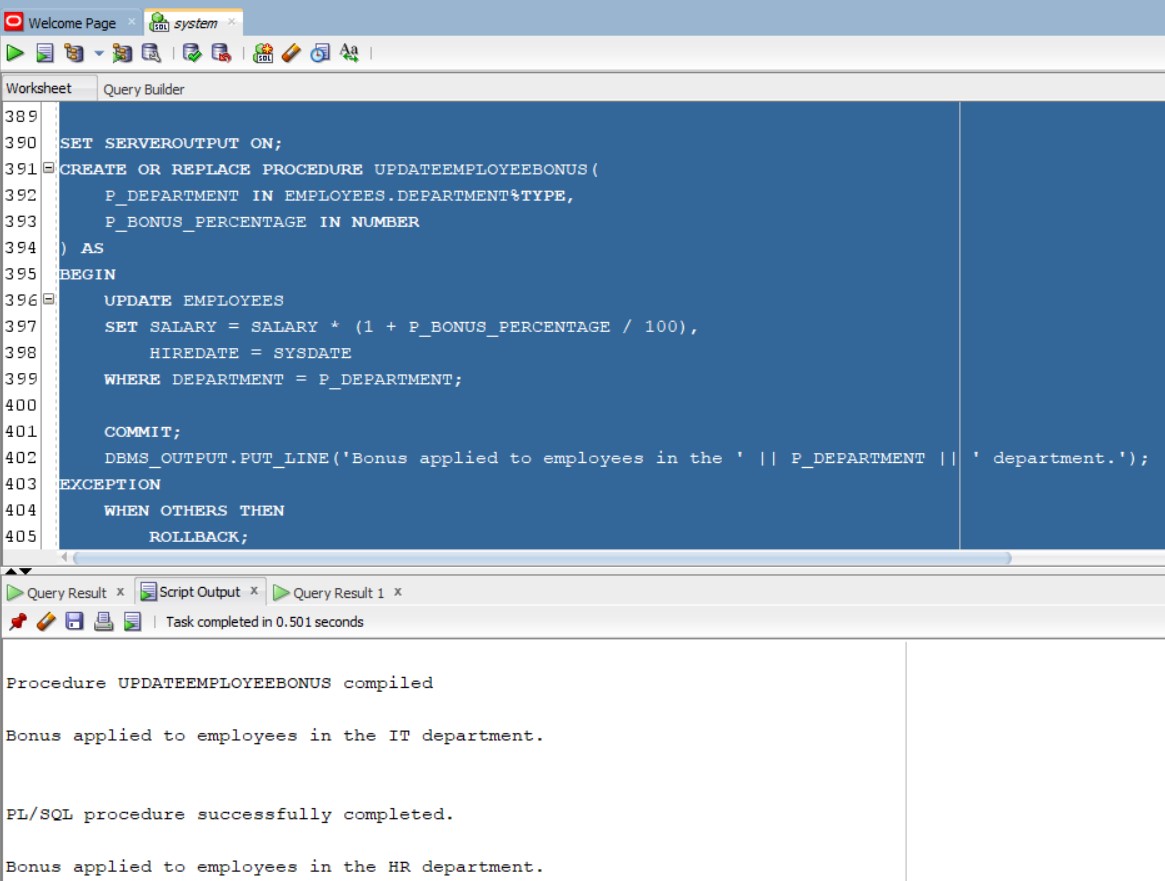
/

EXEC UPDATEEMPLOYEEBONUS('IT',5);

EXEC UPDATEEMPLOYEEBONUS('HR',3);

SELECT \* FROM EMPLOYEES;

**Output:**



**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:**

SELECT \* FROM ACCOUNTS;

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE TRANSFERFUNDS(

P\_FROM\_ACCOUNT\_ID IN ACCOUNTS.ACCOUNTID%TYPE,

P\_TO\_ACCOUNT\_ID IN ACCOUNTS.ACCOUNTID%TYPE,

P\_AMOUNT IN NUMBER

) AS

V\_FROM\_BALANCE ACCOUNTS.BALANCE%TYPE;

BEGIN

SELECT BALANCE INTO V\_FROM\_BALANCE

FROM ACCOUNTS

WHERE ACCOUNTID = P\_FROM\_ACCOUNT\_ID

FOR UPDATE;

-- Check for sufficient funds

IF V\_FROM\_BALANCE < P\_AMOUNT THEN

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

END IF;

-- Perform the transfer

UPDATE ACCOUNTS

SET BALANCE = BALANCE - P\_AMOUNT,

LASTMODIFIED = SYSDATE

WHERE ACCOUNTID = P\_FROM\_ACCOUNT\_ID;

UPDATE ACCOUNTS

SET BALANCE = BALANCE + P\_AMOUNT,

LASTMODIFIED = SYSDATE

WHERE ACCOUNTID = P\_TO\_ACCOUNT\_ID;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transfer of ' || P\_AMOUNT || ' from account ' || P\_FROM\_ACCOUNT\_ID || ' to account ' || P\_TO\_ACCOUNT\_ID || ' completed successfully.');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Transfer failed: ' || SQLERRM);

END TRANSFERFUNDS;

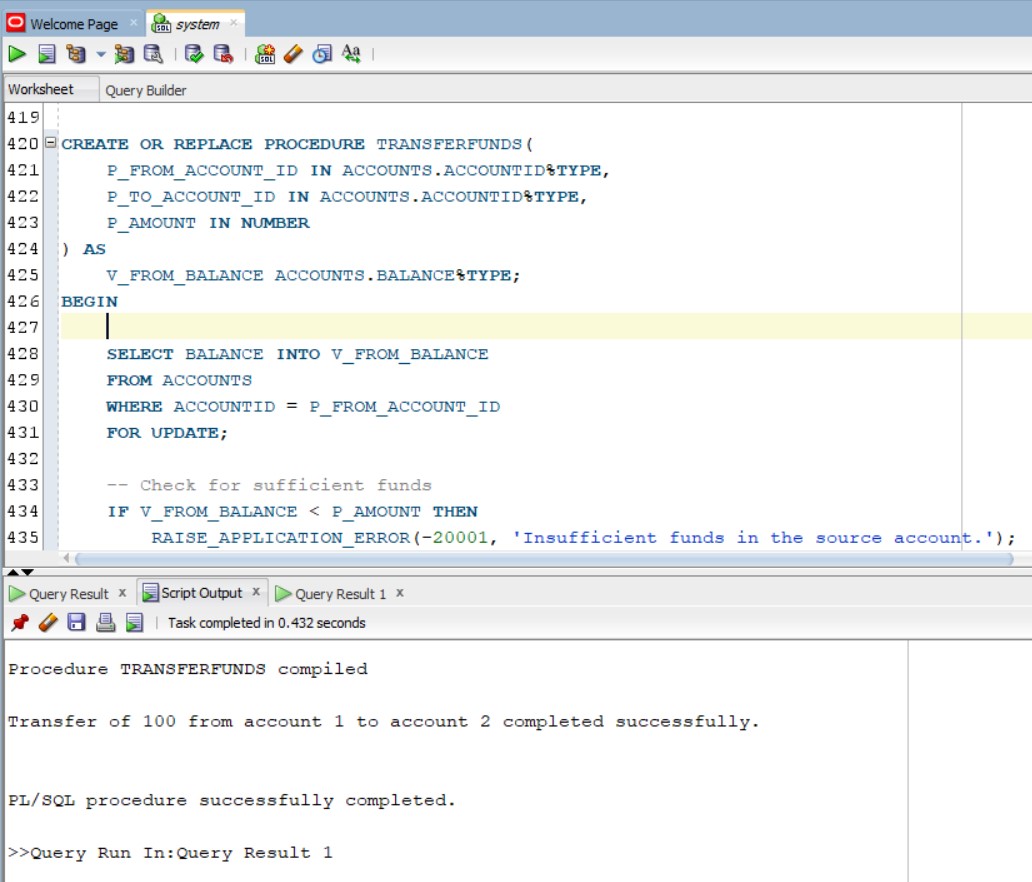
/

EXEC TRANSFERFUNDS(1,2,100);

SELECT \* FROM ACCOUNTS;

/\*

**Output:**



**-- Schema to be Created**

CREATE TABLE CUSTOMERS (

CUSTOMERID NUMBER PRIMARY KEY,

NAME VARCHAR2(100),

DOB DATE,

BALANCE NUMBER,

LASTMODIFIED DATE

);

CREATE TABLE ACCOUNTS (

ACCOUNTID NUMBER PRIMARY KEY,

CUSTOMERID NUMBER,

ACCOUNTTYPE VARCHAR2(20),

BALANCE NUMBER,

LASTMODIFIED DATE,

FOREIGN KEY ( CUSTOMERID )

REFERENCES CUSTOMERS ( CUSTOMERID )

);

CREATE TABLE TRANSACTIONS (

TRANSACTIONID NUMBER PRIMARY KEY,

ACCOUNTID NUMBER,

TRANSACTIONDATE DATE,

AMOUNT NUMBER,

TRANSACTIONTYPE VARCHAR2(10),

FOREIGN KEY ( ACCOUNTID )

REFERENCES ACCOUNTS ( ACCOUNTID )

);

CREATE TABLE LOANS (

LOANID NUMBER PRIMARY KEY,

CUSTOMERID NUMBER,

LOANAMOUNT NUMBER,

INTERESTRATE NUMBER,

STARTDATE DATE,

ENDDATE DATE,

FOREIGN KEY ( CUSTOMERID )

REFERENCES CUSTOMERS ( CUSTOMERID )

);

CREATE TABLE EMPLOYEES (

EMPLOYEEID NUMBER PRIMARY KEY,

NAME VARCHAR2(100),

POSITION VARCHAR2(50),

SALARY NUMBER,

DEPARTMENT VARCHAR2(50),

HIREDATE DATE

);

-- Example Scripts for Sample Data Insertion

-- INSERT INTO CUSTOMERS

INSERT INTO CUSTOMERS (CUSTOMERID, NAME, DOB, BALANCE, LASTMODIFIED)

VALUES (1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);

INSERT INTO CUSTOMERS (CUSTOMERID, NAME, DOB, BALANCE, LASTMODIFIED)

VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);

-- INSERT INTO ACCOUNTS

INSERT INTO ACCOUNTS (ACCOUNTID, CUSTOMERID, ACCOUNTTYPE, BALANCE, LASTMODIFIED)

VALUES (1, 1, 'Savings', 1000, SYSDATE);

INSERT INTO ACCOUNTS (ACCOUNTID, CUSTOMERID, ACCOUNTTYPE, BALANCE, LASTMODIFIED)

VALUES (2, 2, 'Checking', 1500, SYSDATE);

-- INSTER INTO TRANSACTIONS

INSERT INTO TRANSACTIONS (TRANSACTIONID, ACCOUNTID, TRANSACTIONDATE, AMOUNT, TRANSACTIONTYPE)

VALUES (1, 1, SYSDATE, 200, 'Deposit');

INSERT INTO TRANSACTIONS (TRANSACTIONID, ACCOUNTID, TRANSACTIONDATE, AMOUNT, TRANSACTIONTYPE)

VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

-- INSERT INTO LOANS

INSERT INTO LOANS (LOANID, CUSTOMERID, LOANAMOUNT, INTERESTRATE, STARTDATE, ENDDATE)

VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 60));

-- INSERT INTO EMPLOYEES

INSERT INTO EMPLOYEES (EMPLOYEEID, NAME, POSITION, SALARY, DEPARTMENT, HIREDATE)

VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', TO\_DATE('2015-06-15', 'YYYY-MM-DD'));

INSERT INTO EMPLOYEES (EMPLOYEEID, NAME, POSITION, SALARY, DEPARTMENT, HIREDATE)

VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', TO\_DATE('2017-03-20', 'YYYY-MM-DD'));