**Exercise 2: Error Handling**

**Scenario 1: Handle exceptions during fund transfers between accounts.**

**o Question: Write a stored procedure SafeTransferFunds that transfers funds between two accounts. Ensure that if any error occurs (e.g., insufficient funds), an appropriate error message is logged and the transaction is rolled back.**

CREATE OR REPLACE PROCEDURE SafeTransferFunds (

from1 IN NUMBER,

to1 IN NUMBER,

amount1 IN NUMBER

)

IS

bal NUMBER;

BEGIN

SELECT Balance INTO bal

FROM Accounts

WHERE AccountID = from1;

IF bal < amount1 THEN

DBMS\_OUTPUT.PUT\_LINE('Transfer failed: insufficient funds in source account.');

ROLLBACK;

RETURN;

END IF;

UPDATE Accounts

SET Balance = Balance - amount1

WHERE AccountID = from1;

UPDATE Accounts

SET Balance = Balance + amount1

WHERE AccountID = to1;

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (

6 ,--

from1,

SYSDATE,

amount1,

'Withdrawal'

);

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (

7,

to1,

SYSDATE,

amount1,

'Deposit'

);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transfer completed successfully.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Transfer failed: source account not found.');

ROLLBACK;

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Transfer failed due to unexpected error: ' || SQLERRM);

ROLLBACK;

END;

/

SELECT \* from Transactions;

BEGIN

SafeTransferFunds(1, 2, 300);

END;

/

select \* from transactions;

Ouptut:

A screenshot of a computer screen

AI-generated content may be incorrect.

**Scenario 2: Manage errors when updating employee salaries.**

**o Question: Write a stored procedure UpdateSalary that increases the salary of an employee by a given percentage. If the employee ID does not exist, handle the exception and log an error message.**

CREATE OR REPLACE PROCEDURE UpdateSalary (

empid IN NUMBER,

percent IN NUMBER

)

IS

oldsal NUMBER;

BEGIN

SELECT Salary INTO oldsal

FROM Employees

WHERE EmployeeID = empid;

UPDATE Employees

SET Salary = Salary + (Salary \* percent / 100)

WHERE EmployeeID = empid;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Salary for Employee ID ' || empid || ' updated from ' || TO\_CHAR(oldsal) || ' to ' ||

TO\_CHAR(oldsal + (oldsal \* percent/100)) );

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Employee ID ' || empid || ' does not exist.');

WHEN OTHERS THEN

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

ROLLBACK;

END;

/

BEGIN

UpdateSalary(3, 10);

END;

/

A close up of a number

AI-generated content may be incorrect.

**Scenario 3: Ensure data integrity when adding a new customer.**

**o Question: Write a stored procedure AddNewCustomer that inserts a new customer into the Customers table. If a customer with the same ID already exists, handle the exception by logging an error and preventing the insertion.**

CREATE OR REPLACE PROCEDURE AddNewCustomer (

custid IN NUMBER,

name IN VARCHAR2,

dob IN DATE,

bal IN NUMBER

)

IS

BEGIN

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

VALUES (custid, name, dob, bal, SYSDATE);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Customer ' || custid || ' inserted successfully.');

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Customer with ID ' || custid || ' already exists. Insertion prevented.');

WHEN OTHERS THEN

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

ROLLBACK;

END;

/

BEGIN

AddNewCustomer(1, 'Dwight Schrute', TO\_DATE('1970-01-20', 'YYYY-MM-DD'), 3000);

END;

/

**OUTPUT:**

A close-up of a number

AI-generated content may be incorrect.