Error Handling

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

# 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.

declare

v\_FromAccountID NUMBER := &senderAccountId;

v\_ToAccountID NUMBER := &receiverAccountId;

v\_Amount NUMBER := &amount;

v\_FromBalance NUMBER;

v\_ToBalance NUMBER;

v\_ErrorMsg VARCHAR2(4000);

BEGIN

SAVEPOINT start\_transfer;

BEGIN

SELECT Balance INTO v\_FromBalance

FROM Accounts

WHERE AccountID = v\_FromAccountID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

v\_ErrorMsg := 'Source account not found.';

ROLLBACK TO start\_transfer;

RAISE\_APPLICATION\_ERROR(-20001, v\_ErrorMsg);

END;

IF v\_FromBalance < v\_Amount THEN

v\_ErrorMsg := 'Insufficient funds in the source account.';

ROLLBACK TO start\_transfer;

RAISE\_APPLICATION\_ERROR(-20002, v\_ErrorMsg);

END IF;

BEGIN

SELECT Balance INTO v\_ToBalance

FROM Accounts

WHERE AccountID = v\_ToAccountID;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

v\_ErrorMsg := 'Desnination account not found.';

ROLLBACK TO start\_transfer;

RAISE\_APPLICATION\_ERROR(-20003, v\_ErrorMsg);

END;

UPDATE Accounts

SET Balance = Balance - v\_Amount,

LastModified = SYSDATE WHERE AccountID = v\_FromAccountID;

UPDATE Accounts

SET Balance = Balance + v\_Amount,

LastModified = SYSDATE WHERE AccountID = v\_ToAccountID;

BEGIN

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

VALUES (&tidForDebit, v\_FromAccountID, SYSDATE, v\_Amount, 'Debit');

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

v\_ErrorMsg := 'Transaction Id already exists';

ROLLBACK TO start\_transfer;

RAISE\_APPLICATION\_ERROR(-20004, v\_ErrorMsg);

END;

BEGIN

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

VALUES (&tidForCredit, v\_ToAccountID, SYSDATE, v\_Amount, 'Credit');

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

v\_ErrorMsg := 'Transaction Id already exists';

ROLLBACK TO start\_transfer;

RAISE\_APPLICATION\_ERROR(-20005, v\_ErrorMsg);

END;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

v\_ErrorMsg := SQLERRM;

ROLLBACK TO start\_transfer;

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || v\_ErrorMsg);

RAISE;

END;

/

# Scenario 2: Manage errors when updating employee salaries.

# 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.

declare

v\_EmployeeID NUMBER := &employeeId;

v\_Percentage NUMBER := &salPercent;

v\_Salary NUMBER;

v\_NewSalary NUMBER;

v\_ErrorMsg VARCHAR2(4000);

BEGIN

SAVEPOINT start\_salary\_update;

BEGIN

SELECT Salary INTO v\_Salary

FROM Employees

WHERE EmployeeID = v\_EmployeeID;

v\_NewSalary := v\_Salary \* (1 + v\_Percentage / 100);

UPDATE Employees

SET Salary = v\_NewSalary WHERE EmployeeID = v\_EmployeeID;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('EmployeeId : ' || v\_EmployeeID || ' salary is increased by ' || v\_Percentage);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

v\_ErrorMsg := 'Employee ID ' || v\_EmployeeID || ' does not exist.';

ROLLBACK TO start\_salary\_update;

RAISE\_APPLICATION\_ERROR(-20001, v\_ErrorMsg);

WHEN OTHERS THEN

v\_ErrorMsg := SQLERRM;

ROLLBACK TO start\_transfer;

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || v\_ErrorMsg);

RAISE\_APPLICATION\_ERROR(-20002, v\_ErrorMsg);

END;

END;

/

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

# 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.

declare

v\_ErrorMsg VARCHAR2(4000);

begin

SAVEPOINT starting\_point;

insert into customers (customerid, name, dob, lastmodified) VALUES (&cusId, '&cusName', to\_date('&v\_dob', 'YYYY-MM-DD'), sysdate);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

v\_ErrorMsg := 'Transaction Id already exists';

ROLLBACK TO starting\_point;

RAISE\_APPLICATION\_ERROR(-20001, v\_ErrorMsg);

end;

/