**Module 3 - PL/SQL Programming**

**Exercise 1: Control Structures**

We'll use these tables:

* Customers
* Loans

**Step 1: Create Tables**

CREATE TABLE Customers (

CustomerID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

DOB DATE,

Balance NUMBER,

LastModified DATE

);

CREATE TABLE Loans (

LoanID NUMBER PRIMARY KEY,

CustomerID NUMBER,

LoanAmount NUMBER,

InterestRate NUMBER,

StartDate DATE,

EndDate DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

**Step 2: Insert Sample Data**

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

VALUES (1, 'John Doe', TO\_DATE('1950-05-15', 'YYYY-MM-DD'), 12000, SYSDATE);

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

VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 8000, SYSDATE);

INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)

VALUES (1, 1, 5000, 9.0, SYSDATE, ADD\_MONTHS(SYSDATE, 1));

INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)

VALUES (2, 2, 3000, 7.5, SYSDATE, ADD\_MONTHS(SYSDATE, 45));

**Scenario 1: Apply 1% Discount for Customers Above 60**

**Code:**

SET SERVEROUTPUT ON;

DECLARE

v\_age NUMBER;

BEGIN

FOR rec IN (

SELECT c.CustomerID, c.DOB, l.LoanID, l.InterestRate

FROM Customers c

JOIN Loans l ON c.CustomerID = l.CustomerID

) LOOP

v\_age := FLOOR(MONTHS\_BETWEEN(SYSDATE, rec.DOB) / 12);

IF v\_age > 60 THEN

UPDATE Loans

SET InterestRate = rec.InterestRate - 1

WHERE LoanID = rec.LoanID;

DBMS\_OUTPUT.PUT\_LINE('1% discount applied to LoanID ' || rec.LoanID);

END IF;

END LOOP;

END;

/

**OUTPUT SCREENSHOT:**

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario 2: Mark Customers with Balance > 10000 as VIP**

Since there's no IsVIP column in Customers, we will add it first:

ALTER TABLE Customers ADD IsVIP VARCHAR2(5);

**Code:**

BEGIN

FOR rec IN (

SELECT CustomerID, Balance FROM Customers

) LOOP

IF rec.Balance > 10000 THEN

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Customer ' || rec.CustomerID || ' marked as VIP.');

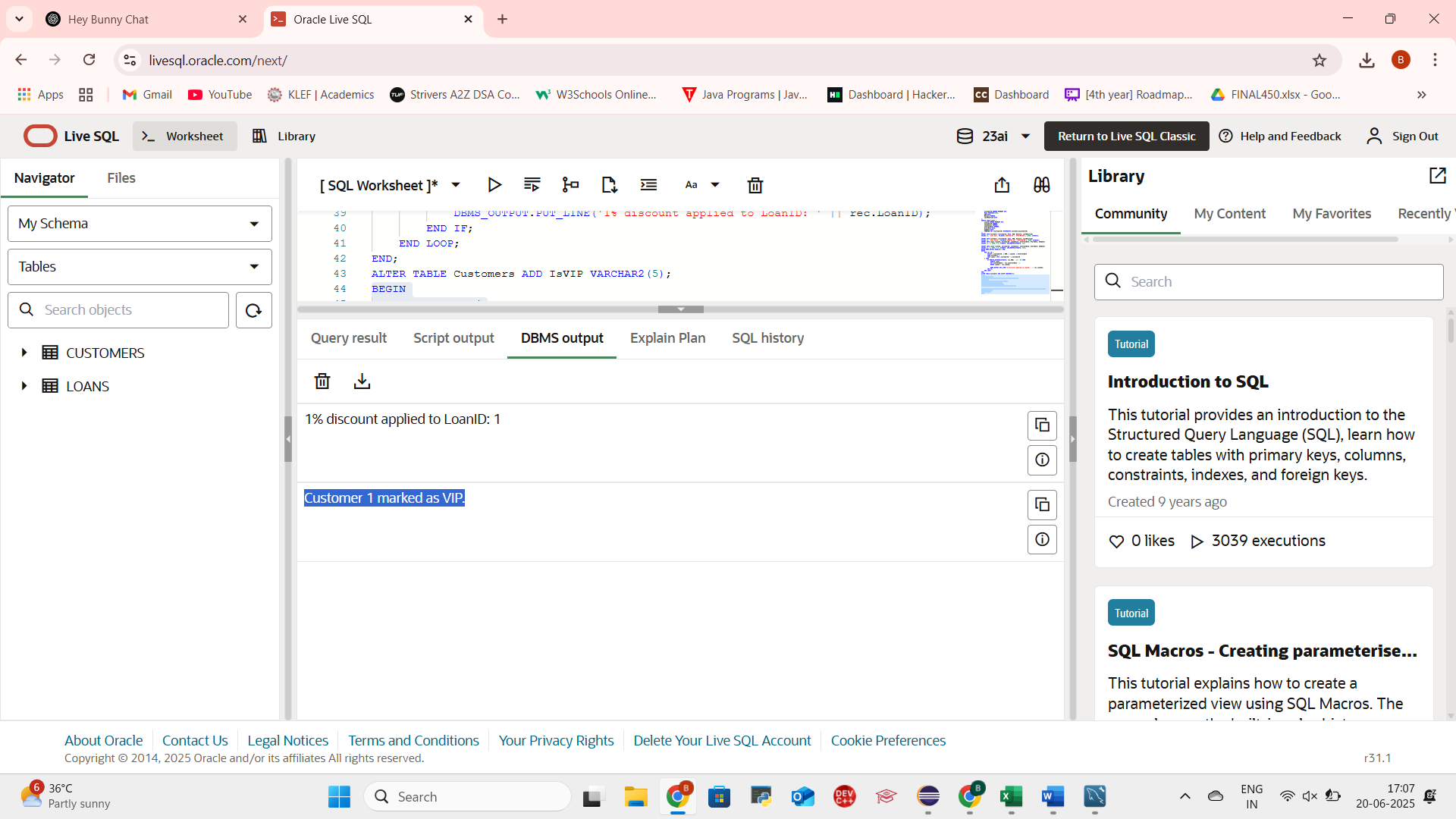
END IF;

END LOOP;

END;

/

**OUTPUT SCREENSHOT:**



**Scenario 3: Reminders for Loans Due in Next 30 Days**

**CODE:**

BEGIN

FOR rec IN (

SELECT l.LoanID, c.Name, l.EndDate

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ID ' || rec.LoanID || ' for ' || rec.Name ||

' is due on ' || TO\_CHAR(rec.EndDate, 'DD-MON-YYYY'));

END LOOP;

END;

/

**OUTPUT SCREENSHOT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Exercise 3: Stored Procedures**

We'll use these tables:

* Accounts — for Scenario 1 & 3
* Employees — for Scenario 2

**Step 1: Create Tables**

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20),

Balance NUMBER,

LastModified 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

);

**Step 2: Insert Sample Data**

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)

VALUES (1, 1, 'Savings', 10000, SYSDATE);

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)

VALUES (2, 2, 'Checking', 7000, SYSDATE);

INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)

VALUES (1, 'Alice Johnson', 'Manager', 60000, 'HR', TO\_DATE('2015-06-15', 'YYYY-MM-DD'));

INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)

VALUES (2, 'Bob Brown', 'Developer', 50000, 'IT', TO\_DATE('2017-03-20', 'YYYY-MM-DD'));

**Scenario 1: ProcessMonthlyInterest – Add 1% Interest to Savings Accounts**

**Code:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

FOR acc IN (

SELECT AccountID, Balance

FROM Accounts

WHERE AccountType = 'Savings'

) LOOP

UPDATE Accounts

SET Balance = acc.Balance \* 1.01,

LastModified = SYSDATE

WHERE AccountID = acc.AccountID;

DBMS\_OUTPUT.PUT\_LINE('Interest applied to AccountID: ' || acc.AccountID);

END LOOP;

END;

/

**Call the Procedure:**

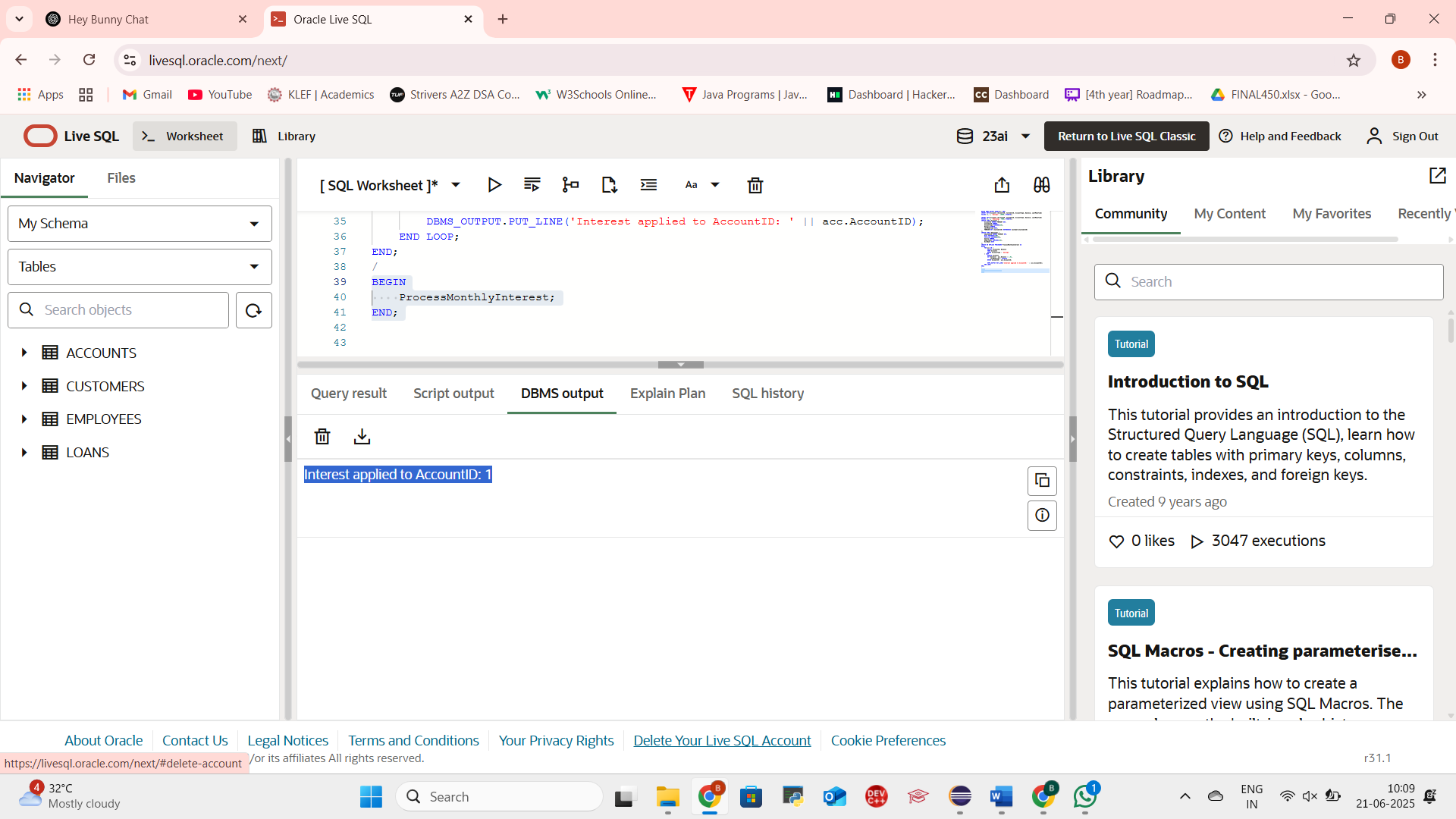
BEGIN

DBMS\_OUTPUT.ENABLE;

ProcessMonthlyInterest;

END;

**OUTPUT SCREENSHOT:**

****

**Scenario 2: UpdateEmployeeBonus – Bonus by Dept and Percentage**

**CODE:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

deptName IN VARCHAR2,

bonusPercent IN NUMBER

) AS

BEGIN

FOR emp IN (

SELECT EmployeeID, Salary

FROM Employees

WHERE Department = deptName

) LOOP

UPDATE Employees

SET Salary = emp.Salary + (emp.Salary \* bonusPercent / 100)

WHERE EmployeeID = emp.EmployeeID;

DBMS\_OUTPUT.PUT\_LINE('Bonus added to EmployeeID: ' || emp.EmployeeID);

END LOOP;

END;

/

**Call the Procedure (example: 10% bonus to IT):**

BEGIN

DBMS\_OUTPUT.ENABLE;

UpdateEmployeeBonus('IT', 10);

END;

**OUTPUT SCREENSHOT:**

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario 3: TransferFunds – From One Account to Another**

**CODE:**

CREATE OR REPLACE PROCEDURE TransferFunds (

fromAccID IN NUMBER,

toAccID IN NUMBER,

amount IN NUMBER

) AS

v\_balance NUMBER;

BEGIN

-- Check balance of source account

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = fromAccID;

IF v\_balance >= amount THEN

-- Deduct from source

UPDATE Accounts

SET Balance = Balance - amount,

LastModified = SYSDATE

WHERE AccountID = fromAccID;

-- Add to destination

UPDATE Accounts

SET Balance = Balance + amount,

LastModified = SYSDATE

WHERE AccountID = toAccID;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || amount || ' from AccountID ' || fromAccID || ' to AccountID ' || toAccID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insufficient funds in AccountID ' || fromAccID);

END IF;

END;

/

**Call the Procedure (example):**

BEGIN

DBMS\_OUTPUT.ENABLE;

TransferFunds(1, 2, 200);

END;

**OUTPUT SCREENSHOT:**

