**Name:Anisha M**

**Superset ID:6371860**

**WEEK-2 HANDS-ON SOLUTION**

**PLSQL PROGRAMMING**

**Exercise 1: Control Structures**

**Code:**

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Loans';

EXECUTE IMMEDIATE 'DROP TABLE Customers';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

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)

);

BEGIN

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1950-01-01', 'YYYY-MM-DD'), 20000, SYSDATE);

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1995-05-01', 'YYYY-MM-DD'), 9000, SYSDATE);

INSERT INTO Customers VALUES (3, 'Alex Grey', TO\_DATE('1960-06-15', 'YYYY-MM-DD'), 15000, SYSDATE);

INSERT INTO Loans VALUES (101, 1, 50000, 9.0, SYSDATE, ADD\_MONTHS(SYSDATE, 10));

INSERT INTO Loans VALUES (102, 2, 30000, 11.0, SYSDATE, ADD\_MONTHS(SYSDATE, 40));

INSERT INTO Loans VALUES (103, 3, 45000, 10.0, SYSDATE, ADD\_MONTHS(SYSDATE, 25));

COMMIT;

END;

/

BEGIN

FOR cust\_rec IN (

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

FROM Customers c

JOIN Loans l ON c.CustomerID = l.CustomerID

) LOOP

IF MONTHS\_BETWEEN(SYSDATE, cust\_rec.DOB)/12 > 60 THEN

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE LoanID = cust\_rec.LoanID;

DBMS\_OUTPUT.PUT\_LINE('Discount applied to Loan ID ' || cust\_rec.LoanID);

END IF;

END LOOP;

END;

/

BEGIN

FOR cust IN (SELECT CustomerID, Balance FROM Customers) LOOP

IF cust.Balance > 10000 THEN

DBMS\_OUTPUT.PUT\_LINE('Customer ' || cust.CustomerID || ' is eligible for VIP status');

END IF;

END LOOP;

END;

/

BEGIN

FOR rec IN (

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

FROM Loans l JOIN Customers c ON c.CustomerID = l.CustomerID

WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ' || rec.LoanID || ' due on ' || TO\_CHAR(rec.EndDate, 'YYYY-MM-DD') || ' for ' || rec.Name);

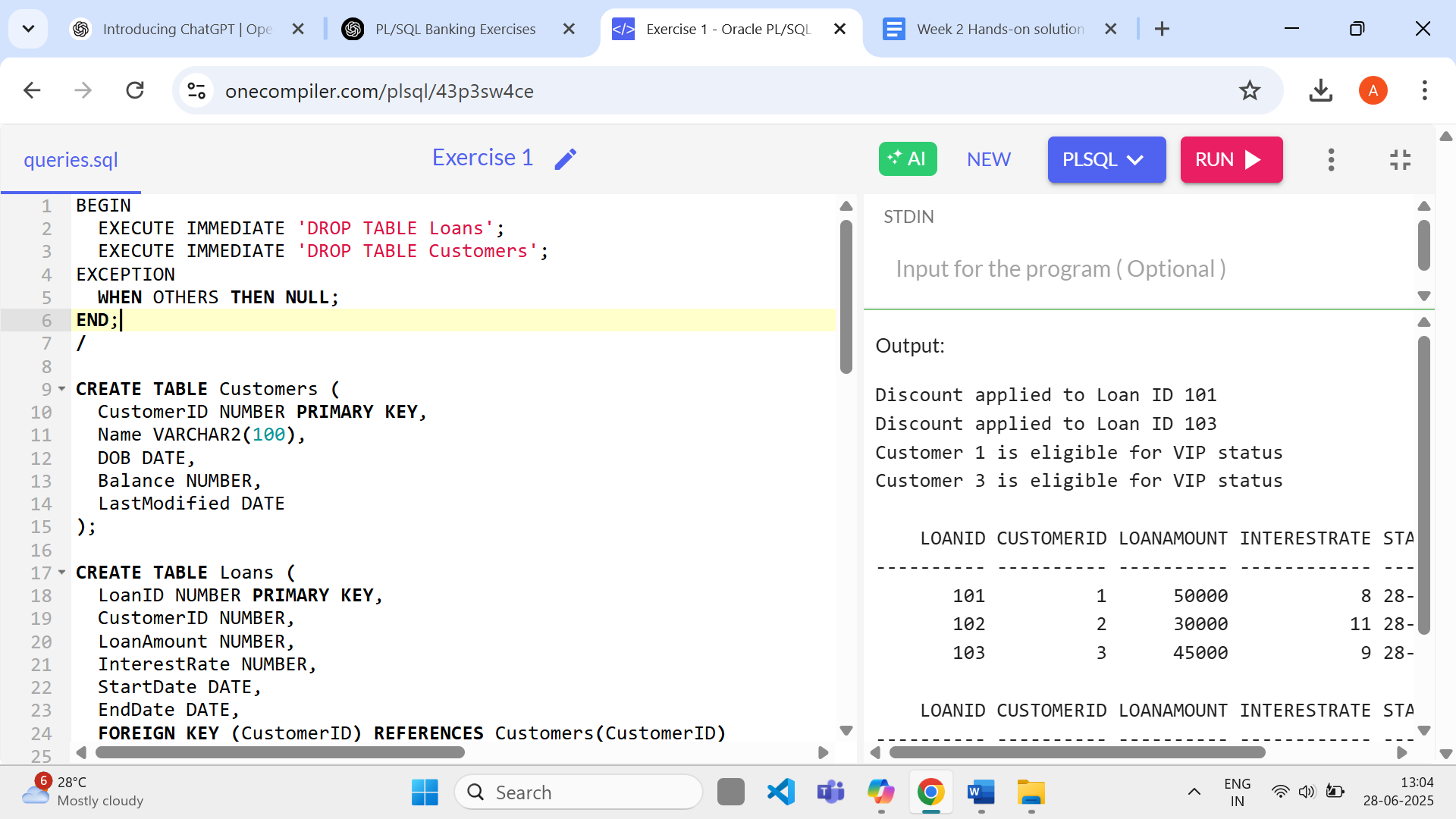
END LOOP;

END;

/

SELECT \* FROM Loans;

**Output:**

****

**Exercise 2: Error Handling**

**Code:**

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

);

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

BEGIN

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

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

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

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

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', SYSDATE);

COMMIT;

END;

/

DECLARE

v\_from\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_from\_balance FROM Accounts WHERE AccountID = 1;

IF v\_from\_balance < 1000 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds');

ELSE

UPDATE Accounts SET Balance = Balance - 1000 WHERE AccountID = 1;

UPDATE Accounts SET Balance = Balance + 1000 WHERE AccountID = 2;

DBMS\_OUTPUT.PUT\_LINE('Funds transferred successfully.');

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END;

/

DECLARE

v\_rows NUMBER := 0;

BEGIN

UPDATE Employees SET Salary = Salary \* 1.1 WHERE EmployeeID = 999;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Employee ID not found.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Salary updated successfully.');

END IF;

END;

/

BEGIN

INSERT INTO Customers VALUES (1, 'Duplicate', TO\_DATE('1990-01-01','YYYY-MM-DD'), 4000, SYSDATE);

EXCEPTION

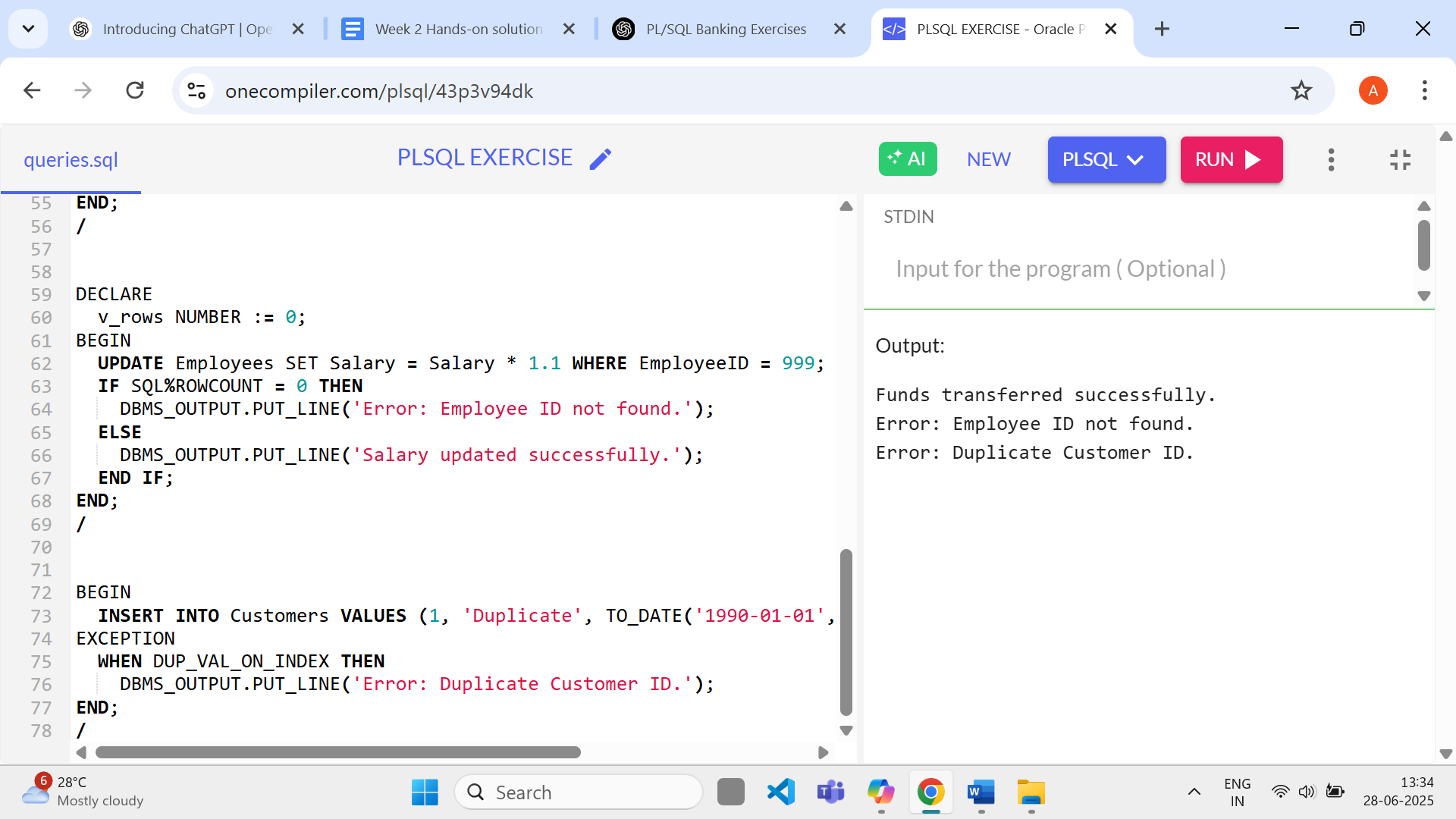
WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Duplicate Customer ID.');

END;

/

**Output:**



**Exercise 3: Stored Procedures**

**Code:**

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

);

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

BEGIN

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

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

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

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

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', SYSDATE);

COMMIT;

END;

/

BEGIN

UPDATE Accounts SET Balance = Balance + (Balance \* 0.01) WHERE AccountType = 'Savings';

DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed.');

END;

/

BEGIN

UPDATE Employees SET Salary = Salary + (Salary \* 0.1) WHERE Department = 'IT';

DBMS\_OUTPUT.PUT\_LINE('Bonus added for IT department.');

END;

/

DECLARE

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = 1;

IF v\_balance >= 500 THEN

UPDATE Accounts SET Balance = Balance - 500 WHERE AccountID = 1;

UPDATE Accounts SET Balance = Balance + 500 WHERE AccountID = 2;

DBMS\_OUTPUT.PUT\_LINE('Transfer successful.');

ELSE

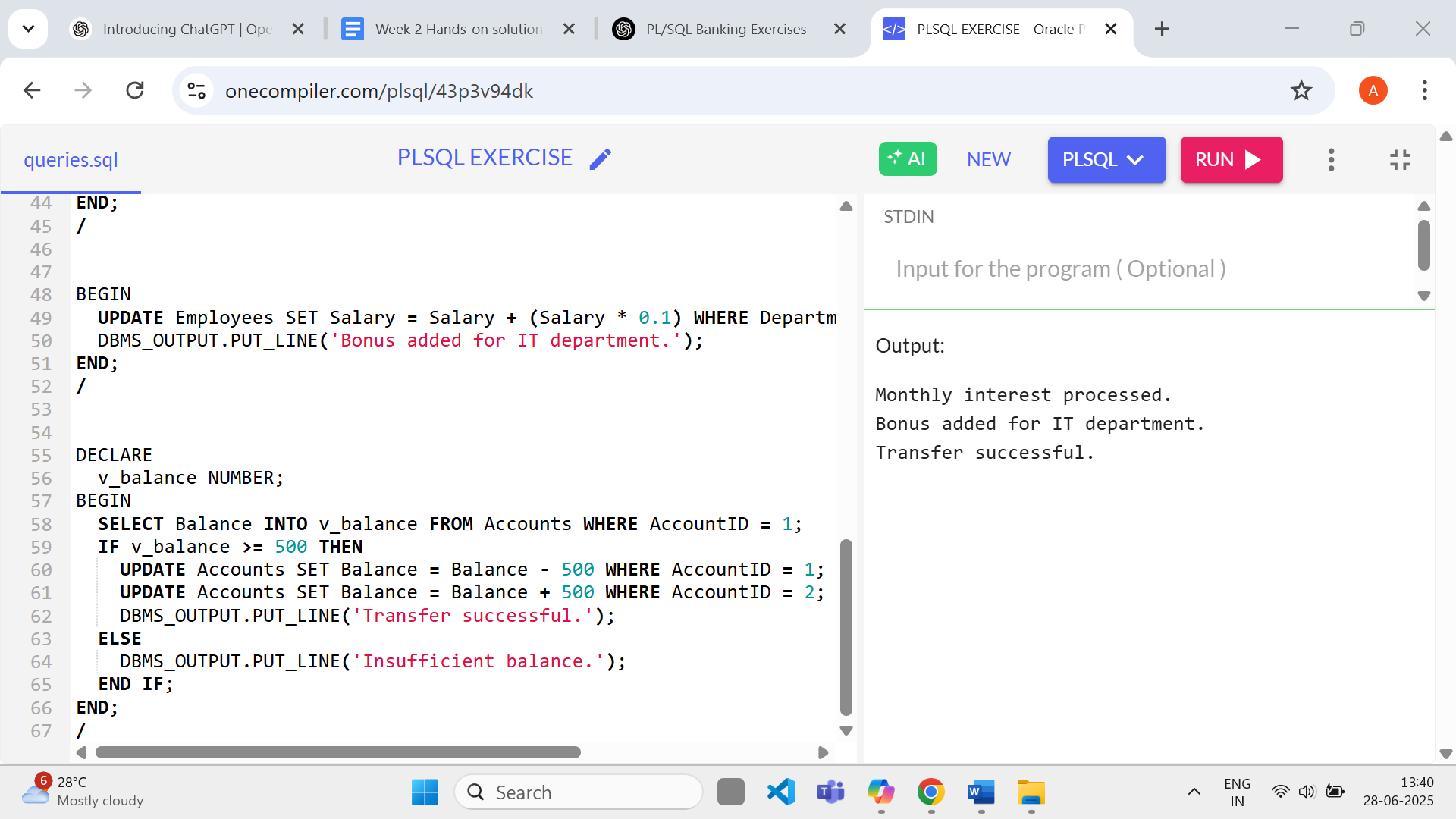
DBMS\_OUTPUT.PUT\_LINE('Insufficient balance.');

END IF;

END;

/

**Output:**

****

**Exercise 4: Functions**

**Code:**

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

);

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

BEGIN

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

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

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

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

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', SYSDATE);

COMMIT;

END;

/

DECLARE

dob DATE := TO\_DATE('1980-01-01', 'YYYY-MM-DD');

age NUMBER;

BEGIN

age := FLOOR(MONTHS\_BETWEEN(SYSDATE, dob)/12);

DBMS\_OUTPUT.PUT\_LINE('Age: ' || age);

END;

/

DECLARE

loan\_amt NUMBER := 10000;

rate NUMBER := 10;

years NUMBER := 2;

monthly\_rate NUMBER := rate / 12 / 100;

months NUMBER := years \* 12;

emi NUMBER;

BEGIN

emi := ROUND(loan\_amt \* monthly\_rate / (1 - POWER(1 + monthly\_rate, -months)), 2);

DBMS\_OUTPUT.PUT\_LINE('Monthly EMI: ' || emi);

END;

/

DECLARE

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = 1;

IF v\_balance >= 300 THEN

DBMS\_OUTPUT.PUT\_LINE('Sufficient balance');

ELSE

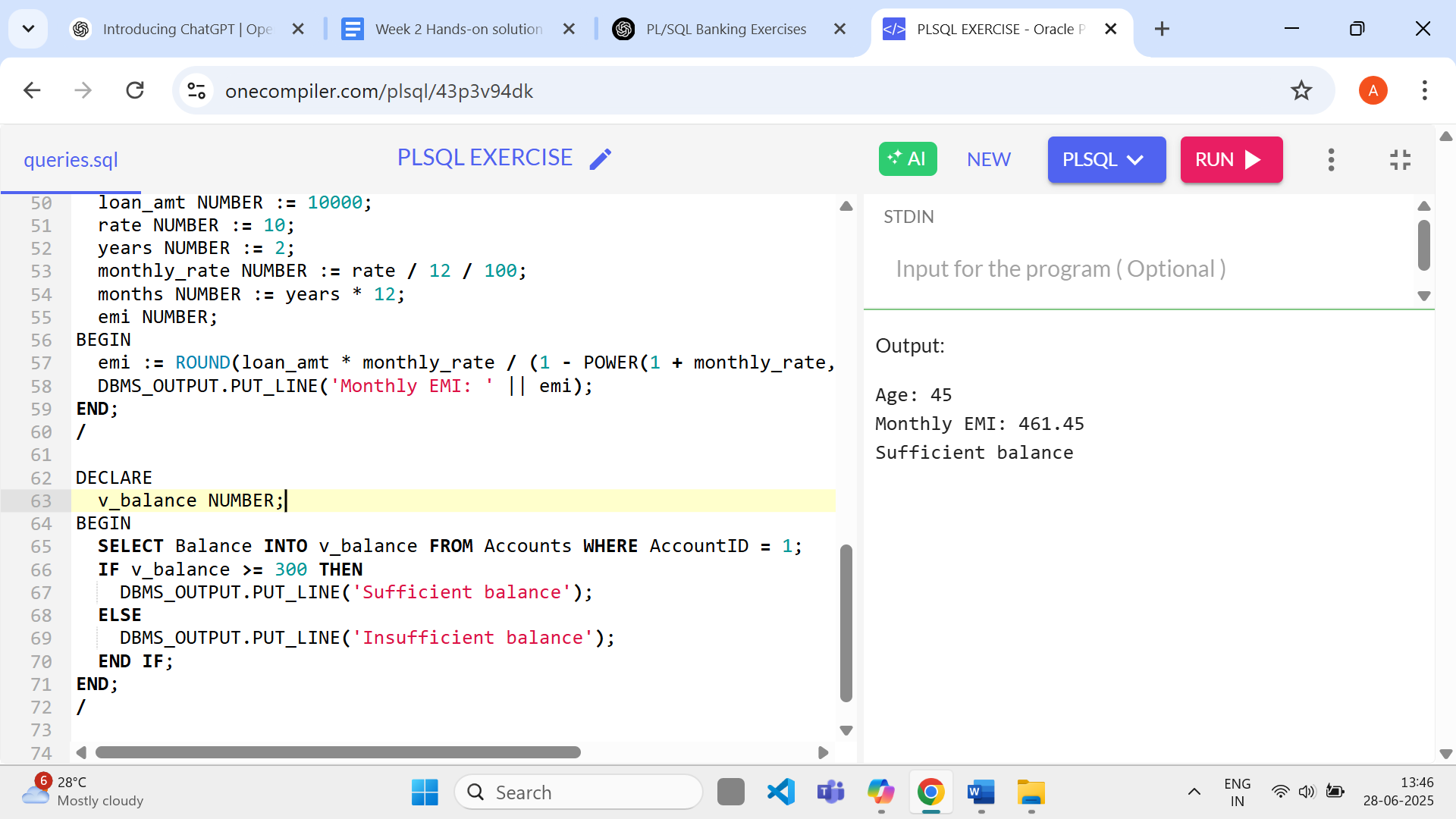
DBMS\_OUTPUT.PUT\_LINE('Insufficient balance');

END IF;

END;

/

**Output:**

****

**Exercise 5:Triggers**

**Code:**

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 Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

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 Transactions (

TransactionID NUMBER PRIMARY KEY,

AccountID NUMBER,

TransactionDate DATE,

Amount NUMBER,

TransactionType VARCHAR2(10),

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

BEGIN

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

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

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

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

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', SYSDATE);

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

INSERT INTO Transactions VALUES (1, 1, SYSDATE, 1000, 'Deposit');

INSERT INTO Transactions VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

COMMIT;

END;

/

BEGIN

UPDATE Customers SET LastModified = SYSDATE WHERE CustomerID = 1;

DBMS\_OUTPUT.PUT\_LINE('LastModified updated.');

END;

/

BEGIN

INSERT INTO Transactions VALUES (3, 1, SYSDATE, 1000, 'Deposit');

DBMS\_OUTPUT.PUT\_LINE('Transaction logged.');

END;

/

DECLARE

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = 1;

IF -200 > v\_balance THEN

DBMS\_OUTPUT.PUT\_LINE('Withdrawal denied.');

ELSE

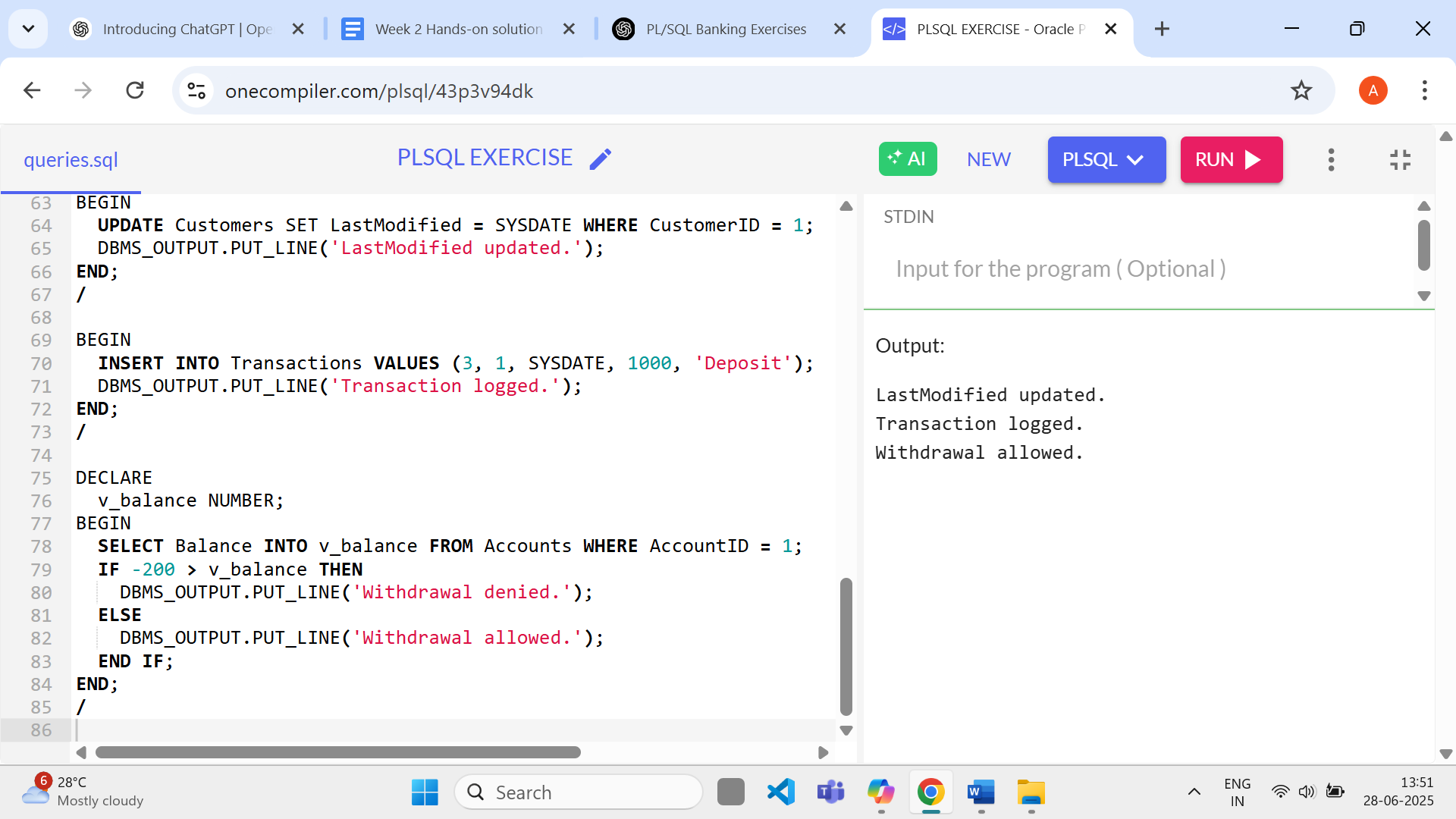
DBMS\_OUTPUT.PUT\_LINE('Withdrawal allowed.');

END IF;

END;

/

**Output:**

****

**Exercise 6: Cursors**

**Code:**

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 Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

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 Transactions (

TransactionID NUMBER PRIMARY KEY,

AccountID NUMBER,

TransactionDate DATE,

Amount NUMBER,

TransactionType VARCHAR2(10),

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

BEGIN

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

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

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

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

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', SYSDATE);

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

INSERT INTO Transactions VALUES (1, 1, SYSDATE, 1000, 'Deposit');

INSERT INTO Transactions VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

COMMIT;

END;

/

DECLARE

CURSOR c1 IS SELECT \* FROM Transactions WHERE TO\_CHAR(TransactionDate, 'MMYYYY') = TO\_CHAR(SYSDATE, 'MMYYYY');

BEGIN

FOR tx IN c1 LOOP

DBMS\_OUTPUT.PUT\_LINE('Transaction ID ' || tx.TransactionID || ' on Account ' || tx.AccountID);

END LOOP;

END;

/

DECLARE

CURSOR acc\_cur IS SELECT AccountID FROM Accounts;

BEGIN

FOR acc IN acc\_cur LOOP

UPDATE Accounts SET Balance = Balance - 100 WHERE AccountID = acc.AccountID;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Annual fee applied.');

END;

/

DECLARE

CURSOR c IS SELECT LoanID, InterestRate FROM Loans;

BEGIN

FOR l IN c LOOP

UPDATE Loans SET InterestRate = InterestRate + 0.5 WHERE LoanID = l.LoanID;

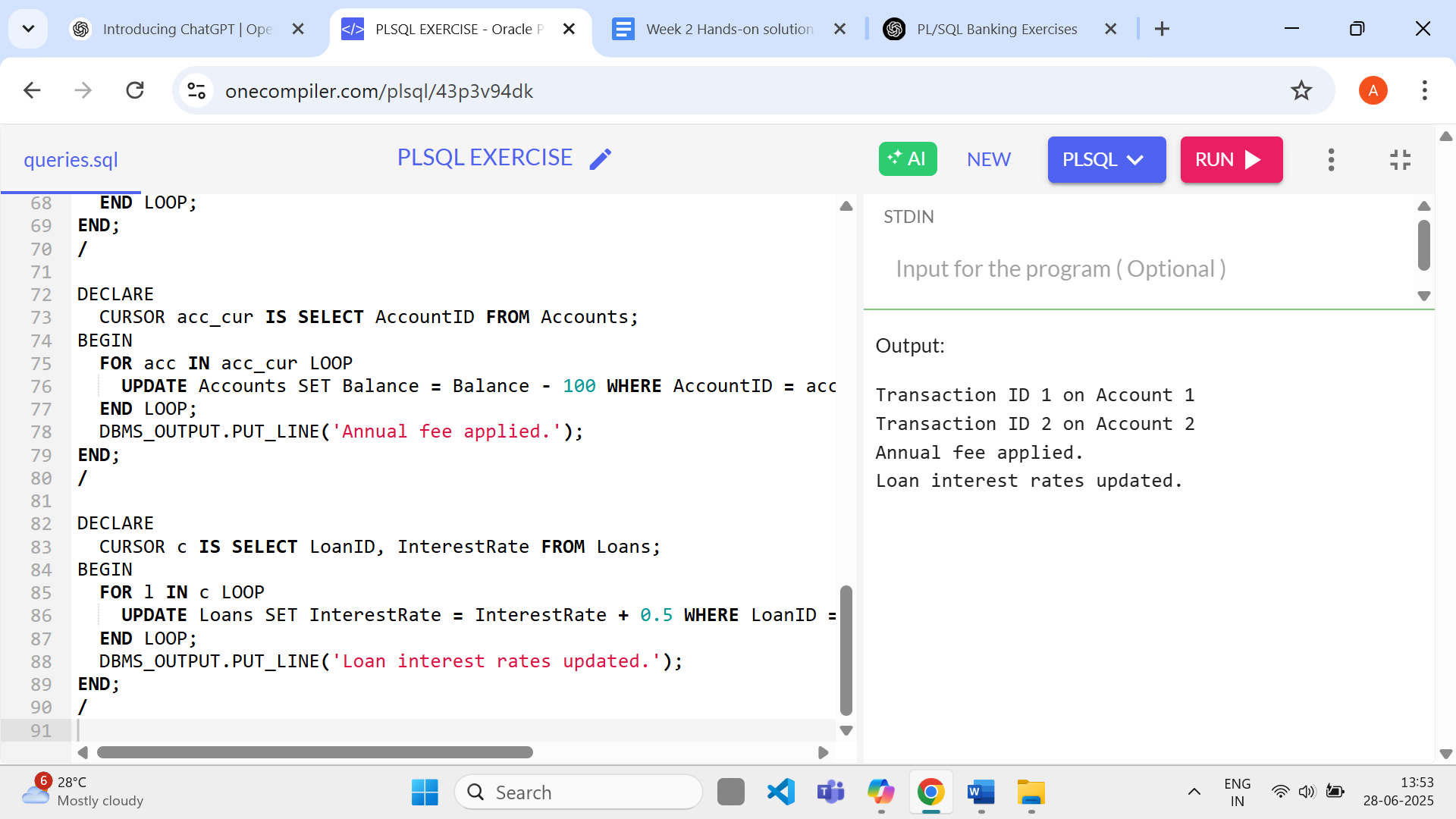
END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Loan interest rates updated.');

END;

/

**Output:**

****

**Exercise 7: Packages**

**Code:**

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 Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

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 Transactions (

TransactionID NUMBER PRIMARY KEY,

AccountID NUMBER,

TransactionDate DATE,

Amount NUMBER,

TransactionType VARCHAR2(10),

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

BEGIN

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

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

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

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

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', SYSDATE);

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', SYSDATE);

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

INSERT INTO Transactions VALUES (1, 1, SYSDATE, 1000, 'Deposit');

INSERT INTO Transactions VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

COMMIT;

END;

/

BEGIN

INSERT INTO Customers VALUES (5, 'Customer Five', TO\_DATE('2000-05-05', 'YYYY-MM-DD'), 6000, SYSDATE);

UPDATE Customers SET Balance = 8000 WHERE CustomerID = 5;

SELECT Balance INTO :BAL FROM Customers WHERE CustomerID = 5;

END;

/

BEGIN

INSERT INTO Employees VALUES (3, 'Chris Green', 'Analyst', 50000, 'Finance', SYSDATE);

UPDATE Employees SET Salary = 55000 WHERE EmployeeID = 3;

SELECT Salary \* 12 INTO :SAL FROM Employees WHERE EmployeeID = 3;

END;

/

BEGIN

INSERT INTO Accounts VALUES (3, 5, 'Savings', 3000, SYSDATE);

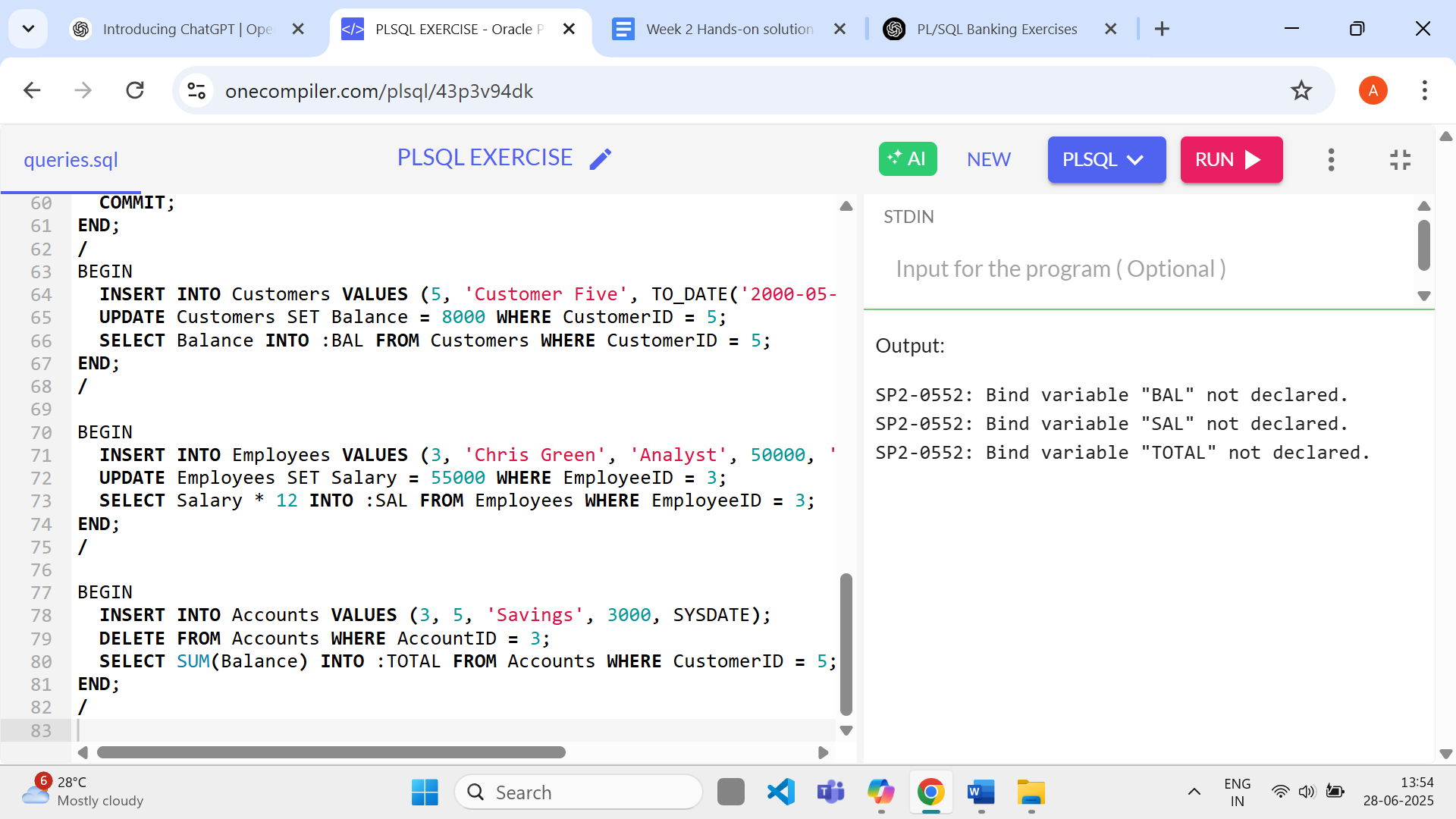
DELETE FROM Accounts WHERE AccountID = 3;

SELECT SUM(Balance) INTO :TOTAL FROM Accounts WHERE CustomerID = 5;

END;

/

**Output:**

****