**PL/SQL Programming**

**Mandatory**

**Exercise 1: Control Structures**

**Create table Customer**

CREATE TABLE CUSTOMERS (

CUSTOMERID NUMBER PRIMARY KEY,

NAME VARCHAR2(100),

DOB DATE,

BALANCE NUMBER,

LASTMODIFIED DATE

);

**Create table Loans**

CREATE TABLE LOANS (

LOANID NUMBER PRIMARY KEY,

CUSTOMERID NUMBER,

LOANAMOUNT NUMBER,

INTERESTRATE NUMBER,

STARTDATE DATE,

ENDDATE DATE,

FOREIGN KEY ( CUSTOMERID )

REFERENCES CUSTOMERS ( CUSTOMERID )

);

**Insert into Customers val1**

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 val2**

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

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

**Insert into Loans val1**

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

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

**Scenario 1:** The bank wants to apply a discount to loan interest rates for customers above 60 years old.

**Question:** Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

DECLARE

CURSOR cur\_customer IS

SELECT customerid, EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM dob) AS age

FROM customers;

v\_id customers.customerid%TYPE;

v\_age NUMBER;

BEGIN

OPEN cur\_customer;

LOOP

FETCH cur\_customer INTO v\_id, v\_age;

EXIT WHEN cur\_customer%NOTFOUND;

IF v\_age > 60 THEN

UPDATE loans

SET interestrate = interestrate - 1

WHERE customerid = v\_id;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_id || ' | Age: ' || v\_age);

DBMS\_OUTPUT.PUT\_LINE('Customer not eligible for interest discount');

END IF;

END LOOP;

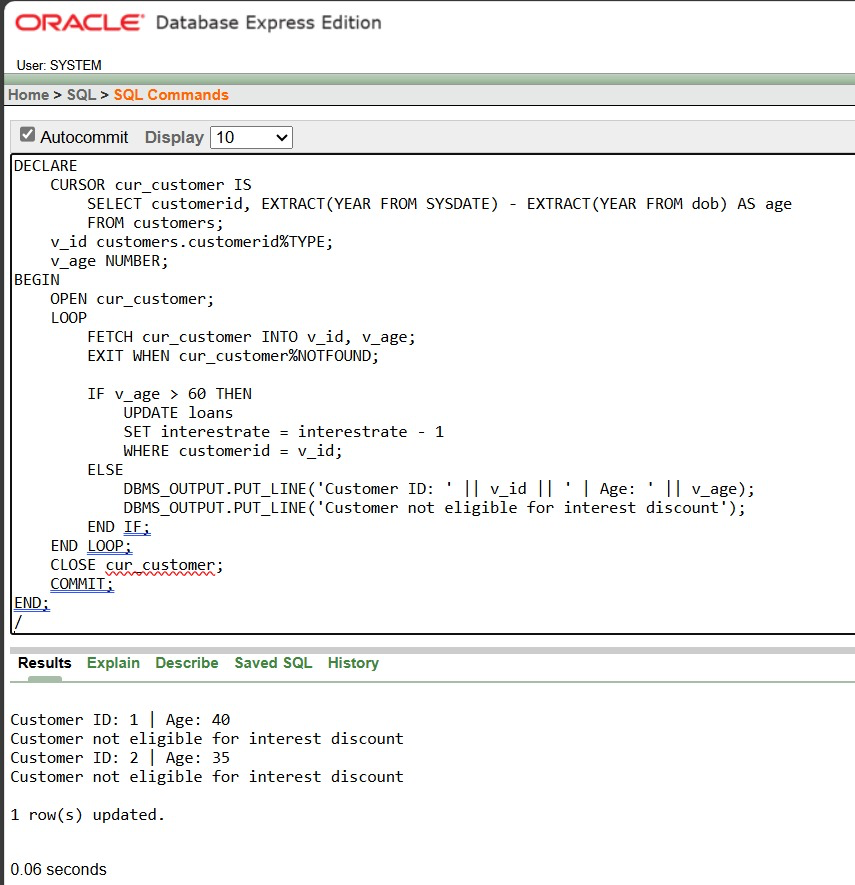
CLOSE cur\_customer;

COMMIT;

END;

/

**Output:**

****

**Scenario 2:** A customer can be promoted to VIP status based on their balance.

**Question:** Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.

ALTER TABLE customers ADD isvip VARCHAR2(10);

BEGIN

FOR user\_rec IN (

SELECT customerid, balance FROM customers

) LOOP

IF user\_rec.balance > 10000 THEN

UPDATE customers

SET isvip = 'TRUE'

WHERE customerid = user\_rec.customerid;

ELSE

UPDATE customers

SET isvip = 'FALSE'

WHERE customerid = user\_rec.customerid;

END IF;

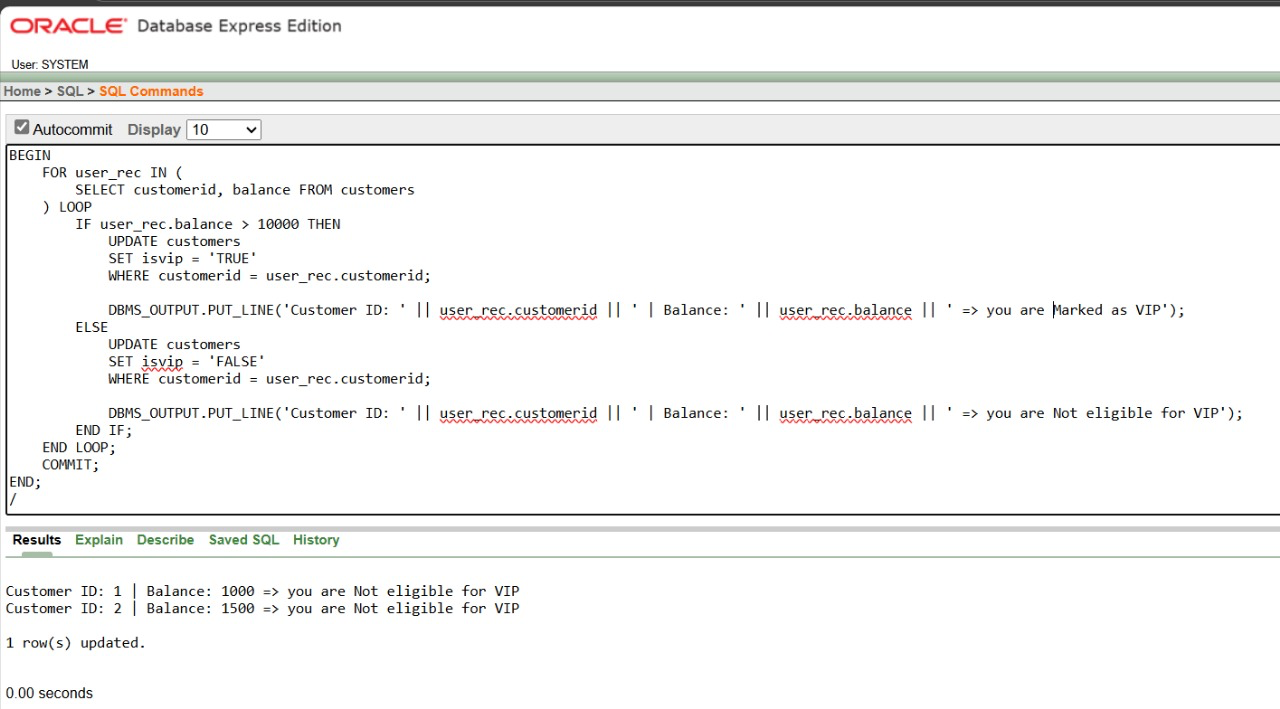
END LOOP;

COMMIT;

END;

/

**Output :**

****

**Scenario 3:** The bank wants to send reminders to customers whose loans are due within the next 30 days.

**Question:** Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

SET SERVEROUTPUT ON;

DECLARE

CURSOR cur\_loans IS

SELECT l.loanid, l.customerid, c.name, l.enddate

FROM loans l

JOIN customers c ON c.customerid = l.customerid

WHERE l.enddate BETWEEN SYSDATE AND SYSDATE + 30;

v\_loanid loans.loanid%TYPE;

v\_custid loans.customerid%TYPE;

v\_name customers.name%TYPE;

v\_due loans.enddate%TYPE;

found BOOLEAN := FALSE;

BEGIN

OPEN cur\_loans;

LOOP

FETCH cur\_loans INTO v\_loanid, v\_custid, v\_name, v\_due;

EXIT WHEN cur\_loans%NOTFOUND;

found := TRUE;

DBMS\_OUTPUT.PUT\_LINE(

'Reminder: Loan ' || v\_loanid ||

' for ' || v\_name ||

' (ID: ' || v\_custid ||

') is due on ' || TO\_CHAR(v\_due, 'DD-Mon-YYYY')

);

END LOOP;

CLOSE cur\_loans;

IF NOT found THEN

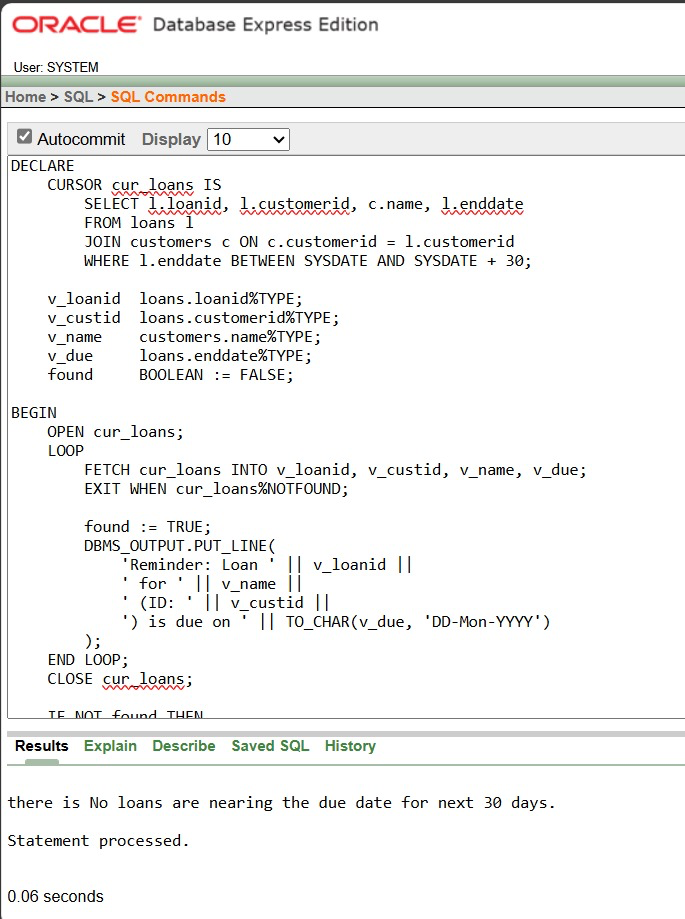
DBMS\_OUTPUT.PUT\_LINE('No loans are nearing due date in next 30 days.');

END IF;

END;

/

**Output :**

****