

# WEEK 2 - PL/SQL Programming

## Exercise 1: Control Structures

### TABLE CREATION:

#### **CUSTOMERS**

```
CREATE TABLE customers (  
  customer_id NUMBER PRIMARY KEY,  
  name        VARCHAR2(100),  
  age         NUMBER,  
  balance     NUMBER(10,2),  
  is_vip      VARCHAR2(5)  
);
```

#### **LOANS**

```
CREATE TABLE loans (  
  loan_id      NUMBER PRIMARY KEY,  
  customer_id  NUMBER,  
  interest_rate NUMBER(5,2),  
  due_date     DATE,  
  FOREIGN KEY (customer_id) REFERENCES customers(customer_id)  
);
```

### DATA INSERTION:

#### **CUSTOMERS**

```
INSERT INTO Customers VALUES (1, 'Alice', 65, 12000, 'FALSE');  
INSERT INTO Customers VALUES (2, 'Bob', 58, 8000, 'FALSE');  
INSERT INTO Customers VALUES (3, 'Charlie', 70, 20000, 'FALSE');
```

#### **LOANS**

```
INSERT INTO Loans VALUES (101, 1, 5.5, SYSDATE + 20);  
INSERT INTO Loans VALUES (102, 2, 6.0, SYSDATE + 40);  
INSERT INTO Loans VALUES (103, 3, 7.0, SYSDATE + 10);
```

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Learn Development', and 'Gallery'. The 'SQL Commands' tab is active, displaying a list of commands. The first command is 'select \* from Loans;'. The 'Run' button is highlighted. Below the command list, the 'Results' tab is active, showing a table with the following data:

LOAN_ID	CUSTOMER_ID	INTEREST_RATE	DUE_DATE
101	1	4.5	7/17/2025
103	3	6	7/7/2025
102	2	6	8/6/2025

The footer of the interface includes the text 'Oracle APEX 24.2.9' and 'Copyright © 1996, 2024, Oracle and/or its affiliates'.

## 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.

## CODE:

BEGIN

FOR cust\_rec IN (SELECT customer\_id, name FROM customers WHERE age > 60) LOOP

    UPDATE loans

    SET interest\_rate = interest\_rate - 1

    WHERE customer\_id = cust\_rec.customer\_id;

    DBMS\_OUTPUT.PUT\_LINE('Updated loan interest for ' || cust\_rec.name);

END LOOP;

COMMIT;

END;

## OUTPUT:

The screenshot displays the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' tab is active, showing a PL/SQL block with the following code:

```
1 BEGIN
2   FOR cust_rec IN (SELECT customer_id, name FROM customers WHERE age > 60) LOOP
3     UPDATE loans
4     SET interest_rate = interest_rate - 1
5     WHERE customer_id = cust_rec.customer_id;
6
7     DBMS_OUTPUT.PUT_LINE('Updated loan interest for ' || cust_rec.name);
8   END LOOP;
9   COMMIT;
10 END;
```

The 'Results' tab at the bottom shows the output of the execution:

```
Updated loan interest for Alice
Updated loan interest for Charlie
1 row(s) updated.
0.01seconds
```

The footer of the interface includes the user 'jothikathangam@gmail.com', the schema 'pl\_training', and the version 'Oracle APEX 24.2.6'.

## 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.

## CODE:

BEGIN

FOR rec IN (SELECT CustomerID FROM Customers WHERE Balance > 10000)

LOOP

UPDATE Customers

SET IsVIP = 'TRUE' -- if it's a VARCHAR2 column; if BOOLEAN, use TRUE

WHERE CustomerID = rec.CustomerID;

END LOOP;

COMMIT;

END;

## OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' tab is active, showing a PL/SQL block that iterates through all customers and sets the 'IsVIP' flag to 'TRUE' for those with a balance greater than 10,000. The 'Run' button is highlighted in green. Below the command editor, the 'Results' tab is active, displaying a table with the following data:

CUSTOMER_ID	NAME	AGE	BALANCE	IS_VIP
2	Bob	58	8000	FALSE
1	Alice	65	12000	TRUE
3	Charlie	70	20000	TRUE

The footer of the interface shows the user 'T Jothika Thangam' and the version 'Oracle APEX 24.2.4'.

### 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.

### CODE:

```
BEGIN
  FOR loan_rec IN (
    SELECT l.loan_id, c.name, l.due_date
    FROM loans l
    JOIN customers c ON l.customer_id = c.customer_id
    WHERE l.due_date BETWEEN SYSDATE AND SYSDATE + 30
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Reminder: ' || loan_rec.name ||
      ', your loan (ID: ' || loan_rec.loan_id ||
      ') is due on ' || TO_CHAR(loan_rec.due_date, 'DD-MON-
YYYYY'));
  END LOOP;
END;
```

### OUTPUT:

The screenshot displays the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' window is active, showing the PL/SQL code. The 'Results' window at the bottom shows the output of the execution. The code is as follows:

```
1 BEGIN
2   FOR loan_rec IN (
3     SELECT l.loan_id, c.name, l.due_date
4     FROM loans l
5     JOIN customers c ON l.customer_id = c.customer_id
6     WHERE l.due_date BETWEEN SYSDATE AND SYSDATE + 30
7   ) LOOP
8     DBMS_OUTPUT.PUT_LINE('Reminder: ' || loan_rec.name ||
9       ', your loan (ID: ' || loan_rec.loan_id ||
10      ') is due on ' || TO_CHAR(loan_rec.due_date, 'DD-MON-YYYYY'));
11   END LOOP;
12 END;
```

The Results window shows the following output:

```
Reminder: Alice, your loan (ID: 101) is due on 17-JUL-2025
Reminder: Charlie, your loan (ID: 103) is due on 07-JUL-2025
Statement processed.
0.01seconds
```

