

## Exercise 1: Control Structures

**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 rec IN (

    SELECT l.loan_id

    FROM customers c

    JOIN loans l ON c.customer_id = l.customer_id

    WHERE c.age > 60

) LOOP

    UPDATE loans

    SET interest_rate = interest_rate - 1

    WHERE loan_id = rec.loan_id;

END LOOP;

COMMIT;

END;

/
```

### Output:



```
24 BEGIN
25     FOR rec IN (
26         SELECT l.loan_id
27         FROM customers c
28         JOIN loans l ON c.customer_id = l.customer_id
29         WHERE c.age > 60
30     ) LOOP
31         UPDATE loans
32         SET interest_rate = interest_rate - 1
33         WHERE loan_id = rec.loan_id;
34     END LOOP;
35
36     COMMIT;
37 END;
38 /
```

Query result   **Script output**   DBMS output   Explain Plan   SQL history

SQL> BEGIN  
      FOR rec IN (  
          SELECT l.loan\_id  
          FROM customers c...  
Show more...

PL/SQL procedure successfully completed.

**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 customer_id

    FROM customer

    WHERE balance > 10000

) LOOP

    UPDATE customer

    SET IsVIP = 'TRUE'

    WHERE customer_id = rec.customer_id;

END LOOP;

COMMIT;

END;

/
```

**Output:**

The screenshot displays a SQL Worksheet interface. The top toolbar includes icons for running queries, saving, and other standard database tools. The main text area contains the following PL/SQL code:

```
17 BEGIN
18   FOR rec IN (
19     SELECT customer_id
20     FROM customer
21     WHERE balance > 10000
22   ) LOOP
23     UPDATE customer
24     SET IsVIP = 'TRUE'
25     WHERE customer_id = rec.customer_id;
26   END LOOP;
27
28   COMMIT;
29 END;
30 /
31
```

Below the code editor, there are tabs for 'Query result', 'Script output', 'DBMS output', 'Explain Plan', and 'SQL history'. The 'Script output' tab is currently selected, showing the execution results:

```
SQL> BEGIN
      FOR rec IN (
        SELECT customer_id
        FROM customer...
Show more...

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.087
```

**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 rec IN (

    SELECT c.name, l.due_date

    FROM loans l

    JOIN customers c ON c.customer_id = l.customer_id

    WHERE l.due_date BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

    DBMS_OUTPUT.PUT_LINE('Reminder: Dear ' || rec.name ||

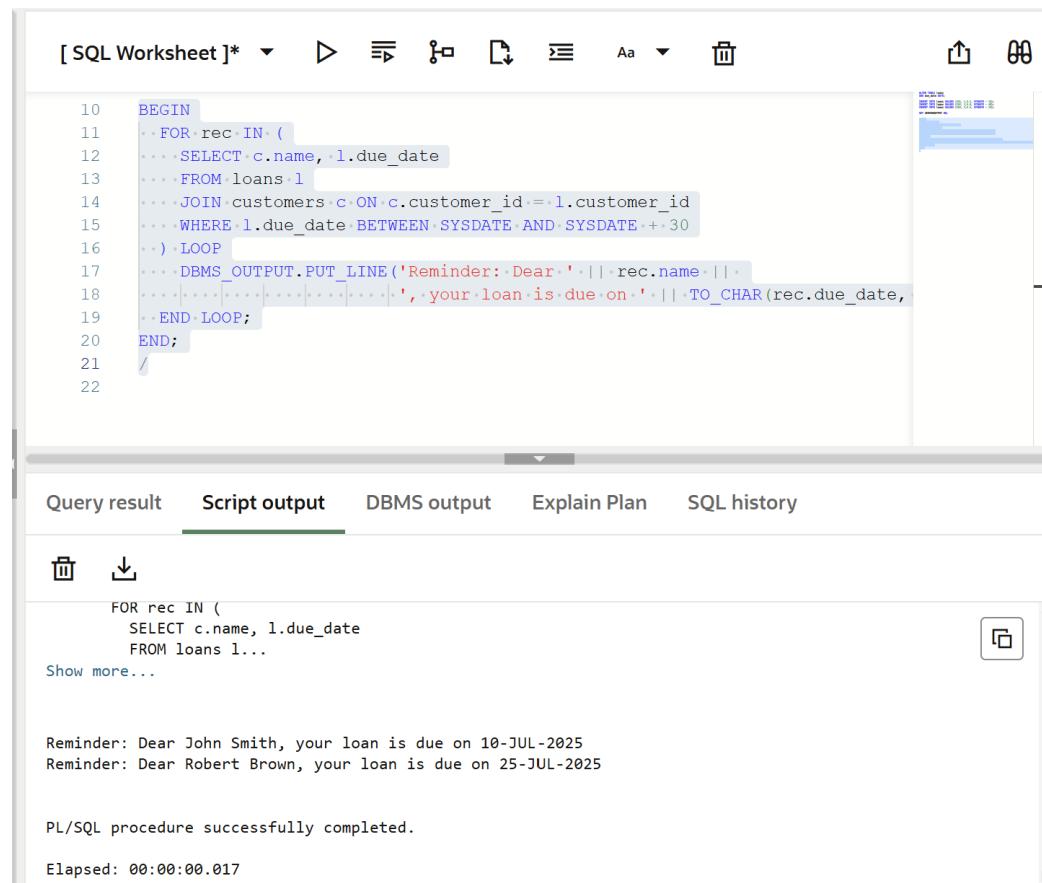
        ', your loan is due on ' || TO_CHAR(rec.due_date, 'DD-MON-YYYY'));

END LOOP;

END;

/
```

### Output:



[ SQL Worksheet ]\*

```
10 BEGIN
11   FOR rec IN (
12     SELECT c.name, l.due_date
13     FROM loans l
14     JOIN customers c ON c.customer_id = l.customer_id
15     WHERE l.due_date BETWEEN SYSDATE AND SYSDATE + 30
16   ) LOOP
17     DBMS_OUTPUT.PUT_LINE('Reminder: Dear ' || rec.name ||
18       ', your loan is due on ' || TO_CHAR(rec.due_date,
19     'DD-MON-YYYY'));
19   END LOOP;
20 END;
21 /
22
```

Query result | **Script output** | DBMS output | Explain Plan | SQL history

FOR rec IN (  
 SELECT c.name, l.due\_date  
 FROM loans l...

Show more...

Reminder: Dear John Smith, your loan is due on 10-JUL-2025  
Reminder: Dear Robert Brown, your loan is due on 25-JUL-2025

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.017