superset id: 6405870

Control Structures in PL/SQL

PL/SQL control structures allow you to control the flow of your program.
They include:

- Conditional Statements: IF, CASE
- Looping Statements: LOOP, WHILE, FOR

In this exercise, we use:

- FOR loops to iterate over rows
- IF conditions to check and apply logic based on column values

---

2. Table Structure and Attributes

- **Table: bank_customers**
  - customer_id: NUMBER (Primary Key)
  - name: VARCHAR2(50)
  - age: NUMBER
  - balance: NUMBER
  - loan_intrest: NUMBER(5,2)
  - is_vip: CHAR(1)
- **Table: loans**
  - loan_id: NUMBER (Primary Key)
  - customer_id: NUMBER (Foreign Key → bank_customers.customer_id)
  - due_date: DATE

---

3. Table Creation Script

```
CREATE TABLE bank_customers (
   customer_id   NUMBER PRIMARY KEY,
   name          VARCHAR2(50),
   age           NUMBER,
   balance       NUMBER,
   loan_intrest  NUMBER(5,2),
   is_vip        CHAR(1)
);
```

```
CREATE TABLE loans (
   loan_id      NUMBER PRIMARY KEY,
   customer_id  NUMBER,
   due_date     DATE,
   CONSTRAINT fk_customer
     FOREIGN KEY (customer_id)
     REFERENCES bank_customers(customer_id)
);
```

---

4. Inserting Sample Data

```
INSERT INTO bank_customers VALUES (1, 'Chaitanya',  23,  5000,   9.50, NULL);
INSERT INTO bank_customers VALUES (2, 'Varma',      67, 15000,  10.00, NULL);
INSERT INTO bank_customers VALUES (3, 'Bunny',      45,  8000,  11.00, NULL);
INSERT INTO bank_customers VALUES (4, 'Pavani',     70, 12000,   8.75, NULL);
INSERT INTO bank_customers VALUES (5, 'Bhanu',      62,  9500,   9.25, NULL);
```

superset id: 6405870

```
INSERT INTO bank_customers VALUES (6, 'Raj',        31, 20000,  10.50, NULL);
INSERT INTO bank_customers VALUES (7, 'Sneha',      55, 11000,   9.00, NULL);
INSERT INTO bank_customers VALUES (8, 'Teja',       65,  7500,  10.25, NULL);
INSERT INTO bank_customers VALUES (9, 'Meena',      28,  6000,   9.75, NULL);
INSERT INTO bank_customers VALUES (10, 'Karthik',   72, 16000,   8.50, NULL);

INSERT INTO loans VALUES (101, 2,  TO_DATE('05-JUL-2025', 'DD-MON-YYYY'));
INSERT INTO loans VALUES (102, 4,  TO_DATE('15-JUL-2025', 'DD-MON-YYYY'));
INSERT INTO loans VALUES (103, 6,  TO_DATE('25-JUL-2025', 'DD-MON-YYYY'));
INSERT INTO loans VALUES (104, 7,  TO_DATE('22-JUN-2025', 'DD-MON-YYYY'));
INSERT INTO loans VALUES (105, 9,  TO_DATE('01-AUG-2025', 'DD-MON-YYYY'));

COMMIT;
```

Tables :

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

Download | Execution time: 0.004 seconds

	CUSTOMER_ID	NAME	AGE	BALANCE	LOAN_INTREST	IS_VIP
1	1	Chaitanya	23	5000	9.5	*(null)*
2	2	Varma	67	15000	10	*(null)*
3	3	Bunny	45	8000	11	*(null)*
4	4	Pavani	70	12000	8.75	*(null)*
5	5	Bhanu	62	9500	9.25	*(null)*
6	6	Raj	31	20000	10.5	*(null)*
7	7	Sneha	55	11000	9	*(null)*
8	8	Teja	65	7500	10.25	*(null)*
9	9	Meena	28	6000	9.75	*(null)*
10	10	Karthik	72	16000	8.5	*(null)*

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

Download | Execution time: 0.001 seconds

	LOAN_ID	CUSTOMER_ID	DUE_DATE
1	101	2	7/5/2025, 12:00:00
2	102	4	7/15/2025, 12:00:0(
3	103	6	7/25/2025, 12:00:0(
4	104	7	6/22/2025, 12:00:0(
5	105	9	8/1/2025, 12:00:00

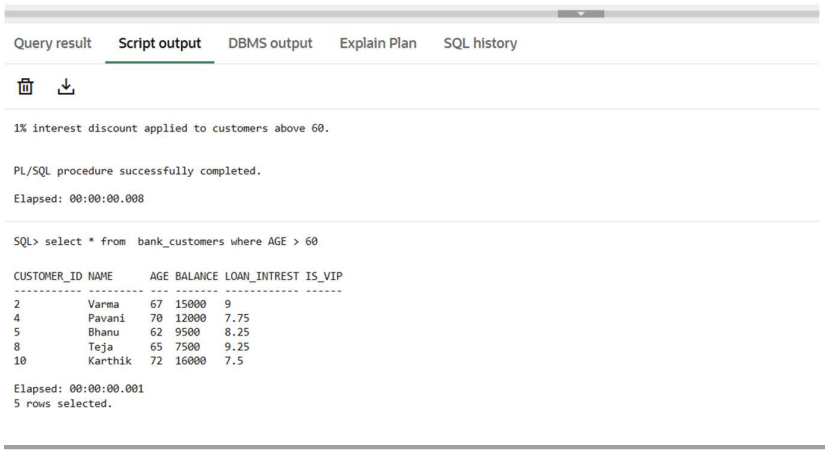
superset id: 6405870

Scenario 1: Senior Citizen Interest Discount
Description:
Reduce loan interest by 1% for all customers older than 60 years.
Code:

```
BEGIN
  FOR cust IN (
    SELECT customer_id, age, loan_intrest
    FROM bank_customers
  ) LOOP
    IF cust.age > 60 THEN
      UPDATE bank_customers
      SET loan_intrest = cust.loan_intrest - 1
      WHERE customer_id = cust.customer_id;
    END IF;
  END LOOP;

  COMMIT;
  DBMS_OUTPUT.PUT_LINE('1% interest discount applied to customers above 60.');
END;
```

Explanation:
The loop checks each customer's age. If the age is above 60, their loan_intrest is reduced by 1%.



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

```
1% interest discount applied to customers above 60.


PL/SQL procedure successfully completed.

Elapsed: 00:00:00.008

SQL> select * from  bank_customers where AGE > 60

CUSTOMER_ID NAME      AGE BALANCE LOAN_INTREST IS_VIP
----------- --------- --- ------- ------------ ------
2           Varma     67  15000   9
4           Pavani    70  12000   7.75
5           Bhanu     62  9500    8.25
8           Teja      65  7500    9.25
10          Karthik   72  16000   7.5

Elapsed: 00:00:00.001
5 rows selected.
```

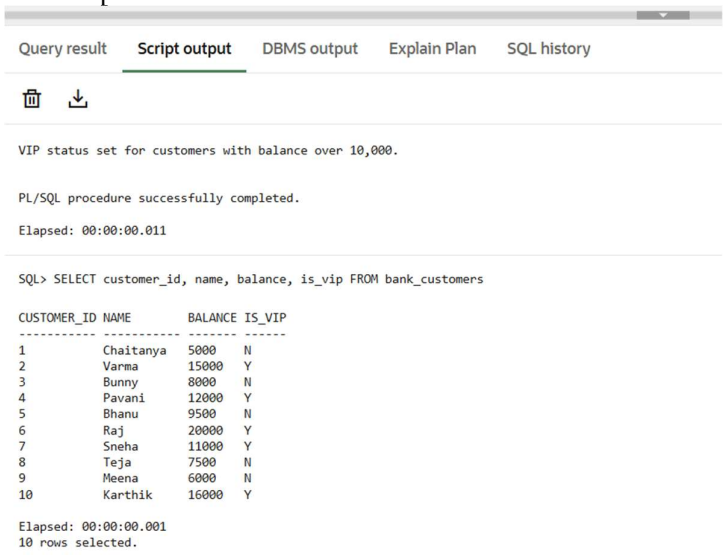
Scenario 2: Promote to VIP Based on Balance
Description:
Customers with a balance greater than 10,000 are marked as VIP (is_vip = 'Y'), else is_vip = 'N'.
Code:

```
BEGIN
  FOR cust IN (
    SELECT customer_id, balance
    FROM bank_customers
  ) LOOP
    IF cust.balance > 10000 THEN
      UPDATE bank_customers
      SET is_vip = 'Y'
      WHERE customer_id = cust.customer_id;
    ELSE
```

```
    UPDATE BANK_CUSTOMERS SET is_vip ='N'
    WHERE CUSTOMER_ID=cust.customer_id;
  END IF;
 END LOOP;

 COMMIT;
 DBMS_OUTPUT.PUT_LINE('VIP status set for customers with balance over 10,000.');
END;
```

Explanation:
This script sets VIP status based on the balance of each customer.



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

```
VIP status set for customers with balance over 10,000.

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.011

SQL> SELECT customer_id, name, balance, is_vip FROM bank_customers

CUSTOMER_ID NAME        BALANCE IS_VIP
----------- ----------- ------- ------
1           Chaitanya   5000    N
2           Varma       15000   Y
3           Bunny       8000    N
4           Pavani      12000   Y
5           Bhanu       9500    N
6           Raj         20000   Y
7           Sneha       11000   Y
8           Teja        7500    N
9           Meena       6000    N
10          Karthik     16000   Y

Elapsed: 00:00:00.001
10 rows selected.
```

Scenario 3: Loan Due Reminders
Description:
Print reminder messages for customers whose loan is due within the next 30 days.
Code:

```
BEGIN
  FOR loan_rec IN (
    SELECT l.loan_id, l.due_date, c.customer_id, c.name
    FROM loans l
    JOIN bank_customers c ON l.customer_id = c.customer_id
    WHERE l.due_date <= SYSDATE + 30
  ) LOOP
    DBMS_OUTPUT.PUT_LINE(
      '🔔 Reminder: Loan ' || loan_rec.loan_id ||
      ' for ' || loan_rec.name ||
      ' (Customer ID: ' || loan_rec.customer_id || ')' ||
      ' is due on ' || TO_CHAR(loan_rec.due_date, 'DD-MON-YYYY')
    );
  END LOOP;
END;
```

Explanation:
Joins loans and bank_customers, and prints reminders if the due date is within the next 30 days.

superset id: 6405870