

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.

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.

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.

Schema

```
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 Transactions (  
    TransactionID NUMBER PRIMARY KEY,  
    AccountID NUMBER,  
    TransactionDate DATE,  
    Amount NUMBER,  
    TransactionType VARCHAR2(10),  
    FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)  
);
```

```
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 Employees (  
    EmployeeID NUMBER PRIMARY KEY,  
    Name VARCHAR2(100),  
    Position VARCHAR2(50),  
    Salary NUMBER,  
    Department VARCHAR2(50),  
    HireDate DATE  
);
```

DATA

```
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 (CustomerID, Name, DOB, Balance, LastModified)  
VALUES (2, 'Jane Smith', TO_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);
```

```
INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)  
VALUES (1, 1, 'Savings', 1000, SYSDATE);
```

```
INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)  
VALUES (2, 2, 'Checking', 1500, SYSDATE);
```

```
INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)  
VALUES (1, 1, SYSDATE, 200, 'Deposit');
```

```
INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)  
VALUES (2, 2, SYSDATE, 300, 'Withdrawal');
```

```
INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)  
VALUES (1, 1, 5000, 5, SYSDATE, ADD_MONTHS(SYSDATE, 60));
```

```
INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)  
VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', TO_DATE('2015-06-15', 'YYYY-MM-DD'));
```

```
INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)
VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', TO_DATE('2017-03-20', 'YYYY-MM-DD'));
```

```
INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)
VALUES (3, 'Senior Citizen', TO_DATE('1950-01-01', 'YYYY-MM-DD'), 9000, SYSDATE);
```

```
INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)
VALUES (2, 3, 7000, 6, SYSDATE, ADD_MONTHS(SYSDATE, 36));
```

```
INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)
VALUES (4, 'Swetha VIP', TO_DATE('1980-01-01', 'YYYY-MM-DD'), 20000, SYSDATE);
```

```
INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)
VALUES (3, 2, 6000, 6.5, SYSDATE, SYSDATE + 10);
```

```
ALTER TABLE Customers ADD IsVIP VARCHAR2(5);
```

Scenario 1:

Code:

```
BEGIN
```

```
  FOR c IN (
    SELECT CustomerID
    FROM Customers
    WHERE MONTHS_BETWEEN(SYSDATE, DOB)/12 > 60
  ) LOOP
```

```
    UPDATE Loans
    SET InterestRate = InterestRate - 1
    WHERE CustomerID = c.CustomerID;
  END LOOP;
```

```
  COMMIT;
```

```
END;
```

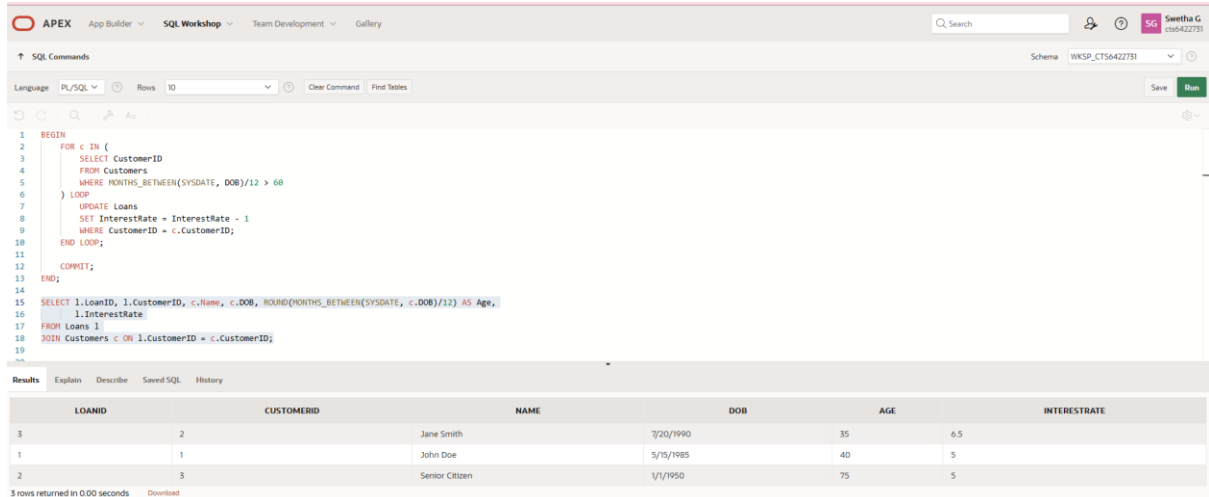
```
SELECT l.LoanID, l.CustomerID, c.Name, c.DOB, ROUND(MONTHS_BETWEEN(SYSDATE, c.DOB)/12)
AS Age,
```

l.InterestRate

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID;

OUTPUT:



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. A search bar and user profile 'Srinitha G' are on the right. The 'SQL Commands' section shows a PL/SQL script with line numbers 1 through 19. The script includes a loop to update interest rates for customers over 60 years old. Below the script, the 'Results' tab is active, displaying a table with 3 rows and 8 columns: LOANID, CUSTOMERID, NAME, DOB, AGE, INTERESTRATE, and two unlabeled columns. The data rows are: (3, 2, Jane Smith, 7/20/1990, 35, 6.5), (1, 1, John Doe, 5/15/1985, 40, 5), and (2, 3, Senior Citizen, 1/1/1950, 75, 5). A status bar at the bottom indicates '3 rows returned in 0.00 seconds' and a 'Download' link.

LOANID	CUSTOMERID	NAME	DOB	AGE	INTERESTRATE		
3	2	Jane Smith	7/20/1990	35	6.5		
1	1	John Doe	5/15/1985	40	5		
2	3	Senior Citizen	1/1/1950	75	5		

3 rows returned in 0.00 seconds [Download](#)

Scenario 2:

Code:

BEGIN

FOR c IN (

SELECT CustomerID FROM Customers WHERE Balance > 10000

) LOOP

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = c.CustomerID;

END LOOP;

COMMIT;

END;

SELECT CustomerID, Name, Balance, IsVIP

FROM Customers;

OUTPUT:

```

21 |
22 | BEGIN
23 |   FOR c IN (
24 |     SELECT CustomerID FROM Customers WHERE Balance > 10000
25 |   ) LOOP
26 |     UPDATE Customers
27 |       SET IsVIP = 'TRUE'
28 |     WHERE CustomerID = c.CustomerID;
29 |   END LOOP;
30 |
31 | COMMIT;
32 | END;
33 |
34 | SELECT CustomerID, Name, Balance, IsVIP
35 | FROM Customers;
36 |
37 |

```

	CUSTOMERID	NAME	BALANCE	ISVIP
2		Jane Smith	1500	-
4		Sreetha VIP	20000	TRUE
3		Senior Citizen	9000	-
1		John Doe	1000	-

rows returned in 0.00 seconds [Download](#)

Scenario 3:

Code:

```
BEGIN
```

```
  FOR l IN (
```

```
    SELECT c.Name, l.EndDate
```

```
    FROM Loans l
```

```
    JOIN Customers c ON l.CustomerID = c.CustomerID
```

```
    WHERE l.EndDate <= SYSDATE + 30
```

```
  ) LOOP
```

```
    DBMS_OUTPUT.PUT_LINE('Reminder: Dear ' || l.Name || ', your loan is due on ' || TO_CHAR(l.EndDate,
'DD-Mon-YYYY'));

```

```
  END LOOP;
```

```
END;
```

OUTPUT:

```

38 | BEGIN
39 |   FOR l IN (
40 |     SELECT c.Name, l.EndDate
41 |     FROM Loans l
42 |     JOIN Customers c ON l.CustomerID = c.CustomerID
43 |     WHERE l.EndDate <= SYSDATE + 30
44 |   ) LOOP
45 |     DBMS_OUTPUT.PUT_LINE('Reminder: Dear ' || l.Name || ', your loan is due on ' || TO_CHAR(l.EndDate, 'DD-Mon-YYYY'));
46 |   END LOOP;
47 | END;
48 |
49 |
50 |
51 |

```

Results	Explain	Describe	Saved SQL	History
Reminder: Dear Jane Smith, your loan is due on 09-Jul-2025				
Statement processed.				