Name : K. Taraka Lakshmi Prasanna

Superset ID : 6314755

Email : 22BQ1A4268@vvit.net

Week – 2

Topic : PL/SQL

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.
- **Solution :**

```
DECLARE
        CURSOR SENIOR_CUSTOMERS IS
                SELECT C.CUSTOMERID FROM CUSTOMERS C
                WHERE  MONTHS_BETWEEN(SYSDATE, C.DOB)/12 >60;
BEGIN
        FOR CUSTOMER IN SENIOR_CUSTOMERS LOOP
                UPDATE LOANS SET INTRESTRATE := INTRESTRATE -1
                WHERE CUSTOMER.CUSTOMERID = SENIOR_CUSTOMER.CUSTOMERID;
                DBMS_OUTPUT.PUT_LINE("DISCOUNT APPLIED FOR : "
                ||CUSTOMER.CUSTOMERID);
        END LOOP;
END;
/
```

This Program will loop through all the customers and check whether they are above 60 years or not using a cursor. If the customer age is a above 60 then they will get 10% discount on intrest rate simply we can reduce 1 from the intrest rate.

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.
- **Solution :**

To implement this scenario first me need to update the schema of the customers table . We need to add a new column named VIP which contains 'Y' or 'N' to indicate the status. We update the table with following code:

```
ALTER TABLE CUSTOMERS ADD VIP CHAR(1);
```

Now we will update the VIP column with the help of a cursor.Solution code is below :

```
BEGIN
    FOR  CUSTOMER IN (SELECT CUSTOMERID , BALANCE FROM CUSTOMERS ) LOOP
            IF CUSTOMER.BALANCE > 10000 THEN
                    UPDATE CUSTOMERS SET VIP = 'Y' WHERE CUSTOMER.ID = CUSTOMERID;
            ELSE
                    UPDATE CUSTOMERS SET VIP='N WHERE CUSTOMER.ID = CUSTOMERID;
            END IF;
    END LOOP;
END;
/
```

This program will set VIP status of the customers.

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.
- **Solution :**

```
BEGIN
    FOR LOAN IN (
            SELECT l.LOANID, l.CUSTOMERID, l.ENDDATE, c.NAME
                    FROM Loans l
                    JOIN CUSTOMERS c ON l.CustomerID = c.CustomerID
                      WHERE l.ENDDATE BETWEEN SYSDATE AND SYSDATE + 30
     ) LOOP
            DBMS_OUTPUT.PUT_LINE('Reminder: Loan ID ' || loan.LoanID ||
                    'for customer '" || loan.Name ||
                     ' " is due on ' || TO_CHAR(loan.EndDate, 'DD-MON-YYYY') || '.')
    END LOOP;

 END;

 /
```

This program will fetch loans due in the next 30 days and print a remainder message.

Stored Procedures :

Scenario 1: The bank needs to process monthly interest for all savings accounts.

- **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.
- **Solution :**

```
CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS
BEGIN
        UPDATE Accounts
        SET Balance = Balance + (Balance * 0.01)
        WHERE AccountType = 'Savings';
   COMMIT;
END;
/
```

This procedure will update the balnce of all savings account by applying 1% intrest on current balance.To run this procedure we need to run the following code :

```
BEGIN

 ProcessMonthlyInterest;

END;

/
```

Scenario 2: The bank wants to implement a bonus scheme for employees based on their performance.

- **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.
- **Solution :**

```
CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (
                DEPT IN VARCHAR2,
                BONUS_PERCENT IN NUMBER
) AS
BEGIN
    UPDATE EMPLOYEES
    SET SALARY = SALARY + (SALARY * (BONUS_PERCENT / 100 ))
    WHERE DEPARTMENT = DEPT ;
    COMMIT;
END ;
```

This program uses a procedure to update the salary of the employees according to their department.To run this procedure we need to run the folloeing code :

```
BEGIN
   UpdateEmployeeBonus('department_name', bonus_percent);
END;
/
```

Scenario 3: Customers should be able to transfer funds between their accounts.

- **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.
- **Solution :**

```
CREATE OR REPLACE PROCEDURE TransferFunds (
    p_from_account_id IN NUMBER,
    p_to_account_id   IN NUMBER,
    p_amount          IN NUMBER
) AS  v_from_balance NUMBER;
BEGIN
    SELECT Balance INTO v_from_balance
    FROM Accounts
    WHERE AccountID = p_from_account_id
    FOR UPDATE;

    IF v_from_balance < p_amount THEN
            RAISE_APPLICATION_ERROR(-20001, 'Insufficient balance in the source account.');
    END IF;
    UPDATE Accounts
            SET Balance = Balance - p_amount
            WHERE AccountID = p_from_account_id;
    UPDATE Accounts
            SET Balance = Balance + p_amount
            WHERE AccountID = p_to_account_id;
    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Transfer successful: ' || p_amount ||
              ' transferred from account ' || p_from_account_id ||
              ' to account ' || p_to_account_id);
EXCEPTION
     WHEN NO_DATA_FOUND THEN
              RAISE_APPLICATION_ERROR(-20002, 'One or both accounts not found.');
     WHEN OTHERS THEN
            ROLLBACK;
            RAISE;
END;
/
```