Islamic University of Technology

Department of Computer Science

Database Management Systems II Lab

Course Code- CSE 4410

Prepared By:
Nazia Krim Khan Oishee
ID: 200042137

Date: February 3, 2023

Contents

1 Introduction

In our fourth lab of DBMS II our task was to implement procedures and functions through PL/SQL by performing queries based on a schema for a Banking database to acquire more expertise over those.

2 PL/SQL

PL/SQL stands for Procedural Language for Structured Query Language. The language allows us to write code and deploy it in the database nearest to data. PL/SQL simplifies application development, optimize execution, and improve resource utilization in the database. The language is a case-insensitive programming language, like SQL.

PL/SQL is a blocked programming language. Program units can be named or unnamed blocks. Unnamed blocks are known as anonymous blocks. Procedures and functions fall in the category of named block.

3 Procedure in PL/SQL

A procedure is a module that performs one or more actions. Procedures are key building blocks of modular code, which allow us to reuse and manage our program logic. Procedures have their associate formal parameter list through which we move data out and into procedure. The parameters cab be of IN, OUT or IN OUT type.

4 Function in PL/SQL

A function is a module that returns data through its RETURN clause, rather than in an OUT or IN OUT parameter.

Both procedures and functions have a name, can take parameters, return values and be called from other blocks. The difference is that a function must return a value, but in a procedure it is optional.

5 Tasks

We were given total 3 tasks in the lab based on a banking schema. Before creating functions and procedures we had to create necessary tables and insert values into them.

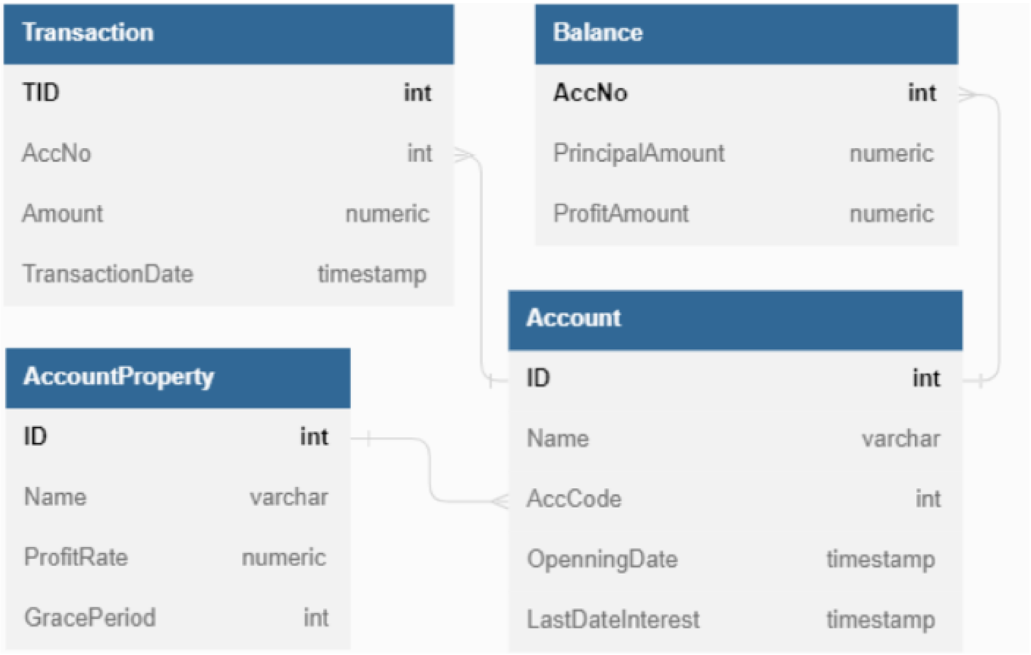

Figure 5.1: Banking Schema

5.1 Creating Tables

I created 4 tables according to the given schema which were- AccountProperty, , Account, Transaction, Balance along with necessary integrity constraints.

Code Snippet

```
drop table Balance;
drop table Transaction;
drop table Account;
drop table AccountProperty;

create table AccountProperty(
  ID number,
  Name varchar2(50),
  ProfitRate number,
  GracePeriod number,
  constraint Pk_AccountProperty PRIMARY KEY(ID)
);

create table Account(
  ID number,
  Name varchar2(50),
  AccCode number,
  OpeningDate date,
  LastDateInterest date,
  constraint Pk_Account PRIMARY KEY(ID),
  constraint Fk_Account FOREIGN KEY(AccCode) REFERENCES AccountProperty(ID) ON DELETE CASCADE
);

create table Transaction(
  TID number,
  AccNo number,
  Amount number,
  TransactionDate date,
  constraint Pk_Transaction PRIMARY KEY(TID),
  constraint Fk_Transaction FOREIGN KEY(AccNo) REFERENCES Account(ID) ON DELETE CASCADE
);

create table Balance(
  AccNo number,
  PrincipalAmount number,
  ProfitAmount number,
  constraint Pk_Balance PRIMARY KEY(AccNo),
  constraint Fk_Balance FOREIGN KEY(AccNo) REFERENCES Account(ID) ON DELETE CASCADE
);
```

5.2 Inserting Data

I inserted neccessary data in each table.

Code Snippet

```
insert into AccountProperty(ID, Name,ProfitRate,GracePeriod) values (2002, ’monthly’, 2.8, 1);
insert into AccountProperty (ID, Name,ProfitRate,GracePeriod) values (3003, ’quarterly’, 4.2, 4);
insert into AccountProperty (ID, Name,ProfitRate,GracePeriod) values (4004, ’biyearly’, 6.8, 6);
insert into AccountProperty (ID, Name,ProfitRate,GracePeriod) values (5005, ’yearly’, 8, 12);

insert into Account(ID,Name,AccCode,OpeningDate,LastDateInterest) values (1, ’A’, 1, TO_DATE(’10-10-2020’, ’DD-MM-YYYY’), TO_DATE(’11-11-2022’, ’DD-MM-YYYY’));
insert into Account(ID,Name,AccCode,OpeningDate,LastDateInterest) values (2, ’B’, 2, TO_DATE(’10-11-2020’, ’DD-MM-YYYY’), TO_DATE(’11-11-2022’, ’DD-MM-YYYY’));
insert into Account(ID,Name,AccCode,OpeningDate,LastDateInterest)values (3, ’C’, 3, TO_DATE(’11-10-2021’, ’DD-MM-YYYY’), TO_DATE(’11-11-2022’, ’DD-MM-YYYY’));
insert into Account(ID,Name,AccCode,OpeningDate,LastDateInterest) values (4, ’D’, 4, TO_DATE(’10-11-2021’, ’DD-MM-YYYY’), TO_DATE(’11-11-2022’, ’DD-MM-YYYY’));

insert into Transaction(TID,AccNo,Amount,TransactionDate) values (1, 1, 10000, TO_DATE(’10-10-2020’, ’DD-MM-YYYY’));
insert into Transaction(TID,AccNo,Amount,TransactionDate) values (3, 2, 120000, TO_DATE(’10-10-2020’, ’DD-MM-YYYY’));
insert into Transaction (TID,AccNo,Amount,TransactionDate)values (5, 3, 22000, TO_DATE(’10-10-2020’, ’DD-MM-YYYY’));
insert into Transaction (TID,AccNo,Amount,TransactionDate)values (7, 4, 22500, TO_DATE(’10-10-2020’, ’DD-MM-YYYY’));

insert into Balance(AccNo,PrincipalAmount,ProfitAmount) values (1, 2000, 500);
insert into Balance(AccNo,PrincipalAmount,ProfitAmount) values (2, 1000, 500);
insert into Balance(AccNo,PrincipalAmount,ProfitAmount) values (3, 4000, 400);
insert into Balance(AccNo,PrincipalAmount,ProfitAmount) values (4, 3000, 400);
```

5.3 Creating Functions and Procedures

Task1: You have to write a function to calculate the current balance from the transactions.

Solution

```
SET SERVEROUTPUT ON SIZE 1000000
CREATE OR REPLACE FUNCTION CurrentBalance(Acc_No IN number)
RETURN number
AS
CurrentBalance number;
Principal_Amount number;
BEGIN
    SELECT PrincipalAmount INTO Principal_Amount FROM BALANCE WHERE AccNo=Acc_No;
    SELECT sum(Amount) INTO CurrentBalance FROM Transaction WHERE AccNo=Acc_No;
    CurrentBalance := CurrentBalance + Principal_Amount;
    return CurrentBalance;
END;
/

DECLARE
    AccNo Transaction.AccNo%TYPE;
BEGIN
    AccNo:='&AccountIID';
    DBMS_OUTPUT.PUT_LINE(CurrentBalance(AccNo));
END;
/
```

Explanation

In this task, I created a function named CurrentBalance which takes a parameter Acc_No and returns the account's current balance.

At first, I selected the principle amount of the account from BALANCE table using SELECT statement. Then I calculated the total amount of transactions that have been done under the account. Then I added these two amount to calculate the current balance. The functions returns the calculated current balance.

I used SERVEROUTPUT to show the result in the console.

Task2: Write another function to calculate the profit based on profitRate, amount and duration. Take account id as input and return profit, balance before profit, and balance after profit.

Solution

```
SET SERVEROUTPUT ON SIZE 1000000
CREATE OR REPLACE FUNCTION Profit(Acc_NO IN Transaction.AccNo%TYPE, Profit OUT Balance.PrincipalAmount%TYPE,FinalBalance OUT Balance.PrincipalAmount%TYPE)
RETURN number
AS
PRate number;
GPeriod number;
code number;
Months number;
Principal number;
BalancebeforeProfit number;
BEGIN
  SELECT AccCode INTO code FROM Account WHERE ID=Acc_NO;
  SELECT PrincipalAmount INTO Principal FROM Balance WHERE AccNo=Acc_NO;
  SELECT ProfitRate INTO PRate FROM AccountProperty WHERE ID=code;
  SELECT GracePeriod INTO GPeriod  FROM AccountProperty WHERE ID=code;
  SELECT (SELECT SYSDATE FROM DUAL)-LastDateInterest INTO Months FROM Account WHERE ID=Acc_NO;
  Months:= (FLOOR(Months/30));
  BalancebeforeProfit:=Principal;
  Profit:=0;
  For i IN 1..Months
  LOOP
    Profit:=(Profit+(PRate/100)*Principal);
    IF MOD(i,GPeriod)=0 THEN
      Principal:=Principal+Profit;
      Profit:=0;
    END IF;
  END LOOP;
  FinalBalance:=Principal;
  RETURN BalancebeforeProfit;
END;
/

DECLARE
    Acc_NO Transaction.AccNo%TYPE;
    Profit Balance.PrincipalAmount%TYPE;
    FinalBalance Balance.PrincipalAmount%TYPE;
BEGIN
    Acc_NO:='&AccountIID';
    DBMS_OUTPUT.PUT_LINE('Balance Before Profit: '||Profit(Acc_NO,Profit,FinalBalance)||' Profit: ' || Profit ||' Final Balance: ' || FinalBalance);
END;
/
```

Explanation

At first I selected the AccCode of the given account into code from Account using SELECT statement. Using the code I selected the ProfitRate and GracePeriod of that type of account. I used this two information later to calculate the profit.

I also selected the PrincipalAmount of the particular account from BALANCE.

The profit depends on the months elapsed between till date and opening of the account. Each type of account has a particular grace period and profit rate. Profit is calculated monthly on the basis of ProfitRate. After the grace period is over, the profit is added to the principal amount. Then the new principal would be this new current balance. Until the next grace period is over, profit would be calculated based on this new principal.

The function returns the initial principal which is the balance before profit through its return clause. As a function can return only one value through its return clause, I have used out parameter to return the total balance and the total profit.

Task3: Write a procedure to calculate all accounts' profit (i.e. profit will be calculated if it satisfies conditions). Use the cursor for loop for this problem. The procedure will insert the appropriate record in its Amounts table.

Solution

```
CREATE OR REPLACE PROCEDURE amounts
AS
  Profit Balance.PrincipalAmount%TYPE;
  FinalBalance Balance.PrincipalAmount%TYPE;
  PreviousBalance Balance.PrincipalAmount%TYPE;
BEGIN
  FOR i IN (SELECT * FROM Account)
  LOOP
    PreviousBalance:=Profit(i.ID,Profit,FinalBalance);
    UPDATE Balance SET PrincipalAmount=FinalBalance WHERE AccNo=i.ID;
    UPDATE Balance SET ProfitAmount=Profit WHERE AccNo=i.ID;
  END LOOP;
END;
/

BEGIN
  amounts();
END;
/
```

Explanation

In this task, I used cursor for loop to select all the accounts from Account table.

For each account I called the function Profit from task 2. The function returns the initial principal amount. Through out parameter I selected the total profit and final balance into Profit and FinalBalance respectively. Then I used UPDATE statement to update the profit and principle amount of the particular account.

6 Problems I faced

In this lab, I faced less problems regarding the syntax. But I faced problems to understand the scenario and to figure out the solution. I have realized over the past lab classes that I need to work on my capability of visualizing scenarios and database to understand and figure out the solution faster. I look forward to achieve that.