Islamic University of Technology (IUT)

Report on Lab 4

Submitted By

Shanta Maria (ID:200042172)

CSE 4410 Database Management Systems II Lab

Submitted To

Dr. Abu Raihan Mostofa Kamal

Professor, Department of CSE

Zannatun Naim Srsity

Lecturer, Department of CSE

Md. Rafid Haque

Lecturer, Department of CSE

February 7, 2023

Introduction

These lab tasks were based on the use of PL/SQL.

Question

Create the necessary tables and insert some values.

Table 1: Account Property.

ID	Name	ProfitRate(per month)	GracePeriod (month)
2002	monthly	2.8	1
3003	quarterly	4.2	4
4004	biyearly	6.8	6
5005	yearly	8	12

1. You have to write a function to calculate the current balance from the transactions.
2. 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.
3. 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.

1 Task 1

1.1 Analysis and Explanation

For the first task, I used two cursors. The first cursor outputs the sum of all the transaction amounts from the transaction table. The second cursor outputs the principal amount for the given account number from the database. I then added the sum of all the transactions to the principal amount to find the current balance. This is assuming that all the transactions are of deposit type.

```
-----1-----
-----function-----
create or replace
function current_balance(a_num in Account.id%type)
return numeric
is

    p_amount Balance.PrincipleAmount%type; ---principle amount---
    t_amount Transaction.Amount%type; ---transaction amount---
    curr_balance numeric;

    cursor transaction_amount(a_num Account.id%type)
    is
        select sum(amount) amount
        from transaction
        where AccNo = a_num;

    cursor prin_am(a_num Account.id%type)
    is
        select PrincipleAmount as pm
        from balance
        where AccNo = a_num;

begin
    curr_balance := 0.0;

    open prin_am(a_num);
    fetch prin_am into p_amount;

    curr_balance := p_amount;

    open transaction_amount(a_num);
    fetch transaction_amount into t_amount;

    curr_balance := curr_balance + t_amount;

    close prin_am;
    close transaction_amount;

return curr_balance;
end;
/

-----call-----
declare
    acc_num int;
begin
    acc_num := '& acc_num';
    dbms_output.put_line('Current Balance: ' || current_balance(acc_num));
end;
/
```

```
SQL> declare
  2       acc_num int;
  3  begin
  4       acc_num := '& acc_num';
  5       dbms_output.put_line('Current Balance: ' || current_balance(acc_num));
  6  end;
  7  /
Enter value for acc_num: 15
Current Balance: 66569

PL/SQL procedure successfully completed.
```

1.2 Difficulties

I did not face any mentionable difficulties while implementing this task.

2 Task 2

2.1 Analysis and Explanation

For the second task, I created two cursors. One cursor returns all the necessary information of the account type for calculation such as profit rate, grace period, and principal amount. The second cursor returns the total number of months between the last interest date and today. Here the value is rounded to the floor value and it is calculated using the months_between function. Since that function uses date input, I cast the timestamp values as the date for this. After creating the cursors, I ran a loop for the number of months found by the second cursor. Inside the loop, I use calculated the profit by multiplying the profit rate by the principal amount. I also kept track of the total profit and the current accumulated profit since the profit is reset every time the grace period is reached. The principal amount is also updated by adding the current accumulated profit to it during the grace period. The balance after profit is just the principal amount and the total profit added together. All these are outputted in the form of a single string by the function.

```
-----2-----
-----function-----
create or replace
function profit(acc_num in Account.id%type)
return varchar2
is
    result varchar2(200);
    pr AccountProperty.ProfitRate%type; ---profit rate---
    pa Balance.PrincipleAmount%type; ---principle amount---
    gp AccountProperty.GracePeriod%type; ---grace period---
    month_num int;
    total_profit numeric;
    temp_profit numeric;
    after_balance numeric;

    cursor info(acc_num Account.id%type)
    is
        select AccountProperty.ProfitRate, AccountProperty.GracePeriod, Balance.PrincipleAmount
        from Account, AccountProperty, Balance
        where Account.id = Balance.AccNo and Account.AccCode = AccountProperty.id and Account.id=acc_num;


    cursor month_diff(acc_num Account.id%type)
    is
        select floor(months_between(cast(systimestamp as date), cast(LastDateInterest as date)))
        from Account, dual
        where id = acc_num;

begin
    after_balance := 0;
    total_profit := 0;
    temp_profit := 0;
    result := 'nothing';

    open info(acc_num);
    fetch info into pr, gp, pa;

    open month_diff(acc_num);
    fetch month_diff into month_num;

    for i in 1..month_num loop

        temp_profit := temp_profit + (pa * (pr/100));
        total_profit := total_profit + temp_profit;

        if(mod(i, gp) = 0) then
            pa := pa + temp_profit;
            temp_profit := 0;
        end if;

    end loop;

    after_balance := pa + total_profit;

    result := 'Profit = ' || total_profit || chr(10);
    result := result || 'Balance before Profit = ' || pa || chr(10);
    result := result || 'Balance after Profit = ' || after_balance || chr(10);

return result;
end;
/

-----call-----
declare
    acc_num int;
begin
    acc_num := '& acc_num';
    dbms_output.put_line(profit(acc_num));
end;
/
```

```
SQL> declare
  2       acc_num int;
  3  begin
  4       acc_num := '& acc_num';
  5       dbms_output.put_line(profit(acc_num));
  6  end;
  7  /
Enter value for acc_num: 29
Profit = 27631
Balance before Profit = 73747
Balance after Profit = 101378


PL/SQL procedure successfully completed.
```

2.2 Difficulties

I faced difficulties in finding a solution to calculate the profit and reset the profit during the grace period. I also faced difficulties using the timestamp data type as it was new for me.

3 Task 3

3.1 Analysis and Explanation

For the third task, I created the same function as in task 2 but changed the return value to the calculated total profit only. Then I created a procedure. Inside the procedure, I used one cursor which returns all the possible account IDs. Then I used a loop to traverse through the results of the cursor and for each result, I used the function created beforehand to output the profit. I then updated the balance table with the calculated profit from the function.

3.2 Difficulties

I did not face any mentionable difficulties while implementing this task.

```
-----3-----
-----function----
create or replace
function profit_for_insertion(acc_num in Account.id%type)
return numeric
is
    pr AccountProperty.ProfitRate%type; ---profit rate---
    pa Balance.PrincipleAmount%type; ---principle amount---
    gp AccountProperty.GracePeriod%type; ---grace period---
    month_num int;
    total_profit numeric;
    temp_profit numeric;
    after_balance numeric;

    cursor info(acc_num Account.id%type)
    is
        select AccountProperty.ProfitRate, AccountProperty.GracePeriod, Balance.PrincipleAmount
        from Account, AccountProperty, Balance
        where Account.id = Balance.AccNo and Account.AccCode = AccountProperty.id and Account.id=acc_num;


    cursor month_diff(acc_num Account.id%type)
    is
        select floor(months_between(cast(systimestamp as date), cast(LastDateInterest as date)))
        from Account, dual
        where id = acc_num;

begin
    after_balance := 0;
    total_profit := 0;
    temp_profit := 0;

    open info(acc_num);
    fetch info into pr, gp, pa;

    open month_diff(acc_num);
    fetch month_diff into month_num;

    for i in 1..month_num loop

        temp_profit := temp_profit + (pa * (pr/100));
        total_profit := total_profit + temp_profit;

        if(mod(i, gp) = 0) then
            pa := pa + temp_profit;
            temp_profit := 0;
        end if;

    end loop;

return total_profit;
end;
/

-----procedure-----
create or replace
procedure insert_profit
is
    acc_num Account.id%type;
    prof Balance.ProfitAmount%type;

    cursor find_acc
    is
        select id
        from account;

begin
    open find_acc;

    loop
        fetch find_acc into acc_num;
        exit when find_acc%notfound;

        prof := profit_for_insertion(acc_num);

        update balance set ProfitAmount = prof where AccNo = acc_num;

    end loop;

end;
/
```

```
SQL> -----call-----
SQL> begin
  2      insert_profit;
  3  end;
  4  /

PL/SQL procedure successfully completed.

SQL> select * from balance;

     ACCNO PRINCIPLEAMOUNT PROFITAMOUNT
---------- --------------- ------------
        15           54496         4360
        72           64190      1018563
        33           63800     80698937
        29           46116        27631
        18           78044       412683
        13           36881         1033

6 rows selected.
```

4 DDL

(Last two pages)

Conclusion

As shown in the report, I completed all the lab tasks. All the commands used were written in VS Code which was then saved with the .sql extension. The .sql file was then run through the SQL command line to execute all the commands.

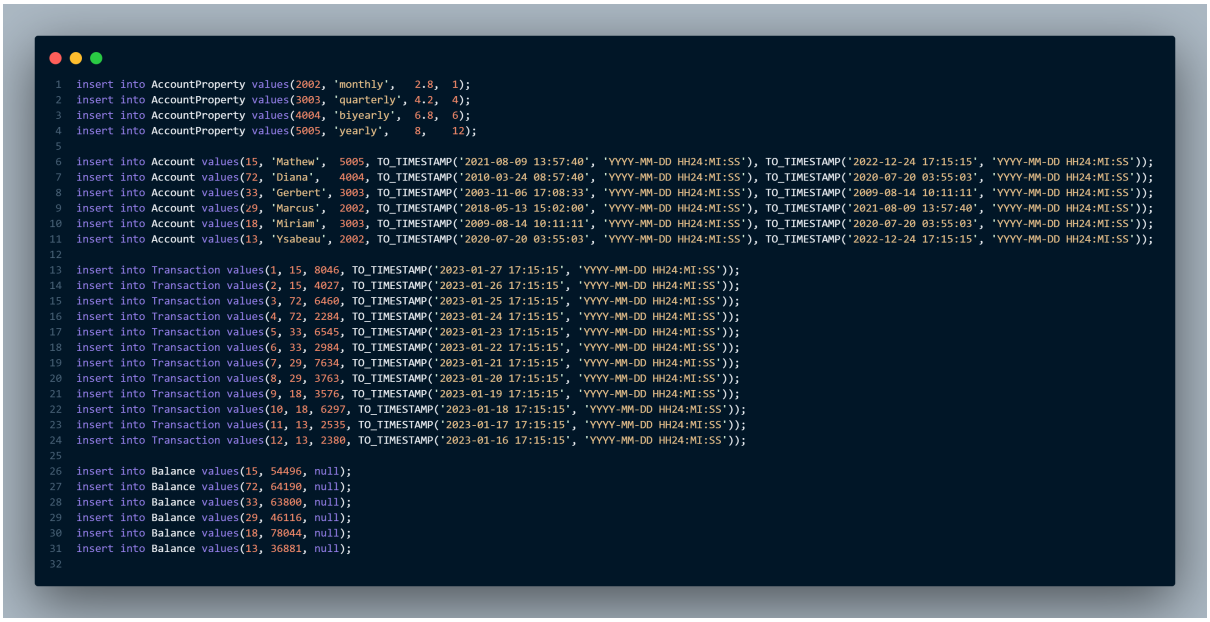
```sql
drop table AccountProperty cascade constraint;
drop table Transaction cascade constraint;
drop table Balance cascade constraint;
drop table Account cascade constraint;

create table AccountProperty
(
    id int,
    name varchar2(20),
    ProfitRate numeric(2, 1),
    GracePeriod int,
    constraint pk_accprop primary key(id)
);

create table Account
(
    id int,
    name varchar2(20),
    AccCode int,
    OpeningDate timestamp,
    LastDateInterest timestamp,
    constraint pk_account primary key(id),
    constraint fk_acc_no foreign key(AccCode) references AccountProperty(id)
);

create table Balance
(
    AccNo int,
    PrincipleAmount numeric,
    ProfitAmount numeric,
    constraint pk_balance primary key(AccNo),
    constraint fk_a_no foreign key(AccNo) references Account(id)
);

create table Transaction
(
    tid int,
    AccNo int,
    Amount numeric,
    TransactionDate timestamp,
    constraint pk_transaction primary key(tid),
    constraint fk_acc_no1 foreign key(AccNo) references Account(id)
);
```



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

insert into Account values(15, 'Mathew',  5005, TO_TIMESTAMP('2021-08-09 13:57:40', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2022-12-24 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Account values(72, 'Diana',   4004, TO_TIMESTAMP('2010-03-24 08:57:40', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2020-07-20 03:55:03', 'YYYY-MM-DD HH24:MI:SS'));
insert into Account values(33, 'Gerbert', 3003, TO_TIMESTAMP('2003-11-06 17:08:33', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2009-08-14 10:11:11', 'YYYY-MM-DD HH24:MI:SS'));
insert into Account values(29, 'Marcus',  2002, TO_TIMESTAMP('2018-05-13 15:02:00', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2021-08-09 13:57:40', 'YYYY-MM-DD HH24:MI:SS'));
insert into Account values(18, 'Miriam',  3003, TO_TIMESTAMP('2009-08-14 10:11:11', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2020-07-20 03:55:03', 'YYYY-MM-DD HH24:MI:SS'));
insert into Account values(13, 'Ysabeau', 2002, TO_TIMESTAMP('2020-07-20 03:55:03', 'YYYY-MM-DD HH24:MI:SS'), TO_TIMESTAMP('2022-12-24 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));

insert into Transaction values(1, 15, 8046, TO_TIMESTAMP('2023-01-27 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(2, 15, 4027, TO_TIMESTAMP('2023-01-26 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(3, 72, 6460, TO_TIMESTAMP('2023-01-25 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(4, 72, 2284, TO_TIMESTAMP('2023-01-24 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(5, 33, 6545, TO_TIMESTAMP('2023-01-23 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(6, 33, 2984, TO_TIMESTAMP('2023-01-22 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(7, 29, 7634, TO_TIMESTAMP('2023-01-21 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(8, 29, 3763, TO_TIMESTAMP('2023-01-20 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(9, 18, 3576, TO_TIMESTAMP('2023-01-19 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(10, 18, 6297, TO_TIMESTAMP('2023-01-18 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(11, 13, 2535, TO_TIMESTAMP('2023-01-17 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));
insert into Transaction values(12, 13, 2380, TO_TIMESTAMP('2023-01-16 17:15:15', 'YYYY-MM-DD HH24:MI:SS'));

insert into Balance values(15, 54496, null);
insert into Balance values(72, 64190, null);
insert into Balance values(33, 63800, null);
insert into Balance values(29, 46116, null);
insert into Balance values(18, 78044, null);
insert into Balance values(13, 36881, null);

```