



Name: K.M. Tahlil Mahfuz Faruk

Student ID: 200042158

Department: CSE(SWE)

Course: CSE 4410

**Database Management System II LAB 4**

K.M. Tahlil Mahfuz Faruk  
Islamic University of Technology

## Create tables:

```
create table AccountProperty(  
    ID int primary key ,  
    name varchar(20),  
    ProfitRate numeric(10,2),  
    GracePeriod int  
);  
  
create table account(  
    ID int primary key,  
    name varchar(50),  
    AccCode int,  
    openingDate timestamp,  
    lastDateInterest timestamp,  
    foreign key (AccCode) references AccountProperty(ID)  
);  
  
create table transaction(  
    TID int primary key,  
    AccNo int,  
    Amount numeric(10,2),  
    transactionDate timestamp,  
    constraint fK_transaction foreign key (AccNo) references account(ID)  
);  
  
create table balance(  
    AccNo int primary key ,  
    PrincipleAmount numeric(10,4),  
    ProfitAmount numeric(10,4),  
    foreign key (AccNo) references account(ID)  
);
```

```
insert into accountProperty values(2002,'monthly',2.2,1);  
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(1,'tahlil',2002,sysdate-10212,sysdate-123142);  
insert into account values(2,'tahlil',3003,sysdate-123124,sysdate-41241);  
insert into account values(3,'tahlil',4004,sysdate-10000,sysdate-900);  
insert into account values(4,'tahlil',5005,sysdate-12312,sysdate-10);  
  
insert into transaction values(1,1,1000,sysdate-100000);  
insert into transaction values(2,2,1000,sysdate-100020);  
insert into transaction values(3,3,1000,sysdate-100040);  
insert into transaction values(4,4,1000,sysdate-100210);  
insert into transaction values(6,1,2000,sysdate-2314);  
  
insert into balance values(1,100,10);  
insert into balance values(2,100,10);  
insert into balance values(3,100,10);  
insert into balance values(4,100,10);
```

## SQL Commands:

In task 1,

```
-- A --
create or replace function
curr_balance (accountid int)
return numeric
As
    curr transaction.AMOUNT%type;
    principle balance.PrincipleAmount%type;
begin
    select sum(Amount) into curr
    from account natural join transaction
    where AccNo=accountid;

    select PrincipleAmount into principle
    from balance
    where AccNo=accountid;

    curr:=curr+principle;

    return curr;
end;
```

## Explanation:

- First find the cum of transaction amount that occurred by joining account and transaction.
- Then fetch principle amount from balance in principle variable.
- Just return the sum of principle and the current amount of transaction.

## Difficulties:

- No significant difficulty appeared.

In task 2,

```
-- B --
create or replace
type profit_tracking as object
(
    profit numeric(6,2),
    balance_bef_profit numeric(6,2),
    balance_after_profit numeric(6,2)
);
```

```
create or replace function
calculateProfit(accountid int)
return profit_tracking
is
    data profit_tracking;
    prof numeric;
    Bal_bef_profit numeric;
    Bal_after_prof numeric;
    grace_period int;
    openingdt date;
    balance numeric;
    prebalance numeric;
    profrate numeric;
    duration number;
    c int default 0;
    preprofit numeric;
begin
    Bal_bef_profit:=curr_balance( accountid: accountid);
    balance:=Bal_bef_profit;
    prebalance:=Bal_bef_profit;
    prof:=0;
    select GracePeriod,openingDate,ProfitRate into grace_period,openingdt,profrate
    from account,AccountProperty
    where account.ID=accountid and AccCode=AccountProperty.ID;

    duration:=sysdate-openingdt;

    loop
        if(duration>0) then
            if c=grace_period then
                prebalance:=prebalance+preprofit;
                c:=0;
                preprofit:=0;
            end if;
            prof:=prof+prebalance*(profrate/100);
            preprofit:=preprofit+prof;
            duration:=duration-30;
            c:=c+1;
        else
            exit;
        end if;
    end loop;
```

```

end loop;

prof:=prebalance-Bal_bef_profit;
Bal_after_prof:=balance;

data:=profit_tracking(prof,Bal_bef_profit,Bal_after_prof);

return data;

end;

```

### **Explanation:**

- First create profit\_tracking as object.
- Find the opening date, grace period and profit rate from account and accountproperty table.
- Then find the duration by subtracting the current data and the opening data.
- Then use a for loop until the duration is greater than 0.
- Each time the profit will be calculated in a variable called preprofit.
- After the grace period the profit amount saved in prof variable will be added to prebalance.
- Now save the values into data variable.
- Now send the data variable that is a profit\_tracking variable needs to be returned.

### **Difficulties:**

- To understand how the profit system works was the main challenge.
- The data variable was not returning if that is not initialized in the shown manner.

In task 3,

```
-- C --
create or replace procedure
tot_profit
as
    type amount is record(profit numeric);
    profit_table amount;

    data profit_tracking;
    cnt number default 0;
    total_profit numeric;

    cursor c is
        select unique id,openingDate
        from account;
begin
    for row in c loop
        data:=calculateProfit( accountid: row.ID);
        cnt:=cnt+1;
        total_profit:=total_profit+data.PROFIT;
    end loop;
    profit_table.profit:=total_profit;
    DBMS_OUTPUT.PUT_LINE( A: total_profit);
end;
```

### **Explanation:**

- First find the account number from account table.
- Then run a loop calling the calculateProfit function for each account number that will return profit\_tracking object.
- Just extract the profit portion for the object and add each time with total\_profit variable.
- Output the total\_profit variable.

### **Difficulties:**

- No such difficulties occurred during the task.