Islamic University of Technology
Lab Task 04
CSE 4410 : DBMS-II Lab

Submitted To :

Dr. Abu Raihan Mostofa Kamal
Professor, CSE Department
Islamic University of Technology

Zannatun Naim Sristy
Lecturer, CSE Department
Islamic University of Technology

Submitted By :

Sian Ashsad
ID : 200042151
Prog. : BSc in SWE
Dept. : CSE

Report :

Analysis : In this task, we need to write appropriate DDL code and PL/SQL procedure/function on those to solve the given questions.

Working code :

DDL code :

```
create table accountproperty(
    ap_id int,
    name VARCHAR2(100),
    profitrate numeric(2,1),
    graceperiod int,
    CONSTRAINT pk_id primary key (ap_id)
);


create table account(
    a_id int,
    name VARCHAR2(100),
    acccode int,
    openningdate date,
    lastdateinterest date,
    CONSTRAINT pk_id_acc primary key (a_id),
    CONSTRAINT fk_acc_prop FOREIGN key (acccode) REFERENCES
accountproperty(ap_id)
);

create table transaction(
    tid int,
    accno int,
    amount number,
    transactionDate date,
    CONSTRAINT pk_tid PRIMARY key (tid),
    CONSTRAINT fk_account FOREIGN key (accno) REFERENCES account(a_id)
);

create table balance(
    accno int,
    principalamount number,
```

```
    profitamount number,
    CONSTRAINT pk_accno primary key (accno),
    CONSTRAINT fk_account_id FOREIGN key (accno) REFERENCES account(a_id)
);
```

Entries :

```
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 (1, 'Sian', 2002, TO_DATE('2023-01-23',
'YYYY-MM-DD'), TO_DATE('2024-01-23', 'YYYY-MM-DD'));
insert into account values (2, 'Dihan', 3003, TO_DATE('2023-01-24',
'YYYY-MM-DD'), TO_DATE('2024-01-23', 'YYYY-MM-DD'));
insert into account values (3, 'Naz', 4004, TO_DATE('2023-01-25',
'YYYY-MM-DD'), TO_DATE('2024-01-23', 'YYYY-MM-DD'));
insert into account values (4, 'Nafisa', 5005, TO_DATE('2023-01-26',
'YYYY-MM-DD'), TO_DATE('2024-01-23', 'YYYY-MM-DD'));

insert into transaction values (1, 1, 1000,TO_DATE('2023-01-31',
'YYYY-MM-DD') );
insert into transaction values (2, 1, 1000,TO_DATE('2023-01-30',
'YYYY-MM-DD') );
insert into transaction values (3, 1, 1000,TO_DATE('2023-01-29',
'YYYY-MM-DD') );


insert into balance values (1, 6900, 0);
insert into balance values (2, 100670, 0);
insert into balance values (3, 10000, 0);
insert into balance values (4, 12000, 0);
```

PL/SQL Code ;

```
-->1 : principal amount in balance + all transactions made

create or replace
function current_balance(id number)
```

```
return number
is
    baseAmount number;
    balance number;
begin
balance := 0;
  select principalamount into baseAmount
    from balance
   where accno = id;
    balance := balance + baseAmount;
for row in (select amount from transaction where accno = id) loop
  balance := balance + row.amount;
end loop;

return balance;

end;
/

DECLARE
    amount number;
begin
    amount := current_balance(1);
  DBMS_OUTPUT.PUT_LINE(amount) ;
end;
/

-->2 : profit rate adds after the grace period has ended (yearly, biyearly
etc.)

create or replace
function GET_PROFIT(id number, fnBalance out number, fnProfit out number)
return varchar2
is
baseAmount number;
grace number;
rate number;
last_date date;
open_date date;
month number;
```

```
cnt number;
total_profit number;
final_balance number;
profit number;
output varchar2(5000);

begin

  baseAmount := 0;
  select principalamount into baseAmount
    from balance
   where accno = id;

    grace := 0;
    select accountproperty.graceperiod into grace
      from accountproperty ,account
     where accountproperty.ap_id = account.acccode and account.a_id = id;

     rate := 0;
     select accountproperty.profitrate into rate
      from accountproperty ,account
     where accountproperty.ap_id = account.acccode and account.a_id = id;

     select lastdateinterest into last_date, openningdate into open_date
       from account
      where account.a_id = id;

    month := TRUNC(MONTHS_BETWEEN(open_date, sysdate));
    cnt := 1;
    total_profit := 0;
    final_balance := baseAmount;

    for i in 1..month loop
      if cnt != grace then
        profit := profit + final_balance*rate;
      elsif cnt = grace then
        final_balance := final_balance + profit;
        total_profit := total_profit + profit;
        profit := 0;
        cnt := 0;
```

```
      end if;
      cnt := cnt + 1;

    end loop;
    fnBalance := final_balance;
    fnProfit := total_profit;
    output := 'Profit : ' || total_profit || ' Balance before profit : '
|| baseAmount || ' Balance after profit : ' || final_balance;
    return output;

end;
/

DECLARE
  output varchar2(1000);
  finalbalance number;
  finalprofit number;
begin
  output:= GET_PROFIT(1,finalbalance, finalprofit);
  DBMS_OUTPUT.PUT_LINE(output);
end;
/

-->3 : do for all account and update ammounts table

CREATE or replace
procedure  all_profit
AS
  output varchar2(1000);
  loop_id int;
  finalbalance number;
  finalprofit number;
  cursor c_profit
  is
    select a_id from account;
begin

  loop
    fetch c_profit into loop_id;
    exit when c_profit%notfound;
```

```
    output:= GET_PROFIT(loop_id,finalbalance, finalprofit);
    update balance set principalamount = finalbalance where accno =
loop_id;
    update balance set profitamount = finalprofit where accno = loop_id;
  end loop;

end;
/


begin
  all_profit();
end;
/
```

Explanation of the code :

1) The function create_balance takes in a number parameter ID and returns another number variable :- the current balance of the account by calculating all transactions made. There are two SQL queries present here. The first query gives the principal amount into a variable baseAmount. The baseAmount is added to another variable, balance. The next sql query is a multirow implicit cursor which gets the amount from the transaction table by matching the account id. The amount parsed is added to the balance variable. After the loop ends the balance variable is returned.
2) The function get_profit has 3 parameters :- input id, out parameter fnBalance and another out parameter fnProfit. It returns a varchar2 variable. Much like in task 1, the principal amount is fetched from the balance table into the baseAmount variable. Then the grace of the account is fetched by joining account and accountproperty table by id. In the same way, the rate is also fetched. Finally the account creation date is parsed from the account table. Now to find the months between the current date (sysdate) and the creation date of the account, we have to use the months_between and trunc function. The variable cnt is kept here to keep count of the months as we iterate through the upcoming loop. The variable total_profit keeps track of the net profit made by the account. The loop iterates till the count of the month that we figured out before. In the loop the cnt variable counts the number of months passed until the grace period is reached. Till then the profit of each month is stored by calculating from the rate and final_balance

(which was initialized with baseAmount). If cnt == grace period then the profit is added to the final_balance. The profit is also added to the total_profit variable. Then the profit variable and cnt variables are set to 0. After the iterations are complete, final_balance has the balance of the account with the profit made which is put into the fnBalance variable. Same thing is done for the fnProfit variable. Finally all the appropriate variables are put into a string and returned.

3) This task is an extension of the previous one. The procedure all_profit has no parameters. The cursor c_profit points towards the account id of the accounts table. In the procedure body the cursor id is used to update the balance table with appropriate profited values using the function in the previous task.

Problems :

- There was some ambiguity in the problem statement which led to difficulty understanding the problem scenario.
- Some modifications were needed to be made in the DDL to remove column ambiguity in my case.