Project Proposal

PROJECT NAME: BANKING SYSTEM

VIEW OF THE PROJECT:

- The main aim of this project is to give information about the Functioning of database in Banking System
- All the functionalities of Banking Systems will be credited in this project

ABOUT THE PROJECT:

Usually all persons want money for personal and commercial purposes. Banks are the oldest lending institutions in US scenario. They are providing all facilities to all citizens for their own purposes by their terms. To survive in this modern market every bank implements so many new innovative ideas, strategies, and advanced technologies. For that they give each and every minute detail about their institution and projects to Public.

They are providing ample facilities to satisfy their customers i.e. Net Banking, Mobile Banking, Door to Door facility, Instant facility, Investment facility, Demat facility, Credit Card facility, Loans and Advances, Account facility etc. And such banks get success to create their own image in public and corporate world. These banks always accept innovative notions in US banking scenario like Credit Cards, ATM machines, Risk Management etc.

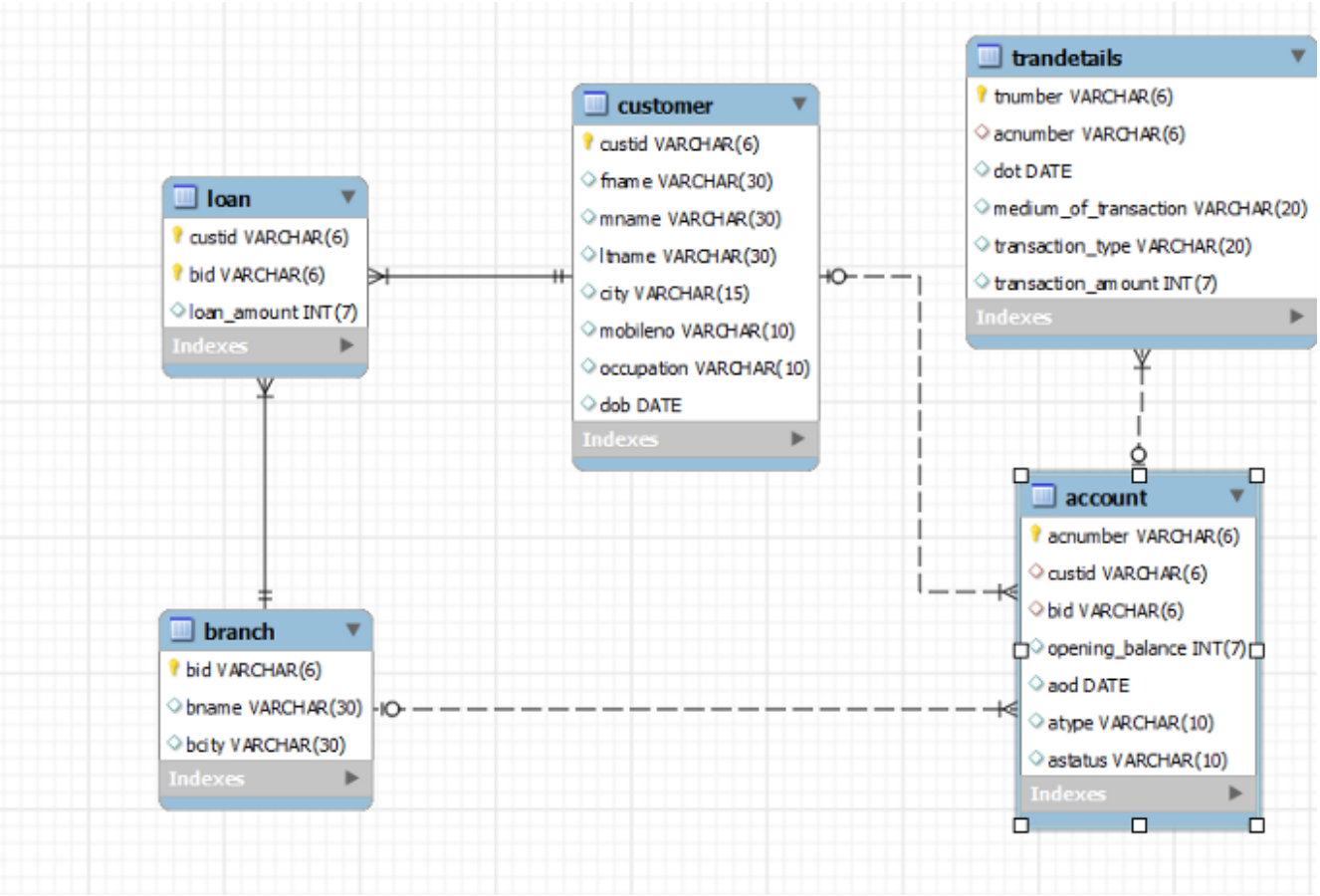
So, this must be the first choice for me to select this topic. At this stage every person must know about new innovation, technology of procedure new schemes and new ventures.

RULES GOVERNING THE PROJECT:

- All the customers of the bank have a unique account number
- The account numbers are not nullable i.e., they cannot take null values
- The customers must have a minimum account balance of Rs.1000
- Any customer is not allowed to withdraw amount from his account, if the withdrawal results in his account balance going below the minimum balance

- A person is eligible to get a loan from the bank if he has an account in the bank
- The percentage of interest imposed on the loan depends on the Company's policies
- The customers are issued cards (ATM or Debit) depending on his/her eligibilities
- All the card holders have unique card and PIN numbers
- The employees of the bank have unique identification numbers

ER Diagram-



User & Privileges-

```
SQL> CREATE USER bank IDENTIFIED by bank;

User created.

SQL> GRANT CONNECT TO bank;

Grant succeeded.

SQL> GRANT CONNECT, RESOURCE, DBA TO bank;

Grant succeeded.

SQL> GRANT CREATE SESSION GRANT ANY PRIVILEGE TO bank;
GRANT CREATE SESSION GRANT ANY PRIVILEGE TO bank
      *
ERROR at line 1:
ORA-00990: missing or invalid privilege


SQL> GRANT CREATE SESSION to bank;

Grant succeeded.

SQL> GRANT ANY PRIVILEGE TO bank;
GRANT ANY PRIVILEGE TO bank
      *
ERROR at line 1:
ORA-00990: missing or invalid privilege


SQL> GRANT PRIVILEGE TO bank;
GRANT PRIVILEGE TO bank
      *
```

Select Create Table-

```
CREATE TABLE bank.customer
AS (SELECT customer_id, name, accno, bal FROM OE.CUSTOMERS WHERE 1=0);
```

Script Output | Query Result

Task completed in 0.104 seconds

```
Table BANK.CUSTOMER created.
```

External Table-

```
CREATE TABLE emp_load
    (employee_number      CHAR(3),
     employee_last_name   CHAR(20),
     employee_middle_name CHAR(15),
     employee_first_name  CHAR(15)
     )
  ORGANIZATION EXTERNAL
    (TYPE ORACLE_LOADER
     DEFAULT DIRECTORY ext_tab_dir
     ACCESS PARAMETERS
       (RECORDS DELIMITED BY NEWLINE
        BADFILE DHHSMAPSIS:'EMP.BAD'
        LOGFILE DHHSMAPSIS:'EMP.LOG'
        FIELDS TERMINATED BY ','
       )
    LOCATION ('external_table_test2.dat')
)
REJECT LIMIT UNLIMITED;
```

Script Output | Query Result

Task completed in 0.149 seconds

```
Table EMP_LOAD dropped.


Table EMP_LOAD created.
```

```
CREATE DIRECTORY ext_tab_dir AS '/home';
GRANT READ ON DIRECTORY ext_tab_dir TO SCOTT
```

Script Output | Query Result

Task completed in 0.767 seconds

```
     LOCATION ( utcase1.txt )
    )
    REJECT LIMIT UNLIMITED
Error report -
ORA-30657: operation not supported on external organized table
30657.0000 -  "operation not supported on external organized table"
*Cause:    User attempted on operation on an external table which is
           not supported.
*Action:   Don't do that!

Directory EXT_TAB_DIR created.


Grant succeeded.
```

Sequence-

```
create sequence s1
start with 1
increment by 1
maxvalue 1000
minvalue 0
nocache
nocycle;
commit;
```

Script Output | Query Result | Task completed in 0.243 s

```
Commit complete.

Table KCB_TRAN_TAB created.

Commit complete.

Sequence S1 created.

Commit complete.
```

Table-

```
create table kcb_tran_tab
(
tid number,
accno number(20) references kcb_acc_tab(accn
trtype char(10) check(trtype in('d','w')),
dot date default sysdate,
amt number(7,2) check(amt>100)
)
;

commit;
```

Script Output | Query Result

Task completed in 0.144 seconds

```
Table KCB_ACC_TAB created.

1 row inserted.

Commit complete.

Table KCB_TRAN_TAB created.

Commit complete.
```

```
create table kcb_acc_tab
(accno number primary key,
name varchar2(20) constraint name_nn not null,
actype char check(actype in('s','c','fd')),
doo date default sysdate,
bal number(8,2) not null
)
;
```

Procedure-

Welcome Page | system | bank | bank~1

Worksheet | Query Builder

```
create or replace procedure upd_bal
(paccno kcb_acc_tab.accno%type,
pamt kcb_tran_tab.amt%type)
is
cbal kcb_acc_tab.bal%type;
begin
select bal into cbal from kcb_acc_tab where accno=paccno;
cbal:=cbal+pamt;
update kcb_acc_tab set bal=cbal where accno=paccno;
insert into kcb_tran_tab values(1001,paccno,'d',sysdate,pamt);
commit;
exception
when no_data_found then
DBMS_OUTPUT.PUT_LINE (paccno||'is not exists');
end upd_bal;
```

Script Output | Query Result

Task completed in 0.136 seconds

```
Procedure UPD_BAL compiled
```

Worksheet | Query Builder

```
select * from kcb_acc_tab;

select * from kcb_tran_tab;

execute upd_bal(37002167543,10000);
```

Script Output | Query Result

Task completed in 0.093 seconds

```
06550. 00000 -  "line %s, column %s:\n%s"
*Cause:    Usually a PL/SQL compilation error.
*Action:
SQL> execute  upd_bal (37002167544,5000);

PL/SQL procedure successfully completed.

SQL>
SQL> execute upd_bal(37002167543,10000);

PL/SQL procedure successfully completed.
```

```
select * from kcb_acc_tab;

select * from kcb_tran_tab;
```

Script Output | Query Result

SQL | All Rows Fetched: 7 in 0.004 seconds

	TID	ACCNO	TRTYPE	DOT	AMT
1	1	37002167546	d	25-APR-19	5000
2	3	37002167544	d	25-APR-19	5000
3	4	37002167544	d	25-APR-19	5000
4	5	37002167549	d	25-APR-19	10000
5	6	37002167544	w	25-APR-19	5000
6	7	37002167543	w	25-APR-19	10000
7	1001	37002167544	d	25-APR-19	5000

Transaction-

```
start transaction;
insert into kcb_acc_tab values(37002167543,'srinivas','s',sysdate,15000);
insert into kcb_acc_tab values(37002167544,'sayali','s',sysdate,25000);
insert into kcb_acc_tab values(37002167545,'shaishav','s',sysdate,35000);
insert into kcb_acc_tab values(37002167546,'saurabh','s',sysdate,45000);
insert into kcb_acc_tab values(37002167547,'sanket','s',sysdate,55000);
insert into kcb_acc_tab values(37002167548,'sailee','s',sysdate,75000);
insert into kcb_acc_tab values(37002167549,'pooja','s',sysdate,30000);
insert into kcb_acc_tab values(37002167550,'apurva','s',sysdate,57000);
commit;
```

Script Output × Query Result ×

Task completed in 0.526 seconds

```
1 row inserted.

1 row inserted.

1 row inserted.

Commit complete.
```

Procedure-

```
declare
i kcb_acc_tab%rowtype;
k kcb_tran_tab%rowtype;
begin
i.accno:=&accno;
k.trtype:='&trtype';
k.amt:=&amount;
select bal into i.bal from kcb_acc_tab
where accno=i.accno;
if k.trtype='D' then
i.bal:=i.bal+k.amt;
end if;
update kcb_acc_tab set bal=i.bal where accno=i.accno;
insert into kcb_tran_tab values(s1.nextval,i.accno,k.trtype,sysdate,k.amt);
commit;
end;
```

Script Output | Query Result

Task completed in 118.842 seconds

```
select bal into i.bal from kcb_acc_tab
where accno=i.accno;
if k.trtype='D' then
i.bal:=i.bal+k.amt;
end if;
update kcb_acc_tab set bal=i.bal where accno=i.accno;
insert into kcb_tran_tab values(s1.nextval,i.accno,k.trtype,sysdate,k.amt);
commit;
end;

PL/SQL procedure successfully completed.
```

```
declare
i kcb_acc_tab%rowtype;
k kcb_tran_tab%rowtype;
begin
i.accno:=&accno;
k.trtype:='&trtype';
k.amt:=&amount;
select bal into i.bal from kcb_acc_tab
where accno=i.accno;
if k.trtype='D' then
i.bal:=i.bal+k.amt;
end if;
update kcb_acc_tab set bal=i.bal where
insert into kcb_tran_tab values(s1.nex
commit;
end;
```

Enter Substitution Variable

Enter value for accno:

37002167546

OK Cancel

Script Output Query Result

ScriptRunner Task

```
     ACCNO NAME                 A DOO              BAL
---------- -------------------- - --------- ----------
3.7002E+10 srinivas             s 24-APR-19      15000
3.7002E+10 sayali               c 25-APR-19      25000
3.7002E+10 shaishav             s 25-APR-19      35000
3.7002E+10 saurabh              s 25-APR-19      45000
3.7002E+10 sanket               c 25-APR-19      55000
3.7002E+10 sailee               s 25-APR-19      75000
3.7002E+10 pooja                c 25-APR-19      30000
3.7002E+10 apurva               c 25-APR-19      57000
```

```
declare
i kcb_acc_tab%rowtype;
k kcb_tran_tab%rowtype;
begin
i.accno:=&accno;
k.trtype:='&trtype';
k.amt:=&amount;
select bal into i.bal from kcb_acc_tab
where accno=i.accno;
if k.trtype='D' then
i.bal:=i.bal+k.amt;
end if;
update kcb_acc_tab set bal=i.bal where
insert into kcb_tran_tab values(s1.nex
commit;
end;
```

Enter Substitution Variable

Enter value for trtype:

d

OK Cancel

Script Output Query Result

ScriptRunner Task

ACCNO	NAME	A	DOO	BAL
3.7002E+10	srinivas	s	24-APR-19	15000
3.7002E+10	sayali	c	25-APR-19	25000
3.7002E+10	shaishav	s	25-APR-19	35000
3.7002E+10	saurabh	s	25-APR-19	45000
3.7002E+10	sanket	c	25-APR-19	55000
3.7002E+10	sailee	s	25-APR-19	75000
3.7002E+10	pooja	c	25-APR-19	30000
3.7002E+10	apurva	c	25-APR-19	57000

```
declare
i kcb_acc_tab%rowtype;
k kcb_tran_tab%rowtype;
begin
i.accno:=&accno;
k.trtype:='&trtype';
k.amt:=&amount;
select bal into i.bal from kcb_acc_tab
where accno=i.accno;
if k.trtype='D' then
i.bal:=i.bal+k.amt;
end if;
update kcb_acc_tab set bal=i.bal where
insert into kcb_tran_tab values(sl.nex
commit;
end;
```

Enter Substitution Variable

Enter value for amount:

5000

OK Cancel

Script Output Query Result

ScriptRunner Task

```
     ACCNO NAME                 A DOO              BAL
---------- -------------------- - --------- ----------
3.7002E+10 srinivas             s 24-APR-19      15000
3.7002E+10 sayali               c 25-APR-19      25000
3.7002E+10 shaishav             s 25-APR-19      35000
3.7002E+10 saurabh              s 25-APR-19      45000
3.7002E+10 sanket               c 25-APR-19      55000
3.7002E+10 sailee               s 25-APR-19      75000
3.7002E+10 pooja                c 25-APR-19      30000
3.7002E+10 apurva               c 25-APR-19      57000
```

```
select * from kcb_tran_tab;

select * from kcb_acc_tab;
```

uery Result ×

SQL | All Rows Fetched: 4 in 0.012 seconds

	TID	ACCNO	TRTYPE	DOT	AMT
1	1	37002167546	d	25-APR-19	5000
2	3	37002167544	d	25-APR-19	5000
3	4	37002167544	d	25-APR-19	5000
4	5	37002167549	d	25-APR-19	10000

Worksheet | Query Builder

```
create or replace procedure upd_bal
(paccno kcb_acc_tab.accno%type,
pamt kcb_tran_tab.amt%type)
is
cbal kcb_acc_tab.bal%type;
begin
select bal into cbal from kcb_acc_tab where accno=paccno;
cbal:=cbal+pamt;
update kcb_acc_tab set bal=cbal where accno=paccno;
insert into kcb_tran_tab values(1001,paccno,'d',sysdate,pamt);
commit;
exception
when no_data_found then
DBMS_OUTPUT.PUT_LINE (paccno||'is not exists');
end upd_bal;
```

Script Output | Query Result

Task completed in 0.136 seconds

```
Procedure UPD_BAL compiled
```

View-

```sql
create or replace view account_details AS
select a.accno, t.tid, a.name, a.actype, a.bal
from kcb_acc_tab a
inner join kcb_tran_tab t
ON a.accno = t.accno;
```

Script Output | Query Result

Task completed in 0.22 seconds

```
PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> create or replace view account_details AS
  2  select a.accno, t.tid, a.name, a.actype, a.bal
  3  from kcb_acc_tab a
  4  inner join kcb_tran_tab t
  5  ON a.accno = t.accno;

View ACCOUNT_DETAILS created.
```

```
select * from account_details;
```

Script Output | Query Result

SQL | All Rows Fetched: 8 in 0.005 seconds

	ACCNO	TID	NAME	ACTYPE	BAL
1	37002167546	1	saurabh	s	45000
2	37002167544	3	sayali	c	30000
3	37002167544	4	sayali	c	30000
4	37002167549	5	pooja	c	40000
5	37002167544	6	sayali	c	30000
6	37002167543	7	srinivas	s	15000
7	37002167544	1001	sayali	c	30000
8	37002167543	1002	srinivas	s	15000

Function-

```
create or replace function chk_bal
(paccno kcb_acc_tab.accno%type,
pamt kcb_tran_tab.amt%type)
return boolean
is
cbal kcb_acc_tab.bal%type;
vatype kcb_acc_tab.actype%type;
begin
select actype,bal into vatype,cbal from kcb_acc_tab where
accno=paccno;
cbal:=cbal-pamt;
if vatype='s' and cbal<5000 then return(false);
elsif vatype='c'and cbal<10000 then
return(false);
else
return(true);
end if;
end chk_bal;
```

Script Output | Query Result

Task completed in 0.222 seconds

```
Function CHK_BAL compiled
```

Call function-

```
begin
if chk_bal(&accno,&amt)then
DBMS_OUTPUT.PUT_LINE ('it is validate');
else
DBMS_OUTPUT.PUT_LINE ('it is not valid
end if;
end;
```

Enter Substitution Variable

Enter value for accno:

37002167550

OK Cancel

Script Output | Query Result

ScriptRunner Task

```
it is validate

PL/SQL procedure successfully completed.
```

```
create or replace package pack_updbal
as
cbal kcb_acc_tab.bal%type;
procedure upd_bal(vaccno kcb_acc_tab.accno%type,
    vtype kcb_tran_tab.trtype%type,
    vamt kcb_tran_tab.amt%type);
function chk_bal(vaccno kcb_acc_tab.accno%type,
    vamt kcb_tran_tab.amt%type)
return boolean;
cbal kcb_acc_tab.bal%type;
end pack_updbal;
```

Script Output | Query Result

Task completed in 0.151 seconds

```
Package PACK_UPDBAL compiled
```

Trigger-

```
create or replace trigger trg_bal
before insert
on kcb_tran_tab
for each row
begin
if :new.trtype='d' then
pack_updbal.upd_bal(:new.accno,:new.trtype,:new.amt);
elsif   :new.trtype='w' then
if pack_updbal.chk_bal(:new.accno,:new.amt)then
pack_updbal.upd_bal(:new.accno,:new.trtype,:new.amt);
else
Raise_application_error(-20451,'the bal is too low so no transaction:');
end if;
end if;
exception
when no_data_found then
DBMS_OUTPUT.PUT_LINE (:new.accno||'is not exists');
end;
```

Script Output x Query Result x

Task completed in 0.187 seconds

```
Trigger TRG_BAL compiled
```

```
select * from kcb_tran_tab;
```

Script Output | Query Result

SQL | All Rows Fetched: 9 in 0.008 seconds

	TID	ACCNO	TRTYPE	DOT	AMT
1	1	37002167546	d	25-APR-19	5000
2	3	37002167544	d	25-APR-19	5000
3	4	37002167544	d	25-APR-19	5000
4	5	37002167549	d	25-APR-19	10000
5	6	37002167544	w	25-APR-19	5000
6	7	37002167543	w	25-APR-19	10000
7	1001	37002167544	d	25-APR-19	5000
8	1002	37002167543	d	25-APR-19	10000
9	1001	37002167543	d	25-APR-19	10000

```
DECLARE
emp_age number;

BEGIN

-- Finding employee age by date of birth
SELECT MONTHS_BETWEEN(TRUNC(sysdate), TO_DATE(:new.DATE_OF_BIRTH,'DD-MON-YYYY'))/12 INTO EMP_AGE FROM DUAL;

   -- Check whether employee age is greater than 18 or not
    IF (EMP_AGE < 18) THEN
      RAISE_APPLICATION_ERROR(-20000,'Employee age must be greater than or equal to 18.');
    END IF;

    -- Allow only past date of death
    IF(:new.DATE_OF_DEATH > sysdate) THEN
      RAISE_APPLICATION_ERROR(-20000,'Date of death can not be Future date.');
    END IF;

END;
/
```

Script Output | Query Result

Task completed in 0.625 seconds

```
14     -- Check whether employee age is greater than 18 or not
15      IF (EMP_AGE < 18) THEN
16        RAISE_APPLICATION_ERROR(-20000,'Employee age must be greater than or equal to 18.');
17      END IF;
18
19      -- Allow only past date of death
20      IF(:new.DATE_OF_DEATH > sysdate) THEN
21        RAISE_APPLICATION_ERROR(-20000,'Date of death can not be Future date.');
22      END IF;
23
24  END;
25  /

Trigger TRG_BEFORE_EMP_INSR compiled
```

```
create or replace TRIGGER trg_before_emp_insr
BEFORE INSERT
  on employee_details
  FOR EACH ROW

DECLARE
emp_age number;

BEGIN

-- Finding employee age by date of birth
SELECT MONTHS_BETWEEN(TRUNC(sysdate), TO_DATE(:new.DATE_OF_BIRTH,'DD-MON-YYYY'))/12 INTO EMP_AGE FROM DUAL;

   -- Check whether employee age is greater than 18 or not
    IF (EMP_AGE < 18) THEN
      RAISE_APPLICATION_ERROR(-20000,'Employee age must be greater than or equal to 18.');
    END IF;

END;
```

Script Output | Query Result

Task completed in 0.27 seconds

```
 8
 9  BEGIN
10
11  -- Finding employee age by date of birth
12  SELECT MONTHS_BETWEEN(TRUNC(sysdate), TO_DATE(:new.DATE_OF_BIRTH,'DD-MON-YYYY'))/12 INTO EMP_AGE FROM DUAL;
13
14     -- Check whether employee age is greater than 18 or not
15      IF (EMP_AGE < 18) THEN
16        RAISE_APPLICATION_ERROR(-20000,'Employee age must be greater than or equal to 18.');
17      END IF;
18
19  END;

Trigger TRG_BEFORE_EMP_INSR compiled
```

Procedure cursor-

```
create or replace procedure cursor
is cursor c1 is
select emp_id,first_name, last_name
from employee_details
where emp_id=1;
vemp_id employee_details.emp_id%type;
vfname employee_details.first_name%type;
vlname employee_details.last_name%type;

begin
open c1;
loop
fetch c1 into vemp_id,vfname,vlname;
exit when c1%notfound;
dbms_output.put_line(vemp_id||' '||vfname||' '||vlname);
end loop;
close c1;
end;
```

Script Output | Query Result

Task completed in 0.373 seconds

```
 7  vfname employee_details.first_name%type;
 8  vlname employee_details.last_name%type;
 9
10  begin
11  open c1;
12  loop
13  fetch c1 into vemp_id,vfname,vlname;
14  exit when c1%notfound;
15  dbms_output.put_line(vemp_id||' '||vfname||' '||vlname);
16  end loop;
17  close c1;
18  end;

Procedure CURSOR compiled
```

Index-

```
create index cust_name_index
on kcb_acc_tab(name);
```

Script Output | Query Result

Task completed in 0.519 seconds

```
SQL> create index cust_name_index
  2  on kcb_acc_tab(name);

Index CUST_NAME_INDEX created.
```

LPAD-

```
select lpad ('', (LEVEL-1*2), ' ') || employee_first_name employee_last_name
from emp_load
start with employee_last_name like 'K%'
connect by employee_first_name = employee_last_name;
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.007 seconds

	EMPLOYEE_LAST_NAME
1	Tom
2	Arya

Materialised View-

```
CREATE MATERIALIZED VIEW acc_trans_mv
BUILD DEFERRED REFRESH FAST ON DEMAND
ENABLE QUERY REWRITE AS
SELECT accno,trtype, SUM(amt)  AS Total
    FROM kcb_tran_tab
    GROUP BY trtype, accno;

create materialized view log on kcb_tran_tab with rowid(tid) including new
values;
```

Script Output | Query Result

Task completed in 0.411 seconds

```
Materialized view log KCB_TRAN_TAB created.
```

```
CREATE MATERIALIZED VIEW acc_trans_mv
BUILD DEFERRED REFRESH FAST ON DEMAND
ENABLE QUERY REWRITE AS
SELECT accno,trtype, SUM(amt)  AS Total
    FROM kcb_tran_tab
    GROUP BY trtype, accno;

create materialized view log on kcb_tran_tab with rowid(accno) including new
values;
```

Script Output | Query Result

Task completed in 0.434 seconds

```
Materialized view log KCB_TRAN_TAB created.
```

```
CREATE MATERIALIZED VIEW acc_trans_mv
 NOLOGGING
    BUILD IMMEDIATE
    REFRESH FAST ON COMMIT
    ENABLE QUERY REWRITE
    AS;
SELECT accno,trtype, SUM(amt)  AS Total
    FROM kcb_tran_tab
    GROUP BY trtype, accno;

create materialized view log on kcb_tran_tab with rowid(accno) including new
values;
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.006 seconds

	ACCNO	TRTYPE	TOTAL
1	37002167549	d	10000
2	37002167544	w	5000
3	37002167543	d	20000
4	37002167544	d	15000
5	37002167543	w	10000
6	37002167546	d	5000

Package-

```
create or replace package pack_updbal
as
cbal kcb_acc_tab.bal%type;
procedure upd_bal(vaccno kcb_acc_tab.accno%type,
    vtype kcb_tran_tab.trtype%type,
    vamt kcb_tran_tab.amt%type);
function chk_bal(vaccno kcb_acc_tab.accno%type,
    vamt kcb_tran_tab.amt%type)
return boolean;
cbal kcb_acc_tab.bal%type;
end pack_updbal;
```

APPENDIX-

KCB_ACC_TAB

```
 1  create table kcb_acc_tab
 2  (
 3  accno number primary key,
 4  name varchar2(20) constraint name_nn not null,
 5  actype char check(actype in('s','c','fd')),
 6  doo date default sysdate,
 7  bal number(8,2) not null
 8* )
QL> /

Table created.

QL> insert into kcb_acc_tab values(37002167543,'srinivas','s',sysdate,15000)
 2  /

 row created.

QL> commit
 2  /

commit complete.
```

KCB_TRAN_TAB

```
create table kcb_tran_tab
```

```
(
tid number,
accno number(20) references kcb_acc_tab(accno),
trtype char(10) check(trtype in('d','w')),
dot date default sysdate,
amt number(7,2) check(amt>100)
)
```

SEQUENCE

```
create sequence s1
start with 1
increment by 1
maxvalue 1000
minvalue 0
nocache
nocycle
```

1) Write a PL/SQL program to modify the balance after deposite the amt and insert the transaction details also.

Table based records- pl/sql block

```
declare
i kcb_acc_tab%rowtype;
k kcb_tran_tab%rowtype;
begin
i.accno:=&accno;
k.trtype:='&trtype';
k.amt:=&amount;
select bal into i.bal from kcb_acc_tab
where accno=i.accno;
if k.trtype='D' then
i.bal:=i.bal+k.amt;
end if;
update kcb_acc_tab set bal=i.bal+k.amt where accno=i.accno;
insert into kcb_tran_tab values(s1.nextval,i.accno,k.trtype,sysdate,k.amt);
commit;
end;
```

2) write a PL/SQL program for enter the transaction details perform the validation
i)if it is deposite update the bal and insert the transaction details
ii) if it is withdraw before withdraw
check the current bal if validationcontrol satisfy then only
perform the withdraw

Table based records- pl/sql block

```
declare
i kcb_acc_tab%rowtype;
k kcb_tran_tab%rowtype;
begin
   i.accno:=&accno;
   k.trtype:='&trtype';
   k.amt:=&amt;
   select actype,bal into i.actype,i.bal from kcb_acc_tab where accno=i.accno;
   if k.trtype='D' then
      i.bal:=i.bal+k.amt;
   else
      i.bal:=i.bal-k.amt;
   end if;
   if i.actype='s' and i.bal<5000 then
      Raise_application_error(-20456,'the bal is too low to perform transaction');
   end if;
   update kcb_acc_tab set bal=i.bal where accno=i.accno;
   insert into kcb_tran_tab values(s1.nextval,i.accno,k.trtype,sysdate,k.amt);
commit;
end;
```

PROCEDURE

```
create or replace procedure upd_bal
(paccno kcb_acc_tab.accno%type,
pamt kcb_tran_tab.amt%type)
is
cbal kcb_acc_tab.bal%type;
begin
select bal into cbal from kcb_acc_tab where accno=paccno;
cbal:=cbal+pamt;
update kcb_acc_tab set bal=cbal where accno=paccno;
insert into kcb_tran_tab values(1001,paccno,'d',sysdate,pamt);
commit;
exception
when no_data_found then
DBMS_OUTPUT.PUT_LINE (paccno||'is not exists');
end upd_bal;

create or replace procedure upd_bal
(paccno kcb_acc_tab.accno%type,
pamt kcb_tran_tab.amt%type)
is
cbal kcb_acc_tab.bal%type;
vatype kcb_acc_tab.atype%type;
begin
select acctype,bal into vatype,cbal from kcb_acc_tab where accno=paccno;
if upper(pttype)='d' then
cbal:=cbal+pamt;
elsif upper(pttype)='w' then
```

```
cbal:=cbal-pamt;
if value='s' and cbal<5000 then
Raise_application_error(-20456,'there is insufficient balance so we cannot do the transaction:');
end if;
end if;
update kcb_acc_tab set bal =cbal
where accno=paccno;
insert into kcb_tran_tab
values(101,paccno,ptrtype,sysdate,pamt);
commit;
exception
when no_data_found then
display(paccno||'is not exist');
end upd_bal;
```

FUNCTIONS
---
write a function the account holder is eligible for the withdraw or not

```
create or replace function chk_bal
(paccno kcb_acc_tab.accno%type,
pamt kcb_tran_tab.amt%type)
return boolean
is
cbal kcb_acc_tab.bal%type;
vatype kcb_acc_tab.actype%type;
begin
select actype,bal into vatype,cbal from kcb_acc_tab where
accno=paccno;
cbal:=cbal-pamt;
if vatype='s' and cbal<5000 then return(false);
elsif vatype='c'and cbal<10000 then
return(false);
else
return(true);
end if;
end chk_bal;
```

call this function with another pl/sql pgm with appropriate msg.

```
begin
if chk_bal(&accno,&amt)then
display('it is validate');
else
display('it is not validate');
end if;
end;
```

call this function in a procedure for the validation

```
create or replace procedure upd_bal
(paccno kcb_acc_tab.accno%type,
ptrtype kcb_tran_tab.trtype%type,
pamt kcb_acc_tab.amt%type)
is
cbal kcb_acc_tab.bal%type;
begin
select bal into cbal
from kcb_acc_Tab
where accno=paccno;
if upper(ptrtype)='D' then
cbal:=cbal+pamt;
elsif upper(ptrtype)='w' then
if chk_bal(paccno,pamt)then
cbal:=cbal-pamt;
else
Raise_application_error(-20456,'There  IB  so we cannot do the transaction:');
end if;
end if;
update kcb_acc_tab set bal=cbalwhere accno=paccno;
insert into kcb_tran_tab values(101,paccno,ptrtype,sysdate,pamt);
commit;
exception
when no_data_found then
display(paccno||'is not exist');
end upd_bal;
```

PACKAGES

PACKAGE SPECIFICATION

```
 create or replace package pack_updbal
as
cbal kcb_acc_tab.bal%type;
procedure upd_bal(vaccno kcb_acc_tab.accno%type,
   vtype kcb_tran_tab.trtype%type,
   vamt kcb_tran_tab.amt%type);
function chk_bal(vaccno kcb_acc_tab.accno%type,
   vamt kcb_tran_tab.amt%type)
return boolean;
cbal kcb_acc_tab.bal%type;
end pack_updbal;
```

PACKAGE BODY

```
create or replace package body pack_updbal
as
procedure upd_bal(vaccno kcb_acc_tab.accno%type,
                  vtrtype kcb_tran_tab.trtype%type,
vamt kcb_tran_tab.amt%type)
```

```
is
begin
select bal into cbal
from kcb_acc_tab
where accno=vaccno;
if upper(vtype)='w' then
cbal:=cbal_vamt;
end if;
update kcb_acc_tab set sal=cbal where accno=vaccno;
commit;
end upd_bal;
function chk_bal(vaccno kcb_acc_tab.accno%type,
                 vamt kcb_tran_tab.amt%type)
return boolean
is
vatype kcb_acc_tab.acctype%type;
begin
select acctype,bal into vatype,cbal from kcb_acc_tab where accno=vaccno;
cbal:=cbal-vamt; (global variable)
if vatype='s' and cbal<5000 then
return(false);
elsif vatype='c' and cbal<10000 then
return(false);
else
return(true);
end if;
end chk_bal;
end pack_updbal;
```

Triggers

```
create or replace trigger trg_bal
before insert
on kcb_tran_tab
for each row
begin
if :new.trtype='d' then
pack_updbal.upd_bal(:new.accno,:new.trtype,:new.amt);
elsif   :new.trtype='w' then
if pack_updbal.chk_bal(:new.accno,:new.amt)then
pack_updbal.upd_bal(:new.accno,:new.trtype,:new.amt);
else
Raise_application_error(-20451,'the bal is too low so no transaction:');
end if;
end if;
exception
when no_data_found then
display(:new.accno||'is not exists');
end;
```

PL/SQL:

It is a programming language which is developed by oracle company.
It is a procedural language it is used to process only a row at a time where as non procedural laguage process a set of rows at a time.

It support to execute a bloc of stmts at once.
Block: collection of executable statements.

struture of block:

```
Declare
[variable Declaration];
Begin
<executable statements>;
[exception
executable statements];
End;
```

```
declare cursor c1 is
  2  select empno,ename,sal
  3  from emp
  4  where deptno=30;
  5  vempno emp.empno%type;
  6  vename emp.ename%type;
  7  vsal emp.sal%type;
  8  begin
  9  open c1;
 10  loop
 11  fetch c1 into vempno,vename,vsal;
 12  exit when c1%notfound;
 13  dbms_output.put_line(vempno||' '||vename||' '||vsal);
 14  end loop;
 15  close c1;
 16* end;
```

```
1  declare
  2  cursor c_sal is
  3  select empno,ename,sal,job,deptno
  4  from emp;
  5  i emp%rowtype;
  6  begin
  7  open c_sal;
  8  loop
  9  fetch c_sal into i.empno,i.ename,i.sal,i.job,i.deptno;
 10  exit when c_sal%notfound;
 11  if i.job='clerk'then
 12  i.sal:=i.sal+i.sal*0.25;
 13  elsif i.job='manager'then
 14  i.sal:=i.sal+i.sal*0.35;
 15  else
 16  i.sal:=i.sal+i.sal*0.15;
 17  end if;
```

```
update emp set sal=i.sal
where empno=i.empno;
dbms_output.put_line(rpad(i.ename,8)||''||rpad(i.sal,6)||''||rpad(i.deptno,10));
```

```
declare
cursor
comm_cur is
select empno,ename,sal,comm,deptno
from emp;
begin
for k in comm_cur
loop
if k.comm is null then
 k.comm:=300;
 elsif k.comm=0 then
 k.comm:=250;
 else
 k.comm:=k.comm+k.sal*0.15;
 end if;
 update emp set comm=k.comm
 where empno=k.empno;
 dbms_output.put_line(rpad(k.empno,8)||''||k.comm||
 ''||k.deptno);
 end loop;
* end;
```

```
declare
cursor dnoc is
select deptno,min(sal) low_pay,
max(sal) high_pay,sum(sal) tot_pay,
count(empno)noe
from emp
group by deptno;
begin
for i in dnoc
 loop
 dbms_output.put_line(i.deptno||''||i.low_pay||''||i.high_pay||''||i.tot_pay||''||i.noe);
 end loop;
* end;
```

```
begin
for i in(select empno, ename,sal,deptno from emp)
loop
if i.deptno=10 then
i.sal:=i.sal+500;
elsif
  i.deptno=20 then
i.sal:=i.sal+600;
else
```

```
10   i.sal:=i.sal+700;
11  end if;
12  update emp set sal=i.sal
13  where empno=i.empno;
14  dbms_output.put_line(rpad(i.empno,8)||''||i.ename||''||i.sal||''||i.deptno);
15  end loop;
16* end;
```