**Procedure code :**

create or replace package pack\_admin\_add\_update

as

procedure proc\_add\_admin(aname adminn.name%type, mail adminn.email%type, phone adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type);

procedure proc\_update\_admin(aname adminn.name%type, mail adminn.email%type, phone adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type);

end pack\_admin\_add\_update;

create or replace package body pack\_admin\_add\_update

is

procedure proc\_add\_admin(aname adminn.name%type, mail adminn.email%type, phone adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type)

is

begin

insert into adminn (ADMINN\_ID,NAME,EMAIL,MOBILE,JOIN\_DATE,NID,PASSWORD) values (adminn\_id.nextval, aname, mail, phone, sysdate, nidd, pass);

end proc\_add\_admin;

procedure proc\_update\_admin(aname adminn.name%type, mail adminn.email%type, phone adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type)

is

begin

update adminn set name = aname, email = mail, mobile =phone, nid=nidd where password = pass;

end proc\_update\_admin;

end pack\_admin\_add\_update;

declare

aname adminn.name%type := 'anushri';

mail adminn.email%type:= 'anushree@gmail.com';

phone adminn.mobile%type := 01631365314;

nidd adminn.nid%type := 5132411122332;

pass adminn.password%type := 'Anu1@shree';

begin

pack\_admin\_add\_update.proc\_add\_admin(aname, mail, phone , nidd , pass );

pack\_admin\_add\_update.proc\_update\_admin(aname, mail, phone , nidd , pass );

end;

create or replace package emp\_grade

as

procedure emp\_grade\_up(id emp.e\_id%type, grd sal\_detail.grade%type);

procedure emp\_grade\_delete(id emp.e\_id%type);

procedure emp\_grade\_insert(id emp.e\_id%type, grd sal\_detail.grade%type);

end emp\_grade;

create or replace package body emp\_grade

is

function VALID\_EMPNO(id emp.e\_id%type)

return boolean

is

chk emp.e\_id%type;

begin

select e\_id into chk from emp where id = e\_id;

return true;

exception

when no\_data\_found then

return false;

end VALID\_EMPNO;

function VALID\_GRADE(grd salgrade.grade%type)

return boolean

is

chk salgrade.grade%type;

begin

select grade into chk from salgrade where grade = grd;

return true;

exception

when no\_data\_found then

return false;

end VALID\_GRADE;

procedure emp\_grade\_up(id emp.e\_id%type, grd sal\_detail.grade%type)

is

begin

if (VALID\_EMPNO(id)) then

if (VALID\_GRADE(grd)) then

update sal\_detail set grade=grd where e\_id=id;

dbms\_output.put\_line('Grade Updated');

else

dbms\_output.put\_line('Invalid Grade');

end if;

else

dbms\_output.put\_line('Employee ID does not exist');

end if;

end emp\_grade\_up;

procedure emp\_grade\_delete(id emp.e\_id%type)

is

begin

if (VALID\_EMPNO(id)) then

delete from sal\_detail where e\_id=id;

else

dbms\_output.put\_line('Employee ID does not exist');

end if;

end emp\_grade\_delete;

procedure emp\_grade\_insert(id emp.e\_id%type, grd sal\_detail.grade%type)

is

begin

if (VALID\_GRADE(grd)) then

insert into sal\_detail values(id, grd);

else

dbms\_output.put\_line('Invalid Grade');

end if;

end emp\_grade\_insert;

end emp\_grade;

declare

id emp.e\_id%type:=3;

grd sal\_detail.grade%type:=7;

begin

emp\_grade.emp\_grade\_insert(id,grd);

end;



create package pack\_sal\_update

as

procedure proc\_sal\_update(grad salgrade.grade%type, sal salgrade.salary%type);

end pack\_sal\_update;

create or replace package body pack\_sal\_update

is

function VALID\_GRADE(grad salgrade.grade%type)

return boolean

is

chk salgrade.grade%type;

begin

select grade into chk from salgrade where grade = grad;

return true;

exception

when no\_data\_found then

return false;

end VALID\_GRADE;

procedure proc\_sal\_update(grad salgrade.grade%type, sal salgrade.salary%type)

is

begin

if (VALID\_GRADE(grad)) then

update salgrade set salary = sal where grade = grad;

dbms\_output.put\_line('Salary Updated.');

else

dbms\_output.put\_line('This Grade does not exist');

end if;

end proc\_sal\_update;

end pack\_sal\_update;

declare

grad salgrade.grade%type:=1;

sal salgrade.salary%type:=50000;

begin

pack\_sal\_update.proc\_sal\_update(grad,sal);

end;

create package pack\_under\_tourist

as

procedure proc\_bkash\_user(id OUT tourist\_info.t\_id%type, tname OUT tourist\_info.name%type, temail OUT tourist\_info.email%type, tage OUT tourist\_info.age%type, tgender OUT tourist\_info.gender%type, tnid OUT tourist\_info.nid%type);

end pack\_under\_tourist;

create or replace package body pack\_under\_tourist

is

procedure proc\_bkash\_user(id OUT tourist\_info.t\_id%type, tname OUT tourist\_info.name%type, temail OUT tourist\_info.email%type, tage OUT tourist\_info.age%type, tgender OUT tourist\_info.gender%type, tnid OUT tourist\_info.nid%type)

is

cursor c1

is

select t\_id, name, email, age, gender, nid from tourist\_info where t\_id in (select t\_id from payment where pay\_type = 'bkash');

i number(25);

begin

for i in c1 loop

id := i.t\_id;

tname := i.name;

temail := i.email;

tage := i.age;

tgender := i.gender;

tnid := i.nid;

dbms\_output.put\_line(id||' '|| tname ||' '|| temail ||' '|| tage ||' '|| tgender||' '|| tnid);

end loop;

end proc\_bkash\_user;

end pack\_under\_tourist;

declare

id tourist\_info.t\_id%type;

tname tourist\_info.name%type;

temail tourist\_info.email%type;

tage tourist\_info.age%type;

tgender tourist\_info.gender%type;

tnid tourist\_info.nid%type;

begin

pack\_under\_tourist.proc\_bkash\_user(id, tname , temail , tage , tgender, tnid);

end;

**View code :**

CREATE OR REPLACE VIEW emp\_detail\_view AS

SELECT e\_id, NAME, email, age, gender

FROM emp

CREATE or replace VIEW admin\_detail\_view AS

SELECT adminn\_id, NAME, email, mobile, join\_date, nid

FROM adminn

CREATE VIEW tourist\_detail\_view AS

SELECT NAME, email, age, gender,nid

FROM emp

CREATE or replace VIEW Hpaid\_emp\_view AS

select e.e\_id, e.name, d.grade, g.salary

from sal\_detail d, salgrade g, emp e

where d.e\_id = e.e\_id and d.grade = g.grade and g.grade<=2

CREATE or Replace VIEW payment\_by\_user AS

SELECT t.t\_id, t.NAME, p.pay\_type, p.pay\_amount

FROM tourist\_info t, payment p where t.t\_id = p.t\_id order by t.t\_id

CREATE or Replace VIEW booked\_by\_user AS

SELECT t.t\_id, t.name, p.pay\_id, b.book\_type

FROM tourist\_info t, payment p, booking b where t.t\_id = p.t\_id and p.pay\_id = b.pay\_id order by t.t\_id

**Trigger code :**

create table admin\_reg\_log ( user\_name varchar(20), opt\_name varchar(20), opt\_date date);

create or replace trigger admin\_reg\_log

after insert on adminn

declare

opt varchar(30);

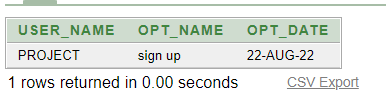
begin

opt := 'sign up';

insert into admin\_reg\_log values(user,opt,sysdate);

end;

select \* from admin\_reg\_log



2.

create or replace trigger emp\_reg\_secure

before insert on emp

declare

begin

if to\_char(sysdate,'HH24') not between '8' and '17' or to\_char(sysdate,'DAY') in('FRIDAY', 'SATURDAY') then

raise\_application\_error(-20202, 'Please come in regular working working hour');

END IF;

End;

3. create table emp\_reg\_log ( user\_name varchar(20), opt\_name varchar(20), opt\_date date);

create or replace trigger emp\_reg\_log

after insert on emp

declare

opt varchar(30);

begin

opt := 'sign up';

insert into emp\_reg\_log values(user,opt,sysdate);

end;

4. create table emp\_grade\_uplog ( e\_id number(20), old\_grade number(20), up\_grade number(20), up\_date date);

create or replace trigger emp\_grade\_up

after update of grade on sal\_detail

for each row

begin

insert into emp\_grade\_uplog values (:old.e\_id, :old.grade, :new.grade, sysdate);

end;

5. create or replace trigger sal\_alart

before update on salgrade

for each row

begin

if :new.salary < :old.salary then

raise\_application\_error(-20202, 'Warning: Updated salary must be greater then previous salary.');

END IF;

End;

update salgrade set salary = 40000 where grade = 1

6. create table emp\_sal\_uplog ( grade number(20), old\_salary number(20), updated\_salary number(20), update\_date date);

create or replace trigger emp\_sal\_up

after update of SALARY on salgrade

for each row

begin

insert into emp\_sal\_uplog values (:old.grade, :old.salary, :new.salary, sysdate);

end;