

DBMS Project

Topic: Buy and Sell of used products

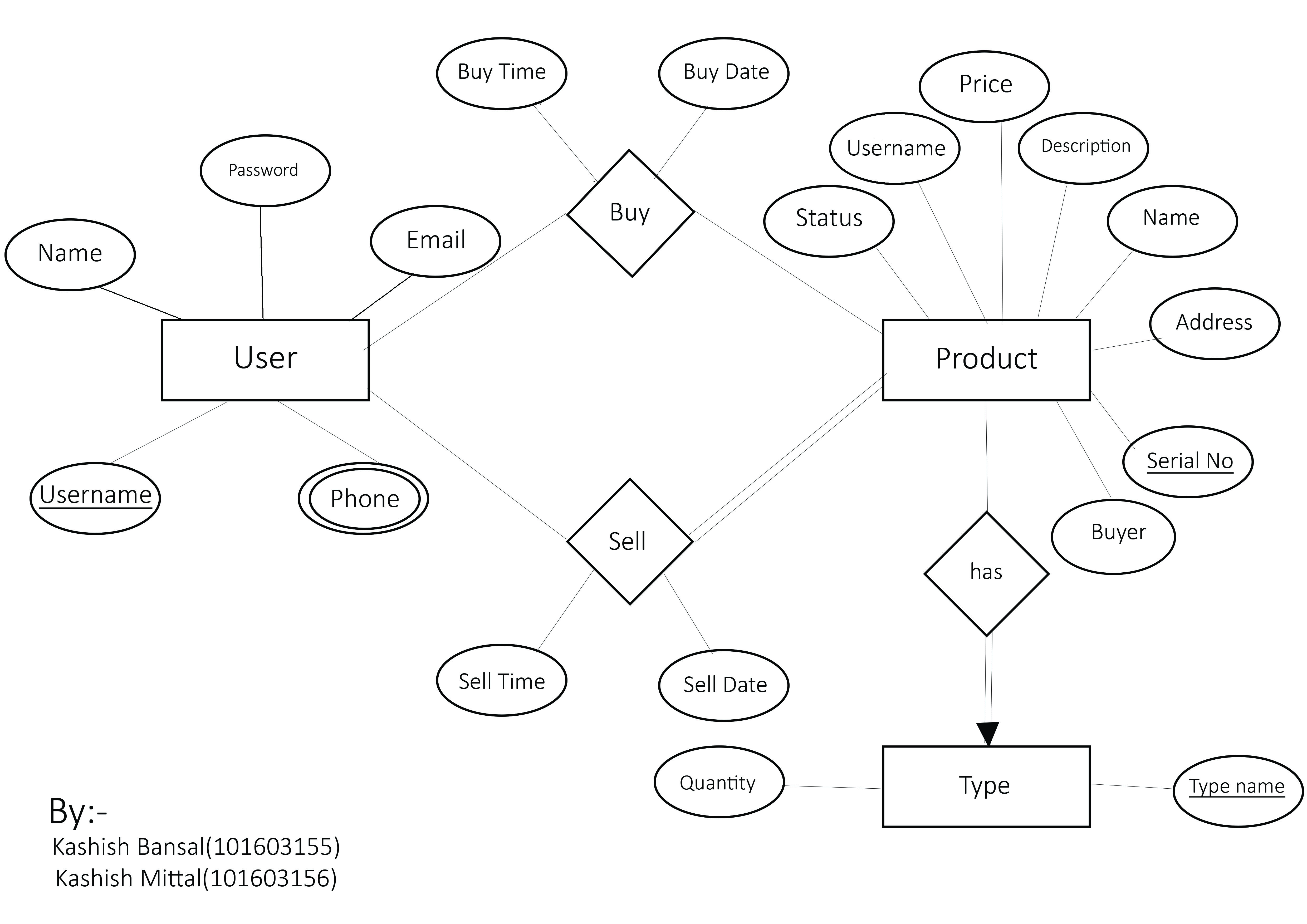
Submitted By:-

Kashish Bansal(101603155)(COE11)

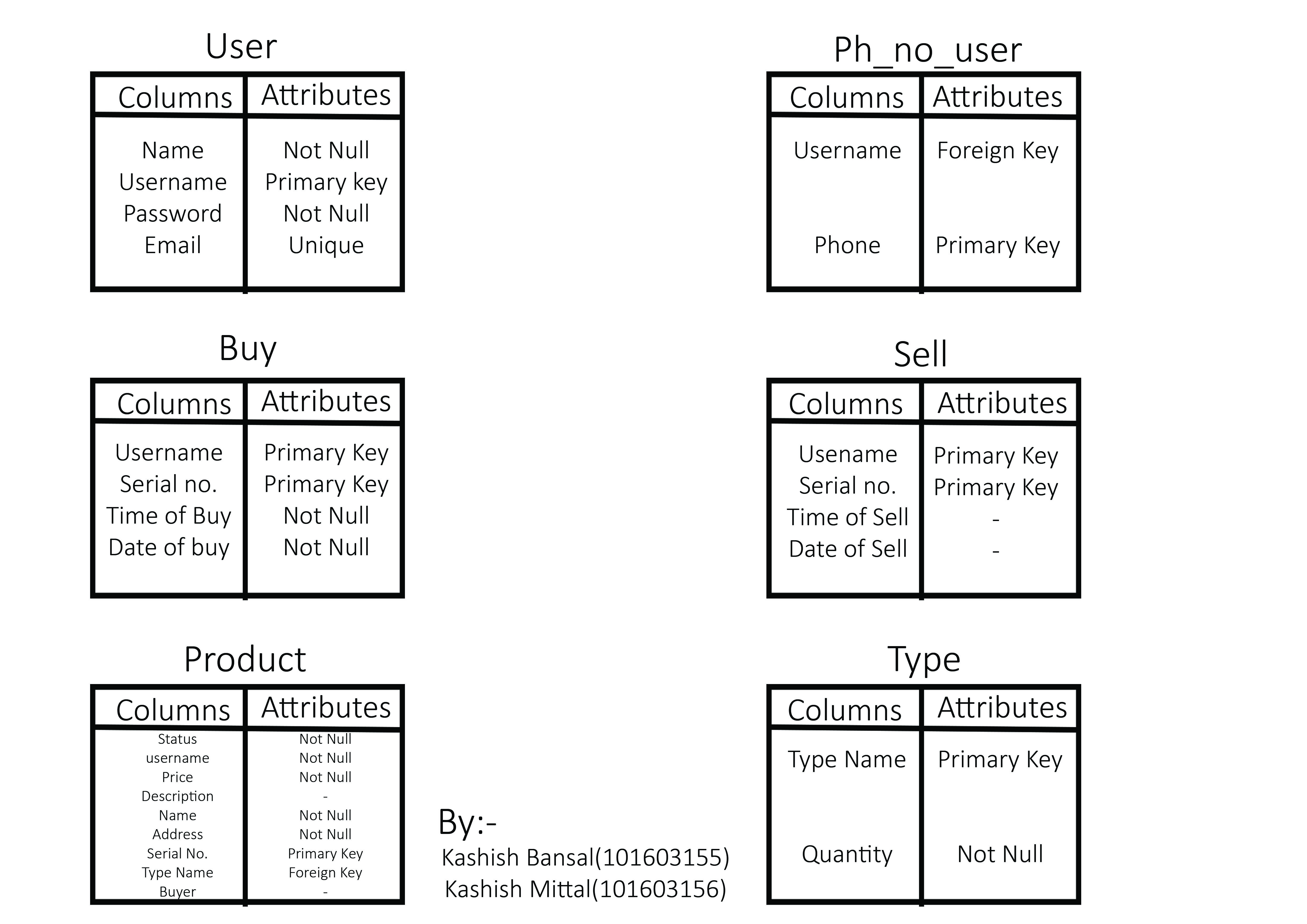
Kashish Mittal(101603156)(COE11)

Submitted to:-

Dr Parteek Bhatia

**ER Diagram**

**Normalized Tables**



**Code for creation of tables**

create table userid(

name varchar(30) not null,

username varchar(20) primary key,

password varchar(20) not null,

email varchar(15) unique

)

create table phone\_number(

username varchar(20) references userid(username) on delete cascade,

phone varchar(10) primary key

);

create table buy(

username varchar(20),

sr\_no number(5),

time\_of\_buy varchar(30) not null,

date\_of\_buy date not null,

primary key(username,sr\_no)

)

create table sell(

username varchar(20),

sr\_no number(5),

time\_of\_sell varchar(30) not null,

date\_of\_sell date not null,

primary key(username,sr\_no)

)

create table product(

sr\_no number(5) primary key,

pname varchar(30) not null,

type\_name varchar(30) references type(type\_name) on delete set null,

price number(10) not null,

description varchar(200),

status varchar(7) not null,

address varchar(100) not null,

username varchar(20) not null

buyer varchar(20);

);

create table type(

type\_name varchar(20) primary key,

quantity number(5) not null

);

**PL/SQL code for different operations**

Register:-

declare

name userid.name%type;

username userid.username%type;

password userid.password%type;

email userid.email%type;

phone1 phone\_number.phone%type;

phone2 phone\_number.phone%type;

phone3 phone\_number.phone%type;

begin

name:= :name;

username:= :username;

password:= :password;

email:= :email;

phone1:= :phone1;

phone2:= :phone2;

phone3:= :phone3;

insert into userid(name,username,password,email) values(name,username,password,email);

insert into phone\_number(username,phone) values(username,phone1);

if phone2 is not null then

insert into phone\_number(username,phone) values(username,phone2);

end if;

if phone3 is not null then

insert into phone\_number(username,phone) values(username,phone3);

end if;

dbms\_output.put\_line('you are registered');

exception

when dup\_val\_on\_index then

dbms\_output.put\_line('found duplicate info');

end;

See the info of user:-

declare

un userid.username%type:=:enter\_username;

cursor c1 is select \* from phone\_number where username=un;

row1 userid%rowtype;

row2 phone\_number%rowtype;

begin

select \* into row1 from userid where username=un;

dbms\_output.put\_line('name'||' '||'username'||' '||'email');

dbms\_output.put\_line(row1.name||' '||row1.username||' '||row1.email);

open c1;

dbms\_output.put\_line('phone numbers:-');

loop

fetch c1 into row2;

exit when c1%notfound;

dbms\_output.put\_line(row2.phone);

end loop;

close c1;

end;

Login:-

declare

usernamec userid.username%type;

passwordc userid.password%type;

namec userid.name%type;

begin

usernamec:= :username;

passwordc:= :password;

select name into namec from userid where username=usernamec and password=passwordc;

if namec is not null then

dbms\_output.put\_line('You are logged in.');

end if;

exception

when no\_data\_found then

dbms\_output.put\_line('invalid');

end;

Insertion of a type:-

declare

type\_name type.type\_name%type;

begin

type\_name:= :typename;

insert into type(type\_name,quantity) values(type\_name,0);

dbms\_output.put\_line('type uploaded');

exception

when dup\_val\_on\_index then

dbms\_output.put\_line('type already exist');

end;

Sequence for auto increment:-

CREATE SEQUENCE pro\_sr START WITH 1;

Trigger for insert of sr\_no:-

CREATE OR REPLACE TRIGGER proserial

BEFORE INSERT ON product

FOR EACH ROW

BEGIN

SELECT pro\_sr.NEXTVAL

INTO :new.sr\_no

FROM dual;

END;

Trigger for foriegn key(product table):-

CREATE OR REPLACE TRIGGER foriegnkey

BEFORE INSERT or update of type\_name ON product

FOR EACH ROW

declare

typename type.type\_name%type;

BEGIN

SELECT type\_name INTO typename FROM type

where type\_name= :new.type\_name;

exception

when no\_data\_found then

raise\_application\_error(-20001,'foriegnkeyviolate');

END;

Sell a product:-

declare

pname product.pname%type;

typename product.type\_name%type;

price product.price%type;

description product.description%type;

address product.address%type;

usename product.username%type;

uname product.username%type;

begin

pname:= :name;

typename:= :typename;

price:= :price;

description:= :description;

address:= :address;

usename:= :username;

select username into uname from userid where username=usename;

insert into product(pname,type\_name,price,description,address,status,username) values(pname,typename,price,description,address,'unsold',usename);

dbms\_output.put\_line('product uploaded');

update type

set quantity=quantity+1

where type\_name=typename;

exception

when no\_data\_found then

dbms\_output.put\_line('you are not registered.');

end;

Trigger sell:-

CREATE OR REPLACE TRIGGER selltrigger

after INSERT ON product

FOR EACH ROW

declare

un product.username%type;

sr product.sr\_no%type;

BEGIN

select username into un from userid where username=:new.username;

sr:=pro\_sr.nextval-1;

insert into sell(username,time\_of\_sell,date\_of\_sell,sr\_no) values(un,to\_char(sysdate,'hh24:mi:ss'),sysdate,sr);

END;

Search by category:-

declare

typename product.type\_name%type:=:typename;

cursor c1 is select \* from product where type\_name=typename;

row product%rowtype;

begin

open c1;

dbms\_output.put\_line('sr'||' '||'name'||' '||'type'||' '||'price'||' '||'desc'||' '||'status'||' '||'address'||' '||'seller');

loop

fetch c1 into row;

if row.pname is null then

dbms\_output.put\_line('not found');

end if;

exit when c1%notfound;

dbms\_output.put\_line(row.sr\_no||' '||row.pname||' '||row.type\_name||' '||row.price||' '||row.description||' '||row.status||' '||row.address||' '||row.username);

end loop;

close c1;

end;

My Products:-

declare

usern product.username%type:=:username;

cursor c1 is select \* from product where username=usern;

row product%rowtype;

begin

open c1;

dbms\_output.put\_line('sr'||' '||'name'||' '||'type'||' '||'price'||' '||'desc'||' '||'status'||' '||'address'||' '||'seller');

loop

fetch c1 into row;

if row.pname is null then

dbms\_output.put\_line('not found');

end if;

exit when c1%notfound;

dbms\_output.put\_line(row.sr\_no||' '||row.pname||' '||row.type\_name||' '||row.price||' '||row.description||' '||row.status||' '||row.address||' '||row.username);

end loop;

close c1;

end;

Display all products:-

declare

cursor c1 is select \* from product where status='unsold';

row product%rowtype;

begin

open c1;

dbms\_output.put\_line('sr'||' '||'name'||' '||'type'||' '||'price'||' '||'desc'||' '||'status'||' '||'address'||' '||'seller');

loop

fetch c1 into row;

if row.pname is null then

dbms\_output.put\_line('not found');

end if;

exit when c1%notfound;

dbms\_output.put\_line(row.sr\_no||' '||row.pname||' '||row.type\_name||' '||row.price||' '||row.description||' '||row.status||' '||row.address||' '||row.username);

end loop;

close c1;

end;

Buy a product:-

declare

sellbuy exception;

pragma exception\_init('buyer cannot be seller',-20005);

typename varchar(20);

buyuser product.buyer%type:=:buyer\_username;

selluser product.username%type;

srno product.sr\_no%type:=:serial\_no\_product;

begin

select type\_name into typename from product where sr\_no=srno;

select username into selluser from product where sr\_no=srno;

if selluser=buyuser then

raise sellbuy;

end if;

update product

set buyer=buyuser,

status='sold'

where sr\_no=srno;

update type

set quantity=quantity-1

where type\_name=typename;

exception

when no\_data\_found then

dbms\_output.put\_line('data mismatch');

when sellbuy then

dbms\_output.put\_line('buyer cannot be seller');

end;

Buy Trigger:-

CREATE OR REPLACE TRIGGER buytrigger

after update ON product

FOR EACH ROW

declare

un product.buyer%type;

sr product.sr\_no%type;

BEGIN

select username into un from userid where username=:new.buyer;

sr:=:new.sr\_no;

insert into buy(username,time\_of\_buy,date\_of\_buy,sr\_no) values(un,to\_char(sysdate,'hh24:mi:ss'),sysdate,sr);

END;

My orders:-

declare

usern product.username%type:=:username;

cursor c1 is select \* from product where buyer=usern;

row product%rowtype;

begin

open c1;

dbms\_output.put\_line('sr'||' '||'name'||' '||'type'||' '||'price'||' '||'desc'||' '||'status'||' '||'address'||' '||'seller');

loop

fetch c1 into row;

if row.pname is null then

dbms\_output.put\_line('not found');

end if;

exit when c1%notfound;

dbms\_output.put\_line(row.sr\_no||' '||row.pname||' '||row.type\_name||' '||row.price||' '||row.description||' '||row.status||' '||row.address||' '||row.username);

end loop;

close c1;

end;

Change info:-

declare

namen userid.name%type;

userna userid.username%type;

passwordn userid.password%type;

emailn userid.email%type;

begin

userna:= :old\_username;

namen:= :name;

passwordn:= :password;

emailn:= :email;

update userid

set name=namen,

password=passwordn,

email=emailn

where username=userna;

dbms\_output.put\_line('info updated');

end;

Delete product:-

declare

srno product.sr\_no%type:=:serial\_no\_product;

un product.username%type:=:your\_username;

pass userid.password%type:=:your\_password;

st product.status%type;

un2 userid.username%type;

begin

select username into un2 from userid where username=un and password=pass;

if un2 is not null then

select status into st from product where sr\_no=srno and username=un;

if st='unsold' then

delete from product where sr\_no=srno;

else

dbms\_output.put\_line('cannot delete as already sold.');

end if;

else

dbms\_output.put\_line('enter correct username and password');

end if;

exception

when no\_data\_found then

dbms\_output.put\_line('enter valid data');

end;

Cancel order:-

declare

srno product.sr\_no%type:=:serial\_no\_product;

un product.username%type:=:your\_username;

pass userid.password%type:=:your\_password;

st product.status%type;

un2 userid.username%type;

begin

select username into un2 from userid where username=un and password=pass;

if un2 is not null then

select status into st from product where sr\_no=srno and buyer=un;

if st='sold' then

delete from product where sr\_no=srno;

else

dbms\_output.put\_line('cannot delete as not bought');

end if;

else

dbms\_output.put\_line('enter correct username and password');

end if;

exception

when no\_data\_found then

dbms\_output.put\_line('enter valid data');

end;

procedure for insertion of type:-

create or replace procedure insert\_type(type in type.type\_name%type)

as

begin

insert into type(type\_name,quantity) values(type,0);

end;

Insertion of a type:-

declare

type\_name type.type\_name%type;

begin

type\_name:= :typename;

insert\_type(type\_name);

dbms\_output.put\_line('type uploaded');

exception

when dup\_val\_on\_index then

dbms\_output.put\_line('type already exist');

end;

function for increment of quantity:-

CREATE or replace function incquan(quantity in type.quantity%type) return number is quantitynew number;

begin

quantitynew:=quantity+1;

return(quantitynew);

end;

Sell a product using function:-

declare

pname product.pname%type;

typename product.type\_name%type;

price product.price%type;

description product.description%type;

address product.address%type;

usename product.username%type;

uname product.username%type;

begin

pname:= :name;

typename:= :typename;

price:= :price;

description:= :description;

address:= :address;

usename:= :username;

select username into uname from userid where username=usename;

insert into product(pname,type\_name,price,description,address,status,username) values(pname,typename,price,description,address,'unsold',usename);

dbms\_output.put\_line('product uploaded');

update type

set quantity=incquan(quantity)

where type\_name=typename;

exception

when no\_data\_found then

dbms\_output.put\_line('you are not registered.');

end;

