Table creation:

create table **depository**(

dp\_name char(20) primary key,

depository char(5) not null

);

create table **dp\_id**(

dp\_name char(20) not null,

dp\_id number(5) primary key,

foreign key(dp\_name) references depository(dp\_name),

check(dp\_id > 0)

);

create table **company\_id**(

id number(5) primary key,

c\_name char(50) not null,

check(id>0)

);

create table **company\_code**(

code char(3) primary key,

c\_name char(50) not null

);

create table **client**(

client\_id number(5) primary key,

dp\_id number(5),

id\_prf char(20),

first\_name char(20) not null,

middle\_name char(20),

last\_name char(20) not null,

pin\_code number(6),

nationality char(30),

city char(20),

street char(30),

h\_no char(20),

foreign key (dp\_id) references dp\_id(dp\_id),

check(id\_prf in('license', 'adhaar card', 'voters card', 'passport')),

check(pin\_code>0)

);

create table **phone**(

client\_id number(5),

ph\_no number(10),

foreign key (client\_id) references client(client\_id),

check(ph\_no > 0),

primary key(client\_id, ph\_no)

);

create table **account**(

client\_id number(5),

acc\_no number(10),

foreign key (client\_id) references client(client\_id),

primary key(client\_id, acc\_no)

);

create table **email\_id**(

client\_id number(5),

email char(30),

foreign key (client\_id) references client(client\_id),

primary key(client\_id, email)

);

create table **id\_no**(

client\_id number(5) ,

id\_prf\_no number(20),

foreign key(client\_id) references client(client\_id),

primary key(client\_id, id\_prf\_no),

check(id\_prf\_no > 0)

);

create table **pan**(

client\_id number(5) ,

pan\_no number(20),

foreign key(client\_id) references client(client\_id),

primary key(client\_id, pan\_no),

check(pan\_no > 0)

);

create table **tax**(

client\_id number(5) primary key,

bkg number(5),

s\_tax number(5),

foreign key(client\_id) references client(client\_id),

check (bkg > 0 and s\_tax > 0)

);

create table **transaction**(

t\_id number(5) constraint pk\_transaction primary key,

client\_id number(5) not null,

code char(3),

date\_ date,

time timestamp,

type char(10),

buy\_p number(10) default 0,

sell\_p number(10) default 0,

qty number(10),

constraint fk\_transaction foreign key(code) references company\_code(code),

check(type in('cash','future')),

);

Drop tables:

drop table transaction;

drop table tax;

drop table client;

drop table pan;

drop table id\_no;

drop table comapany\_code;

drop table company\_id;

drop table depository;

drop table dp\_id;

drop table account;

drop table phone;

drop table email;

Value insertion:

Insert into

Triggers and Exceptions:

create or replace trigger **transaction\_restrict**

before insert or update or delete

on transaction

begin

if (((upper(to\_char(sysdate,'DAY')))='SATURDAY') or (upper(to\_char(sysdate,'DAY'))='SUNDAY')) then

raise\_application\_error(-20001,'Database cannot be modified today');

end if;

exception

when others then

dbms\_output.put\_line('Error encountered');

end;

create or replace trigger **phone\_no**

before

insert or update of ph\_no

on phone

for each row

begin

if(length(:new.ph\_no)<>10) then

raise\_application\_error(-20002, 'Incorrect phone number');

end if;

exception

when others then

dbms\_output.put\_line('Error Encountered');

end;

create or replace trigger **buy\_sell\_qty**

before

insert or update on transaction

for each row

begin

if((:new.buy\_p <0) or (:new.sell\_p <0) or (:new.qty <0))then

raise\_application\_error(-20003, 'Negative Values Entered');

end if;

exception

when others then

dbms\_output.put\_line('Error Encountered');

end;

--**with cursor**

create or replace trigger **fk\_transaction\_client**

before update or insert of client\_id

on transaction

for each row

declare

cursor c1(a number) is select \* from client where client\_id=a ;

begin

open c1(:new.client\_id);

if (c1%notfound) then

raise\_application\_error(-20004, 'Foreign key violated' );

end if;

close c1;

end;

create or replace trigger **upper\_c\_name**

before insert or update of c\_name on company\_code

begin

:new.c\_name:= upper(:new.c\_name);

end;

create or replace trigger **client\_name** after update of first\_name on client for each row

begin

dbms\_output.put\_line(:old.first\_name || ' is now ' || :new.first\_name);

end;

Drop Triggers:

drop trigger transaction\_restrict ;

drop trigger phone\_no;

drop trigger buy\_sell\_qty;

drop trigger fk\_transaction\_client;

drop trigger upper\_c\_name;

drop trigger client\_name;

Procedures and Exceptions:

create or replace procedure **insert\_depository**(a in char, b in char) is

begin

insert into depository values(a, b);

exception

when others then

dbms\_output.put\_line('Error Encountered');

end;

begin

insert\_depository(‘abc’, ‘xyz’);

end;

create or replace procedure **insert\_company\_id**(a in number, b in char) is zero\_entered exception;

begin

if (a=0) then

raise zero\_entered;

end if;

insert into company\_id values(a,b);

exception

when zero\_entered then

dbms\_output.put\_line('Company id cannot be 0');

end;

create or replace procedure **delete\_client**(a in number) is

begin

delete from client where client\_id= a;

exception

when no\_data\_found then

dbms\_output.put\_line('No data');

when others then

dbms\_output.put\_line('Error Encountered');

end;

create or replace procedure **delete\_transaction**(a in char) is illegal\_value exception;

begin

if (length(a)>3) then

raise illegal\_value;

end if;

delete from transaction where code=a;

exception

when illegal\_value then

dbms\_output.put\_line('Code length must not be greater than 3');

when others then

dbms\_output.put\_line('Error Encountered');

end;

create or replace procedure **update\_bkg**(a in number, b in number) is illegal\_value exception;

begin

if (length(a)>5 or length(b)>5) then

raise illegal\_value;

end if;

update tax set bkg=b where client\_id=a ;

exception

when illegal\_value then

dbms\_output.put\_line('Length must not be greater than 5');

when others then

dbms\_output.put\_line('Error Encountered');

end;

--**with cursor**

create or replace procedure client\_city(a in char) is cursor c1(b char) is select \* from client where upper(city)=upper(b); rec client%rowtype; no\_record exception;

begin

for rec in c1(a) loop

if(c1%notfound) then

raise no\_record;

end if;

dbms\_output.put\_line(rec.first\_name || ' ' || rec.last\_name || ' ' || rec.client\_id);

end loop;

exception

when no\_record then

dbms\_output.put\_line('No such record');

when others then

dbms\_output.put\_line('Error Encountered');

end;

--**with cursor**

create or replace procedure client\_phone(a in char) is cursor c1(b char)is select client\_id from transaction where upper(code)=upper(b); cursor c2(c number) is select ph\_no from phone where client\_id=c and rownum=1; c\_id transaction.client\_id%type ; pno phone.ph\_no%type;

begin

for c\_id in c1(a) loop

for pno in c2(c\_id) loop

dbms\_output.put\_line(pno);

end loop;

end loop;

exception

when others then

dbms\_output.put\_line('Error Encountered');

end;

Functions and Exceptions:

create or replace function **profit**(a in number) return number is diff number(10):=0 ; cursor c1(b number) is select \* from transaction where client\_id= b; rec transaction%rowtype; f number

begin

for rec in c1(a) loop

diff:= diff+ ( -rec.buy\_p + rec.sell\_p)\*rec.qty;

end loop;

select bkg/100 into f from tax where client\_id=a;

diff:= diff- diff\*(f);

diff:= diff- diff\*(select s\_tax/100 from tax where client\_id=a);

return diff;

exception

when others then

dbms\_output.put\_line('Error Encountered');

end;