create table Books

(

book\_id int NOT NULL Primary Key,

book\_name varchar2(200),

edition varchar2(50),

isbn int ,

language varchar2(50),

price number(6),

discount number (6),

page int,

cat\_name varchar2(50),

subcat\_name varchar2(50),

author\_id int,

publisher\_id int

);

create sequence seq\_book\_id

start with 10000

increment by 1

minvalue 1

maxvalue 99999

cache 10;

insert into books values (seq\_book\_id.nextval, 'Digital Logic Design', '2nd', 44356751,'English', 250, 8, 220, 'Engineering', 'Electrical Engineering', 501, 2001);

insert into books values (seq\_book\_id.nextval, 'Introduction to Database', '5th', 33256751,'English', 300, 7, 670, 'Science', 'Computer Science', 502, 2001);

select \* from books;

drop table books;

drop sequence seq\_book\_id;

DELETE FROM books WHERE book\_id=10000;

UPDATE books

SET book\_name = 'Advanced Database Book', price='350',CAT\_NAME= 'Science',SUBCAT\_NAME='CS'

WHERE book\_ID = 10002;

----------------------------------------------------------------

create table users

(

user\_id int NOT NULL PRIMARY KEY,

user\_name varchar2(50),

user\_email varchar2(50),

user\_phone varchar2(14),

password varchar2(15),

user\_address varchar2(100),

user\_district varchar2(20),

user\_city varchar2(20),

created\_date date

);

create sequence seq\_user\_id

start with 1

increment by 1

minvalue 1

maxvalue 9999

nocache;

insert into users values(seq\_user\_id.nextval, 'mehedi', 'hmehedi995@gmail.com', '01764760509','1234','uttora,dhaka','dhaka','dhaka','21-apr-2018');

select \* from users;

drop table users;

drop sequence seq\_user\_id;

DELETE FROM users WHERE user\_id=3;

UPDATE users

SET user\_name = 'rimu', user\_phone='0176666626',user\_email='rimu@yahoo.com'

WHERE user\_ID = 2;

---------------------------------------------------------------------------------

create table authors

(

author\_id int not null primary Key,

author\_name varchar2(50),

author\_email varchar2(30),

author\_phone varchar2(14),

author\_social\_id varchar2(100)

);

create sequence seq\_author\_id

start with 500

increment by 5

minvalue 500

maxvalue 5000

nocache;

insert into authors values(seq\_author\_id.nextval,'Donal Norman','norman@gmail.com','01700000001','https://www.facebook.com/don.norman.18');

select \* from authors;

drop table authors;

drop sequence seq\_author\_id;

DELETE FROM authors WHERE author\_id=510;

UPDATE authors

SET author\_name = 'Pressman',author\_email='pressman@gmail.com', author\_social\_id='https://www.facebook.com/pressman.18'

WHERE author\_ID = 505;

----------------------------------------------------------------------------------------------

create table publishers

(

publisher\_id int not null primary Key,

publisher\_name varchar2(50),

publisher\_email varchar2(30),

publisher\_phone varchar2(14),

publisher\_address varchar(100),

publisher\_social\_id varchar2(100)

);

create sequence seq\_publisher\_id

start with 1000

increment by 10

minvalue 1000

maxvalue 2000

nocache;

insert into publishers values(seq\_publisher\_id.nextval,'Rimu Publications','rimu@publications.com','01702300001','BanglaBazar, Dhaka','https://www.facebook.com/rimuahmmed.raj');

select \* from publishers;

drop table publishers;

drop sequence seq\_publisher\_id;

DELETE FROM publishers WHERE publisher\_id=1010;

UPDATE publishers

SET publisher\_name = 'Sieam Publications',publisher\_email='sieam@gmail.com', publisher\_social\_id='https://www.facebook.com/sieam'

WHERE publisher\_ID =1020;

----------------------------------------------------------------------------

create table carts

(

cart\_id int not null primary key,

cart\_quantity number,

book\_id int,

user\_id int

);

create sequence seq\_cart\_id

start with 50000

increment by 1

minvalue 50000

maxvalue 100000

nocache;

insert into carts values (seq\_cart\_id.nextval, 1, 10001, 1);

select \* from carts;

drop table carts;

drop sequence seq\_cart\_id;

DELETE FROM carts WHERE cart\_id=50002;

UPDATE carts

SET cart\_quantity=5

WHERE cart\_ID =50001;

------------------------------------------------------------------------

create table orders

(order\_id int not null primary key,

status varchar2(15),

price number,

total number,

due number,

pay\_type varchar2(20),

order\_date varchar2(30),

user\_id int,

cart\_id int

);

create sequence seq\_order\_id

start with 30000

increment by 1

minvalue 30000

maxvalue 100000

nocache;

insert into orders values (seq\_order\_id.nextval, 'abcd', 500,500, 0, 'cash\_on\_delivery', '21-apr-2018',1,50001);

select \* from orders;

drop table orders;

drop sequence seq\_order\_id;

DELETE FROM orders WHERE order\_id=30002;

UPDATE orders

SET STATUS='mnop',PAY\_TYPE='cash'

WHERE order\_ID =30001;

-------------------------------------------------------------

create table order\_details

(

order\_de\_id int not null primary key,

order\_de\_quantity number,

order\_id int,

book\_id int,

user\_id int

);

create sequence seq\_order\_de\_id

start with 70000

increment by 1

minvalue 70000

maxvalue 80000

nocache;

insert into order\_details values(seq\_order\_de\_id.nextval, 3,30001,10001,1);

select \* from order\_details;

drop table order\_details;

drop sequence seq\_order\_de\_id;

DELETE FROM order\_details WHERE order\_de\_id=70002;

UPDATE order\_details

SET ORDER\_DE\_QUANTITY=9

WHERE order\_de\_id =70001;

------------------------------------------------------------------------------

create table categorys

(

category\_id int NOT NULL primary key ,

category\_name varchar2(50)

);

create sequence seq\_category\_id

start with 3500

increment by 1

minvalue 3500

maxvalue 4500

nocache;

insert into categorys values(seq\_category\_id.nextval,'science');

select \* from categorys;

drop table categorys;

drop sequence seq\_category\_id;

DELETE FROM categorys WHERE category\_id=3501;

UPDATE categorys

SET category\_name='Arts'

WHERE category\_ID =3502;

---------------------------------------------------------------------------

create table sub\_categorys

(

sub\_category\_id int NOT NULL primary key ,

category\_id int,

sub\_category\_name varchar2(50)

);

create sequence seq\_sub\_category\_id

start with 4500

increment by 1

minvalue 4500

maxvalue 5500

nocache;

insert into sub\_categorys values(seq\_sub\_category\_id.nextval,3500,'computer science');

select \* from sub\_categorys;

drop table sub\_categorys;

drop sequence seq\_sub\_category\_id;

DELETE FROM sub\_categorys WHERE sub\_category\_id=4500;

UPDATE sub\_categorys

SET sub\_category\_name='Eco arts'

WHERE sub\_category\_ID =4502;

----------------------------------------------------------------------------------------

create table reviews

(review\_id int not null primary key,

book\_id int,

user\_id int,

description varchar2(200),

review\_date varchar2(30)

);

create sequence seq\_review\_id

start with 5500

increment by 1

minvalue 5500

maxvalue 10500

nocache;

insert into reviews values(seq\_review\_id.nextval,10001,1,'Programming Language Book', '21-apr-2018');

select \* from reviews;

drop table reviews;

drop sequence seq\_review\_id;

DELETE FROM reviews WHERE review\_id=5502;

UPDATE reviews

SET description='Something'

WHERE review\_ID =5501;

----------------------------------------------------------------------------------

**FUNCTIONS:**

//Creating a function in Books Table. Total Book Count

CREATE OR REPLACE FUNCTION totalBooks

RETURN number IS

total number(5) := 0;

BEGIN

SELECT count(\*) into total

FROM books;

RETURN total;

END;

/

//Calling a function in Books Table. Total Book Count Output.

DECLARE

c number(5);

BEGIN

c := totalBooks();

dbms\_output.put\_line('Total no. of Books: ' || c);

END;

/

-----------------------------------------------------------------------

//CREATE A FUNCTION THAT TAKES ORDERNO AND RETURNS CUSTOMER NAME OF THAT ORDER.

CREATE OR REPLACE FUNCTION GETUSERNAME ( B\_ORDERID NUMBER) RETURN VARCHAR2

IS

V\_USERNAME VARCHAR2(50);

BEGIN

SELECT USER\_NAME INTO V\_USERNAME

FROM USERS

WHERE USER\_ID = ( SELECT USER\_ID FROM ORDERS WHERE ORDER\_ID = B\_ORDERID);

RETURN V\_USERNAME;

END;

/

//Calling A FUNCTION THAT TAKES ORDERNO AND RETURNS CUSTOMER NAME OF THAT ORDER.

DECLARE

c varchar2(50);

BEGIN

c := GETUSERNAME(30001);

dbms\_output.put\_line('Customer Name is: ' || c);

END;

/

-----------------------------------------------------------------------------------------

//CREATE A FUNCTION THAT RETURNS THE NEXT ORDER NUMBER?

CREATE OR REPLACE FUNCTION GETNEXTORDNO RETURN NUMBER

IS

V\_ORDER\_ID ORDERS.ORDER\_ID%TYPE;

BEGIN

SELECT MAX(ORDER\_ID) + 1 INTO V\_ORDER\_ID

FROM ORDERS;

RETURN V\_ORDER\_ID;

END;

/

//Calling a function in orders Table. next order no Output.

DECLARE

c NUMBER(5);

BEGIN

c := GETNEXTORDNO();

dbms\_output.put\_line('Next Order No is: ' || c);

END;

/

**RECORD:**

DECLARE

CURSOR users\_cur is

SELECT user\_id, user\_name, user\_email, user\_phone,password, user\_address, user\_district, user\_city,created\_date FROM users;

users\_record users\_cur%rowtype;

BEGIN

OPEN users\_cur;

LOOP

FETCH users\_cur into users\_record ;

EXIT WHEN users\_cur%notfound;

dbms\_output.put\_line(users\_record.user\_id || ' ' || users\_record.user\_name || ' ' ||users\_record.user\_email || ' ' || users\_record.user\_phone || ' ' || users\_record.password

|| ' ' || users\_record.user\_address || ' ' || users\_record.user\_district || ' ' || users\_record.user\_city || ' ' || users\_record.created\_date);

END LOOP;

END;

/

---------------------------------------------------------------------

DECLARE

CURSOR orders\_cur is

SELECT order\_id, status, price, total, due, pay\_type, order\_date, user\_id, cart\_id FROM orders;

orders\_record orders\_cur%rowtype;

BEGIN

OPEN orders\_cur;

LOOP

FETCH orders\_cur into orders\_record;

EXIT WHEN orders\_cur%notfound;

dbms\_output.put\_line(orders\_record.order\_id || ' ' || orders\_record.status || ' ' || orders\_record.price || ' ' || orders\_record.total || ' ' ||

orders\_record.due || ' ' || orders\_record.pay\_type || ' ' || orders\_record.order\_date || ' ' || orders\_record.user\_id || ' ' || orders\_record.cart\_id);

END LOOP;

END;

/

**TRIGGERS:**

1.DO NOT ALLOW ANY CHANGE TO ITEM RATE IN SUCH A WAY DIFFERENCE IS MORE THAN 25% OF THE EXISTING RATE.

CREATE OR REPLACE TRIGGER TRGDIFFRATE

BEFORE UPDATE

ON ITEMS

FOR EACH ROW

DECLARE

V\_DIFF NUMBER(5);

BEGIN

V\_DIFF := ABS(:NEW.RATE - :OLD.RATE);

IF V\_DIFF > :OLD.RATE \* 0.25 THEN

RAISE\_APPLICATION\_ERROR(-20014,'INVALID RATE FOR AMOUNT. CHANGE IS TOO BIG');

END IF;

END;

2.//DO NOT ALLOW ANY CHANGES TO ITEMS TABLE AFTER 9PM BEFORE 9AM

CREATE OR REPLACE TRIGGER CHECKTIME

BEFORE INSERT OR DELETE OR UPDATE

ON BOOKS

BEGIN

IF TO\_CHAR(SYSDATE,'HH24') < 9 OR TO\_CHAR(SYSDATE,'HH24') > 21 THEN

RAISE\_APPLICATION\_ERROR(-200011,'NO CHANGES CAN BE MADE BEFORE 9 A.M AND AFTER 9 P.M');

END IF;

END;

/

3.CREATE OR REPLACE TRIGGER t

BEFORE

INSERT OR

UPDATE OR

DELETE

ON books

BEGIN

CASE

WHEN INSERTING THEN

DBMS\_OUTPUT.PUT\_LINE('Inserting');

WHEN UPDATING THEN

DBMS\_OUTPUT.PUT\_LINE('Updating');

WHEN DELETING THEN

DBMS\_OUTPUT.PUT\_LINE('Deleting');

END CASE;

END;

/

**PACKAGES:**

create or replace package users\_pack as

procedure insert\_users(u\_id users.user\_id%type, u\_name users.user\_name%type,

u\_email users.user\_email%type, u\_phone users.user\_phone%type, u\_password users.password%type,

u\_address users.user\_address%type, u\_district users.user\_district%type,

u\_city users.user\_city%type, u\_created\_date users.created\_date%type);

procedure del\_users(u\_id users.user\_id%type);

procedure show\_users;

end users\_pack;

package body:

create or replace package body users\_pack as

procedure insert\_users(u\_id users.user\_id%type, u\_name users.user\_name%type,

u\_email users.user\_email%type, u\_phone users.user\_phone%type, u\_password users.password%type,

u\_address users.user\_address%type, u\_district users.user\_district%type, u\_city users.user\_city%type,

u\_created\_date users.created\_date%type) is

begin

insert into users values(seq\_user\_id.nextval, 'mehedi', 'hmehedi995@gmail.com', '01764760509','1234','uttora,dhaka','dhaka','dhaka','21-apr-2018');

end insert\_users;

-----------------------------------------------------------------------------------------------------------------------------

delete:

procedure del\_users(u\_id users.user\_id%type) is

begin

delete from users where user\_id = u\_id;

end del\_users;

--------------------------------------------------------------------------------------------

procedure show\_users is

cursor users\_cur is select \* from users;

begin

dbms\_output.put\_line('user\_id user\_name user\_email user\_phone password user\_address user\_district user\_city created\_date');

for users\_record in users\_cur loop

dbms\_output.put\_line(users\_record.user\_id || ' ' || users\_record.user\_name || ' ' ||users\_record.user\_email || ' ' || users\_record.user\_phone || ' ' || users\_record.password

|| ' ' || users\_record.user\_address || ' ' || users\_record.user\_district || ' ' || users\_record.user\_city || ' ' || users\_record.created\_date);

end loop;

end show\_users;

end users\_pack;

--------------------------------------------------------------------------------------

Annonomus block:

begin

users\_pack.show\_users;

end;

/

-------------------------------------------------------------------------------------------

delete:

begin

users\_pack.show\_users;

users\_pack.del\_users('CA');

users\_pack.show\_users;

end;

**VIEW:**

1.create view SHOW\_BOOK\_INFO(BOOK\_NAME,EDITION,ISBN\_NO,PRICE,PAGES,CATEGORY\_NAME,SUB\_CATEGORY\_NAME,AUTHOR\_NAME)

as SELECT b.book\_name,b.edition,b.isbn,b.price,b.page,b.cat\_name,b.subcat\_nam11e,a.author\_name from books b,authors a where b.author\_id=a.author\_id;

2.create view ORDER\_CHART(ORDER\_ID,ORDER\_STATUS,PRICE,TOTAL,DUE,PAYMENT\_TYPE,ORDER\_DATE,QUANTITY,USER\_ID)

as SELECT o.order\_id,o.status,o.price,o.total,o.due,o.order\_date,od.quantity,od.user\_id from orders o,order\_details od where o.order\_id= od.order\_id ;

Package By mam:

Package specification:

create or replace package users\_pack as procedure insert\_users

( u\_name users.user\_name%type, u\_email users.user\_email%type, u\_phone users.user\_phone%type, u\_password users.password%type, u\_address users.user\_address%type, u\_district users.user\_district%type, u\_city users.user\_city%type, u\_created\_date users.created\_date%type);

procedure del\_users(u\_id users.user\_id%type);

end users\_pack;

Package Body:

reate or replace package body users\_pack as procedure insert\_users( u\_name users.user\_name%type, u\_email users.user\_email%type, u\_phone users.user\_phone%type, u\_password users.password%type, u\_address users.user\_address%type, u\_district users.user\_district%type, u\_city users.user\_city%type, u\_created\_date users.created\_date%type)

is begin insert into users values( seq\_user\_id.nextval,u\_name, u\_email, u\_phone,u\_password,u\_address,u\_district,u\_city,u\_created\_date);

end insert\_users;

procedure del\_users(u\_id users.user\_id%type) is begin delete from users where user\_id = u\_id;

end del\_users;

end users\_pack;