CREATE TABLE department(

department\_id INTEGER PRIMARY KEY,

name\_department VARCHAR(50) NOT NULL

);

CREATE TABLE specialty(

specialty\_id INTEGER PRIMARY KEY,

name\_specialty VARCHAR(50) NOT NULL,

department\_id INTEGER REFERENCES department (department\_id)

);

CREATE TABLE groupp (

groupp\_id INTEGER PRIMARY KEY,

name\_group VARCHAR(50) NOT NULL,

specialty\_id INTEGER REFERENCES specialty(specialty\_id)

);

drop table student;

CREATE TABLE student (

student\_id INTEGER PRIMARY KEY,

name\_student VARCHAR(50) NOT NULL,

b\_date DATE NOT NULL,

email VARCHAR(50) NOT NULL,

phonenumber INTEGER NOT NULL,

groupp\_id INTEGER REFERENCES groupp(groupp\_id)

);

CREATE TABLE teacher(

teacher\_id INTEGER PRIMARY KEY,

name\_teacher VARCHAR(50) NOT NULL,

email VARCHAR(50) NOT NULL,

department\_id INTEGER REFERENCES department(department\_id)

);

CREATE TABLE subject(

subject\_id INTEGER PRIMARY KEY,

name\_subject VARCHAR(50) NOT NULL,

credits INTEGER NOT NULL

);

CREATE TABLE author(

author\_id INTEGER PRIMARY KEY,

name\_author VARCHAR(50) NOT NULL

);

CREATE TABLE book(

book\_id INTEGER PRIMARY KEY,

title VARCHAR(50) NOT NULL,

price NUMBER(10),

author\_id INTEGER REFERENCES author(author\_id),

subject\_id INTEGER REFERENCES subject(subject\_id)

);

Create table timee(

time\_id INTEGER PRIMARY KEY,

student\_id INTEGER REFERENCES student(student\_id),

name\_student VARCHAR(50) NOT NULL,

book\_id INTEGER REFERENCES book(book\_id),

tt DATE NOT NULL

);

INSERT INTO department VALUES (1, 'КИТ');

INSERT INTO department VALUES (2, 'МКМ');

INSERT INTO department VALUES (3, 'ИС');

INSERT INTO department VALUES (4, 'SIS');

INSERT INTO specialty VALUES (1, 'CSSE',1);

INSERT INTO specialty VALUES (2, 'CS',1);

INSERT INTO specialty VALUES (3, 'IS',3);

INSERT INTO groupp VALUES (1, 'CSSE1608K', 1);

INSERT INTO groupp VALUES (2, 'CSSE1603R', 3);

INSERT INTO groupp VALUES (3, 'CSSE1602K', 2);

INSERT INTO student VALUES (1, 'Arai',DATE'1998-07-17' ,'d.arai1798@gmail.com',823,1);

INSERT INTO student VALUES (2, 'Bota',DATE'1998-11-22' ,'bota@gmail.com',426,2);

INSERT INTO student VALUES (3, 'Toha',DATE'1998-07-07' ,'toha@gmail.com',745,1);

INSERT INTO teacher VALUES (1, 'Mariya','maria@gmail.com', 1);

INSERT INTO teacher VALUES (2, 'Ludmila','kozina@gmail.com', 1);

INSERT INTO teacher VALUES (3, 'Askar','askar@gmail.com', 3);

INSERT INTO subject VALUES (1, 'Machine Learning',3);

INSERT INTO subject VALUES (2, 'Data Base',3);

INSERT INTO subject VALUES (3, 'Python',3);

INSERT INTO author VALUES (1, 'Lishev');

INSERT INTO author VALUES (2, 'Ostin');

INSERT INTO author VALUES (3, 'Mark');

INSERT INTO book VALUES (1, 'Deep learning',4200,1,1 );

INSERT INTO book VALUES (2, 'Pandas',3400,2,3 );

INSERT INTO book VALUES (3, 'Database',2200,3,2 );

INSERT INTO book VALUES (4, 'Database',5200,3,1 );

INSERT INTO timee VALUES (1, 1,'Arai',2,DATE'2019-02-19');

INSERT INTO timee VALUES (2, 3,'Toha',3,DATE'2019-03-12');

create or replace package lib\_pkg is

function get\_info(

t\_id teacher.teacher\_id%TYPE

) RETURN VARCHAR2;

procedure upd\_book(

b\_id in book.book\_id%type,

b\_price in book.price%type

);

end lib\_pkg;

create or replace package body lib\_pkg is

FUNCTION get\_info(

t\_id teacher.teacher\_id%TYPE

) RETURN VARCHAR2 IS

t\_info VARCHAR2(50);

BEGIN

SELECT name\_teacher || ' ' || email INTO t\_info

FROM teacher

WHERE teacher\_id = t\_id;

RETURN t\_info;

END get\_info;

procedure upd\_book(

b\_id in book.book\_id%type,

b\_price in book.price%type

) is

cursor a\_cursor is

select title from book

where book\_id=b\_id;

t\_book book.title%type;

begin

open a\_cursor;

loop

fetch a\_cursor into t\_book;

exit when a\_cursor%NOTFOUND;

DBMS\_OUTPUT.put\_line(t\_book);

DBMS\_OUTPUT.put\_line(b\_price);

end loop;

close a\_cursor;

update book

set price=b\_price

where book\_id=b\_id;

end upd\_book;

end lib\_pkg;

EXECUTE DBMS\_OUTPUT.PUT\_LINE(lib\_pkg.get\_info(3));

EXECUTE lib\_pkg.upd\_book(2,2400);

select \* from timee;

create or replace trigger ch\_price\_trg

before delete or insert or update on book

for each row

when(new.book\_id>0)

declare

differ number;

begin

differ := :new.price - :old.price;

dbms\_output.put\_line('Old price: ' || :OLD.price);

dbms\_output.put\_line('New price: ' || :NEW.price);

dbms\_output.put\_line('price difference: ' || differ);

end;

insert into book values (4,'Oracle PL/SQL',1200,3,2);

update book

set price = price +800

where book\_id=3;

select \* from student;

delete from student where student\_id=4;

create or replace trigger t\_trg

before insert or update on timee

for each row

begin

if (to\_char(sysdate,'DY') in ('SAT','SUN')) then

raise\_application\_error(-20202,'On the weekend library is closed');

end if;

end ;

drop trigger t\_trg;

select \* from timee;

insert into timee values(5,3,'Toha',2,DATE'2019-05-06');

create or replace trigger price\_trg

before insert or update on book

for each row

begin

if not (:new.title in ('Database')) and :new.price>2000 then

raise\_application\_error(-20202,'Price of book Database is less than 2000');

end if;

end;

update book

set price = 2300

where title='Pandas';