**create function totalPrice()**

**returns integer as $total$**

**declare**

**total integer;**

**begin**

**select Sum(price) into total from item;**

**return total;**

**end**

**$total$ language plpgsql;**

**select totalPrice()**

CREATE TABLE Producers(

Producer\_Id serial PRIMARY KEY,

Producer\_Name varchar(100) not null UNIQUE

)

CREATE TABLE Items(

Item\_Id serial PRIMARY KEY,

Item\_Name varchar(100) not null,

Price integer CHECK(Price > 0),

Stock integer DEFAULT 0,

Producer\_Id integer REFERENCES Producers,

Status varchar(50)

)

CREATE TABLE Employees(

Employee\_Id serial PRIMARY KEY,

Employee\_Name varchar(100) not null,

Salary real not null,

Address text

)

CREATE TABLE Custommers(

Custommer\_Id serial PRIMARY KEY,

Custommer\_Name varchar(100) not null,

Address text

)

CREATE TABLE Oders(

Oder\_Id serial PRIMARY KEY,

Oder\_Number varchar(100) not null,

Oder\_Date timestamp DEFAULT CURRENT\_TIMESTAMP,

Employee\_Id integer REFERENCES Employees,

Custommer\_Id integer REFERENCES Custommers

)

CREATE TABLE OderItems(

Oder\_Id integer REFERENCES Oders,

Item\_Id integer REFERENCES Items,

Quantity smallint,

PRIMARY KEY(Oder\_Id,Item\_Id)

)

INSERT INTO Producers(Producer\_Name) VALUES('SamSung');

INSERT INTO Producers(Producer\_Name) VALUES('HTC');

INSERT INTO Producers(Producer\_Name) VALUES('Apple');

INSERT INTO Producers(Producer\_Name) VALUES('Sony');

INSERT INTO Producers(Producer\_Name) VALUES('LG');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('S8',900,100,1,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('S7',700,50,1,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('Note7',800,80,1,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('A8',500,0,1,'Unavailable');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('M8',400,100,2,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('U10',700,50,2,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('A9',450,0,2,'Unavailable');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('Iphone7',800,80,3,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('Iphone7+',1000,120,3,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('Z5',800,80,4,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('Z6',1000,0,4,'Unavailable');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('G5',500,80,5,'Available');

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status) VALUES('G6',800,10,5,'Available');

INSERT INTO Employees(Employee\_Name,Salary,Address) VALUES('Thien Tam',800,'Ho Chi Minh');

INSERT INTO Employees(Employee\_Name,Salary,Address) VALUES('Thanh Nhan',700,'Ho Chi Minh');

INSERT INTO Employees(Employee\_Name,Salary,Address) VALUES('Tram An',700,'Bien Hoa');

INSERT INTO Custommers(Custommer\_Name,Address) VALUES('Thanh Tam','Ho Chi Minh');

INSERT INTO Custommers(Custommer\_Name,Address) VALUES('Long Nhan','Ho Chi Minh');

INSERT INTO Custommers(Custommer\_Name,Address) VALUES('Van An','Bien Hoa');

INSERT INTO Custommers(Custommer\_Name,Address) VALUES('Vo Canh','Vung Tau');

INSERT INTO Custommers(Custommer\_Name,Address) VALUES('Tran Van','Ha Noi');

INSERT INTO Oders(Oder\_Number,Employee\_Id,Custommer\_Id) VALUES('NO:001',1,1);

CREATE OR REPLACE FUNCTION AddNewItem(name varchar(100), price integer, stock integer, pro\_id integer)

RETURNS VOID AS

$BODY$

BEGIN

IF stock > 0 THEN

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status)

VALUES(name,price,stock,pro\_id,'Available');

ELSE

INSERT INTO Items(Item\_Name,Price,Stock,Producer\_Id,Status)

VALUES(name,price,stock,pro\_id,'Unavailable');

END IF;

END

$BODY$ Language plpgsql;

select \* from AddNewItem('J7',300,12,1);

select \* from AddNewItem('J5',300,0,1);

select \* from items

CREATE OR REPLACE FUNCTION EditItem(\_id integer, \_price integer, \_stock integer)

RETURNS VOID AS

$BODY$

BEGIN

UPDATE Items

SET Price = \_price, Stock = \_stock

WHERE Item\_Id = \_id;

END

$BODY$ Language plpgsql;

select EditItem(11,910,0);

select \* from items

CREATE OR REPLACE FUNCTION DeleteItem(\_id integer)

RETURNS VOID AS

$BODY$

BEGIN

DELETE FROM Items

WHERE Item\_Id = \_id;

END

$BODY$ Language plpgsql;

select AddNewItem('dele',300,0,1);

select DeleteItem(18);

select \* from items

CREATE OR REPLACE FUNCTION AddNewCustommer(name varchar(100), address text)

RETURNS VOID AS

$BODY$

BEGIN

INSERT INTO Custommers(Custommer\_Name,Address)

VALUES(name,address);

END

$BODY$ Language plpgsql;

select AddNewCustommer('Thuan Nguyen','Thai Nguyen');

select \* from Custommers

CREATE OR REPLACE FUNCTION EditCustommer(\_id integer,\_name varchar(100), \_address text)

RETURNS VOID AS

$BODY$

BEGIN

UPDATE Custommers

SET Custommer\_Name = \_name, Address = \_address

WHERE Custommer\_Id = \_id;

END

$BODY$ Language plpgsql;

CREATE OR REPLACE FUNCTION DeleteCustommer(\_id integer)

RETURNS VOID AS

$BODY$

BEGIN

DELETE FROM Custommers

WHERE Custommer\_Id = \_id;

END

$BODY$ Language plpgsql;

select EditCustommer(7,'Thuan Nguyen1','Thai Nguyen1');

select DeleteCustommer(7);

select \* from Custommers

CREATE OR REPLACE FUNCTION NewOder(\_Oder\_Number varchar(100), \_Employee\_Id integer, \_Custommer\_Id integer)

RETURNS VOID AS

$BODY$

BEGIN

INSERT INTO Oders(Oder\_Number,Employee\_Id,Custommer\_Id) VALUES(\_Oder\_Number,\_Employee\_Id,\_Custommer\_Id);

END

$BODY$ Language plpgsql;

CREATE OR REPLACE FUNCTION DeleteOder(\_Oder\_Id integer)

RETURNS VOID AS

$BODY$

BEGIN

DELETE FROM Oders

WHERE Oder\_Id = \_Oder\_Id;

END

$BODY$ Language plpgsql;

select NewOder('NO:002',2,2);

select \* from Oders

CREATE OR REPLACE FUNCTION AddOderItem(\_Oder\_Id integer,\_Item\_Id integer ,\_Quantity integer)

RETURNS VOID AS

$BODY$

DECLARE \_Stock integer;

BEGIN

IF \_Quantity <1 THEN

RAISE EXCEPTION 'Quantity must bigger than 1';

END IF;

SELECT Stock INTO \_Stock

FROM Items

WHERE Item\_Id = \_Item\_Id;

IF \_Quantity <= \_Stock THEN

INSERT INTO OderItems(Oder\_Id,Item\_Id,Quantity) VALUES(\_Oder\_Id,\_Item\_Id,\_Quantity);

IF \_Quantity = \_Stock THEN

UPDATE Items

SET Stock = \_Stock - \_Quantity, Status = 'Unavailable'

WHERE Item\_Id = \_Item\_Id;

ELSE

UPDATE Items

SET Stock = \_Stock - \_Quantity, Status = 'Available'

WHERE Item\_Id = \_Item\_Id;

END IF;

ELSE

RAISE EXCEPTION 'Out of stock';

END IF;

END

$BODY$ Language plpgsql;

SELECT AddOderItem(4,16,2)

CREATE OR REPLACE FUNCTION DeleteOderItem(\_Oder\_Id integer,\_Item\_Id integer )

RETURNS VOID AS

$BODY$

DECLARE \_Stock integer;

BEGIN

SELECT Quantity INTO \_Stock

FROM OderItems

WHERE Item\_Id = \_Item\_Id AND Oder\_Id = \_Oder\_Id;

DELETE FROM OderItems

WHERE Item\_Id = \_Item\_Id AND Oder\_Id = \_Oder\_Id;

UPDATE Items

SET Stock = Stock + \_Stock, Status = 'Available'

WHERE Item\_Id = \_Item\_Id;

END

$BODY$ Language plpgsql;

SELECT DeleteOderItem(4,16)

CREATE OR REPLACE FUNCTION SalaryIncrease(\_Employee\_Id integer,\_percent integer )

RETURNS VOID AS

$BODY$

BEGIN

IF \_percent > 20 THEN

RAISE EXCEPTION 'Percent of increase must smaller than 20';

ELSE

UPDATE Employees

SET Salary = Salary + ((Salary\*\_percent)/100)

WHERE Employee\_Id = \_Employee\_Id;

END IF;

END

$BODY$ Language plpgsql;

SELECT SalaryIncrease(1,19)

CREATE OR REPLACE FUNCTION CheckItem()

RETURNS TRIGGER AS

$BODY$

BEGIN

IF new.Stock <> old.Stock THEN

IF new.Stock < 1 THEN

UPDATE Items

SET Status = 'Unavailable'

WHERE Item\_Id = new.Item\_Id;

ELSE

UPDATE Items

SET Status = 'Available'

WHERE Item\_Id = new.Item\_Id;

END IF;

END IF;

RETURN NEW;

END

$BODY$ Language plpgsql;

CREATE TRIGGER CheckUpdateInsertItem

AFTER UPDATE ON Items

FOR EACH ROW

EXECUTE PROCEDURE CheckItem();

select EditItem(11,910,0);