CREATE TABLE IF NOT EXISTS customers

(

idcustomers INT AUTO\_INCREMENT PRIMARY KEY NOT NULL UNIQUE,

Name VARCHAR(30) NOT NULL UNIQUE,

ImageURL VARCHAR(50) NOT NULL UNIQUE,

Email VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE IF NOT EXISTS products

(

idproduct INT AUTO\_INCREMENT PRIMARY KEY NOT NULL UNIQUE,

ProductName VARCHAR(30) NOT NULL UNIQUE,

ProductType VARCHAR(30) NOT NULL,

ProductPrice INT NOT NULL,

ProductCount INT NOT NULL,

ProductImage VARCHAR(50)

);

CREATE TABLE IF NOT EXISTS orders

(

idorder INT AUTO\_INCREMENT PRIMARY KEY NOT NULL UNIQUE,

CustomerId INT,

CustomerName VARCHAR(30) NOT NULL,

Email VARCHAR(30) NOT NULL,

Address VARCHAR(50) NOT NULL,

City VARCHAR(20) NOT NULL,

State VARCHAR(20) NOT NULL,

Zip INT NOT NULL,

NameOnCard VARCHAR(30) NOT NULL,

Creditcardnumber BIGINT(20) NOT NULL,

Exp\_Month INT NOT NULL,

Exp\_Year INT NOT NULL,

CVV INT NOT NULL,

ProductId INT NOT NULL,

Status VARCHAR(30) DEFAULT 'Undelivered',

CONSTRAINT numb CHECK(Creditcardnumber REGEXP '[5]{1}[0-9]{15}$'),

CONSTRAINT zip CHECK(LENGTH(Zip)=6),

CONSTRAINT mon CHECK(LENGTH(Exp\_Month)=2),

CONSTRAINT year CHECK(LENGTH(Exp\_Year)=4),

CONSTRAINT cvv CHECK(LENGTH(CVV)=3),

CONSTRAINT fokey FOREIGN KEY(CustomerId) REFERENCES customers(idcustomers)

);

CREATE TABLE IF NOT EXISTS views

(

idcustomers INT,

CONSTRAINT fk FOREIGN KEY(idcustomers) REFERENCES customers(idcustomers),

idproduct INT,

CONSTRAINT fok FOREIGN KEY(idproduct) REFERENCES products(idproduct)

);

CREATE TABLE IF NOT EXISTS has

(

idorder INT,

CONSTRAINT foke FOREIGN KEY(idorder) REFERENCES orders(idorder),

idproduct INT,

CONSTRAINT forkey FOREIGN KEY(idproduct) REFERENCES products(idproduct)

);

INSERT INTO customers (Name,ImageURL,Email) VALUES ('".$\_GET["name"]."','".$\_GET["imageurl"]."','".$\_GET["email"]."');

INSERT INTO products (ProductName,ProductType,ProductPrice,ProductCount,ProductImage) VALUES ('".$\_POST["name"]."','".$\_POST["type"]."','".$\_POST["price"]."','".$\_POST["count"]."','".$target\_file."');

SELECT MIN(ProductPrice) AS MinPrice FROM products;

SELECT MAX(ProductPrice) AS MaxPrice FROM products;

SELECT \* FROM products WHERE ProductType LIKE '%".$\_GET['search']."%' AND ProductPrice BETWEEN ".$price[0]." AND ".$price[1]." AND ProductCount>0;

INSERT INTO orders (CustomerName,Email,Address,City,State,Zip,NameOnCard,Creditcardnumber,Exp\_Month,Exp\_Year,CVV,ProductId)

VALUES ('".$\_SESSION["id"]."','".$\_SESSION["email"]."','".$\_POST["address"]."','".$\_POST["city"]."','".$\_POST["state"]."','".$\_POST["zip"]."',

'".$\_POST["cardname"]."','".$\_POST["cardnumber"]."','".$\_POST["expmonth"]."','".$\_POST["expyear"]."','".$\_POST["cvv"]."','".$\_SESSION["productid"]."');

UPDATE orders SET Status='Delivered' WHERE idorder='".$\_GET[id]."';

DELETE FROM orders WHERE idorder=".$\_GET["id"].""

CREATE OR REPLACE TRIGGER decproductcount AFTER INSERT ON orders

FOR EACH ROW

BEGIN

UPDATE products

SET ProductCount = ProductCount - 1

WHERE idproduct = new.ProductId;

END;

CREATE OR REPLACE TRIGGER incproductcount AFTER DELETE ON orders

FOR EACH ROW

BEGIN

UPDATE products

SET ProductCount = ProductCount + 1

WHERE idproduct = old.ProductId;

END;

CREATE OR REPLACE FUNCTION `GETCUSTNAME`

(

CUSTNAME VARCHAR(30)

)

RETURNS VARCHAR(30)

BEGIN

SELECT Name INTO CUSTNAME

FROM Customers

WHERE Name=(SELECT DISTINCT Name FROM customers);

RETURN CUSTNAME;

END;

CREATE OR REPLACE TRIGGER `CHECKITEMCOUNT`

BEFORE INSERT

ON orders

FOR EACH ROW

BEGIN

DECLARE

CNT NUMERIC(5);

SELECT COUNT(\*) INTO CNT

FROM orders WHERE idorder = NEW.idorder;

IF CNT >= 5 THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Number of items per order is greater than 5';

END IF;

END;

CREATE TABLE IF NOT EXISTS customerLogs

(

idcustomers INT,

name VARCHAR(50),

email VARCHAR(50),

date\_log DATE

);

CREATE OR REPLACE TRIGGER afterinsertlog AFTER INSERT ON customers FOR EACH ROW INSERT INTO customerLogs VALUES (new.idcustomers, NEW.name, NEW.email,CURDATE());

CREATE OR REPLACE TRIGGER afterupdatelog AFTER UPDATE ON customers FOR EACH ROW INSERT INTO customerLogs VALUES (NEW.idcustomers, NEW.name, NEW.email,CURDATE());

CREATE OR REPLACE TRIGGER afterdeleteLog BEFORE DELETE ON customers FOR EACH ROW INSERT INTO customerLogs VALUES (OLD.idcustomers, OLD.name, OLD.email,CURDATE());

SELECT orders.idorder,products.ProductName,products.ProductType,products.ProductPrice,orders.Status

FROM orders

INNER JOIN products ON orders.ProductID=products.idproduct and orders.CustomerName='".$\_SESSION["id"]."';

SELECT orders.idorder,products.ProductName,products.ProductType,products.ProductPrice,orders.CustomerName,orders.Status

FROM orders

INNER JOIN products ON orders.ProductID=products.idproduct;

CREATE OR REPLACE TRIGGER `CHECK\_PRICE\_INCREASE`

BEFORE UPDATE

ON products

FOR EACH ROW

BEGIN

IF OLD.ProductPrice < NEW.ProductPrice THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'There is increase in price';

END IF;

END