Proiect S.G.B.D

Baza de date folosita este una luata de pe site-ul MYSQL, pe care am modificat-o ulterior, dupa necesitati. B.D este construita pe modelul unui comerciant de masini clasice.

Informatii privind tabelele din baza de date :

* Customers: stochează datele clienților.
* Products: stochează o listă de mașini.
* ProductLines: stochează o listă de categorii de linii de produse.
* Orders : stochează comenzile de vânzare plasate de clienți.
* OrderDetails: stochează elementele de linie ale comenzilor de vânzare pentru fiecare comandă de vânzare.
* Payments: stochează plățile efectuate de clienți pe baza conturilor acestora.
* Employees: stochează toate informațiile referitoare la angajați, precum și structura organizațională, cum ar fi cine raportează cui.
* Offices: stochează datele birourilor de vânzări.

Diagrama bazei de date:

g

Crearea bazei de date:

Create database ‘proiect’;

CREATE TABLE `customers` (

`customerNumber` int(11) NOT NULL,

`customerName` varchar(50) NOT NULL,

`contactLastName` varchar(50) NOT NULL,

`contactFirstName` varchar(50) NOT NULL,

`phone` varchar(50) NOT NULL,

`addressLine1` varchar(50) NOT NULL,

`addressLine2` varchar(50) DEFAULT NULL,

`city` varchar(50) NOT NULL,

`postalCode` varchar(15) DEFAULT NULL,

`country` varchar(50) NOT NULL,

`salesRepEmployeeNumber` int(11) DEFAULT NULL,

PRIMARY KEY (`customerNumber`),

KEY `salesRepEmployeeNumber` (`salesRepEmployeeNumber`),

FOREIGN KEY (`salesRepEmployeeNumber`) REFERENCES `employees` (`employeeNumber`)

);

insert into `customers`(`customerNumber`,`customerName`,`contactLastName`,`contactFirstName`,`phone`,`addressLine1`,`addressLine2`,`city`,`postalCode`,`country`,`salesRepEmployeeNumber`) values

(103,'Atelier graphique','Schmitt','Carine ','40.32.2555','54, rue Royale',NULL,'Nantes','44000','France',1370),

(112,'Signal Gift Stores','King','Jean','7025551838','8489 Strong St.',NULL,'Las Vegas','83030','USA',1166),

(114,'Australian Collectors, Co.','Ferguson','Peter','03 9520 4555','636 St Kilda Road','Level 3','Melbourne','3004','Australia',1611),

(119,'La Rochelle Gifts','Labrune','Janine ','40.67.8555','67, rue des Cinquante Otages',NULL,'Nantes','44000','France',1370),

(121,'Baane Mini Imports','Bergulfsen','Jonas ','07-98 9555','Erling Skakkes gate 78',NULL,'Stavern','4110','Norway',1504),

(124,'Mini Gifts Distributors Ltd.','Nelson','Susan','4155551450','5677 Strong St.',NULL,'San Rafael','97562','USA',1165),

(125,'Havel & Zbyszek Co','Piestrzeniewicz','Zbyszek ','(26) 642-7555','ul. Filtrowa 68',NULL,'Warszawa','01-012','Poland',NULL),

(128,'Blauer See Auto, Co.','Keitel','Roland','+49 69 66 90 2555','Lyonerstr. 34',NULL,'Frankfurt','60528','Germany',1504),

(129,'Mini Wheels Co.','Murphy','Julie','6505555787','5557 North Pendale Street',NULL,'San Francisco','94217','USA',1165),

(131,'Land of Toys Inc.','Lee','Kwai','2125557818','897 Long Airport Avenue',NULL,'NYC','10022','USA',1323),

(141,'Euro+ Shopping Channel','Freyre','Diego ','(91) 555 94 44','C/ Moralzarzal, 86',NULL,'Madrid','28034','Spain',1370),

(144,'Volvo Model Replicas, Co','Berglund','Christina ','0921-12 3555','Berguvsvägen 8',NULL,'Luleå','S-958 22','Sweden',1504),

(145,'Danish Wholesale Imports','Petersen','Jytte ','31 12 3555','Vinbæltet 34',NULL,'Kobenhavn','1734','Denmark',1401),

(146,'Saveley & Henriot, Co.','Saveley','Mary ','78.32.5555','2, rue du Commerce',NULL,'Lyon','69004','France',1337),

(148,'Dragon Souveniers, Ltd.','Natividad','Eric','+65 221 7555','Bronz Sok.','Bronz Apt. 3/6 Tesvikiye','Singapore','079903','Singapore',1621),

(151,'Muscle Machine Inc','Young','Jeff','2125557413','4092 Furth Circle','Suite 400','NYC','10022','USA',1286),

(157,'Diecast Classics Inc.','Leong','Kelvin','2155551555','7586 Pompton St.',NULL,'Allentown','70267','USA',1216),

(161,'Technics Stores Inc.','Hashimoto','Juri','6505556809','9408 Furth Circle',NULL,'Burlingame','94217','USA',1165),

(166,'Handji Gifts& Co','Victorino','Wendy','+65 224 1555','106 Linden Road Sandown','2nd Floor','Singapore','069045','Singapore',1612),

(167,'Herkku Gifts','Oeztan','Veysel','+47 2267 3215','Brehmen St. 121','PR 334 Sentrum','Bergen','N 5804','Norway ',1504),

(168,'American Souvenirs Inc','Franco','Keith','2035557845','149 Spinnaker Dr.','Suite 101','New Haven','97823','USA',1286),

(169,'Porto Imports Co.','de Castro','Isabel ','(1) 356-5555','Estrada da saúde n. 58',NULL,'Lisboa','1756','Portugal',NULL),

(171,'Daedalus Designs Imports','Rancé','Martine ','20.16.1555','184, chaussée de Tournai',NULL,'Lille','59000','France',1370);

CREATE TABLE `employees` (

`employeeNumber` int(11) NOT NULL,

`lastName` varchar(50) NOT NULL,

`firstName` varchar(50) NOT NULL,

`extension` varchar(10) NOT NULL,

`email` varchar(100) NOT NULL,

`officeCode` varchar(10) NOT NULL,

`reportsTo` int(11) DEFAULT NULL,

`jobTitle` varchar(50) NOT NULL,

PRIMARY KEY (`employeeNumber`),

KEY `reportsTo` (`reportsTo`),

KEY `officeCode` (`officeCode`),

CONSTRAINT `employees\_ibfk\_1` FOREIGN KEY (`reportsTo`) REFERENCES `employees` (`employeeNumber`),

CONSTRAINT `employees\_ibfk\_2` FOREIGN KEY (`officeCode`) REFERENCES `offices` (`officeCode`)

) ;

insert into `employees`(`employeeNumber`,`lastName`,`firstName`,`extension`,`email`,`officeCode`,`reportsTo`,`jobTitle`) values

(1002,'Murphy','Diane','x5800','dmurphy@classicmodelcars.com','1',NULL,'President'),

(1056,'Patterson','Mary','x4611','mpatterso@classicmodelcars.com','1',1002,'VP Sales'),

(1076,'Firrelli','Jeff','x9273','jfirrelli@classicmodelcars.com','1',1002,'VP Marketing'),

(1088,'Patterson','William','x4871','wpatterson@classicmodelcars.com','6',1056,'Sales Manager (APAC)'),

(1102,'Bondur','Gerard','x5408','gbondur@classicmodelcars.com','4',1056,'Sale Manager (EMEA)'),

(1143,'Bow','Anthony','x5428','abow@classicmodelcars.com','1',1056,'Sales Manager (NA)'),

(1165,'Jennings','Leslie','x3291','ljennings@classicmodelcars.com','1',1143,'Sales Rep'),

(1166,'Thompson','Leslie','x4065','lthompson@classicmodelcars.com','1',1143,'Sales Rep'),

(1188,'Firrelli','Julie','x2173','jfirrelli@classicmodelcars.com','2',1143,'Sales Rep'),

(1216,'Patterson','Steve','x4334','spatterson@classicmodelcars.com','2',1143,'Sales Rep'),

(1286,'Tseng','Foon Yue','x2248','ftseng@classicmodelcars.com','3',1143,'Sales Rep'),

(1323,'Vanauf','George','x4102','gvanauf@classicmodelcars.com','3',1143,'Sales Rep'),

(1337,'Bondur','Loui','x6493','lbondur@classicmodelcars.com','4',1102,'Sales Rep'),

(1370,'Hernandez','Gerard','x2028','ghernande@classicmodelcars.com','4',1102,'Sales Rep'),

(1401,'Castillo','Pamela','x2759','pcastillo@classicmodelcars.com','4',1102,'Sales Rep');

CREATE TABLE `offices` (

`officeCode` varchar(10) NOT NULL,

`city` varchar(50) NOT NULL,

`phone` varchar(50) NOT NULL,

`addressLine1` varchar(50) NOT NULL,

`addressLine2` varchar(50) DEFAULT NULL,

`state` varchar(50) DEFAULT NULL,

`country` varchar(50) NOT NULL,

`postalCode` varchar(15) NOT NULL,

`territory` varchar(10) NOT NULL,

PRIMARY KEY (`officeCode`)

);

insert into `offices`(`officeCode`,`city`,`phone`,`addressLine1`,`addressLine2`,`state`,`country`,`postalCode`,`territory`) values

('1','San Francisco','+1 650 219 4782','100 Market Street','Suite 300','CA','USA','94080','NA'),

('2','Boston','+1 215 837 0825','1550 Court Place','Suite 102','MA','USA','02107','NA'),

('3','NYC','+1 212 555 3000','523 East 53rd Street','apt. 5A','NY','USA','10022','NA'),

('4','Paris','+33 14 723 4404','43 Rue Jouffroy D\'abbans',NULL,NULL,'France','75017','EMEA'),

('5','Tokyo','+81 33 224 5000','4-1 Kioicho',NULL,'Chiyoda-Ku','Japan','102-8578','Japan'),

('6','Sydney','+61 2 9264 2451','5-11 Wentworth Avenue','Floor #2',NULL,'Australia','NSW 2010','APAC'),

('7','London','+44 20 7877 2041','25 Old Broad Street','Level 7',NULL,'UK','EC2N 1HN','EMEA');

CREATE TABLE `orderdetails` (

`orderNumber` int(11) NOT NULL AUTO\_INCREMENT,

`productCode` varchar(15) NOT NULL,

`quantityOrdered` int(11) NOT NULL,

`priceEach` decimal(10,2) NOT NULL,

`orderLineNumber` smallint(6) NOT NULL,

PRIMARY KEY (`orderNumber`,`productCode`)

);

/\*Data for the table `orderdetails` \*/

insert into `orderdetails`(`productCode`,`quantityOrdered`,`priceEach`,`orderLineNumber`) values

('S18\_1749',30,'136.00',3),

('S18\_2248',50,'55.09',2),

('S18\_4409',22,'75.46',4),

('S24\_3969',49,'35.29',1),

('S18\_2325',25,'108.06',4),

('S18\_2795',26,'167.06',1),

('S24\_1937',45,'32.53',3);

CREATE TABLE `orders` (

`orderNumber` int(11) NOT NULL AUTO\_INCREMENT,

`orderDate` date NOT NULL,

`requiredDate` date NOT NULL,

`shippedDate` date DEFAULT NULL,

`status` varchar(15) NOT NULL,

`comments` text,

`customerNumber` int(11) NOT NULL,

PRIMARY KEY (`orderNumber`),

KEY `customerNumber` (`customerNumber`),

FOREIGN KEY (`customerNumber`)

REFERENCES `customers` (`customerNumber`)

);

insert into `orders`(`orderDate`,`requiredDate`,`shippedDate`,`status`,`comments`,`customerNumber`) values

('2003-01-06','2003-01-13','2003-01-10','Shipped',NULL,363),

('2003-01-09','2003-01-18','2003-01-11','Shipped','Check on availability.',128),

('2003-01-10','2003-01-18','2003-01-14','Shipped',NULL,181),

('2003-01-29','2003-02-07','2003-02-02','Shipped',NULL,121),

('2003-01-31','2003-02-09','2003-02-01','Shipped',NULL,141),

('2003-02-11','2003-02-21','2003-02-12','Shipped',NULL,145),

('2003-02-17','2003-02-24','2003-02-21','Shipped',NULL,278);

CREATE TABLE `payments` (

`customerNumber` int(11) NOT NULL,

`checkNumber` varchar(50) NOT NULL,

`paymentDate` date NOT NULL,

`amount` decimal(10,2) NOT NULL,

PRIMARY KEY (`customerNumber`,`checkNumber`),

FOREIGN KEY (`customerNumber`) REFERENCES `customers` (`customerNumber`)

);

insert into `payments`(`customerNumber`,`checkNumber`,`paymentDate`,`amount`) values

(103,'HQ336336','2004-10-19','6066.78'),

(103,'JM555205','2003-06-05','14571.44'),

(103,'OM314933','2004-12-18','1676.14'),

(112,'BO864823','2004-12-17','14191.12'),

(112,'HQ55022','2003-06-06','32641.98'),

(112,'ND748579','2004-08-20','33347.88'),

(114,'GG31455','2003-05-20','45864.03'),

(114,'MA765515','2004-12-15','82261.22'),

(114,'NP603840','2003-05-31','7565.08'),

(114,'NR27552','2004-03-10','44894.74'),

(119,'DB933704','2004-11-14','19501.82'),

(119,'LN373447','2004-08-08','47924.19'),

(119,'NG94694','2005-02-22','49523.67'),

(121,'DB889831','2003-02-16','50218.95'),

(121,'FD317790','2003-10-28','1491.38'),

(121,'KI831359','2004-11-04','17876.32'),

(121,'MA302151','2004-11-28','34638.14'),

(124,'AE215433','2005-03-05','101244.59'),

(124,'BG255406','2004-08-28','85410.87'),

(124,'CQ287967','2003-04-11','11044.30'),

(124,'ET64396','2005-04-16','83598.04'),

(124,'HI366474','2004-12-27','47142.70'),

(124,'HR86578','2004-11-02','55639.66'),

(124,'KI131716','2003-08-15','111654.40'),

(124,'LF217299','2004-03-26','43369.30'),

(124,'NT141748','2003-11-25','45084.38'),

(128,'DI925118','2003-01-28','10549.01'),

(128,'FA465482','2003-10-18','24101.81'),

(128,'FH668230','2004-03-24','33820.62'),

(128,'IP383901','2004-11-18','7466.32'),

(129,'DM826140','2004-12-08','26248.78'),

(129,'ID449593','2003-12-11','23923.93'),

(129,'PI42991','2003-04-09','16537.85'),

(131,'CL442705','2003-03-12','22292.62'),

(131,'MA724562','2004-12-02','50025.35'),

(131,'NB445135','2004-09-11','35321.97'),

(141,'AU364101','2003-07-19','36251.03'),

(141,'DB583216','2004-11-01','36140.38'),

(141,'DL460618','2005-05-19','46895.48'),

(141,'HJ32686','2004-01-30','59830.55'),

(141,'ID10962','2004-12-31','116208.40'),

(141,'IN446258','2005-03-25','65071.26'),

(141,'JE105477','2005-03-18','120166.58'),

(141,'JN355280','2003-10-26','49539.37'),

(141,'JN722010','2003-02-25','40206.20'),

(141,'KT52578','2003-12-09','63843.55'),

(141,'MC46946','2004-07-09','35420.74'),

(141,'MF629602','2004-08-16','20009.53'),

(141,'NU627706','2004-05-17','26155.91'),

(144,'IR846303','2004-12-12','36005.71'),

(144,'LA685678','2003-04-09','7674.94'),

(145,'CN328545','2004-07-03','4710.73'),

(145,'ED39322','2004-04-26','28211.70'),

(145,'HR182688','2004-12-01','20564.86'),

(145,'JJ246391','2003-02-20','53959.21'),

(146,'FP549817','2004-03-18','40978.53'),

(146,'FU793410','2004-01-16','49614.72'),

(146,'LJ160635','2003-12-10','39712.10'),

(148,'BI507030','2003-04-22','44380.15'),

(148,'DD635282','2004-08-11','2611.84'),

(148,'KM172879','2003-12-26','105743.00'),

(148,'ME497970','2005-03-27','3516.04'),

(151,'BF686658','2003-12-22','58793.53'),

(151,'GB852215','2004-07-26','20314.44'),

(151,'IP568906','2003-06-18','58841.35'),

(151,'KI884577','2004-12-14','39964.63'),

(157,'HI618861','2004-11-19','35152.12'),

(157,'NN711988','2004-09-07','63357.13'),

(161,'BR352384','2004-11-14','2434.25'),

(161,'BR478494','2003-11-18','50743.65'),

(161,'KG644125','2005-02-02','12692.19'),

(161,'NI908214','2003-08-05','38675.13'),

(166,'BQ327613','2004-09-16','38785.48'),

(166,'DC979307','2004-07-07','44160.92'),

(166,'LA318629','2004-02-28','22474.17'),

(167,'ED743615','2004-09-19','12538.01'),

(167,'GN228846','2003-12-03','85024.46'),

(171,'GB878038','2004-03-15','18997.89'),

(171,'IL104425','2003-11-22','42783.81');

CREATE TABLE `productlines` (

`productLine` varchar(50) NOT NULL,

`htmlDescription` mediumtext,

`image` mediumblob,

PRIMARY KEY (`productLine`)

);

insert into `productlines`(`productLine`,`htmlDescription`,`image`) values

('Classic Cars',NULL,NULL),

('Motorcycles',NULL,NULL),

('Planes',NULL,NULL),

('Ships',NULL,NULL),

('Trains',NULL,NULL),

('Trucks and Buses',NULL,NULL),

('Vintage Cars',NULL,NULL);

CREATE TABLE `products` (

`productCode` varchar(15) NOT NULL,

`productName` varchar(70) NOT NULL,

`productLine` varchar(50) NOT NULL,

`productScale` varchar(10) NOT NULL,

`productVendor` varchar(50) NOT NULL,

`quantityInStock` smallint(6) NOT NULL,

`buyPrice` decimal(10,2) NOT NULL,

PRIMARY KEY (`productCode`),

KEY `productLine` (`productLine`),

FOREIGN KEY (`productLine`) REFERENCES `productlines` (`productLine`)

);

insert into `products`(`productCode`,`productName`,`productLine`,`productScale`,`productVendor`,`quantityInStock`,`buyPrice`) values

('S700\_3167','F/A 18 Hornet 1/72','Planes','1:72','Motor City Art Classics',551,'54.40'),

('S700\_3505','The Titanic','Ships','1:700','Carousel DieCast Legends',1956,'51.09'),

('S700\_3962','The Queen Mary','Ships','1:700','Welly Diecast Productions',5088,'53.63'),

('S700\_4002','American Airlines: MD-11S','Planes','1:700','Second Gear Diecast',8820,'36.27'),

('S72\_1253','Boeing X-32A JSF','Planes','1:72','Motor City Art Classics',4857,'32.77'),

('S72\_3212','Pont Yacht','Ships','1:72','Unimax Art Galleries',414,'33.30');

**Functii:**

//functie care returneaza cat a platit, in total, clientul introdus

DELIMITER $$

CREATE FUNCTION SumaTotalClient(

nr\_client int(11)

)

RETURNS int(20)

DETERMINISTIC

BEGIN

DECLARE total decimal(10,2);

select sum(amount) into total from payments where customerNumber=nr\_client;

RETURN total;

END$$

DELIMITER ;

//functie care calculeaza daca s-a respectat perioada de 3 zile lucratoare pentru a livra totalitatea comenzilor

DELIMITER $$

CREATE FUNCTION timp\_livrare(nr\_pcomanda int(10), nr\_ucomanda int(10)

)

RETURNS varchar(50)

BEGIN

DECLARE ret varchar(50);

declare t1,t2 date;

declare i,p int(10);

set ret="Perioada livrare respectata";

set i=nr\_pcomanda;

WHILE i<=nr\_ucomanda DO

select shippedDate into t1 from orders where orderNumber=i;

select orderDate into t2 from orders where orderNumber=i;

SELECT DATEDIFF(t1,t2) into p;

if p>3 then

set ret="Perioada livrare nerespectata";

end if;

set i=i+1;

END WHILE;

RETURN ret;

END$$

DELIMITER ;

//functie care ne arata daca o comanda a fost livrata sau nu

DELIMITER $$

CREATE FUNCTION status\_comanda(nr\_comanda int(10)

)

RETURNS varchar(50)

BEGIN

DECLARE ret,comanda varchar(50);

set ret="Comanda a fost livrata";

select status into comanda from orders where orderNumber=nr\_comanda;

if comanda="Shipped" then

return ret;

else

set ret="Comanda in curs de livrare";

return ret;

end if;

END$$

DELIMITER ;

**Proceduri stocate:**

//procedura care calculeaza numarul de zile trecute pana la livrarea comenzii. Daca nr de zile depaseste numarul de 3 lucratoare se va face un discount de 1000 de lei, pentru depasirea termenului de livrare

DELIMITER //

CREATE PROCEDURE livrari(IN nr\_pcomanda int(10),IN nr\_ucomanda int(10))

BEGIN

declare t1,t2 date;

declare i,p int(10);

set i=nr\_pcomanda;

select orderNumber, sum(quantityOrdered\*priceEach) as 'cost\_total' from orderdetails group by orderNumber;

WHILE i<=nr\_ucomanda DO

select shippedDate into t1 from orders where orderNumber=i;

select orderDate into t2 from orders where orderNumber=i;

SELECT DATEDIFF(t1,t2) into p;

if p>3 then

select orderNumber, sum(quantityOrdered\*priceEach)-1000 as 'cost\_total',"Discount aplicat" as 'Reducere' from orderdetails where orderNumber=i;

else

select orderNumber, sum(quantityOrdered\*priceEach) as 'cost\_total',"Discount neaplicat" as 'Reducere' from orderdetails where orderNumber=i;

end if;

set i=i+1;

END WHILE;

END //

DELIMITER ;

//procedura (are in compozitie cursor) care afiseaza, pe un rand, lista cu emailuri

DELIMITER $$

CREATE PROCEDURE listaEmail (

INOUT emailList varchar(4000)

)

BEGIN

DECLARE break INTEGER DEFAULT 0;

DECLARE emailAddress varchar(100) DEFAULT "";

DEClARE curEmail CURSOR FOR SELECT email FROM employees;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET break = 1;

OPEN curEmail;

email: LOOP

FETCH curEmail INTO emailAddress;

IF break = 1 THEN

LEAVE email;

END IF;

SET emailList = CONCAT(emailAddress,";",emailList);

END LOOP email;

CLOSE curEmail;

END$$

DELIMITER ;

//procedura care afiseaza functiile si numarul de angajati de pe o anumita functie

DELIMITER //

CREATE PROCEDURE functii()

BEGIN

select jobTitle as 'Lista functii' ,count(\*) as 'Numar functii' from employees group by jobTitle;

END //

DELIMITER ;

**Declansatori:**

//trigger care afiseaza intr un tabel 'demiteri' angajatii care sunt demisi(delete trigger)

CREATE TRIGGER demitere\_emp

BEFORE DELETE ON employees

FOR EACH ROW

INSERT INTO demiteri

SET action = 'delete',

employeeNumber = OLD.employeeNumber,

prenume = OLD.lastName,

nume = OLD.firstName,

data\_demiterii = NOW();

//trigger care afiseaza intr un tabel 'angajari' angajatii angajati dintr un anumit punct(insert trigger)

CREATE TRIGGER angajari\_emp

AFTER INSERT ON employees

FOR EACH ROW

INSERT INTO angajari

SET action = 'insert',

prenume = NEW.lastName,

nume = NEW.firstName,

data\_angajarii = NOW();

//trigger care creste pretul per produs, daca stocul este unul limitat(<10)

DELIMITER $$

CREATE TRIGGER stoc\_pret

BEFORE UPDATE

ON products FOR EACH ROW

BEGIN

if NEW.quantityInStock<10 then

set NEW.buyPrice=OLD.buyPrice+10;

end if;

END$$

DELIMITER ;

Cod sursa al interfetei grafice(Java GUI):

import javax.swing.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.sql.\*;  
  
public class test {  
 private JPanel Main;  
 private JTextField txtProd;  
 private JTextField txtQty;  
 private JTextField txtPrice;  
 private JTextField txtNr;  
 private JButton INSERTButton;  
 private JButton DELETEButton;  
 private JButton UPDATEButton;  
 private JTextField txtorderNumber;  
 private JButton SEARCHButton;  
  
 public static void main(String[] args) {  
 JFrame frame = new JFrame("test");  
 frame.setContentPane(new test().Main);  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.pack();  
 frame.setVisible(true);  
 }  
  
 Connection con;  
 PreparedStatement pst;  
  
 public test() {  
 Connect();  
  
 INSERTButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
  
  
 String prod,qty,price,nr;  
  
 prod = txtProd.getText();  
 qty = txtQty.getText();  
 price = txtPrice.getText();  
 nr = txtNr.getText();  
  
  
 try {  
 pst = con.prepareStatement("insert into orderdetails(productCode,quantityOrdered,priceEach,orderLineNumber)values(?,?,?,?)");  
 pst.setString(1, prod);  
 pst.setString(2, qty);  
 pst.setString(3, price);  
 pst.setString(4, nr);  
 pst.executeUpdate();  
 JOptionPane.*showMessageDialog*(null,"Record Added!");  
  
 txtProd.setText("");  
 txtQty.setText("");  
 txtPrice.setText("");  
 txtNr.setText("");  
 txtProd.requestFocus();  
 }  
  
 catch (SQLException e1)  
 {  
 e1.printStackTrace();  
 }  
  
 }  
 });  
 SEARCHButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
  
  
 try {  
 String orderNumber;  
 orderNumber = txtorderNumber.getText();  
 pst = con.prepareStatement("select productCode,quantityOrdered,priceEach,orderLineNumber from orderdetails where orderNumber = ?");  
 pst.setString(1, orderNumber);  
 ResultSet rs = pst.executeQuery();  
  
 if(rs.next()==true)  
 {  
 String prod = rs.getString(1);  
 String qty = rs.getString(2);  
 String price = rs.getString(3);  
 String nr = rs.getString(4);  
  
 txtProd.setText(prod);  
 txtQty.setText(qty);  
 txtPrice.setText(price);  
 txtNr.setText(nr);  
 }  
 else  
 {  
 txtProd.setText("");  
 txtQty.setText("");  
 txtPrice.setText("");  
 txtNr.setText("");  
 JOptionPane.*showMessageDialog*(null,"Numar de comanda invalid");  
  
 }  
 }  
  
 catch (SQLException ex)  
 {  
 ex.printStackTrace();  
 }  
 }  
 });  
 UPDATEButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
  
 String prod,qty,price,nr,orderNumber;  
  
 prod = txtProd.getText();  
 qty = txtQty.getText();  
 price = txtPrice.getText();  
 nr = txtNr.getText();  
 orderNumber = txtorderNumber.getText();  
  
 try {  
  
 pst = con.prepareStatement("update orderdetails set productCode = ?,quantityOrdered = ?,priceEach = ?,orderLineNumber = ? where orderNumber = ?");  
 pst.setString(1, prod);  
 pst.setString(2, qty);  
 pst.setString(3, price);  
 pst.setString(4, nr);  
 pst.setString(5, orderNumber);  
  
 pst.executeUpdate();  
 JOptionPane.*showMessageDialog*(null, "Record Updated!");  
  
 txtProd.setText("");  
 txtQty.setText("");  
 txtPrice.setText("");  
 txtNr.setText("");  
 txtProd.requestFocus();  
 txtorderNumber.setText("");  
 }  
  
 catch (SQLException e1)  
 {  
  
 e1.printStackTrace();  
 }  
 }  
 });  
 DELETEButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String orderNumber;  
  
 orderNumber = txtorderNumber.getText();  
  
  
 try {  
 pst = con.prepareStatement("delete from orderdetails where orderNumber = ?");  
 pst.setString(1, orderNumber);  
  
 pst.executeUpdate();  
 JOptionPane.*showMessageDialog*(null, "Record Deleted!");  
  
 txtProd.setText("");  
 txtQty.setText("");  
 txtPrice.setText("");  
 txtNr.setText("");  
 txtProd.requestFocus();  
 txtorderNumber.setText("");  
 }  
  
 catch (SQLException e1)  
 {  
  
 e1.printStackTrace();  
 }  
 }  
 });  
 }  
  
  
  
 public void Connect()  
 {  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 con = DriverManager.*getConnection*("jdbc:mysql://localhost/proiect", "root","");  
 System.*out*.println("Success");  
 }  
 catch (ClassNotFoundException ex)  
 {  
 ex.printStackTrace();  
 }  
 catch (SQLException ex)  
 {  
 ex.printStackTrace();  
 }  
 }  
  
}

