CARLOS JIMÉNEZ SÁNCHEZ – CRISTIAN KNELLEKEN BARROSO

CAL 3 - DATABASES

2014 - 2015

ÍNDICE

[**TRIGGERS** 2](#_Toc418552031)

[--Bill 2](#_Toc418552032)

[--Packs 3](#_Toc418552033)

[--Offer 4](#_Toc418552034)

[--Services 5](#_Toc418552035)

[--Generated code product 6](#_Toc418552036)

[--Generated code offer 6](#_Toc418552037)

[**CONNECTING THE DATABASE WITH JAVA** 7](#_Toc418552038)

[**VIEWS** 15](#_Toc418552039)

[**CREATING USERS** 19](#_Toc418552040)

[1- Administrators 19](#_Toc418552041)

[2- Managers 19](#_Toc418552042)

[3- Commercial 19](#_Toc418552043)

[**CONNECTION TO THE DATABASE FROM EXTERNAL PROGRAMS AND SECURITY** 22](#_Toc418552044)

[**Login:** 22](#_Toc418552045)

[**Pec3**: 27](#_Toc418552046)

# **TRIGGERS**

We create a series of triggers to provide functionality to the fundamentals and use it to efficiently manage the business logic of the company Mediamarkt.

/\* Triggers work correctly\*/

## --Bill

create or replace function my\_bill() returns trigger as $bill$

begin

if new.number\_bill is null then

raise exception 'Not insert number of bill';

end if;

if new.shopping\_number is null then

raise exception '% Not show number of shopping',new.shopping\_number;

end if;

return new;

end;

$bill$ language plpgsql;

create trigger my\_bill before insert on bill

for each row --para cada linea

execute procedure my\_bill();

/\*Mostramos la raise exception\*/

insert into bill (number\_bill,shopping\_number)

values (2000,null);

## --Packs

create or replace function my\_pack() returns trigger as $pack$

begin

if new.id is null then

raise exception 'Not insert ID';

end if;

if new.price is null then

raise exception '% Don´t insert price',new.price;

end if;

return new;

end;

$pack$ language plpgsql;

create trigger pack before insert or update on packs

for each row --para cada linea

execute procedure my\_pack();

/\*Con este codigo podemos insertar y actualizar packs, ademas usando null provocamos excepciones \*/

insert into packs (id,price)

values(100011,2000);

update packs set

id =100011,

price = 2000

where id = 100023

select \* from packs

--Offer Tiene problemas

create function my\_offer() returns trigger as $offer$

declare

x integer;

y integer;

begin

if new.id\_offer != old.id\_offer then

select count(\*) into x

from offer

where new.id\_offer = offer.id\_offer;

if (x = 0) then

raise exception 'Cant show';

end if;

end if;

return old;

end;

&offer&

language 'plpgsql';

Create trigger my\_offer after insert or update or delete on offer

for each row

execute procedure my\_offer();

## --Services

create or replace function my\_service() returns trigger as $service$

begin

if new.code is null then

raise exception 'Not insert code';

end if;

if new.name is null then

raise exception '% Don´t insert name',new.price;

end if;

if new.description is null then

raise exception ' Not insert description',new.price;

end if;

/\*if new.additional\_cost\_with\_VAT is null then

raise exception '% Don´t insert additional\_cost\_with\_VAT',new.price;

end if; \*/

return new;

end;

$service$ language plpgsql;

create trigger service2 before insert or update or delete on services

for each row --para cada linea

execute procedure my\_service();

/\*Con este codigo podemos insertar y actualizar services, ademas usando null provocamos excepciones \*/

insert into services (code,name,description)

values(null,'ServiceVIP2','New Service two');

delete from "services"

where new."code" = 15000;

update services set

code = null,

description = 'ServiceVIP2'

where code = 15000

select \* from services

## --Generated code product

CREATE function generated\_customer() RETURNS trigger AS $customer$

DECLARE

nifCustomer INTEGER;

BEGIN

SELECT count(\*) INTO nifCustomer

FROM product;

UPDATE customer SET nif = nifCustomer WHERE new.nif = customer.nif;

RAISE NOTICE 'Code customer generated successfully';

RETURN new;

END;

$customer$

LANGUAGE plpgsql;

CREATE TRIGGER generated\_customer AFTER INSERT ON customer

FOR EACH ROW

execute procedure generated\_customer();

## --Generated code offer

CREATE function generated\_offer() RETURNS trigger AS

DECLARE

idOffer INTEGER;

BEGIN

SELECT count(\*) INTO idOffer FROM offer;

UPDATE offer SET id = idOffer WHERE new.id = offer.id;

RAISE NOTICE 'Code offer generated successfully';

RETURN new;

END;

LANGUAGE plpgsql;

CREATE TRIGGER generated\_offer AFTER INSERT ON offer

FOR EACH ROW

execute procedure generated\_offer();

# **CONNECTING THE DATABASE WITH JAVA**

In this section we proceed to the creation of a JAVA application to connect to the database and allows us to perform queries described in PECL2. The source of this connection is as follows:

package CAL3;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

**public class CAL3 extends javax.swing.JFrame {**

public CAL3() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

jTextArea\_SQL = new javax.swing.JTextArea();

jButton\_Aceptar = new javax.swing.JButton();

jScrollPane2 = new javax.swing.JScrollPane();

jTable\_Datos = new javax.swing.JTable();

jLabel1 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jTextArea\_SQL.setColumns(20);

jTextArea\_SQL.setRows(5);

jScrollPane1.setViewportView(jTextArea\_SQL);

jButton\_Aceptar.setText("Aceptar");

jButton\_Aceptar.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton\_AceptarActionPerformed(evt);

}

});

jTable\_Datos.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{null, null, null, null},

{null, null, null, null},

{null, null, null, null},

{null, null, null, null}

},

new String [] {

"Title 1", "Title 2", "Title 3", "Title 4"

}

));

jScrollPane2.setViewportView(jTable\_Datos);

jLabel1.setText("Realice la consulta:");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton\_Aceptar)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jScrollPane1)

.addComponent(jLabel1)

.addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT\_SIZE, 711, Short.MAX\_VALUE)))

.addContainerGap(74, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

.addGap(27, 27, 27)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jButton\_Aceptar)

.addGap(13, 13, 13)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 261, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(168, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

**private void jButton\_AceptarActionPerformed(java.awt.event.ActionEvent evt) {**

// TODO add your handling code here:

try{

try {

Class.forName("org.postgresql.Driver");

} catch (ClassNotFoundException cnfe) {

JOptionPane.showMessageDialog(this, "driver no disponible", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

Connection c = null;

int numero=0;

ResultSet rs = null;

Statement s = null;

BufferedReader entrada=new BufferedReader(new InputStreamReader(System.in));

try{

c = DriverManager.getConnection("jdbc:postgresql://localhost:5432/PECL2", "postgres", "oqk&2435");

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "No se pudo realizar la conexión: " + se.toString(), "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

try {

s = c.createStatement();

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "problema al crear la consulta", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

try {

String variableSQL = this.jTextArea\_SQL.getText().trim() + ";";

rs =s.executeQuery(variableSQL) ;

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "Excepción al ejecutar consulta: error de sintaxis en el SQL", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

DefaultTableModel modelo = new DefaultTableModel();

this.jTable\_Datos.setModel(modelo);

ResultSetMetaData rsmd = rs.getMetaData();

int numberOfColumns = rsmd.getColumnCount();

Object[] etiquetas = new Object[numberOfColumns];

Object[] BD = new Object[numberOfColumns];

try {

for (int i = 0; i < numberOfColumns; i++)

{

// Nuevamente, para ResultSetMetaData la primera columna es la 1.

etiquetas[i] = rsmd.getColumnLabel(i + 1);

}

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "Error grave al recoger los resultados3", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

modelo.setColumnIdentifiers(etiquetas);

int aux = 0;

try {

while (rs.next()) {

for (int i = 0; i < numberOfColumns; i++)

{

// Nuevamente, para ResultSetMetaData la primera columna es la 1.

BD[i] = rs.getString(i + 1);

}

modelo.addRow(BD);

aux++;

}

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "Error grave al recoger los resultados", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

rs.close();

s.close();

c.close();

} catch (SQLException ex) {

JOptionPane.showMessageDialog(this, "Error grave1", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}catch(Exception e){

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

**java.awt.EventQueue.invokeLater(new Runnable() {**

public void run() {

new CAL3().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton\_Aceptar;

private javax.swing.JLabel jLabel1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTable jTable\_Datos;

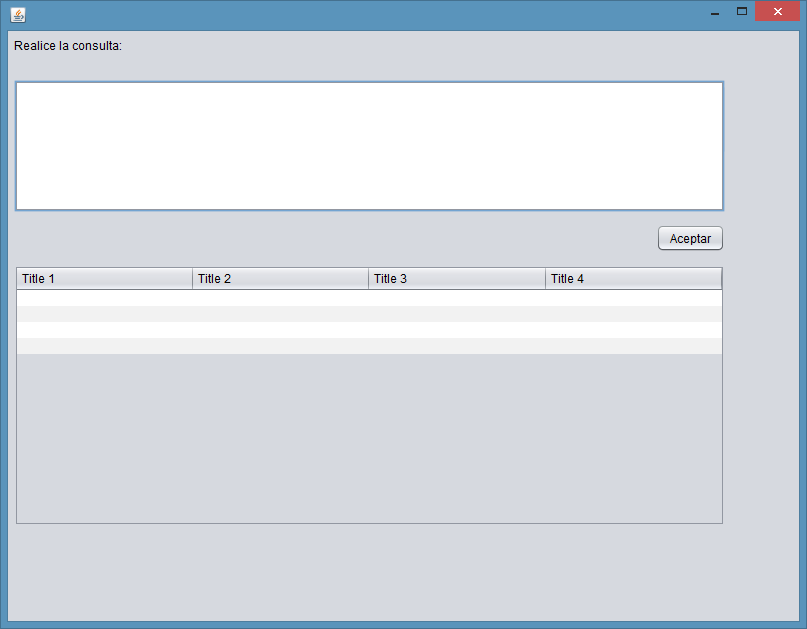
private javax.swing.JTextArea jTextArea\_SQL;

// End of variables declaration

}

In this we differ by color code: First yellow see the package name; On the other hand gray we appreciate the comments of the program itself and comments already created directly by the program; Red observe all the code that we have inserted for connecting the database; Finally in blue, are the interface buttons and deprived of them created by Neatbeans variables.

When it comes to represent the graphical interface of the program, as shown below, we perform a simple graphical interface and will be easier to interact with it more easily. The appearance is as follows:



The first blank is reserved to write the query to run, where we introduce each of the consultations PECL2, and pressing the OK button will appear in the space below.  
Moreover, we can see how the code is divided into four distinct segments.

1. loaded the driver to interact with the database.
2. We establish connection to the database (name of database, user name and password)
3. Consultation, if correct will be shown on screen

Then we will make some consultations called for in the PECL2 to verify that makes what is required for the program  
\*\*\* We will have to make changes to adapt to the new model queries \*\*\*

# **VIEWS**

In this part of the practice we conducted views, where we can see the tables in the database by simply clicking on each of the views

/\*CONSULTA VIEW1:\*/

create view consultaview1 as (select sum("amount\_with\_VAT")

from "shopping")

/\*CONSULTA VIEW2:\*/

SELECT COUNT(\*), "categorie"

FROM "product"

GROUP BY "product"."categorie";

/\*CONSULTA VIEW3:\*/

select \*

from public. "product"

where "categorie" = 'Phone';

/\*CONSULTA VIEW4:\*/

SELECT "name"

FROM "product"

WHERE "name" not in (

SELECT "name"

FROM "product","is\_offered"

WHERE "product"."code\_product" = "is\_offered"."id");

/\*CONSULTA VIEW5:\*/

SELECT "customer"."name\_customer", "shopping"."nif", SUM("amount\_with\_VAT")

FROM "shopping","customer"

WHERE "shopping"."nif"="customer"."nif"

GROUP BY "customer"."name\_customer","shopping"."nif";

/\*CONSULTA VIEW6:\*/

SELECT "Name", "product"."code\_product"

FROM "product", "is\_offered"

WHERE "product"."code\_product"="is\_offered"."code\_product";

/\*CONSULTA VIEW7:\*/

SELECT "name", AVG("score")

FROM "product", "reviews"

WHERE "product"."code\_product"="reviews"."code\_product"

GROUP BY "name";

/\*CONSULTA VIEW8:\*/

SELECT COUNT(\*) AS time, "product"."name"

FROM "product","shopping"

WHERE "product"."code\_product"="shopping"."code\_product"

GROUP BY "name"

ORDER BY time desc

limit 1;

/\*CONSULTA VIEW9:\*/

select SUM("shopping"."amount\_with\_VAT" - "shopping"."amount\_without\_VAT")

from "shopping","offer"

where "offer"."id\_offer"= "shopping"."id\_offer";

/\*CONSULTA VIEW10:\*/

SELECT COUNT(\*), "name"

FROM "product", "refund"

WHERE "product"."code\_product"="refund"."codeP"

GROUP BY "name";

/\*CONSULTA VIEW11:\*/

select sum("amount\_with\_VAT")

from "customer\_online","customer", "shopping"

where "customer"."nif"="customer\_online"."nif" and "customer"."nif"="customer"."nif";

select sum("amount\_with\_VAT")

from "customer\_physical","customer", "shopping"

where "customer"."nif"="customer\_physical"."nif" and "customer"."nif"="shopping"."nif";

/\*CONSULTA VIEW12:\*/

select "customer"."name\_customer","shopping"."amount\_with\_VAT"

from "customer","shopping","bill"

where "customer"."nif" = "shopping"."nif" and "bill"."shopping\_number" = "shopping"."shopping\_number";

/\*CONSULTA VIEW13:\*/

select avg("date\_return"), "categorie"

from (select ("refund"."date\_return"-"shopping"."date") from "refund","shopping" )as "date\_return","product";

/\*CONSULTA VIEW14:\*/

SELECT "services"."name", count(\*) as time

FROM "services","shopping"

where "services"."code"="shopping"."code"

GROUP BY "services"."name"

ORDER BY time desc;

/\*CONSULTA VIEW15:\*/

SELECT sum("services"."additional\_cost\_with\_VAT")

from "services","shopping"

where "services"."code"="shopping"."code";

/\*CONSULTA VIEW16:\*/

select count(\*)

from "refund"

where "imperfection" = true ;

/\*CONSULTA 17:\*/

/\*CONSULTA VIEW18:\*/

SELECT "code\_product", "product"."name"

FROM "product", "offer"

WHERE "product"."code\_product"<>"offer"."id\_offer";

--WHERE "Code\_product" NOT IN (SELECT "Code\_product" FROM "have");

/\*CONSULTA VIEW19:\*/

SELECT "date","amount\_with\_VAT", "offer"."name"

FROM "shopping", "offer"

WHERE "shopping"."date" BETWEEN "offer"."start\_date" AND "offer"."end\_date";

/\*CONSULTA VIEW20:\*/

select "categorie",count(\*) as time

from "product"

group by "categorie"

order by "time" desc limit 10;

# **CREATING USERS**

In a database it is important to consider the different profiles or roles of users who will access it, and clearly identify how can interact with the database, or what is the same, specify what will be permissions  
Management considered three types of profiles:

1. Administrators: have all kinds of permissions to the database
2. Managers: You will manage the database (insert, update, delete and query data), the downside of this is that users cannot create new tables and elements that affect the structure of the database.
3. Commercial: Only may consult the product information

The code for creating these users is:

/\* Rol de los Administradores \*/

CREATE ROLE "administrator" SUPERUSER INHERIT CREATEDB CREATEROLE;

GRANT ALL ON associated,bill,customer,customer\_online,customer\_physical,have,offer,packs,product,refund,reviews,services,shopping,is\_offered TO "administrator";

CREATE USER "admin" WITH PASSWORD 'admin' CREATEROLE CREATEUSER INHERIT;

GRANT "administrator" TO "admin";

/\* Rol de los Gestores\*/

CREATE ROLE "manager" NOSUPERUSER NOINHERIT NOCREATEDB NOCREATEROLE;

GRANT SELECT, INSERT, UPDATE, DELETE ON associated,bill,customer,customer\_online,customer\_physical,have,offer,packs,product,refund,reviews,services,shopping,is\_offered TO "manager";

CREATE USER "manger" WITH PASSWORD 'manger';

GRANT "manager" TO "manger";

/\* Rol de Comerciales\*/

CREATE ROLE "cashier" NOSUPERUSER NOINHERIT NOCREATEDB NOCREATEROLE;

GRANT SELECT ON associated,bill,customer,customer\_online,customer\_physical,have,offer,packs,product,refund,reviews,services,shopping,is\_offered TO "cashier";

CREATE USER "cash" WITH PASSWORD 'cash';

GRANT "cashier" TO "cash";

set role "administrator"

REVOKE CREATE ON SCHEMA public FROM "manager"

REVOKE CREATE ON TABLESPACE pg\_default FROM "manager"

REVOKE CREATE ON SCHEMA public FROM "cashier"

REVOKE CREATE ON TABLESPACE pg\_default FROM "cashier"

The operations that are requested for each user are shown:

SET ROLE "administrator"

INSERT INTO product VALUES (181, 'photography', 'Camara reflex' ,'Canon EOS 1200D',349, 'Canon',8);

UPDATE product SET Stock = 7 WHERE product.Stock = 8;

SELECT \* FROM product WHERE Stock = 5;

DELETE FROM product WHERE Stock = 7;

ALTER TABLE product ADD COLUMN Example text

ALTER TABLE product DROP COLUMN Example

CREATE TABLE Example (example1 text)

DROP TABLE Example

select \* from product;

SET ROLE "manager"

INSERT INTO product VALUES (150, 'kitchen', 'Coffee capsules' ,'coffee nespressoX470',519, 'Nespresso',13);

UPDATE product SET Stock = 11 WHERE product.Stock = 13;

SELECT \* FROM product WHERE Stock = 1;

DELETE FROM product WHERE Stock = 11;

ALTER TABLE product ADD COLUMN Example text

ALTER TABLE product DROP COLUMN Example

CREATE TABLE Example (example1 text)

DROP TABLE Example

select \* from product;

SET ROLE "cashier"

INSERT INTO product VALUES (281, 'photography', 'Camara reflex' ,'Canon EOS 1200D',249, 'Canon',58);

UPDATE product SET Stock = 10 WHERE product.Stock = 58;

DELETE FROM product WHERE Stock = 10;

/\*Aqui vemos como solo puede consultar\*/

Select \* from product;

# **CONNECTION TO THE DATABASE FROM EXTERNAL PROGRAMS AND SECURITY**

In this last part we modify the JAVA program that we previously created to select one of the three users created in the database. Modify it depending on the privileges held by each user.

We have created two classes to login and then show you the code:

## **Login:**

package CAL3;

import javax.swing.JOptionPane;

public class Login extends javax.swing.JFrame {

public Login() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jPasswordField = new javax.swing.JPasswordField();

jButtonInicioSesion = new javax.swing.JButton();

jLabel2 = new javax.swing.JLabel();

jLabel1 = new javax.swing.JLabel();

jComboBox\_Usuario = new javax.swing.JComboBox();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jButtonInicioSesion.setText("Iniciar Sesion");

jButtonInicioSesion.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButtonInicioSesionActionPerformed(evt);

}

});

jLabel2.setText("Password");

jLabel1.setText("Escoja el usuario");

jComboBox\_Usuario.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "admin", "manger", "cash" }));

jComboBox\_Usuario.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jComboBox\_UsuarioActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(57, 57, 57)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jLabel2)

.addComponent(jLabel1))

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(jPasswordField, javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jButtonInicioSesion, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jComboBox\_Usuario, javax.swing.GroupLayout.Alignment.LEADING, 0, 154, Short.MAX\_VALUE))

.addContainerGap(65, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(25, 25, 25)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(jComboBox\_Usuario, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(24, 24, 24)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(jPasswordField, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(28, 28, 28)

.addComponent(jButtonInicioSesion)

.addContainerGap(139, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButtonInicioSesionActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

if (this.jComboBox\_Usuario.getSelectedItem().toString().compareTo("admin") == 0) {

if (this.jPasswordField.getText().compareTo("admin") == 0) {

new CAL3("admin").setVisible(true);

} else {

JOptionPane.showMessageDialog(this, "Password incorrecta", "Error de inicio de sesión", JOptionPane.ERROR\_MESSAGE);

}

}

if (this.jComboBox\_Usuario.getSelectedItem().toString().compareTo("manager") == 0) {

if (this.jPasswordField.getText().compareTo("manger") == 0) {

new CAL3("manger").setVisible(true);

} else {

JOptionPane.showMessageDialog(this, "Password incorrecta", "Error de inicio de sesión", JOptionPane.ERROR\_MESSAGE);

}

}

if (this.jComboBox\_Usuario.getSelectedItem().toString().compareTo("cash") == 0) {

if (this.jPasswordField.getText().compareTo("cash") == 0) {

new CAL3("cash").setVisible(true);

} else {

JOptionPane.showMessageDialog(this, "Password incorrecta", "Error de inicio de sesión", JOptionPane.ERROR\_MESSAGE);

}

}

}

private void jComboBox\_UsuarioActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Login().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButtonInicioSesion;

private javax.swing.JComboBox jComboBox\_Usuario;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JPasswordField jPasswordField;

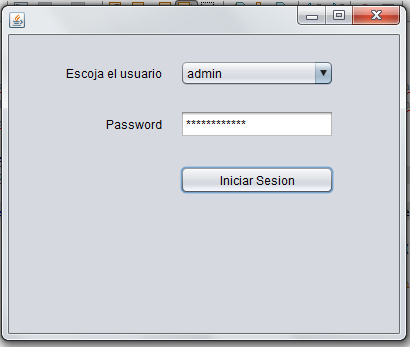
// End of variables declaration

}

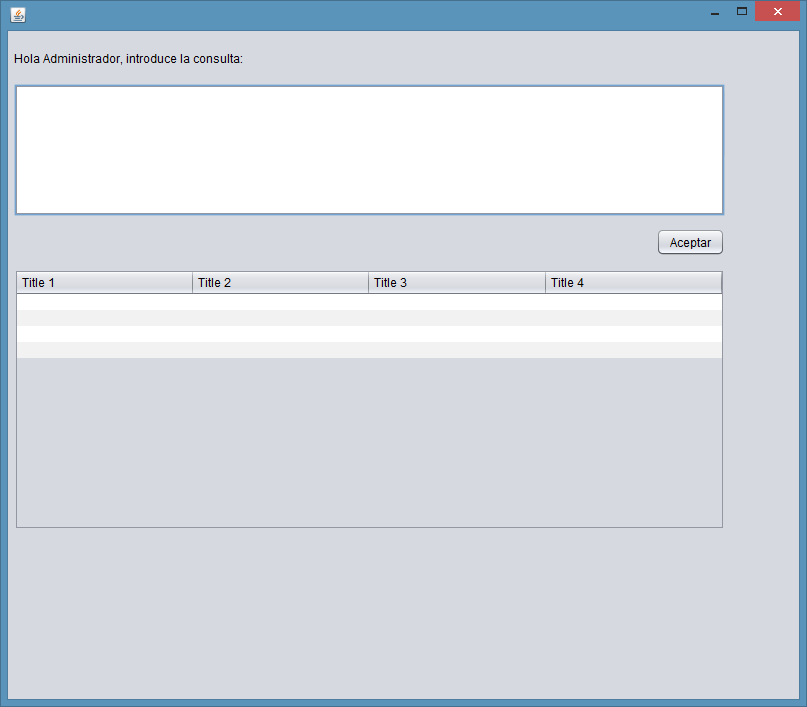
As we have previously done in this we differ by color code: First yellow see the package name; On the other hand gray we appreciate the comments of the program itself and comments already created directly by the program; Red observe all the code that we have inserted for connecting the database; Finally in blue, are the interface buttons and deprived of them created by Neatbeans variables.

## Pec3:

And then we'll see how it turned out we interface this class:  
First we see the sign, which can enter one of the three users



In the next picture we can enter the query to perform and changes the welcome message as the user has entered



If you enter the wrong password we will jump a message like the following:



Finally we see that the query works correctly. We can check with the following code:

package CAL3;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

public class CAL3 extends javax.swing.JFrame {

public static String users;

public CAL3(String users) {

initComponents();

this.users = users;

jLabelSaludo.setText("Hola "+users+", introduce la consulta: ");

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jButton\_Aceptar = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTextArea\_SQL = new javax.swing.JTextArea();

jScrollPane2 = new javax.swing.JScrollPane();

jTable\_Datos = new javax.swing.JTable();

jLabelSaludo = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jButton\_Aceptar.setText("Aceptar");

jButton\_Aceptar.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton\_AceptarActionPerformed(evt);

}

});

jTextArea\_SQL.setColumns(20);

jTextArea\_SQL.setRows(5);

jScrollPane1.setViewportView(jTextArea\_SQL);

jTable\_Datos.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{null, null, null, null},

{null, null, null, null},

{null, null, null, null},

{null, null, null, null}

},

new String [] {

"Title 1", "Title 2", "Title 3", "Title 4"

}

));

jScrollPane2.setViewportView(jTable\_Datos);

jLabelSaludo.setText("jLabel1");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton\_Aceptar)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jScrollPane1)

.addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT\_SIZE, 711, Short.MAX\_VALUE)))

.addComponent(jLabelSaludo))

.addContainerGap(74, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(19, 19, 19)

.addComponent(jLabelSaludo)

.addGap(18, 18, 18)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 132, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jButton\_Aceptar)

.addGap(13, 13, 13)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 261, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(169, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButton\_AceptarActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try{

try {

Class.forName("org.postgresql.Driver");

} catch (ClassNotFoundException cnfe) {

JOptionPane.showMessageDialog(this, "driver no disponible", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

Connection c = null;

int numero=0;

ResultSet rs = null;

Statement s = null;

BufferedReader entrada=new BufferedReader(new InputStreamReader(System.in));

try{

if(users.compareTo("admin") == 0){

c = DriverManager.getConnection("jdbc:postgresql://localhost:5432/PECL2", "admin", "admin");

}

if(users.compareTo("manger") == 0){

c = DriverManager.getConnection("jdbc:postgresql://localhost:5432/PECL2","manger", "manger");

}

if(users.compareTo("cash") == 0){

c = DriverManager.getConnection("jdbc:postgresql://localhost:5432/PECL2","cash", "cash");

}

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "No se pudo realizar la conexión: " + se.toString(), "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

try {

s = c.createStatement();

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "problema al crear la consulta", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

try {

String variableSQL = this.jTextArea\_SQL.getText().trim() + ";";

rs =s.executeQuery(variableSQL) ;

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "Excepción al ejecutar consulta: error de sintaxis en el SQL", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

DefaultTableModel modelo = new DefaultTableModel();

this.jTable\_Datos.setModel(modelo);

ResultSetMetaData rsmd = rs.getMetaData();

int numberOfColumns = rsmd.getColumnCount();

Object[] etiquetas = new Object[numberOfColumns];

Object[] BD = new Object[numberOfColumns];

try {

for (int i = 0; i < numberOfColumns; i++)

{

// Nuevamente, para ResultSetMetaData la primera columna es la 1.

etiquetas[i] = rsmd.getColumnLabel(i + 1);

}

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "Error grave al recoger los resultados3", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

modelo.setColumnIdentifiers(etiquetas);

int aux = 0;

try {

while (rs.next()) {

for (int i = 0; i < numberOfColumns; i++)

{

// Nuevamente, para ResultSetMetaData la primera columna es la 1.

BD[i] = rs.getString(i + 1);

}

modelo.addRow(BD);

aux++;

}

} catch (SQLException se) {

JOptionPane.showMessageDialog(this, "Error grave al recoger los resultados", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}

rs.close();

s.close();

c.close();

} catch (SQLException ex) {

JOptionPane.showMessageDialog(this, "Error grave1", "Mensaje", JOptionPane.INFORMATION\_MESSAGE);

}catch(Exception e){

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(CAL3.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new CAL3(users).setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton\_Aceptar;

private javax.swing.JLabel jLabelSaludo;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

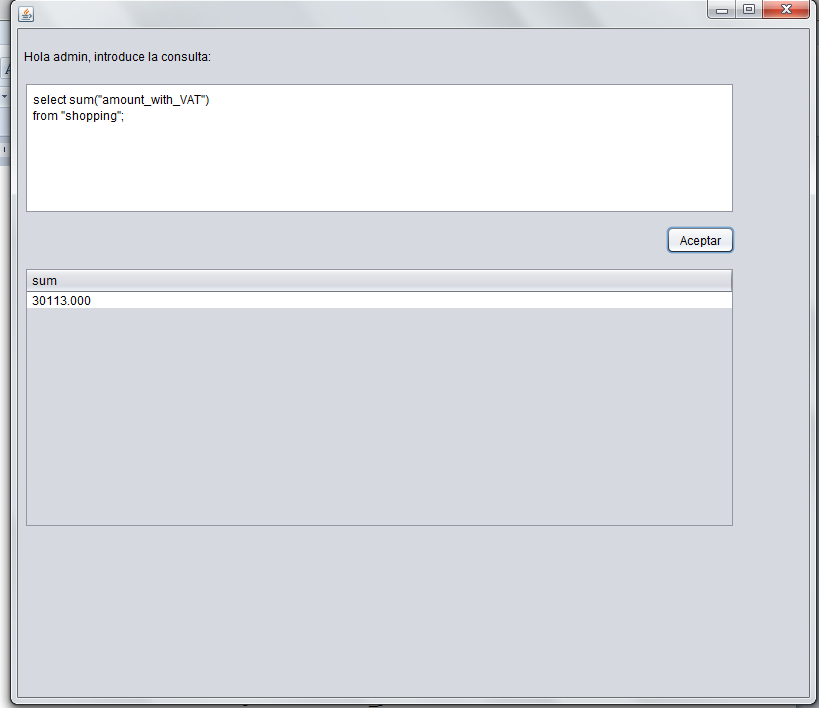
private javax.swing.JTable jTable\_Datos;

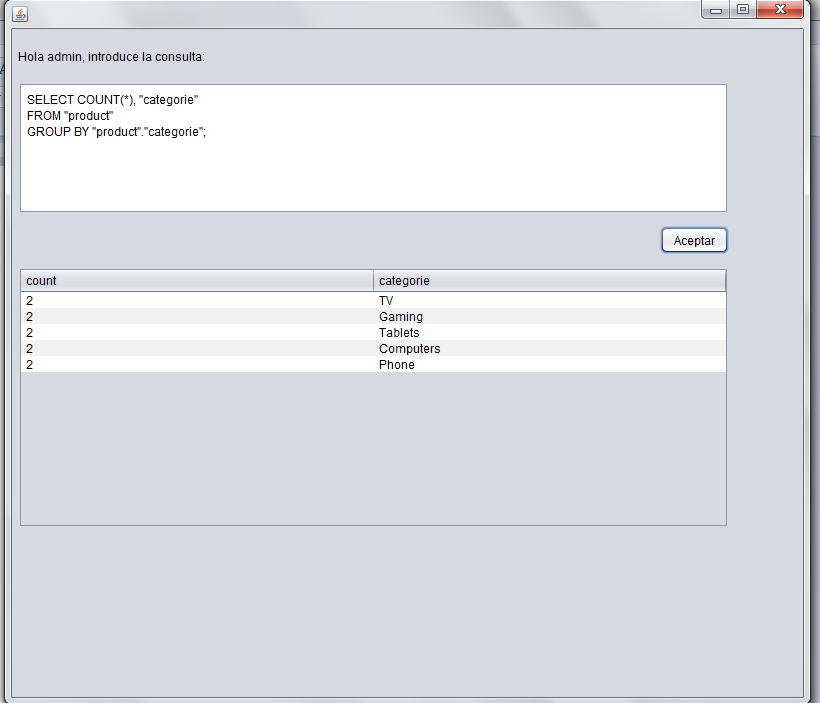
private javax.swing.JTextArea jTextArea\_SQL;

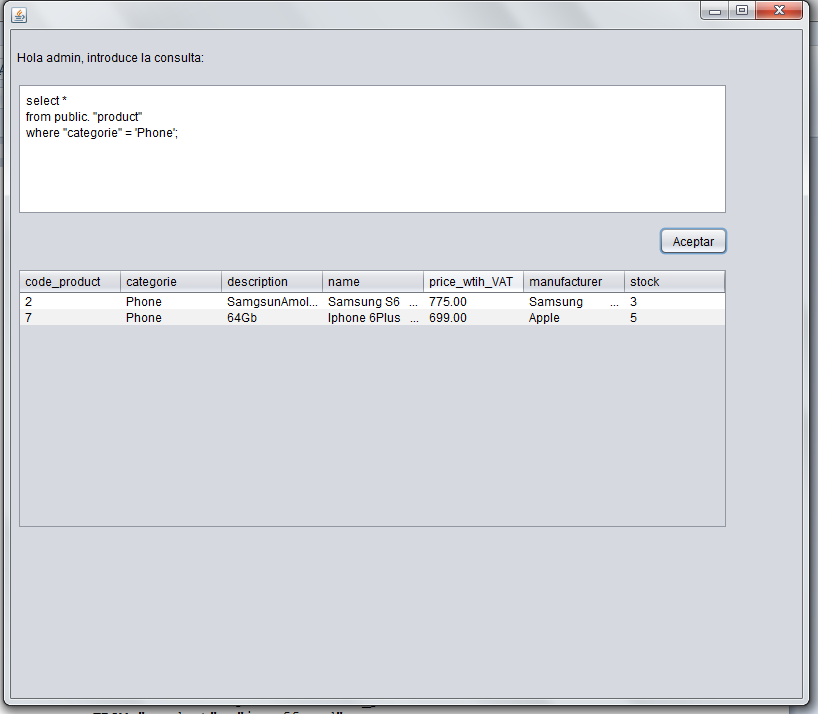
// End of variables declaration

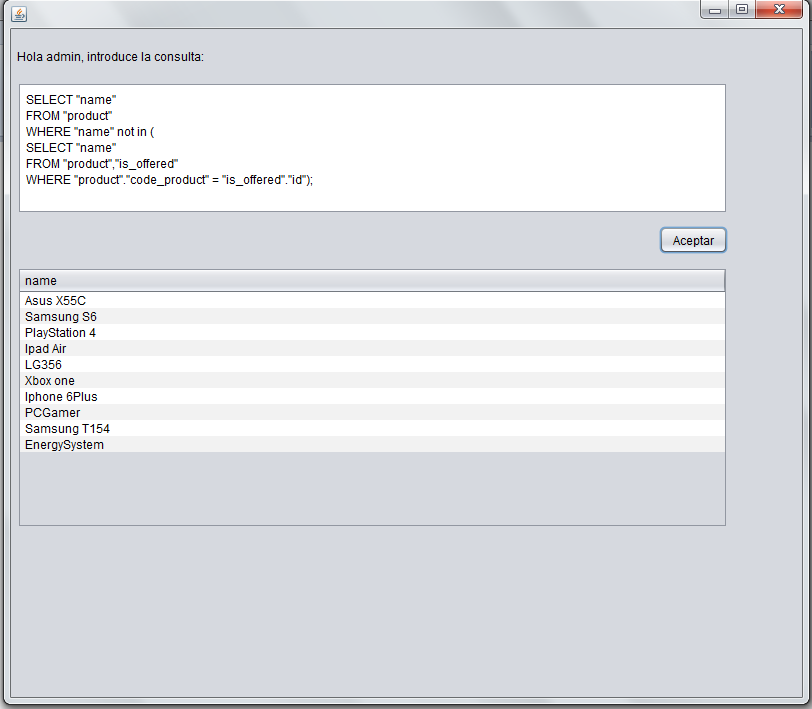
}

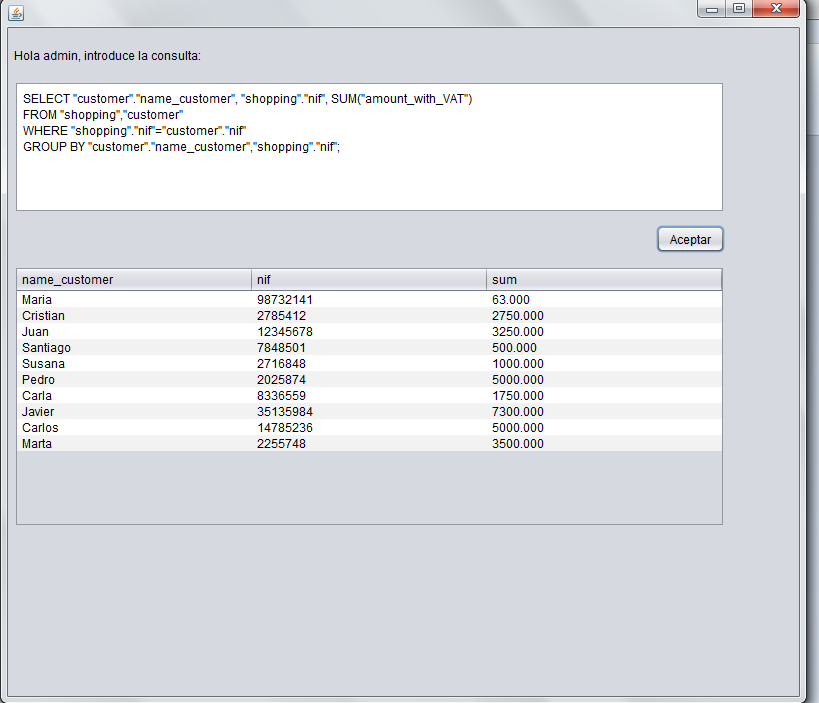
As we have previously done in this we differ by color code: First yellow see the package name; On the other hand gray we appreciate the comments of the program itself and comments already created directly by the program; Red observe all the code that we have inserted for connecting the database; Finally in blue, are the interface buttons and deprived of them created by Neatbeans variables.

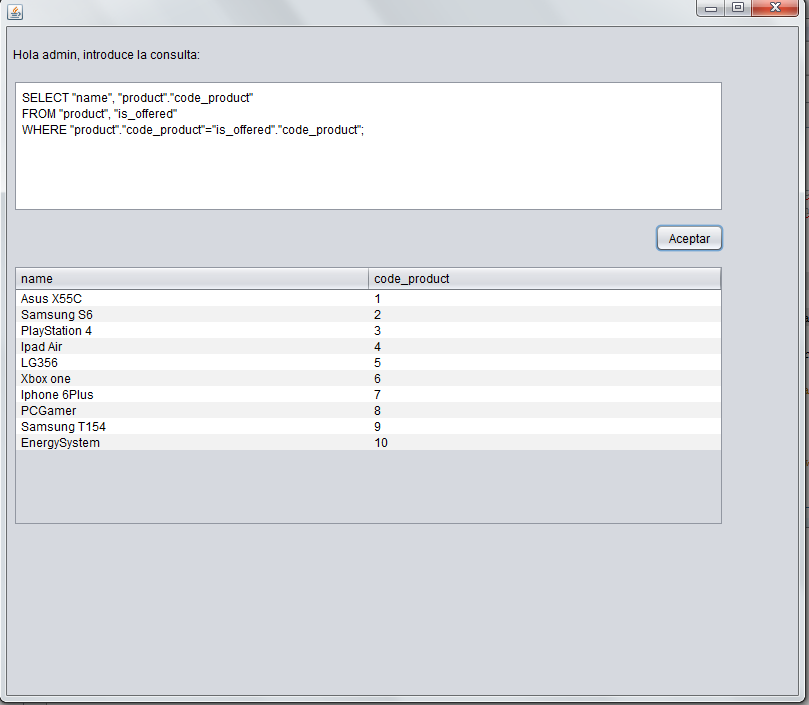


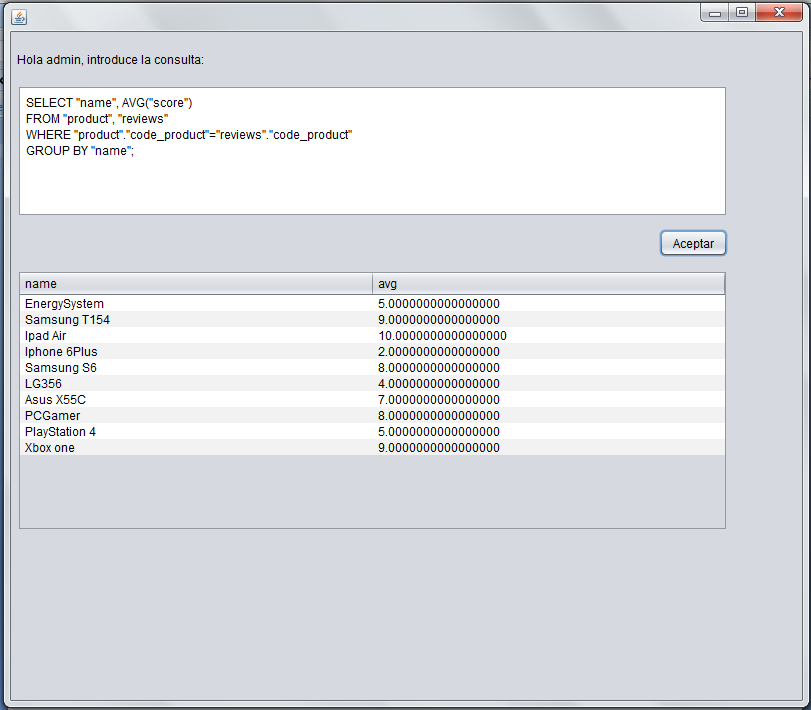


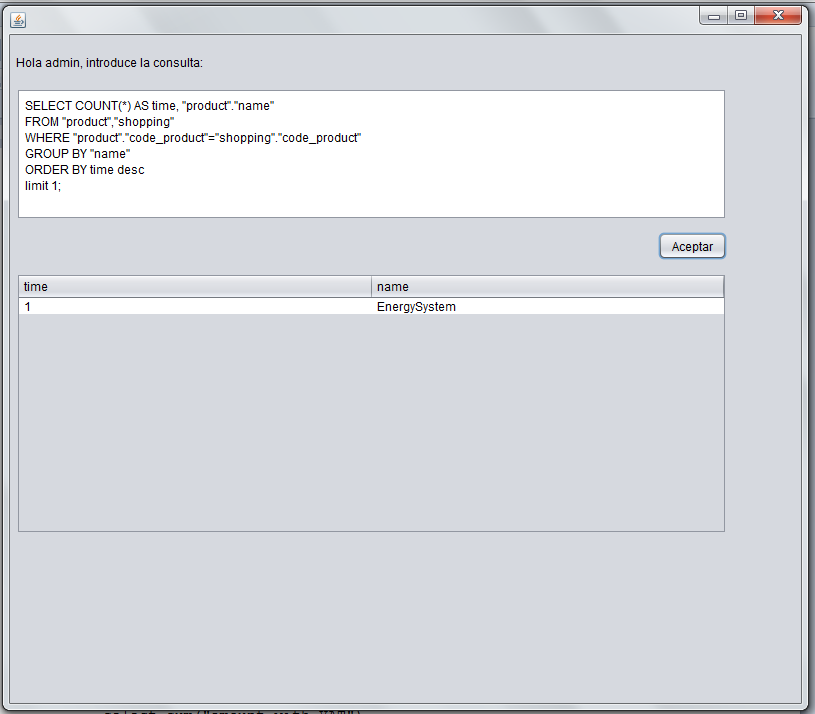


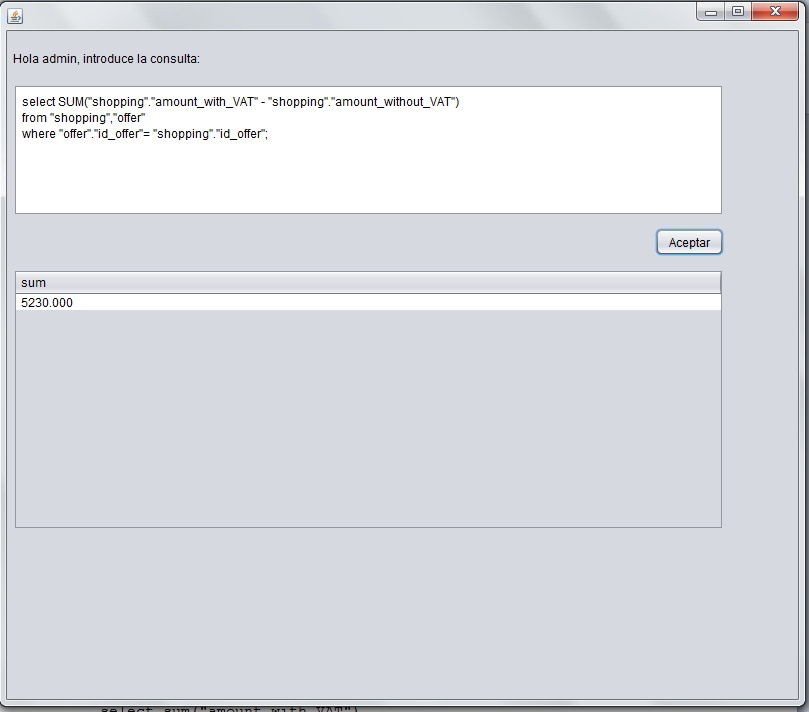


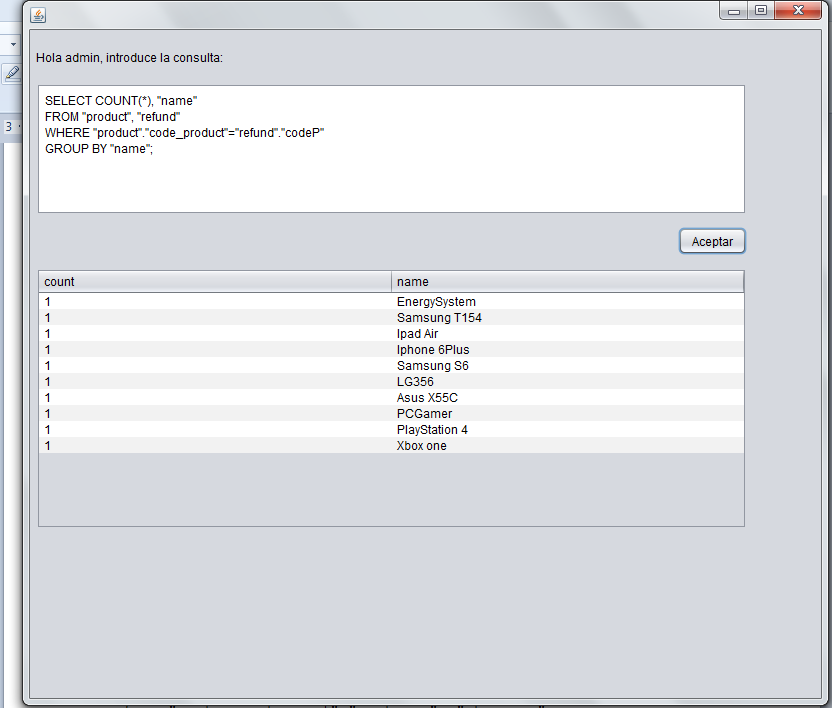


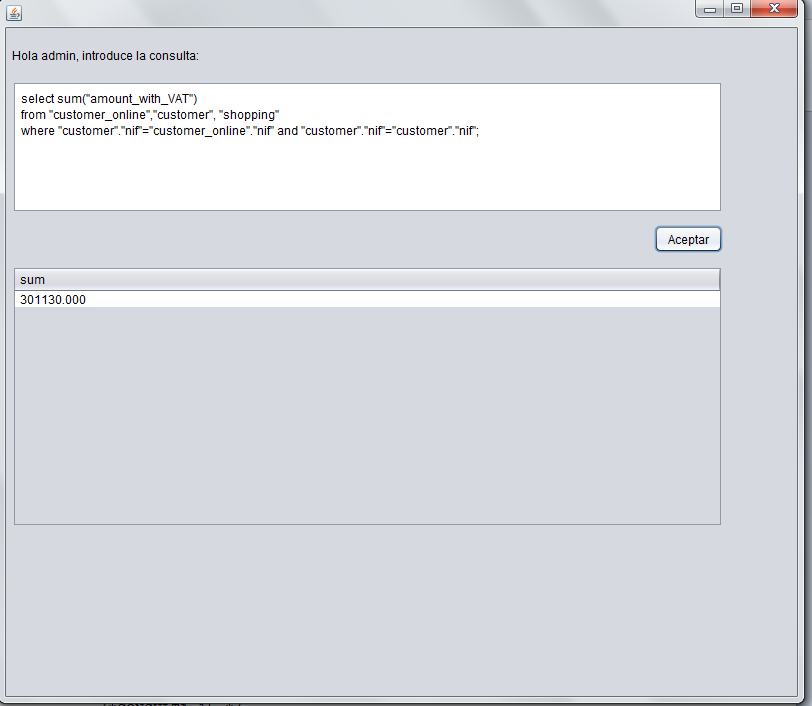


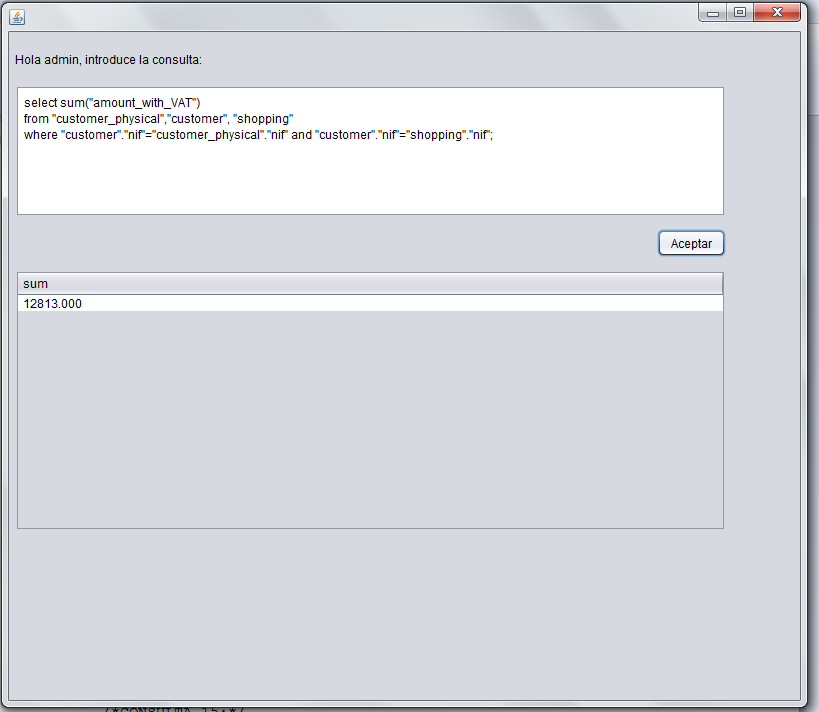


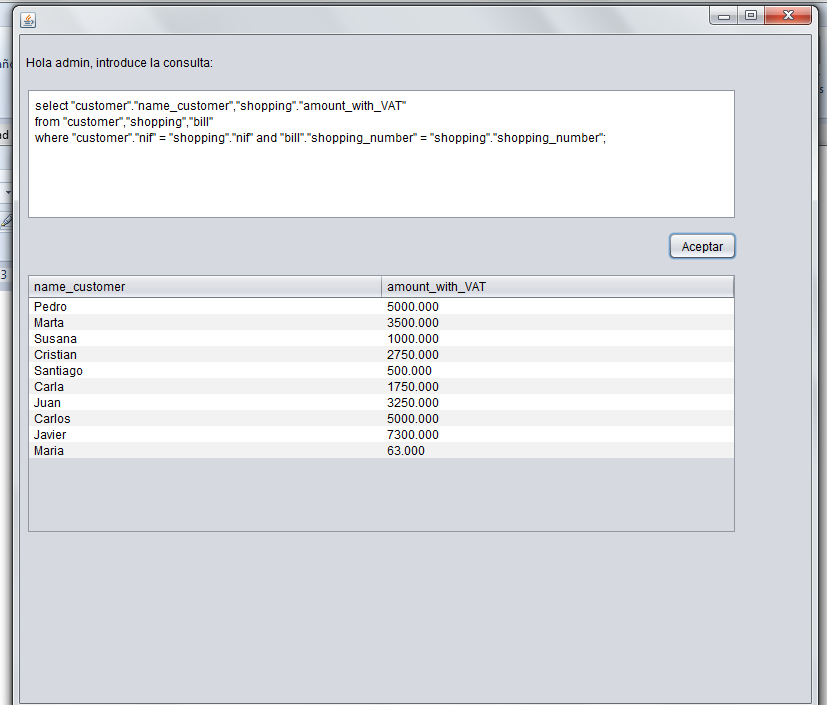


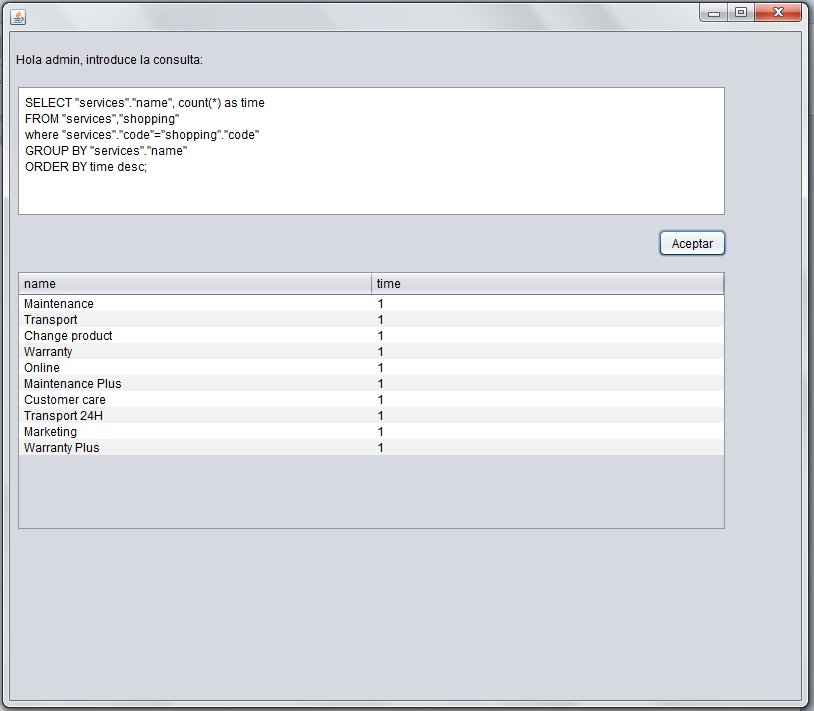


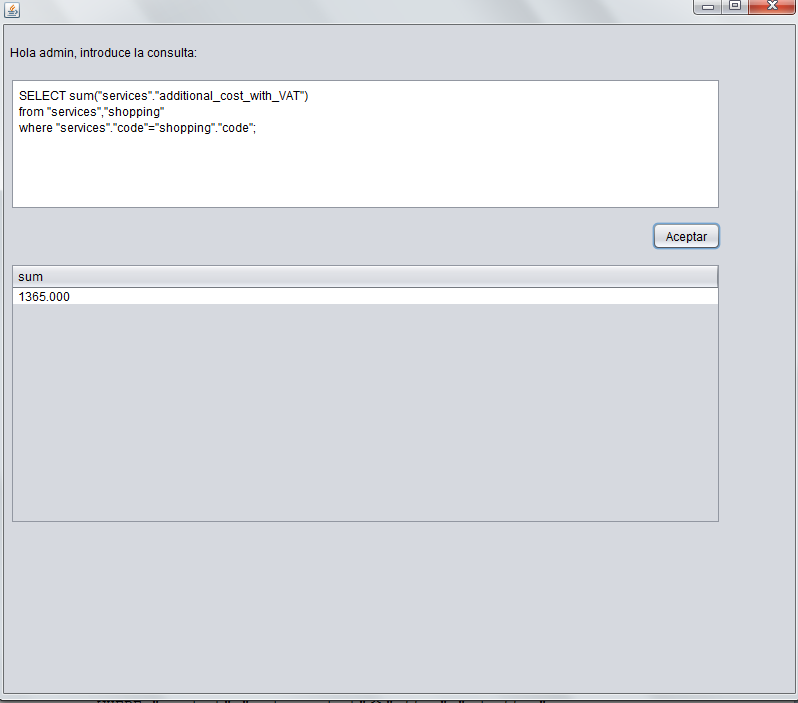


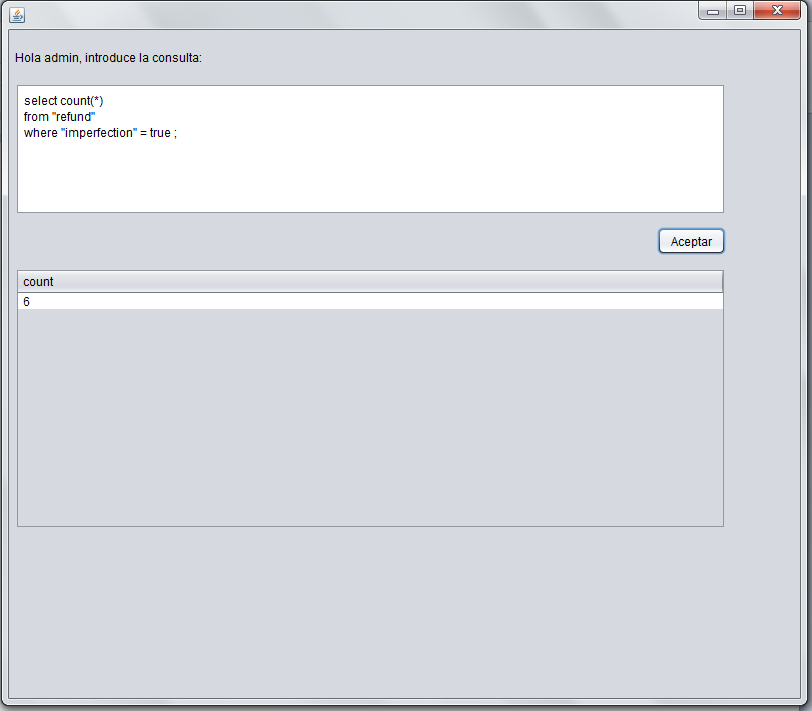


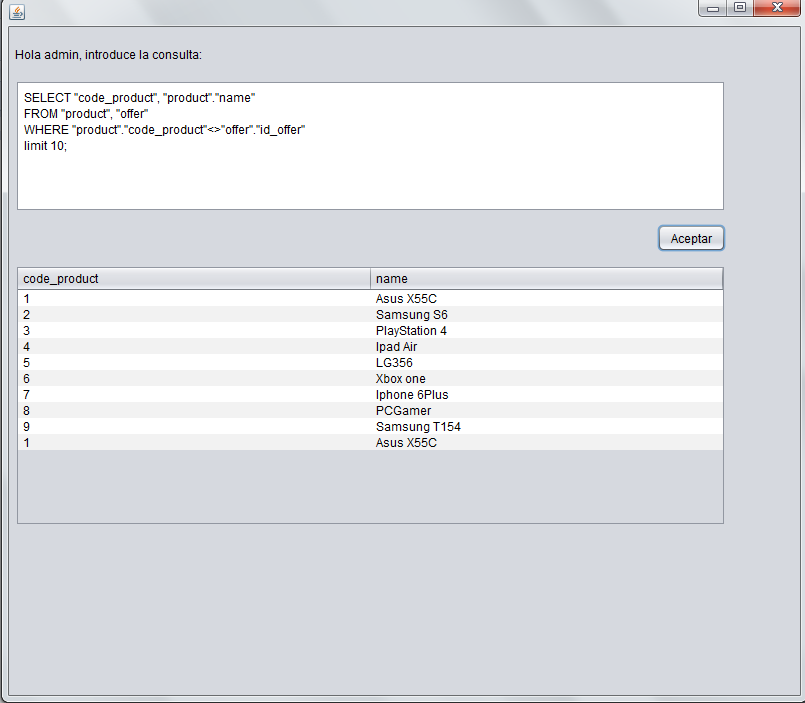


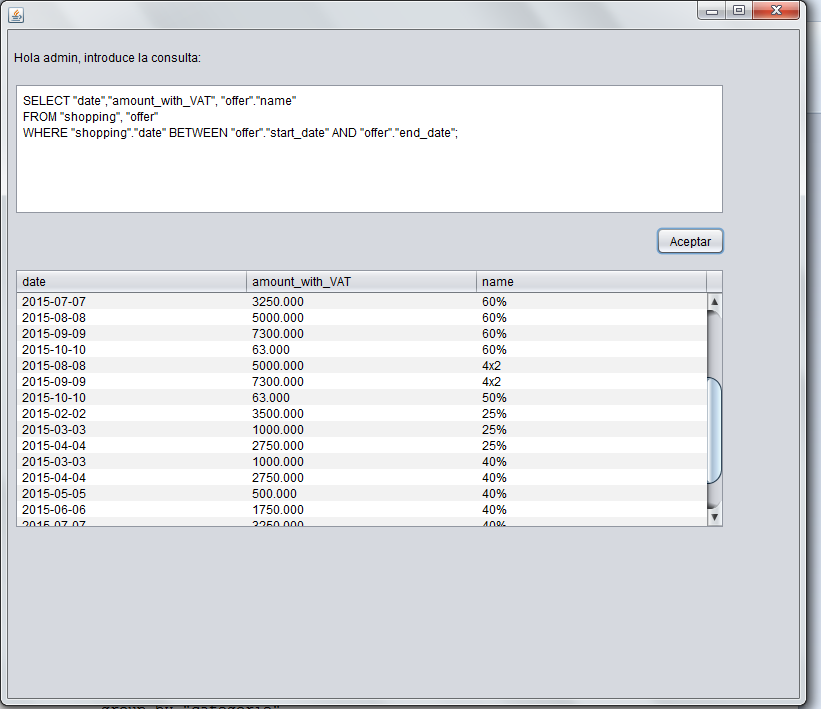


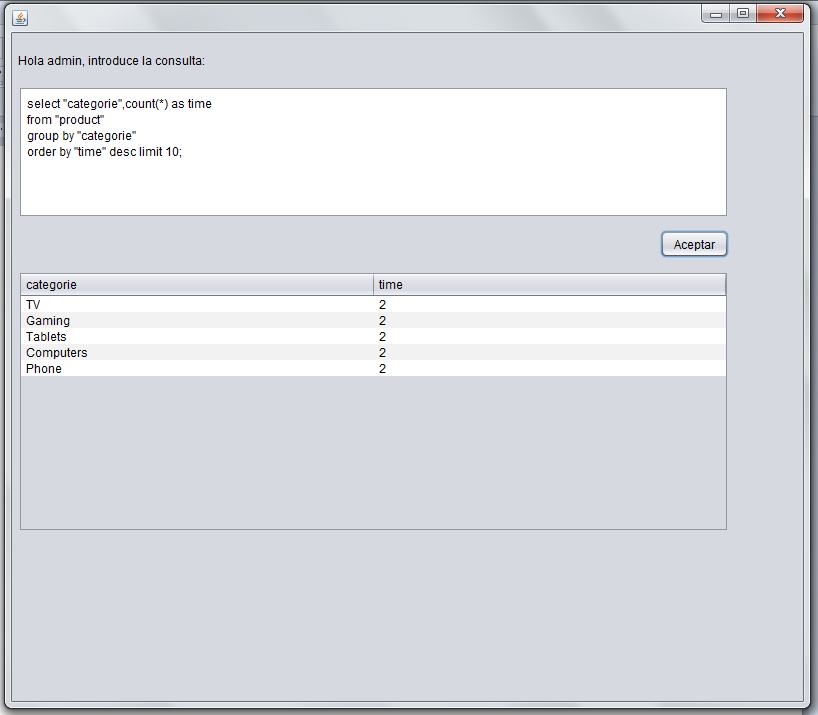




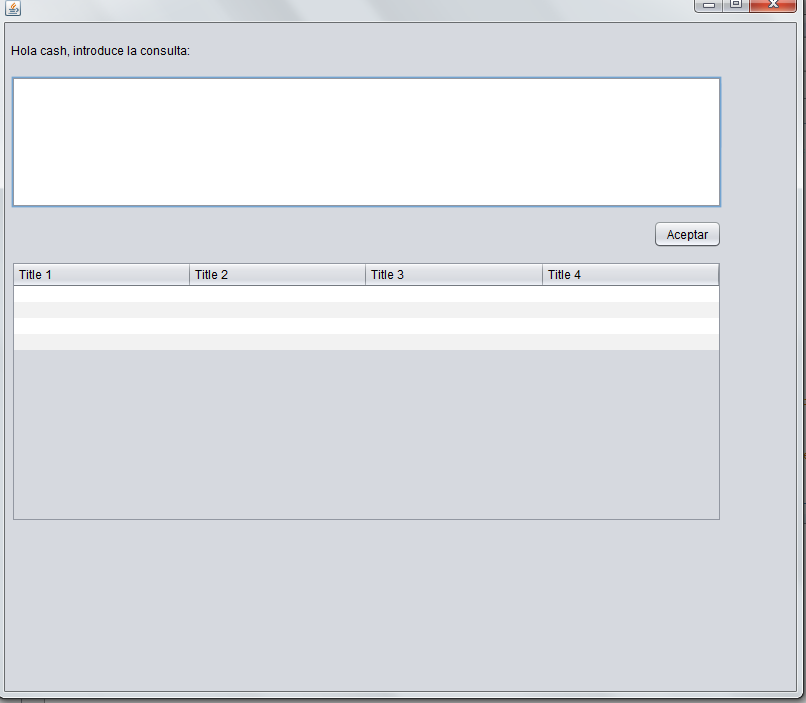








We can go as cash:



We can go as manager:

