Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №3**

По дисциплине: «Современные платформы программирования»

Выполнила:

Студентка 3 курса

Группы ПО-6

Юсковец М.А.

Проверил:

Монтик Н.С.

Брест, 2023

**Цель работы:** приобрести практические навыки разработки многооконных приложений на JavaFX для работы с базами данных.

Ход работы:

**Задание:**

(Вариант 25)

На основе БД, разработанной в лабораторной работе No9, реализовать многооконное приложение-клиент, позволяющее выполнять основные операции над таблицей в БД (добавление, удаление, модификацию данных).

Основные требования к приложению:

• Для отображения выбирать таблицу с внешними ключами;

• Осуществлять вывод основных данных в табличном представлении;

• При выводе краткого представления записи в таблице (т.е. если выводятся не все поля), по щелчку мышкой на запись осуществлять вывод всех полей в подготовленные компоненты на форме;

• Для всех полей, представленных внешними ключами, выводить их текстовое представление из связанных таблиц (например, таблица-справочник «Времена года» содержит два поля – идентификатор и название сезона, в связанной таблице «Месяц года» есть внешний ключ на таблицу «Времена года»; в этом случае при выводе таблицы «Месяц года» нужно выводить название сезона, а не его идентификатор);

• При выводе предусмотреть упорядочивание по столбцу;

• Реализовать простейший фильтр данных по одному-двум полям;

• При добавлении новых данных в таблицу использовать дополнительное окно для ввода;

• При модификации данных можно использовать ту же форму, что и для добавления, но с внесенными актуальными значениями полей;

• При добавлении/модификации выводить варианты значений полей с внешним ключом с помощью выпадающего списка;

• При удалении данных осуществлять удаление записи, на которой в данных момент находится фокус.

Текст программы:

Main.java

package com.example.spp\_lab3;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
import java.lang.reflect.InvocationTargetException;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
  
  
public class Main extends Application {  
 private static Scene *scene*;  
 public static Connection *conn*;  
  
 @Override  
 public void start(Stage stage) throws IOException, ClassNotFoundException, SQLException, NoSuchMethodException, InstantiationException, IllegalAccessException, IllegalArgumentException, InvocationTargetException {  
 String url = "jdbc:mysql://localhost:3306/warehouse";  
 String username = "root";  
 String password = "";  
 *conn* = DriverManager.*getConnection*(url, username, password);  
  
 *scene* = new Scene(*loadFXML*("mainWindow"), 640, 350);  
 stage.setScene(*scene*);  
 stage.show();  
 }  
  
 static void setRoot(String fxml) throws IOException {  
 *scene*.setRoot(*loadFXML*(fxml));  
 }  
  
 private static Parent loadFXML(String fxml) throws IOException {  
 FXMLLoader fxmlLoader = new FXMLLoader(Main.class.getResource(fxml + ".fxml"));  
 return fxmlLoader.load();  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
}

Controller.java

package com.example.spp\_lab3;  
  
import java.io.IOException;  
  
import javafx.fxml.FXML;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.GridPane;  
import javafx.stage.Modality;  
import javafx.stage.Stage;  
import javafx.stage.StageStyle;  
  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.util.Objects;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.scene.control.Alert.AlertType;  
import javafx.scene.control.cell.PropertyValueFactory;  
  
public class Controller {  
 @FXML  
 private TableView storagesTable;  
 @FXML  
 private ComboBox filterCombo;  
 @FXML  
 private Button showTableButton;  
 @FXML  
 private Button addButton;  
 @FXML  
 private Button editButton;  
 @FXML  
 private Button deleteButton;  
 @FXML  
 private Button refreshButton;  
 @FXML  
 private GridPane recordForm;  
 @FXML  
 private Label idRowLabel;  
 @FXML  
 private Label productRowLabel;  
 @FXML  
 private Label warehouseRowLabel;  
 @FXML  
 private Label amountRowLabel;  
 @FXML  
 private Label idLabel;  
 @FXML  
 private Label productLabel;  
 @FXML  
 private Label warehouseLabel;  
 @FXML  
 private Label amountLabel;  
 @FXML  
 private TableColumn idColumn;  
 @FXML  
 private TableColumn productColumn;  
 @FXML  
 private TableColumn warehouseColumn;  
 @FXML  
 private TableColumn amountColumn;  
  
 public Warehouse warehouseFilter = null;  
  
 @FXML  
 public void initialize() throws IOException, SQLException {  
 storagesTable.getSelectionModel().setSelectionMode(SelectionMode.*MULTIPLE*);  
  
 Statement statement = Main.*conn*.createStatement();  
  
 String sqlCommand = "SELECT \* FROM warehouses";  
 ResultSet resultSet = statement.executeQuery(sqlCommand);  
 ObservableList<Warehouse> warehouses = FXCollections.*observableArrayList*();  
  
 while (resultSet.next()) {  
 warehouses.add(  
 new Warehouse(  
 resultSet.getInt("number\_warehouse"),  
 resultSet.getString("name"),  
 resultSet.getString("address"),  
 resultSet.getString("phone"),  
 resultSet.getInt("companyid")  
 )  
 );  
 }  
  
 filterCombo.setItems(warehouses);  
 showTableButtonClicked();  
 }  
  
 @FXML  
 private void switchToSecondary() throws IOException {  
 FXMLLoader loader = new FXMLLoader(getClass().getResource("addEditWindow.fxml"));  
  
 Stage stage = new Stage(StageStyle.*DECORATED*);  
 stage.setScene(new Scene(loader.load()));  
  
 SecondaryController controller = loader.getController();  
 stage.initModality(Modality.*APPLICATION\_MODAL*);  
 stage.show();  
 }  
  
 @FXML  
 private void refreshButtonClicked() throws IOException, SQLException {  
 filterCombo.setValue(null);  
 showTableButtonClicked();  
 }  
  
 @FXML  
 private void showTableButtonClicked() throws IOException, SQLException {  
 Statement statement = Main.*conn*.createStatement();  
 String sqlCommand;  
 ResultSet resultSet;  
  
 warehouseFilter = (Warehouse) filterCombo.getValue();  
  
 if (Objects.*isNull*(warehouseFilter))  
 sqlCommand = "SELECT storage.idstorage, product.name, warehouses.name, storage.amount FROM storage INNER JOIN product ON product.idproduct = storage.product INNER JOIN warehouses ON warehouses.number\_warehouse = storage.warehouse";  
 else  
 sqlCommand = "SELECT storage.idstorage, product.name, warehouses.name, storage.amount FROM storage INNER JOIN product ON product.idproduct = storage.product INNER JOIN warehouses ON warehouses.number\_warehouse = storage.warehouse WHERE warehouses.number\_warehouse = '" + warehouseFilter.getId\_warehouse() +"'";  
 resultSet = statement.executeQuery(sqlCommand);  
  
 ObservableList<Storage> storages = FXCollections.*observableArrayList*();  
  
 while (resultSet.next()) {  
 storages.add(  
 new Storage(  
 resultSet.getInt("idstorage"),  
 resultSet.getString("product.name"),  
 resultSet.getString("warehouses.name"),  
 resultSet.getInt("amount")  
 )  
 );  
 }  
  
 storagesTable.setItems(storages);  
  
 productColumn.setCellValueFactory(new PropertyValueFactory<Storage, String>("product\_name"));  
 warehouseColumn.setCellValueFactory(new PropertyValueFactory<Storage, String>("warehouse\_name"));  
 amountColumn.setCellValueFactory(new PropertyValueFactory<Storage, Integer>("amount"));  
 }  
  
 @FXML  
 private void newButtonClicked() throws IOException, SQLException {  
 FXMLLoader fxmlLoader = new FXMLLoader(Main.class.getResource("addEditWindow.fxml"));  
  
 Stage stage = new Stage(StageStyle.*DECORATED*);  
 stage.setScene(new Scene(fxmlLoader.load()));  
  
 SecondaryController controller = fxmlLoader.getController();  
  
 stage.initModality(Modality.*APPLICATION\_MODAL*);  
  
 Statement statement = Main.*conn*.createStatement();  
 String sqlCommand = "SELECT \* FROM product";  
 ResultSet resultSet = statement.executeQuery(sqlCommand);  
 ObservableList<Product> products = FXCollections.*observableArrayList*();  
  
 while (resultSet.next()) {  
 products.add(  
 new Product(  
 resultSet.getInt("idproduct"),  
 resultSet.getString("name"),  
 resultSet.getString("model"),  
 resultSet.getInt("price")  
 )  
 );  
 }  
  
 sqlCommand = "SELECT \* FROM warehouses";  
 resultSet = statement.executeQuery(sqlCommand);  
 ObservableList<Warehouse> warehouses = FXCollections.*observableArrayList*();  
  
 while (resultSet.next()) {  
 warehouses.add(  
 new Warehouse(  
 resultSet.getInt("number\_warehouse"),  
 resultSet.getString("name"),  
 resultSet.getString("address"),  
 resultSet.getString("phone"),  
 resultSet.getInt("companyid")  
 )  
 );  
 }  
  
 controller.productCombo.setItems(products);  
 controller.warehouseCombo.setItems(warehouses);  
  
 Storage storage = controller.showDialog(stage, null, null, 0);  
  
 if (storage.getProduct\_name() == null || storage.getWarehouse\_name() == null || storage.getAmount() == null) {  
 Alert alert = new Alert(AlertType.*INFORMATION*);  
 alert.setTitle("Adding error");  
 alert.setHeaderText("");  
 alert.setContentText("All fields must be filled in!");  
  
 alert.showAndWait();  
  
 return;  
 }  
  
 sqlCommand = "INSERT INTO storage (product, warehouse, amount) VALUES ('" + Integer.*valueOf*(storage.getProduct\_name()) + "', '" + Integer.*valueOf*(storage.getWarehouse\_name()) + "', '" + storage.getAmount() + "')";  
 statement.execute(sqlCommand);  
  
 showTableButtonClicked();  
 }  
  
  
 @FXML  
 private void editButtonClicked() throws IOException, SQLException {  
 FXMLLoader fxmlLoader = new FXMLLoader(Main.class.getResource("addEditWindow.fxml"));  
  
 Stage stage = new Stage(StageStyle.*DECORATED*);  
 stage.setScene(new Scene(fxmlLoader.load()));  
  
 SecondaryController controller = fxmlLoader.getController();  
  
 stage.initModality(Modality.*APPLICATION\_MODAL*);  
  
 Storage selectedStorage = (Storage) storagesTable.getSelectionModel().getSelectedItem();  
 Product selectedProduct = null;  
 Warehouse selectedWarehouse = null;  
 Integer selectedProductId = null;  
 Integer selectedWarehouseId = null;  
  
 Statement statement = Main.*conn*.createStatement();  
 String sqlCommand = "SELECT product, warehouse FROM storage WHERE idstorage = '" + selectedStorage.getId\_storage() + "'";  
 ResultSet resultSet = statement.executeQuery(sqlCommand);  
 while (resultSet.next()) {  
 selectedProductId = resultSet.getInt("product");  
 selectedWarehouseId = resultSet.getInt("warehouse");  
 }  
  
  
 sqlCommand = "SELECT \* FROM product";  
 resultSet = statement.executeQuery(sqlCommand);  
 ObservableList<Product> products = FXCollections.*observableArrayList*();  
  
 while (resultSet.next()) {  
 if (resultSet.getInt("idproduct") == selectedProductId) {  
 selectedProduct = new Product(  
 resultSet.getInt("idproduct"),  
 resultSet.getString("name"),  
 resultSet.getString("model"),  
 resultSet.getInt("price")  
 );  
 }  
  
 products.add(  
 new Product(  
 resultSet.getInt("idproduct"),  
 resultSet.getString("name"),  
 resultSet.getString("model"),  
 resultSet.getInt("price")  
 )  
 );  
 }  
  
 sqlCommand = "SELECT \* FROM warehouses";  
 resultSet = statement.executeQuery(sqlCommand);  
 ObservableList<Warehouse> warehouses = FXCollections.*observableArrayList*();  
  
 while (resultSet.next()) {  
 if (resultSet.getInt("number\_warehouse") == selectedWarehouseId) {  
 selectedWarehouse = new Warehouse(  
 resultSet.getInt("number\_warehouse"),  
 resultSet.getString("name"),  
 resultSet.getString("address"),  
 resultSet.getString("phone"),  
 resultSet.getInt("companyid")  
 );  
 }  
  
 warehouses.add(  
 new Warehouse(  
 resultSet.getInt("number\_warehouse"),  
 resultSet.getString("name"),  
 resultSet.getString("address"),  
 resultSet.getString("phone"),  
 resultSet.getInt("companyid")  
 )  
 );  
 }  
  
 controller.productCombo.setItems(products);  
 controller.warehouseCombo.setItems(warehouses);  
  
 Storage storage = controller.showDialog(stage, selectedProduct, selectedWarehouse, selectedStorage.getAmount());  
  
 if (storage.getProduct\_name() == null || storage.getWarehouse\_name() == null || storage.getAmount() == null) {  
 Alert alert = new Alert(AlertType.*INFORMATION*);  
 alert.setTitle("Editing error");  
 alert.setHeaderText("");  
 alert.setContentText("All fields must be filled in!");  
  
 alert.showAndWait();  
  
 return;  
 }  
  
 sqlCommand = "UPDATE storage SET product = '" + Integer.*valueOf*(storage.getProduct\_name()) + "', warehouse = '" + Integer.*valueOf*(storage.getWarehouse\_name()) + "', amount = '" + storage.getAmount() + "' WHERE idstorage = '" + selectedStorage.getId\_storage() + "'";  
 statement.execute(sqlCommand);  
  
 showTableButtonClicked();  
 }  
  
 @FXML  
 private void deleteButtonClicked() throws IOException, SQLException {  
 ObservableList<Storage> storages = storagesTable.getSelectionModel().getSelectedItems();  
  
 for (int i = 0; i < storages.size(); i++) {  
 Storage storage = (Storage) storages.get(i);  
 Statement statement = Main.*conn*.createStatement();  
 String sqlCommand = "DELETE FROM storage WHERE idstorage = '" + storage.getId\_storage() + "'";  
 statement.execute(sqlCommand);  
 }  
  
 showTableButtonClicked();  
 }  
  
 @FXML  
 private void handleRowSelect() {  
 Storage row = (Storage) storagesTable.getSelectionModel().getSelectedItem();  
 if (row == null) return;  
  
 idRowLabel.setText(String.*valueOf*(row.getId\_storage()));  
 productRowLabel.setText(row.getProduct\_name());  
 warehouseRowLabel.setText(row.getWarehouse\_name());  
 amountRowLabel.setText(row.getAmount().toString());  
 }  
}

SecondaryController.java

package com.example.spp\_lab3;  
  
import java.io.IOException;  
import java.sql.SQLException;  
import javafx.fxml.FXML;  
import javafx.scene.control.\*;  
  
import javafx.stage.Stage;  
  
public class SecondaryController {  
 @FXML  
 public ComboBox productCombo;  
 @FXML  
 public ComboBox warehouseCombo;  
 @FXML  
 public TextField idAmount;  
 @FXML  
 public Button submitButton;  
  
 public Product product = null;  
 public Warehouse warehouse = null;  
 public Integer amount = null;  
  
 public Storage showDialog(Stage stage, Product productComboValue, Warehouse warehouseComboValue, Integer amountInputValue) throws IOException, SQLException {  
 productCombo.setValue(productComboValue);  
 warehouseCombo.setValue(warehouseComboValue);  
 idAmount.setText(Integer.*toString*(amountInputValue));  
  
 submitButton.setOnAction(e -> {  
 if (productCombo.getValue() != null)  
 product = (Product) productCombo.getValue();  
 if (warehouseCombo.getValue() != null)  
 warehouse = (Warehouse) warehouseCombo.getValue();  
 if (idAmount.getText() != null)  
 amount = Integer.*valueOf*(idAmount.getText());  
 stage.close();  
 });  
  
 stage.showAndWait();  
  
 if (product == null || warehouse == null || amount == null) {  
 return new Storage(0, null, null, 0);  
 }  
  
 return new Storage(0, product.getId\_product().toString(), warehouse.getId\_warehouse().toString(), amount);  
 }  
}

Storage.java

package com.example.spp\_lab3;  
  
public class Storage {  
 private Integer id\_storage;  
 private String product\_name;  
 private String warehouse\_name;  
 private Integer amount;  
  
 public Storage(Integer id, String pr\_name, String war\_name, Integer am) {  
 this.id\_storage = id;  
 this.product\_name = pr\_name;  
 this.warehouse\_name = war\_name;  
 this.amount = am;  
 }  
  
 public Integer getId\_storage() {  
 return this.id\_storage;  
 }  
  
 public String getProduct\_name() {  
 return this.product\_name;  
 }  
  
 public String getWarehouse\_name() {  
 return this.warehouse\_name;  
 }  
  
 public Integer getAmount() {  
 return this.amount;  
 }  
  
 public void setId\_storage(Integer value) {  
 this.id\_storage = value;  
 }  
  
 public void setProduct\_name(String value) {  
 this.product\_name = value;  
 }  
  
 public void setWarehouse\_name(String value) {  
 this.warehouse\_name = value;  
 }  
  
 public void setAmount(Integer value) {  
 this.amount = value;  
 }  
}

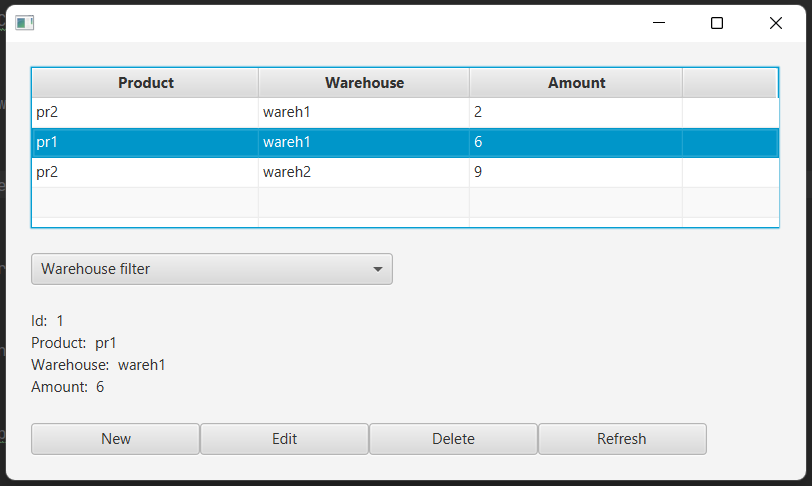
Product.java

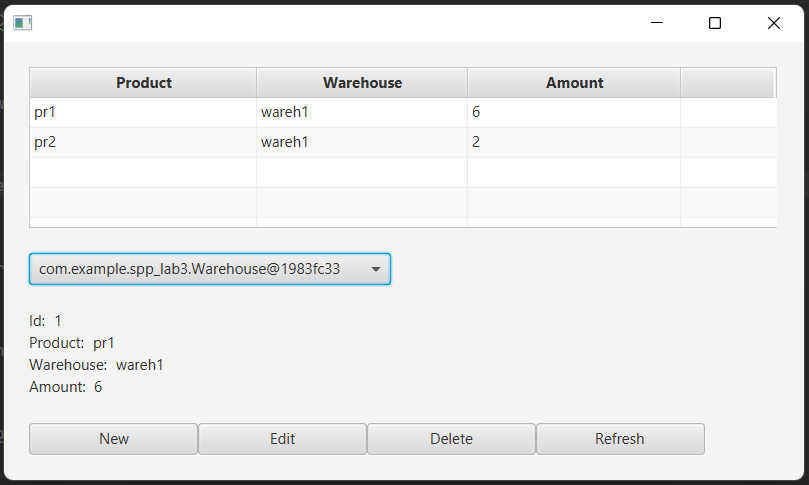
package com.example.spp\_lab3;  
  
public class Product {  
 private Integer id\_product;  
 private String name;  
 private String model;  
 private Integer price;  
  
 public Product(Integer id, String pr\_name, String mod, Integer pr) {  
 this.id\_product = id;  
 this.name = pr\_name;  
 this.model = mod;  
 this.price = pr;  
 }  
  
 public Integer getId\_product() {  
 return this.id\_product;  
 }  
  
 public String getName() {  
 return this.name;  
 }  
  
 public String getModel() {  
 return this.model;  
 }  
  
 public Integer getPrice() {  
 return this.price;  
 }  
  
 public void setId\_product(Integer value) {  
 this.id\_product = value;  
 }  
  
 public void setName(String value) {  
 this.name = value;  
 }  
  
 public void setModel(String value) {  
 this.model = value;  
 }  
  
 public void setPrice(Integer value) {  
 this.price = value;  
 }  
}

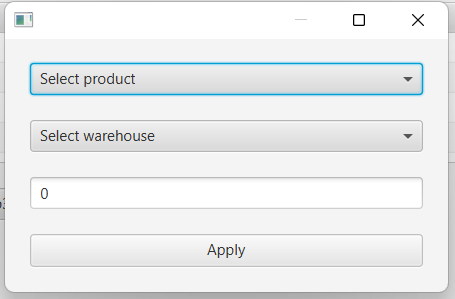
Warehouse.java

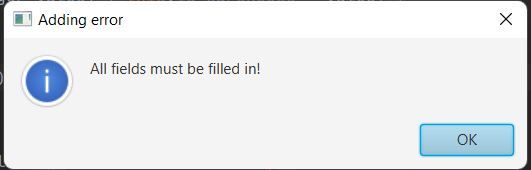
package com.example.spp\_lab3;  
  
public class Warehouse {  
 private Integer id\_warehouse;  
 private String name;  
 private String address;  
 private String phone;  
 private Integer companyid;  
  
 public Warehouse(Integer id, String nm, String addr, String ph, Integer ci) {  
 this.id\_warehouse = id;  
 this.name = nm;  
 this.address = addr;  
 this.phone = ph;  
 this.companyid = ci;  
 }  
  
 public Integer getId\_warehouse() {  
 return this.id\_warehouse;  
 }  
  
 public String getName() {  
 return this.name;  
 }  
  
 public String getAddress() {  
 return this.address;  
 }  
  
 public String getPhone() {  
 return this.phone;  
 }  
  
 public Integer getCompanyid() {  
 return this.companyid;  
 }  
  
 public void setId\_warehouse(Integer value) {  
 this.id\_warehouse = value;  
 }  
  
 public void setName(String value) {  
 this.name = value;  
 }  
  
 public void setAddress(String value) {  
 this.address = value;  
 }  
  
 public void setPhone(String value) {  
 this.phone = value;  
 }  
  
 public void setCompanyid(Integer value) {  
 this.companyid = value;  
 }  
}

Результат программы:









**Вывод:** приобрели практические навыки разработки многооконных приложений на JavaFX для работы с базами данных.