**Лабораторная работа 6**

**Разработка пользовательского интерфейса для работы с базами данных**

**Цель работы:**

Научиться разрабатывать GUI-приложения с использованием баз данных.

**Задание:**

Разработать подсистему учета и регистрации посещений поликлиники больными.

Рекомендуется использовать классы *TextArea*, *Choice,* *диалог фильтрации*.

**Порядок выполнения работы:**

С использованием графического интерфейса пользователя требуется разработать приложение, взаимодействующее с базой данных.

Приложение должно позволять:

1. добавлять, удалять, редактировать записи;

1. осуществлять поиск информации;

1. осуществлять сортировку информации;

1. сохранять результаты в файл.

В качестве СУБД рекомендовано использовать файловую БД(*Microsoft Access, DBase* ит.п. В базе данных должны присутствовать поля разных типов

(минимальные требования: числа, текст, дата, логический тип).

**Рекомендации по выполнению лабораторной работы:**

**Последовательность работы с базами данных**

Приложение, работающее с базами данных, имеет следующую обычную последовательность действий:

1. Загрузка класса драйвера базы данных при отсутствии экземпляра этого класса.

Для СУБД *MySQL:*

*String driverName=”org.gjt.mm.mysql.Driver”;*

Для СУБД *MsAccess:*

*String driverName=”sun.jdbc.odbc.JdbcOdbcDriver”;*

После этого выполняется собственно загрузка драйвера в память:

*Class.forName(driverName);*

и становится возможным соединение с СУБД.

Эти же действия можно выполнить, импортируя библиотеку и создавая объект явно:

*Import СОМ.ibm.db2.jdbc.net. DB2Driver;*

а затем

*new Db2Driver();*

для СУБД *DB2.*

1. Установка соединения с базами данных в виде:

*Connection сп=*

*DriverManager.getConnection*

*(“jdbc:mysql://localhost/mydb”, ’’login ”, ’’pass ”);*

или

*Connection cn=*

*DriverManager.getConnection(“jdbc:odbc:Konditerskaya”);*

В результате будет возвращен объект *Connection* и будет установлено соединение с соответствующей базой данных.

Класс *DriverManager* предоставляет средства для управления набором драйверов баз данных. Методу *getConnection()* необходимо передать тип и физическое месторасположение базы данных, а также логин и пароль для доступа. С помощью метода *registerDriver()* драйверы регистрируются, а методом *getDrivers()* можно получить список всех драйверов.

1. Создание объекта для передачи запросов

*Statement st=cn.createSlatement();*

Объект класса *Statement* используется для выполнения запроса без его предварительной подготовки и команд *SQL.* Могут применяться также операторы для выполнения подготовленных запросов и хранимых процедур *РreparedStatement* и *CallableStatement.* Созданный объект можно использовать для выполнения запроса.

1. Выполнение запроса

Результаты выполнения запроса помещаются в объект класса *ResultSet:*

*ResutSet rs=st.executeQuery(“SELECT \* FROM my table”);*

Для добавления или изменения информации в таблице вместо метода *ехecuteQuery()* запрос помещается в метод *executeUpdate().*

1. Обработка результатов выполнения запроса производится методами интерфейса *ResultSet,* где самыми распространенными являются *next()* и *getString(),* а также методы *getDate() getlnt(), getShort(), getBytes(), getFloat(), getTime(). getDouble(), getLong(), getClob(), getBlob().* Среди них следует выделить методы *getClob()* и *getBlob(),* позволяющие извлекать из полей таблицы специфические объекты *(Character Large Object, Binary Large Object),* которые могут быть, например, графическими или архивными файлами. Эффективным способом извлечения значения поля из таблицы является обращение к этому полю по его позиции в строке.

При первом вызове метода *next()* указатель перемещается на таблицу результатов выборки в позицию первой строки таблицы ответа. Когда строки закончатся, метод возвратит *false.*

1. Закрытие соединения

*cn.close();*

Если база больше не нужна, соединение закрывается.

Следующий пример демонстрирует консольное приложение для работы с базой данных. Создается консольное приложение, которое связывается с базой данных, реализованной в *Microsoft Access.* В базе данных в одной табличке хранится информация о кондитерском ассортименте. Программа выводит информацию, хранимую в таблице, добавляет записи, удаляет и редактирует их.

Пример 4.1

Листинг DBClass.java

import java.awt.\*; // импортирование пакета awt

import java.net.\*; // импортирование пакета для работы в сети

import java.sql.\*; // импортирование пакета sql

import java.io.\*; // импортирование пакета ввода-вывода

public class DBClass{ // объявление класса DBClass

static void menu(){// объявление метода menu ()

// вывод в консоль пунктов меню

System.out.println("Vyberite punkt menu:");

System.out.println(" 1 -Prosmotr assortimenta");

System.out.println("2-Vstavka");

System.out.println("3-Udalenie");

System.out.println("4-Redaktirovanie’’);

System.out.println("5-Vyhod");

}

public static void main(String args[]){// объявление метода main ()

// объявление переменных

String nazv, cena, naim; int massa;

// создание буферизированного символьного потока ввода

BufferedReader stdin = new BufferedReader(new

InputStreamReader(System.in));

BufferedReader in=new BufferedReader(new

InputStreamReader(System.in));

String url="jdbc:odbc:Konditerskaya";

// ссылка на драйвер базы данных

try{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");// подключение //интерфейса JAVА-ODBC, без которого JAVA не сможет связаться //с установленным ODBC-драйвером.

Connection db=DriverManager.getConnection(url); // установление //соединения с базой данных

while(true) {//бесконечный цикл

menu();//вывод пунктов меню

Statement sq=db.createStatement(),//создание объекта

//для передачи запросов

String vybor = stdin.readLine();

// чтение строки, вводимой с клавиатуры if(vybor.equalsIgnoreCase("1")){// выполнение, если введено «1»

// формирование строки запроса

String sq\_str="SELECT \* FROM Assortiment";

ResultSet rs= sq.executcQuery(sq\_str);

// команда на выполнение запроса

System.out.println("|Naimenovanie\t|Nazvanie\t|Massa\t|Cena|"); while(rs.next()){// начало цикла для доступа к найденным записям nazv=rs.getString("Nazvanie");

// возвращает содержимое поля Nazvanie cena=rs.getString("Cena");// возвращает содержимое поля Cena

// возвращает содержимое поля Naimenovanie

naim=rs.getString( "Naimenovanie");

// возвращает содержимое поля Massa

massa=rs.getShort( "Massa");

System.out.println("|"+naim+"\t|"+nazv+"\t|"+cena+"\t|"+massa+"\t|");

} }

else if (vybor.equalsIgnoreCase("2"))

{// выполнение, если введено «2»

System.out.println("Mogete dobavit’ novuyu produkciyu:");

System.out.println("Vvedite naimenovanie:");

naim=in.readLine();

// чтение наименования, вводимого с клавиатуры System.out.println("Vveditc nazvanie:");

nazv=in.readLine();// чтение названия, вводимого с клавиатуры

System.out.println("Vvedite cenu:");

cena=in.readLine();// чтение цены, вводимой с клавиатуры

System.out.println("Vvedite massu:");

// чтение массы, вводимой с клавиатуры

massa=Integer.parseInt(in.readLine());

// формирование строки запроса

String sq\_str="INSERT INTO Assortment VALUES

("'+naim+"', "' +nazv+"','"+

cena+"', "+massa+")";

int rs= sq.exceuteUpdate(sq\_str); // команда на выполнение запроса

}

else if (vybor.equalsIgnoreCase("3")){

// выполнение, если введено «3»

System.out.println("Mogetce udalit' assortment:");

System.out.println("Vvedite naimenovanie assortimenta:");

naim=in.readLine();

// чтение наименования, вводимого с клавиатуры

// формирование строки запроса

String sq\_str="DELETE FROM Assortiment WHERE Naimenovanic = '"+naim+"'";

int rs= sq.exceuteUpdate(sq\_str); // команда на выполнение запроса

}

else if (vybor.equalsIgnoreCase("4")){

// выполнение, если введено «4»

String vibor;

System.out.println("Vvedite naimenovanic assortimenta,kotoroe hotite redaktirovat");

vibor=in.readLine();

// чтение наименования, вводимого с клавиатуры

System.out.println("Vvedite novoe naimenovanie:");

naim=in.readLine();

// чтение наименования, вводимого с клавиатуры

System.out.println("Vvedite novoe nazvanie:");

nazv=in.readLine();// чтение названия, вводимого с клавиатуры

System.out.println("Vvedite novuju cenu:");

cena=in.readLine();// чтение цены, вводимой с клавиатуры

System.out.println("Vvedite novuju massu:");

// чтение массы, вводимой с клавиатуры

massa=Integer.parseInt(in.readLine());

// формирование строки запроса

String sq\_str="UPDATE Assortiment SET Naimenovanie="'+naim+

"',Nazvanie='"+nazv+"',Cena="'+cena+'",Massa="+massa+

" WHERE Naimenovanie="'+vibor+""';

int rs= sq.exceuteUpdate(sq\_str); // команда на выполнение запроса

else if (vybor.equalsIgnoreCase("5")){

// выполнение, если введено «5»

db.close( )//закрытие соединения

System.exit(0);//выход из программы

}

}

}

catch(Exception e){

System.out.println("Error"+e);

}

}

}

Для запуска приложения необходимо сначала в Microsoft Access создать таблицу, которая должна содержать следующие поля:

Описание таблицы Assortment

| Имя поля | Тип данных |
| --- | --- |
| Naimcnovanic | Текстовый |
| Nazvanie | Текстовый |
| Cena | Денежный |
| Massa | Числовой |

Далее необходимо заполнить таблицу данными, например:

Данные таблицы Assortment

| Naimcnovanic | Nazvanie | Cena | Massa |
| --- | --- | --- | --- |
| Tort | Napoleon | 50,00 p. | 2000 |
| Pirognoc | Oduvanchik | 5,00 p. | 100 |
| Rule! | Orcshck | 10,00 p. | 500 |
| Pirognoc | Zavamoc | 4,00 p. | 150 |

Файл базы данных необходимо сохранить на диске, например *d:\dbl.mdb.*

Далее следует прописать имя источника данных. Для этого необходимо выбрать

«Пуск—> Настройка—> Панель управления—> Администрирование—» Источники данных *(ODBC)».*

Появится окно *«ODBC Data Source Administrator».* В этом окне необходимо нажать кнопку *«Add».* В появившемся окне выбрать драйвер, для которого необходимо прописать имя источника данных. В данном случае необходимо выбрать *«Driver do Microsoft Access (\*.mdb)»* и нажать кнопку «Готово». В появившемся окне «Установка драйвера *ODBC* для *Microsoft Access»* в поле «Имя источника данных» прописать «Konditerskaya». Потом нажать кнопку «Выбрать» и выбрать мышью сам файл *d:\dbl.mdb.* После выполнения необходимо нажать кнопку «ОК»