Объект класса **Statement** используется для выполнения SQL-запроса без его предварительной подготовки. Могут применяться также объекты классов **PreparedStatement** и **CallableStatement** для выполнения подготовленных запросов и хранимых процедур. Созданные объекты можно использовать для выполнения запроса SQL, передавая его в один из методов **executeQuery(String sql)** или **executeUpdate(String sql)**.

4. *Выполнение запроса.*

Результаты выполнения запроса помещаются в объект **ResultSet**:

```
ResultSet rs = st.executeQuery(
"SELECT * FROM my_table");//выборка всех данных таблицы my_table
```

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

5. *Обработка результатов выполнения запроса* производится методами интерфейса **ResultSet**, где самыми распространенными являются **next()** и **getString(int pos)** а также аналогичные методы, начинающиеся с **get**Тип**(int pos)** (**getInt(int pos)**, **getFloat(int pos)** и др.) и **update**Тип**()**. Среди них следует выделить методы **getClob(int pos)** и **getBlob(int pos)**, 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 ответа 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 значение **false**.

6. *Закрытие соединения*

```
cn.close();
```

После того как база больше не нужна, соединение закрывается.

Для того чтобы правильно пользоваться приведенными методами, программисту требуется знать типы полей БД. В распределенных системах это знание предполагается изначально.

СУБД MySQL

СУБД MySQL совместима с JDBC и будет применяться для создания экспериментальных БД. Последняя версия СУБД может быть загружена с сайта **www.mysql.com**. Для корректной установки необходимо следовать инструкциям мастера установки. Каталог лучше выбирать по умолчанию. В процессе установки следует создать администратора СУБД с именем `root` и паролем `pass`. Если планируется разворачивать реально работающее приложение, необходимо исключить тривиальных пользователей сервера БД (иначе злоумышленники могут получить полный доступ к БД). Для запуска следует использовать команду из папки **/mysql/bin**:

```
mysqld-nt -standalone
```

Если не появится сообщение об ошибке, то СУБД MySQL запущена. Для создания БД и таблиц используются команды языка SQL.

Дополнительно требуется подключить библиотеку, содержащую драйвер MySQL

mysql-connector-java-3.1.12.jar,

и разместить ее в каталоге **/WEB-INF/lib** проекта.

Простое соединение и простой запрос

Теперь следует воспользоваться всеми предыдущими инструкциями и создать пользовательскую БД с именем **db2** и одной таблицей **users**. Таблица должна содержать два поля: символьное – **name** и числовое – **phone** и несколько занесенных записей. Сервлет, осуществляющий простейший запрос на выбор всей информации из таблицы, выглядит следующим образом.

```
/* пример # 1 : соединение с базой : ServletToBase.java */
package chapt20;
import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class ServletToBase extends HttpServlet {
      public void doGet(HttpServletRequest req,
                        HttpServletResponse resp)
                        throws ServletException {
            performTask(req, resp);
      }
      public void doPost(HttpServletRequest req,
                         HttpServletResponse resp)
                         throws ServletException {
            performTask(req, resp);
      }

      public void showInfo(PrintWriter out, ResultSet rs)
                                    throws SQLException {
            out.print("From DataBase:");
        while (rs.next()) {
            out.print("<br>Name:-> " + rs.getString(1)
                        + " Phone:-> " + rs.getInt(2));
            }
      }
      public void performTask(HttpServletRequest req,
                              HttpServletResponse resp) {
      resp.setContentType("text/html; charset=Cp1251");
      PrintWriter out = null;
            try {//1
                  out = resp.getWriter();
                  try {//2

      Class.forName("org.gjt.mm.mysql.Driver");
            // для MSAccess
```