JDBC 结果集

一、实验介绍

1.1 实验内容

本实验将学习java.sql.ResultSet接口，即数据库查询的结果集相关的知识。

1.2 实验知识点

- ResultSet接口

1.3 实验环境

- JDK1.7
- Eclipse
- Mysql 5.5.50

二、实验步骤

2.1 ResultSet 介绍

结果集通常是通过执行查询数据库的语句生成，表示数据库查询结果的数据表。ResultSet 对象具有指向其当前数据行的光标。最初，光标被置于第一行之前。next 方法将光标移动到下一行；因为该方法在 ResultSet 对象没有下一行时返回 false，所以可以在 while 循环中使用它来迭代结果集。光标可以方便我们对结果集进行遍历。默认的 ResultSet 对象不可更新，仅有一个向前移动的光标。因此，只能迭代它一次，并且只能按从第一行到最后一行的顺序进行。

ResultSet接口的方法可分为三类：

- 导航方法：用于移动光标
- 获取方法：用于查看当前行的光标所指向的列中的数据
- 更新方法：用于更新当前行的列中的数据

JDBC 提供下列连接方法来创建所需的ResultSet语句：

```
createStatement(int RSType, int RSConcurrency);

prepareStatement(String SQL, int RSType, int RSConcurrency);

prepareCall(String sql, int RSType, int RSConcurrency);
```

RSType 表示 ResultSet 对象的类型，RSConcurrency 是 ResultSet 常量，用于指定一个结果集是否为只读或可更新。

ResultSet 的类型，如果不指定 ResultSet 类型，将自动获得一个是 TYPE_FORWARD_ONLY：

类型	**描述**
ResultSet.TYPE_FORWARD_ONLY	游标只能向前移动的结果集
ResultSet.TYPE_SCROLL_INSENSITIVE	游标可以向前和向后滚动，但不及时更新， 就是如果数据库里的数据修改过， 并不在ResultSet中反应出来
ResultSet.TYPE_SCROLL_SENSITIVE	游标可以向前和向后滚动， 并及时跟踪数据库的更新， 以便更改ResultSet中的数据

并发性的ResultSet，如果不指定任何并发类型，将自动获得一个为 CONCUR_READ_ONLY

并发	**描述**
ResultSet.CONCUR_READ_ONLY	创建结果集只读。这是默认的
ResultSet.CONCUR_UPDATABLE	创建一个可更新的结果集

如初始化一个 Statement 对象来创建一个双向、可更新的ResultSet对象：

```
try {
   Statement stmt = conn.createStatement(
                            ResultSet.TYPE_SCROLL_INSENSITIVE,
                            ResultSet.CONCUR_UPDATABLE);
}
catch(Exception ex) {
   ....
}
finally {
   ....
}
```

2.2 导航

我们在上面已经知道了，导航方法是用于移动光标的。我们先来看一看，在 ResultSet 接口中有哪些方法会涉及光标的移动。

方法	说明
public void beforeFirst() throws SQLException	将光标移动到正好位于第一行之前
public void afterLast() throws SQLException	将光标移动到刚刚结束的最后一行
public boolean first() throws SQLException	将光标移动到第一行
public void last() throws SQLException	将光标移动到最后一行
public boolean absolute(int row) throws SQLException	将光标移动到指定的行
public boolean relative(int row) throws SQLException	从它目前所指向向前或向后移动光标行的给定数量
public boolean previous() throws SQLException	将光标移动到上一行。 上一行关闭的结果集此方法返回false
public boolean next() throws SQLException	将光标移动到下一行。 如果没有更多的行结果集中的此方法返回false
public int getRow() throws SQLException	返回的行号，该光标指向的行
public void moveToInsertRow() throws SQLException	将光标移动到一个特殊的行， 可以用来插入新行插入到数据库中的结果集。 当前光标位置被记住
public void moveToCurrentRow() throws SQLException	移动光标返回到当前行，如果光标在当前插入行，否则， 这个方法不执行任何操作

什么也不说了，我们还是看代码吧！

```
package com.shiyanlou;

import java.sql.*;

public class JdbcTest {
   // JDBC 驱动器名称 和数据库地址
   static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
   //数据库的名称为 EXAMPLE
   static final String DB_URL = "jdbc:mysql://localhost/EXAMPLE";

   // 数据库用户和密码
   static final String USER = "root";
   static final String PASS = "";

   public static void main(String[] args) {
       Connection conn = null;
       Statement stmt = null;
       try{
           //注册JDBC 驱动程序
           Class.forName("com.mysql.jdbc.Driver");

           //打开连接
           System.out.println("Connecting to database...");
           conn = DriverManager.getConnection(DB_URL,USER,PASS);



           System.out.println("Creating statement...");
           //创建所需的ResultSet，双向，只读
           stmt = conn.createStatement(
                          ResultSet.TYPE_SCROLL_INSENSITIVE,
                          ResultSet.CONCUR_READ_ONLY);
           String sql;
           sql = "SELECT id, name, age FROM Students";
           ResultSet rs = stmt.executeQuery(sql);

           // 将光标移到最后一行
           System.out.println("Moving cursor to the last...");
           rs.last();

           //处理结果集
           System.out.println("Displaying record...");
           int id  = rs.getInt("id");
           int age = rs.getInt("age");
           String name = rs.getString("name");


           //显示
           System.out.print("ID: " + id);
           System.out.print(", Age: " + age);
           System.out.print(", Name: " + name);
           System.out.println();

           // 将光标移到第一行
           System.out.println("Moving cursor to the first row...");
           rs.first();
```

```
            System.out.println("Displaying record...");
            id  = rs.getInt("id");
            age = rs.getInt("age");
            name = rs.getString("name");


            //显示
            System.out.print("ID: " + id);
            System.out.print(", Age: " + age);
            System.out.print(", Name: " + name);

            //将光标移至下一行
            System.out.println("Moving cursor to the next row...");
            rs.next();


            System.out.println("Displaying record...");
            id  = rs.getInt("id");
            age = rs.getInt("age");
            name = rs.getString("name");

            // 显示
            System.out.print("ID: " + id);
            System.out.print(", Age: " + age);
            System.out.print(", Name: " + name);

            rs.close();
            stmt.close();
            conn.close();
        }catch(SQLException se){
            se.printStackTrace();
        }catch(Exception e){
            e.printStackTrace();
        }finally{
            try{
                if(stmt!=null)
                    stmt.close();
            }catch(SQLException se2){
            }
            try{
                if(conn!=null)
                    conn.close();
            }catch(SQLException se){
                se.printStackTrace();
            }
        }
        System.out.println("Goodbye!");
    }
}
```

JDBC 入门教程 (/courses/110)

运行结果：

```
Connecting to database...
Creating statement...
Moving cursor to the last...
Displaying record...
ID: 4, Age: 20, Name: Tomson
Moving cursor to the first row...
Displaying record...
ID: 1, Age: 22, Name: TomMoving cursor to the next row...
Displaying record...
ID: 2, Age: 20, Name: AbyGoodbye!
```


2.3 获取

ResultSet接口中我们经常使用 get 方法来查看结果集。

方法	说明
public int getInt(String columnName) throws SQLException	当前行中名为 ColumnName 列的值
public int getInt(int columnIndex) throws SQLException	当前行中指定列的索引的值。列索引从1开始， 意味着一个行的第一列是1，行的第二列是2，依此类推

当然还有 getString()等等。

代码示例参看上面的示例。

2.4 更新

更新的方法如下：

方法	说明
public void updateString(int columnIndex, String s) throws SQLException	指定列中的字符串更改为s的值
public void updateString(String columnName, String s) throws SQLException	类似于前面的方法， 不同之处在于由它的名称， 而不是它的索引指定的列

类似的还有 updateDouble()等等。

我们在更新了结果集中的内容，当然需要更新一下数据库了。我们可以调用下面的方法更新数据库。

方法	说明
public void updateRow()	通过更新数据库中相应的行更新当前行
public void deleteRow()	从数据库中删除当前行
public void refreshRow()	刷新在结果集的数据，以反映最新变化在数据库中
public void cancelRowUpdates()	取消所做的当前行的任何更新
public void insertRow()	插入一行到数据库中。 当光标指向插入行此方法只能被调用

具体的例子我们便留在作业里吧，我们这里对上面的方法做一个小小的举例：

```
Statement stmt = conn.createStatement(
                           ResultSet.TYPE_SCROLL_INSENSITIVE,
                           ResultSet.CONCUR_UPDATABLE);

String sql = "SELECT id, name, age FROM Students";
ResultSet rs = stmt.executeQuery(sql);

//结果集中插入新行
rs.moveToInsertRow();
rs.updateInt("id",5);
rs.updateString("name","John");
rs.updateInt("age",21);
//更新数据库
rs.insertRow();
```

三、实验总结

本次课程介绍了 JDBC 结果集的相关知识，下一节我们将进入 JDBC 数据类型与事务的学习。

四、课后习题

JDBC 入门教程 (/courses/110)

写完上面的更新验证代码吧！

上一节：JDBC 接口 (/courses/110/labs/1194/document)

下一节：JDBC 数据类型与事务 (/courses/110/labs/1200/document)

JDBC 入门教程 (/courses/110)

JDBC 入门教程 (/courses/110)