# JDBC

JDBC流程：

第一步：加载Driver类，注册数据库驱动；

第二步：通过DriverManager,使用url，用户名和密码建立连接(Connection)；

第三步：通过Connection，使用sql语句打开Statement对象；

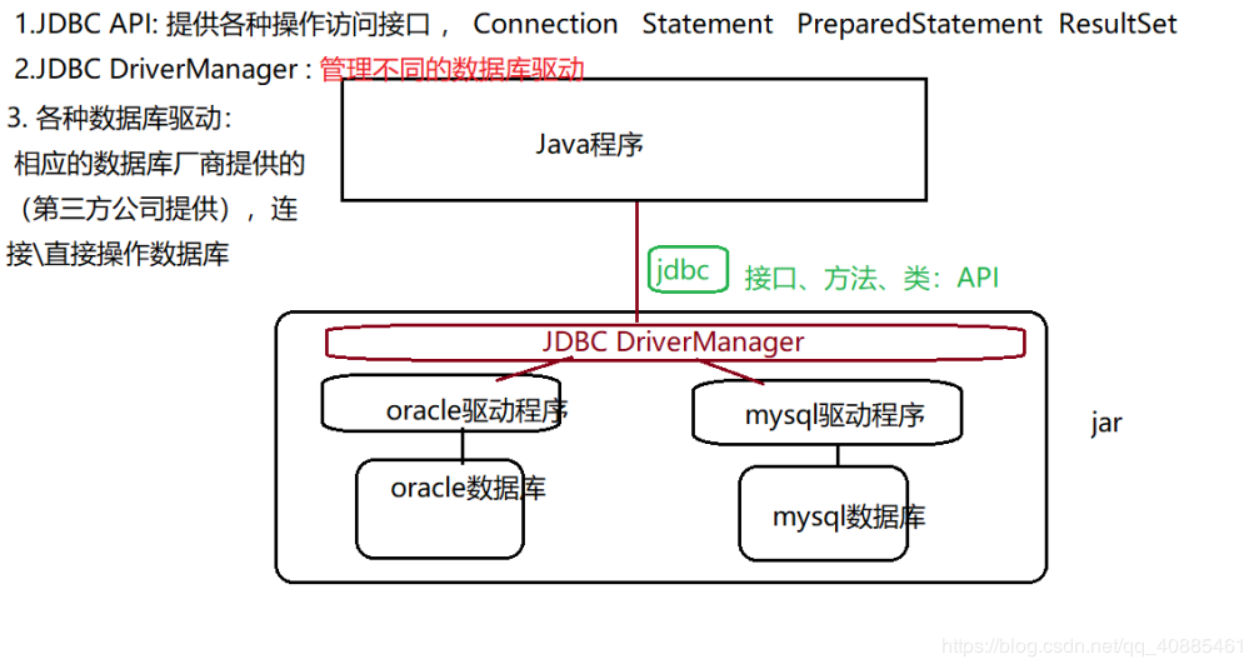
第四步：执行语句，将结果返回resultSet；

第五步：对结果resultSet进行处理；

第六步：倒序释放资源resultSet-》preparedStatement-》connection。

参考资料：<https://www.jianshu.com/p/b2525edd9ec0>

Jdbc操作数据库的实例：<https://blog.csdn.net/wcc27857285/article/details/82287205>



## 原生jdbc操作数据库

* 导入JDBC包：使用Java语言的import语句在Java代码开头位置导入所需的类。
* 注册JDBC驱动程序：使JVM将所需的驱动程序实现加载到内存中，从而可以满足JDBC请求。

方法I - Class.forName()

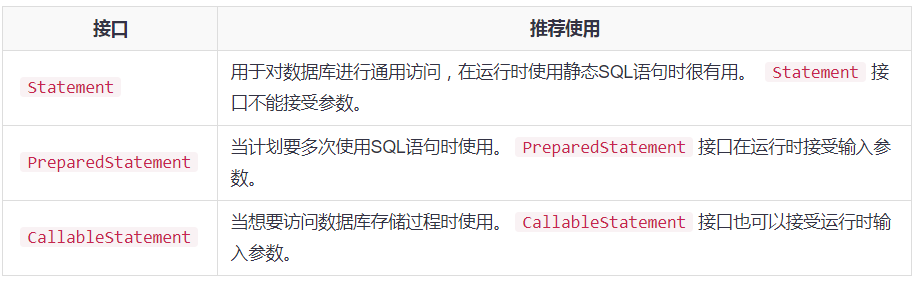
注册驱动程序最常见的方法是使用Java的Class.forName()方法，将驱动程序的类文件动态加载到内存中，并将其自动注册。这个方法是推荐使用的方法，因为它使驱动程序注册可配置和便携。

方法II - DriverManager.registerDriver()

如果使用的是非JDK兼容的JVM(如Microsoft提供的)，则应使用registerDriver()方法

* 数据库URL配置：创建一个正确格式化的地址，指向要连接到的数据库(如：MySQL,Oracle和MSSQL等等)。
* 创建连接对象：最后，调用DriverManager对象的getConnection()方法来建立实际的数据库连接。

## JDBC Statements, PreparedStatement和CallableStatement语句



### Statement对象

* 创建Statement对象  
  在使用Statement对象执行SQL语句之前，需要使用Connection对象的createStatement()方法创建一个Statement对象  
  在创建Statement对象后，可以使用它来执行一个SQL语句，它有三个执行方法可以执行。它们分别是 ：
  + boolean execute (String SQL) ： 如果可以检索到ResultSet对象，则返回一个布尔值true; 否则返回false。使用此方法执行SQLDDL语句或需要使用真正的动态SQL，可使用于执行创建数据库，创建表的SQL语句等等。
  + int executeUpdate (String SQL): 返回受SQL语句执行影响的行数。使用此方法执行预期会影响多行的SQL语句，例如:INSERT，UPDATE或DELETE语句。
  + ResultSet executeQuery(String SQL)：返回一个ResultSet对象。 当您希望获得结果集时，请使用此方法，就像使用SELECT语句一样。

### PreparedStatement对象

PreparedStatement接口扩展了Statement接口，它添加了比Statement对象更好一些优点的功能。此语句可以动态地提供/接受参数。

* 创建PreparedStatement对象  
  JDBC中的所有参数都由 ? 符号作为占位符，这被称为参数标记。 在执行SQL语句之前，必须为每个参数(占位符)提供值。  
  setXXX()方法将值绑定到参数，其中XXX表示要绑定到输入参数的值的Java数据类型。 如果忘记提供绑定值，则将会抛出一个SQLException。  
  每个参数标记是它其顺序位置引用。第一个标记表示位置1，下一个位置2等等。 该方法与Java数组索引不同(它不从0开始)。  
  所有Statement对象与数据库交互的方法(a)execute()，(b)executeQuery()和(c)executeUpdate()也可以用于PreparedStatement对象。 但是，这些方法被修改为可以使用输入参数的SQL语句。
* 关闭PreparedStatement对象

### CallableStatement对象

类似Connection对象创建Statement和PreparedStatement对象一样，它还可以使用同样的方式创建CallableStatement对象，该对象将用于执行对数据库存储过程的调用。

## JDBC结果集ResultSet

SQL语句执行后从数据库查询读取数据，返回的数据放在结果集中。 SELECT语句用于从数据库中选择行并在结果集中查看它们的标准方法。 java.sql.ResultSet接口表示数据库查询的结果集。

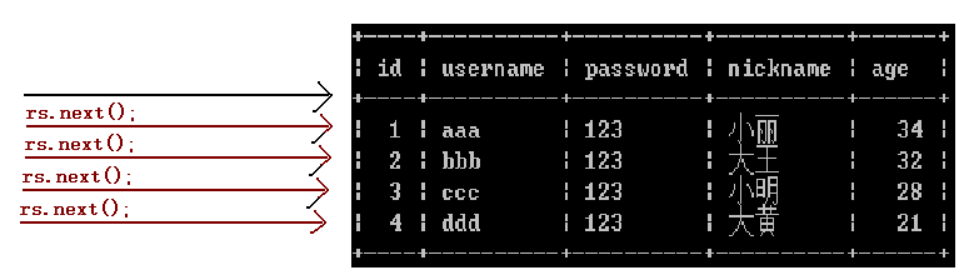
ResultSet对象维护指向结果集中当前行的游标。 术语“结果集”是指包含在ResultSet对象中的行和列数据。

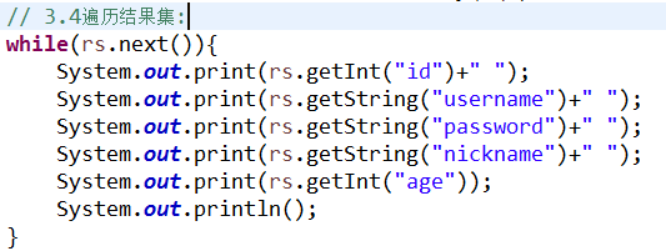
ResultSet接口的方法可以分为三类：

* 浏览方法：用于移动光标。
* 获取方法：用于查看光标指向的当前行的列中的数据。
* 更新方法：用于更新当前行的列中的数据。 然后在基础数据库中更新数据。  
  光标可以基于ResultSet的属性移动。当创建生成ResultSet的相应Statement时，将指定这些属性。

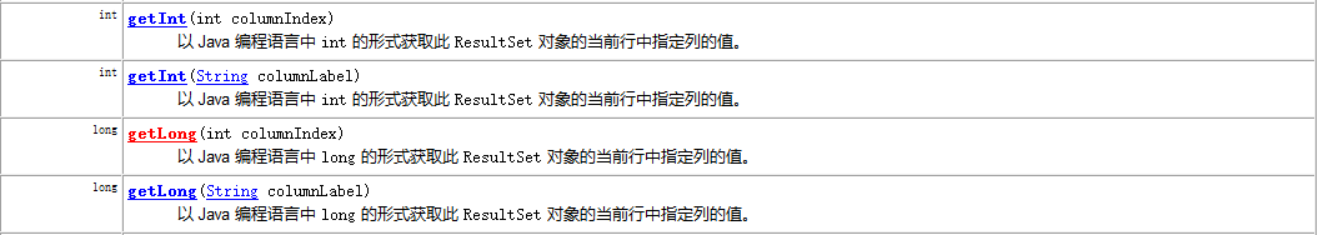
JDBC提供以下连接方法来创建具有所需ResultSet的语句

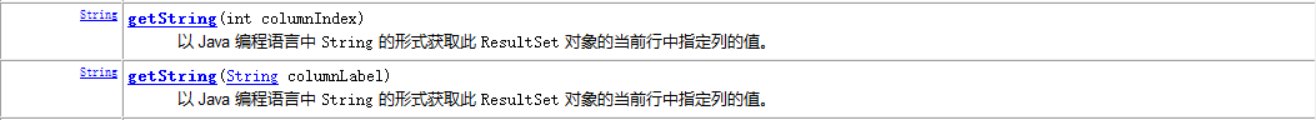
* createStatement(int RSType, int RSConcurrency);
* prepareStatement(String SQL, int RSType, int RSConcurrency);
* prepareCall(String sql, int RSType, int RSConcurrency);  
  第一个参数表示ResultSet对象的类型，第二个参数是两个ResultSet常量之一，用于指定结果集是只读还是可更新。





结果集的获取：





*/\*\*  
 \* 我算是明白了，ResultSet执行next()方法后，就是会指向当前一行的数据，这一行的数据可以通过getInt(index[列索引])  
 \* getString(index[列索引])来获取相应列的值，具体用getInt、getString()、或者getFloat()等等，是根据返回来的数据集中的数据类型来决定的，  
 \* 比如：数据集中明明是String类型的，此时不想通过getString("列名")来获取，还是想通过列索引index来获取的话，只能写getString(index),而不能  
 \* 继续再写getInt(index);  
 \*/*

查询结果封装为对象：<https://blog.csdn.net/qq_23569917/article/details/96439840>

## JDBC事务

如果JDBC连接处于自动提交模式，默认情况下，则每个SQL语句在完成后都会提交到数据库。

对于简单的应用程序可能没有问题，但是有三个原因需要考虑是否关闭自动提交并管理自己的事务 :

* 提高性能
* 保持业务流程的完整性
* 使用分布式事务  
  事务能够控制何时更改提交并应用于数据库。 它将单个SQL语句或一组SQL语句视为一个逻辑单元，如果任何语句失败，整个事务将失败。

要启用手动事务支持，而不是使用JDBC驱动程序默认使用的自动提交模式，请调用Connection对象的setAutoCommit()方法。 如果将布尔的false传递给setAutoCommit()，则关闭自动提交。 也可以传递一个布尔值true来重新打开它。

例如，如果有一个名为conn的Connection对象，请将以下代码关闭自动提交 -

conn.setAutoCommit(false);

## JDBC批量处理

批量处理允许将相关的SQL语句分组到批处理中，并通过对数据库的一次调用来提交它们，一次执行完成与数据库之间的交互。一次向数据库发送多个SQL语句时，可以减少通信开销，从而提高性能。不需要JDBC驱动程序来支持此功能。应该使用DatabaseMetaData.supportsBatchUpdates()  
方法来确定目标数据库是否支持批量更新处理。如果JDBC驱动程序支持此功能，该方法将返回true。

Statement，PreparedStatement和CallableStatement的addBatch()方法用于将单个语句添加到批处理。 executeBatch()用于执行组成批量的所有语句。  
executeBatch()返回一个整数数组，数组的每个元素表示相应更新语句的更新计数。就像将批处理语句添加到处理中一样，可以使用clearBatch()方法删除它们。此方法将删除所有使用addBatch()方法添加的语句。 但是，无法指定选择某个要删除的语句。

### 使用Statement对象进行批处理

以下是使用Statement对象的批处理的典型步骤序列:

* 使用createStatement()方法创建Statement对象。
* 使用setAutoCommit()将自动提交设置为false。
* 使用addBatch()方法在创建的Statement对象上添加SQL语句到批处理中。
* 在创建的Statement对象上使用executeBatch()方法执行所有SQL语句。
* 最后，使用commit()方法提交所有更改。

# PrepareStatement对象

.setString()方法

setString是定义了字符串中第n个”?“字符的替换。

举例：

string strsql = "select emp\_id from employee where emp\_id = ?";

preparedstatement pstmt = conn.preparestatement(strsql);  
pstmt.setstring(1,"pma42628m");

解释：此时sql中的"?"的值就是”pma42628m“。