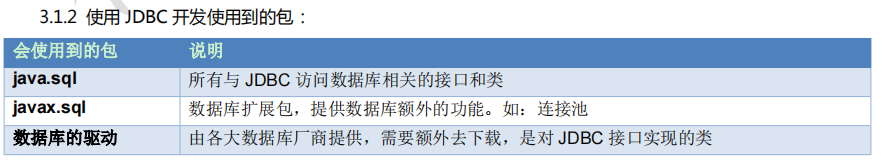
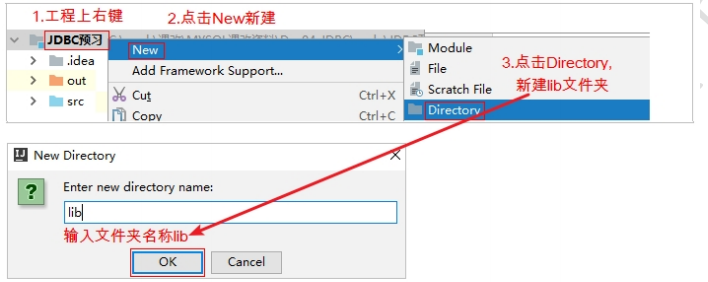
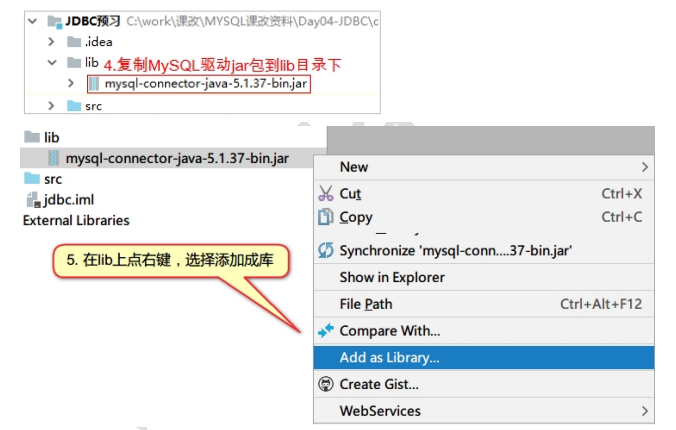
# JDBC jar包

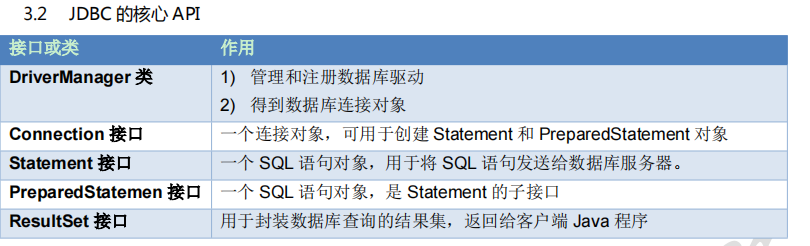


## 驱动jar包的导入





# JDBC核心API



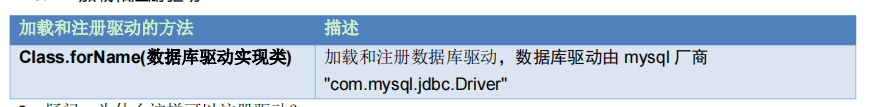
## (1)DriverManager类

DriverManager 作用：

管理和注册驱动

创建数据库的连接

### 加载和注册驱动



疑问：为什么这样可以注册驱动？

|  |
| --- |
| **public class** Demo1 {  **public static void** main(String[] args) **throws** ClassNotFoundException {**4** / **21**  *//抛出类找不到的异常，注册数据库驱动*  Class.*forName*(**"com.mysql.jdbc.Driver"**);  }  } |

 com.mysql.jdbc.Driver 源代码：

// Driver 接口，所有数据库厂商必须实现的接口，表示这是一个驱动类。

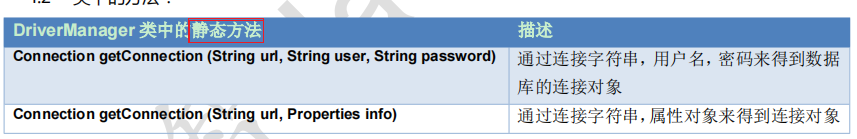
|  |
| --- |
| **public class** Driver **implements** java.sql.Driver {  **public** Driver() **throws** SQLException {  }  **static** {  **try** {  DriverManager.registerDriver(**new** Driver()); //注册数据库驱动  } **catch** (SQLException var1) {  **throw new** RuntimeException(**"Can't register driver!"**);  }  }  } |

说明：Class.forName() 加载某个类会触发其初始化，初始化会执行类静态代码块。

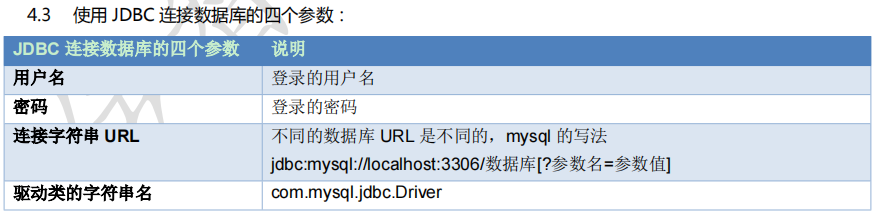
注：从 JDBC3 开始，目前已经普遍使用的版本。可以不用注册驱动而直接使用。Class.forName 这句话可以省略

### 获取数据库连接对象

1. DriverManager的静态方法：

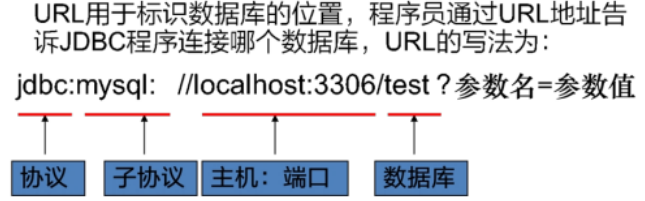


1. 参数说明：



* 连接数据库的 URL 地址格式：

协议名:子协议://服务器名或 IP 地址:端口号/数据库名?参数=参数值



* MySQL 中可以简写：

前提：必须是本地服务器，端口号是 3306

jdbc:mysql:///数据库名

1. 连接Mysql的乱码的处理

如果数据库出现乱码，可以指定参数: ?characterEncoding=utf8，表示让数据库以 UTF-8 编码来处理数据。

|  |
| --- |
| jdbc:mysql://localhost:3306/数据库?characterEncoding=utf8 |

### 案例：得到 MySQL 的数据库连接对象

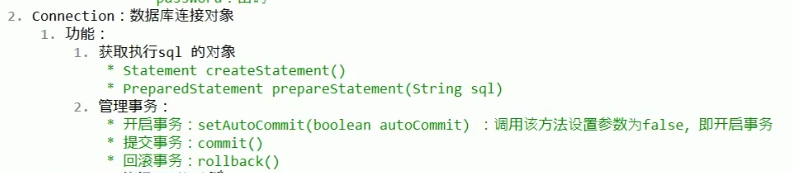
#### 1) 使用用户名、密码、URL 得到连接对象

|  |
| --- |
| **package** com.itheima;  **import** java.sql.Connection;  **import** java.sql.DriverManager;  **import** java.sql.SQLException;  */\*\**  *\* 得到连接对象*  *\*/*  **public class** Demo2 {  **public static void** main(String[] args) **throws** SQLException {  String url = **"jdbc:mysql://localhost:3306/day24"**;  *//1) 使用用户名、密码、URL 得到连接对象*  Connection connection = DriverManager.*getConnection*(url, **"root"**, **"root"**);  *//com.mysql.jdbc.JDBC4Connection@68de145*  System.***out***.println(connection);  }  } |

#### **2) 使用属性文件和 url 得到连接对象**

|  |
| --- |
| package com.itheima;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.util.Properties;  public class Demo3 {  public static void main(String[] args) throws SQLException {  //url 连接字符串  String url = "jdbc:mysql://localhost:3306/day24";  //属性对象  Properties info = new Properties();  //把用户名和密码放在 info 对象中6 / 21  info.setProperty("user","root");  info.setProperty("password","root");  Connection connection = DriverManager.getConnection(url, info);  //com.mysql.jdbc.JDBC4Connection@68de145  System.out.println(connection);  }  } |

## (2)Connection类



## (3)Statement类

* Statement 作用

代表一条语句对象，用于发送 SQL 语句给服务器，用于执行静态 SQL 语句并返回它所生成结果的对象。

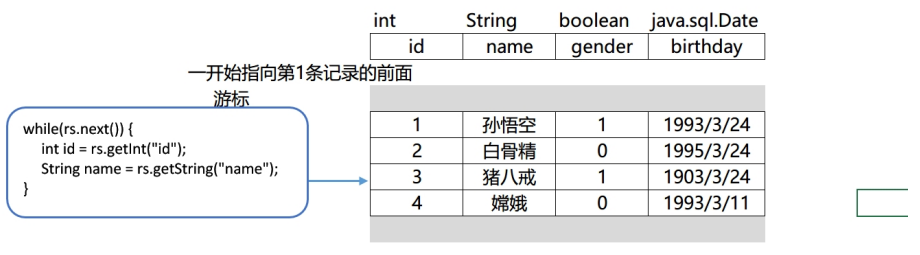
* Statement 中的方法

executeUpdate(String sql) 还可以执行DDL,但是不常用;

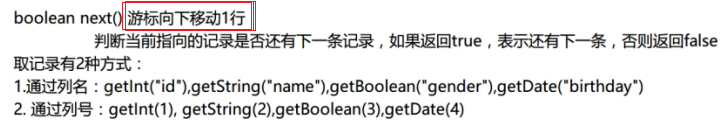
返回值>0表示执行成功，反之失败。

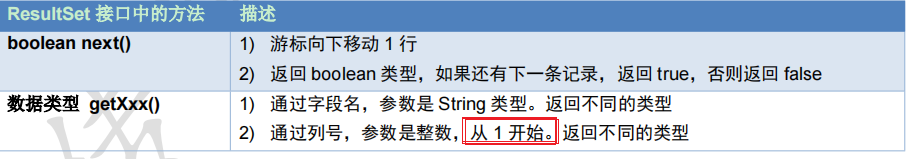
## ResultSet类

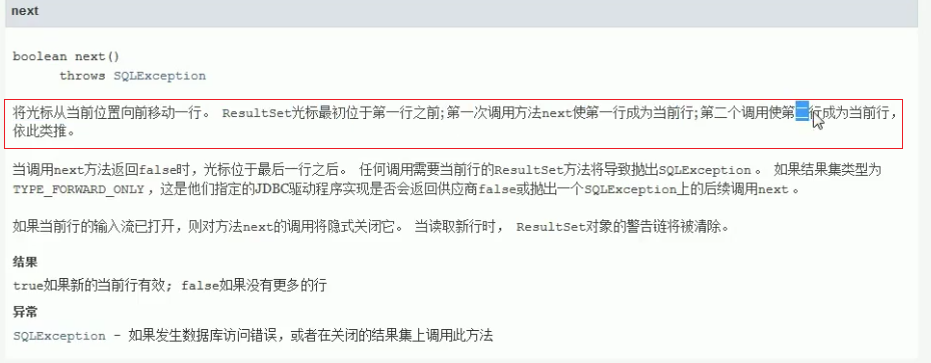
作用：封装数据库查询的结果集，对结果集进行遍历，取出每一条记录



### 核心方法







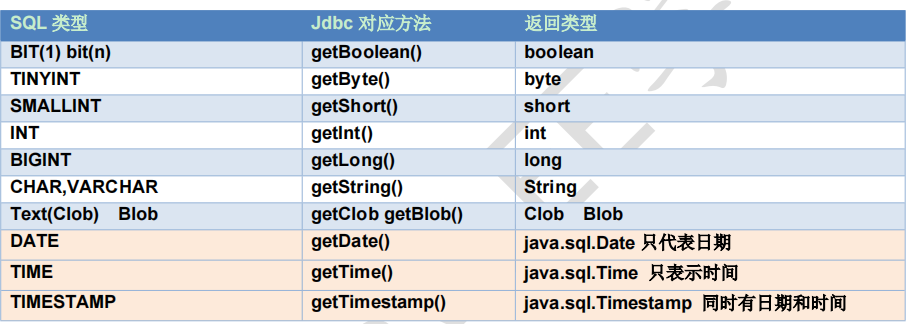
关于 ResultSet 接口中的注意事项：

1) 如果光标在第一行之前，使用 rs.getXX()获取列值，报错：Before start of result set

2) 如果光标在最后一行之后，使用 rs.getXX()获取列值，报错：After end of result set

3) 使用完毕以后要关闭结果集 ResultSet，再关闭 Statement，再关闭 Connection

### 常用数据类型转换表



java.sql.Date、Time、Timestamp(时间戳)，三个共同父类是：java.util.Date

|  |
| --- |
| public class Demo6DQL {  public static void main(String[] args) throws SQLException {  //1) 得到连接对象  Connection connection =  DriverManager.getConnection("jdbc:mysql://localhost:3306/day24","root","root");  //2) 得到语句对象  Statement statement = connection.createStatement();  //3) 执行 SQL 语句得到结果集 ResultSet 对象  ResultSet rs = statement.executeQuery("select \* from student");  //4) 循环遍历取出每一条记录  while(rs.next()) {  int id = rs.getInt("id");  String name = rs.getString("name");  boolean gender = rs.getBoolean("gender");  Date birthday = rs.getDate("birthday");  //5) 输出的控制台上  System.out.println("编号：" + id + ", 姓名：" + name + ", 性别：" + gender + ", 生日：" +  birthday);  }  //6) 释放资源  rs.close();  statement.close();  connection.close();  }  } |

# 数据库工具类 JdbcUtils

什么时候自己创建工具类？

如果一个功能经常要用到，我们建议把这个功能做成一个工具类，可以在不同的地方重用

需求：

上面写的代码中出现了很多重复的代码，可以把这些公共代码抽取出来

创建类 JdbcUtil 包含 3 个方法：

1) 可以把几个字符串定义成常量：用户名，密码，URL，驱动类

2) 得到数据库的连接：getConnection()

3) 关闭所有打开的资源：

close(Connection conn, Statement stmt)，close(Connection conn, Statement stmt, ResultSet rs)