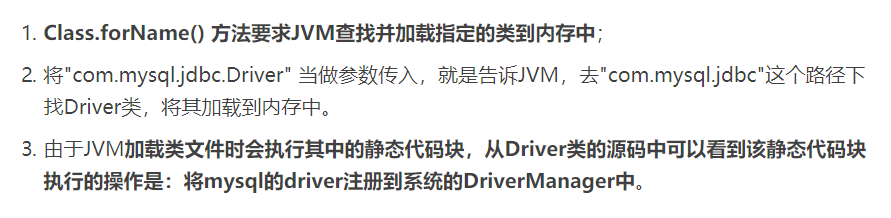
使用JDBC的步骤

加载驱动 → 建立数据库连接Connection → 创建和执行SQL的语句 → 处理执行结果ResultSet → 释放资源

* Driver(驱动程序)接口

Driver接口由数据库厂家提供，只需要使用Driver接口就可以了

在编程中要连接数据库，必须先装载特定厂商的数据库驱动程序，不同的数据库有不同的装载方法



如：加载MySql驱动：1. Class.forName("com.mysql.jdbc.Driver");(class类的静态方法)

* Connection接口

Connection与特定数据库连接（一个Connection代表一个数据库连接）

连接MySql数据库：

2. Connection conn =

DriverManager.getConnection("jdbc:mysql://host:port/database", "user", "password");

常用方法：

createStatement()：创建向数据库发送sql的statement对象。

prepareStatement(sql) ：创建向数据库发送预编译sql的PrepareSatement对象

3. PreparedStatement ps = con.prepareStatement("select \* from emp where sal between ? and ?");

* Statement接口

用于执行静态SQL语句并返回它所生成结果的对象

三种Statement类：

Statement：只能发送简单的SQL语句（不带参数）

PreparedStatement ：

1. 可以发送含有一个或多个参数的SQL语句。
2. 数据库可以将编译优化后的SQL语句缓存起来, PreparedStatement的效率更高
3. 代表预编译的语句，可以防止SQL注入

CallableStatement：继承自PreparedStatement接口，由方法prepareCall创建，用于调用存储过程

存储过程是一组为了完成特定功能的SQL语句集，经编译后存储在数据库中，用户通过指定存储过程的名字并给定参数（如果该存储过程带有参数）来调用执行它

* ResultSet接口

ResultSet提供检索不同类型字段的方法

getString(int index)、getString(String columnName)：获得在数据库里是varchar、char等类型的数据对象

4. ResultSet rs = ps.executeQuery();

5. while(rs.next()) {

System.out.println(rs.getInt("empno") + " - " + rs.getString("ename"));

}

6. finally {

if(con != null) {

try {

~~

con.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

关闭外部资源的顺序应该和打开的顺序相反，也就是说先关闭ResultSet、再关闭Statement、在关闭Connection

预编译原理：

在数据库完成 SQL 指令的编译后，才套用参数运行，因此就算参数中含有恶意的指令，由于已经编译完成，就不会被数据库所运行

数据库不会将参数的内容视为SQL执行的一部分，而是作为一个字段的属性值来处理，这样就算参数中包含破环性语句（or ‘1=1’）也不会被执行