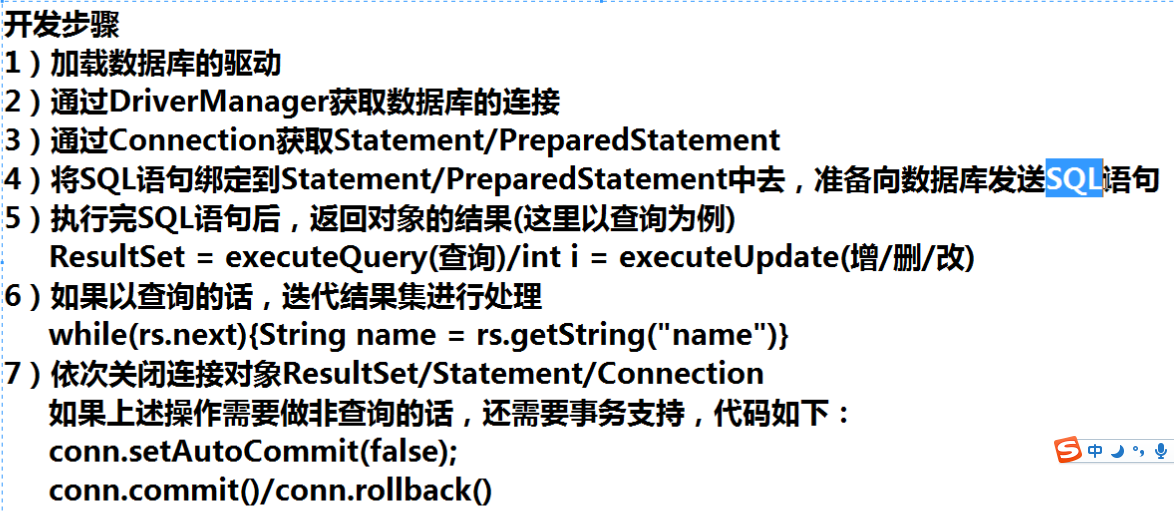
JDBC的分析及问题总结

DAO: Data Access Object

## JDBC的开发方式



## 原生态的jdbc代码

**public static void** main(String[] args) {

Connection connection = **null**;

PreparedStatement preparedStatement = **null**;

ResultSet resultSet = **null**;

**try** {

//1、加载数据库驱动

Class.*forName*("com.mysql.jdbc.Driver");

//2、通过驱动管理类获取数据库链接

connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/mybatis?characterEncoding=utf-8", "root", "root");

//3、定义sql语句 ?表示占位符

String sql = "select \* from user where username = ?";

//4、获取预处理statement

preparedStatement = connection.prepareStatement(sql);

//5、设置参数，第一个参数为sql语句中参数的序号（从1开始），第二个参数为设置的参数值

preparedStatement.setString(1, "王五");

//6、向数据库发出sql执行查询，查询出结果集

resultSet = preparedStatement.executeQuery();

//7、遍历查询结果集

**while**(resultSet.next()){

User user

System.*out*.println(resultSet.getString("id")+" "+resultSet.getString("username"));

}

} **catch** (Exception e) {

e.printStackTrace();

}**finally**{

//8、释放资源

**if**(resultSet!=**null**){

**try** {

resultSet.close();

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**if**(preparedStatement!=**null**){

**try** {

preparedStatement.close();

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**if**(connection!=**null**){

**try** {

connection.close();

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

## 问题总结

1. 在创建连接时，存在硬编码

配置文件（全局配置文件）

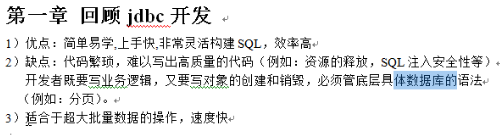
1. 在执行statement时存在硬编码

**配置文件（映射文件）**

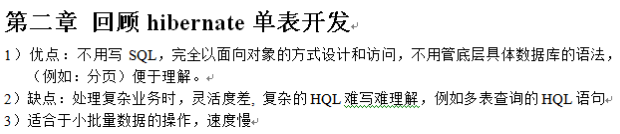
1. 频繁的开启和关闭数据库连接，会造成数据库性能下降。

**数据库连接池（全局配置文件）**

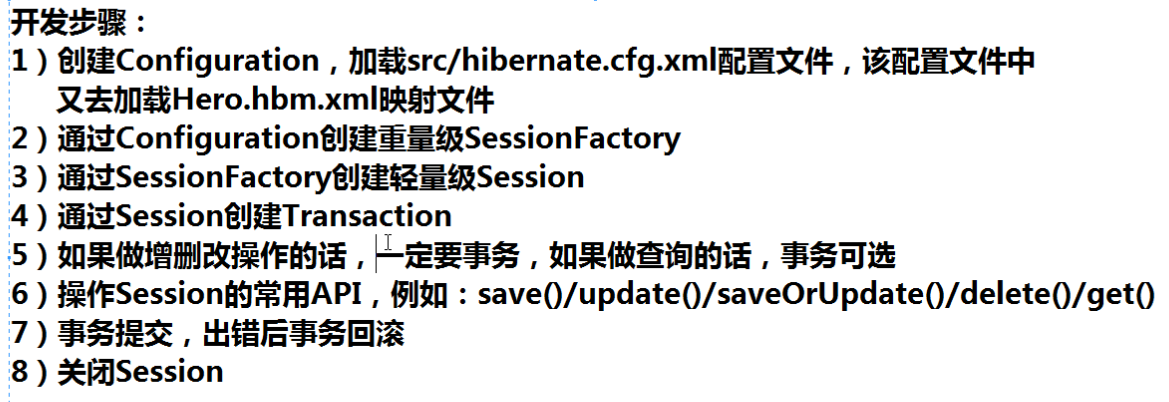
## JDBC开发的优点与缺点：



## Hibernate开发



**Hibernate的开发步骤**：



## MyBatis特点



**JDBC/dbutils/springdao、Hibernate、springorm、MyBatis**都是ORM的解决方案。