面向对象编程出现多年，对象化在应用程序端已经得到很好的实践，但在流行的关系数据库端，数据的并不是以对象方式组织。应用程序为了使用数据库的数据，不得不将从数据库得到的数据转为对象模式。ORM框架的出现，省去了转换数据这些繁重的工作，抹平了应用程序和数据库之间使用数据方式的差异。

MyBatis 是一款优秀的持久层框架，它支持定制化 SQL、存储过程以及高级映射。MyBatis 避免了几乎所有的 JDBC 代码和手动设置参数以及获取结果集。MyBatis 可以使用简单的 XML 或注解来配置和映射原生类型、接口和 Java 的 POJO为数据库中的记录。

工具



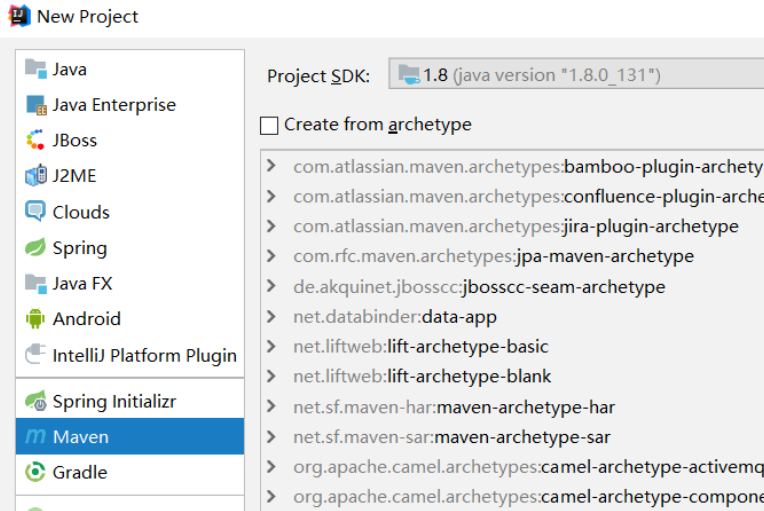
IntelliJ IDEA 2018.1.3 x64

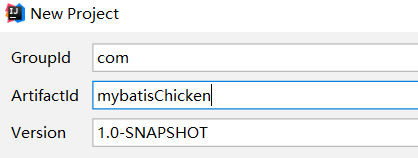


MySQL Workbench 8.0 CE



首先从一个maven项目开始，新建一个maven项目





打开maven配置文件pom.xml，添加以下依赖项

Mybatis

mysql-connector-java

log4j

junit

准备数据库

添加名为mybatis的数据库

CREATE DATABASE mybatis DEFAULT CHARACTER SET utf8 COLLATE utf8\_general\_ci;

添加table

CREATE TABLE `country` (  
 `id` int(11) NOT NULL AUTO\_INCREMENT,  
 `countryname` varchar(255) DEFAULT NULL,  
 `countrycode` varchar(255) DEFAULT NULL,  
 PRIMARY KEY (`id`)  
) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8;

使用XML 形式配置mybatis

MyBatis包含两种类型的xml文件，配置文件（config文件）和映射文件（mapper文件）。

config文件包含了影响 MyBatis 行为的设置和属性信息

mapper文件包含了配置的sql语句，存储过程以及映射规则

在src/main/resources 下面创建mybatis-config.xml 配置文件，并配置mybatis

logImpl属性配置指定使用LOG4J 输出日志

<settings>  
 <setting name="logImpl" value="LOG4J"/>  
</settings>

配置数据库连接

<dataSource type="UNPOOLED">  
 <property name="driver" value="com.mysql.cj.jdbc.Driver"/>  
 <property name="url" value="jdbc:mysql://localhost:3306/mybatis?serverTimezone=UTC"/>  
 <property name="username" value="root"/>  
 <property name="password" value="root"/>  
</dataSource>

创建实体类

src/main/java下添加一个包 com.mybatisChicken.model，在这个包下再新建实体类Country

public class Country {  
 private Long id;  
 private String countryname;  
 private String countrycode;

}

Country中的id时用类型Long而不是long。

由于Java 中的基本类型会有默认值，例如当某个类中存在private int age ；宇段时，创建这个类时， age 会有默认值0 。当使用age 属性时，它总会有值。因此在某些情况下，便无法实现使age 为口ull 。并且在动态SQL 的部分，如果使用age != null 进行判断，结果总会为true ，因而会导致很多隐藏的问题。所以，在实体类中不要使用基本类型。基本类型包括byte 、int , short 、long 、float 、  
double 、char 、boolean 。

创建mapper文件

src/main/resources下新建目录“mapper”，再在这个目录下新建CountryMapper文件。文件中mapper元素的namespace特性设置为noInterface.mapper.CountryMapper

<mapper namespace="noInterface.mapper.CountryMapper">  
 <select id="selectAll" resultType="com.mybatisChicken.model.Country">  
 select id,countryname,countrycode from country  
 </select>  
</mapper>

Namespace的值可以任意取，多个mapper文件可以共用一个namespace值，但同一namespace下不可以存在相同id的配置。上面配置了一个id为selectAll的查询语句。查询结果resultType特性指定的类型上，com.mybatisChicken.model.Country是刚才创建的实体类

回到config文件添加配置的映射

在mybatis-config.xml 配置文件，中添加对mapper 。resource特性填入mapper文件的路径

添加一个映射文件

<mappers>  
 <mapper resource="mapper/CountryMapper.xml"/>  
</mappers>

配置日志输出

在 src/main/resources下添加log4j.properties文件

添加

log4j.logger.noInterface.mapper.CountryMapper=TRACE

log4j.logger.noInterface.mapper.CountryMapper 对应的包是noInterface.mapper.CountryMapper

在MyBatis 的日志实现中，包名实际上是XML 配置中的namespace 特性值的一部分。所以这里的配置要和mapper文件中namespace特性的值一致

编写测试代码

在src/test/java中创建包com.mybatisChicken，在这个包下创建测试类。CountryMapperTest

在@ BeforeClass注解的方法中使用mybatis-config.xml配置文件初始化，sqlSessionFactory实例

Reader reader = Resources.getResourceAsReader("mybatis-config.xml");  
sqlSessionFactory= new SqlSessionFactoryBuilder().build(reader);

SqlSessionFactory 是MyBatis的核心，通常一个数据库连接配置对应一个SqlSessionFactory实例。它的最佳作用域是应用作用域。最简单的做法是使用单例模式或者静态单例模式。

在测试方法中获取数据

List<Country> countryList = sqlSession.selectList( "noInterface.mapper.CountryMapper.selectAll" );

通过SqlSession 的selectList 方法使用namespace+id作为参数，在CountryMapper.xml中查找到要执行SQL 查询。MyBatis 底层使用JDBC 执行SQL ，获得查询结果集ResultSet 后，根据resultType特性的值将结果映射为Country 类型的集合， 返回查询结果。

以上是mybatis的简单用法

Mapper文件中每次指定类名时需使用类的完全限定名。在config文件中添加包名后可以简化类名的书写

<typeAliases>  
 <package name="com.mybatisChicken.model"/>  
</typeAliases>

Mapper文件中使用类名时，不需要再写类的包名。

<select id="selectAll" resultType="com.mybatisChicken.model.Country">简化为

<select id="selectAll" resultType=" Country">

<http://www.mybatis.org/mybatis-3/zh/index.html>

使用xml配置

MyBatis 3.0 相比2.0 版本的一个最大变化，就是支持使用接口来调用方法。 以前使用SqlSession 通过命名空间调用MyBatis 方法时，首先需要用到命名空间和方法id 组成的字符串来调用相应的方法。当参数多于1 个的时候，需要将所有参数放到一个Map

对象中。通过Map 传递多个参数

使用接口调用方式就会方便很多， MyBatis 使用Java 的动态代理可以直接通过接口来调

用相应的方法，不需要提供接口的实现类，尤其在Spring 中使用的时候，可以配置为自动扫描所有的

接口类，直接将接口注入需要用到的地方

Country查询的示例采用接口调用方式可以写成

CountryMapper countrMapper = sqlSession.getMapper(CountryMapper.class);

countrMapper.selectAll();

下面演示配置接口的调用方式使用mybatis

在src/main/resources下新建路径com/mybatisChicken/mapper。在路径下新建UserMapper.xml

在src/main/java 下面创建包com.mybatisChicken.mapper在该包下创建XML 文件对应的接口类UserMapper.java

设置mapper文件的namespace

<mapper namespace="com.mybatisChicken.mapper.UserMapper">

需要注意的是<mapper>根标签的 namespace特性。当Mapper接口和XML文件关联的时候,命名空间 namespace的值就需要配置成接口的完全限定名称,例如 UserMapper接口对应的com.mybatisChicken.mapper.UserMapper, MyBatis内部就是通过这个值将接口和XML关联起来的

将这个mapper文件添加到config文件的mappers元素中

<mappers>  
 <mapper resource="com/ mybatisChicken /mapper/UserMapper.xml"/>  
</mappers>

在UserMapper 接口中添加一个selectByid 方法

**public interface** UserMapper {

SysUser selectById(Long id);

}

<**select id="selectById" resultMap="userMap"**>  
 select *\** from sys\_user where id = #{id}  
</**select**>

然后在对应的U serMapper.xml 中添加如下的＜ resultMap ＞和＜ select ＞部分的代码。

接口和XML 是通过将name space 的值设置为接口的全

限定名称来进行关联的，XML 中的select 标签的id 属性值和定义的接口方法名是一样的。MyBatis

就是通过这种方式将接口方法和XML 中定义的SQL 语句关联到一起的。

如果config中要添加很多mapper文件，有一个简化添加mapper文件的方法。使用包名的方式添加mapper文件

<mappers>  
 <package name="com. mybatisChicken.mapper"/>  
</mappers>

这种配置方式会先查找com.lile.mybatisExer.mapper包下所有的接口。加载接口对应的XML映射文件,将接口全限定名转换为路径,例如,将接口com. mybatisChicken.mapper.UserMapper转换为 com/ mybatisChicken /mapper/UserMapper.xml 以xml为后缀搜索XML资源,如果找到就解析XML。

因此，使用接口调用，并使用包名搜索mapper文件时，对mapper文件放置的路径和文件名是有要求的。路径名称+文件名要和接口完全限定名一致，而且mapper文件的namespce的值也要和接口完全限定名一致

Country查询的示例中mapper文件的路径和mapper文件中指定的namespace是没有关联的。

<mappers>  
 <mapper resource="mapper/CountryMapper.xml"/>  
</mappers>