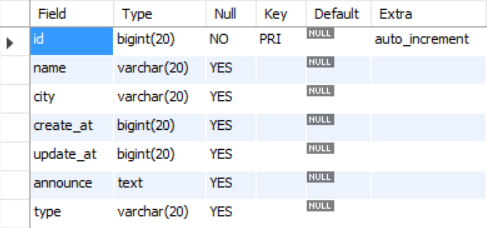
**一.SQL部分**

建表的sql在文件夹的sql文件中(草稿形式),学生信息表结构:



遇到的问题及解决:

1. 任务中没有说明对create\_at字段和update\_at字段怎么插入,我使用触发器在插入和更新时对相应字段进行更新.
2. create\_at字段和update\_at字段为long,但没有说明是怎样的表示形式:”20161121”还是” 1479713849”这样的Unix时间戳,我设置为后一种,方便转化. FROM\_UNIXTIME(unix\_timestamp)和UNIX\_TIMESTAMP(NOW());
3. MySQL中没有long类型,若设置了long则会自动映射成mediumtext就不能设置自增了,mysql中对应long的为bigint,且自增字段必需为key.
4. 在同一个表中修改的触发器不能同时使用(update,insert,delete),需要同时更新字段时结合使用set,new.

eg:

delimiter //

create trigger auto\_insert before insert

on stu\_info for each row

begin

set new.create\_at=UNIX\_TIMESTAMP(NOW());

end//

delimiter ;

1. 触发器中,new的更新操作不能在after字段上进行,因为after后已经对数据库写完毕,不能再对new进行修改.能用new取值.
2. 安全模式下更新需要在where中需指定主键
3. 最后对任务中的插入十条数据,然后在name字段建立索引,比较前后查找数据吐下槽,十条数据的表建不建索引查找时间都忽略不计,Duration和fetch time都为0.000s.所以任务略奇怪,好歹插多点再比.

收获:

复习了一遍SQL语法.

**二.java部分**

Eclipse maven之前装好了,这里直接使用mybatis框架访问数据库.

对于mybatis有一个generator自动映射工具 ,这里描述一下配置.

首先是pom文件:

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>cn.com.lenovo</groupId>

<artifactId>db\_operate</artifactId>

<packaging>war</packaging>

<version>0.0.1-SNAPSHOT</version>

<name>db\_operate Maven Webapp</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.9</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.39</version>

</dependency>

<!-- dbcp连接池的依赖包 -->

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.4</version>

</dependency>

<!-- mybatis依赖包 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>3.4.1</version>

</dependency>

<!-- 添加日志支持 -->

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.17</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>1.7.7</version>

</dependency>

</dependencies>

<build>

<finalName>db\_operate</finalName>

<plugins>

<plugin>

<groupId>org.mybatis.generator</groupId>

<artifactId>mybatis-generator-maven-plugin</artifactId>

<version>1.3.2</version>

<configuration>

<configurationFile>src/main/resources/mybatis-generator.xml</configurationFile>

<overwrite>true</overwrite>

</configuration>

<dependencies>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.39</version>

</dependency>

</dependencies>

</plugin>

</plugins>

</build>

</project>

说明:在pom文件中导入mysql连接工具,,mybatis包,日志工具(其中光是log4j不够,还需一个slf4j-log4j12),添加mybatis-generator插件,其中配置mysql连接,指定配置文件位置.

接下来是配置文件:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE generatorConfiguration

PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"

"http://mybatis.org/dtd/mybatis-generator-config\_1\_0.dtd">

<generatorConfiguration>

<!-- <classPathEntry location="E:\repo\mysql\mysql-connector-java\5.1.26\mysql-connector-java-5.1.26.jar" /> -->

<context id=*"generatorTables"* targetRuntime=*"MyBatis3"*>

<plugin type=*"org.mybatis.generator.plugins.MapperConfigPlugin"*>

<property name=*"fileName"* value=*"mybatis-config.xml"* />

<property name=*"targetPackage"* value=*"/"* />

<property name=*"targetProject"* value=*"src/main/resources"* />

</plugin>

<plugin type=*"org.mybatis.generator.plugins.SerializablePlugin"* />

<!-- commentGenerator 去除自动生成的注释 -->

<commentGenerator>

<property name=*"suppressAllComments"* value=*"true"* />

<property name=*"suppressDate"* value=*"true"* />

</commentGenerator>

<jdbcConnection

driverClass=*"com.mysql.jdbc.Driver"*

connectionURL=*"jdbc:mysql://127.0.0.1:3306/jnshu"*

userId=*"root"*

password=*"941114"*>

</jdbcConnection>

<!-- javaModelGenerator是模型的生成信息，这里将指定这些Java model类的生成路径； -->

<javaModelGenerator targetPackage=*"pojo"* targetProject=*"src/main/java"*>

<property name=*"enableSubPackages"* value=*"true"* />

<property name=*"trimStrings"* value=*"true"* />

</javaModelGenerator>

<!-- sqlMapGenerator是mybatis 的sqlMapper XML文件的生成信息，包括生成路径等； 先生成xml，在生成java -->

<sqlMapGenerator targetPackage=*"mapper"* targetProject=*"src/main/java"*>

<property name=*"enableSubPackages"* value=*"true"* />

</sqlMapGenerator>

<!-- javaClientGenerator是应用接口的生成信息； -->

<javaClientGenerator type=*"XMLMAPPER"*

targetPackage=*"mapper"* targetProject=*"src/main/java"*>

<property name=*"enableSubPackages"* value=*"true"* />

</javaClientGenerator>

<!-- table是用户指定的被生成相关信息的表，它必须在指定的jdbc连接中已经被建立。?是否可以多个 -->

<table tableName=*"stu\_info"* domainObjectName=*"StuInfo"*

enableCountByExample=*"false"* enableUpdateByExample=*"false"*

enableDeleteByExample=*"false"* enableSelectByExample=*"false"*

selectByExampleQueryId=*"false"*></table>

<!-- <table tableName="app\_bind\_status" domainObjectName="AppBindStatus"

enableCountByExample="false" enableUpdateByExample="false" enableDeleteByExample="false"

enableSelectByExample="false" selectByExampleQueryId="false"></table> -->

</context>

</generatorConfiguration>

<!-- mybatis-generator:generate 执行命令-->

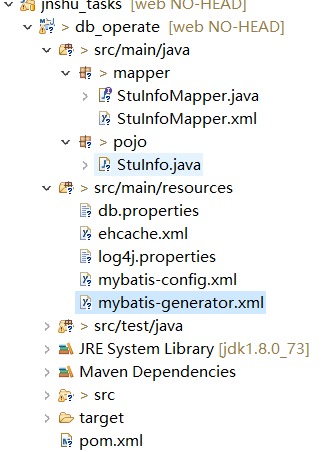
说明:基本都有注释,可以看官方文档有详细说明.其中生成器会生成三个文件:xml,dao,pojo.

为什么木有impl文件呢?因为impl文件中的操作都是类似的,打开session创建连接,执行操作返回数据,关闭连接.所以mybatis自动完成了这些,但是需要遵循一些规范:xml文件和dao接口需要放在一个文件夹下并且文件名必需相同.

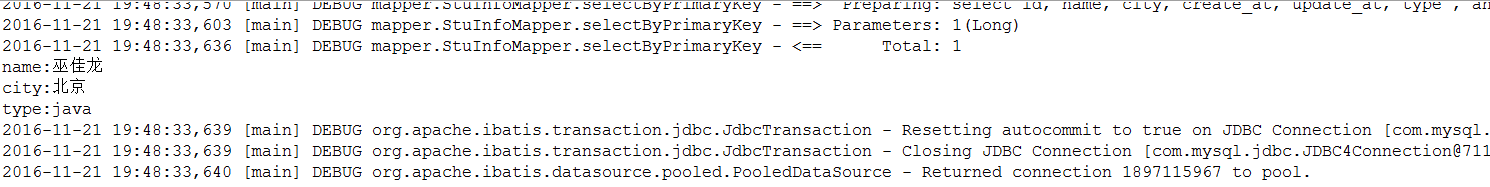
另外配置文件中必需还有数据库连接的driver,表名映射来的pojo类名.

最后在eclipse中debug as->maven build->在globals中填入maven指令:mybatis-generator:generate

自动生成的文件目录:



创建测试用例:右键StuInfoMapper.java文件new一个Junit test case,测试连接读取数据库:



成功!(因为没有spring需要出现的地方就暂不加进来了)

此处截个快照,把源码放进任务文件夹中,明天开始折腾阿里云.