**第1步：**Intellij下打开工程：MyBatisGenerator

**第2步：修改配置文件：generatorConfig.xml，特别是新增加table后，必须告知那些表生成配置文件和xml以及java代码。**

**第3步：Run the project now。**

**第4步：拷贝修改数据库表对应配置文档和实体类，注意不用拷贝文件，而是拷贝文件内容防止出错！**

**改变的三类文件：**

**1. domain： entity class**

**2. mapper xml**

**3. mapper class，由于业务代码添加在此，需要核对后更新，不用直接拷贝文件！！！**

工程建立步骤参考：<http://blog.csdn.net/z69183787/article/details/46560071>

Intellij IDEA 14 作为Java IDE 神器，接触后发现，非常好用，对它爱不释手，打算离开eclipse和myeclipse，投入Intellij IDEA的怀抱。

     然而在使用的过程中会发现Intellij IDEA也有一些不尽如意的地方，难免会有些不爽：Intellij IDEA 的插件库远不及eclipse的丰富。 mybatis-generator在eclipse中有专门的插件，而没有开发出Intellij IDEA能够使用的插件。

    不过不用灰心，如果你的项目是使用maven组织的，那么我们可以在Intellij IDEA中使用 mybatis-generator-maven-plugin插件来完成MyBatis model 和Mapper文件的自动生成。

#### STEP 0.在Intellij IDEA创建maven项目（本过程比较简单，略）

#### STEP 1. 在maven项目的pom.xml 添加mybatis-generator-maven-plugin 插件

**[html]** [view plain](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "copy" \t "_blank)[CODE上查看代码片](https://code.csdn.net/snippets/590183)

1. **<build>**
2. **<finalName>**xxx**</finalName>**
3. **<plugins>**
4. **<plugin>**
5. **<groupId>**org.mybatis.generator**</groupId>**
6. **<artifactId>**mybatis-generator-maven-plugin**</artifactId>**
7. **<version>**1.3.2**</version>**
8. **<configuration>**
9. **<verbose>**true**</verbose>**
10. **<overwrite>**true**</overwrite>**
11. **</configuration>**
12. **</plugin>**
13. **</plugins>**
14. **</build>**

#### STEP 2. 在maven项目下的src/main/resources 目录下建立名为 generatorConfig.xml的配置文件，作为mybatis-generator-maven-plugin 插件的执行目标，模板如下：

**[html]** [view plain](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "copy" \t "_blank)[CODE上查看代码片](https://code.csdn.net/snippets/590183)

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. <!DOCTYPE generatorConfiguration
3. PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"
4. "http://mybatis.org/dtd/mybatis-generator-config\_1\_0.dtd"**>**
5. **<generatorConfiguration>**
6. <!--导入属性配置 -->
7. **<properties** resource="generator.properties"**></properties>**
9. <!--指定特定数据库的jdbc驱动jar包的位置 -->
10. **<classPathEntry** location="${jdbc.driverLocation}"**/>**
12. **<context** id="default" targetRuntime="MyBatis3"**>**

15. <!-- optional，旨在创建class时，对注释进行控制 -->
16. **<commentGenerator>**
17. **<property** name="suppressDate" value="true" **/>**
18. **</commentGenerator>**

21. <!--jdbc的数据库连接 -->
22. **<jdbcConnection** driverClass="${jdbc.driverClass}" connectionURL="${jdbc.connectionURL}" userId="${jdbc.userId}" password="${jdbc.password}"**>**
23. **</jdbcConnection>**


27. <!-- 非必需，类型处理器，在数据库类型和java类型之间的转换控制-->
28. **<javaTypeResolver** **>**
29. **<property** name="forceBigDecimals" value="false" **/>**
30. **</javaTypeResolver>**
32. <!-- Model模型生成器,用来生成含有主键key的类，记录类 以及查询Example类
33. targetPackage     指定生成的model生成所在的包名
34. targetProject     指定在该项目下所在的路径
35. --**>**
36. **<javaModelGenerator** targetPackage="org.louis.hometutor.po" targetProject="src/main/java"**>**
37. <!-- 是否对model添加 构造函数 -->
38. **<property** name="constructorBased" value="true"**/>**
40. <!-- 是否允许子包，即targetPackage.schemaName.tableName -->
41. **<property** name="enableSubPackages" value="false"**/>**
43. <!-- 建立的Model对象是否 不可改变  即生成的Model对象不会有 setter方法，只有构造方法 -->
44. **<property** name="immutable" value="true"**/>**
46. <!-- 给Model添加一个父类 -->
47. **<property** name="rootClass" value="com.foo.louis.Hello"**/>**
49. <!-- 是否对类CHAR类型的列的数据进行trim操作 -->
50. **<property** name="trimStrings" value="true"**/>**
51. **</javaModelGenerator>**
53. <!--Mapper映射文件生成所在的目录 为每一个数据库的表生成对应的SqlMap文件 -->
54. **<sqlMapGenerator** targetPackage="org.louis.hometutor.domain" targetProject="src/main/java"**>**
55. **<property** name="enableSubPackages" value="false"**/>**
56. **</sqlMapGenerator>**

59. <!-- 客户端代码，生成易于使用的针对Model对象和XML配置文件 的代码
60. type="ANNOTATEDMAPPER",生成Java Model 和基于注解的Mapper对象
61. type="MIXEDMAPPER",生成基于注解的Java Model 和相应的Mapper对象
62. type="XMLMAPPER",生成SQLMap XML文件和独立的Mapper接口
63. --**>**
64. **<javaClientGenerator** targetPackage="com.foo.tourist.dao" targetProject="src/main/java" type="MIXEDMAPPER"**>**
65. **<property** name="enableSubPackages" value=""**/>**
66. <!--
67. 定义Maper.java 源代码中的ByExample() 方法的可视性，可选的值有：
68. public;
69. private;
70. protected;
71. default
72. 注意：如果 targetRuntime="MyBatis3",此参数被忽略
73. --**>**
74. **<property** name="exampleMethodVisibility" value=""**/>**
75. <!--
76. 方法名计数器
77. Important note: this property is ignored if the target runtime is MyBatis3.
78. --**>**
79. **<property** name="methodNameCalculator" value=""**/>**
81. <!--
82. 为生成的接口添加父接口
83. -->
84. **<property** name="rootInterface" value=""**/>**
86. **</javaClientGenerator>**


90. **<table** tableName="lession" schema="louis"**>**
92. <!-- optional   , only for mybatis3 runtime
93. 自动生成的键值（identity,或者序列值）
94. 如果指定此元素，MBG将会生成**<selectKey>**元素，然后将此元素插入到SQL Map的**<insert>** 元素之中
95. sqlStatement 的语句将会返回新的值
96. 如果是一个自增主键的话，你可以使用预定义的语句,或者添加自定义的SQL语句. 预定义的值如下:
97. Cloudscape    This will translate to: VALUES IDENTITY\_VAL\_LOCAL()
98. DB2:      VALUES IDENTITY\_VAL\_LOCAL()
99. DB2\_MF:       SELECT IDENTITY\_VAL\_LOCAL() FROM SYSIBM.SYSDUMMY1
100. Derby:        VALUES IDENTITY\_VAL\_LOCAL()
101. HSQLDB:   CALL IDENTITY()
102. Informix:     select dbinfo('sqlca.sqlerrd1') from systables where tabid=1
103. MySql:        SELECT LAST\_INSERT\_ID()
104. SqlServer:    SELECT SCOPE\_IDENTITY()
105. SYBASE:   SELECT @@IDENTITY
106. JDBC:     This will configure MBG to generate code for MyBatis3 suport of JDBC standard generated keys. This is a database independent method of obtaining the value from identity columns.
107. identity: 自增主键  If true, then the column is flagged as an identity column and the generated **<selectKey>** element will be placed after the insert (for an identity column). If false, then the generated **<selectKey>** will be placed before the insert (typically for a sequence).
109. --**>**
110. **<generatedKey** column="" sqlStatement="" identity="" type=""**/>**

113. <!-- optional.
114. 列的命名规则：
115. MBG使用 **<columnRenamingRule>** 元素在计算列名的对应 名称之前，先对列名进行重命名，
116. 作用：一般需要对BUSI\_CLIENT\_NO 前的BUSI\_进行过滤
117. 支持正在表达式
118. searchString 表示要被换掉的字符串
119. replaceString 则是要换成的字符串，默认情况下为空字符串，可选
120. --**>**
121. **<columnRenamingRule** searchString="" replaceString=""**/>**


125. <!-- optional.告诉 MBG 忽略某一列
126. column，需要忽略的列
127. delimitedColumnName:true ,匹配column的值和数据库列的名称 大小写完全匹配，false 忽略大小写匹配
128. 是否限定表的列名，即固定表列在Model中的名称
129. --**>**
130. **<ignoreColumn** column="PLAN\_ID"  delimitedColumnName="true" **/>**

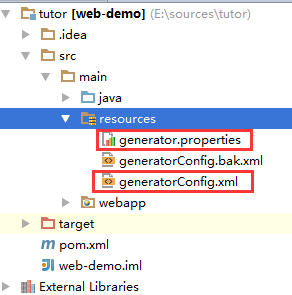
133. <!--optional.覆盖MBG对Model 的生成规则
134. column: 数据库的列名
135. javaType: 对应的Java数据类型的完全限定名
136. 在必要的时候可以覆盖由JavaTypeResolver计算得到的java数据类型. For some databases, this is necessary to handle "odd" database types (e.g. MySql's unsigned bigint type should be mapped to java.lang.Object).
137. jdbcType:该列的JDBC数据类型(INTEGER, DECIMAL, NUMERIC, VARCHAR, etc.)，该列可以覆盖由JavaTypeResolver计算得到的Jdbc类型，对某些数据库而言，对于处理特定的JDBC 驱动癖好 很有必要(e.g. DB2's LONGVARCHAR type should be mapped to VARCHAR for iBATIS).
138. typeHandler:
140. --**>**
141. **<columnOverride** column="" javaType=""    jdbcType="" typeHandler=""  delimitedColumnName="" **/>**
143. **</table>**
144. **</context>**
145. **</generatorConfiguration>**

这里使用了外置的配置文件generator.properties，可以将一下属性配置到properties文件之中，增加配置的灵活性：

**[plain]** [view plain](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "copy" \t "_blank)[CODE上查看代码片](https://code.csdn.net/snippets/590183)

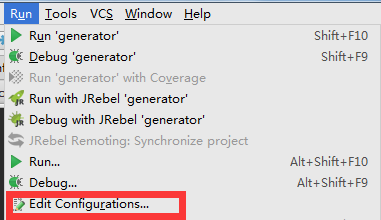
1. jdbc.driverLocation=D:\\maven\\com\\oracle\\ojdbc14\\10.2.0.4.0\\ojdbc14-10.2.0.4.0.jar
2. jdbc.driverClass=oracle.jdbc.driver.OracleDriver
3. jdbc.connectionURL=jdbc:oracle:thin:@//localhost:1521/XE
4. jdbc.userId=LOUIS
5. jdbc.password=123456

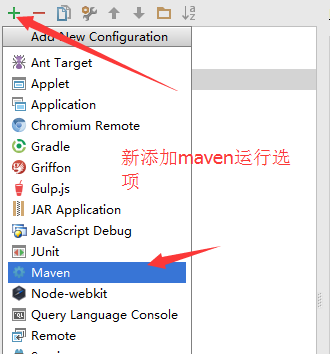
项目目录如下：



#### STEP 3. 在Intellij IDEA添加一个“Run运行”选项，使用maven运行mybatis-generator-maven-plugin插件 ：

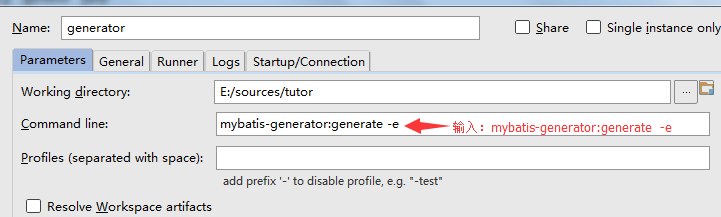
#### 在Intellij IDEA添加一个“Run运行”选项，使用maven运行mybatis-generator-maven-plugin插件 ：

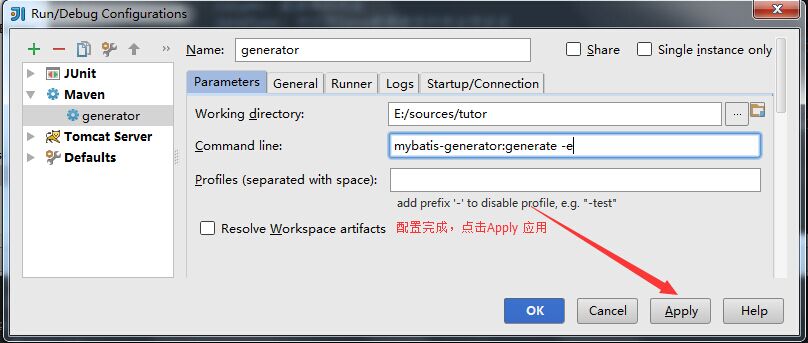




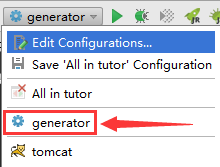
之后弹出运行配置框，为当前配置配置一个名称，这里其名为"generator",然后在 “Command line” 选项中输入“mybatis-generator:generate  -e”

这里加了“-e ”选项是为了让该插件输出详细信息，这样可以帮助我们定位问题。





如果添加成功，则会在run 选项中有“generator” 选项，如下：



点击运行，然后不出意外的话，会在控制台输出：

**[plain]** [view plain](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "view plain" \t "_blank)[copy](http://blog.csdn.net/sunny243788557/article/details/45166397" \o "copy" \t "_blank)[CODE上查看代码片](https://code.csdn.net/snippets/590183)

1. C:\Java\jdk1.7.0\_71\bin\java -Dmaven.home=D:\software\apache-maven-3.0.5-bin -Dclassworlds.conf=D:\software\apache-maven-3.0.5-bin\bin\m2.conf -Didea.launcher.port=7533 "-Didea.launcher.bin.path=D:\applications\JetBrains\IntelliJ IDEA 14.0.2\bin" -Dfile.encoding=UTF-8 -classpath "D:\software\apache-maven-3.0.5-bin\boot\plexus-classworlds-2.4.jar;D:\applications\JetBrains\IntelliJ IDEA 14.0.2\lib\idea\_rt.jar" com.intellij.rt.execution.application.AppMain org.codehaus.classworlds.Launcher -Didea.version=14.0.2 -s D:\software\apache-maven-3.0.5-bin\conf\settings.xml mybatis-generator:generate -e
2. [INFO] Error stacktraces are turned on.
3. [INFO] Scanning for projects...
4. [INFO]
5. [INFO] ------------------------------------------------------------------------
6. [INFO] Building hometutor Maven Webapp 1.0-SNAPSHOT
7. [INFO] ------------------------------------------------------------------------
8. [INFO]
9. [INFO] --- mybatis-generator-maven-plugin:1.3.2:generate (default-cli) @ hometutor ---
10. [INFO] Connecting to the Database
11. [INFO] Introspecting table louis.lession
12. log4j:WARN No appenders could be found for logger (org.mybatis.generator.internal.db.DatabaseIntrospector).
13. log4j:WARN Please initialize the log4j system properly.
14. log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
15. [INFO] Generating Example class for table LOUIS.LESSION
16. [INFO] Generating Record class for table LOUIS.LESSION
17. [INFO] Generating Mapper Interface for table LOUIS.LESSION
18. [INFO] Generating SQL Map for table LOUIS.LESSION
19. [INFO] Saving file LessionMapper.xml
20. [INFO] Saving file LessionExample.java
21. [INFO] Saving file Lession.java
22. [INFO] Saving file LessionMapper.java
23. [WARNING] Root class com.foo.louis.Hello cannot be loaded, checking for member overrides is disabled for this class
24. [WARNING] Existing file E:\sources\tutor\src\main\java\org\louis\hometutor\po\Lession.java was overwritten
25. [WARNING] Existing file E:\sources\tutor\src\main\java\com\foo\tourist\dao\LessionMapper.java was overwritten
26. [INFO] ------------------------------------------------------------------------
27. [INFO] BUILD SUCCESS
28. [INFO] ------------------------------------------------------------------------
29. [INFO] Total time: 2.334s
30. [INFO] Finished at: Tue Jan 27 12:04:08 CST 2015
31. [INFO] Final Memory: 8M/107M
32. [INFO] ------------------------------------------------------------------------
34. Process finished with exit code 0

好，看到BUILD SUCCESS，则大功告成，如果有错误的话，由于添加了-e 选项，会把具体的详细错误信息打印出来的，根据错误信息修改即可