一、入门

通用Mapper都可以极大的方便开发人员。可以随意的按照自己的需要选择通用方法，还可以很方便的开发自己的通用方法。

极其方便的使用MyBatis单表的增删改查。

支持单表操作，不支持通用的多表联合查询

考虑到基本数据类型在 Java 类中都有默认值，会导致 MyBatis 在执行相关操作 时很难判断当前字段是否为 null，所以在 MyBatis 环境下使用 Java 实体类时尽量不 要使用基本数据类型，都使用对应的包装类型。

```
<!-- 整合通用 Mapper 所需要做的配置修改： -->
<!-- 原始全类名：org.mybatis.spring.mapper.MapperScannerConfigurer -->
<!-- 通用 Mapper 使用：tk.mybatis.spring.mapper.MapperScannerConfigurer -->
```

具体操作数据库的Mapper接口，需要继承通用Mapper提供的核心接口：Mapper<>

二、注解

- @Table：建立实体类和数据库表之间的对应关系。
- @Column：建立实体类字段和数据库表字段之间的对应关系。
- @Id：明确标记和数据库表中主键字段对应的实体类字段。
- @GeneratedValue：让通用 Mapper 在执行 insert 操作之后将数据库自动生成的主键值回写到实体类对象中。
- @Transient：用于标记不与数据库表字段对应的实体类字段

三、方法

- selectOne 方法
- xxxByPrimaryKey 方法：需要使用@Id 主键明确标记和数据库表主键字段对应的实体类字段
- xxxSelective 方法：非主键字段如果为 null 值，则不加入到 SQL 语句中

四、QBC查询

Query By Criteria

Criteria 是 Criterion 的复数形式。意思是：规则、标准、准则。在 SQL 语句中相当 于**查询条件**。

QBC 查询是将查询条件通过 **Java 对象**进行模块化封装。

五、逆向工程

原生 MyBatis 逆向工程： MBG工具

通用 Mapper 逆向工程： 通用 Mapper 专用代码生成器——mappergenerator，在 MBG 中使用该插件

通用 Mapper 专用代码生成器生成的 Model 会在原有基础上增加 `@Table,@Id,@Column` 等注解，方便自动会数据库字段进行映射。

generatorConfig.xml中，和一般的配置相比，这里只是多了一个插件的配置：

```
<plugin type="tk.mybatis.mapper.generator.MapperPlugin">
    <property name="mappers" value="tk.mybatis.mapper.common.Mapper"/>
    <property name="caseSensitive" value="true"/>
    <property name="forceAnnotation" value="true"/>
    <property name="beginningDelimiter" value="`"/>
    <property name="endingDelimiter" value="`"/>
</plugin>
```

这里最关键的参数就是 `mappers`，配置后生成的 Mapper 接口都会自动继承上改接口，

- caseSensitive 是否区分大小写，默认值 `false`。如果数据库区分大小写，这里就需要配置为 `true`，这样当表名为 `USER` 时，会生成 `@Table(name = "USER")` 注解，否则使用小写 `user` 时会找不到表。
- forceAnnotation 是否强制生成注解，默认 `false`，如果设置为 `true`，不管数据库名和字段名是否一致，都会生成注解（包含 `@Table` 和 `@Column`）。
- beginningDelimiter 和 endingDelimiter 开始和结束分隔符，对于有关键字的情况下适用。
- useMapperCommentGenerator 是否使用通用 Mapper 提供的注释工具，默认 `true` 使用，这样在生成代码时会包含字段的注释（目前只有 mysql 和 oracle 支持），设置 `false` 后会用默认的，或者你可以配置自己的注释插件。
- generateColumnConsts 在生成的 model中，增加字段名的常量，便于使用 Example 拼接查询条件的时候使用。
- lombok 增加 model 代码生成时，可以直接生成 lombok 的 `@Getter@Setter@ToString@Accessors(chain = true)` 四类注解， 使用者在插件配置项中增加 `<property name="lombok" value="Getter,Setter,ToString,Accessors"/>` 即可生成对应包含注解的 model 类。

六、自定义Mapper<T>接口

让我们可以根据开发的实际需要对 Mapper接口进行定制。

Mapper

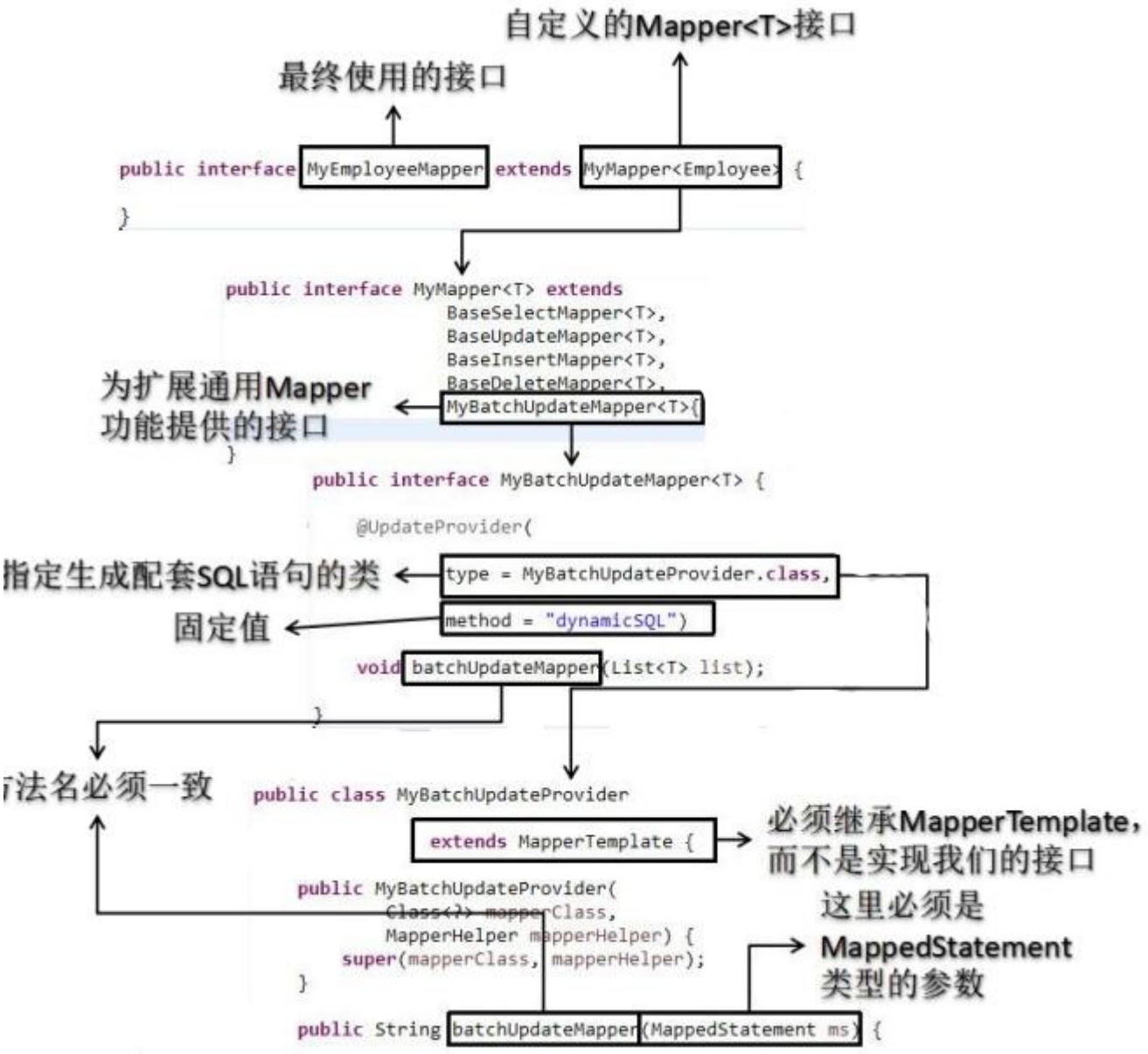
1. BaseMapper
    1. BaseXXXMapper
2. ExampleMapper
    1. XXXByExampleMapper
3. RowBoundsMapper

1. SelectByExampleRowBoundsMapper
2. SelectRowBoundsMapper

4. Marker

七、通用Mapper接口扩展

增加通用 Mapper 没有提供的功能。



八、二级缓存

在 XxxMapper 接口上使用@CacheNamespace 注解

九、类型处理器：TypeHandler

- BaseTypeHandler
- 自定义类型转换器类
- 枚举类型
  - 方法一：让通用 Mapper 把枚举类型作为简单类型处理
    - 本质使用了 org.apache.ibatis.type.EnumTypeHandler
  - 方法二：为枚举类型配置对应的类型处理器
    - 类型处理器
      - 内置 org.apache.ibatis.type.EnumTypeHandler 在数据库中存储枚举值本身
      - org.apache.ibatis.type.EnumOrdinalTypeHandler 在数据库中仅仅存储枚举值的索引
    - 自定义内置枚举类型处理器注册
      - 不能使用@ColumnType 注解
      - 需要在 MyBatis 配置文件中配置专门的类型处理器并在字段上使用