> 本篇文章是JavaGuide收集自网络，原出处不明。

MyBatis 技术内幕系列博客，从原理和源码角度，介绍了其内部实现细节，无论是写的好与不好，我确实是用心写了，由于并不是介绍如何使用 MyBatis 的文章，所以，一些参数使用细节略掉了，我们的目标是介绍 MyBatis 的技术架构和重要组成部分，以及基本运行原理。

博客写的很辛苦，但是写出来却不一定好看，所谓开始很兴奋，过程很痛苦，结束很遗憾。要求不高，只要读者能从系列博客中，学习到一点其他博客所没有的技术点，作为作者，我就很欣慰了，我也读别人写的博客，通常对自己当前研究的技术，是很有帮助的。

尽管还有很多可写的内容，但是，我认为再写下去已经没有意义，任何其他小的功能点，都是在已经介绍的基本框架和基本原理下运行的，只有结束，才能有新的开始。写博客也积攒了一些经验，源码多了感觉就是复制黏贴，源码少了又觉得是空谈原理，将来再写博客，我希望是“精炼博文”，好读好懂美观读起来又不累，希望自己能再写一部开源分布式框架原理系列博客。

有胆就来，我出几道 MyBatis 面试题，看你能回答上来几道（都是我出的，可不是网上找的）。

1、#{}和${}的区别是什么?

注：这道题是面试官面试我同事的。

答：

- `${}`是 Properties 文件中的变量占位符，它可以用于标签属性值和 sql 内部，属于静态文本替换，比如${driver}会被静态替换为`com.mysql.jdbc.Driver`。
- `#{}`是 sql 的参数占位符，MyBatis 会将 sql 中的`#{}`替换为?号，在 sql 执行前会使用 PreparedStatement 的参数设置方法，按序给 sql 的?号占位符设置参数值，比如 ps.setInt(0, parameterValue)，`#{item.name}` 的取值方式为使用反射从参数对象中获取 item 对象的 name 属性值，相当于`param.getItem().getName()`。

2、Xml 映射文件中，除了常见的 select|insert|update|delete 标签之外，还有哪些标签?

注：这道题是京东面试官面试我时问的。

答：还有很多其他的标签，`<resultMap>`、`<parameterMap>`、`<sql>`、`<include>`、`<selectKey>`，加上动态 sql 的 9 个标签，`trim|where|set|foreach|if|choose|when|otherwise|bind`等，其中为 sql 片段标签，通过`<include>`标签引入 sql 片段，`<selectKey>`为不支持自增的主键生成策略标签。

3、最佳实践中，通常一个 Xml 映射文件，都会写一个 Dao 接口与之对应，请问，这个 Dao 接口的工作原理是什么？Dao 接口里的方法，参数不同时，方法能重载吗?

注：这道题也是京东面试官面试我被问的。

答：Dao 接口，就是人们常说的`Mapper`接口，接口的全限名，就是映射文件中的 namespace 的值，接口的方法名，就是映射文件中`MappedStatement`的 id 值，接口方法内的参数，就是传递给 sql 的参数。`Mapper`接口是没有实现类的，当调用接口方法时，接口全限名+方法名拼接字符串作为 key 值，可唯一定位一个`MappedStatement`，举例：`com.mybatis3.mappers.StudentDao.findStudentById`，可以唯一找到 namespace 为`com.mybatis3.mappers.StudentDao`下面`id = findStudentById`的`MappedStatement`。在

MyBatis 中，每一个`<select>`、`<insert>`、`<update>`、`<delete>`标签，都会被解析为一个`MappedStatement`对象。

~~Dao 接口里的方法，是不能重载的，因为是全限名+方法名的保存和寻找策略。~~

Dao 接口里的方法可以重载，但是Mybatis的XML里面的ID不允许重复。

Mybatis版本3.3.0，亲测如下：

```
/**
 * Mapper接口里面方法重载
 */
public interface StuMapper {

    List<Student> getAllStu();

    List<Student> getAllStu(@Param("id") Integer id);
}
```

然后在`StuMapper.xml`中利用Mybatis的动态sql就可以实现。

```
    <select id="getAllStu" resultType="com.pojo.Student">
        select * from student
        <where>
            <if test="id != null">
                id = #{id}
            </if>
        </where>
    </select>
```

能正常运行，并能得到相应的结果，这样就实现了在Dao接口中写重载方法。

Mybatis 的 Dao 接口可以有多个重载方法，但是多个接口对应的映射必须只有一个，否则启动会报错。

相关 issue：更正：Dao 接口里的方法可以重载，但是Mybatis的XML里面的ID不允许重复！。

Dao 接口的工作原理是 JDK 动态代理，MyBatis 运行时会使用 JDK 动态代理为 Dao 接口生成代理 proxy 对象，代理对象 proxy 会拦截接口方法，转而执行`MappedStatement`所代表的 sql，然后将 sql 执行结果返回。

==补充：==

Dao接口方法可以重载，但是需要满足以下条件：

1. 仅有一个无参方法和一个有参方法
2. 多个有参方法时，参数数量必须一致。且使用相同的`@Param`，或者使用`param1`这种

测试如下：

`PersonDao.java`

```
Person queryById();

Person queryById(@Param("id") Long id);

Person queryById(@Param("id") Long id, @Param("name") String name);
```

PersonMapper.xml

```
<select id="queryById" resultMap="PersonMap">
    select
      id, name, age, address
    from person
    <where>
        <if test="id != null">
            id = #{id}
        </if>
        <if test="name != null and name != ''">
            name = #{name}
        </if>
    </where>
    limit 1
</select>
```

org.apache.ibatis.scripting.xmltags.DynamicContext.ContextAccessor#getProperty方法用于获取<if>标签中的条件值

```
public Object getProperty(Map context, Object target, Object name) {
  Map map = (Map) target;

  Object result = map.get(name);
  if (map.containsKey(name) || result != null) {
    return result;
  }

  Object parameterObject = map.get(PARAMETER_OBJECT_KEY);
  if (parameterObject instanceof Map) {
    return ((Map)parameterObject).get(name);
  }

  return null;
}
```

parameterObject为map，存放的是Dao接口中参数相关信息。

((Map)parameterObject).get(name)方法如下

```
public V get(Object key) {
  if (!super.containsKey(key)) {
    throw new BindingException("Parameter '" + key + "' not found. Available
parameters are " + keySet());
  }
  return super.get(key);
}
```

1. queryById()方法执行时，parameterObject为null，getProperty方法返回null值，<if>标签获取的所有条件值都为null，所有条件不成立，动态sql可以正常执行。
2. queryById(1L)方法执行时，parameterObject为map，包含了id和param1两个key值。当获取<if>标签中name的属性值时，进入((Map)parameterObject).get(name)方法中，map中key不包含name，所以抛出异常。
3. queryById(1L,"1")方法执行时，parameterObject中包含id,param1,name,param2四个key值，id和name属性都可以获取到，动态sql正常执行。

4、MyBatis 是如何进行分页的？分页插件的原理是什么？

注：我出的。

答：MyBatis 使用 RowBounds 对象进行分页，它是针对 ResultSet 结果集执行的内存分页，而非物理分页，可以在 sql 内直接书写带有物理分页的参数来完成物理分页功能，也可以使用分页插件来完成物理分页。

分页插件的基本原理是使用 MyBatis 提供的插件接口，实现自定义插件，在插件的拦截方法内拦截待执行的 sql，然后重写 sql，根据 dialect 方言，添加对应的物理分页语句和物理分页参数。

举例：`select _ from student`，拦截 sql 后重写为：`select t._ from (select \* from student) t limit 0, 10`

5、简述 MyBatis 的插件运行原理，以及如何编写一个插件。

注：我出的。

答：MyBatis 仅可以编写针对 `ParameterHandler`、`ResultSetHandler`、`StatementHandler`、`Executor` 这 4 种接口的插件，MyBatis 使用 JDK 的动态代理，为需要拦截的接口生成代理对象以实现接口方法拦截功能，每当执行这 4 种接口对象的方法时，就会进入拦截方法，具体就是 `InvocationHandler` 的 `invoke()`方法，当然，只会拦截那些你指定需要拦截的方法。

实现 MyBatis 的 Interceptor 接口并复写 `intercept()`方法，然后在给插件编写注解，指定要拦截哪一个接口的哪些方法即可，记住，别忘了在配置文件中配置你编写的插件。

6、MyBatis 执行批量插入，能返回数据库主键列表吗？

注：我出的。

答：能，JDBC 都能，MyBatis 当然也能。

7、MyBatis 动态 sql 是做什么的？都有哪些动态 sql？能简述一下动态 sql 的执行原理不？

注：我出的。

答：MyBatis 动态 sql 可以让我们在 Xml 映射文件内，以标签的形式编写动态 sql，完成逻辑判断和动态拼接 sql 的功能，MyBatis 提供了 9 种动态 sql 标签 `trim|where|set|foreach|if|choose|when|otherwise|bind`。

其执行原理为，使用 OGNL 从 sql 参数对象中计算表达式的值，根据表达式的值动态拼接 sql，以此来完成动态 sql 的功能。

8、MyBatis 是如何将 sql 执行结果封装为目标对象并返回的？都有哪些映射形式？

注：我出的。

答：第一种是使用`<resultMap>`标签，逐一定义列名和对象属性名之间的映射关系。第二种是使用 sql 列的别名功能，将列别名书写为对象属性名，比如 T_NAME AS NAME，对象属性名一般是 name，小写，但是列名不区分大小写，MyBatis 会忽略列名大小写，智能找到与之对应对象属性名，你甚至可以写成 T_NAME AS NaMe，MyBatis 一样可以正常工作。

有了列名与属性名的映射关系后，MyBatis 通过反射创建对象，同时使用反射给对象的属性逐一赋值并返回，那些找不到映射关系的属性，是无法完成赋值的。

9、MyBatis 能执行一对一、一对多的关联查询吗？都有哪些实现方式，以及它们之间的区别。

注：我出的。

答：能，MyBatis 不仅可以执行一对一、一对多的关联查询，还可以执行多对一，多对多的关联查询，多对一查询，其实就是一对一查询，只需要把 `selectOne()`修改为 `selectList()`即可；多对多查询，其实就是一对多查询，只需要把 `selectOne()`修改为 `selectList()`即可。

关联对象查询，有两种实现方式，一种是单独发送一个 sql 去查询关联对象，赋给主对象，然后返回主对象。另一种是使用嵌套查询，嵌套查询的含义为使用 join 查询，一部分列是 A 对象的属性值，另外一部分列是关联对象 B 的属性值，好处是只发一个 sql 查询，就可以把主对象和其关联对象查出来。

那么问题来了，join 查询出来 100 条记录，如何确定主对象是 5 个，而不是 100 个？其去重复的原理是`<resultMap>`标签内的`<id>`子标签，指定了唯一确定一条记录的 id 列，MyBatis 根据列值来完成 100 条记录的去重复功能，`<id>`可以有多个，代表了联合主键的语意。

同样主对象的关联对象，也是根据这个原理去重复的，尽管一般情况下，只有主对象会有重复记录，关联对象一般不会重复。

举例：下面 join 查询出来 6 条记录，一、二列是 Teacher 对象列，第三列为 Student 对象列，MyBatis 去重复处理后，结果为 1 个老师 6 个学生，而不是 6 个老师 6 个学生。

t_id t_name s_id

| 1 | teacher | 38 | | 1 | teacher | 39 | | 1 | teacher | 40 | | 1 | teacher | 41 | | 1 | teacher | 42 | | 1 | teacher | 43 |

10、MyBatis 是否支持延迟加载？如果支持，它的实现原理是什么？

注：我出的。

答：MyBatis 仅支持 association 关联对象和 collection 关联集合对象的延迟加载，association 指的就是一对一，collection 指的就是一对多查询。在 MyBatis 配置文件中，可以配置是否启用延迟加载 `lazyLoadingEnabled=true|false。`

它的原理是，使用 `CGLIB` 创建目标对象的代理对象，当调用目标方法时，进入拦截器方法，比如调用 `a.getB().getName()`，拦截器 `invoke()`方法发现 `a.getB()`是 null 值，那么就会单独发送事先保存好的查询关联 B 对象的 sql，把 B 查询上来，然后调用 a.setB(b)，于是 a 的对象 b 属性就有值了，接着完成 `a.getB().getName()`方法的调用。这就是延迟加载的基本原理。

当然了，不光是 MyBatis，几乎所有的包括 Hibernate，支持延迟加载的原理都是一样的。

11、MyBatis 的 Xml 映射文件中，不同的 Xml 映射文件，id 是否可以重复?

注：我出的。

答：不同的 Xml 映射文件，如果配置了 namespace，那么 id 可以重复；如果没有配置 namespace，那么 id 不能重复；毕竟 namespace 不是必须的，只是最佳实践而已。

原因就是 namespace+id 是作为 `Map<String, MappedStatement>`的 key 使用的，如果没有 namespace，就剩下 id，那么，id 重复会导致数据互相覆盖。有了 namespace，自然 id 就可以重复，namespace 不同，namespace+id 自然也就不同。

12、MyBatis 中如何执行批处理?

注：我出的。

答：使用 BatchExecutor 完成批处理。

13、MyBatis 都有哪些 Executor 执行器？它们之间的区别是什么?

注：我出的

答：MyBatis 有三种基本的 Executor 执行器，`SimpleExecutor`、`ReuseExecutor`、`BatchExecutor`。

`SimpleExecutor`：每执行一次 update 或 select，就开启一个 Statement 对象，用完立刻关闭 Statement 对象。

``ReuseExecutor`：执行 update 或 select，以 sql 作为 key 查找 Statement 对象，存在就使用，不存在就创建，用完后，不关闭 Statement 对象，而是放置于 Map<String, Statement>内，供下一次使用。简言之，就是重复使用 Statement 对象。

`BatchExecutor`：执行 update（没有 select，JDBC 批处理不支持 select），将所有 sql 都添加到批处理中（addBatch()），等待统一执行（executeBatch()），它缓存了多个 Statement 对象，每个 Statement 对象都是 addBatch()完毕后，等待逐一执行 executeBatch()批处理。与 JDBC 批处理相同。

作用范围：Executor 的这些特点，都严格限制在 SqlSession 生命周期范围内。

14、MyBatis 中如何指定使用哪一种 Executor 执行器?

注：我出的

答：在 MyBatis 配置文件中，可以指定默认的 ExecutorType 执行器类型，也可以手动给 `DefaultSqlSessionFactory` 的创建 SqlSession 的方法传递 ExecutorType 类型参数。

15、MyBatis 是否可以映射 Enum 枚举类?

注：我出的

答：MyBatis 可以映射枚举类，不单可以映射枚举类，MyBatis 可以映射任何对象到表的一列上。映射方式为自定义一个 `TypeHandler`，实现 `TypeHandler` 的 `setParameter()`和 `getResult()`接口方法。`TypeHandler` 有两个作用，一是完成从 javaType 至 jdbcType 的转换，二是完成 jdbcType 至 javaType 的转换，体现为 `setParameter()`和 `getResult()`两个方法，分别代表设置 sql 问号占位符参数和获取列查询结果。

16、MyBatis 映射文件中，如果 A 标签通过 include 引用了 B 标签的内容，请问，B 标签能否定义在 A 标签的后面，还是说必须定义在 A 标签的前面?

注：我出的

答：虽然 MyBatis 解析 Xml 映射文件是按照顺序解析的，但是，被引用的 B 标签依然可以定义在任何地方，MyBatis 都可以正确识别。

原理是，MyBatis 解析 A 标签，发现 A 标签引用了 B 标签，但是 B 标签尚未解析到，尚不存在，此时，MyBatis 会将 A 标签标记为未解析状态，然后继续解析余下的标签，包含 B 标签，待所有标签解析完毕，MyBatis 会重新解析那些被标记为未解析的标签，此时再解析 A 标签时，B 标签已经存在，A 标签也就可以正常解析完成了。

17、简述 MyBatis 的 Xml 映射文件和 MyBatis 内部数据结构之间的映射关系?

注：我出的

答：MyBatis 将所有 Xml 配置信息都封装到 All-In-One 重量级对象 Configuration 内部。在 Xml 映射文件中，`<parameterMap>`标签会被解析为 `ParameterMap` 对象，其每个子元素会被解析为 ParameterMapping 对象。`<resultMap>`标签会被解析为 `ResultMap` 对象，其每个子元素会被解析为 `ResultMapping` 对象。每一个 `<select>`、`<insert>`、`<update>`、`<delete>`标签均会被解析为 `MappedStatement` 对象，标签内的 sql 会被解析为 BoundSql 对象。

18、为什么说 MyBatis 是半自动 ORM 映射工具？它与全自动的区别在哪里?

注：我出的

答：Hibernate 属于全自动 ORM 映射工具，使用 Hibernate 查询关联对象或者关联集合对象时，可以根据对象关系模型直接获取，所以它是全自动的。而 MyBatis 在查询关联对象或关联集合对象时，需要手动编写 sql 来完成，所以，称之为半自动 ORM 映射工具。

面试题看似都很简单，但是想要能正确回答上来，必定是研究过源码且深入的人，而不是仅会使用的人或者用的很熟的人，以上所有面试题及其答案所涉及的内容，在我的 MyBatis 系列博客中都有详细讲解和原理分析。