# 大厂高级Java面试题合集——Mybatis

**MyBatis 面试题**

**1、什么是 Mybatis？**

1、Mybatis 是一个半 ORM（对象关系映射）框架，它内部封装了 JDBC，开发时只需要关注 SQL 语句本身，不需要花费精力去处理加载驱动、创建连接、创建statement 等繁杂的过程。程序员直接编写原生态 sql，可以严格控制 sql 执行性能，灵活度高。

2、MyBatis 可以使用 XML 或注解来配置和映射原生信息，将 POJO 映射成数据库中的记录，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。

3、通过 xml 文件或注解的方式将要执行的各种 statement 配置起来，并通过java 对象和 statement 中 sql 的动态参数进行映射生成最终执行的 sql 语句，最后由 mybatis 框架执行 sql 并将结果映射为 java 对象并返回。（从执行 sql 到返回 result 的过程）。

**2、Mybaits 的优点：**

1、基于 SQL 语句编程，相当灵活，不会对应用程序或者数据库的现有设计造成任何影响，SQL 写在 XML 里，解除 sql 与程序代码的耦合，便于统一管理；提供 XML 标签，支持编写动态 SQL 语句，并可重用。

2、与 JDBC 相比，减少了 50%以上的代码量，消除了 JDBC 大量冗余的代码，不需要手动开关连接；

3、很好的与各种数据库兼容（因为 MyBatis 使用 JDBC 来连接数据库，所以只要 JDBC 支持的数据库 MyBatis 都支持）。

4、能够与 Spring 很好的集成；

5、提供映射标签，支持对象与数据库的 ORM 字段关系映射；提供对象关系映射标签，支持对象关系组件维护。

**3、MyBatis 框架的缺点：**

1、SQL 语句的编写工作量较大，尤其当字段多、关联表多时，对开发人员编写 SQL 语句的功底有一定要求。

2、SQL 语句依赖于数据库，导致数据库移植性差，不能随意更换数据库。

**4、MyBatis 框架适用场合：**

1、MyBatis 专注于 SQL 本身，是一个足够灵活的 DAO 层解决方案。

2、对性能的要求很高，或者需求变化较多的项目，如互联网项目，MyBatis 将是不错的选择。

**5、MyBatis 与 Hibernate 有哪些不同？**

1、Mybatis 和 hibernate 不同，它不完全是一个 ORM 框架，因为 MyBatis 需要程序员自己编写 Sql 语句。

2、Mybatis 直接编写原生态 sql，可以严格控制 sql 执行性能，灵活度高，非常适合对关系数据模型要求不高的软件开发，因为这类软件需求变化频繁，一但需求变化要求迅速输出成果。但是灵活的前提是 mybatis 无法做到数据库无关性，如果需要实现支持多种数据库的软件，则需要自定义多套 sql 映射文件，工作量大。

3、Hibernate 对象/关系映射能力强，数据库无关性好，对于关系模型要求高的软件，如果用 hibernate 开发可以节省很多代码，提高效率。

**6、#{}和${}的区别是什么？**

#{}是预编译处理，${}是字符串替换。

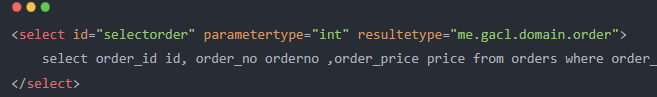
Mybatis 在处理#{}时，会将 sql 中的#{}替换为?号，调用 PreparedStatement 的 set 方法来赋值；

Mybatis 在处理${}时，就是把${}替换成变量的值。

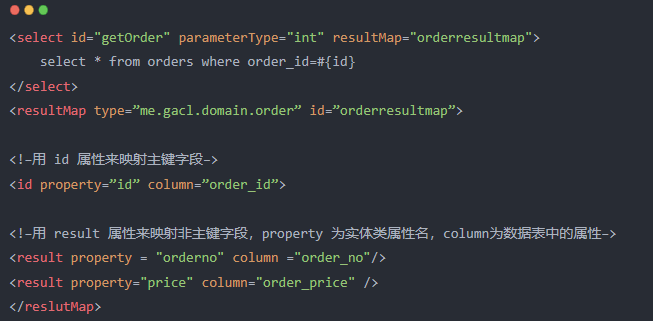
使用#{}可以有效的防止 SQL 注入，提高系统安全性。

**7、当实体类中的属性名和表中的字段名不一样 ，怎么办 ？**

第 1 种： 通过在查询的 sql 语句中定义字段名的别名，让字段名的别名和实体类 的属性名一致。

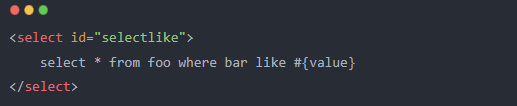
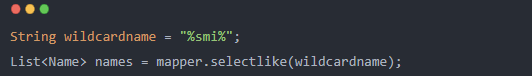


第 2 种： 通过来映射字段名和实体类属性名的一一对应的关系。

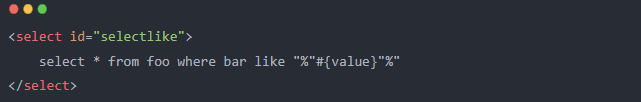
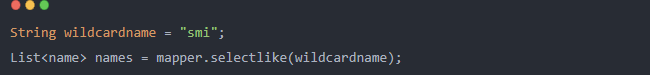


**8、 模糊查询 like 语句该怎么写?**

第 1 种：在 Java 代码中添加 sql 通配符。



第 2 种：在 sql 语句中拼接通配符，会引起 sql 注入



**9、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？**

Dao 接口即 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 Mapper 的 Statement 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 MapperStatement。在 Mybatis 中，每一个<select>、<insert>、<update>、<delete>标签，都会被解析为一个 MapperStatement 对象。

举例：com.mybatis3.mappers.StudentDao.findStudentById，可以唯一找到 namespace 为 com.mybatis3.mappers.StudentDao 下面 id 为 findStudentById 的 MapperStatement。

Mapper 接口里的方法，是不能重载的，因为是使用 全限名+方法名 的保存和寻找策略。Mapper 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 Mapper 接口生成代理对象 proxy，代理对象会拦截接口方法，转而执行 MapperStatement 所代表的 sql，然后将 sql 执行结果返回。

**10、Mybatis 是如何进行分页的？分页插件的原理是什么？**

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。

**11、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？**

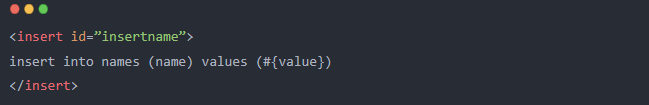
第一种是使用标签，逐一定义数据库列名和对象属性名之间的映射关系。

第二种是使用 sql 列的别名功能，将列的别名书写为对象属性名。

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。

**12、如何执行批量插入?**

首先,创建一个简单的 insert 语句:



然后在 java 代码中像下面这样执行批处理插入:

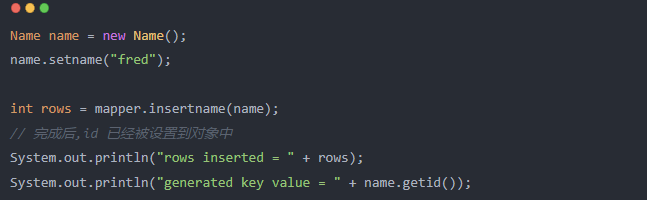
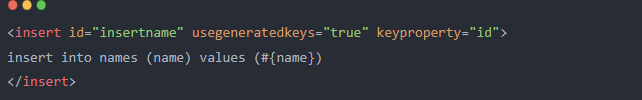


**13、如何获取自动生成的(主)键值?**

insert 方法总是返回一个 int 值 ，这个值代表的是插入的行数。

如果采用自增长策略，自动生成的键值在 insert 方法执行完后可以被设置到传入的参数对象中。

示例：

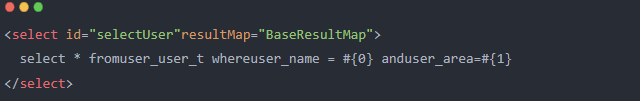


**14、在 mapper 中如何传递多个参数?**

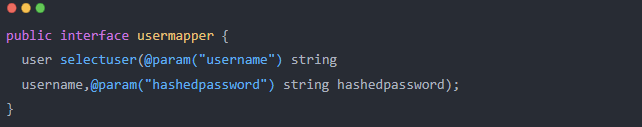
1、第一种：

DAO 层的函数

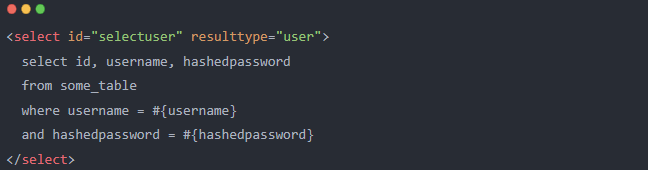


对应的 xml,#{0}代表接收的是 dao 层中的第一个参数，#{1}代表 dao 层中第二参数，更多参数一致往后加即可

2、第二种： 使用 @param 注解



然后,就可以在 xml 像下面这样使用(推荐封装为一个 map,作为单个参数传递给 mapper):



3、第三种：多个参数封装成 map



**15、Mybatis 动态 sql 有什么用？执行原理？有哪些动态 sql？**

Mybatis 动态 sql 可以在 Xml 映射文件内，以标签的形式编写动态 sql，执行原理是根据表达式的值 完成逻辑判断并动态拼接 sql 的功能。

Mybatis 提供了 9 种动态 sql 标签：trim | where | set | foreach | if | choose| when | otherwise | bind。

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

答：、、、、，加上动态 sql 的 9 个标签，其中为 sql 片段标签，通过标签引入 sql 片段，为不支持自增的主键生成策略标签。

**17、Mybatis 的 Xml 映射文件中，不同的 Xml 映射文件，id 是否可以重复？**

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；

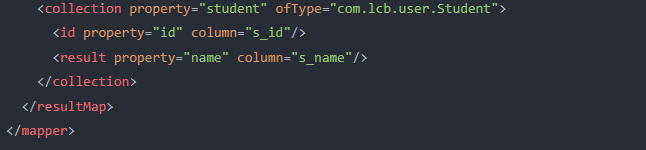
原因就是 namespace+id 是作为 Map<String, MapperStatement>的 key 使用的，如果没有 namespace，就剩下 id，那么，id 重复会导致数据互相覆盖。有了 namespace，自然 id 就可以重复，namespace 不同，namespace+id 自然也就不同。

**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？**

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。

**19、一对一、一对多的关联查询 ？**





**20、MyBatis 实现一对一有几种方式?具体怎么操作的？**

有联合查询和嵌套查询,联合查询是几个表联合查询,只查询一次, 通过在 resultMap 里面配置 association 节点配置一对一的类就可以完成；

嵌套查询是先查一个表，根据这个表里面的结果的 外键 id，去再另外一个表里面查询数据,也是通过 association 配置，但另外一个表的查询通过 select 属性配置。

**21、MyBatis 实现一对多有几种方式,怎么操作的？**

有联合查询和嵌套查询。联合查询是几个表联合查询,只查询一次,通过在resultMap 里面的 collection 节点配置一对多的类就可以完成；嵌套查询是先查一个表,根据这个表里面的结果的外键 id,去再另外一个表里面查询数据,也是通过配置collection,但另外一个表的查询通过 select 节点配置。

**22、Mybatis 是否支持延迟加载？如果支持，它的实现原理是什么？**

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。

**23、Mybatis 的一级、二级缓存:**

1）一级缓存: 基于 PerpetualCache 的 HashMap 本地缓存，其存储作用域为 Session，当 Session flush 或 close 之后，该 Session 中的所有 Cache 就将清空，默认打开一级缓存。

2）二级缓存与一级缓存其机制相同，默认也是采用 PerpetualCache，HashMap 存储，不同在于其存储作用域 Mapper(Namespace)，并且可自定义存储源，如 Ehcache。默认不打开二级缓存，要开启二级缓存，使用二级缓存属性类需要实现 Serializable 序列化接口(可用来保存对象的状态),可在它的映射文件中配置 ；

3）对于缓存数据更新机制，当某一个作用域(一级缓存 Session/二级缓存 Namespaces)的进行了 C/U/D 操作后，默认该作用域下所有 select 中的缓存将被 clear。

**24、什么是 MyBatis 的接口绑定？有哪些实现方式？**

接口绑定，就是在 MyBatis 中任意定义接口,然后把接口里面的方法和 SQL 语句绑定, 我们直接调用接口方法就可以,这样比起原来了 SqlSession 提供的方法我们可以有更加灵活的选择和设置。

接口绑定有两种实现方式,一种是通过注解绑定，就是在接口的方法上面加上@Select、@Update 等注解，里面包含 Sql 语句来绑定；另外一种就是通过 xml 里面写 SQL 来绑定, 在这种情况下,要指定 xml 映射文件里面的 namespace 必须为接口的全路径名。当 Sql 语句比较简单时候,用注解绑定, 当 SQL 语句比较复杂时候,用 xml 绑定,一般用 xml 绑定的比较多。

**25、使用 MyBatis 的 mapper 接口调用时有哪些要求？**

1、Mapper 接口方法名和 mapper.xml 中定义的每个 sql 的 id 相同；

2、Mapper 接口方法的输入参数类型和 mapper.xml 中定义的每个 sql 的 parameterType 的类型相同；

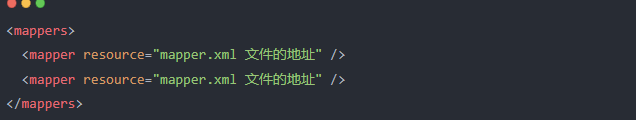
3、Mapper 接口方法的输出参数类型和 mapper.xml 中定义的每个 sql 的 resultType 的类型相同；

4、Mapper.xml 文件中的 namespace 即是 mapper 接口的类路径。

**26、Mapper 编写有哪几种方式？**

第一种：接口实现类继承 SqlSessionDaoSupport：使用此种方法需要编写 mapper 接口，mapper 接口实现类、mapper.xml 文件。

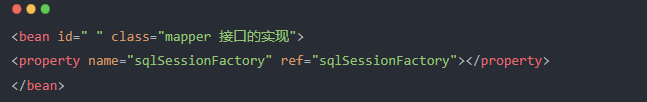
1、在 sqlMapConfig.xml 中配置 mapper.xml 的位置



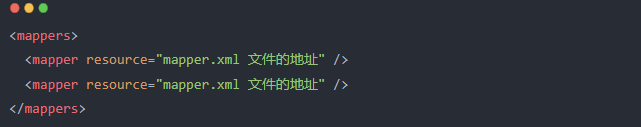
2、定义 mapper 接口

3、实现类集成 SqlSessionDaoSupport

mapper 方法中可以 this.getSqlSession()进行数据增删改查。

1. spring 配置

第二种：使用 org.mybatis.spring.mapper.MapperFactoryBean：

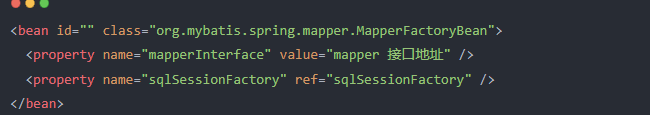
1、在 sqlMapConfig.xml 中配置 mapper.xml 的位置，如果 mapper.xml 和 mapper 接口的名称相同且在同一个目录，这里可以不用配置

2、定义 mapper 接口：

1、mapper.xml 中的 namespace 为 mapper 接口的地址

2、mapper 接口中的方法名和 mapper.xml 中的定义的 statement 的 id 保持一致

1. Spring 中定义



第三种：使用 mapper 扫描器：

1、mapper.xml 文件编写：

mapper.xml 中的 namespace 为 mapper 接口的地址；

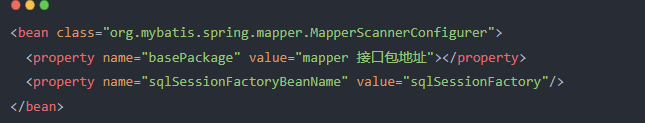
mapper 接口中的方法名和 mapper.xml 中的定义的 statement 的 id 保持一致；

如果将 mapper.xml 和 mapper 接口的名称保持一致则不用在 sqlMapConfig.xml 中进行配置。

2、定义 mapper 接口：

注意 mapper.xml 的文件名和 mapper 的接口名称保持一致，且放在同一个目录

3、配置 mapper 扫描器：



4、使用扫描器后从 spring 容器中获取 mapper 的实现对象。

**27、简述 Mybatis 的插件运行原理，以及如何编写一个插件。**

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。

编写插件：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。