**MyBatis是什么？**

MyBatis是一个半 ORM（对象关系映射）框架，可以使用XML 或注解来映射POJO为数据库中的记录

**ORM是什么**

ORM就是把定义的对象映射到对应的数据库的表上，所以ORM就是代码层面对于数据库表和关系的一种抽象。

**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？**

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。

**JDBC编程有哪些不足之处，MyBatis是如何解决这些问题的？**

1、数据库链接创建、释放频繁造成系统资源浪费从而影响性能

解决：在配置文件中配置连接池，使用连接池管理数据库连接。

2、Sql写在代码中造成代码不易维护，sql改变需要改变java代码。

解决：SQL写在XML里，解除sql与程序代码的耦合

3、向sql语句传参数麻烦，因为sql语句的where条件不一定，可能多也可能少，占位符需要和参数一一对应。

解决： Mybatis自动将java对象映射至sql语句。

4、对结果集解析麻烦

解决：Mybatis自动将sql执行结果映射至java对象。

**Mybatis优缺点**

优点：

SQL写在XML里，解除sql与程序代码的耦合

支持编写动态SQL语句，并可重用

消除了JDBC大量冗余的代码

很好的与各种数据库兼容

缺点

SQL语句的编写工作量较大

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**MyBatis编程步骤是什么样的？**

1、 创建SqlSessionFactory

2、 通过SqlSessionFactory创建SqlSession

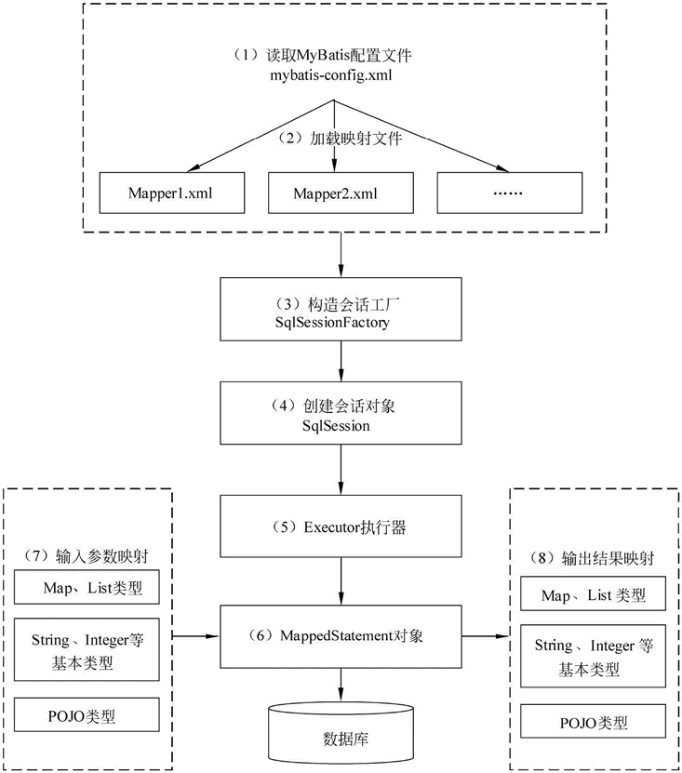
3、 通过sqlsession执行数据库操作

4、 调用session.commit()提交事务

5、 调用session.close()关闭会话

**请说说MyBatis的工作原理**

读取 MyBatis全局配置文件，加载SQL映射文件，创建会话工厂SqlSessionFactory，创建会话SqlSession，Executor执行器将MappedStatement对象进行解析，sql参数转化、动态sql拼接，生成jdbc Statement对象，JDBC执行sql，借助MappedStatement中的结果映射关系，将返回结果转化成HashMap、JavaBean等存储结构并返回。

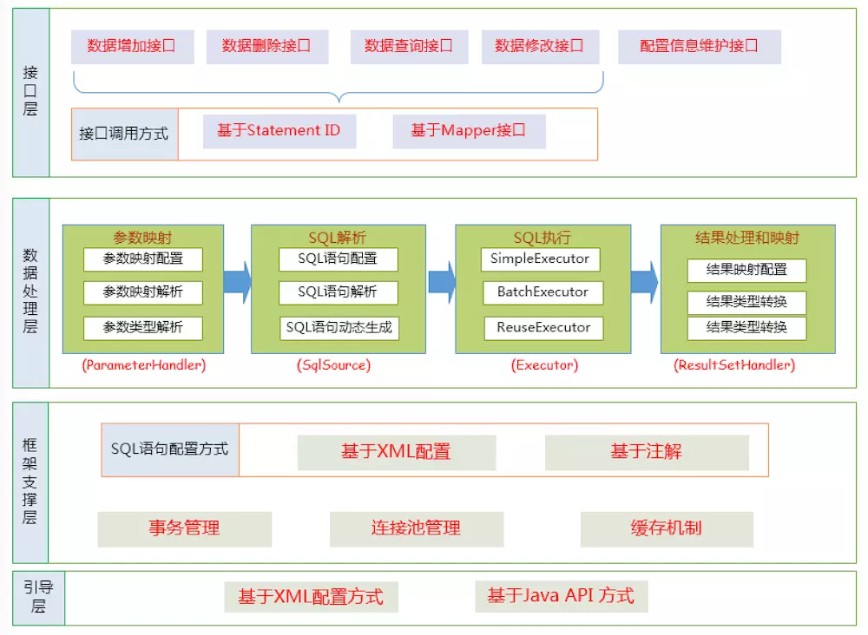


**MyBatis的功能架构是怎样的**

API接口层：提供给外部使用的接口，如mapper接口

数据处理层：负责SQL查找、SQL解析、SQL执行和执行结果映射处理等。

基础支撑层：负责连接管理、配置加载、事务管理



**MyBatis 中 #{}和 ${}的区别是什么？**

#{}是占位符，预编译处理；${}是拼接符，字符串替换，没有预编译处理。

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

处理${}时，就是把${}替换成变量的值。

#{} 对应的变量会自动加上单引号

${} 对应的变量不会加上单引号

#{} 能防止sql 注入

${} 不能防止sql 注入

**在mapper中如何传递多个参数**

@Param注解传参法，#{}里面的名称对应的是注解@Param括号里面的名称。

Map传参法, #{}里面的名称对应的是Map里面的key名称

Java Bean传参法, #{}里面的名称对应的是类里面的成员变量

**Mybatis如何执行批量操作**

foreach的主要用在构建in条件中，它可以在SQL语句中进行迭代一个集合。foreach标签的属性有item，index，collection，open，separator，close。

item表示集合中的元素的别名

index 在list、Set和数组中,index表示当前迭代的位置，在map中，index代指是元素的key

open表示该语句以什么开始，常用“(”；

separator表示分隔符，常用“,”；

close表示以什么结束，常用“)”。

collection表示迭代集合的名称，可以使用@Param注解指定

**如何获取生成的主键**

useGeneratedKeys：是否返回自动生成的主键，默认的是false

KeyProperty：把获取的主键传递给对象的某个属性。

**当实体类中的属性名和表中的字段名不一样，怎么办**

通过<resultMap>来映射字段名和实体类属性名的对应关系

<select id="getOrder" parameterType="int" resultMap="orderResultMap">

select \* from orders where order\_id=#{id}

</select>

<resultMap type="com.jourwon.pojo.Order" id="orderResultMap">

<id property="id" column="order\_id">

<result property ="orderno" column ="order\_no"/>

<result property="price" column="order\_price" />

</reslutMap>

**Mapper 编写有哪几种方式？**

…

**什么是MyBatis的接口绑定？有哪些实现方式？**

接口绑定，把接口里面的方法和SQL语句绑定，直接调用接口方法就可以。

接口绑定有两种实现方式

通过注解绑定，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；

通过xml里面写SQL来绑定，要指定xml映射文件里面的namespace为接口的全路径名。

**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接口方法的返回值类型和mapper.xml中定义的每个sql的resultType的类型相同。

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

**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**

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，所以Dao接口里的方法，是不能重载的。

工作原理就是：使用JDK动态代理为Dao接口生成代理对象proxy，代理对象proxy会拦截接口方法，通过接口全限名+方法名组成key的方式，查找XML映射文件中的MapperStatement，返回执行SQL后的结果。转而执行MappedStatement所代表的sql (MappedStatement包括了参数映射配置、动态SQL语句、结果映射配置，对应着<select | update | delete | insert>标签项)

（通过JDK动态代理接口，当调用接口方法时，代理拦截调用，并通过接口全限名+方法名组成key的方式，查找XML映射文件中的MapperStatement，返回执行SQL后的结果）

**简述Mybatis的Xml映射文件和Mybatis内部数据结构之间的映射关系？**

在Xml映射文件中，<resultMap>标签被解析为ResultMap对象，每个子元素会被解析为ResultMapping对象。<select>、<insert>、<update>、<delete>标签被解析为MappedStatement对象，标签内的sql会被解析为BoundSql对象。

**MyBatis实现一对一，(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association，(collection)节点配置一对一，(一对多)的类就可以完成

嵌套查询是先查一个表，根据这个表的结果的外键id，再去另外一个表里面查询数据，另外一个表的查询通过association (collection)节点的select属性。

association适用于一对一和多对一，collection适用于多对多和一对多

**MyBatis一对一有几种方式**

在resultMap标签中使用 association 标签

**MyBatis一对多有几种方式**

在resultMap标签中使用collection 标签。

分步查询，第二次查询在collection的select属性上，通过column属性传入参数，property属性指定对应的实体类中的集合字段

联合查询，几个表联合查询，只查询一次，collection配置一对多的类就可以完成

**MyBatis多对一有几种方式**

分步查询，第二次查询在association的select属性上，通过column属性传入参数，property属性指定对应的实体类中的字段

联合查询，只查询一次，association配置多对一的类就可以完成

**Mybatis动态sql是做什么的？都有哪些动态sql？**

主要是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。

**动态sql原理**

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

**ResultType和ResultMap的区别**

通过<resultMap>来映射字段名和实体类属性名的对应关系，而resultType是表示返回类型的

**Mybatis 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？**

<resultMap>、<parameterMap>、<sql>、<include>、

其中 <sql> 为 sql 片段标签，通过<include> 标签引入 sql 片段

**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；

**MyBatis有哪些Executor执行器，他们之间的区别是什么**

SimpleExecutor：每执行一次update或者select就开启一个Statement对象，用完就立即关闭 （Statement对象是jdbc的）

ReuseExecutor： 使用一个Map把Statement都缓存起来，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内。简言之，使用一个Map把Statement都缓存起来，重复使用Statement对象。

BatchExecutor：批处理执行器，用于将多个SQL一次性输出到数据库

**模糊查询like语句该怎么写**

"%"#{question}"%" 注意：因为#{…}解析成sql语句时候，会在变量外侧自动加单引号’ '，所以这里 % 需要使用双引号" "，不能使用单引号 ’ '，不然会查不到任何结果。

CONCAT(’%’,#{question},’%’)函数

**MyBatis 有几种分页方式？分页插件的原理?**

分页方式：逻辑分页和物理分页。

逻辑分页：使用MyBatis自带的RowBounds进行分页，一次性查询很多数据，然后再进行检索。

物理分页：手写SQL分页或使用分页插件PageHelper，去数据库查询指定条数的分页数据。

拦截待执行的sql，然后重写sql，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

**Mybatis的一级、二级缓存**

一级缓存是在同一个 SqlSession 中多次执行同一个查询，除第一次走数据库，剩下的都走缓存

二级缓存指的是SqlSessionFactory对象的缓存。由同一个SqlSessionFactory对象创建的SqlSession共享其缓存 (开启MyBatis的二级缓存 cache-enabled: true)

三级缓存是整合Ehcache，Redis等工具来实现的

**Mybatis分页**

使用sql中的limit手写分页，

Pagehelper插件分页。

拦截器分页(物理分页)，通过拦截器给sql语句末尾加limit语句来查询