Mybatis整体架构

接口层 sqlsession

核心处理层 配置解析 参数映射 sql解析

Sql执行 结果集映射 插件

基础支持层 数据源 事务管理 缓存 binding模块

反射 类型转换 日志 资源加载 解析器

**日志模块**：mybatis没有提供日子的实现类，需要接入第三方的日志组件，但第三方日志组件都有各自的log级别，且各不相同，mybatis统一提供了trace，debug，warn，error四个级别

Mybatis会自动扫描日志实现，并且第三方日志插件加载优先级如下：slf4j->commonsLoging->Log4j2->Log4j->jdkLog

**数据源模块**：

常见的数据源组件都实现了javax.sql.DataSource接口

mybatis不但要集成第三方的数据源组件，自身也提供了数据源的实现

数据源初始化过程参数较多，比较复杂

PooledConnection:使用动态代理封装真正的数据库连接对象

PoolState:用来管理PooledConnection对象状态的组件，通过两个list分别管理空闲状态的连接资源封装连接池的状态信息

和活跃状态的连接资源

PooledDataSource：一个简单，同步的，线程安全的数据库连接池

（数据连接池）

**缓存模块**：

Mybatis缓存的实现是基于Map的，从缓存里面读写数据是缓存模块的核心基础功能；

除核心功能之外，有很多额外的附加功能，如：防止缓存击穿，添加缓存情况策略，序列化功能，日志能力，定时清空能力

附加功能可以以任意的组合附加到核心基础功能之上

Cache

PerpetualCache

BlockingCache

Mybatis缓存的key:CacheKey(判断是否从缓存中取数据)，mybatis中设计到动态sql的原因，缓存向的key不能仅仅通过一个String来表示，所以通过CacheKey来封装key值，cachekey可以封装多个影响缓存项的因素，判断两个cachekey是否相同关键是比较两个对象因素包括下面的

sql的id（包含命名空间）+分页信息+查询的sql语句+查询的参数

查询

创建对象，属性设置

mybatis中的反射

ObjectFactory：mybatis每次创建结果对象的新实例时，它都会使用对象工厂去构件pojo

ReflectorFactory：创建reflector（reflector封装的一个类的元信息，包括方法，属性等）的工厂类，reflector是mybatis反射模块的基础，每个reflector对象都对应一个类，在其中缓存了反射操作所需要的类元信息

ObjectWrapper：对对象的包装类，抽象了对象的属性信息，它定义了一些列查询对象属性信息的方法，以及更新属性的方法（也就是给对象赋值）

ObjectWrapperFactory：ObjectWrapper的工厂类，用于创建ObjectWrapper

MetaObject:封装了对象元信息，包装了mybatis中五个核心的反射类，也提供给外部使用的反射工具类，可以利用它读取或者修改对象的属性信息

ObjectFactory—》创建对象

ReflectorFactory—》获取类中的信息（方法，属性等）

ObjectWrapper—》给对象赋值，

Mybatis核心流程三大阶段

1. 读取xml配置文件和注解中的配置信息，创建配置对象，并完成各个模块的初始化工作（初始化阶段）
2. 封装ibatis的编程模型，使用mapper接口开发的初始化工作（代理阶段）
3. 通过sqlsession完成sql的解析，参数的映射，sql的执行，结果的解析过程（数据读写阶段）