1. Mybatis 动态 SQL ，可以让我们在 XML 映射文件内，以 XML 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：<if/>、<choose/>、<when/>、<otherwise/>、<trim/>、<when/>、<set/>、<foreach/>、<bind/>。

其执行原理为，使用 OGNL 的表达式，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。

1. 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。

#### 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。

1. MyBatis的一级缓存是基于SqlSession级别的，也就是说某个SqlSession进行某个查询操作后会将该结果暂时缓存起来，而后在所有的SqlSession没有对该表进行插入、修改、删除操作的情况下，当这个SqlSession再次发起此查询时SqlSession不会去数据库执行查询操作，而是直接从缓存拿出上次查询的结果。不同的SqlSession之间缓存的数据互不影响。

MyBatis的二级缓存是基于Mapper级别的，也就是说多个SqlSession去使用某个Mapper的查询语句时，得到的缓存数据是可共用的。

1. mybatis可以编写针对Executor、StatementHandler、ParameterHandler、ResultSetHandler四个接口的插件，mybatis使用JDK的动态代理为需要拦截的接口生成代理对象，然后实现接口的拦截方法，所以当执行需要拦截的接口方法时，会进入拦截方法（AOP面向切面编程的思想）

1.编写Intercepror接口的实现类

2.设置插件的签名，告诉mybatis拦截哪个对象的哪个方法

3.最后将插件注册到全局配置文件中