1. Mybatis 动态 sql 是做什么的？都有哪些动态 sql？简述一下动态 sql 的执行原理

答：在以配置文件的形式编写 SQL 语句的时候，可以使用动态 SQL，完成 SQL 语句的动态拼装。

动态 SQL 标签有：<if>,<where><choose><when><otherwise><foreach><sql><include><set><trim>

执行原理：在解析 sql 配置文件 mapper.xml 的时候进行解析动态 SQL 标签，根据关键标签封装成对应的 handler 处理对象，封装成 sqlSource 对象存在 mappedStatement 中。在调用的时候，获取 sqlSource 执行相应的标签 handler。

2.Mybatis 是否支持延迟加载？如果支持，它的实现原理是什么？

答：支持延迟加载，但是在编写 SQL 配置文件的时候如果用到链接查询是无法实现延迟的，因为在查询时只执行了一次 SQL 语句。如图：

```
<resultMap id="orderMap" type="com.lagou.pojo.Order">
    <result property="id" column="id"></result>
    <result property="orderTime" column="orderTime"></result>
    <result property="total" column="total"></result>

    <association property="user" javaType="com.lagou.pojo.User" >
        <result property="id" column="uid"></result>
        <result property="username" column="username"></result>
    </association>
</resultMap>
```

如下图所示，可以用延迟加载：

```
<resultMap id="orderMap" type="com.lagou.pojo.Order">
    <result property="id" column="id"></result>
    <result property="orderTime" column="orderTime"></result>
    <result property="total" column="total"></result>

    <association property="user" javaType="com.lagou.pojo.User" column="uid" select="selectUserById">
    </association>
</resultMap>

<select id="selectUserById" resultType="com.lagou.pojo.User" parameterMap="int">
    select * from user where id = #{id}
</select>
```

在 Mybatis 配置文件中，可以配置是否启用延迟加载 lazyLoadingEnabled=true|false。实现原理是：生成代理对象,对象方法调用时执行查询语句。

下·

3．Mybatis 都有哪些 Executor 执行器？它们之间的区别是什么？

答：mybatis 有三种 executor 执行器，分别为：

Simpleexecutor：在每执行一次 update 或 select，就开启一个 statement 对象，用完后就关闭。

Reuseexecutor：在执行 update 或 select 时以 sql 作为 key 去查找 statement，有就直接使用，没有就创建，使用完毕后不关闭，放入 Map<String,Statement>中，供下次使用。重复使用 statement。

Batchexecutor：执行 update（jdbc 批处理不支持 select），会把所有 sql 添加到批处理中 addbatch（）；等待统一批处理 executorbatch（）；它缓存了·多个 statement，每一个 statement 都是 addbatch（），后等待进行 executorbatch（）批处理。

4、简述下 Mybatis 的一级、二级缓存（分别从存储结构、范围、失效场景。三个方面来作答）？

答：一级缓存：MyBatis 是默认开启一级缓存的。存储结构是 HashMap，

保存的是 pojo 对象。作用域是 SqlSession，在同一个 SqlSession 中，相同的 Sql 查询的时候，第一次查询的时候，就会从缓存中取，如果发现没有数据，那么就从数据库查询出来，并且缓存到 HashMap 中，如果下次还是相同的查询，就直接从缓存中查询，就不在去查询数据库。进行增删改的操作的时候，缓存将会失效。

二级缓存：MyBatis 的二级缓存需要手动配置开启。存储结构是 HashMap，保存的是数据而非 pojo 对象。二级缓存是 mapper 级别的缓存，多个 SqlSession 去操作同一个 mapper 的 sql 语句，多个 SqlSession 可以共用二级缓存。第一次调用 mapper 下的 sql 的时候去查询信息，查询到的信息会存放到该 mapper 对应的二级缓存区域，第二次调用 namespace 下的 mapper 映射文件中，相同的 SQL 去查询，回去对应的二级缓存内取结果。如果在相同的 namespace 下的 mapper 映射文件中增删改，二级缓存失效。

5．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。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。最后在配置文件中配置编写的插件。