通过intercept 拦截器 定义数据库中createtime 之类的自动插入相关信息。

根据选择哪个Statement:对需要传入的参数发送到sql语句里。 然后得到的返回类型再执行typehandler的处理，最终实现 插入createtime的操作。此时数据库createtime字段类型为BIGINT 而通过JAVA插入的是LOCALDATETIME 所以需要通过typehandler进行类型转换。

1.MapperProxy 调用方法

MapperProxy 是一个代理类，实现了Mapper接口

代码上是调用接口的方法，实际上是调用了MapperProxy的方法

在MapperProxy内，会有拦截器链，依次调用

拦截器的方法链完成后，会最后进入Executer 对应的方法执行 update/query

四核8g mysql可以发生200个并发

接下来介绍 springboot 事务的传播级别

**REQUIRED(0)：**

在事务1中包含两个事务，并且这两个事务的传播属性都是REQUIRED，

当执行到事务2 的时候，若事务2发生异常，则事务1回滚到初始状态，若未发生异常，则继续执行事务3

当执行事务3的时候，同样，若事务3发生异常，事务1回滚到初始状态，若未发生异常，则事务1执行成功

SUPPORTS(1),  
MANDATORY(2),  
**REQUIRES\_NEW(3),**：

类似于 子类与父类 transaction1{ 语句A1 transaction2{ 语句B2}语句A2}

如果在语句A2前发生了异常，则只回滚transaction1的事务，而transaction2没有发生异常则执行成功。

现在在事务1中包含事务2和事务3，事务2的传播属性是REQUIRED\_NEW,事务3的传播属性是REQUIRED

事务1开始执行，当执行到事务2，若事务2发生异常，则事务2自己回滚，换句话说，事务1不会回滚到自己的初始状态，而是由事务2自己回滚到自己的初始状态。

接下来继续执行事务3，由于事务3的传播属性是REQUIRED，所以当事务3发生异常时，事务1会回滚到初始状态

原文链接：<https://blog.csdn.net/wojiaoxubo/article/details/85158445>

NOT\_SUPPORTED(4),  
NEVER(5),  
NESTED(6);

Springboot的事务必须从service切入。事务不能再同一个类多个方法使用不同的事务。

Required：在任何一个方法中发生了异常，所有连续的执行操作都会回滚到最初的状态

Required new： 在任何一个方法中发生了异常，则在该方法中回滚该方法发生的操作，其他连续的操作如果没有发生异常都不会回滚。

加密算法：

常见的对称加密算法：DES,3DES,AE5 非对称有RSA,OSA等，数列算法主要有SHA-1,MD5

通过登录的时候加盐（在密码后面加多一个salt字段，类似版本号的方法。）可缓解暴力破解的方法（暴力破解意思是一个一个的试）

账号密码不能使用MD5加密

账号密码最好是不用可逆的加密

DigestUtil工具类

Sha签名加密

RSA：不想让其他人看到

公钥私钥区别：公钥可以每个人都拥有，私钥只有唯一一个。

接收信息的人拿私钥，发信息的人拿公钥

面试中回答用户登录的解决方案：

用了很复杂的加密算法，sha512，

所需依赖：[Apache Commons Codec](https://mvnrepository.com/artifact/commons-codec/commons-codec)

这个工具包很牛逼：GooleGuawa.

并发：同时发生同一件事情

在controller中 HttpServletRequest/HttpServletResponse 是根据类型来的，创建时机：在一次请求的时候会创建对应对象，在这一次请求只有这两个。

Httpsession是存入在服务器中。

是怎么存入服务器？

Session类似hashmap结构， 用sessionid存放，以sessionid与浏览器保持一个联系状态存放在cookie中，因此保持会话的状态。

解决线程安全：让变量都跑到线程里变成局部变量，就没有这个漏洞

ThreadLocal 上下文 == ThreadId 作为Key Thread的值 ==map的值

一个请求对应一个线程，不是一个线程对应一个请求

ThreadMap<String,Object >.put(threadId,object);

ThreadLocal{

String threadId;

Set(object){

threadMap.put(this.threadId,object);

}

}