《软件安全》实验报告

姓名：王众　　学号：2313211

实验名称：

SQL盲注

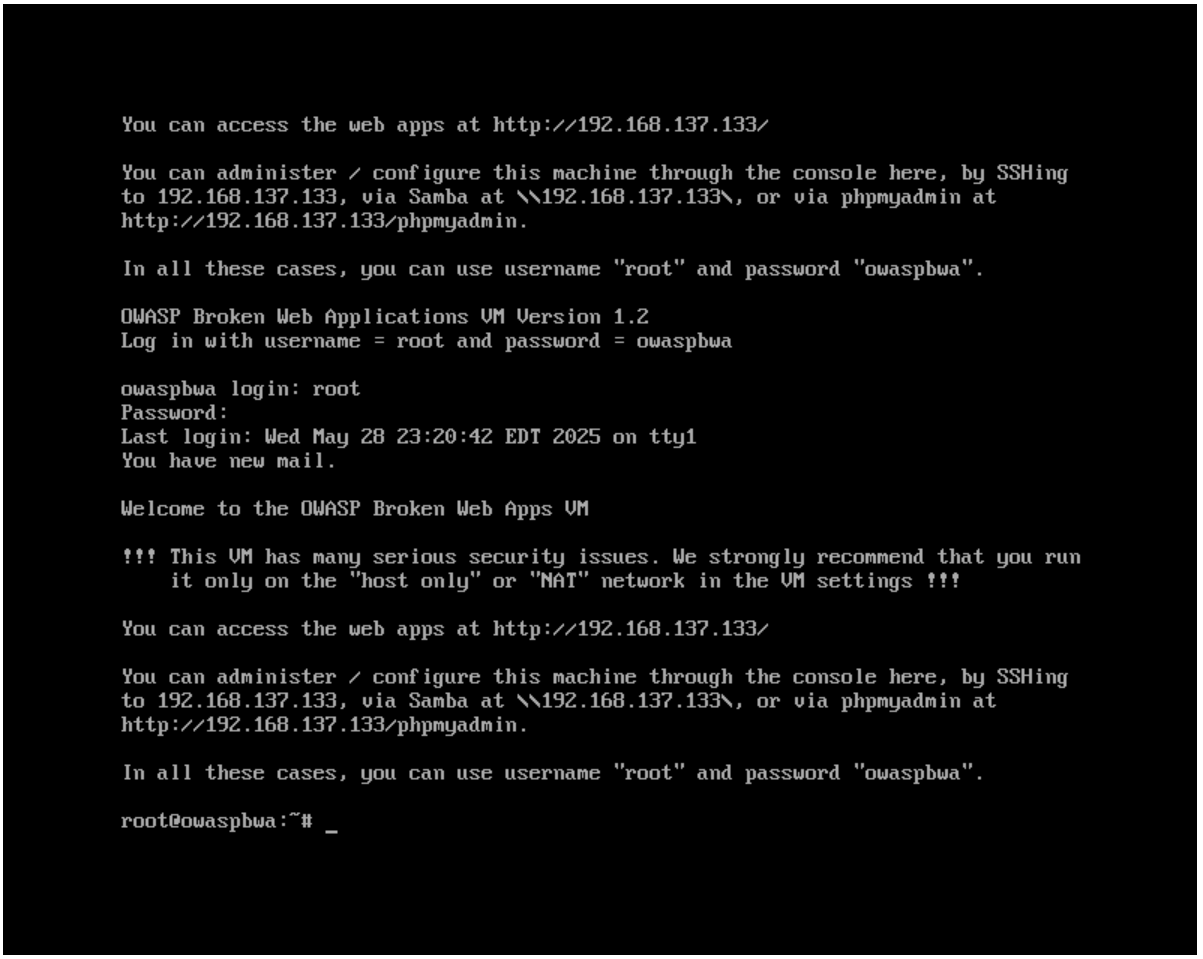
实验要求：

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。

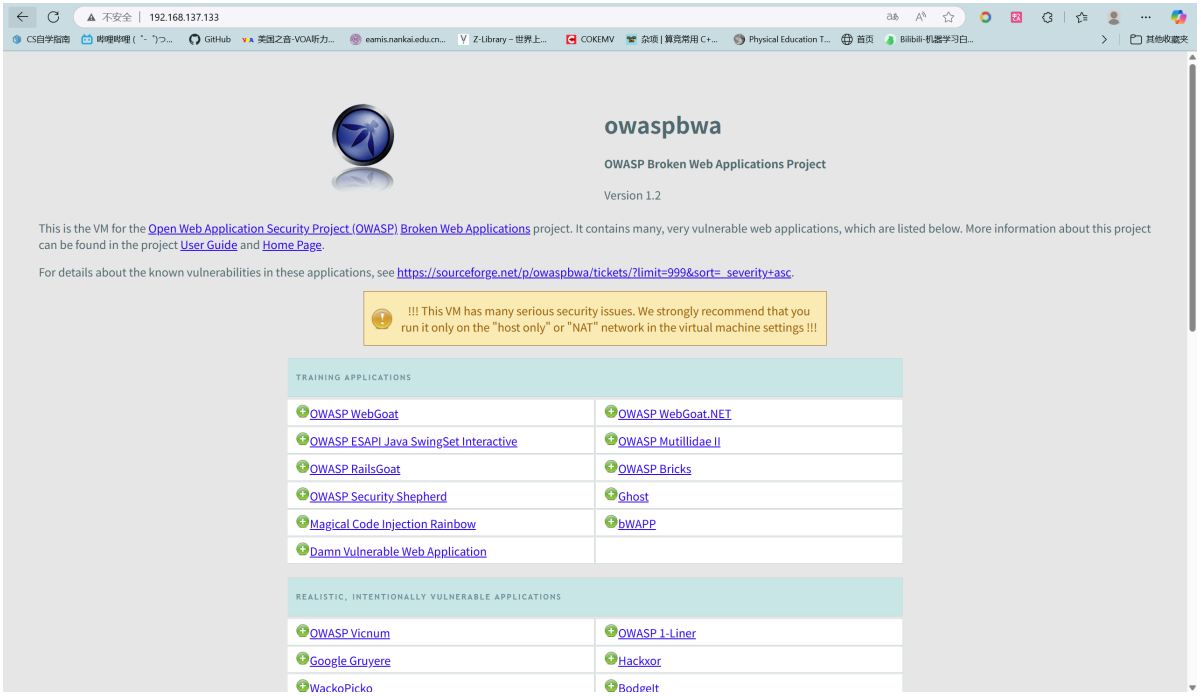
实验过程：

环境配置

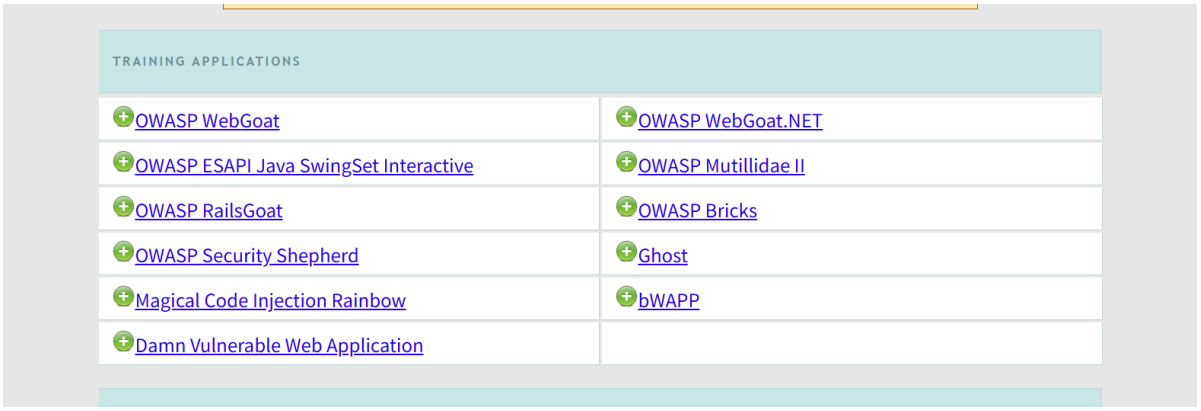
首先，解压附带工具中的软件，并在$Vmware$中运行。



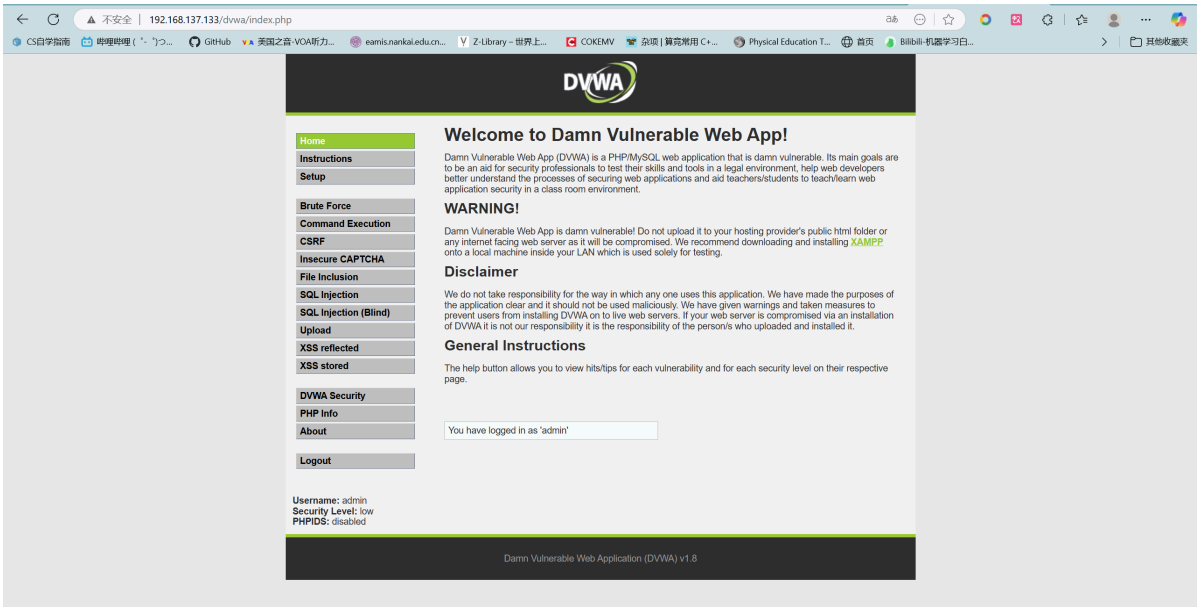
接着我们在本地的地址中打开：



然后我们进行登录，选择`DVWA`（`Damn Vulnerable Web Application`）即可。账号和密码均为`admin`。



登录完之后进入到以下界面：



在左边选择`DVWA Security`，将等级设置为`low`，我们发现初始值就是`low`，那就说明不需要进行修改了。

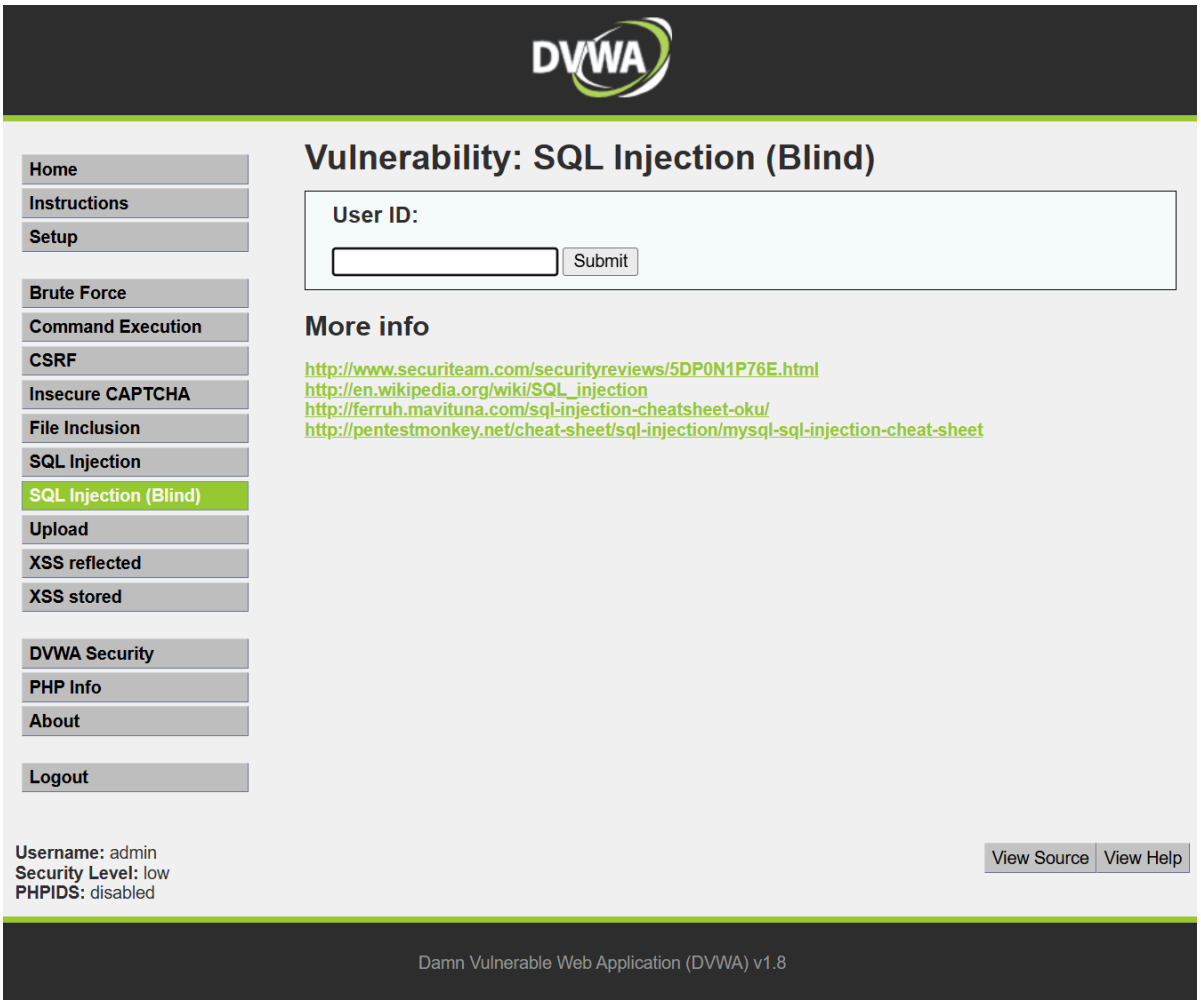
以上就是本次实验的环境配置过程，下面我们就马上开始本次实验。

实验复现

1 初步了解

接下来，我们通过 `DVWA` 中提供的注入案例，进行手工盲注，目标是推测出数据库、表和字段。

在`DVWA`界面左侧选择 `SQL Injection(Blind)`，界面如下：

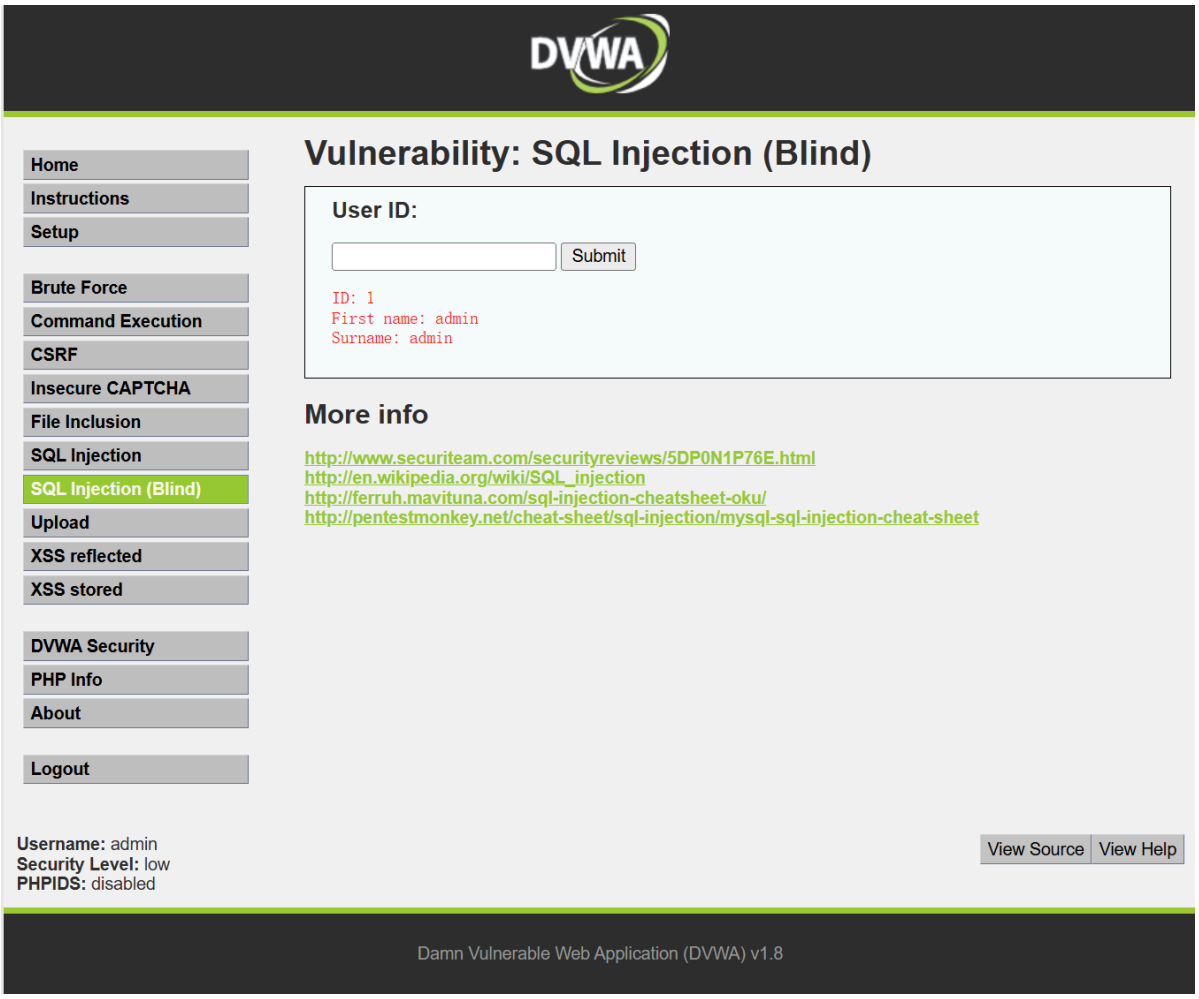


这就是本次的`SQL`盲注实验的输入端口。我们需要通过手动输入字符串，向系统“骗取”一些他所知道的东西，然后再通过我们骗取到的知识去猜测数据库的构成以及其中的信息。

机器人只会回答是或者不是，所以我们需要询问它这样的问题，例如“数据库名字的第一个字母是不是d啊？”，通过这种机械的询问，最终获得你想要的数据。接下来，我们逐步展开推测过程。

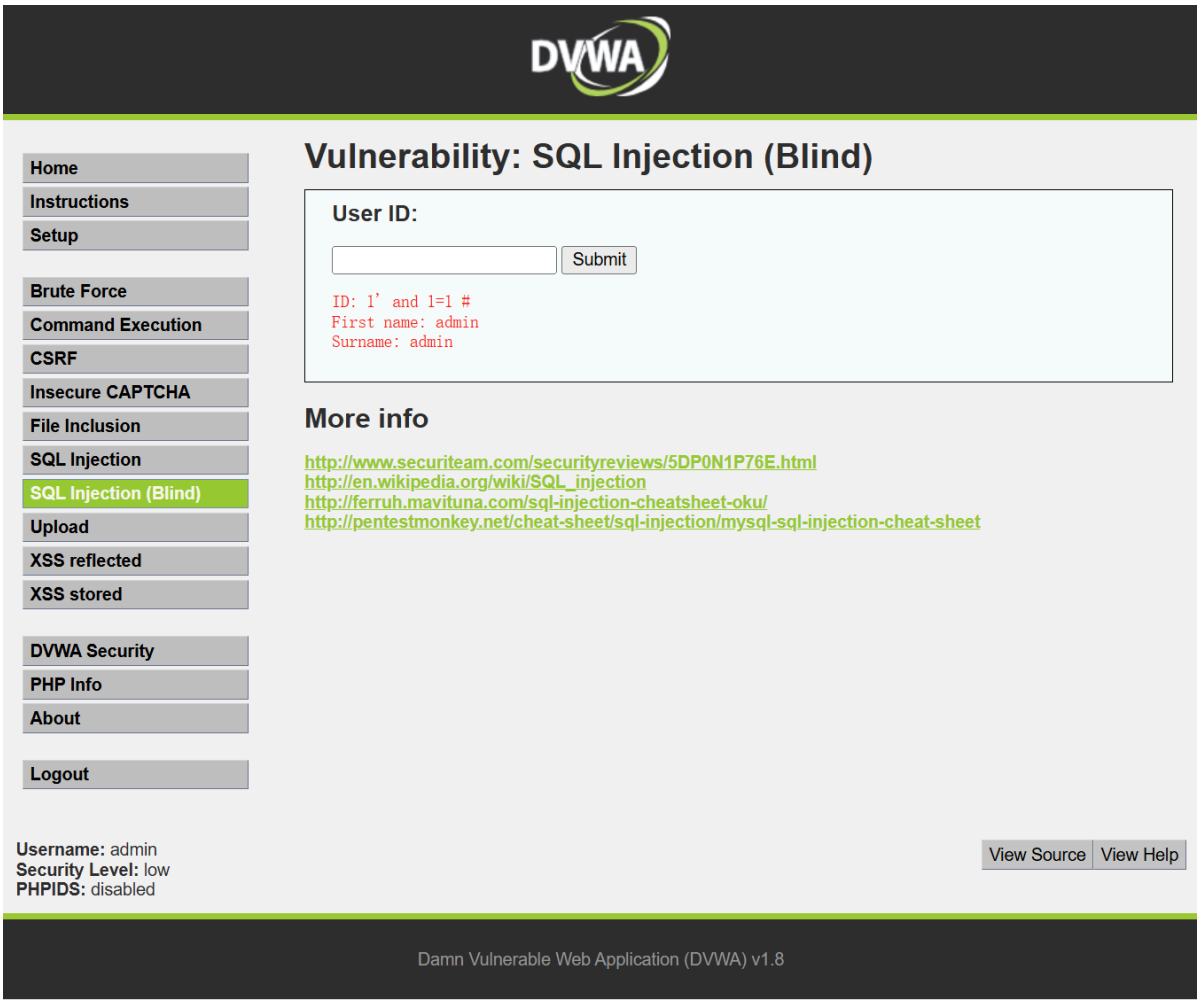
2 判断是否存在注入，注入是字符型还是数字型

首先我们输入 1，可以查询到结果，说明相应用户存在：

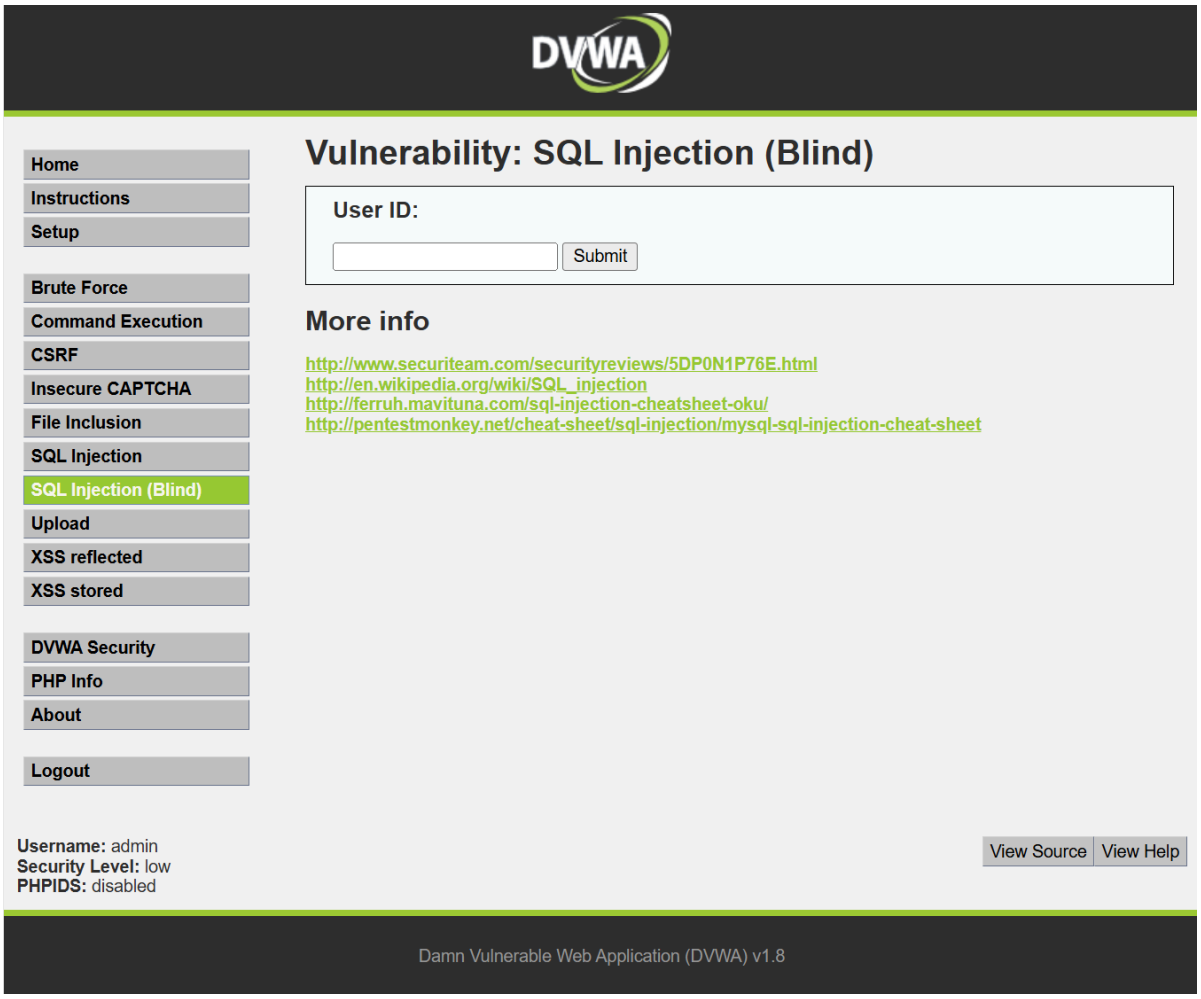


接下来，我们利用永真永假法来检测是否存在注入点。我们输入 `1' and 1=1 #`（单引号为了闭合原来 SQL 语句中的第一个单引号，而后面的#为了闭合后面的单引号）。

运行后，显示存在：



为了进一步验证我们的永真永假法，我们对上一部分的内容进行修改，输入一条逻辑有问题的句子进行测试，输入`1' and 1=2 #`，如下所示，显示不存在。

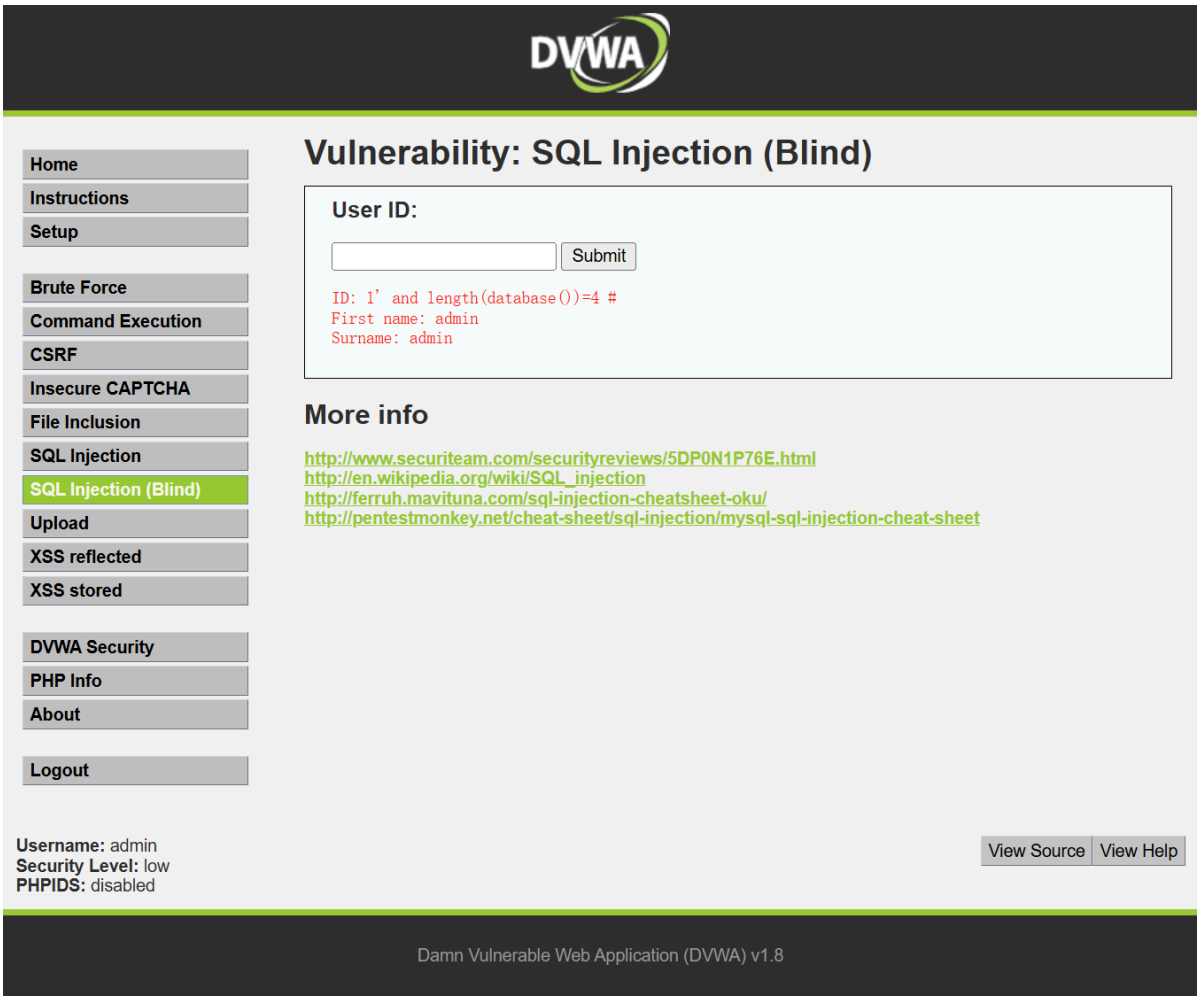


说明在内部存在字符型的`SQL`盲注。我们接下来进一步分析具体的字符内容。

3 爆破当前数据库名

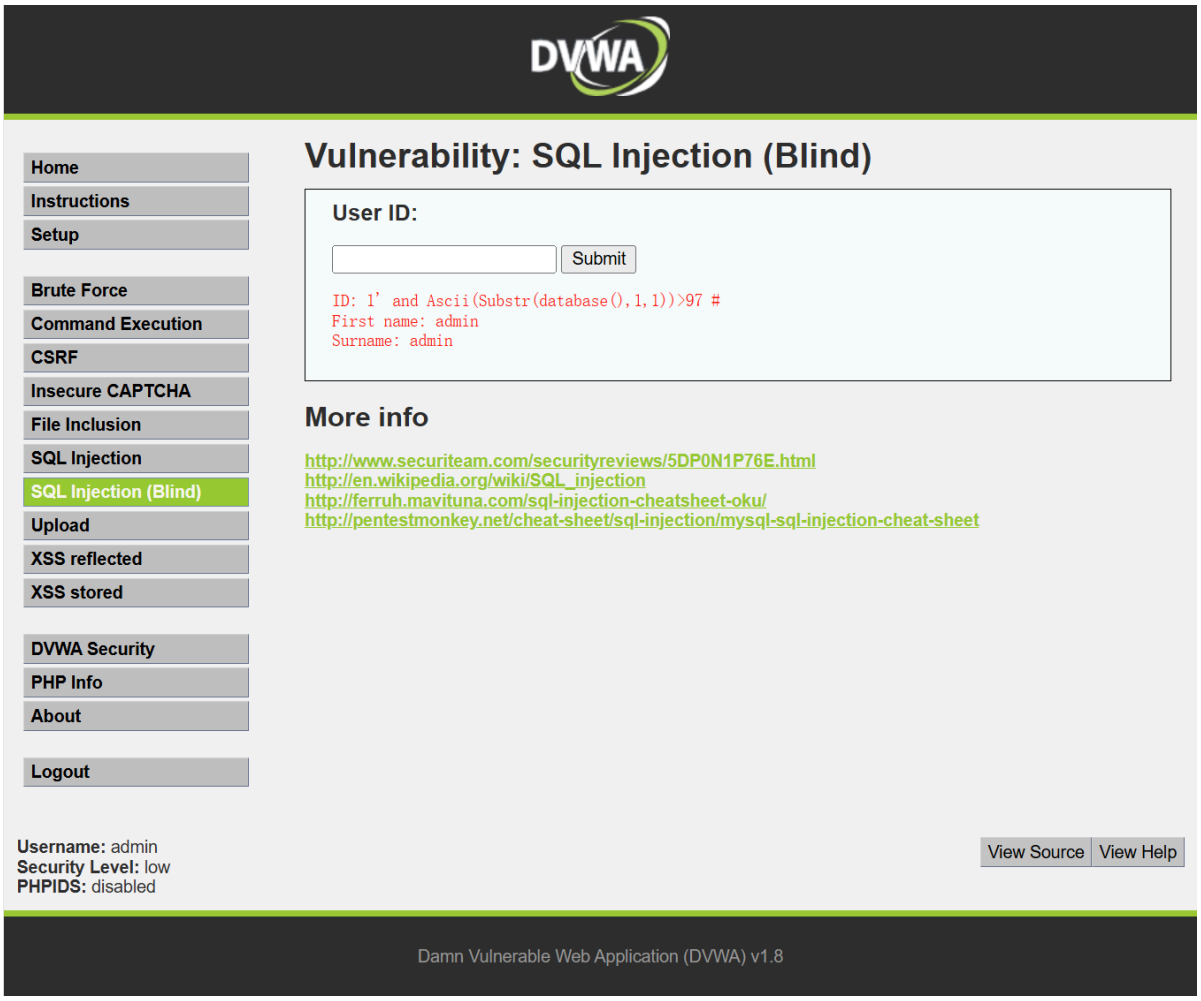
在得知了数据库存在盲注后，我们开始下一步的行动，就是去爆破数据库的名字，那么首先我们肯定是先猜测数据库的名字长度，我们还是通过`SQL`语句去一个一个尝试。

我们输入以下的`SQL`语句：`1' and length(database())=1 #`，该条语句的意思是判断数据库的名字是不是1个字，很显然没有输出；然后我们接着尝试其他的数字，比如说2,3,4。终于在测试4的时候，输入`1' and length(database())=4 #`，数据库显示该内容存在。于是我们确定了数据库名的长度是4。

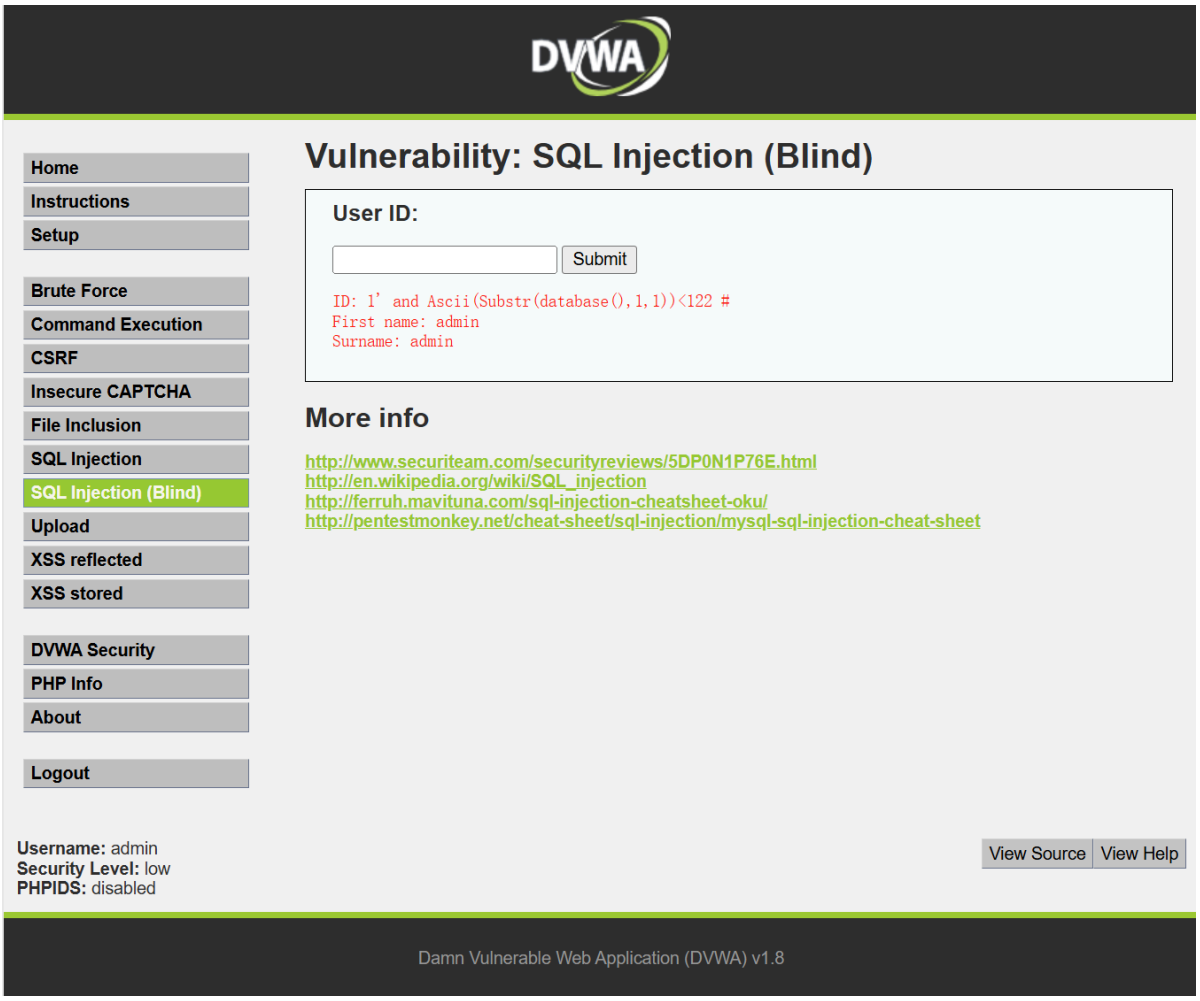


下一步，我们就需要去猜测数据库那四个字符分别是什么了。我们还是用同样的方法来进行盲注。最简单粗暴的方法就是对四个字符挨个遍历26个字母，但这样下来效率显然极低无比，因此我们选择采用**二分法**，根据字符的**ASCII码**值大小来不断筛选其可能的值。下面我们以猜测第一个字符的取值为例进行说明：

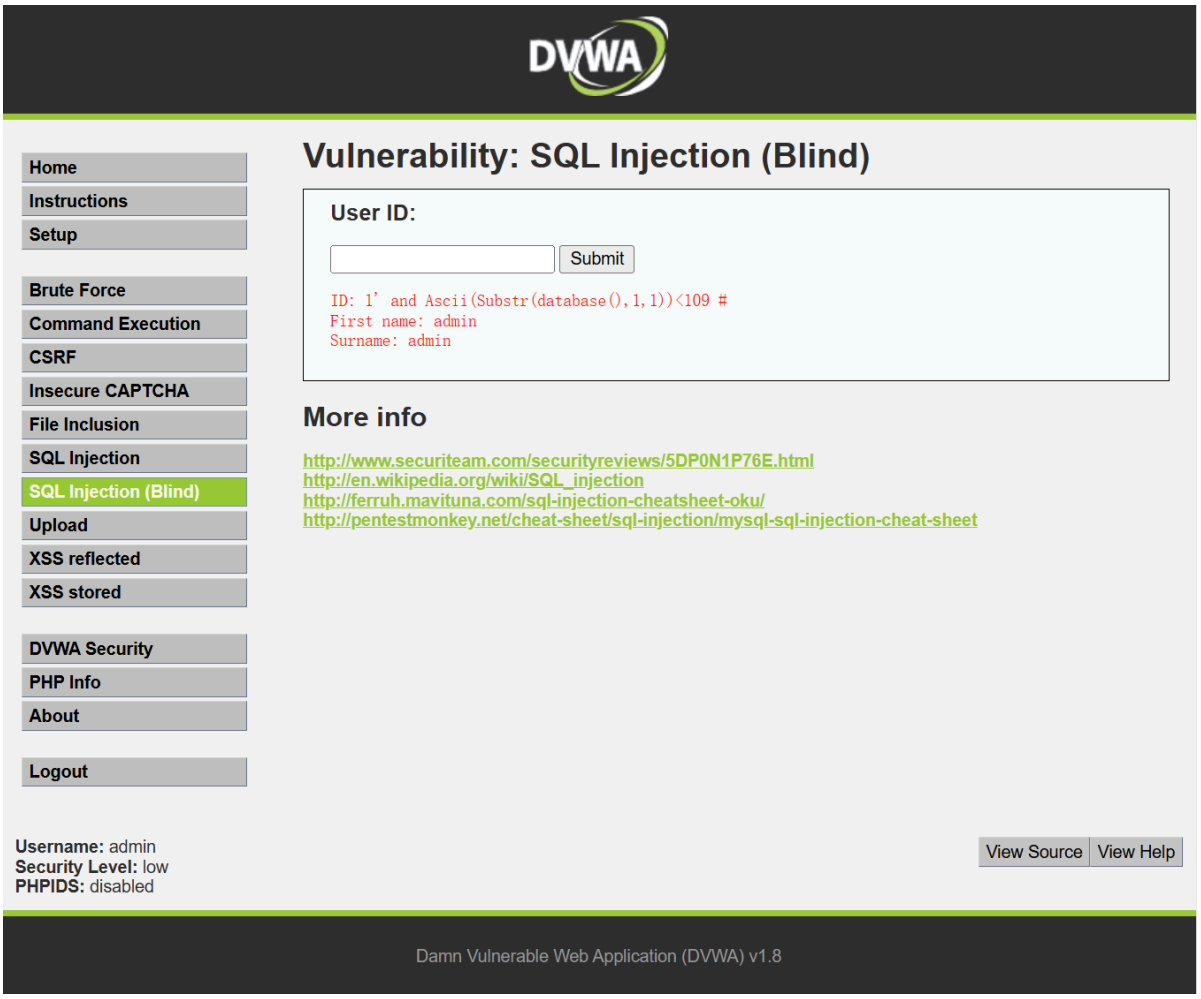
我们输入`1' and Ascii(Substr(database(),1,1))>97 #`，数据库显示存在，说明我们第一个字符的ASCII码是大于97的。



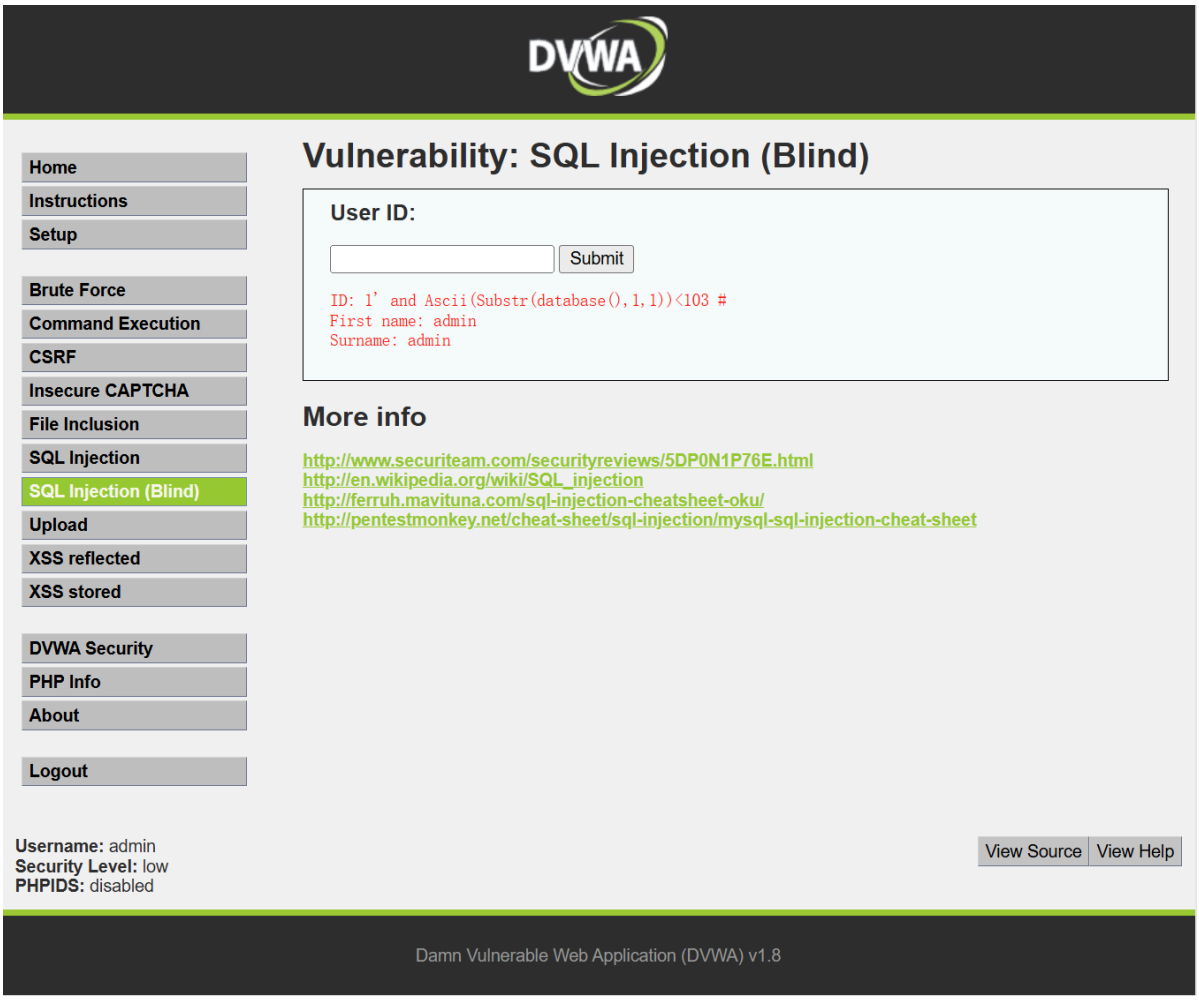
然后我们再进一步测试，输入`1' and Ascii(Substr(database(),1,1))<122 #`，这个就是判定第一个字母的`ASCII`码是否小于122。我们进行测试，结果如下所示：



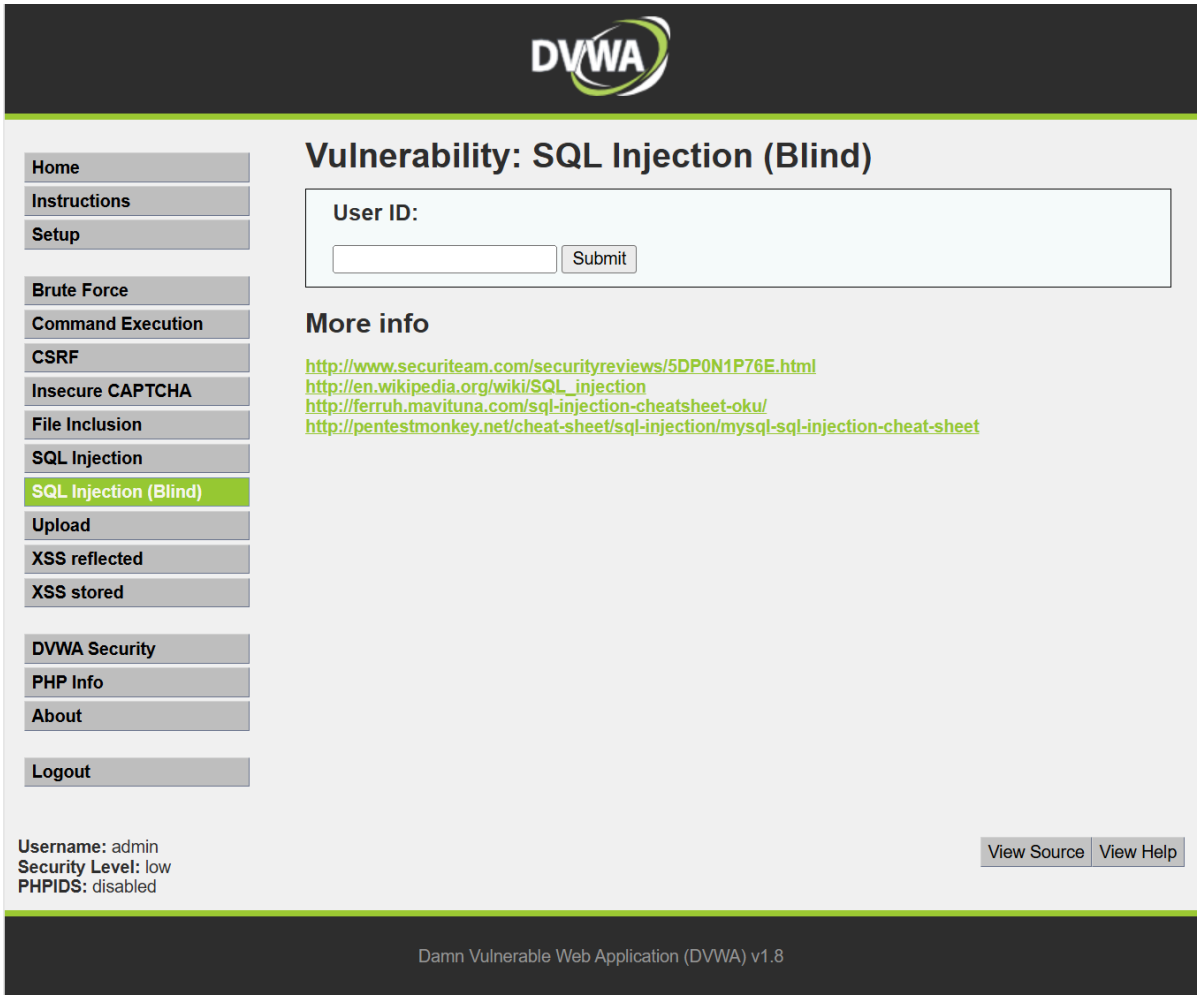
然后我们再测试中间值，一步一步缩小其范围，输入 `1' and Ascii(Substr(database(),1,1))<109 #`，显示存在，说明数据库名的第一个字符的`ASCII`值小于109；



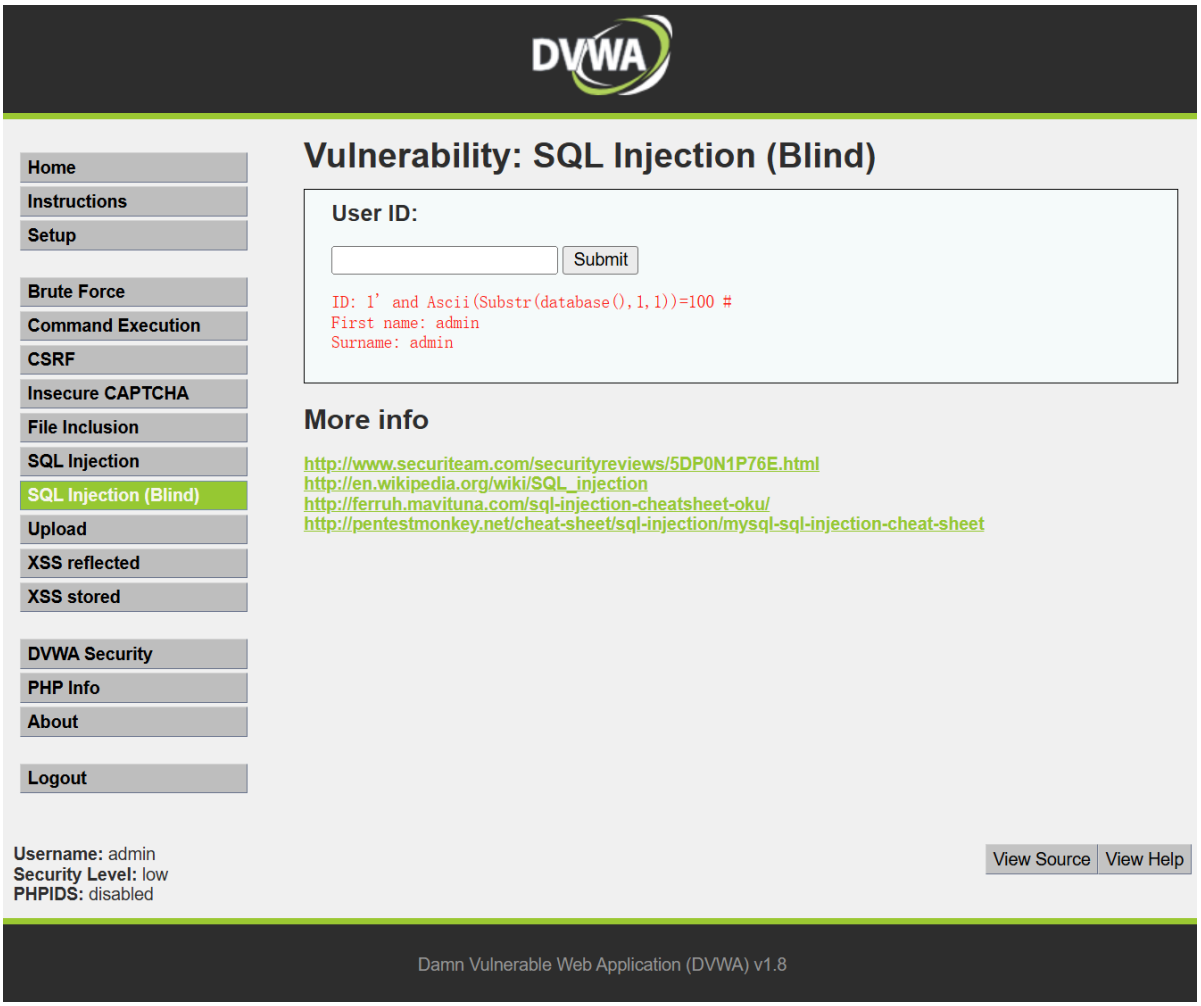
再输入 `1' and Ascii(Substr(database(),1,1))<103 #`，显示存在，说明数据库名的第一个字符的`ASCII`值小于103；



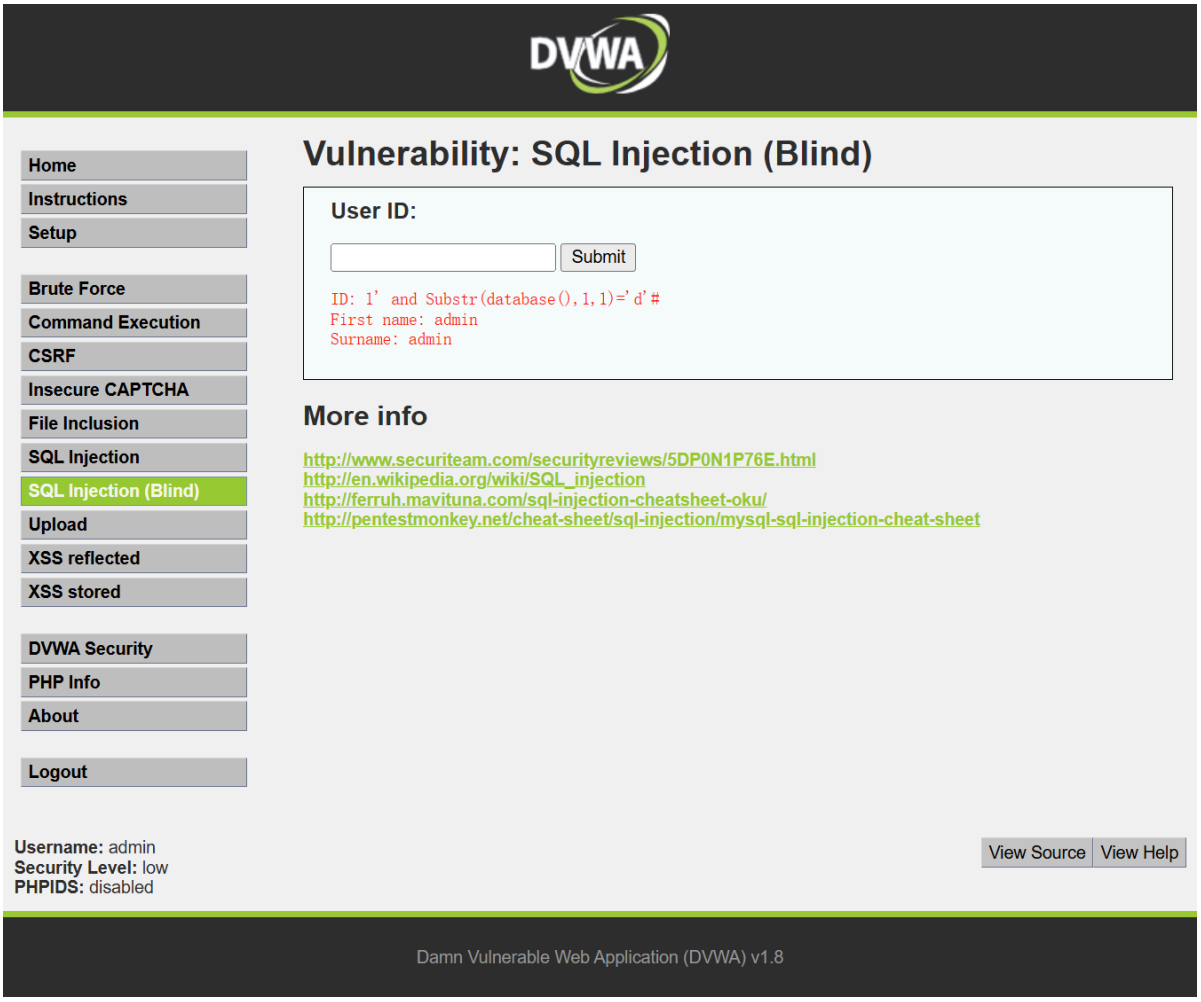
再输入 `1' and Ascii(Substr(database(),1,1))<100 #`，显示不存在，说明数据库名的第一个字符的 `ASCII` 值不小于100；



这样来看，`ASCII`码的范围就在100-103之间了，我们一个一个测试即可。我们首先输入`1' and Ascii(Substr(database(),1,1))=100 #`，发现数据库显示存在，说明我们第一个字母的`ASCII`值就是100，即小写字母d。



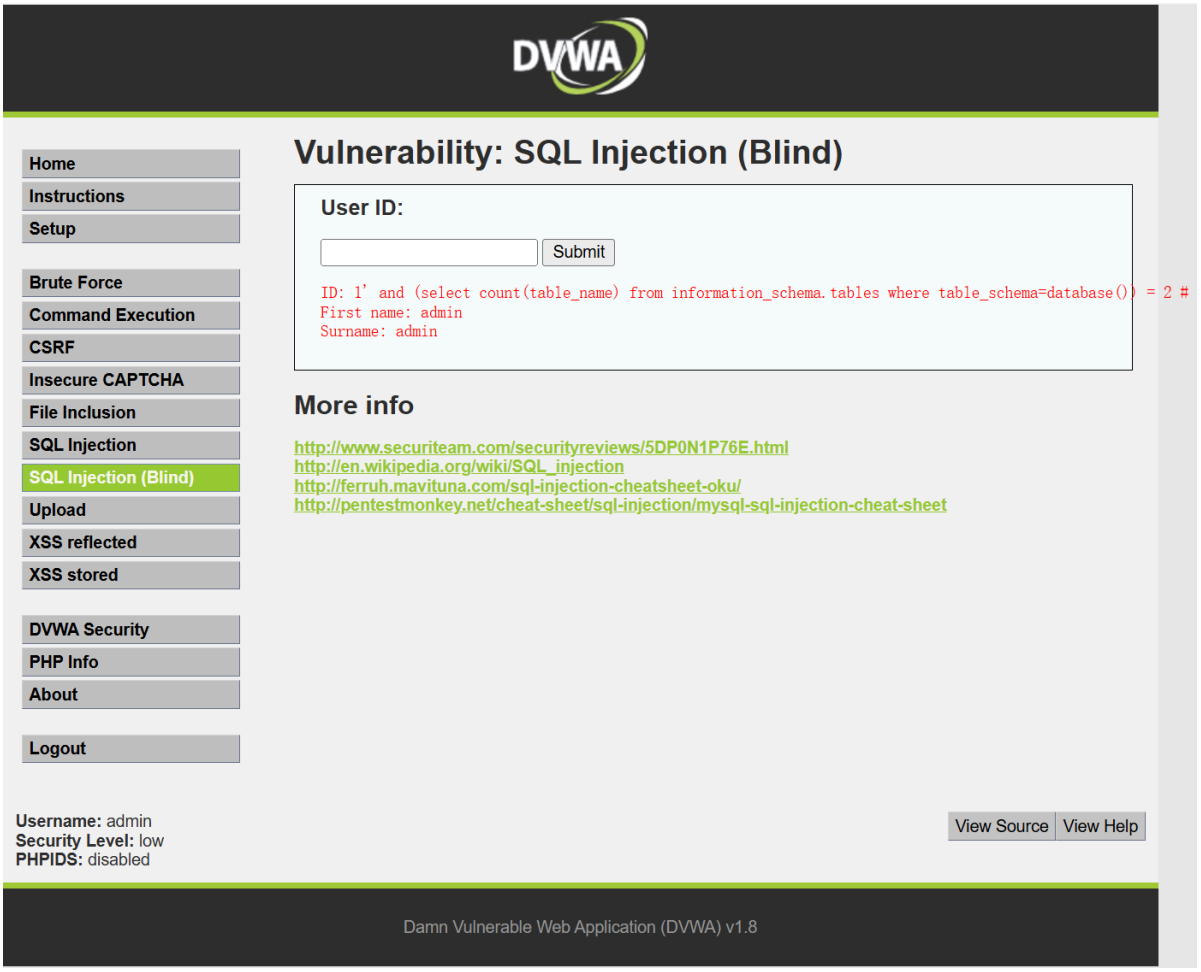
我们验证一下是不是d，输入语句`1' and Substr(database(),1,1)='d' #`，发现数据库中存在，说明我们的第一个字母就是d。



依次，我们分别推断出剩下的三个字母，他们分别是`v`，`w`，`a`。所以我们组合起来，数据库的名字就是`dvwa`。

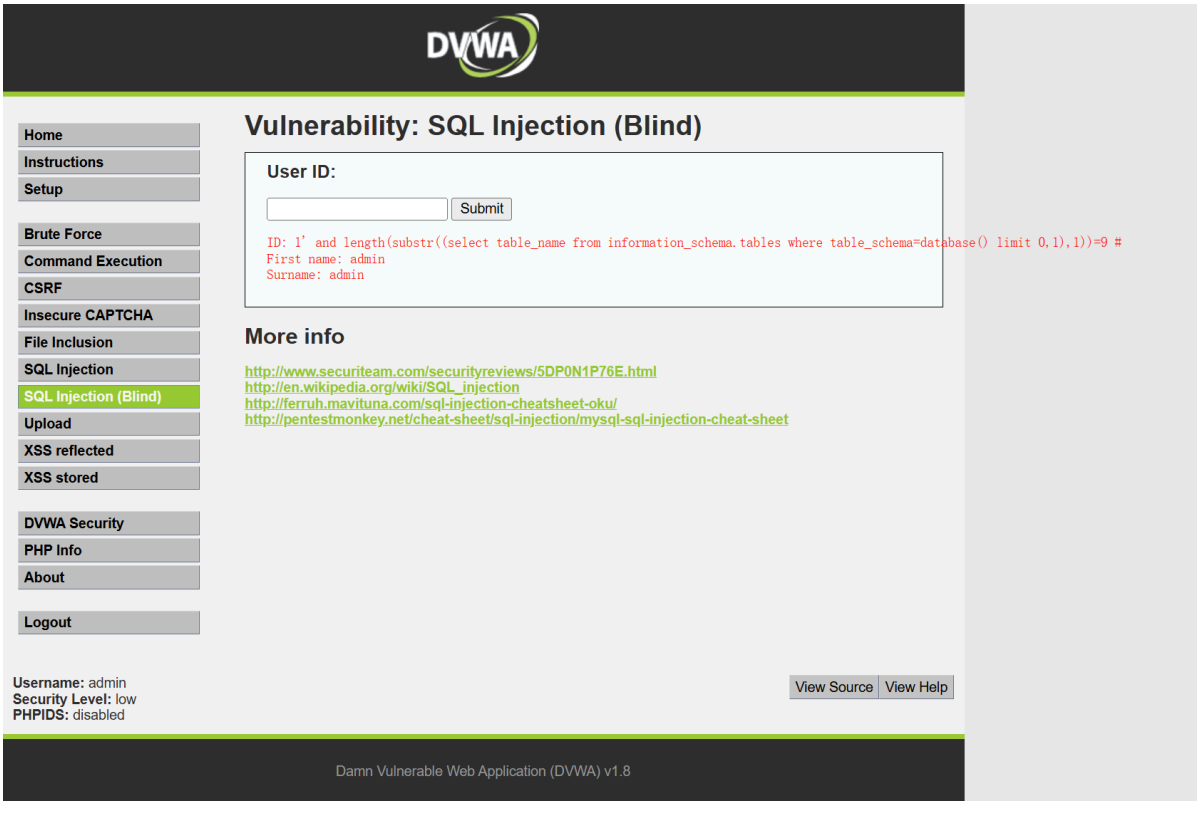
4 猜测数据库的表名

首先我们猜测数据库的表的数量，我们还是跟前面一样，输入`1' and (select count (table_name) from information_schema.tables where table_schema=database())=2 #`，显示不存在，说明数据表的数量不是1；我们继续测试，输入数量为2时，发现成功显示。



说明该数据库共有**两张表**。然后我们还是按照一样的方法来猜测表名，这一部分与上面的第一部分是大体相同的。我们直接给出最后的结果：

输入：`1' and length(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1))=9 #`



说明第一张表有9位。

然后对第一张表的九个字母都进行二分法查找，最后求解出来的结果是：**guestbook**

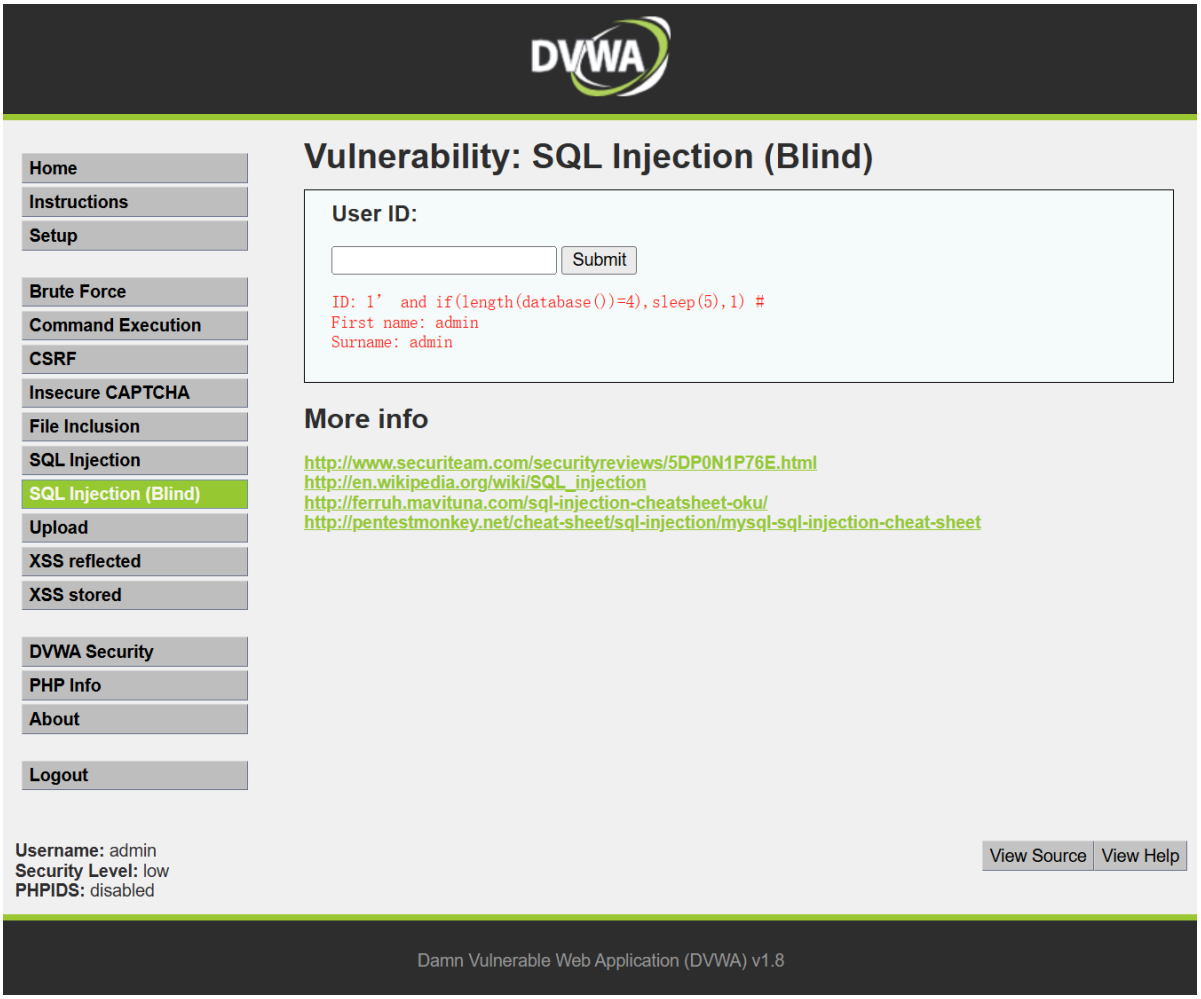
同样的，我们对第二张表也这样操作，最后求得的结果是，位数有四位，最后的结果是：**users**

5 后续步骤

后面的步骤与前面都是大同小异的，无非是将查询语句变化一下，最后我们甚至可以获得整个数据库的信息，只是所花费的时间很长罢了。我们后续还能够确定`guestbook`表中有几个字段，每个字段里都有些啥。

我们还可以尝试另外一种盲注的方法，就是通过实践的延迟去测试数据库，如果信息正确的话，系统会明显变得反应迟缓，这样的话我们就是猜对了对应的信息。我们可以通过以下的语句来进行查询：`1' and if(length(database())=4),sleep(5),1) #`

发现程序有了明显的延迟，这就说明我们本次猜测是正确的。



那么以上，我们就完成了本次实验的所有内容了！

心得体会：

通过本次实验，我基本了解了`SQL`盲注的一些常见手段与方法，了解到不需要知道数据库内部的消息，我们也能够将其中的信息给挖掘出来；对于`SQL`语句的单引号法和永真永假法两种检测注入点的方法也更加熟悉了；我还学会了几种常见的`SQL`函数的使用，如`Substr`，`length`，`ASCII`等；还学会了学会了在`OWASP`环境下进行`SQL`盲注，利用二分法来逐步推测出我们想要的数据。

一般情况下，盲注可分为三类：基于布尔 SQL 盲注、基于时间的 SQL 盲注、基于报错的 SQL 盲注。本次实验主要是基于布尔 SQL 盲注，在课后时间我也会去尝试其他几种盲注方法，加深对数据库 SQL 盲注的理解。通过本次实验，还提高了我对于应用开发过程中 SQL 注入的防范意识，需要采取适当的检测和过滤手段，不然的话我们数据库的信息很容易就被别人通过简单的输入 SQL 语句就获取了。