Query Results

In this section, we will learn how to control the results output of any query.

Sorting Results

We can sort the results of any query using ORDER BY and specifying the column to sort by:

Query Results

```
mysql> SELECT * FROM logins ORDER BY password;

+----+---------------+------------+---------------------+
| id | username      | password   | date_of_joining     |
+----+---------------+------------+---------------------+
|  2 | administrator | adm1n_p@ss | 2020-07-02 11:30:50 |
|  3 | john          | john123!   | 2020-07-02 11:47:16 |
|  1 | admin         | p@ssw0rd   | 2020-07-02 00:00:00 |
|  4 | tom           | tom123!    | 2020-07-02 11:47:16 |
+----+---------------+------------+---------------------+
4 rows in set (0.00 sec)
```

By default, the sort is done in ascending order, but we can also sort the results by ASC or DESC:

Query Results

```
mysql> SELECT * FROM logins ORDER BY password DESC;

+----+---------------+------------+---------------------+
| id | username      | password   | date_of_joining     |
+----+---------------+------------+---------------------+
|  4 | tom           | tom123!    | 2020-07-02 11:47:16 |
|  1 | admin         | p@ssw0rd   | 2020-07-02 00:00:00 |
|  3 | john          | john123!   | 2020-07-02 11:47:16 |
|  2 | administrator | adm1n_p@ss | 2020-07-02 11:30:50 |
+----+---------------+------------+---------------------+
4 rows in set (0.00 sec)
```

It is also possible to sort by multiple columns, to have a secondary sort for duplicate values in one column:

Query Results

```
mysql> SELECT * FROM logins ORDER BY password DESC, id ASC;

+----+---------------+-----------------+---------------------+
| id | username      | password        | date_of_joining     |
+----+---------------+-----------------+---------------------+
|  1 | admin         | p@ssw0rd        | 2020-07-02 00:00:00 |
|  2 | administrator | change_password | 2020-07-02 11:30:50 |
|  3 | john          | change_password | 2020-07-02 11:47:16 |
|  4 | tom           | change_password | 2020-07-02 11:50:20 |
+----+---------------+-----------------+---------------------+
4 rows in set (0.00 sec)
```

LIMIT results

In case our query returns a large number of records, we can LIMIT the results to what we want only, using LIMIT and the number of records we want:

Query Results

```
mysql> SELECT * FROM logins LIMIT 2;

+----+---------------+------------+---------------------+
| id | username      | password   | date_of_joining     |
+----+---------------+------------+---------------------+
|  1 | admin         | p@ssw0rd   | 2020-07-02 00:00:00 |
|  2 | administrator | adm1n_p@ss | 2020-07-02 11:30:50 |
+----+---------------+------------+---------------------+
2 rows in set (0.00 sec)
```

If we wanted to LIMIT results with an offset, we could specify the offset before the LIMIT count:

Query Results

```
mysql> SELECT * FROM logins LIMIT 1, 2;

+----+---------------+------------+---------------------+
| id | username      | password   | date_of_joining     |
+----+---------------+------------+---------------------+
|  2 | administrator | adm1n_p@ss | 2020-07-02 11:30:50 |
|  3 | john          | john123!   | 2020-07-02 11:47:16 |
+----+---------------+------------+---------------------+
2 rows in set (0.00 sec)
```

Note: the offset marks the order of the first record to be included, starting from 0. For the above, it starts and includes the 2nd record, and returns two values.

WHERE Clause

To filter or search for specific data, we can use conditions with the SELECT statement using the WHERE clause, to fine-tune the results:

Cheat Sheet

Go to Questions

Table of Contents

Introduction

Databases

Intro to Databases

Types of Databases

MySQL

Intro to MySQL

SQL Statements

Query Results

SQL Operators

SQL Injections

Intro to SQL Injections

Subverting Query Logic

Using Comments

Union Clause

Union Injection

Exploitation

Database Enumeration

Reading Files

Writing Files

Mitigations

Mitigating SQL Injection

Closing It Out

Skills Assessment - SQL Injection Fundamentals

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left

Code: sql

```sql
SELECT * FROM table_name WHERE <condition>;
```

The query above will return all records which satisfy the given condition. Let us look at an example:

Query Results

```
mysql> SELECT * FROM logins WHERE id > 1;

+----+---------------+------------+---------------------+
| id | username      | password   | date_of_joining     |
+----+---------------+------------+---------------------+
|  2 | administrator | adm1n_p@ss | 2020-07-02 11:30:50 |
|  3 | john          | john123!   | 2020-07-02 11:47:16 |
|  4 | tom           | tom123!    | 2020-07-02 11:47:16 |
+----+---------------+------------+---------------------+
3 rows in set (0.00 sec)
```

The example above selects all records where the value of `id` is greater than `1`. As we can see, the first row with its `id` as 1 was skipped from the output. We can do something similar for usernames:

Query Results

```
mysql> SELECT * FROM logins where username = 'admin';

+----+----------+----------+---------------------+
| id | username | password | date_of_joining     |
+----+----------+----------+---------------------+
|  1 | admin    | p@ssw0rd | 2020-07-02 00:00:00 |
+----+----------+----------+---------------------+
1 row in set (0.00 sec)
```

The query above selects the record where the username is `admin`. We can use the `UPDATE` statement to update certain records that meet a specific condition.

Note: String and date data types should be surrounded by single quote (') or double quotes (''), while numbers can be used directly.

LIKE Clause

Another useful SQL clause is LIKE, enabling selecting records by matching a certain pattern. The query below retrieves all records with usernames starting with `admin`:

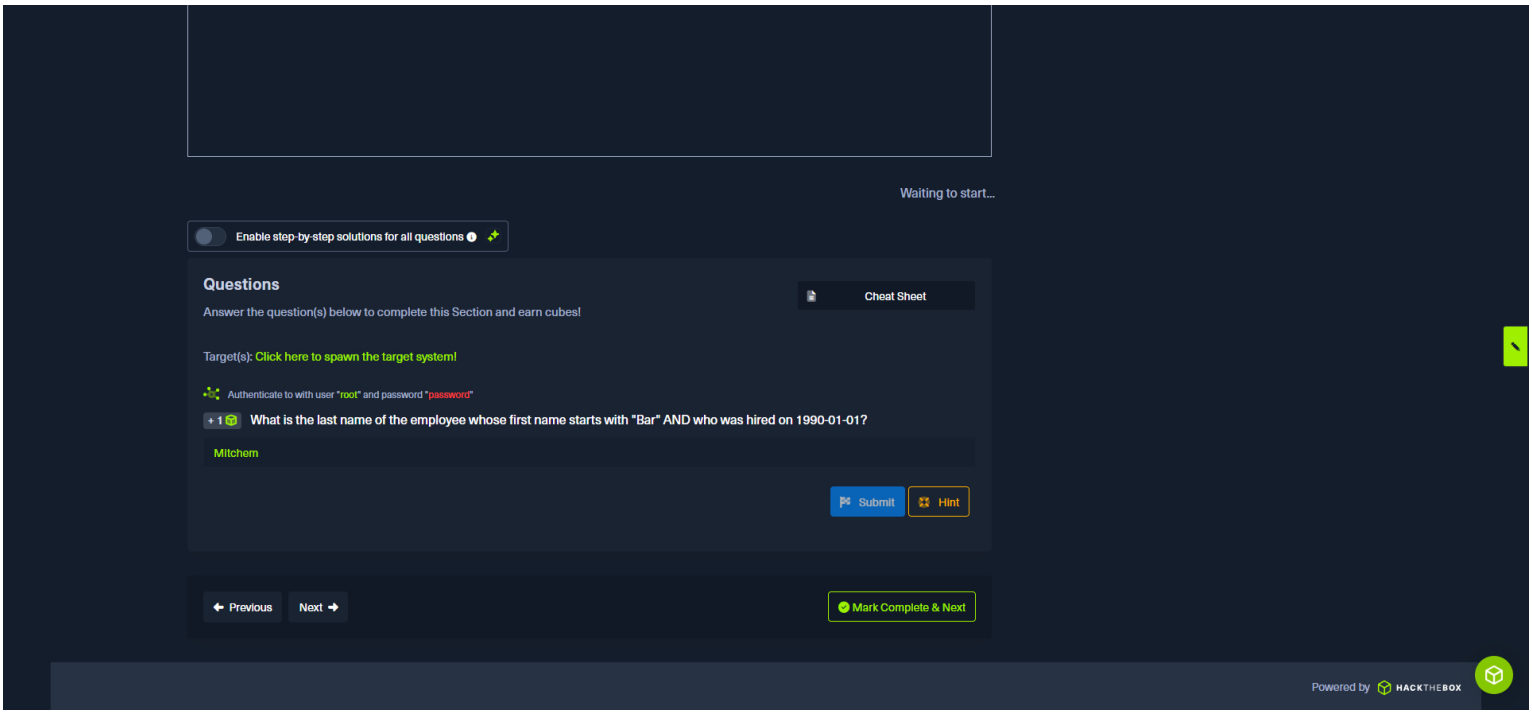
Query Results

```
mysql> SELECT * FROM logins WHERE username LIKE 'admin%';

+----+---------------+------------+---------------------+
| id | username      | password   | date_of_joining     |
+----+---------------+------------+---------------------+
|  1 | admin         | p@ssw0rd   | 2020-07-02 00:00:00 |
|  4 | administrator | adm1n_p@ss | 2020-07-02 15:19:02 |
+----+---------------+------------+---------------------+
2 rows in set (0.00 sec)
```

The `%` symbol acts as a wildcard and matches all characters after `admin`. It is used to match zero or more characters. Similarly, the `_` symbol is used to match exactly one character. The below query matches all usernames with exactly three characters in them, which in this case was `tom`:

Query Results

```
mysql> SELECT * FROM logins WHERE username like '___';

+----+----------+----------+---------------------+
| id | username | password | date_of_joining     |
+----+----------+----------+---------------------+
|  3 | tom      | tom123!  | 2020-07-02 15:18:56 |
+----+----------+----------+---------------------+
1 row in set (0.01 sec)
```


Waiting to start...
Enable step-by-step solutions for all questions
Questions
Answer the question(s) below to complete this Section and earn cubes!
Cheat Sheet
Target(s): Click here to spawn the target system!
Authenticate to with user "root" and password "password"
+ 1
What is the last name of the employee whose first name starts with "Bar" AND who was hired on 1990-01-01?
Mitchem
Submit
Hint
Previous
Next
Mark Complete & Next
Powered by HACKTHEBOX