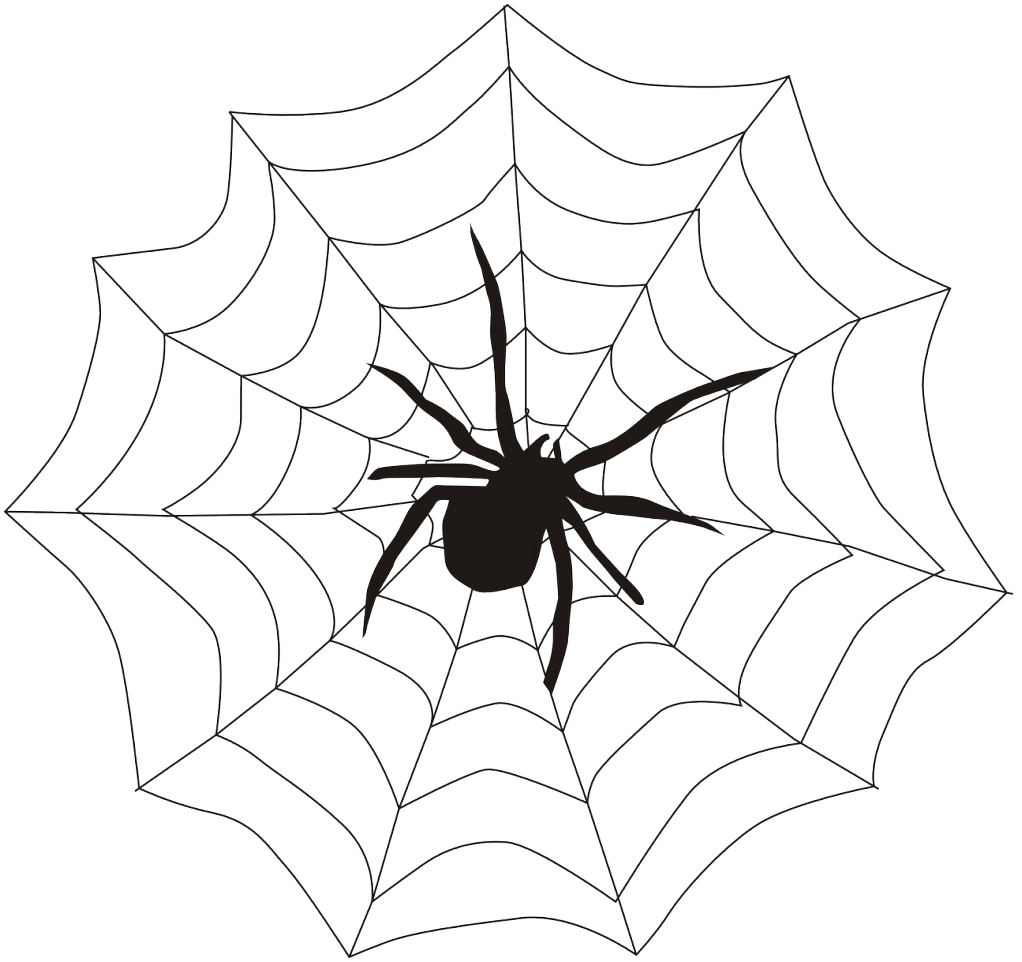
HackerFrogs Afterschool
Intro to SQL /w SQL Murder Mystery

Class:
Web App Hacking

Workshop Number:
AS-WEB-05

Document Version:
1.75

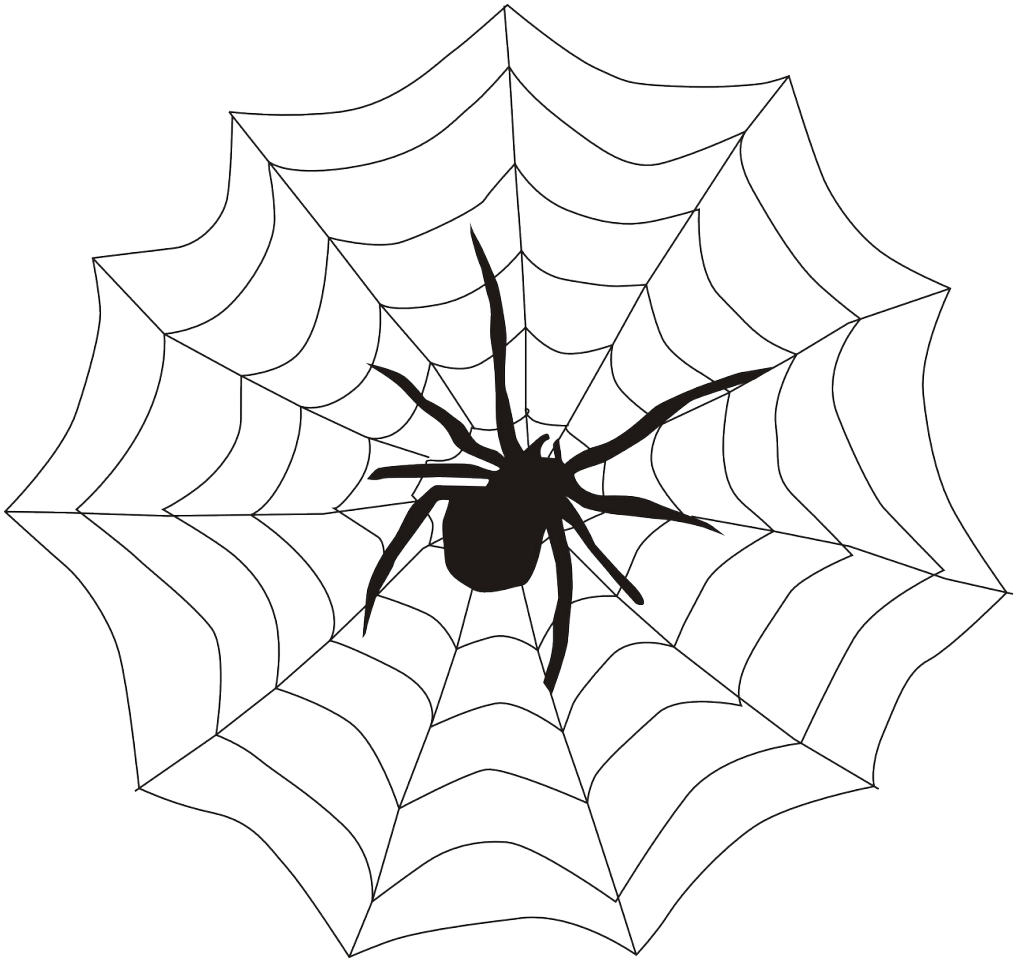
Special Requirements:
None



What We Learned In The Previous Workshop

This is the fifth workshop for intro to web app hacking.

Let's take a few moments to review the concepts we learned in the previous workshop.



OS Command Injection

Operating System (OS) Command Injection is a web app vulnerability where arbitrary OS commands can be performed on the webserver through a web interface.

OS Command Injection

OS Command Injection can often lead to complete compromise of the webserver, and if so, the server can be used as a foothold to attack other machines on the network.

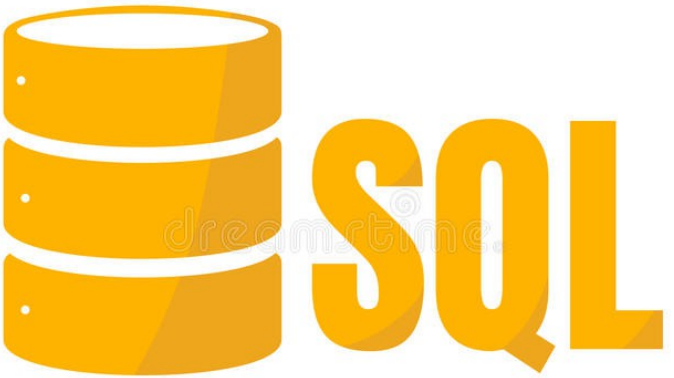
Now On To Our Topic!

Let's move on to the topic of this workshop:

The SQL database language

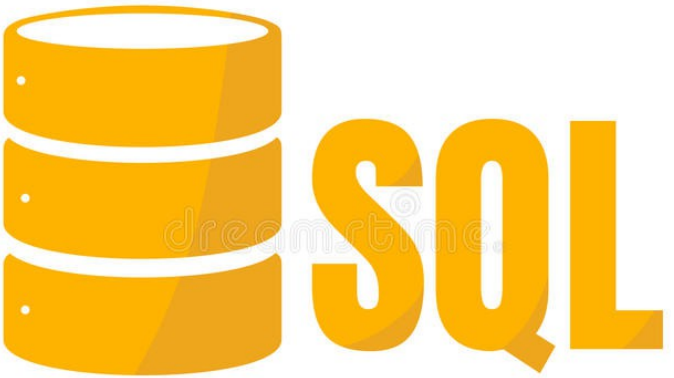
What is SQL?

Structured Query Language (SQL) is a programming language used for managing data held in relational databases.



What is SQL?

Simply put, SQL allows users to access and manage data contained in relational databases, which are common components in most web applications.



Why Learn About SQL?

As far as web app hacking is concerned, insecure implementation of SQL in web apps can lead to a very serious vulnerability called SQL Injection.

Why Learn About SQL?

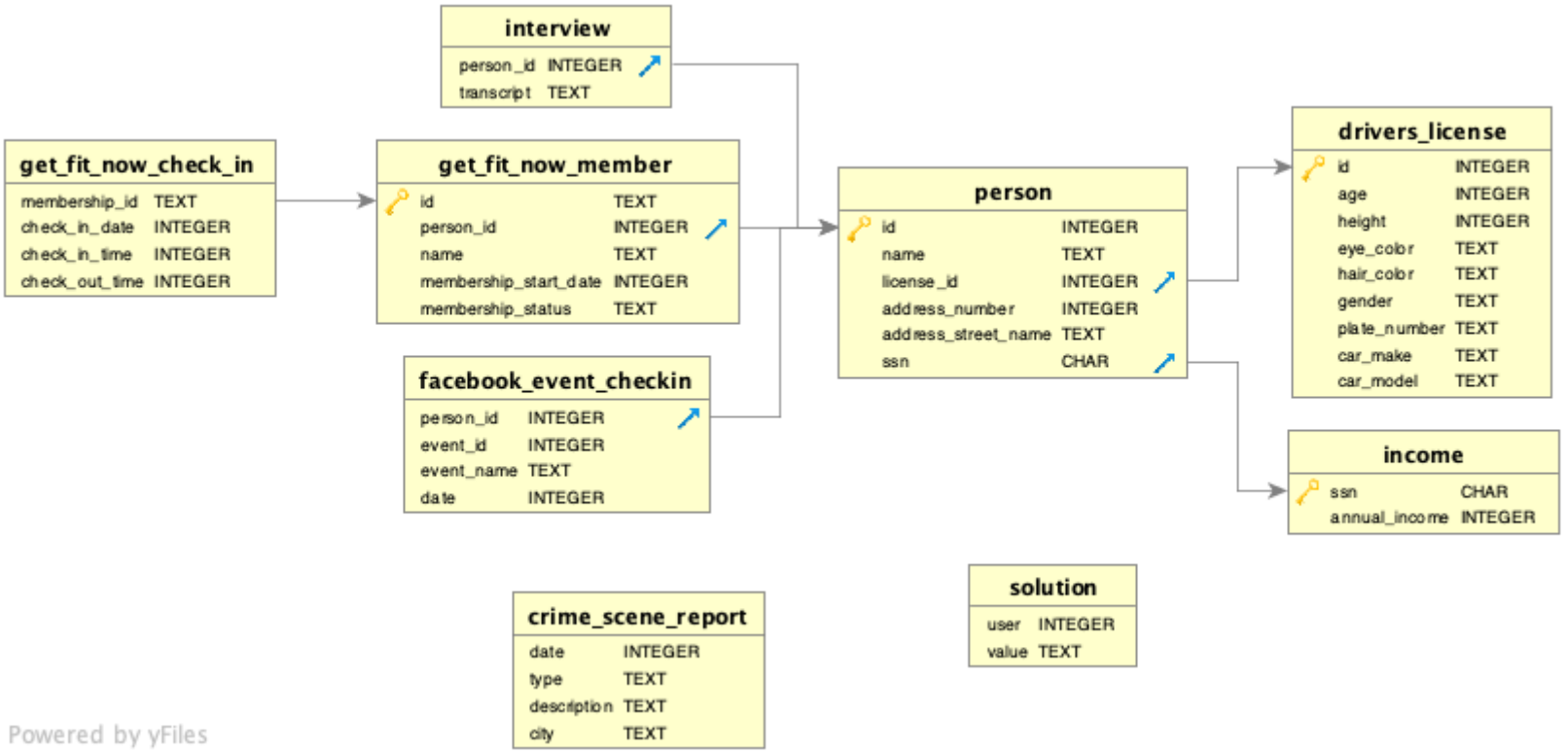
We will learn about SQL Injection in a later session, but we should first learn how to use SQL in its intended manner before we learn how to abuse it.

Let's Learn SQL Through a Game

In a web browser, navigate to the following URL:

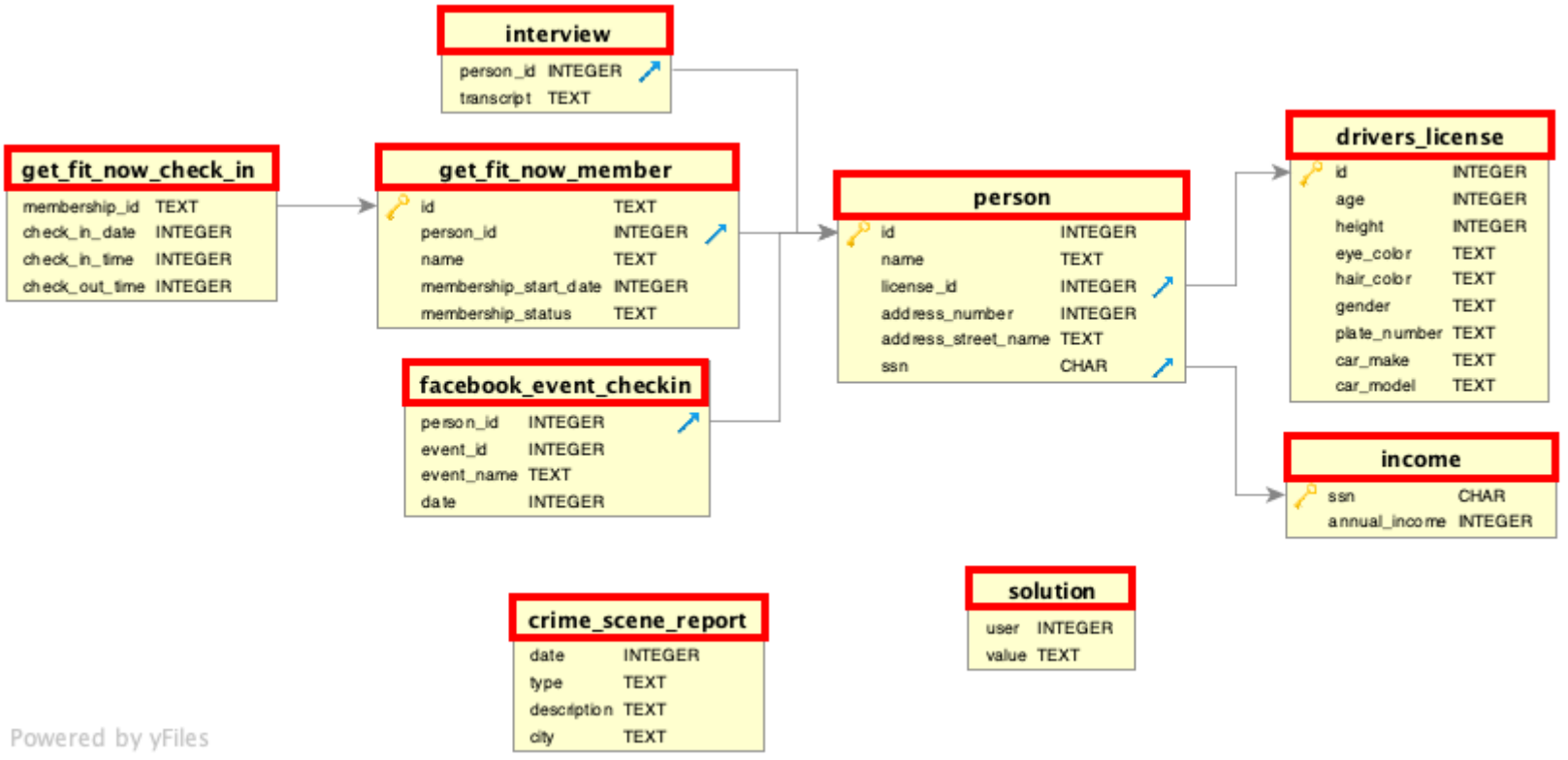
https://mystery.knightlab.com/walkthrough.html

SQL Database Structure



The largest unit in a SQL system is the **database**,

SQL Database Structure



The largest unit in a SQL system is the **database**, which contain one or more **tables**.

SQL Database Structure

id	name	license_id	address_number	address_street_name	ssn
10000	Christoper Peteuil	993845	624	Bankhall Ave	747714076
10007	Kourtney Calderwood	861794	2791	Gustavus Blvd	477972044
10010	Muoi Cary	385336	741	Northwestern Dr	828638512

Tables themselves can be subdivided into two elements:

SQL Database Structure

id	name	license_id	address_number	address_street_name	ssn
10000	Christoper Peteuil	993845	624	Bankhall Ave	747714076
10007	Kourtney Calderwood	861794	2791	Gustavus Blvd	477972044
10010	Muoi Cary	385336	741	Northwestern Dr	828638512

Columns, whose names identify the data that should be entered into them...

SQL Database Structure

id	name	license_id	address_number	address_street_name	ssn
10000	Christoper Peteuil	993845	624	Bankhall Ave	747714076
10007	Kourtney Calderwood	861794	2791	Gustavus Blvd	477972044
10010	Muoi Cary	385336	741	Northwestern Dr	828638512

And **rows**, which contain data entries that correspond to the columns they fall under.

Determining the Database Software

```
1 SELECT sqlite_version();
2
```

There are a lot of different types of SQL software, (MySQL, PostgreSQL, SQLite, etc) so it's important to ID the type of SQL we're using

Determining the Database Software

```
1 SELECT sqlite_version();
2
```

The SQL Murder Mystery website uses SQLite, by the way, and using the `select sqlite_version();` statement will retrieve version information

Determining the Database Software

The SQL Zoo website uses MySQL by default, but you can switch its system to other SQL types

Determining Database Names

```
1 SELECT sqlite_version();
2
```

Database systems can contain multiple databases in the same system, so its important to know the names of all the databases in the system

Determining Database Names

```
show databases;
```

MySQL uses the command: `show databases` to output databases on the system

Selecting a Database

```
1 SQLite only uses one database at a time;
2 TIL :D
```

If there are multiple databases, we need to select the database we want to interact with

Selecting a Database

```
use database_name;
```

Using MySQL, we use the command

`use <database_name>;`

Listing out the Database Tables

```
1 SELECT name FROM sqlite_master
2 WHERE type='table';
```

The next step is to list out which tables exist in the database

Listing out the Database Tables

```
show tables;
```

In MySQL, we use the following command:

`show tables;`

Determining Columns in Tables

```
1 PRAGMA table_info(person);
2
```

And finally, we find determine the names of columns in the tables

Determining Columns in Tables

```
1 PRAGMA table_info(person);
2
```

Using MySQL, we would use this command:

`describe <table_name>;`

The SELECT Command

```
1 SELECT count(*)
2 FROM person;
```

The **SELECT** command is the most common SQL command, which returns SQL entries depending on what additional parameters are provided.

The FROM Command

```
1 SELECT count(*)
2 FROM person;
```

The **FROM** command specifies which table to retrieve data from.

The WHERE Filter

```
1 SELECT * FROM person WHERE name = 'Kinsey Erickson'
```

id	name	license_id	address_number	address_street_name	ssn
89906	Kinsey Erickson	510019	309	Northwestern Dr	635287661

And the **WHERE** filter allows very specific information to be returned by a query.

The = Operator

```
1 SELECT * FROM person WHERE name = 'kinsey Erickson'
```

No data returned

The = operator is used with the WHERE filter to return exact matches. If the contents of a row differ by even a single character, the match will not return. (The 'k' in the above example is not capitalized, so a valid match is not found)

The AND Clause

```
1 SELECT * FROM person WHERE name = 'John Dile'
2 AND address_street_name = 'Icehouse Ave'
```

id	name	license_id	address_number	address_street_name	ssn
13915	John Dile	909334	3783	Icehouse Ave	957131634

The **AND** clause is used with the **WHERE** filter to further restrict the results returned, only returning rows that match both parameters on either side of the **AND** clause.

The Like Operator and % Wildcard

```
1 SELECT * FROM person WHERE name like '%John%'
2 AND address_street_name like '%Icehouse%'
```

id	name	license_id	address_number	address_street_name	ssn
13915	John Dile	909334	3783	Icehouse Ave	957131634

The **like** operator is used with the % wildcard character to return partial matches. The % symbol matches any characters on its respective side of the string.

UNION select

```
SELECT name, population
FROM world
UNION SELECT winner, yr
FROM nobel
LIMIT 4;
```

name	population
Walther Nernst	1920
Grenada	112579
Wendell Stanley	1946
Alice Munro	2013

We can use UNION select statements to retrieve data from more than table at the same time

UNION select

```
SELECT name, population
FROM world
UNION SELECT winner, yr
FROM nobel
LIMIT 4;
```

name	population
Walther Nernst	1920
Grenada	112579
Wendell Stanley	1946
Alice Munro	2013

There are a couple of restrictions on how we can use UNION select. First, we have to return the same number of colums from each table

UNION select

```
SELECT name, population
FROM world
UNION SELECT winner, yr
FROM nobel
LIMIT 4;
```

name	population
Walther Nernst	1920
Grenada	112579
Wendell Stanley	1946
Alice Munro	2013

Second, columns with text cannot be mixed with columns with numbers. In this example, **name** and **winner** are both text columns

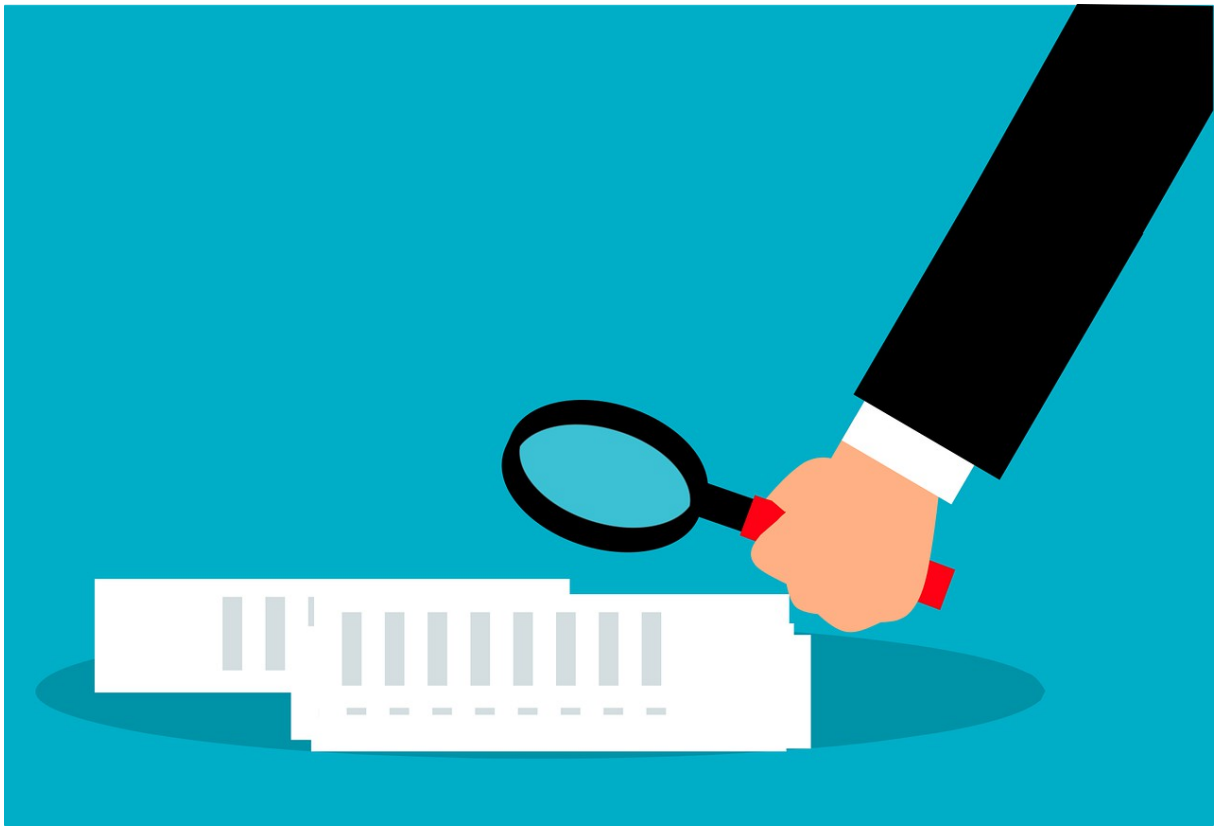
UNION select

```
SELECT name, population
FROM world
UNION SELECT winner, yr
FROM nobel
LIMIT 4;
```

name	population
Walther Nernst	1920
Grenada	112579
Wendell Stanley	1946
Alice Munro	2013

Likewise, the **population** and **yr** columns match because they both contain numbers.

Summary



Let's review the SQL concepts we learned in this workshop:

Structured Query Language (SQL)

Structured Query Language (SQL) is a programming language used for managing data held in relational databases.

The SELECT Command

```
1 SELECT count(*)
2 FROM person;
```

The **SELECT** command is common to all SQL queries looking to return specific info from the database.

The WHERE Filter

```
1 SELECT * FROM person WHERE name = 'Kinsey Erickson'
```

id	name	license_id	address_number	address_street_name	ssn
89906	Kinsey Erickson	510019	309	Northwestern Dr	635287661

And the **WHERE** filter allows very specific information to be returned by a query.

What's Next?

In the next HackerFrogs Afterschool web exploitation workshop, we'll learn about SQL injection vulnerabilities with the TryHackMe platform.

Extra Credit

Looking for more study material on this workshop's topics?

See this video's description for links to supplemental documents and exercises!

Until Next Time, HackerFrogs!