Recall the concept

- Database
- Database Model
- DBMS – Benefits of a DBMS
- XAMPP
- Entity relationship
- RDBMS – MySQL

Queries

Learning Objective:

Create, evaluate and improve search queries that use multiple criteria and relational operators to find specific information

Success criteria

- know what is Queries
- know the purpose of the Queries
- can create Queries using the structure
- can create Queries using commands SQL: SELECT, WHERE

MySQL – RDBMS

SQL stands for the Structured Query Language.

It defines how to insert, retrieve, modify and delete data.

Создание базы данных

```
CREATE DATABASE my_first_db;
```

DROP DATABASE: Удалить базу данных

DROP TABLE: Удалить таблицу

EXPLAIN: Показать структуру таблицы

USE: Выбор базы данных

Создать таблицу

```
CREATE TABLE users (
    username VARCHAR(20),
    create_date DATE
);
```

Первичный ключ

```
CREATE TABLE users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(20),
    create_date DATE
);
```

ALTER TABLE: Изменить таблицу

Удаляем столбец

ALTER TABLE users DROP email;

Изменение столбца

ALTER TABLE users

CHANCE username

User_name VARCHAR<30>;

INSERT: Добавляем данные в таблицу

INSERT INTO users VALUES (”Alex”,’2002-07-25’);

Select

SELECT is used to retrieve rows selected from one or more tables.

The SELECT statement allows you to ask the database a question (Query it), and specify what data it returns.

```
SELECT name, DoB --what to return
FROM crooks      --where are you returning it from
```

name	DoB
Geoff	12/05/1982
Jane	05/08/1956
Keith	07/02/1999
Oliver	22/08/1976
Kelly	11/11/1911
Marea	14/07/1940

SELECT, WHERE

We need to use another statement, the WHERE clause, allowing us to give the query some criteria (or options):

```
SELECT ID, name, DoB
FROM crooks
WHERE town = 'Snape' AND gender = 'male'  --Criteria
```

ID	name	DoB
3	Keith	07/02/1999

Operators in The WHERE Clause

So you can see we used AND statement, we also can use OR, NOT and others like:

= !=	Equal Not Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

Example

Say the police knew that a crime had been committed by a heavily scarred woman (4+ scars), they want a list of all the scarred women:

```
SELECT name, town, scars
FROM crooks
WHERE numScars >= 4 AND gender = 'female'  --Criteria
```

This would return:

name	town	numScars
Kelly	East Ham	10
Marea	Wythenshawe	6

Example

However, the police want to quickly sort through and see who is the most heavily scarred. We are going to use an **ORDER** command:

```
SELECT name, town
FROM crooks
WHERE numScars >= 4 AND gender = 'female'  --Criteria
ORDER BY numScars DESC --sorts the numScars values in big to small order
```

ORDER BY numScars sorts your returned data
into **DESC**ending (big to small) or **ASC**ending (small to big) order

Select with Comparison Operators

For numbers (INT, DECIMAL, FLOAT)

```
mysql> SELECT name, price FROM products WHERE price < 1.0;
+-----------+-------+
| name      | price |
+-----------+-------+
| Pencil 2B |  0.48 |
| Pencil 2H |  0.49 |
+-----------+-------+
2 rows in set (0.00 sec)

mysql> SELECT name, quantity FROM products WHERE quantity <= 2000;
+-----------+----------+
| name      | quantity |
+-----------+----------+
| Pen Black |     2000 |
+-----------+----------+
1 row in set (0.00 sec)
```

For strings, you could also use '=', '<>', '>', '<', '>=', '<=' to compare two strings (e.g., productCode = 'PEC').

```
mysql> SELECT name, price FROM products WHERE productCode = 'PEN';
                                          -- String values are quoted
+-----------+-------+
| name      | price |
+-----------+-------+
| Pen Red   |  1.23 |
| Pen Blue  |  1.25 |
| Pen Black |  1.25 |
+-----------+-------+
3 rows in set (0.00 sec)
```

String Pattern Matching - LIKE and NOT LIKE

we can perform pattern matching using operator LIKE (or NOT LIKE) with wildcard characters. The wildcard '_' matches any single character; '%' matches any number of characters (including zero). For example,

```
-- "name" begins with 'PENCIL'
mysql> SELECT name, price FROM products WHERE name LIKE 'PENCIL%';
+-----------+-------+
| name      | price |
+-----------+-------+
| Pencil 2B |  0.48 |
| Pencil 2H |  0.49 |
+-----------+-------+

-- "name" begins with 'P', followed by any two characters,
--   followed by space, followed by zero or more characters
mysql> SELECT name, price FROM products WHERE name LIKE 'P__ %';
+-----------+-------+
| name      | price |
+-----------+-------+
| Pen Red   |  1.23 |
| Pen Blue  |  1.25 |
| Pen Black |  1.25 |
+-----------+-------+
```

Arithmetic Operators - +, -, *, /, DIV, %

Logical Operators - AND, OR, NOT, XOR

```
mysql> SELECT * FROM products WHERE quantity >= 5000 AND name LIKE 'Pen %';
+-----------+-------------+----------+----------+-------+
| productID | productCode | name     | quantity | price |
+-----------+-------------+----------+----------+-------+
|      1001 | PEN         | Pen Red  |     5000 |  1.23 |
|      1002 | PEN         | Pen Blue |     8000 |  1.25 |
+-----------+-------------+----------+----------+-------+

mysql> SELECT * FROM products WHERE quantity >= 5000 AND price < 1.24 AND name LIKE 'Pen %';
+-----------+-------------+---------+----------+-------+
| productID | productCode | name    | quantity | price |
+-----------+-------------+---------+----------+-------+
|      1001 | PEN         | Pen Red |     5000 |  1.23 |
+-----------+-------------+---------+----------+-------+
```

Further Reading…..

IN, NOT IN

SELECT * FROM products WHERE name IN ('Pen Red', 'Pen Black');

BETWEEN, NOT BETWEEN

SELECT * FROM products WHERE (price BETWEEN 1.0 AND 2.0) AND (quantity BETWEEN 1000 AND 2000);

IS NULL, IS NOT NULL

SELECT * FROM products WHERE productCode IS NULL;

ORDER BY Clause

SELECT * FROM products WHERE name LIKE 'Pen %' ORDER BY price DESC;

- **create table Employee(empno int(5) primary key, ename varchar(30), job varchar(25), hiredate date, sal double(10,2), commission double(6,2), deptt int(2));**

- **INSERT INTO employee VALUES (1001,”Alex”,”Teacher”,’2017-07-25’, 5678.90, 100.0, 10);**

- **Select** * from Employee where commission>0

- **Select jobs from employee;**

- **SELECT * FROM EMPLOYEE WHERE ENAME LIKE “_ _ _ _ _”;**

- **SELECT * FROM EMPLOYEE WHERE ENAME LIKE “_ _ _ _p%”;**

- **SELECT * FROM employee WHERE deptt=** 'computer ' **ORDER BY ename;**

- **Select ename, hiredate from employee where job not like “history”;**

- http://jtest.ru/bazyi-dannyix/sql-dlya-nachinayushhix-chast-3.html
- https://www.ntu.edu.sg/home/ehchua/programming/sql/MySQL_Beginner.html
- https://myrusakov.ru/
- http://www.firststeps.ru/sql/r.php?9