MySQL

MySQL command Line Client : BASE

- **Database name**

World.city => database.table

`mysql> SHOW DATABASES;`	
`mysql> CREATE DATABASE pets;`	
`mysql> USE pets`	The USE statement tells MySQL to use pets as the default database for subsequent statements
`mysql> SHOW TABLES;`	
`mysql> DESCRIBE cats;`	shows information on all columns of a table
`mysql> SHOW CREATE TABLE cats`	shows a CREATE TABLE statement, which provides even more details on the table
`DROP DATABASE IF EXISTS test2;`	Delete test2 database

`RENAME TABLE`
`   Absence to absences,`
`   Classe to classes ;`	

`DELETE FROM absences`
`    WHERE student_id = 6;`	

- **Créer une table**

```
CREATE TABLE cats
(
  id              INT unsigned NOT NULL AUTO_INCREMENT, # Unique ID for the record
  name            VARCHAR(150) NOT NULL,                # Name of the cat
  owner           VARCHAR(150) NOT NULL,                # Owner of the cat
  birth           DATE NOT NULL,                        # Birthday of the cat
  PRIMARY KEY     (id)                                  # Make the id the primary key
);
```

- VARCHAR(30) : Characters with an expected max length of 30
- NOT NULL : Must contain a value
- NULL : Doesn't require a value
- CHAR(2) : Contains exactly 2 characters
- DEFAULT "PA" : Receives a default value of PA
- MEDIUMINT : Value no greater then 8,388,608
- UNSIGNED : Can't contain a negative value
- INT : Contains a number without decimals070
- AUTO_INCREMENT : Generates a number automatically that is one greater then the previous row

- **Insérer des valeurs**

```
INSERT INTO cats ( name, owner, birth) VALUES
  ( 'Sandy', 'Lennon', '2015-01-03' ),
  ( 'Cookie', 'Casey', '2013-11-13' ),
  ( 'Charlie', 'River', '2016-05-21' );
```

- **Select Statement**

```
mysql> SELECT * FROM cats;
mysql> SELECT name FROM cats WHERE owner = 'Casey';
mysql> DELETE FROM cats WHERE name='Cookie';
```

```
SELECT CONCAT(first_name, " ", last_name) AS 'Name',          => create new variables
        CONCAT(city, ", ", state) AS 'Hometown'
FROM students;

SELECT DISTINCT state                               => Distinct eliminate duplicates
FROM students
ORDER BY state;

SELECT sex, COUNT(*)                   =>  count by group
FROM students
GROUP BY sex;

SELECT  varname   FROM   Datatable
        WHERE  function( varname ) < 2 OR  varname  = ''char''
        WHERE   varname  IS (NOT) NULL                    =>  check valeur (non) nulle
        WHERE first_name LIKE 'D%' OR last_name LIKE '%n'  =>  if first name start with D
        WHERE first_name LIKE '___y'                      => _ match any single char
        WHERE birth_date
                BETWEEN '1960-1-1' AND '1970-1-1';
        WHERE first_name IN ('Bobby', 'Lucy', 'Andy')
        ORDER BY  varname DESC,  varname ASC
        LIMIT 5, 10                                       => display results 5 to 10
```

<, >, =<, =>, =, !=	OR, ||	AND, &&	NOT, !

YEAR()	MONTH()	DAY()	Min()
MAX()	SUM()	AVg()	

```
SELECT state, COUNT(state) AS 'Amount'
FROM students
GROUP BY state
HAVING Amount > 1;            => HAVING allows you to narrow the results after the query is executed

SELECT *
FROM scores
WHERE student_id = 4;

UPDATE scores SET score=25
WHERE student_id=4 AND test_id=3;            => Use UPDATE to change a value in a row
```

- **Adding or deleting a column from a table**

ALTER TABLE cats ADD gender CHAR(1) AFTER name;	
ALTER TABLE cats DROP gender;	
ALTER TABLE score CHANGE event_id test_id INT UNSIGNED NOT NULL;	Change the name of event_id in score to test_id
ALTER TABLE absences MODIFY COLUMN test_taken ENUM('T','F') NOT NULL DEFAULT 'F';	You can change the data type with ALTER and MODIFY COLUMN

TINYINT	127 to -128
INT	2^31to -2^31
BIGINT	2^63to -2^63
FLOAT	Decimal spaces : 1.1^{E}38 tot -1.1^E 38
DOUBLE	Decimal spaces : 1.7^{E}308 tot -1.7^E 308
CHAR	Character string with fixed length
VARCHAR	Character string with variable length
BLOP	Can contain 2^16 bytes of data
EMUM	Character string that has a limited number of total values, which you must define
SET	A list of legal possible character string.UNlike EMUM, a SET can contain multiple values in comparison to the one legal value

- **Combining dataset**

```
SELECT scores.student_id, tests.date, scores.score, tests.maxscore
FROM tests, scores
WHERE date = '2014-08-25'
AND tests.test_id = scores.test_id;
```

⇨ To combine data from multiple tables you can perform a JOIN by matching up common data like we did here with the test ids

```
SELECT students.student_id, CONCAT(students.first_name, " ", students.last_name) AS Name,
        COUNT(absences.date) AS Absences
FROM students LEFT JOIN absences
        ON students.student_id = absences.student_id
GROUP BY students.student_id;
```

⇨ If we need to include all information from the table listed first "FROM students", even if it doesn't exist in the table on the right "LEFT JOIN absences", we can use a LEFT JOIN.

```
SELECT students.first_name, students.last_name, scores.test_id, scores.score
FROM students
INNER JOIN scores
        ON students.student_id=scores.student_id
WHERE scores.score <= 15
ORDER BY scores.test_id;
```

⇨ An INNER JOIN gets all rows of data from both tables if there is a match between columns in both tables