**NoSQL** :

En [informatique](https://fr.wikipedia.org/wiki/Informatique) et en [bases de données](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es), **NoSQL** désigne une famille de [systèmes de gestion de base de données](https://fr.wikipedia.org/wiki/Syst%C3%A8me_de_gestion_de_base_de_donn%C3%A9es) (SGBD) qui s'écarte du [paradigme](https://fr.wikipedia.org/wiki/Paradigme) classique des [bases relationnelles](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es_relationnelle). L'explicitation la plus populaire de l'acronyme est *Not only SQL* (« pas seulement SQL » en anglais) même si cette interprétation peut être discutée[1](https://fr.wikipedia.org/wiki/NoSQL#cite_note-1).

La définition exacte de la famille des SGBD **NoSQL** reste sujette à débat. Le terme se rattache autant à des caractéristiques techniques qu'à une génération historique de SGBD qui a émergé autour des années 2010[2](https://fr.wikipedia.org/wiki/NoSQL#cite_note-2). D'après [Pramod J. Sadalage](https://fr.wikipedia.org/w/index.php?title=Pramod_J._Sadalage&action=edit&redlink=1) et [Martin Fowler](https://fr.wikipedia.org/wiki/Martin_Fowler), la raison principale de l'émergence et de l'adoption des SGBD NoSQL serait le développement des [centres de données](https://fr.wikipedia.org/wiki/Centre_de_donn%C3%A9es) et la nécessité de posséder un paradigme de bases de données adapté à ce modèle d'infrastructure matérielle[3](https://fr.wikipedia.org/wiki/NoSQL#cite_note-distilled-3).

L'architecture machine en [clusters](https://fr.wikipedia.org/wiki/Grappe_de_serveurs) induit une structure logicielle distribuée fonctionnant avec des agrégats répartis sur différents serveurs permettant des accès et modifications concurrentes mais imposant également de remettre en cause de nombreux fondements de l'architecture SGBD relationnelle traditionnelle, notamment les [propriétés ACID](https://fr.wikipedia.org/wiki/Propri%C3%A9t%C3%A9s_ACID).

**fonctionnalité of mongodb:**

MongoDB est une base de données NoSQL orientée documents. Comme nous le verrons, l’ensemble du système tourne autour de cette gestion de documents, y compris le langage d’interrogation, ce qui en fait son point fort. Nous allons nous attaquer dès maintenant à la mise en place d’un serveur Mongo et comment intégrer vos données dans cet environnement.

#### **Installation**

Pour bien fonctionner, une base *MongoDB* a besoin de trois choses :

* L’installation du serveur, que vous pouvez télécharger [ici](https://www.mongodb.com/download-center?jmp=nav#community) (version 3.4.7)

L’installation sous *Windows* est assez simple (appuyez sur suivant).

Sous Mac, décompressez l’archive et déplacez le répertoire dans vos applications “/Applications”

Sous Debian/Ubuntu, utilisez la commande “*sudo apt install mongodb*”.

Le serveur sera alors installé dans le répertoire que je nommerai : **$MONGO**.

Sous *Windows* : C:\Program Files\MongoDB\Server\3.4\

Sous *Linux* : /opt/local/bin/

Sous *Mac* : /Applications

#### **Création d'une collection**

Ca y est, nous pouvons manipuler la base de données et y ajouter une collection de documents. Une collection c’est l’équivalent d’une table pour une base de données relationnelle. Mais dans notre cas, une collection n’a pas de schéma donnant la structure d’une collection. Comme nous le verrons par la suite un document n’a pas de structure fixe, de fait, une base de données gère une ‘collection’ de documents, et non une table.

Créons tout d’abord une base de données ‘ma\_bd’ (bouton droit sur la connexion “Create database”), puis sur les collections de cette base, créer une collection “test” (bouton droit sur “Collections(0)”)

**sql:**

SQL est un langage informatique normalisé servant à exploiter des bases de données relationnelles. La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles. Outre le langage de manipulation des données .

**fonctionnalité of sql:**

[Structured Query Language](https://www.edureka.co/blog/what-is-sql/) aka SQL is used to handle data in databases. It provides various in-built functions and [commands](https://www.edureka.co/blog/sql-commands) to access and manage databases according to our requirements. In this article on SQL Functions, I will discuss the various in-built functions to perform different types of calculation on the data.

The following topics will be covered in this article:

* [What are SQL functions?](https://www.edureka.co/blog/sql-functions#functions)
* [Aggregate Functions](https://www.edureka.co/blog/sql-functions#aggregate)
  1. [SUM()](https://www.edureka.co/blog/sql-functions#sum)
  2. [COUNT()](https://www.edureka.co/blog/sql-functions#count)
  3. [AVG()](https://www.edureka.co/blog/sql-functions#avg)
  4. [MIN()](https://www.edureka.co/blog/sql-functions#min)
  5. [MAX()](https://www.edureka.co/blog/sql-functions#max)
  6. [FIRST()](https://www.edureka.co/blog/sql-functions#first)
  7. [LAST()](https://www.edureka.co/blog/sql-functions#last)
* [Scalar Functions](https://www.edureka.co/blog/sql-functions#scalar)
  1. [LCASE()](https://www.edureka.co/blog/sql-functions#lcase)
  2. [UCASE()](https://www.edureka.co/blog/sql-functions#ucase)
  3. [LEN()](https://www.edureka.co/blog/sql-functions#len)
  4. [MID()](https://www.edureka.co/blog/sql-functions#mid)
  5. [ROUND()](https://www.edureka.co/blog/sql-functions#round)
  6. [NOW()](https://www.edureka.co/blog/sql-functions#now)
  7. [FORMAT()](https://www.edureka.co/blog/sql-functions#format)

Before we delve into the different types of functions offered by SQL, let us understand what are functions.

## What are functions?

Functions are methods used to perform [data operations](https://www.edureka.co/blog/sql-operators/). SQL has many in-built functions used to perform string concatenations, mathematical calculations etc.

SQL functions are categorized into the following two categories:

1. Aggregate Functions
2. Scalar Functions

Let us look into each one of them, one by one.

## Aggregate SQL Functions

The Aggregate Functions in SQL perform calculations on a group of values and then return a single value. Following are a few of the most commonly used Aggregate Functions:

| Function | Description |
| --- | --- |
| SUM() | Used to return the sum of a group of values. |
| COUNT() | Returns the number of rows either based on a condition, or without a condition. |
| AVG() | Used to calculate the average value of a numeric column. |
| MIN() | This function returns the minimum value of a column. |
| MAX() | Returns a maximum value of a column. |
| FIRST() | Used to return the first value of the column. |
| LAST() | This function returns the last value of the column. |

Let us look into each one of the above functions in depth. For your better understanding, I will be considering the following table to explain to you all the examples.

| StudentID | StudentName | Marks |
| --- | --- | --- |
| 1 | Sanjay | 64 |
| 2 | Varun | 72 |
| 3 | Akash | 45 |
| 4 | Rohit | 86 |
| 5 | Anjali | 92 |

### SUM()

Used to return a total sum of numeric column which you choose.

#### Syntax:

| 1  2 | SELECT SUM(ColumnName)  FROM TableName; |
| --- | --- |

#### Example:

Write a query to retrieve the sum of marks of all students from the Students table.

| 1  2 | SELECT SUM(Marks)  FROM Students; |
| --- | --- |

#### Output:

| 1 | 359 |
| --- | --- |

### COUNT()

Returns the number of rows present in the table either based on some condition or without any condition.

#### Syntax:

| 1  2  3 | SELECT COUNT(ColumnName)  FROM TableName  WHERE Condition; |
| --- | --- |

#### Example:

Write a query to count the number of students from the Students table.

| 1  2 | SELECT COUNT(StudentID)  FROM Students; |
| --- | --- |

#### Output:

| 1 | 5 |
| --- | --- |

#### Example:

Write a query to count the number of students scoring marks > 75 from the Students table.

| 1  2  3 | SELECT COUNT(StudentID)  FROM Students  WHERE Marks >75; |
| --- | --- |

Output:

| 1 | 2 |
| --- | --- |

### AVG()

This function is used to return the average value of a numeric column.

#### Syntax:

| 1  2 | SELECT AVG(ColumnName)  FROM TableName; |
| --- | --- |

#### Example:

Write a query to calculate the average marks of all students from the Students table.

| 1  2 | SELECT AVG(Marks)  FROM Students; |
| --- | --- |

#### Output:

| 1 | 71.8 |
| --- | --- |

### MIN()

Used to return the minimum value of a numeric column.

**difference between sql and mongo DB:**

SQL databases are used to store structured data while NoSQL databases like MongoDB are used to save unstructured data. MongoDB is used to save unstructured data in JSON format. MongoDB does not support advanced analytics and joins like SQL databases support.