**NoSQL**

**Introduction:** NoSQL databases (aka "not only SQL") are non-tabular databases and store data differently than relational tables. NoSQL databases come in a variety of types based on their data model. The main types are document, key-value, wide-column, and graph. They provide flexible schemas and scale easily with large amounts of data and high user loads.

### **Types of NoSQL databases:** Over time, four majors [types of NoSQL databases](https://www.mongodb.com/scale/types-of-nosql-databases) emerged: document databases, key-value databases, wide-column stores, and graph databases.

* Document databases store data in documents similar to JSON (JavaScript Object Notation) objects. Each document contains pairs of fields and values. The values can typically be a variety of types including things like strings, numbers, Booleans, arrays, or objects.
* Key-value databases are a simpler type of database where each item contains keys and values.
* Wide-column stores store data in tables, rows, and dynamic columns.
* Graph databases store data in nodes and edges. Nodes typically store information about people, places, and things, while edges store information about the relationships between the nodes.

### **Why NoSQL?**

### NoSQL databases are used in nearly every [industry](https://www.mongodb.com/industries). Use cases range from the highly critical (e.g., storing [financial data](https://www.mongodb.com/industries/financial-services) and [healthcare records](https://www.mongodb.com/industries/healthcare)) To the more fun and frivolous (e.g., [storing IoT readings from a smart kitty litter box](https://www.mongodb.com/developer/how-to/internet-of-toilets/)).

### **When should NoSQL be used?**

When deciding which database to use, decision-makers typically find one or more of the following factors lead them to selecting a NoSQL database:

* Fast-paced Agile development
* Storage of structured and semi-structured data
* Huge volumes of data
* Requirements for scale-out architecture
* Modern application paradigms like microservices and real-time streaming

# **MongoDB advantages over RDBMS**

In recent days, MongoDB is a new and popularly used database. It is a document based, non-relational database provider.

Although it is 100 times faster than the traditional database but it is early to say that it will broadly replace the traditional RDBMS. But it may be very useful in term to gain performance and scalability.

A Relational database has a typical schema design that shows number of tables and the relationship between these tables, while in MongoDB there is no concept of relationship.

Mongo commands

## ****Log Into MongoDB****

The following command can be used to log into the MongoDB database.

## ****Show All Databases****

Once logged in as a user with the appropriate role as userAdmin or userAdminAnyDatabase, the user can see all the databases using a command

## ****Select Database to Work With****

To start working with a particular database

## ****Authenticate and Log Out From Database****

When switching to a different database using the use dbName command, the user is required to authenticate using a valid database user for that database.

## ****Create a Collection****

The following command can be used to create a collection.

## ****Insert a Document in a Collection****

Once a collection is created, the next step is to insert one or more documents. Following is a sample command for inserting a document in a collection.

## ****Save or Update Document****

The save command can be used to either update an existing document or insert a new one depending on the document parameter passed to it. If the \_id passed matches an existing document, the document is updated. Otherwise, a new document is created. Internally, thesave method uses either the insert or the update command.

1

## ****Display Collection Records****

The following commands can be used to retrieve collection records:

1

//

2

// Retrieve all records

3

//

4

db.<collectionName>.find();

5

//

6

// Retrieve limited number of records; Following command will print 10 results;

7

//

8

db.<collectionName>.find().limit(10);

9

//

10

// Retrieve records by id

11

//

12

db.<collectionName>.find({"\_id": ObjectId("someid")});

13

//

14

// Retrieve values of specific collection attributes by passing an object having

15

// attribute names assigned to 1 or 0 based on whether that attribute value needs

16

// to be included in the output or not, respectively.

17

//

18

db.<collectionName>.find({"\_id": ObjectId("someid")}, {field1: 1, field2: 1});

19

db.<collectionName>.find({"\_id": ObjectId("someid")}, {field1: 0}); // Exclude field1

20

//

21

// Collection count

22

//

23

db.<collectionName>.count();

## ****Administrative Commands****

Following are some of the administrative commands that can be helpful in finding collection details such as storage size, total size, and overall statistics.

1

//

2

// Get the collection statistics

3

//

4

db.<collectionName>.stats()

5

db.printCollectionStats()

6

//

7

// Latency statistics for read, writes operations including average time taken for reads, writes

8

// and related umber of operations performed

9

//

10

db.<collectionName>.latencyStats()

11

//

12

// Get collection size for data and indexes

13

//

14

db.<collectionName>.dataSize() // Size of the collection

15

db.<collectionName>.storageSize() // Total size of document stored in the collection

16

db.<collectionName>.totalSize() // Total size in bytes for both collection data and indexes

17

db.<collectionName>.totalIndexSize() // Total size of all indexes in the collection