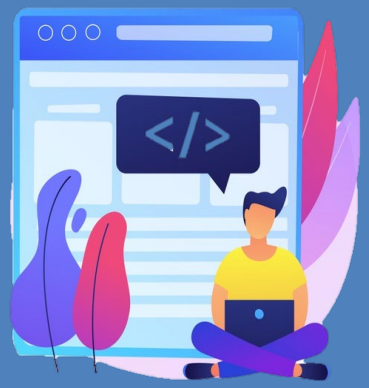


MONGODB FOR ABSOLUTE BEGINNERS

Agenda

1. Introduction
2. Introduction to NoSQL
3. SQL vs NoSQL
4. Installation
5. Create Databases
6. Create Collection
7. Create Documents

Introduction

Introduction

Front end languages

Back end languages

Database languages

Introduction

A database Management System provides the mechanism to store and retrieve the data. There are different kinds of database Management Systems:

1. RDBMS (Relational Database Management Systems)

2. OLAP (Online Analytical Processing)

3. NoSQL (Not only SQL)

Introduction to NoSQL

Limitations of RDBMS

1. In relational database we need to define structure and schema of data first and then only we can process the data.

2. Relational database systems provides consistency and integrity of data by enforcing ACID properties (Atomicity, Consistency, Isolation and Durability). However in most of the other cases these properties are significant performance overhead and can make your database response very slow.

3. Most of the applications store their data in JSON format and RDBMS don’t provide you a better way of performing operations such as create, insert, update, delete etc on this data.

What is NoSQL Database?

Not Only SQL (NoSQL) or non relational databases provides a mechanism for storage and retrieval of data other than tabular relations model used in relational databases. NoSQL database doesn't use tables for storing data. It is generally used to store big data and real-time web applications.

Why NoSQL Database?

NoSQL databases were created in response to the limitations of traditional relational database technology.

When compared against relational databases, NoSQL databases are more scalable and provide superior performance, and their data model addresses several shortcomings of the relational model.

The advantages of NoSQL include being able to handle:

- Large volumes of structured, semi-structured, and unstructured data
- Agile sprints, quick iteration, and frequent code pushes
- Object-oriented programming that is easy to use and flexible
- Efficient, scale-out architecture instead of expensive, monolithic architecture

When to use NoSQL Database?

- When you want to store and retrieve huge amount of data.
- The relationship between the data you store is not that important
- The data is not structured and changing over time
- Constraints and Joins support is not required at database level
- The data is growing continuously and you need to scale the database regular to handle the data.

Where to use NoSQL Database?

- Data isn't relational (e.g. Documents)
- Too much data to fit in a relational database

SQL vs NoSQL

SQL vs NoSQL

SQL Database	NoSQL Database
SQL databases are primarily called RDBMS or Relational Databases	NoSQL databases are primarily called as Non-relational or distributed database
SQL databases are table based databases	NoSQL databases can be document based, key-value pairs, graph databases
MySQL uses SQL to query database	MongoDB uses BSON to query database
Developed in the 1970s with a focus on reducing data duplication	Developed in the late 2000s with a focus on scaling and allowing for rapid application change driven by agile and DevOps practices.
Oracle, MySQL, Microsoft SQL Server, and PostgreSQL	Document: MongoDB and CouchDB, Key-value: Redis and DynamoDB, Wide-column: Cassandra and HBase, Graph: Neo4j and Amazon Neptune

SQL vs NoSQL

FEATURE DESCRIPTION	COLUMN TITLE	COLUMN TITLE	COLUMN TITLE	COLUMN TITLE
First Row	●	●	●	●
Second Row	●	●	●	●
Third Row	●	●	●	●
Rowmance	●	●	●	●
Berow	●	●	●	●
Rowanow	●	●	●	●

```
{
        "_id" : ObjectId("5c2f30f4ee0336a91328dab0"),
        "fname" : "Vivan",
        "lname" : "Shirag",
        "course" : "Java App Dev",
        "duration" : "3 Months"
}
{
        "_id" : ObjectId("5c2f3737ee0336a91328dab1"),
        "fname" : "Subiya",
        "lname" : "Siraj",
        "course" : "Dot Net",
        "duration" : "2Months"
}
{
        "_id" : ObjectId("5c2f3737ee0336a91328dab2"),
        "fname" : "Niyaz",
        "lname" : "Ahmed",
        "course" : "Java",
        "duration" : "4Months"
}
```

Installation

Installation

- C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe --version
- C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe --version
- C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe
- C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe
- mongo

Create Database

Create Database

Once you are on MongoDb shell, Use the following command to create a database

use database_name;

if the database doesn't exists, above command creates a new database otherwise opens the existing one.

Eg: use greatlearning

```
>
> use greatlearning
switched to db greatlearning
>
```

Note:
In MongoDB, databases hold collections of documents

Delete Database

Use the following command to delete a database

db.dropDatabase()

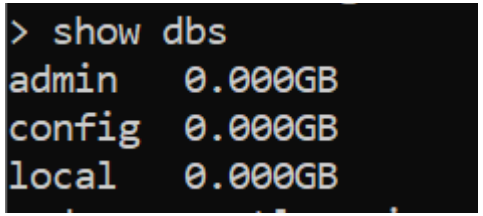
```
> db.dropDatabase()
{ "ok" : 1 }
>
```

List all Databases

- To list down all the databases, use the command below

  show dbs

- This command lists down all the databases and their size on the disk.



Note:

As you can see that the database "greatlearning" that we have created is not present in the list of all the databases. This is because a database is not created until you save a document in it

Create Collection

Collection

- MongoDB stores documents in collections. Collections are analogous to tables in relational databases
- A collection exists within a single database. Collections do not enforce a schema.
- A collection may store documents those which are not same in structure because its schema free database.

Create Collection – Method 1

- In MongoDB you need not to create collection before you insert document in it. With a single command you can insert a document in the collection and the MongoDB creates that collection on the fly.

  Syntax: *db.collection_name.insert({key:value, key:value…})*

- Eg: To create a collection in the database iprimeddb, we use the following command.

  db.gla.insert({name:"CSS",source:"GLA", type:"Front end"});

```
> db.greatlearning.insertOne({name:"HTML", source:"GreatLearningAcademy", type:"Front end", videos:5, active:true})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("6050421c8300d7d14ad1d46d")
}
>
```

```
> db.gla.insert({name:"CSS", source: "GLA", type: "Front end"})
WriteResult({ "nInserted" : 1 })
>
```

Create Collection – Method 2

We can also create collection before we actually insert data in it. This method provides you the options that you can set while creating a collection.

Syntax: *db.createCollection(name, options)*

- name is the collection name
- options is an optional field that we can use to specify certain parameters such as size, max number of documents etc. in the collection.

Eg: db.createCollection("PGprograms");

```
> db.createCollection("PGprograms");
{ "ok" : 1 }
>
```

```
> db
greatlearning
> show collections
PGprograms
gla
greatlearning
>
```

Deleting the Collection

To drop a collection , first connect to the database in which you want to delete collection and then type the following command to delete the collection:

db.collection_name.drop()

Note: Once you drop a collection all the documents and the indexes associated with them will also be dropped. To preserve the indexes we use remove() function that only removes the documents in the collection but doesn't remove the collection itself and the indexes created on it.

Eg: db.gla.drop();

```
> db.gla.drop()
true
> db.PGprograms.drop()
true
> show collections
greatlearning
>
```

Create Documents

Documents

MongoDB stores data records as BSON documents. BSON is a binary representation of JSON documents.

MongoDB documents are composed of field-and-value pairs and have the following structure:

```
{
  field1: value1,
  field2: value2,
  field3: value3,
  ...
  fieldN: valueN
}
```

Collection and Documents

```
> show dbs
admin          0.000GB
config         0.000GB
greatlearning  0.000GB
local          0.000GB
> db
greatlearning
> show collections
greatlearning
> db.greatlearning.find()
{ "_id" : ObjectId("6050421c8300d7d14ad1d46d"), "name" : "HTML", "source" : "GreatLearningAcademy", "type" : "Front end", "videos" : 5, "active" : true }
>
```

```
> db.greatlearning.find().pretty()
{
        "_id" : ObjectId("6050421c8300d7d14ad1d46d"),
        "name" : "HTML",
        "source" : "GreatLearningAcademy",
        "type" : "Front end",
        "videos" : 5,
        "active" : true
}
>
```

Thank You