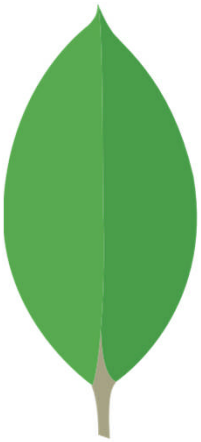


UNIT 5
MongoDB

edunet foundation

Objective

- What is MongoDB
- What is Document based storage?
- Key Features of MongoDB
- Organizations that use MongoDB
- MongoDB: CAP approach
- CRUD operations
- MongoDB vs SQL Databases
- Aggregated & Map reduce functionality
- Indexing in MongoDB
- Replication of data
- Consistency of data

What is MongoDB?

- MongoDB is a NoSQL database which stores the data in form of key-value pairs. It is an Open Source, Document Database which provides high performance and scalability along with data modelling and data management of huge sets of data in an enterprise application.
- MongoDB also provides the feature of Auto-Scaling. Since MongoDB is a cross platform database and can be installed across different platforms like Windows, Linux etc.

edunet foundation

What is Document based storage?

- A Document is nothing but a data structure with name-value pairs like in JSON. It is very easy to map any custom Object of any programming language with a MongoDB Document. For example : Student object has attributes name, roll no and subjects, where subjects is a List.
- Document for Student in MongoDB will be like:

```
{
  name : "Stduytonight",
  rollno : 1,
  subjects : ["C Language", "C++", "Core Java"]
}
```

Key Features of MongoDB

- Apart from most of the NoSQL default features, MongoDB does bring in some more, very important and useful features :

MongoDB provides high performance. Input/Output operations are lesser than relational databases due to support of embedded documents(data models) and Select queries are also faster as Indexes in MongoDB supports faster queries.

MongoDB has a rich Query Language, supporting all the major CRUD operations. The Query Language also provides good Text Search and Aggregation features.

Auto Replication feature of MongoDB leads to High Availability. It provides an automatic failover mechanism, as data is restored through backup(replica) copy if server fails.

Key Features of MongoDB

- Apart from most of the NoSQL default features, MongoDB does bring in some more, very important and useful features :

Sharding is a major feature of MongoDB. Horizontal Scalability is possible due to sharding.

MongoDB supports multiple Storage Engines. When we save data in form of documents(NoSQL) or tables(RDBMS) who saves the data? It's the Storage Engine. Storage Engines manages how data is saved in memory and on disk.

Organizations that use MongoDB

- Below are some of the big and notable organizations which are using MongoDB as database for most of their business applications.

Overview of MongoDB

- MongoDB consists of a set of databases. Each database again consists of Collections. Data in MongoDB is stored in collections. The below figure depicts the typical database structure in MongoDB.

Database in MongoDB

- Database in MongoDB is nothing but a container for collections. We will learn how to create a new Database, drop a Database and how to use an existing Database in the coming lessons.

Collections in MongoDB

- Collection is nothing but a set of MongoDB documents. These documents are equivalent to the row of data in tables in RDBMS. But collections in MongoDB do not relate to any set schema as compared to RDBMS. Collections are a way of storing related data. Being schemaless, any type of Document can be saved in a collection, although similarity is recommended for index efficiency. Documents can have a maximum size of 4MB

Document in MongoDB

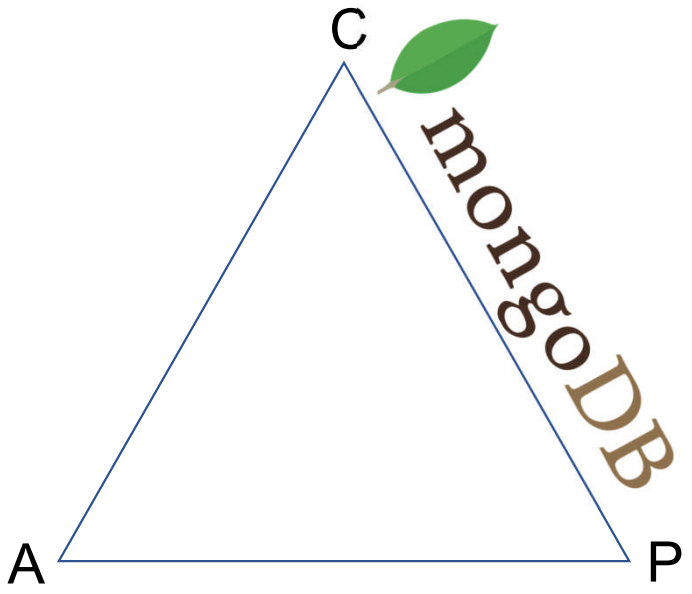
- Document in MongoDB is nothing but the set of key-value pairs. These documents will have dynamic schema which means that the documents in the same collection do not need to possess the same set of fields.

edunet foundation

MongoDB: CAP approach

Focus on Consistency and Partition tolerance

- Consistency
  - all replicas contain the same version of the data
- Availability
  - system remains operational on failing nodes
- Partition tolerance
  - multiple entry points
  - system remains operational on system split



CRUD operations

CRUD operations *create*, *read*, *update*, and *delete* documents.

Create Operations

- Create or insert operations add new documents to a collection. If the collection does not currently exist, insert operations will create the collection.
- Create
  - db.collection.insert( )
  - db.collection.save( )
  - db.collection.update(, , { upsert: true })

CRUD operations

Read Operations

- Read operations retrieve documents from a collection; i.e. query a collection for documents. MongoDB provides the following methods to read documents from a collection:
- Read
  - db.collection.find(,)
  - db.collection.findOne(,)

CRUD operations

Update Operations

- Update operations modify existing documents in a collection. MongoDB provides the following methods to update documents of a collection:
- Update
  - db.collection.update(, ,)

edunet foundation

CRUD operations

Delete Operations

- Delete operations remove documents from a collection. MongoDB provides the following methods to delete documents of a collection:
- Delete
  - db.collection.remove(,)

MongoDB vs SQL Databases

SQL Database	NoSQL Database (MongoDB)
Relational database	Non-relational database
Supports SQL query language	Supports JSON query language
Table based	Collection based and key-value pair
Row based	Document based
Column based	Field based
Contains schema which is predefined	Contains dynamic schema

edunet foundation

Aggregated functionality

Aggregation framework provides SQL-like aggregation functionality

- Pipeline documents from a collection pass through an aggregation pipeline, which transforms these objects as they pass through
- Expressions produce output documents based on calculations performed on input documents
- Example db.parts.aggregate ({$group : {_id: type, totalquantity : { $sum: quanity} } })

Map reduce functionality

- Performs complex aggregator functions given a collection of keys, value pairs
- Must provide at least a map function, reduction function and a name of the result set
- db.collection.mapReduce(, , { out: , query: , sort: , limit: , finalize: , scope: , jsMode: , verbose: })
- db.collection.mapReduce(, , { out: , query: , sort: , limit: , finalize: , scope: , jsMode: , verbose: })

edunet foundation

Indexing in MongoDB

- Typically used for frequently used queries
- Necessary when the total size of the documents exceeds the amount of available RAM.
- Only 1 index can be used by the query optimizer when retrieving data
- Index covers a query - match the query conditions and return the results using only the index;

Types of Indexes in MongoDB

Index Type	Description
Single field index	Used to create an index on a single field and it can be a user defined as well apart from the default _id one.
Compound index	MongoDB supports the user-defined indexes on multiple fields.
Multi key index	MongoDB uses multi key indexes basically to store the arrays. MongoDB creates a separate index for each element in an array. MongoDB intelligently identifies to create a multi key index if the index contains elements from an array.
Geospatial index	Used to support the queries required for the geospatial coordinate data.
Text index	This index is used to search for a string content in a collection
Hashed index	Used for hash based Sharding

Replication of data

- Ensures redundancy, backup, and automatic failover
  - Recovery manager in the RDMS
- Replication occurs through groups of servers known as replica sets
  - Primary set – set of servers that client tasks direct updates to
  - Secondary set – set of servers used for duplication of data
  - At the most can have 12 replica sets
    - Many different properties can be associated with a secondary set i.e. secondary-only, hidden delayed, arbiters, non-voting
  - If the primary set fails the secondary sets ‘vote’ to elect the new primary set

edunet foundation

Consistency of data

- All read operations issued to the primary of a replica set are consistent with the last write operation
  - Reads to a primary have strict consistency
    - Reads reflect the latest changes to the data
  - Reads to a secondary have eventual consistency
    - Updates propagate gradually
  - If clients permit reads from secondary sets – then client may read a previous state of the database
  - Failure occurs before the secondary nodes are updated
    - System identifies when a rollback needs to occur
    - Users are responsible for manually applying rollback changes

Reference

- https://www.mongodb.com/docs/manual/core/views/
- https://www.geeksforgeeks.org/mongodb-an-introduction/
- https://www.mongodb.com/developer/how-to/SQL-to-Aggregation-Pipeline/
- https://www.mongodb.com/docs/compass/current/import-export/#export-data-from-a-collection