mongoDB

Introduction

Prof. P. M. Jadav
Associate Professor
Computer Engineering Department
Faculty of Technology
D. D. University, Nadiad

Background (NoSQL)

- NoSQL databases are a category of databases that provide a flexible, scalable, and high-performance alternative to traditional relational databases
- The term "NoSQL" stands for **"Not Only SQL"**, meaning they can store data in ways other than the traditional tabular format used by SQL databases

Key Features of NoSQL Databases

- Schema-less: No predefined schema; structure can vary across records
- Scalable: Easily handle large volumes of data across many servers (horizontal scaling)
- High performance: Optimized for fast read/write operations
- Flexible data models: Can store structured, semi-structured, or unstructured data
- Designed for distributed computing: Built for cloud-native applications

NoSQL Database Types

Type	Description	Example
Document-based	Store data as documents (usually JSON, BSON, or XML). Great for hierarchical data.	MongoDB, CouchDB
Key-Value Store	Store data as key-value pairs. Simple and fast.	Redis, DynamoDB
Column-based	Store data in columns instead of rows. Good for analytical queries.	Apache Cassandra, HBase
Graph-based	Store data as nodes and edges. Good for relationships and networks.	Neo4j, Amazon Neptune

Example Use Cases

- MongoDB: E-commerce catalogs, content management systems
- Redis: Caching, real-time analytics
- Cassandra: Time-series data, IoT data
- Neo4j: Social networks, recommendation engines

NoSQL vs. SQL

Feature	SQL	NoSQL
Schema	Fixed schema	Dynamic schema
Scaling	Vertical	Horizontal
Data type	Structured	Semi/unstructured
ACID compliance	Strong	May relax (eventual consistency)
Query language	SQL	Varies (custom APIs or query languages)

Introduction

- Open-source, document based NoSQL database
- developed by Eliot Horowitz and Dwight Merriman in the year 2007
- Stores the data in form of key-value pairs
- High performance and scalable
- Cross-platform database (Windows, Linux etc.)
- name derived from the word *humongous* to support the idea of processing large amount of data.

Document-based database

- data structure with name-value pairs
- Hierarchical data storage
- JSON representation of custom Objects
- Schema-less
- Data is stored in BSON (Binary JSON)

Sample Document

```
{
  "title": "Clean Code",
  "publishedYear": 2008,
  "publisher": "Prentice Hall”,
  "authors": [ { "name": "Robert C. Martin",  "email": "unclebob@ex.com”} ],
  "categories": ["Software Engineering", "Programming"],
  "price": 45.99,
  "ratings":  [3, 4, 3, 5, 4]
  "stock": 30
}
```

Features of MongoDB

- Document-Oriented Storage:
  - Data is stored in documents (similar to JSON) with key-value pairs
- Schema-Less:
  - No fixed schema – each document can have a different structure
  - Easy to evolve the data model over time
- Scalability:
  - Supports horizontal scaling through sharding
  - Can handle large volumes of data efficiently

Features of MongoDB

- **High Availability:**
  - Provides replication through replica sets, ensuring fault tolerance
- **Rich Query Language:**
  - Supports powerful queries, filters, aggregations, and indexing
- **Integrated Tools:**
  - Comes with built-in tools for backup, monitoring, and data visualization (e.g., MongoDB Compass)

Advantages of MongoDB

1. Flexible Document Model (Schema-less)

- Stores data in JSON-like BSON documents
- No need to predefine schemas — each document can have a different structure
- Ideal for agile development and frequent schema changes.

2. High Performance

- Fast read/write op^n due to its in-memory computing and efficient indexing
- Embedded documents reduce the need for joins, improving query speed

3. Easy Horizontal Scalability

- Supports sharding, which splits data across multiple servers
- Suitable for applications with large data volumes or growing workloads

Advantages of MongoDB

4. High Availability with Replication

- Uses replica sets for redundancy and automatic failover
- If the primary node fails, a secondary automatically takes over

5. Powerful Query and Aggregation Framework

- Rich query language supports: Filters, Projections, Sorting, Joins, Aggregation pipelines for data transformation

6. Support for Complex/Nested Data Structures

- Naturally supports arrays and nested documents
- Allows modeling one-to-many and many-to-many relationships inside a single document

MongoDB Terminology

Term	Description
Database	A container for collections
Collection	A group of MongoDB documents (like a table in RDBMS)
Document	A record in BSON format (like a row in RDBMS)
Field	A key-value pair in a document (like a column)

_id (primary key)

- **_id** is a 12 bytes hexadecimal number
- assures the uniqueness of every document
- If you don't provide then MongoDB provides a unique id for every document
- These 12 bytes
  - first 4 bytes for the current timestamp
  - next 3 bytes for machine id
  - next 2 bytes for process id of MongoDB server and
  - remaining 3 bytes are simple incremental VALUE.

MongoDB Data Types

BSON Type Name	Description
Double	64-bit floating point (e.g., 3.14)
String	UTF-8 encoded text (e.g., "MongoDB")
Object	Embedded document (e.g., { name: "Alice", age: 30 })
Array	List of values (e.g., [1, 2, 3] or ["a", "b"])
Binary Data	Binary data (e.g., images, files)
Undefined	Deprecated (exists for legacy reasons)
ObjectId	12-byte unique identifier automatically generated for _id
Boolean	true or false
Date	Date and time (stored as 64-bit integer representing milliseconds since epoch)
Null	Null value

MongoDB Data Types

BSON Type Name	Description
Regular Expression	Regular expression (e.g., /pattern/i)
JavaScript	JavaScript code (without scope)
JavaScript with scope	JavaScript code with scope (rarely used)
32-bit Integer	Signed 32-bit integer
Timestamp	Special internal timestamp used for replication
64-bit Integer	Signed 64-bit integer
Decimal128	128-bit high-precision decimal (for financial and scientific data)
Min Key	Special value that compares lower than all others
Max Key	Special value that compares higher than all others
Symbol	Deprecated, similar to string

MongoDB Server Installation

- Download and install the MongoDB community server from the following url:

  https://www.mongodb.com/try/download/community

- On Windows the mongodb executables will be in the folder something like this:

  C:\Program Files\MongoDB\Server\8.0\bin

MongoDB GUI client (Compass) Installation

- Download and install the MongoDB GUI client/shell from the following url:

  https://www.mongodb.com/products/tools/compass

- On Windows the mongodb compass executables will be in the folder something like this:

C:\Users\<YourUserName>\AppData\Local\MongoDBCompass\

Setup MongoDB Environment

- MongoDB requires a data directory to store all data
- MongoDB's default data directory path is the absolute path \data\db on the drive from which you start MongoDB
- Create this folder by running the following command:

```
mkdir c:\data\db
```

MongoDB Components

Component Set	Binaries
Server	**mongod**.exe
GUI Client/shell	**MongoDBCompass**.exe
Router	mongos.exe

Running MongoDB

- Start the server:

```
mongod
```

- If your data path is different other than default, then run:

```
mongod --dbpath d:\data\db
```

- Start the GUI shell to connect to server:

```
MongoDBCompass
```

References

- https://docs.mongodb.com/manual/