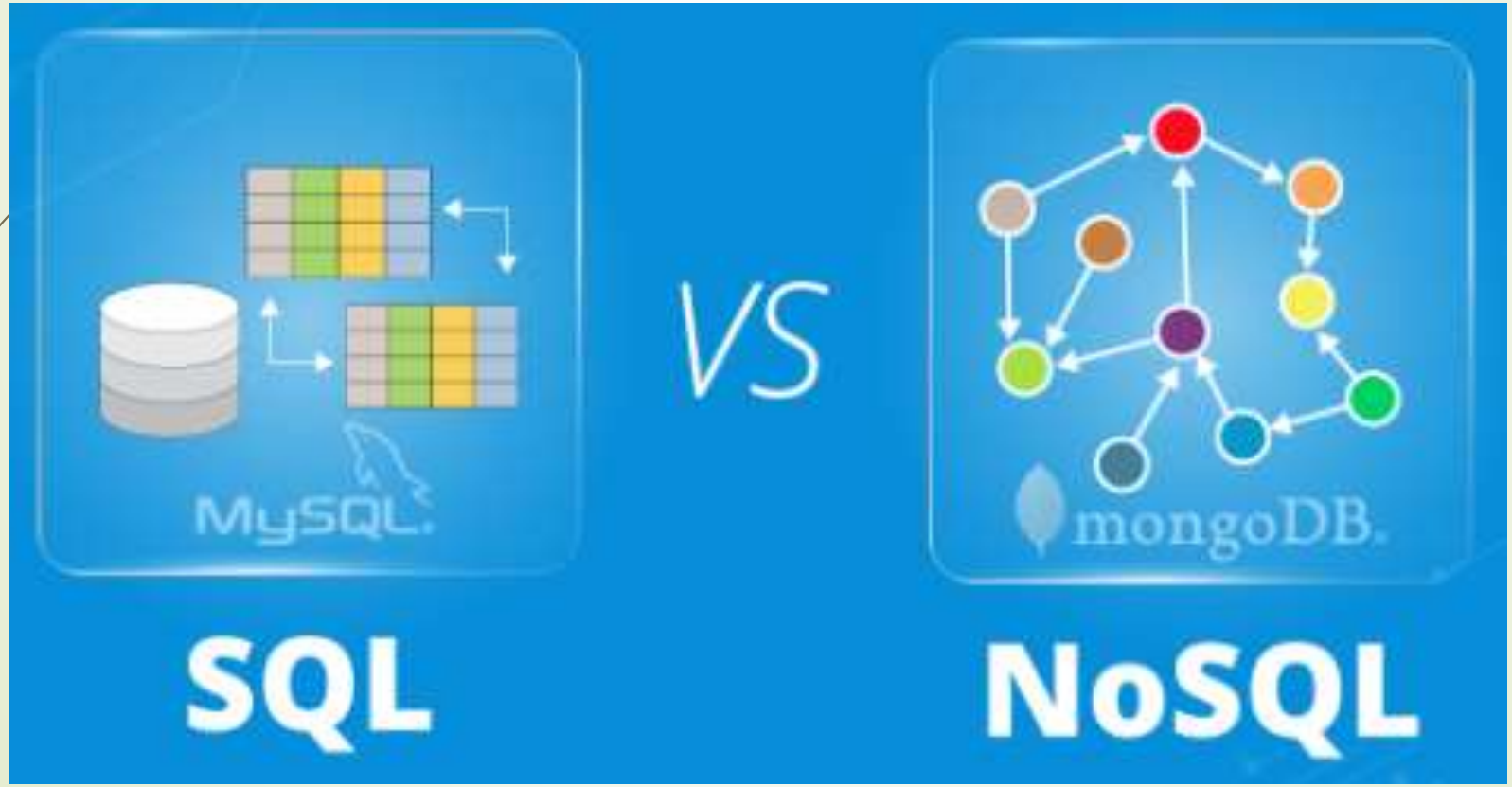
UNIT- IV

NoSQL Data Management – Storage Retrieval Of Data



Main Differences Between SQL and NoSQL Databases

Feature	SQL Databases	NoSQL Databases
Key Focus	Reducing data duplication	Scaling and rapid application change
Data Storage Model	Tables with fixed rows and colums	Document: JSON documents; Key-value: key-value pairs; Wide-column: tables with rows and dynamic columns; Graph: nodes and edges
Schemas	Rigid	Flexible
Data to Object Mapping	Requires ORM (object-relational mapping)	Typically doesn't require ORMs. E.g. MongoDB documents map directly to data structures in popular programming languages.
Scaling	Vertical (scale-up with a larger server)	Horizontal (scale-out across commodity servers)

✓**Horizontal Scaling**, also known as scaling out, involves adding more machines.

✓**Vertical Scaling**, or scaling up, involves increasing the resources of a single machine or instance, such as adding more memory or CPU.

✓**Object-Relational Mapping** (ORM) is a programming technique that allows you to interact with databases using object-oriented programming (OOP) languages.

- NoSQL Data Management is a broad term that refers to the **Storage And Retrieval Of Data** using **Non-Relational Database Management Systems** (DBMS).
- NoSQL databases are designed to handle large volumes of **Unstructured And Semi-structured Data**, and to scale horizontally to handle growing amounts of data.

Common Types Of NoSQL Databases:-

NoSQL databases come in a variety of types, each with its own strengths and weaknesses.

- **Document Databases**: Document databases store data in **Json-like Documents**. for storing **Complex Data Structures.**

Example: User Profiles Or Product Catalogs.

- **Key-value Databases:** Key-value databases store data as a collection of **Key-value Pairs**. storing simple data.

Example: Cache Entries Or Session Data.

- **Wide-column Databases:** Wide-column databases store data in **Tables With Columns Of Varying Widths**.

✓ storing large amounts of data that needs to be queried frequently.

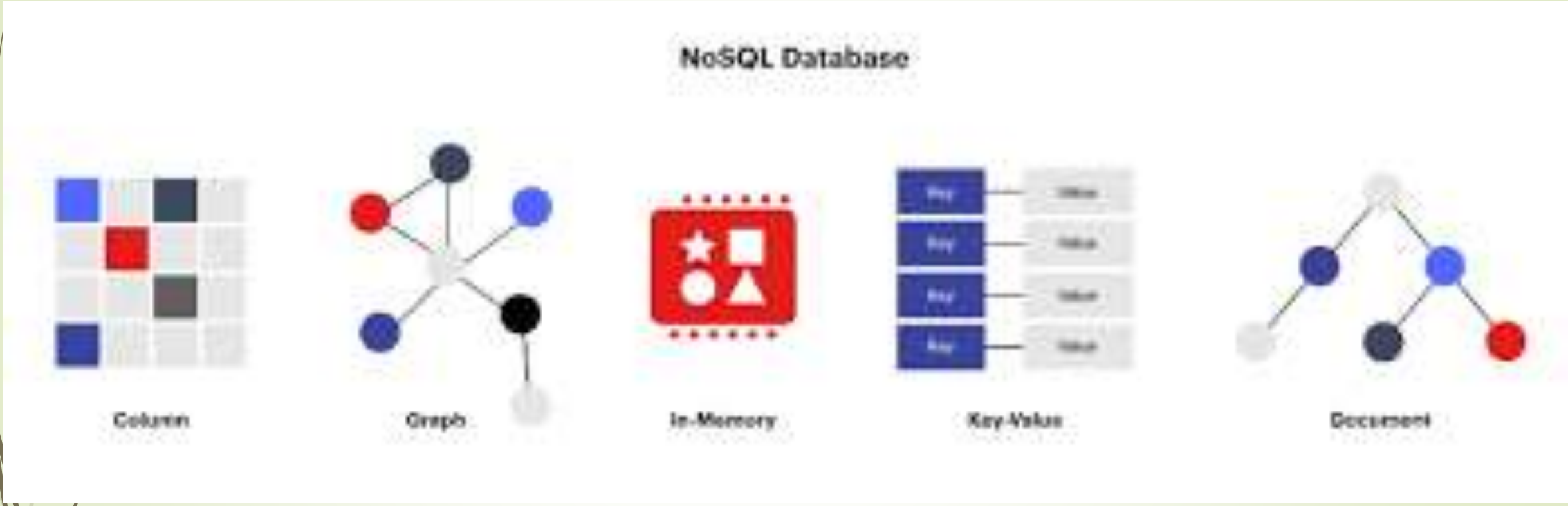
- **Graph DATABASES:** Graph databases store data as **Nodes And Edges**, which allows them to represent complex relationships between data points.

✓ **Example**: Social Networks, Recommendation Systems, And Fraud Detection.

✓NoSQL databases are often used in **Conjunction With Each Other** to provide a complete data management solution.

✓For example : a web application might use a **Document Database** to store user profiles &

A **Key-Value Database** to store session data, and a wide-column database to store product data.


NoSQL Database
Column
Graph
In-Memory
Key-Value
Document

NoSQL In Different Industries:-

- **E-Commerce**: NoSQL databases are used to store product catalogs, customer data, and order history.
- **Social Media**: NoSQL databases are used to store user profiles, social graphs, and posts.
- **Gaming**: NoSQL databases are used to store player data, game state, and leaderboard information.
- **Financial Services**: NoSQL databases are used to store account data, transaction history, and fraud detection data.
- **Internet Of Things** (IoT): NoSQL databases are used to store sensor data, device data, and event data.

Advantages :-

- **Flexibility**: NoSQL databases have flexible data models that can adapt to changes in data structures.
- ✓ This makes them a good choice for storing complex and unstructured data.

- **Scalability**: NoSQL databases can scale horizontally to handle large amounts of data and high user loads.
- ✓ This makes them a good choice for high-performance applications.

- **Performance**: NoSQL databases are often faster than traditional relational databases for certain types of queries.

Dis-Advantages

- **Consistency**: NoSQL databases can sometimes **Sacrifice Consistency In Favor Of Performance Or Scalabilit**y.
- ✓ This may not be suitable for applications where data consistency is critical.

- **Complexity**: NoSQL databases can be more complex to manage than traditional relational databases.
- ✓ This requires specialized knowledge and expertise.

Query Models

- NoSQL databases offer a variety of query models for big data.
- The specific query model that is best for your application will depend on the **Type Of Data You Are Storing And The Types Of Queries You Need To Perform**.

Common Query Models For NoSQL Databases:

- **Key-value Queries**: Key-value queries are the simplest type of NoSQL query.
- ✓ They allow you to retrieve a value based on a given key.
- ✓ Key-value queries are often used for **Caching And Lookup Applications**.
- **Document Queries**: Document queries allow you to query documents based on their content.
- ✓ Document queries can be very complex, and can include support for nested documents and arrays.
- ✓ Document queries are often used for **Web Applications And Content Management Systems.**

- **Column Queries**: Column queries allow you to query columns of data efficiently. Column queries are often used for **Analytical Applications And Data Warehouses**.

- **Graph Queries**: Graph queries allow you to query relationships between data nodes. Graph queries are often used for **Social Networking Applications And Fraud Detection Applications**.

In addition to these basic query models, many NoSQL databases also offer support for more advanced features, such as **Full-text Search And Geospatial Queries**.

Here Are Some Tips For Querying Big Data In NoSQL Databases:

- **Use The Right Data Model**: As mentioned above, the specific query model that is best for your application will depend on the **Type Of Data You Are Storing And The Types Of Queries You Need To Perform**.
- ✓ Make sure to choose a data model that is optimized for the types of queries you will be running.

- **Use Indexes**: Indexes can help to **Improve The Performance Of Your Queries**.
- ✓ Make sure to create indexes on the fields that you query most often.

- **Use Sharding And Replication**: Sharding and replication can help you to **Scale Your Database Horizontally**.
- ✓ If you have a large amount of data or high traffic, you should consider using sharding and replication.

- **Use A Query Planner**: A query planner can help you to **Optimize Your Queries For Performance**.

✓ Many NoSQL databases have built-in query planners, but there are also third-party query planners available.

- **Monitor Your Queries**: It is important to monitor your queries to **Identify Any Performance Bottlenecks**.

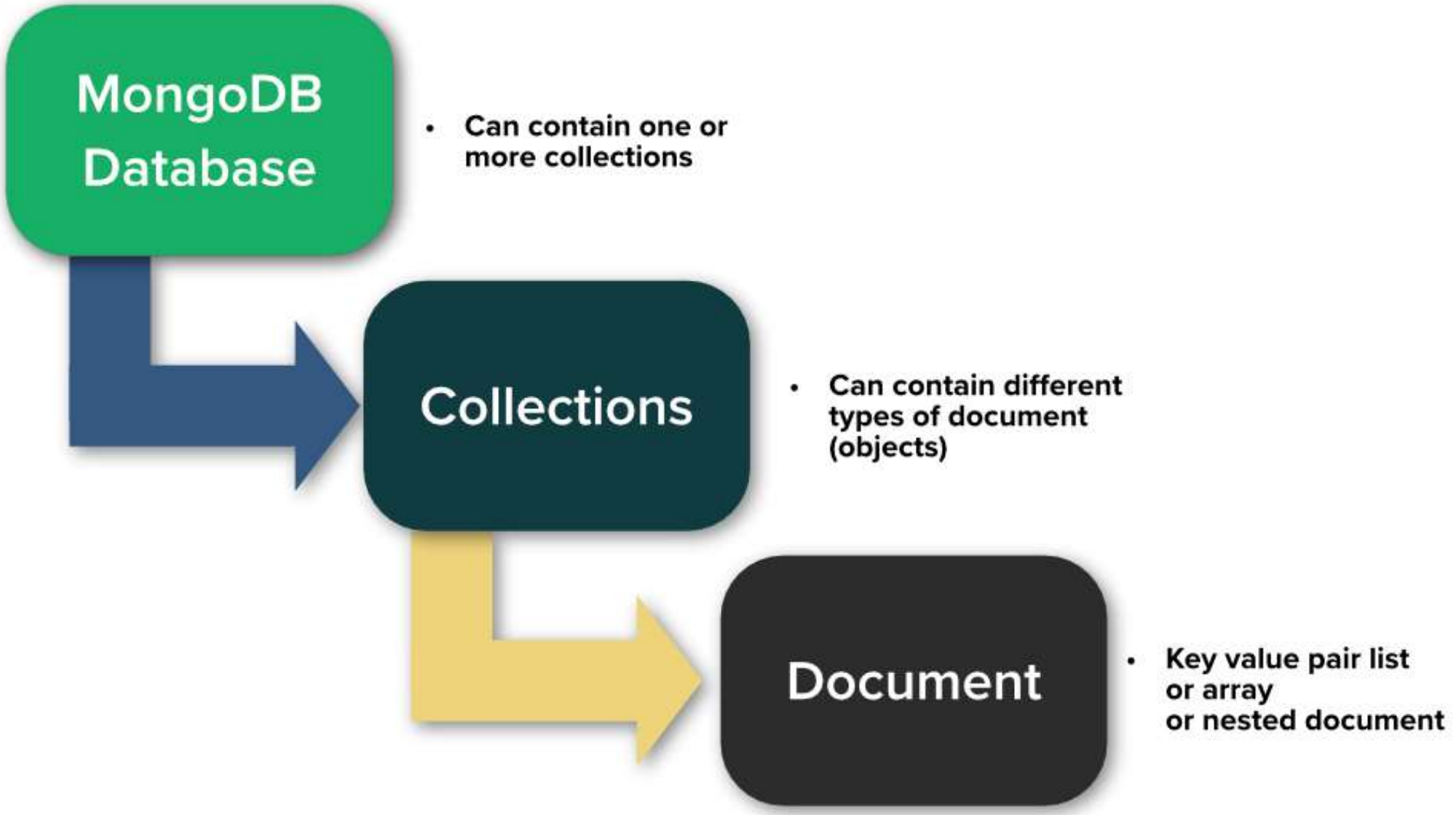
✓ You can use a monitoring tool to track the latency and throughput of your queries.

Mongo DB

- MongoDB is a **Document-Oriented** NoSQL database.
- It stores data in documents, which are similar to **JSON Objects.**
- Documents can have different fields and structures, and can be nested to **Create Complex Data Models**.
- MongoDB is a popular choice for NoSQL data management because it is **Flexible, Scalable, And Easy To Use**.
- It is also well-suited for applications that need to handle large amounts of data or that need to be able to **Scale Quickly**.
- MongoDB is also **Schema-less**, which means that there is no fixed schema for documents.
- This makes it easy to add new fields to documents or change the structure of existing documents without having to modify the database schema.

Benefits Of Using Mongo DB For No SQL Data Management:-

- **Flexibility**: Mongo DB's **Schema-less Design** makes it easy to store and query data of any structure.
- ✓ This is ideal for applications that need to handle a variety of different data types or that need to be able to change their data model quickly.
- **Performance**: MongoDB is designed for performance and can handle high-volume queries and updates.
- ✓ This makes it a good choice for applications that need to be able to respond quickly to user requests.
- **Ease Of Use**: MongoDB is easy to learn and use, even for developers who are new to NoSQL databases. There are also a variety of tools and resources available to help developers get started with MongoDB.
- **Scalability**: MongoDB can be scaled horizontally to handle large amounts of data and traffic. This makes it a good choice for applications that need to be able to scale to meet demand.


MongoDB Database
Can contain one or more collections
Collections
Can contain different types of document (objects)
Document
Key value pair list or array or nested document

Examples Of Mongo DB:-

- **Web Applications**: MongoDB is often used to store and serve data for web applications. This is because it is fast, scalable, and easy to integrate with popular web frameworks.
- **Real-time Applications**: MongoDB is also a good choice for real-time applications, such as chat applications and social media platforms. This is because it can handle high-volume updates and queries in real time.
- **Mobile Applications**: MongoDB is also used to store and sync data for mobile applications. This is because it is lightweight and can be easily deployed on mobile devices.
- ✓ Overall, MongoDB is a powerful and flexible NoSQL database that is well-suited for a wide variety of applications.
- ✓ If you are looking for a NoSQL database that is easy to use, scalable, and performance, then MongoDB is a good option to consider.

Tips For Using Mongo DB For NoSQL Data Management

- **Use The Right Data Model**: Mongo DB's schema-less design gives you a lot of flexibility, but it is important to choose the right data model for your application. This will help you to optimize your database for performance and scalability.
- **Use Indexes**: Indexes can help to improve the performance of your queries. Make sure to create indexes on the fields that you query most often.
- **Use Sharding And Replication**: Sharding and replication can help you to scale your database horizontally. If you have a large amount of data or high traffic, you should consider using sharding and replication.
- **Use A Monitoring Tool**: A monitoring tool can help you to track the performance and health of your database. This can help you to identify and troubleshoot problems early on.

✓ Sharding is a type of database partitioning that separates large databases into smaller, faster, more easily managed parts.