MongoDB Introduction

- NOSQL DATABASE
- KEY FEATURES
- HISTORY
- MONGODB APPLICATIONS
- ENVIRONMENT SETUP

What is NoSQL Database?

NotOnlySQL

- NoSQL or “non-SQL” a non-structured database.
- It provides a facility for storage and retrieval of data using fields.
- While in SQL the data stores in a tabular form.
- Companies are using a NoSQL database in big data and real-time applications.
- NoSQL offers “eventual consistency” so that it may not meet the real-time application requirements.
- Still, its use to merits over relational databases.

NoSQL Databases

Databases can be divided in 3 types:

- RDBMS (Relational Database Management System)
- OLAP (Online Analytical Processing)
- NoSQL (recently developed database)

NoSQL Database

NoSQL Database is used to refer a non-SQL or non relational database.

It 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.

History behind the creation of NoSQL Databases

In the early 1970, Flat File Systems are used.

Data were stored in flat files and the biggest problems with flat files are each company implement their own flat files and there are no standards.

It is very difficult to store data in the files, retrieve data from files because there is no standard way to store data.

Then the relational database was created by E.F. Codd and these databases answered the question of having no standard way to store data.

But later relational database also get a problem that it could not handle big data.

Due to this problem there was a need of database which can handle every types of problems then NoSQL database was developed.

What is MongoDB?

- *MongoDB is an open source platform written in C++ and has* a very *easy setup environment*.
- It is a cross-platform, document-oriented and non-structured database.
- MongoDB provides high performance, high availability, and auto-scaling.
- It is a NoSQL database and has flexibility with querying and indexing.
- MongoDB has very rich query language resulting in high performance.

MongoDB Introduction

- **MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling.**
- **Mongo DB is a document-oriented database.**
- **It is an open source product, developed and supported by a company named 10gen.**
- **MongoDB is available under General Public license for free, and it is also available under Commercial license from the manufacturer.**
- **The manufacturing company 10gen has defined MongoDB as:**
- ***"MongoDB is a scalable, open source, high performance, document-oriented database."* - 10gen**

Purpose of building MongoDB

- It may be a very genuine question that - "what was the need of MongoDB although there were many databases in action?“ There is a simple answer:
- All the modern applications require big data, fast features development, flexible deployment, and the older database systems not competent enough, so the MongoDB was needed.
- The primary purpose of building MongoDB is:
  - Scalability
  - Performance
  - High Availability
  - Scaling from single server deployments to large, complex multi-site architectures.

Big Data

- What is Data?
- The quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.
- Big Data is also data but with a huge size.
- Big Data is a term used to describe a collection of data that is huge in size and yet growing exponentially with time.
- In short such data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently.

Examples Of Big Data

- The New York Stock Exchange generates about *one terabyte* of new trade data per day.
- Social Media -The statistic shows that *500+terabytes* of new data get ingested into the databases of social media site Facebook, every day.
- This data is mainly generated in terms of photo and video uploads, message exchanges, putting comments etc.
- A single Jet engine can generate *10+terabytes* of data in *30 minutes* of flight time.
- With many thousand flights per day, generation of data reaches up to many *Petabytes.*

facebook

Types Of Big Data

- BigData' could be found in three forms:
- Structured - Any data that can be stored, accessed and processed in the form of fixed format is termed as a 'structured' data. An 'Employee' table in a database is an example of Structured Data.
- Unstructured - The output returned by 'Google Search'
- Semi-structured -Semi-structured data can contain both the forms of data. Example of semi-structured data is a data represented in an XML file.

Key points of MongoDB

- Develop Faster
- Deploy Easier
- Scale Bigger

History of MongoDB

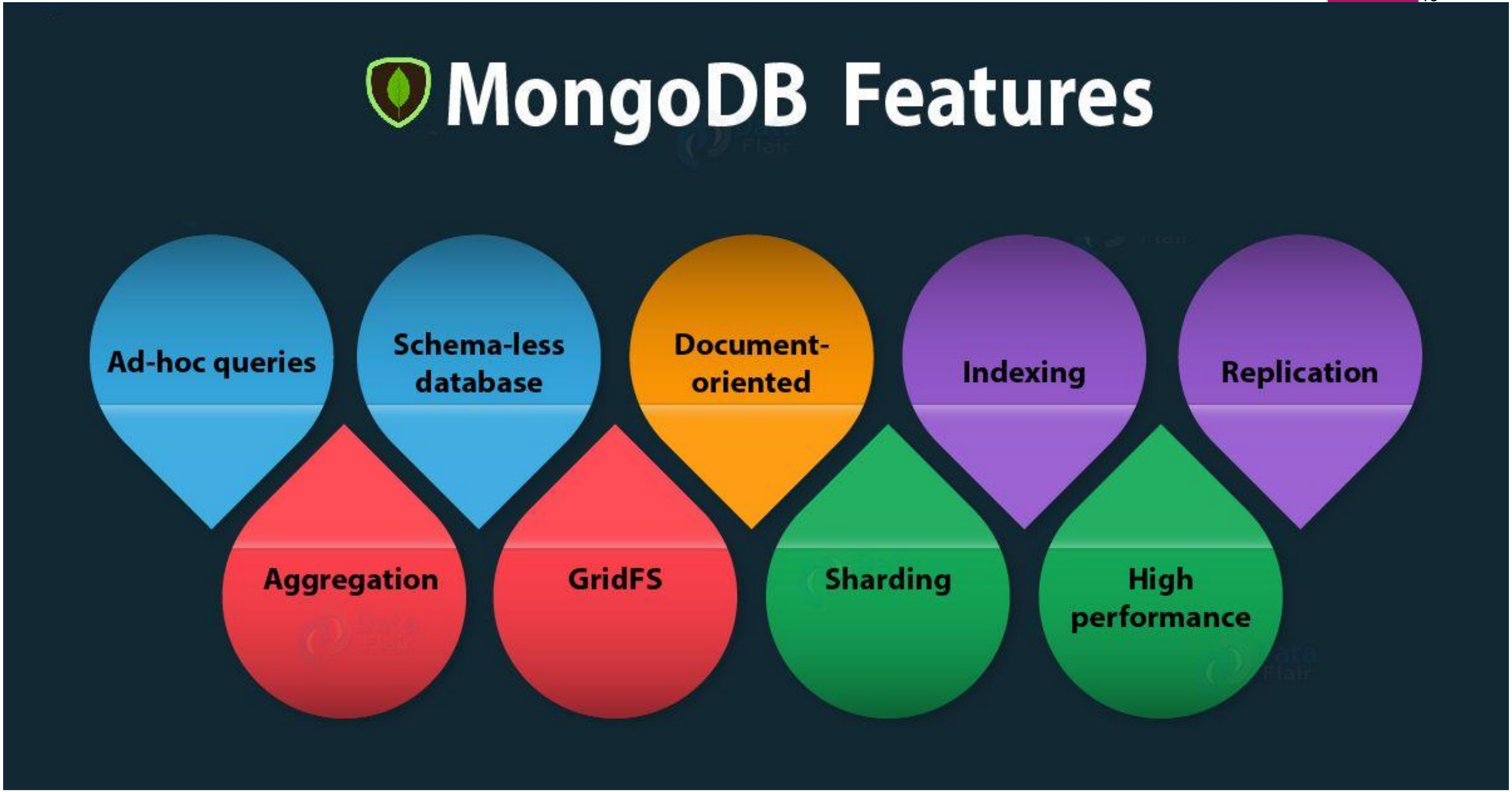
- The initial development of MongoDB began in 2007 when the company was building a platform as a service similar to window azure.
- "Window azure is a cloud computing platform and infrastructure, created by Microsoft, to build, deploy and manage applications and service through a global network."
- MongoDB was developed by a NewYork based organization named 10gen which is now known as MongoDB Inc.
- It was initially developed as a PAAS (Platform As A Service).
- Later in 2009, it is introduced in the market as an open source database server that was maintained and supported by MongoDB Inc.
- The first ready production of MongoDB has been considered from version 1.4 which was released in March 2010.
- MongoDB2.4.9 was the latest and stable version which was released on January 10, 2014.

What is document oriented database?

- MongoDB is a document oriented database.
- It is a key feature of MongoDB.
- It offers a document oriented storage. It is very simple you can program it easily.
- MongoDB stores data as documents, so it is known as document oriented database.
- There is also a broad category of database known as No SQL Databases.

FirstName = “Robin",
Address = “Wood Street",
Spouse = [{Name: “Anne"}].
FirstName =“Mike",
Address = “Park Lane"

- There are two different documents (separated by ".").
- Storing data in this manner is called as document oriented database.
- Mongo DB falls into a class of databases that calls Document Oriented Databases.



MongoDB Features

- Ad-hoc queries
- Schema-less database
- Document-oriented
- Indexing
- Replication
- Aggregation
- GridFS
- Sharding
- High performance

MongoDB Features

Being a NoSQL database, MongoDB has so many great features.

These amazing features make this technology very unique and attractive.

Also, these features are making MongoDB widely usable and popular.

Ad-hoc Queries

Generally, when we design a schema of a database, we don't know in advance about the queries we will perform.

Ad-hoc queries are the queries not known while structuring the database.

So, MongoDB provides ad-hoc query support which makes it so special in this case.

Ad-hoc queries are updated in real time, leading to an improvement in performance.

In MongoDB, you can search by field, range query and it also supports regular expression searches.

Schema-Less Database

In MongoDB, one collection holds different documents.

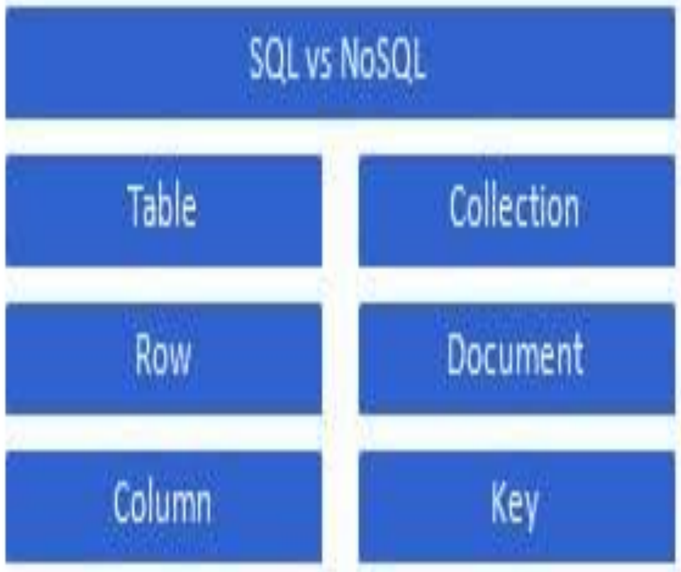
It has no schema so can have many fields, content, and size different than another document in the same collection.

This is why MongoDB shows flexibility in dealing with the databases.

It is a schema-less database written in C++.

Document-Oriented

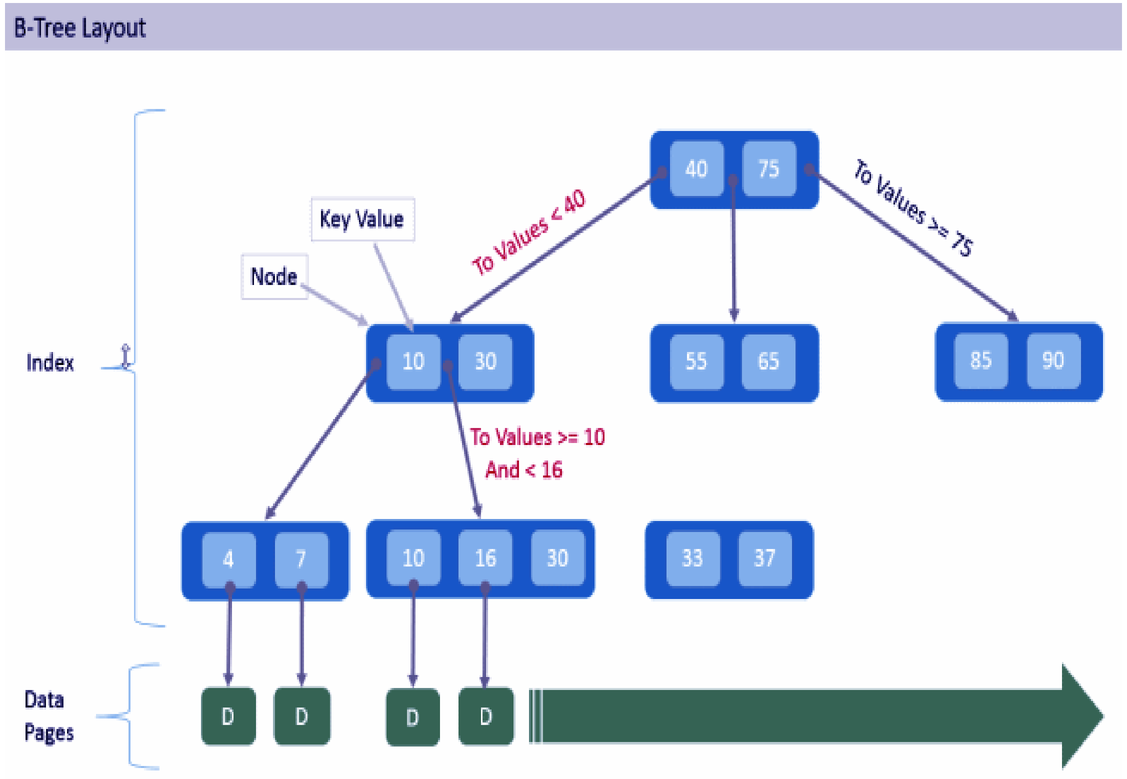
- MongoDB is a document-oriented database, which is a great feature itself.
- In the relational databases, there are tables and rows for arrangements of the data.
- Every row has specific no. of columns & those can store a specific type of data.
- Here comes the flexibility of NoSQL where there are fields instead of tables and rows.
- There are different documents which can store different types of data.
- There are collections of similar documents.
- Each document has a unique key id or object id which can both be user or system defined.



SQL vs NoSQL	
Table	Collection
Row	Document
Column	Key

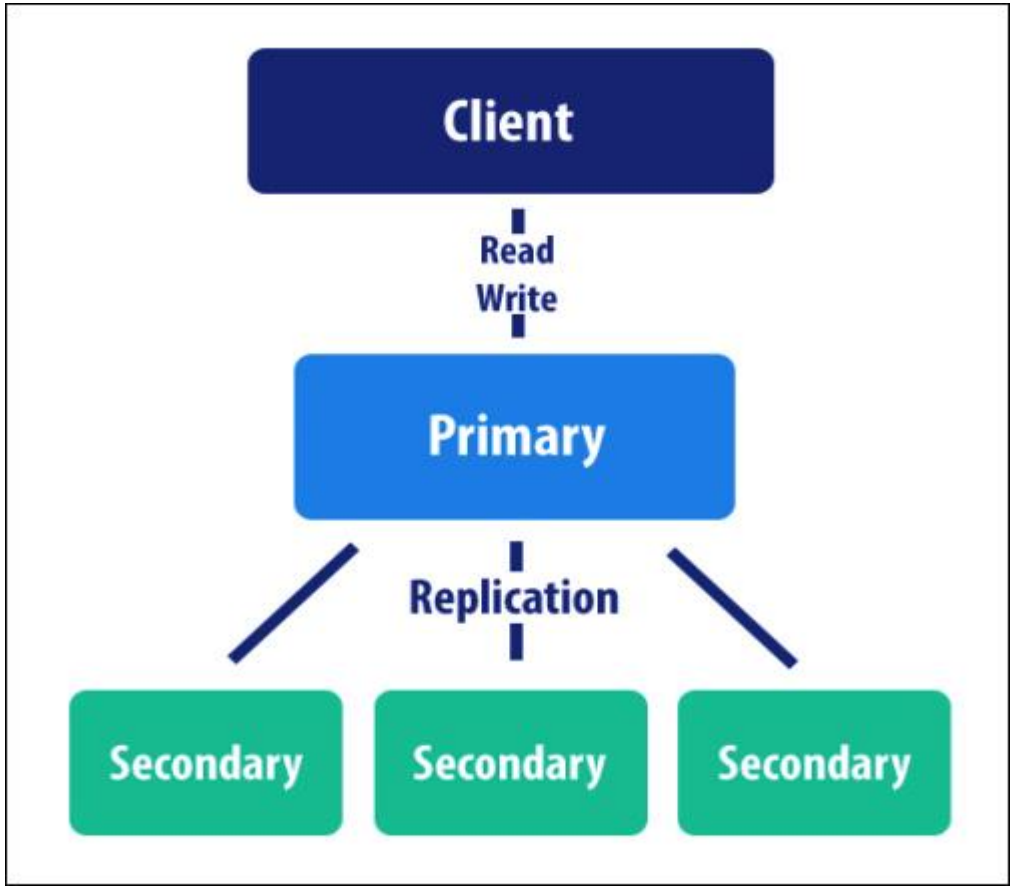
Indexing

- Indexing is very important for improving the performances of search queries.
- When we continuously perform searches in a document, we should index those fields that match our search criteria.
- In MongoDB, we can index any field indexed with primary and secondary indices.
- Making query searches faster, MongoDB indexing enhances the performance.



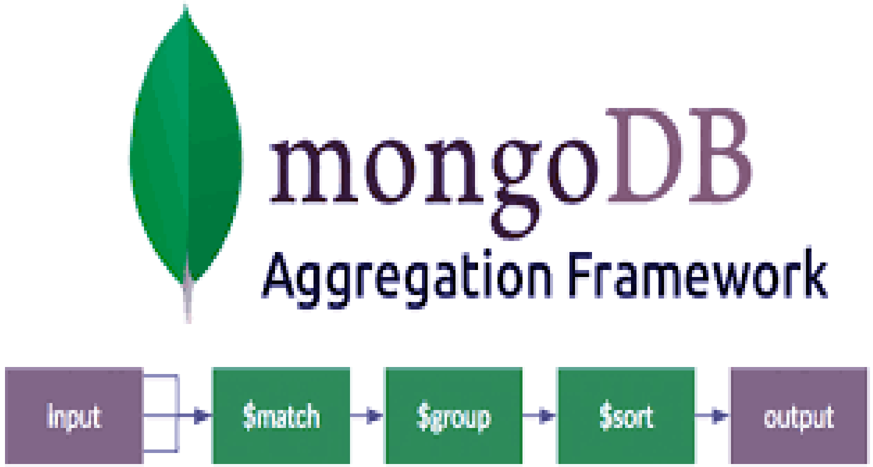
Replication

- When it comes to redundancy, replication is the tool that MongoDB uses.
- This feature distributes data to multiple machines.
- It can have primary nodes and their one or more replica sets.
- Basically, replication makes ready for contingencies.
- When the primary node is down for some reasons, the secondary node becomes primary for the instance.
- This saves our time for maintenance and makes operations smooth.



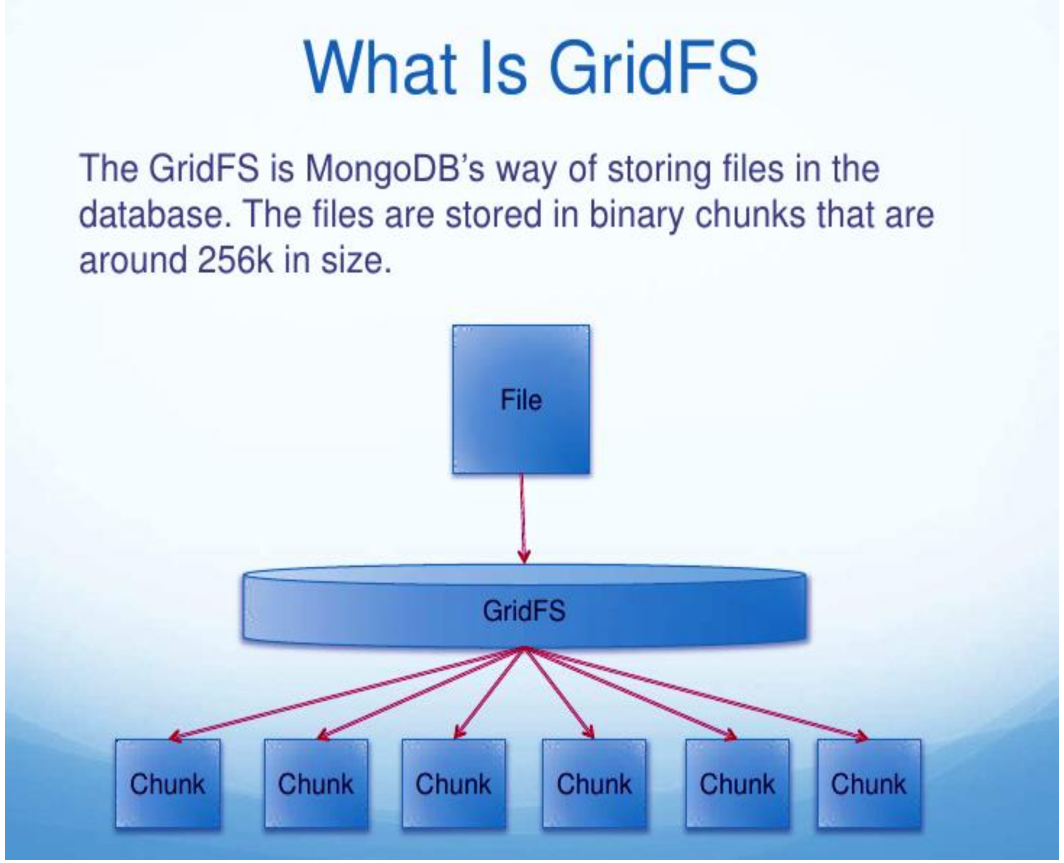
Aggregation – MongoDB Features

- MongoDB has an aggregation framework for efficient usability.
- We can batch process data and get a single result even after performing different operations on the group data.
- The aggregation pipeline, map-reduce function, and single purpose aggregation methods are the three ways to provide an aggregation framework.



GridFS

- GridFS is a feature of storing and retrieving files.
- For files larger than 16 MB this feature is very useful.
- GridFS divides a document in parts called chunks and stores them in a separate document.
- These chunks have a default size of 255kB except the last chunk.
- When we query GridFS for a file, it assembles all the chunks as needed.



What Is GridFS

The GridFS is MongoDB's way of storing files in the database. The files are stored in binary chunks that are around 256k in size.

High Performance

MongoDB is an open source database with high performance.

This shows high availability and scalability.

It has faster query response because of indexing and replication.

This makes it a better choice for big data and real-time applications.

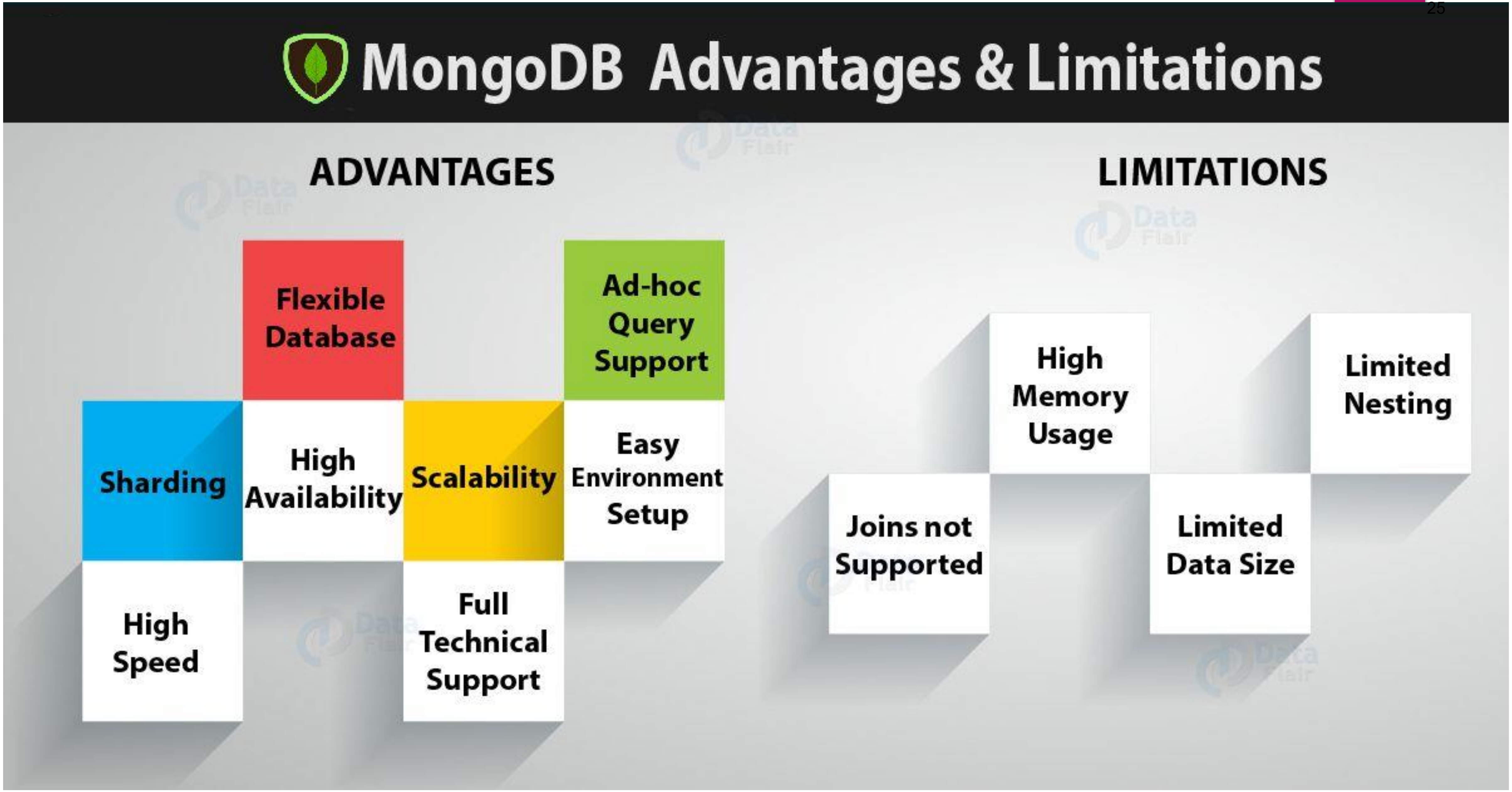
Uses JavaScript instead of Procedures.

MongoDB can run over multiple servers.

The data is duplicated to keep the system up and also keep its running condition in case of hardware failure.

Stores files of any size easily without complicating your stack.

Easy to administer in the case of failures.



MongoDB Advantages & Limitations

ADVANTAGES

- Flexible Database
- Ad-hoc Query Support
- Sharding
- High Availability
- Scalability
- Easy Environment Setup
- High Speed
- Full Technical Support

LIMITATIONS

- High Memory Usage
- Limited Nesting
- Joins not Supported
- Limited Data Size

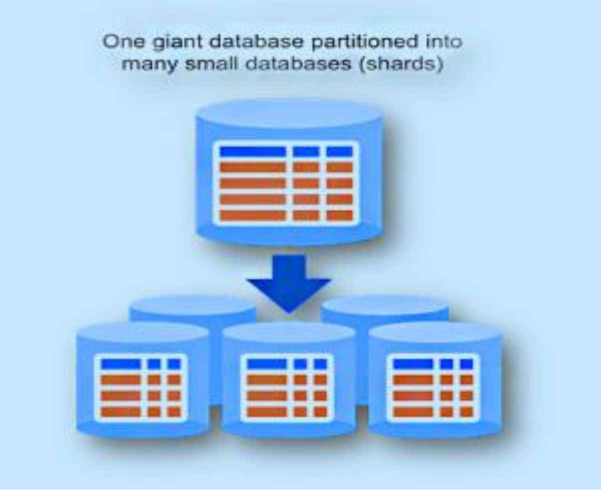
MonogoDB -Pros

Flexible Database

- MongoDB is schema less.
- It is a document database in which one collection holds different documents.
- This thing gives us flexibility and a freedom to store data of different types.

Sharding

- We can store a large data by distributing it to several servers connected to the application.
- If a server cannot handle such a big data then there will be no failure condition.
- The term we can use here is “auto-sharding”.


One giant database partitioned into
many small databases (shards)

MonogoDB -Pros

High Speed

- MongoDB is a document-oriented database.
- It is easy to access documents by indexing.
- Hence, it provides fast query response.
- The speed of MongoDB is 100 times faster than the relational database.

High Availability

- MongoDB has features like replication and gridFS.
- These features help to increase data availability in MongoDB.
- Hence the performance is very high.

MonogoDB -Pros

Scalability

- A great advantage of MongoDB is that it is a horizontally scalable database. When you have to handle a large data, you can distribute it to several machines.

Ad-hoc Query Support

- MongoDB has a very advanced feature for ad hoc queries. This is why we don't need to worry about fore coming queries coming in the future.

Easy Environment Setup

- It is easier to setup MongoDB then RDBMS. It also provides JavaScript client for queries.

Full Technical Support

- MongoDB Inc. provides professional support to its clients. If there is any problem, you can directly reacha MongoDB client support system.

Disadvantages Of MongoDB

Joins & Transactions not Supported

- MongoDB doesn’t support joins like a relational database. Yet one can use joins functionality by adding by coding it manually. But it may slow execution and affect performance.

High Memory Usage

- MongoDB stores key names for each value pairs. Also, due to no functionality of joins, there is data redundancy. This results in increasing unnecessary usage of memory.

Limited Data Size

- You can have document size, not more than 16MB.

Limited Nesting

- You cannot perform nesting of documents for more than 100 levels.

Prerequisites for Learning MongoDB

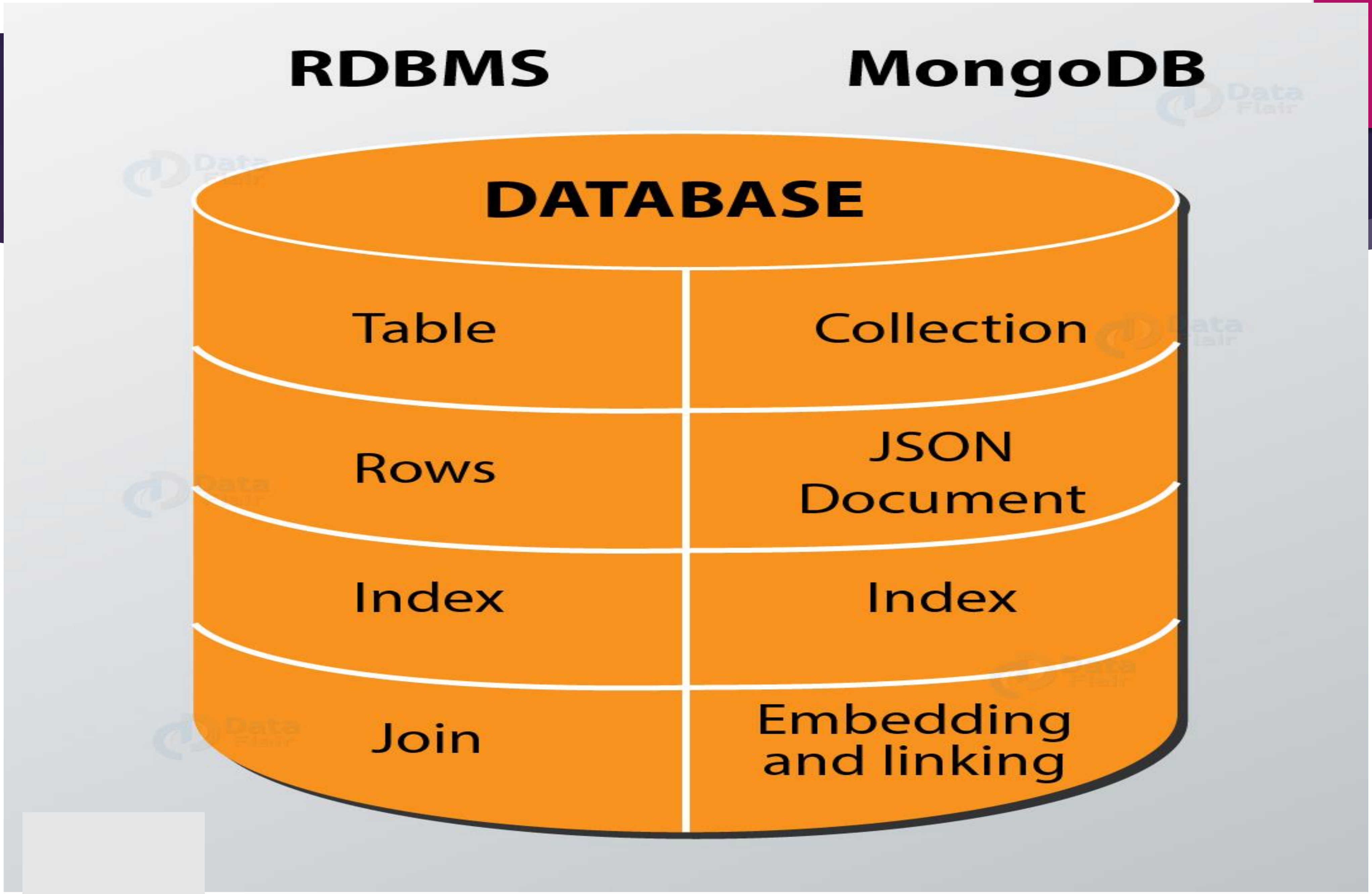
Knowledge of any programming language.

Knowledge of JavaScript

Knowing JSON will be helpful

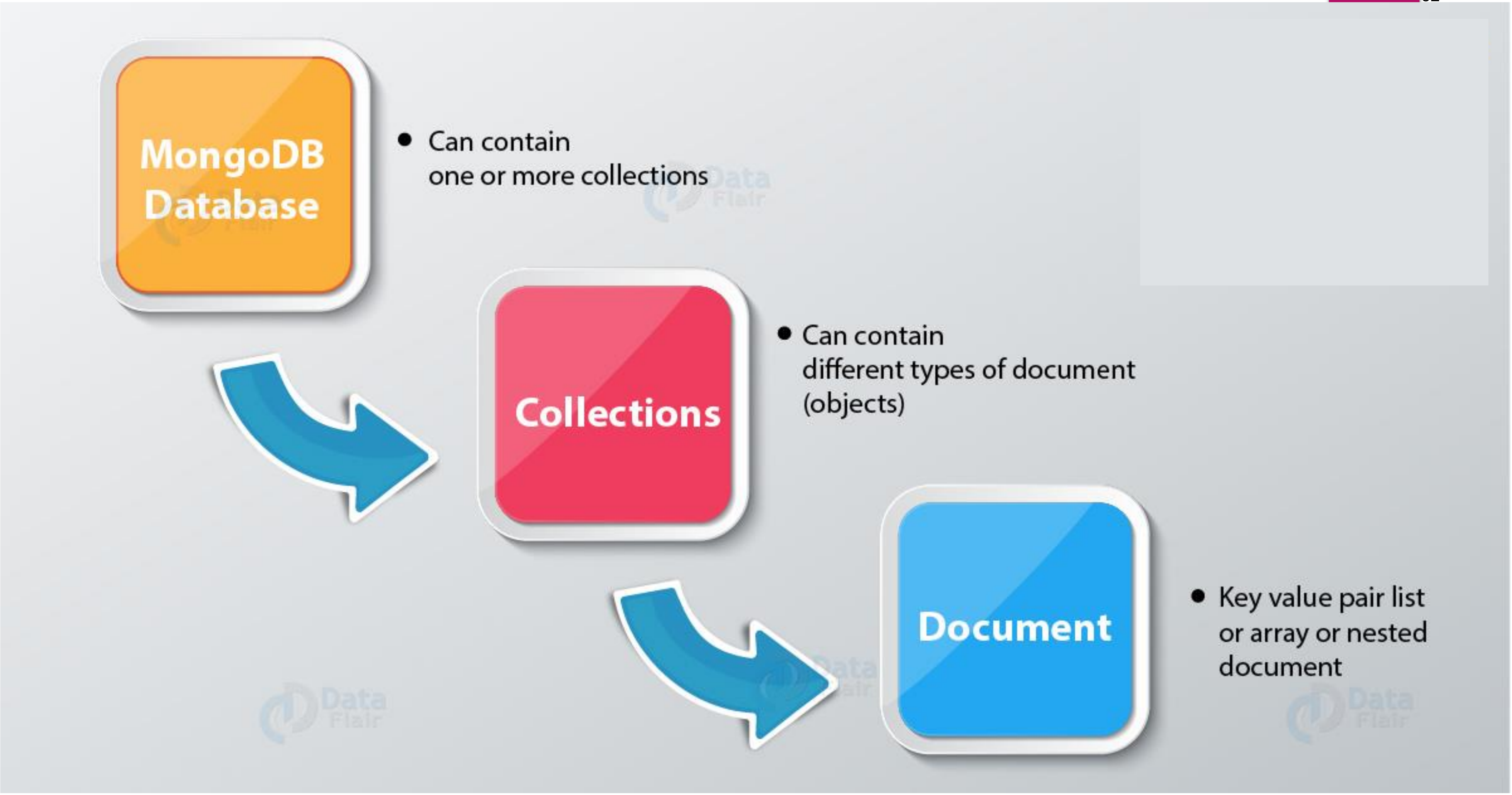
Some knowledge of RDBMS

Knowledge of any text editor



DATABASE

RDBMS	MongoDB
Table	Collection
Rows	JSON Document
Index	Index
Join	Embedding and linking



- Can contain one or more collections

- Can contain different types of document (objects)

- Key value pair list or array or nested document

MongoDB Vs RDBMS

- RDBMS is having a relational database but MongoDB has a non-relational database.
- In RDBMS we need to design the table then only we can start coding but in MongoDB, we can directly start coding.
- RDBMS supports SQL language and MongoDB supports SQL as well as JSON query language.
- RDBMS is table based whereas MongoDB is key-value based.
- MongoDB is document based whereas RDBMS is row based.
- RDBMS is column based whereas MongoDB is field based.
- RDBMS is not that easy to set up but MongoDB is comparatively easy to set up.
- MongoDB is horizontally scalable, on the other hand, RDBMS is vertically scalable.
- RDBMS processes the data very slow as compared to the unstructured data of MongoDB.
- RDBMS accentuates on ACID (Atomicity, Consistency, Isolation, Durability) properties. On the other hand, MongoDB accentuates on CAP (Consistency, Availability, Partition tolerance) theorem.

Data Modeling in MongoDB

- In MongoDB, data has a flexible schema.
- It is totally different from SQL database where you had to determine and declare a table's schema before inserting data.
- MongoDB collections do not enforce document structure.
- MongoDB deals in collections, documents, and fields.
- We can have documents containing different sets of fields or structures in the same collection.
- Also, common fields in a collection can contain different types of data. This helps in easy mapping.
- The main challenge in data modeling is balancing the need of the application, the performance characteristics of the database engine, and the data retrieval patterns.

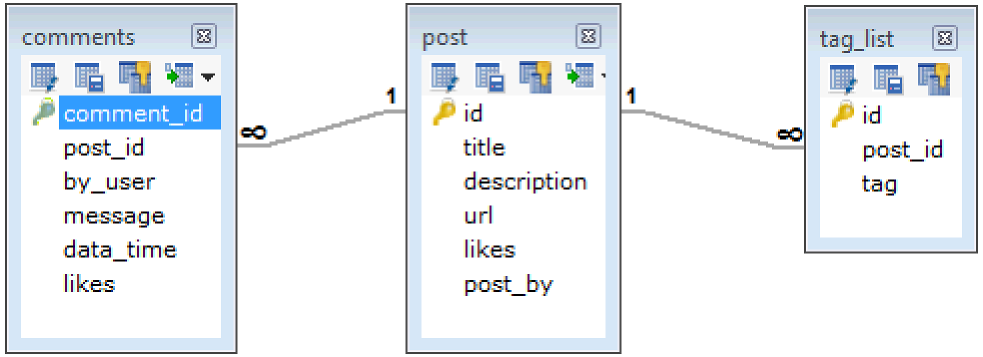
Designing the schema in MongoDB

- Always design schema according to user requirements.
- Do join on write operations not on read operations.
- Objects which you want to use together, should be combined into one document. Otherwise they should be separated (make sure that there should not be need of joins).
- Optimize your schema for more frequent use cases.
- Do complex aggregation in the schema.
- You should duplicate the data but in a limit, because disc space is cheaper than compute time.

Example - MongoDB

A client who needs a database design for his website. His website has the following requirements:

- Every post is distinct (contains unique title, description and url).
- Every post can have one or more tags.
- Every post has the name of its publisher and total number of likes.
- Each post can have zero or more comments and the comments must contain user name, message, data-time and likes.
- For the above requirement, a minimum of three tables are required in RDBMS.



comments
- comment_id
- post_id
- by_user
- message
- data_time
- likes

post
- id
- title
- description
- url
- likes
- post_by

tag_list
- id
- post_id
- tag

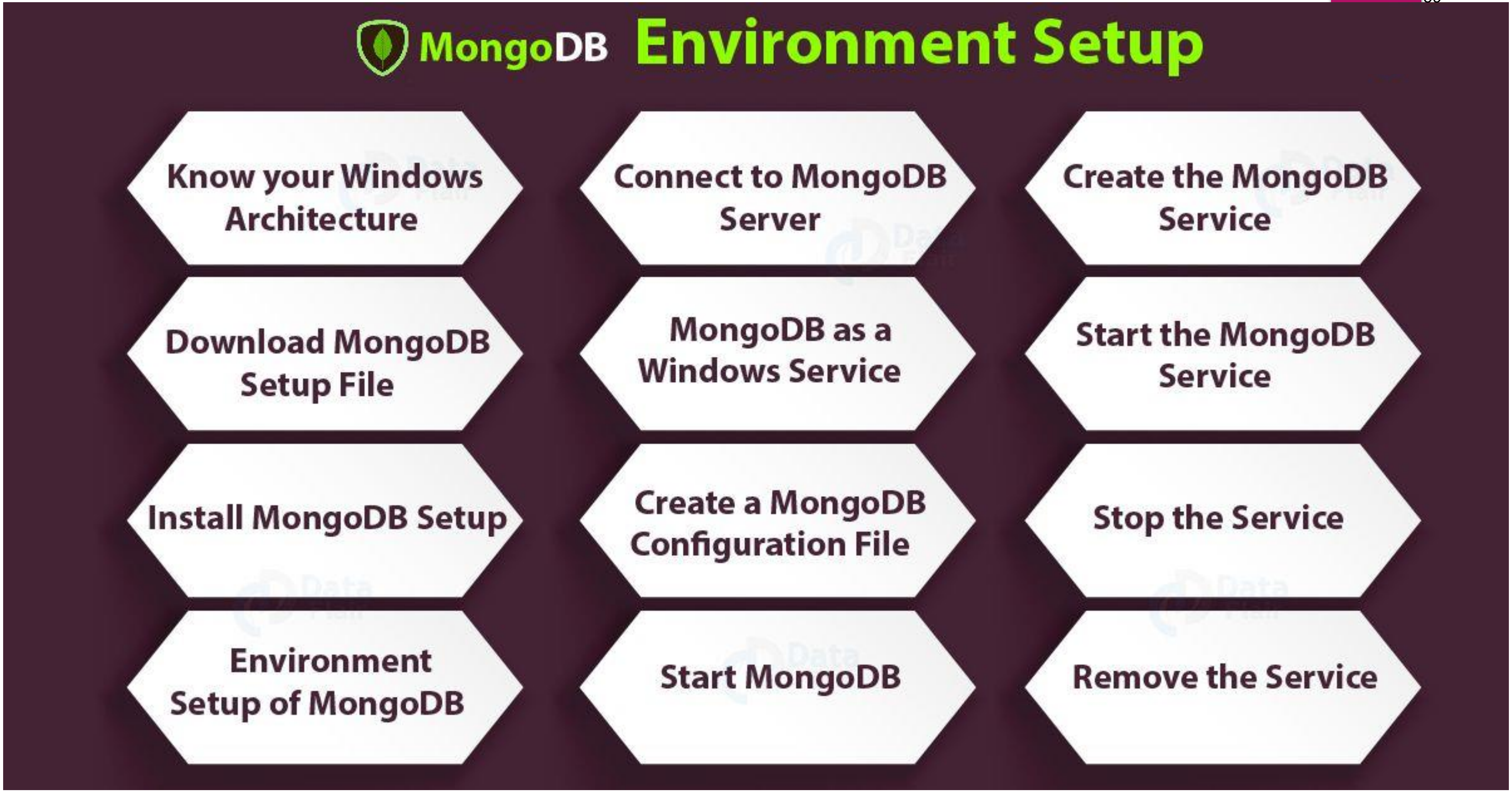
Example - MongoDB

- **But in MongoDB, schema design will have one collection post and has the following structure:**

```
{
_id: POST_ID
title: TITLE_OF_POST,
description: POST_DESCRIPTION,
by: POST_BY,
url: URL_OF_POST,
tags: [TAG1, TAG2, TAG3],
likes: TOTAL_LIKES,
comments: [
{
user: 'COMMENT_BY',
message: TEXT,
datecreated: DATE_TIME,
like: LIKES
},
{
user: 'COMMENT_BY',
message: TEST,
dateCreated: DATE_TIME,
like: LIKES
}]}
```

MongoDB Datatypes

Data Types	Description
String	String is the most commonly used datatype. It is used to store data. A string must be UTF 8 valid in mongodb.
Integer	Integer is used to store the numeric value. It can be 32 bit or 64 bit depending on the server you are using.
Boolean	This datatype is used to store boolean values. It just shows YES/NO values.
Double	Double datatype stores floating point values.
Min/Max Keys	This datatype compare a value against the lowest and highest bson elements.
Arrays	This datatype is used to store a list or multiple values into a single key.
Object	Object datatype is used for embedded documents.
Null	It is used to store null values.
Symbol	It is generally used for languages that use a specific type.
Date	This datatype stores the current date or time in unix time format. It makes you possible to specify your own date time by creating object of date and pass the value of date, month, year into it.



MongoDB Environment Setup

- Know your Windows Architecture
- Download MongoDB Setup File
- Install MongoDB Setup
- Environment Setup of MongoDB
- Connect to MongoDB Server
- MongoDB as a Windows Service
- Create a MongoDB Configuration File
- Start MongoDB
- Create the MongoDB Service
- Start the MongoDB Service
- Stop the Service
- Remove the Service

MongoDB Shell

MongoDB have a JavaScript shell that allows interaction with MongoDB instance from the command line.

If you want to create a table, you should name the table and define its column and each column's data type.

The shell is useful for performing administrative functions and running instances.

Start MongoDB

To start MongoDB you need to execute this command.

This will start the main MongoDB process.

You will see at the bottom of the command prompt a message as “waiting for a connection”. This means that the process has started successfully.

```
“C:\Program Files\MongoDB\Server\3.6\bin\mongod.exe”
```

Connect to MongoDB Server

After executing the MongoDB.exe file, the process has begun.

Now we need to connect it to the MongoDB server.

We will connect it through Mongo.exe shell by opening another command prompt.

Now you need to execute a command to connect with the shell.

“C:\Program Files\MongoDB\Server\3.6\bin\mongo.exe”

Now MongoDB is ready to use. You can terminate the running Process of MongoDB by pressing “ctrl+c”.

MongoDB

Create database

a. The “use” Command

b. Check selected database

c. List database

d. Save a document

MongoDB Create Database

- If you are looking for a command to create a database, you must stop now.
- Because MongoDB does not provide any command to create a database.
- In fact, we don’t create a database in MongoDB.
- Unlike SQL, where we need to create a database, tables and then insert the values manually, MongoDB creates the database automatically.
- You just need to save a value in the defined collection with the preferred name.
- You don’t even have to mention that you want to create a database.
- However, you can create collections manually.

The “use” Command

- **If there is no database, use the following command to define the name for your database and the database is created if it doesn’t exist already.**

Syntax

```
use database_name
```

For example, Let’s say the database name is “dataflair”.

```
>use dataflair
Switched to db dataflair
```

Check Selected Database

- You can check that database you have selected already.
- Follow this command to do that.

```
>db
dataflair
```

List Database

If you don’t know the databases that already exist in the system, you can list them and know if they do. Just use the following command for this.

```
>show dbs
local 0.52938GB
test 0.49231GB
```

Save a Document

- **Run the following command to insert a document into the database.**

```
> db.user.insert({name: “John", age: 23})
WriteResult({ "nInserted" : 1 })
> show dbs
local        0.52938GB
dataflair      0.49231GB
test         0.49231GB
```

MongoDB Drop Database

- The dropDatabase command is used to drop a database.
- It also deletes the associated data files.
- It operates on the current database.

Syntax:

- db.dropDatabase()

- show dbs
- use dataflair
- db.dropDatabase()

MongoDB Create Collection

- In MongoDB, db.createCollection(name, options) is used to create collection.
- But usually you don?t need to create collection.
- MongoDB creates collection automatically when you insert some documents.
- It will be explained later. First see how to create collection:

Syntax:

- db.createCollection(name, options)
- Name: is a string type, specifies the name of the collection to be created.
- Options: is a document type, specifies the memory size and indexing of the collection. It is an optional parameter
- >db.createCollection(“Quinnox")

How does MongoDB create collection automatically

- MongoDB creates collections automatically when you insert some documents.
- For example: Insert a document named seomount into a collection named Quinnox.
- The operation will create the collection if the collection does not currently exist.
- >db.Quinnox.insert({"name" : "seomount"})
- >show collections

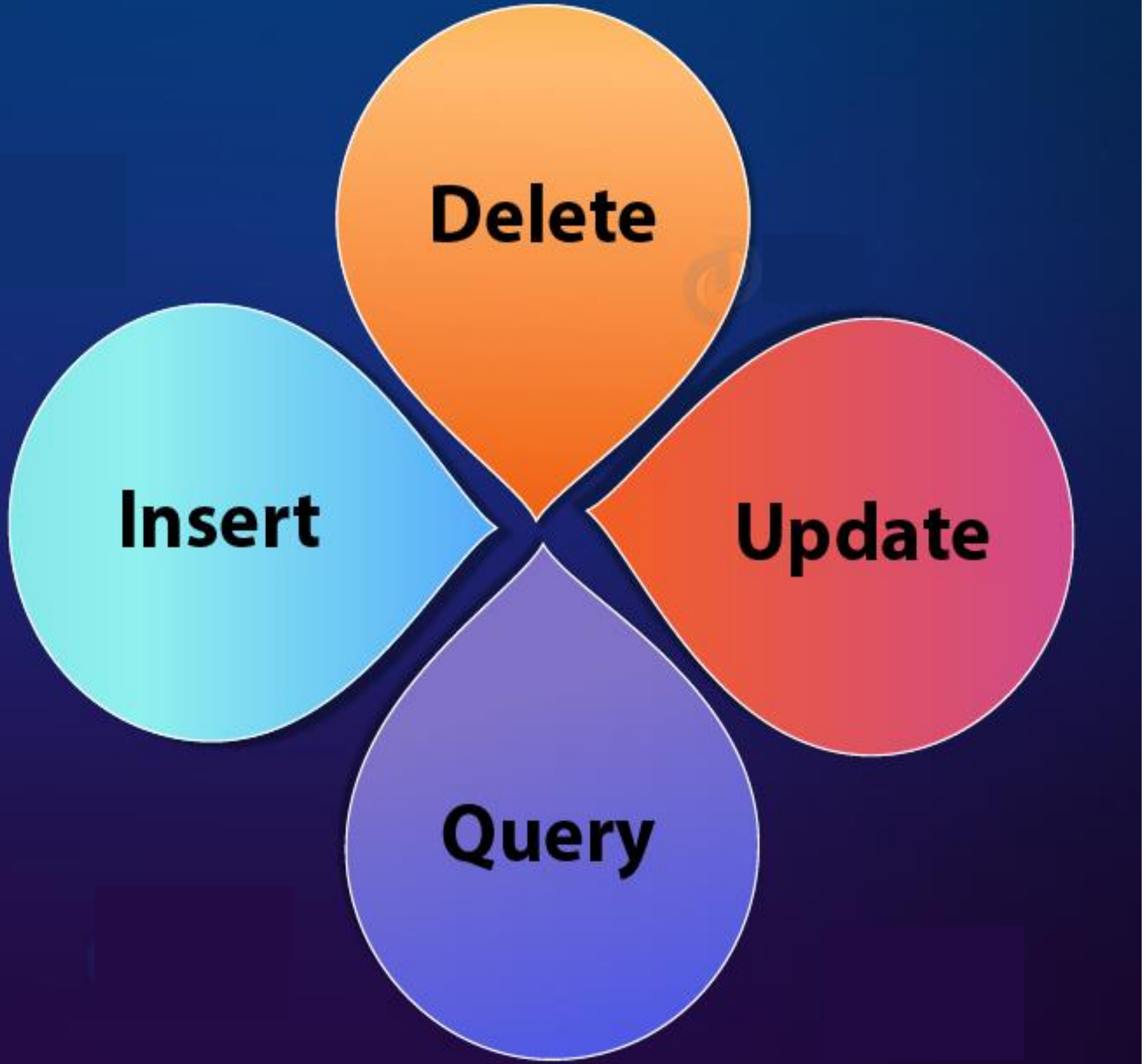
  Quinnox

- If you want to see the inserted document, use the find() command.

Syntax:

- db.collection_name.find()

Operations for MongoDB Document



Delete

Insert

Update

Query

MongoDB insert documents

- In MongoDB, the db.collection.insert() method is used to add or insert new documents into a collection in your database.

Upsert

- There are also two methods "db.collection.update()" method and "db.collection.save()" method used for the same purpose.
- These methods add new documents through an operation called upsert.
- Upsert is an operation that performs either an update of existing document or an insert of new document if the document to modify does not exist.
- >db.COLLECTION_NAME.insertOne(document)

Insert a MongoDB document

- Insert a MongoDB document in the collection named “examples”.
- Here if we don’t specify the _id field then MongoDB by itself adds the _id field with an ObjectId value to the document that we have created.

```
db.examples.insertOne(
{ item: "c", qty: 100, tags: ["cotton"], size: { h: 30, w: 37.5, uom:
"cm" } }
)
```

- Here insertOne() returns a document that includes newly inserted document’s _id value.

db.collection.insertMany()

- To insert many values at a time in to the MongoDB document.
- You can use the following command to insert many values at a time:

db.collection.insertMany()

```
 db.examples.insertMany([
{ item: "j", qty: 25, tags: ["blank", "white"], size: { h: 17, w: 28, uom: "cm" } },
{ item: "mats", qty: 85, tags: ["blue"], size: { h: 29.9, w: 39.5, uom: "cm" } },
{ item: "mouse", qty: 25, tags: ["gel", "grey"], size: { h: 27, w: 29.85, uom: "cm" } }
])
```

Query operations in MongoDB

- When we want to select all documents in a collection, we can pass an empty document to the find method.

  db.examples.find({})

- This operation is similar to SQL command as follows:

  SELECT * FROM example

MongoDB pretty()

- In the Mongo universe, the pretty() method specifies the cursor object to display the Mongo query results in an easy-to-read attractive format. The cursor.pretty() method has the following prototype form:

Syntax

```
> db.collection_name.find(<query_string>).pretty()
```

- The query_string is an optional input argument that retrieves the documents from a collection on the basis of a specified choice criteria
- The cursor.pretty() method beautify the JSON documents or the collections within in the Mongo shell.

```
db.products.find().pretty()
```

Specify Equality Condition - Query

- We can use <field>:<value> expressions to specify equality condition.

```
{ <field1>: <value1>, ... }
```

- Now we will understand this by our above example:

```
db.examples.find( { status: "D" } )
```

- This is similar to SQL command as follows:

```
SELECT * FROM example WHERE status = "D"
```

Specify conditions using query operators

- The syntax for the following operation is as follows:

```
{ <field1>: { <operator1>: <value1> }, ... }
```

- Now we will understand this syntax with our example as follows:

```
db.example.find( { status: { $in: [ "A", "D" ] } } )
```

- In the above example, we are retrieving all the documents from example collection where the status is equal to “A” or”D”.
- This is similar to SQL command as follows:

```
SELECT * FROM example WHERE status in ("A", "D")
```

Specify AND Condition

- Here we will be performing operations on more than one documents.
- In the AND operation all the conditions should be true otherwise the command will not be executed.

```
db.examples.find( { status: "A", qty: { $lt: 30 } } )
```

- In the above example, all the documents from example collection where the status equals “A” and qty is less than ($lt) 30.
- This is similar to SQL command as follows:

```
SELECT * FROM example WHERE status = "A" AND qty < 30
```

Specify OR Condition

- In the logical OR operation, at least one condition should be true so that we can execute the operation.
- In the below example we will retrieve all the documents in which status is equal to “A” or qty is less than 30.

```
db.examples.find( { $or: [ { status: "A" }, { qty: { $lt: 30 } } ] } )
```

- This is similar to SQL command as follows:

```
SELECT * FROM example WHERE status = "A" OR qty < 30
```

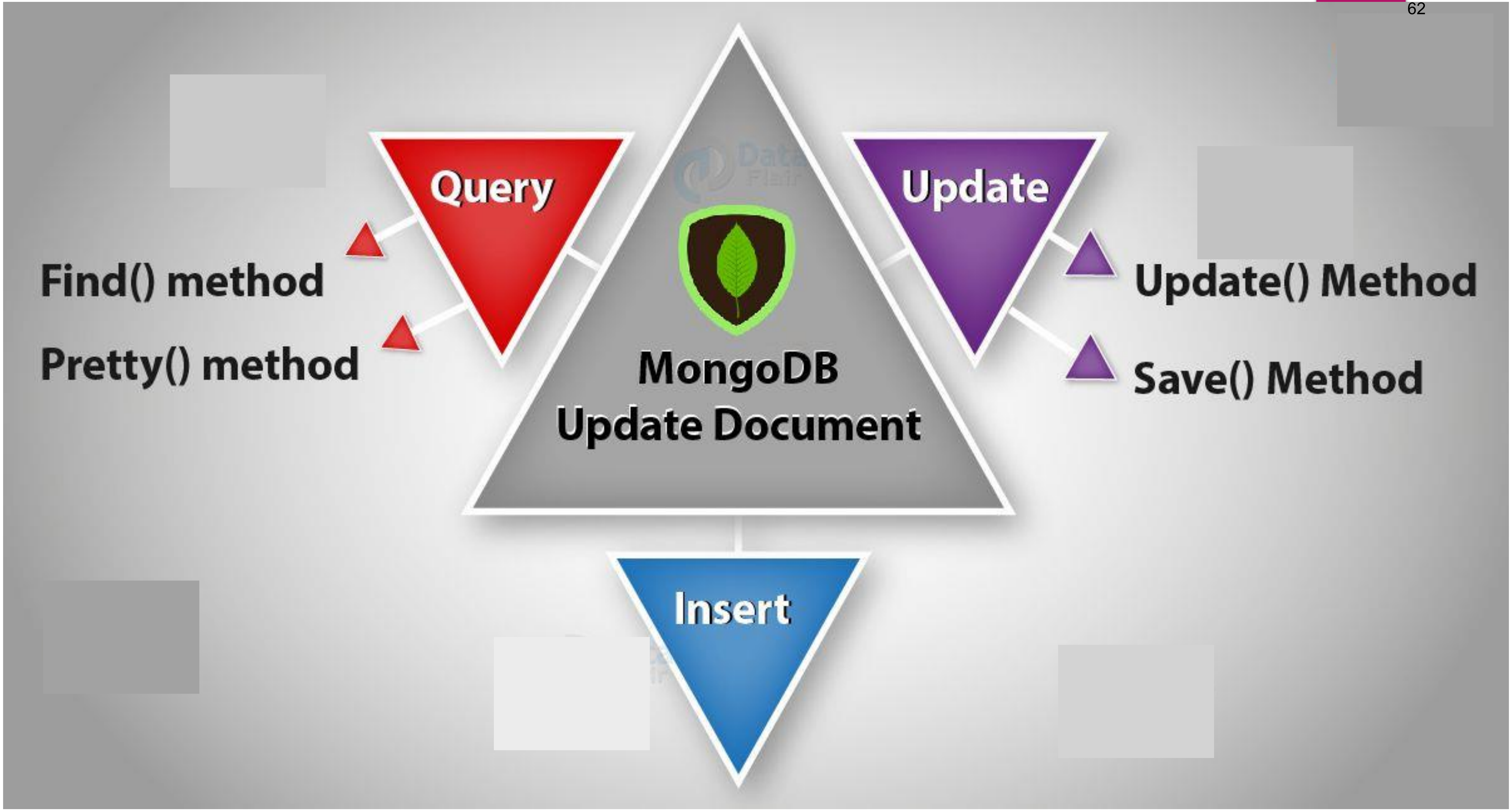
Specify AND as well as OR Condition

- In this compound query only those documents will be printed in output where status is equal to “A” and either qty is less than 30 or item name starts with “p”.

```
db.examples.find( {
status: "A",
$or: [ { qty: { $lt: 30 } }, { item: /^p/ } ]
} )
```

- This is similar to SQL command as follows:

```
SELECT * FROM example WHERE status = "A" AND ( qty < 30 OR item LIKE "p%")
```


Query
Find() method
Pretty() method
MongoDB
Update Document
Update
Update() Method
Save() Method
Insert

MongoDB CRUD

MongoDB CRUD is a short form of all this MongoDB operation:

C – Create/Insert

R – Read /Query

U – Update

D – Delete

MongoDB Update Document

There are two methods for MongoDB update document- existing.

- **Update() Method**
- **Save() Method**

We use update() method when we need to update the values of the existing document.

While we use the save() method when we need to replace the existing document with another document passed in it as a parameter.

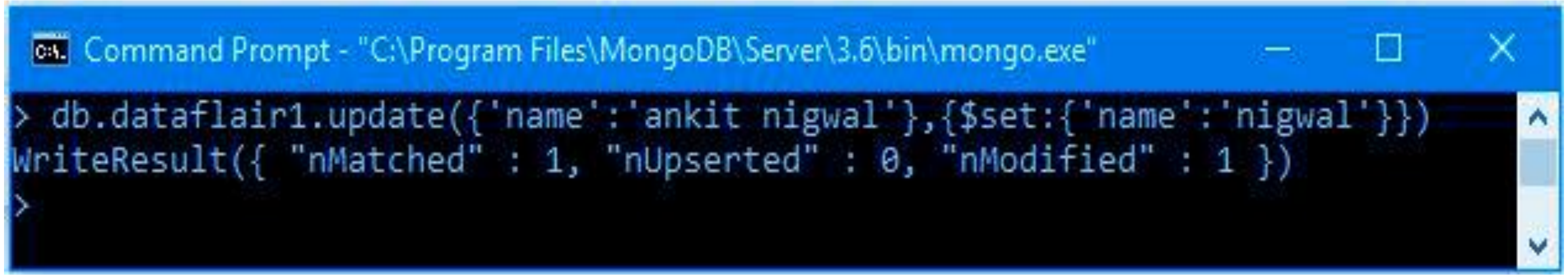
However, update() method is also used to replace the whole document depending upon the parameter passed.

The Update() Method

- The syntax to update a data with this method is below.

db.collection_name.update(criteria, update_data)

- The $set operator is used to update a single field.



Command Prompt - "C:\Program Files\MongoDB\Server\3.6\bin\mongo.exe"

```
> db.dataflair1.update({'name':'ankit nigwal'},{$set:{'name':'nigwal'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

- The $set operator adds a new field with the specified value if it doesn’t exist.
- For this, the field should not violate a type constraint.

Update a Single MongoDB Document

- updateOne() method is used for updating single document
- The syntax is as follows:

```
db.examples.updateOne(
{ item: "papers" },
{
$set: { "size.uom": "cm", status: "P" },
$currentDate: { lastModified: true }
})
```

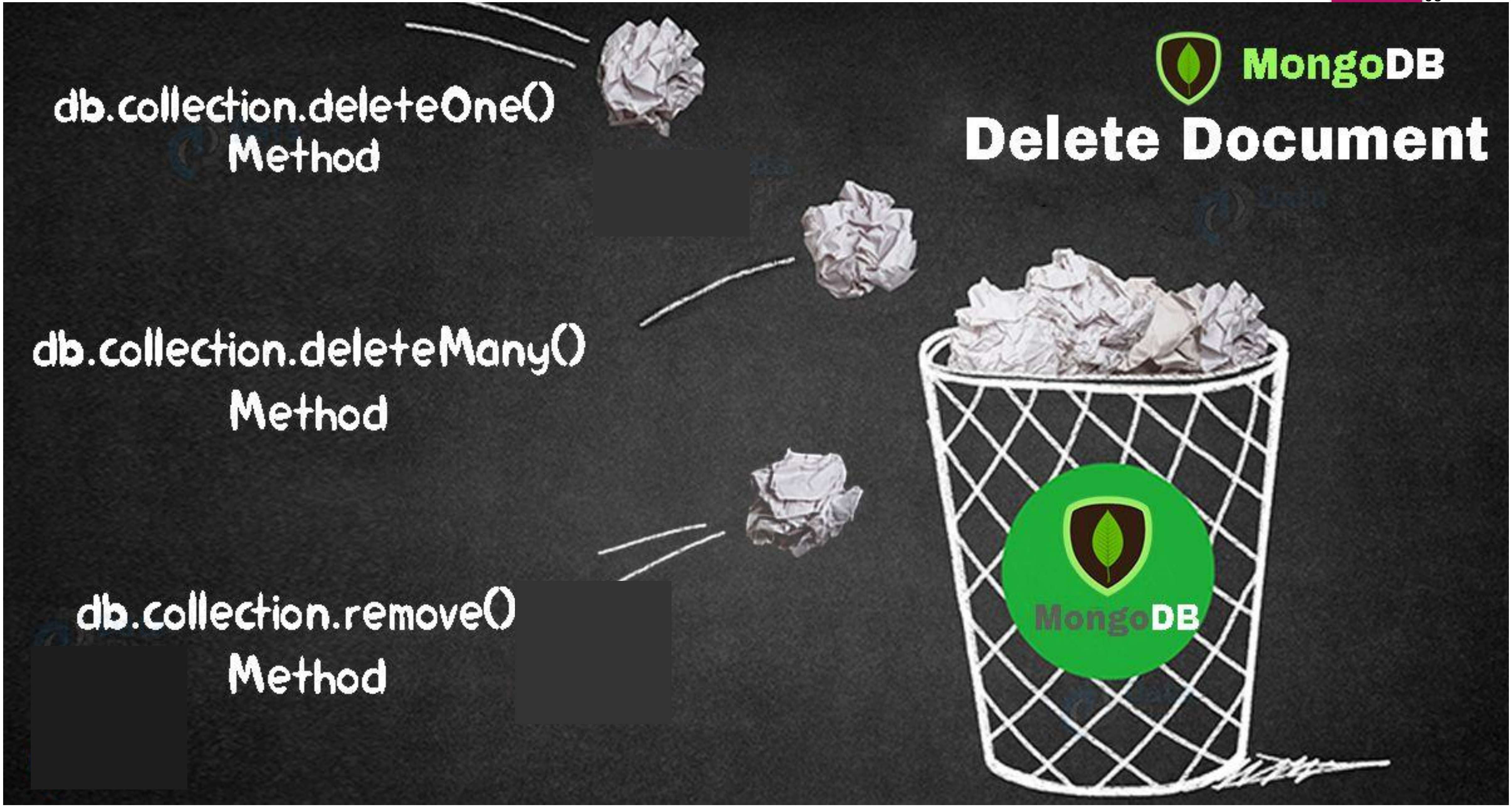
- Here the document will be updated where an item is equal to “papers”.

Update a Multiple MongoDB Documents

- updateMany() method is used for updating multiple documents.
- Here the update will reflect on more than one document.

```
db.examples.updateMany(
{ "qty": { $lt: 50 } },
{
$set: { "size.uom": "in", status: "P" },
$currentDate: { lastModified: true }
})
```

- Here, the documents of example collection will be updated where qty is less than 50.



[redacted]

[redacted]

[redacted]

MongoDB Delete Document

MongoDB provides the option to delete the documents from the collection.

You can delete one, many or all of the documents.

There are three methods in MongoDB to delete documents as discussed below.

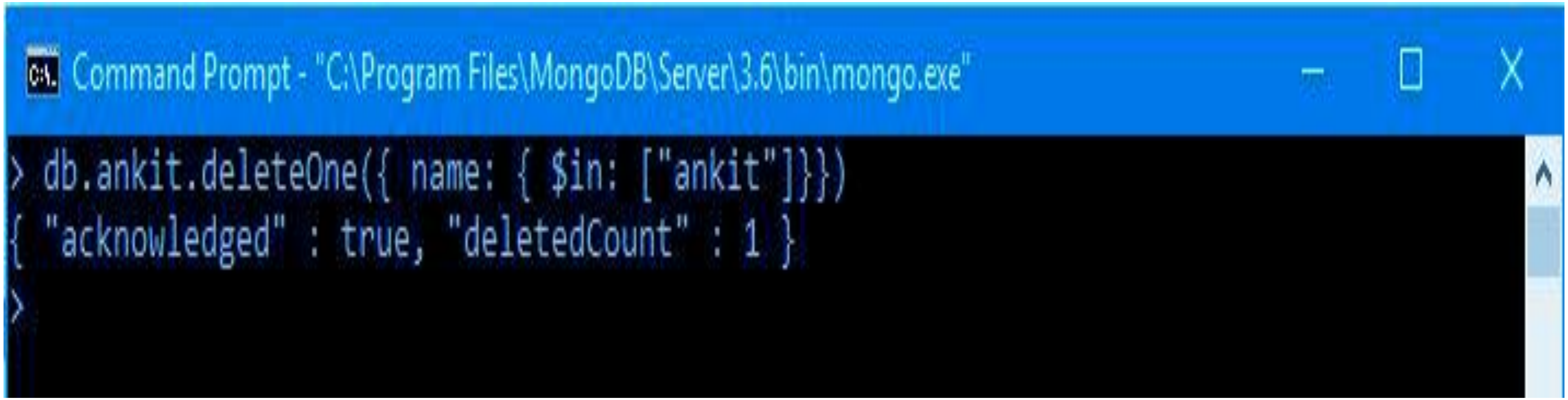
db.collection.deleteOne() Method

db.collection.deleteMany() Method

db.collection.remove() Method

The db.collection.deleteOne() Method

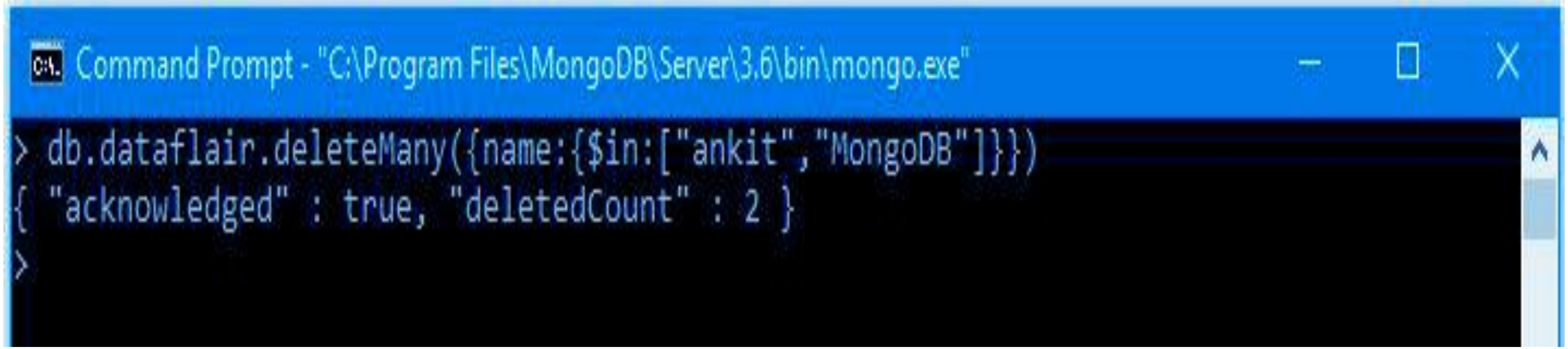
- **This method deletes only one document from the collection, even if multiple documents match the criteria.**


Command Prompt - "C:\Program Files\MongoDB\Server\3.6\bin\mongo.exe"

```
> db.ankit.deleteOne({ name: { $in: ["ankit"]}})
{ "acknowledged" : true, "deletedCount" : 1 }
>
```

The db.collection.deleteMany() Method

- **This method deletes all the documents that match the criteria.**

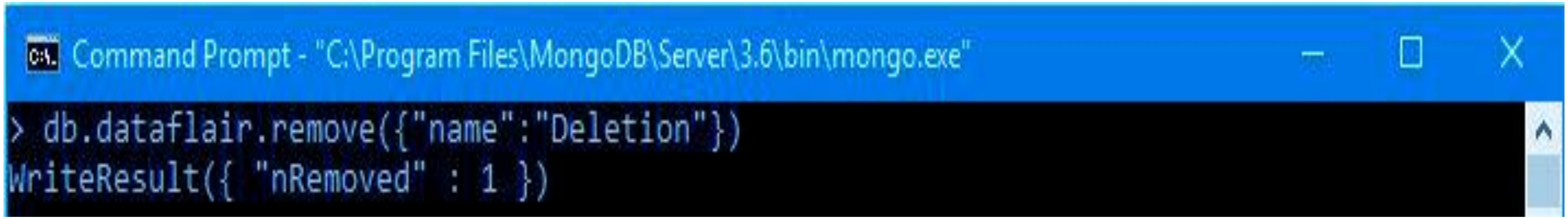


Command Prompt - "C:\Program Files\MongoDB\Server\3.6\bin\mongo.exe"

```
> db.dataflair.deleteMany({name:{$in:["ankit","MongoDB"]}})
{ "acknowledged" : true, "deletedCount" : 2 }
>
```

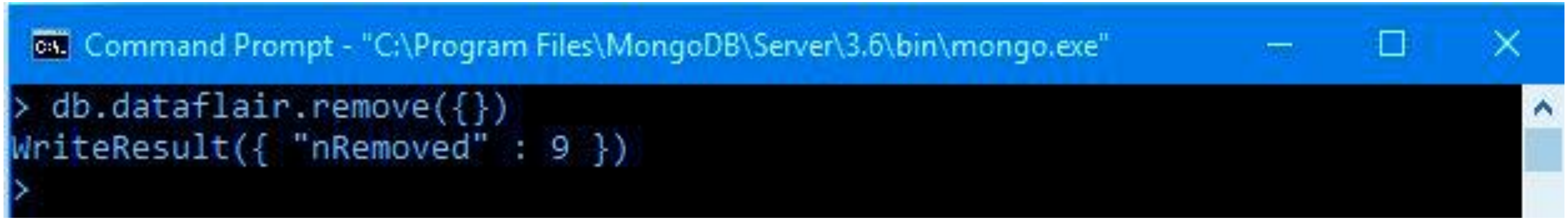
The db.collection.remove() Method

- You can delete one or all the documents that match the specified criteria by using this method.


Command Prompt - "C:\Program Files\MongoDB\Server\3.6\bin\mongo.exe"

```
> db.dataflair.remove({"name":"Deletion"})
WriteResult({ "nRemoved" : 1 })
```

- We can also delete all the documents just by leaving the filter in remove().


Command Prompt - "C:\Program Files\MongoDB\Server\3.6\bin\mongo.exe"

```
> db.dataflair.remove({})
WriteResult({ "nRemoved" : 9 })
>
```

MongoDB

Limit Record

- skip() method

MongoDB Limit Records

When we query a collection, it shows all the documents contained in it.

The limit cursor is used for retrieving only numbers of documents that we need.

We can use MongoDB limit() method to specify the number of documents we want it to return.

When we query a document using the db.collection_name.find() method, we can append this method to specify the limit.

limit() Method

- MongoDB limit() Method
- In MongoDB, limit() method is used to limit the fields of document that you want to show.
- Sometimes, you have a lot of fields in collection of your database and have to retrieve only 1 or 2.
- In such case, limit() method is used.
- The MongoDB limit() method is used with find() method.

Syntax:

```
db.COLLECTION_NAME.find().limit(NUMBER)

db.dataflair.find().limit(1)
```

MongoDB Skip() Method

- When we use the MongoDB limit() method, it shows from the beginning of the collection to the specified limit.
- If we want it to start not from the beginning and skipping some documents, we can use the Skip() method for this task.
- To do this, we need to add the skip() in the cursor and specify the number of documents that we want to skip.
- It is used with find() and limit() methods.

Syntax:

```
db.COLLECTION_NAME.find().limit(NUMBER).skip(NUMBER)

db.dataflair.find().skip(3)

db.dataflair.find().limit(1).skip(2)
```

MongoDB sort() method

- In MongoDB, sort() method is used to sort the documents in the collection.
- This method accepts a document containing list of fields along with their sorting order.
- The sorting order is specified as 1 or -1.
  - 1 is used for ascending order sorting.
  - -1 is used for descending order sorting.

Syntax:

```
db.COLLECTION_NAME.find().sort({KEY:1})
```

```
db.dataflair.find().sort({“qty":-1})
```

Cassandra vs MongoDB

- **Cassandra and MongoDB both are types of NoSQL databases.**
- **Cassandra is a distributed database system designed to handle large amount of data and known for its high scalability and high performance.**
- **While, MongoDB is document oriented database which also provides high scalability, high performance and automatic scaling.**
- **In terms of simplicity, databases can be divided in two types:**
  - **Development simplicity**
  - **Operational simplicity**
- **While MongoDB is known for an easy out-of-the-box experience, Cassandra is known for easy to manage at scale.**

Differences between Cassandra & MongoDB

Cassandra	Mongodb
Cassandra is high performance distributed database system.	MongoDB is cross-platform document-oriented database system.
Cassandra is written in Java.	MongoDB is written in C++.
Cassandra stores data in tabular form like SQL format.	MongoDB stores data in JSON format.
Cassandra is got license by Apache.	MongoDB is got license by AGPL and drivers by Apache.
Cassandra is mainly designed to handle large amounts of data across many commodity servers.	MongoDB is designed to deal with JSON-like documents and access applications easier and faster.
Cassandra provides high availability with no single point of failure.	MongoDB is easy to administer in the case of failure.