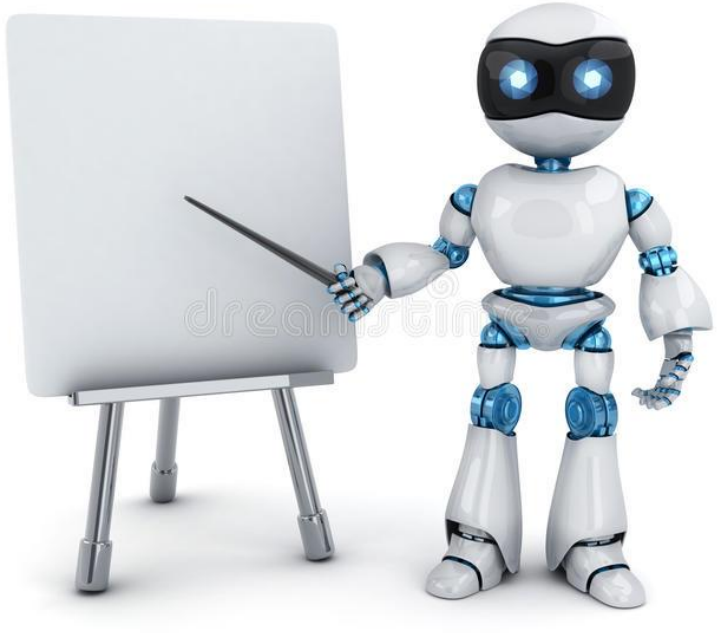


Document DB/No-SQL DB

Datamites

What is Document Database ?

- A document database is a type of database that is designed to store and retrieve documents, which are typically represented in a format such as JSON (JavaScript Object Notation). These databases are designed to be flexible, allowing for the storage of a wide variety of data types, including text, numbers, images, and more.
- Document databases are often used to store large volumes of data that is semi-structured or unstructured, meaning that it does not fit neatly into a traditional table-based database structure. They are also often used to store data that is frequently changing or needs to be accessed quickly, as they are typically faster to query than traditional databases.
- In a document database, each document is typically stored as a separate entity, and can have its own unique set of fields and data. This allows for a high degree of flexibility, as the structure of the data can be changed easily without the need to update the database schema.
- Some examples of popular document databases include MongoDB and Couchbase.

Datamites

Why Document DB ?

There are several advantages of using a document database over a traditional relational database:

- Flexibility: Document databases allow for a high degree of flexibility in terms of data storage, as each document can have its own unique set of fields and data. This is in contrast to a relational database, where data must be structured in a specific way and conform to a predefined schema.
- Scalability: Document databases are often more scalable than relational databases, as they can handle large volumes of data more efficiently. This is due to their ability to store data in a flexible, semi-structured format, which allows them to scale horizontally by distributing data across multiple servers.
- Performance: Document databases are typically faster to query than relational databases, as they do not require complex joins between tables. This makes them well-suited for applications that require fast data access, such as real-time analytics or online e-commerce systems.

Datamites

Document DB vs SQL DB

Document databases and SQL databases are both types of databases that are used to store and retrieve data. However, there are some key differences between the two types of databases:

- Data structure: Document databases store data in a semi-structured format, typically using a format such as JSON (JavaScript Object Notation). This allows for a high degree of flexibility, as each document can have its own unique set of fields and data. In contrast, SQL databases store data in a structured format, using tables with rows and columns.
- Data modeling: Document databases allow for more natural data modeling, as they can store data in a way that more closely resembles the way it is represented in the real world. This can make it easier to store and retrieve data, as it requires fewer relationships and joins between tables. In contrast, SQL databases require the user to define a schema upfront, which can make data modeling more difficult.
- Query language: Document databases typically use a query language that is different from SQL, such as MongoDB's MongoDB Query Language (MQL). SQL databases, on the other hand, use the Structured Query Language (SQL) for querying data.

Datamites

Document DB vs SQL DB

- Performance: Document databases are typically faster to query than SQL databases, as they do not require complex joins between tables. This makes them well-suited for applications that require fast data access, such as real-time analytics or online e-commerce systems.
- Use cases: Document databases are often used for applications that require the storage of large volumes of semi-structured or unstructured data, such as web-based applications, mobile apps, and real-time analytics systems. SQL databases, on the other hand, are often used for applications that require the storage of structured data and support transactions, such as financial systems and customer relationship management (CRM) systems.

Datamites

Popular Document DBs

- **MongoDB**: MongoDB is a popular open-source document database that is widely used for storing large volumes of semi-structured data. It uses the MongoDB Query Language (MQL) for querying data, and supports a wide range of data types, including text, numbers, images, and more.
- **Couchbase**: Couchbase is a distributed NoSQL document database that is known for its scalability and high performance. It uses the Couchbase Query Language (CBL) for querying data, and supports a wide range of data types.
- **Amazon DocumentDB**: Amazon DocumentDB is a managed document database service offered by Amazon Web Services (AWS). It is fully compatible with the MongoDB API, and uses the MongoDB Query Language (MQL) for querying data.
- **Azure Cosmos DB**: Azure Cosmos DB is a globally distributed, multi-model database service offered by Microsoft Azure. It supports a variety of data models, including document, key-value, graph, and column-family, and can be accessed using a range of APIs, including the MongoDB API.
- **Google Cloud Firestore**: Google Cloud Firestore is a document database service offered by Google Cloud. It is designed to be used with the Google Cloud Firebase platform, and supports real-time updates and offline access to data.

Datamites

MongoDB

MongoDB is a popular open-source document database that is widely used for storing large volumes of semi-structured data. Here are some basic concepts and features of MongoDB:

- Documents: In MongoDB, data is stored in the form of documents, which are similar to JSON objects. Each document can have its own unique set of fields and data, and can be easily modified or updated.
- Collections: A collection is a group of related documents in MongoDB. Collections are similar to tables in a relational database and can be used to store data for a particular purpose or topic.
- Query language: MongoDB uses the MongoDB Query Language (MQL) for querying data. MQL is a flexible and expressive language that allows users to filter, sort, and transform data in a variety of ways.

Datamites

MongoDB Data Format

- In MongoDB, data is stored in the form of documents, which are similar to JSON objects. A document in MongoDB is a collection of key-value pairs, where the keys are strings and the values can be any of a number of data types, including strings, numbers, arrays, and nested documents.
- Here is an example of a simple document in MongoDB:

```
{
    "name": "John Smith",
    "age": 30,
    "location": "New York",
    "hobbies": ["running", "reading", "cooking"]
}
```

Datamites

MongoDB Data Format

- In MongoDB, documents can contain any number of fields, and the data types of those fields can vary. For example, the following document contains a mix of string, numeric, and Boolean fields:

```
{
    "product_name": "Shampoo",
    "price": 5.99,
    "in_stock": true,
    "description": "A nourishing shampoo for all hair types."
}
```

MongoDB Data Format

- Documents can also contain nested documents and arrays. For example, consider the following document:

```
{
    "name": "John Smith",
    "age": 30,
    "location": {
        "city": "New York",
        "state": "NY",
        "zipcode": 10001
    },
    "hobbies": ["running", "reading", "cooking"]
}
```

Datamites

MongoDB Key methods

- **insertOne() :** This method is used to insert a single document into a collection.
- **insertMany()** : This method is used to insert multiple documents into a collection.
- **find() :** This method is used to retrieve documents from a collection. It takes a query object as an argument, which can be used to specify criteria for filtering the documents that are returned
- **updateOne()** :This method is used to update a single document in a collection. It takes a query object and an update object as arguments, and can be used to modify specific fields in the matching document.
- **updateMany()** : This method is used to update multiple documents in a collection. It takes a query object and an update object as arguments, and can be used to modify specific fields in all matching documents.

Datamites

MongoDB Key methods

- **deleteOne() :** This method is used to delete a single document from a collection. It takes a query object as an argument, and can be used to specify criteria for determining which document should be deleted.
- **deleteMany()** : This method is used to delete multiple documents from a collection. It takes a query object as an argument, and can be used to specify criteria for determining which documents should be deleted.

Datamites

Aggregation

- In MongoDB, the aggregation framework is a pipeline for data aggregation that allows you to process and analyze large data sets. It provides a way to perform complex data processing and analysis on data stored in MongoDB collections, and returns the results in the form of a cursor or as a document.
- The aggregation pipeline consists of a series of stages, each of which processes the documents as they pass through the pipeline. Each stage transforms the documents as they pass through and passes the resulting documents down the pipeline.

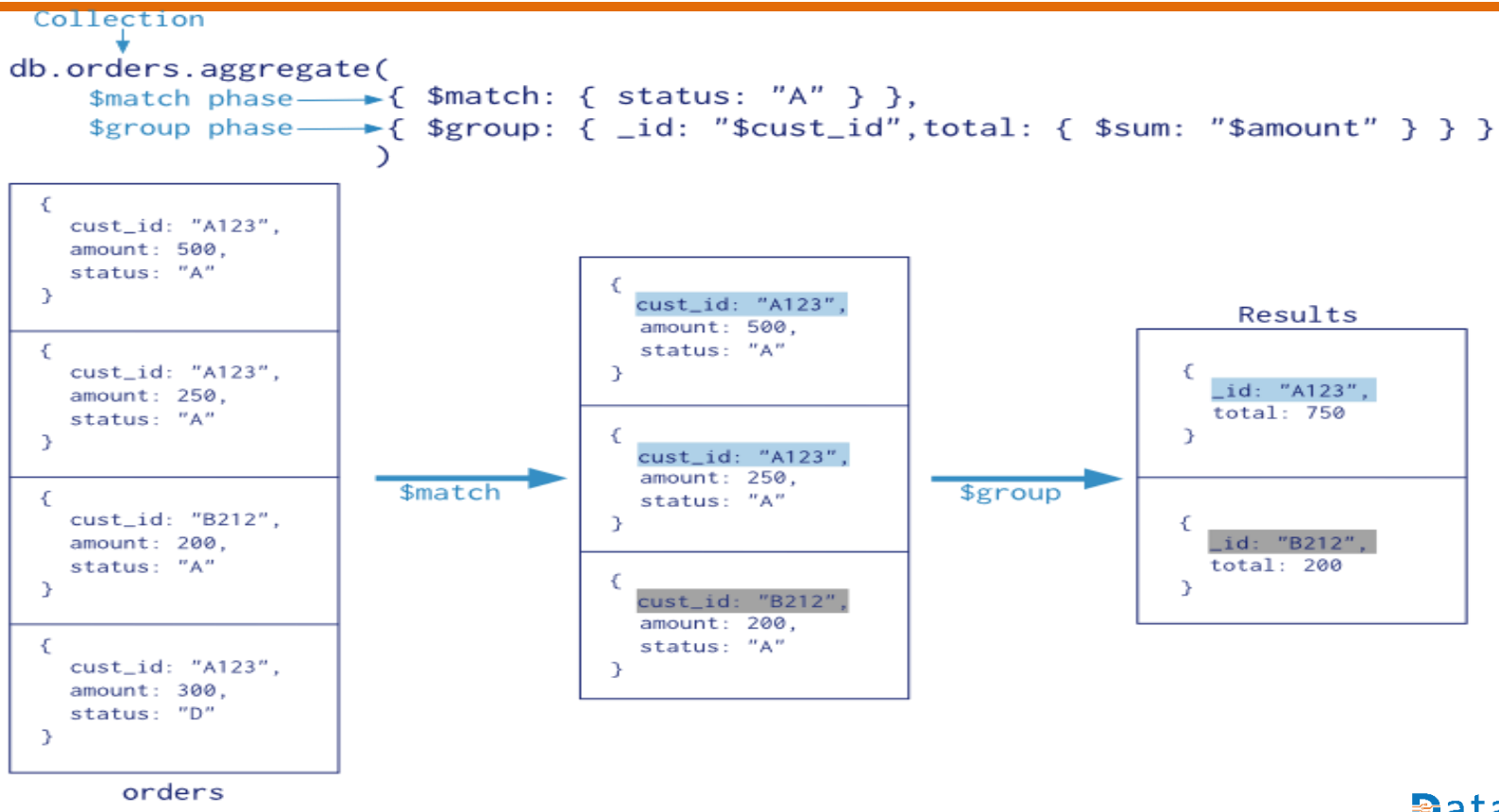
```
db.collection.aggregate([
   { $match: { status: "A" } },
   { $group: { _id: "$cust_id", total: { $sum: "$amount" } } }
])
```

Datamites

Aggregation

- This example will match all documents with the '**status**' field set to "A" and group the matching documents by '**cust_id**' calculating the total ' **amount'** for each group. The resulting documents will contain the '**cust_id'** and the calculated '**total'** for each group

Aggregation



Datamites