What is NoSQL:
https://youtu.be/qUV2j3XBRHc
NoSQL is not a relational database. It provides more flexibility since all records are not restricted by the same column names and types defined across the entire table.

Its main characteristic is its non-adherence to Relational Database Concepts. NOSQL means "Not only SQL".

Concept of NoSQL databases grew with internet giants such as Google, Facebook, Amazon etc who deal with gigantic volumes of data.

When you use relational database for massive volumes of data , the system starts getting slow in terms of response time.

To overcome this , we could of course "scale up" our systems by upgrading our existing hardware.

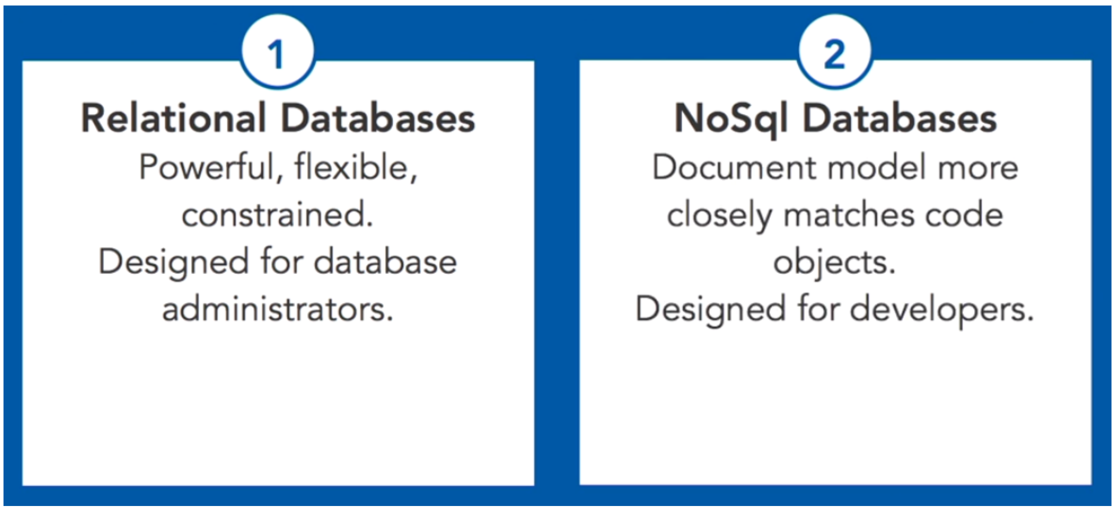
The alternative to the above problem would be to distribute our database load on multiple hosts as the load increases.

This is known as "scaling out".

NOSQL database are non-relational databases that scale out better than relational databases and are designed with web applications in mind.

They do not use SQL to query the data and do not follow strict schemas like relational models.With NoSQL, ACID (Atomicity, Consistency, Isolation, Durability) features are not guaranteed always.

Why NoSQL?


1
Relational Databases
Powerful, flexible, constrained.
Designed for database administrators.

2
NoSql Databases
Document model more closely matches code objects.
Designed for developers.

MongoDB:
Mongodb is a document-oriented NoSQL database used for high volume data storage.

MongoDB Document Structure:

Documents = JSON objects
Store data as BSON(Binary JSON)
Easy to access related information
Flexible indexing capability
Easy to adopt to common coding practices

Common Terms in MongoDB

Below are the a few of the common terms used in MongoDB

1. **JSON** – This is known as JavaScript Object Notation. This is a human-readable, plain text format for expressing structured data. JSON is currently supported in many programming languages.
2. **_id** – This is a field required in every MongoDB document. The _id field represents a unique value in the MongoDB document. The _id field is like the document's primary key. If you create a new document without an _id field, MongoDB will automatically create the field.
3. **Database** – This is a container for collections like in RDMS wherein it is a container for tables. Each database gets its own set of files on the file system. A MongoDB server can store multiple databases.
4. **Collection** – This is a grouping of MongoDB documents. A collection is the equivalent of a table which is created in any other RDMS such as Oracle or MS SQL. A collection exists within a single database. As seen from the introduction collections don't enforce any sort of structure.
5. **Document** - A record in a MongoDB collection is basically called a document. The document in turn will consist of field name and values.
6. **Field** - A name-value pair in a document. A document has zero or more fields. Fields are analogous to columns in relational databases.

   The following diagram shows an example of Fields with Key value pairs. So in the example below CustomerID and 11 is one of the key value pair's defined in the document.

   { CustomerID: 11 , CustomerName : XXX, OrderID :111} --Examples of Key value pair

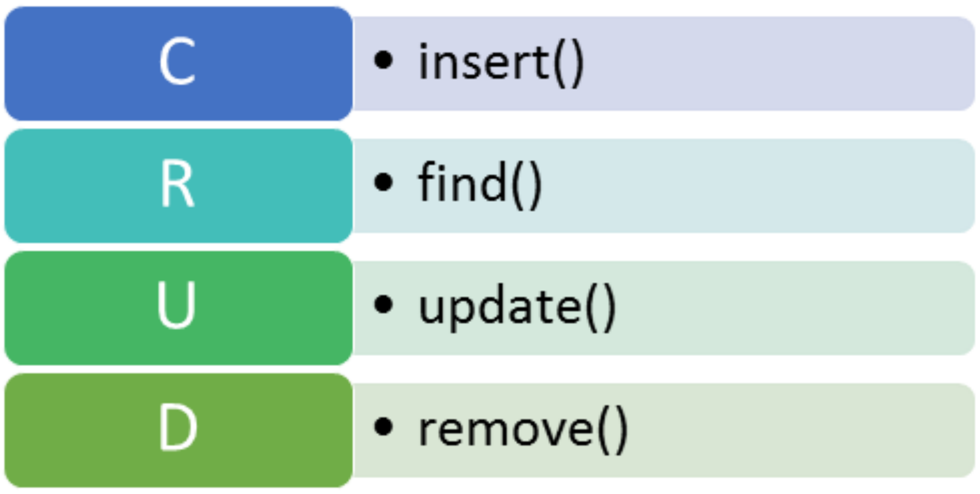
Just a quick note on the key difference between the _id field and a normal collection field. The _id field is used to uniquely identify the documents in a collection and is automatically added by MongoDB when the collection is created.

Key term differences between MongoDB & RDBMS:

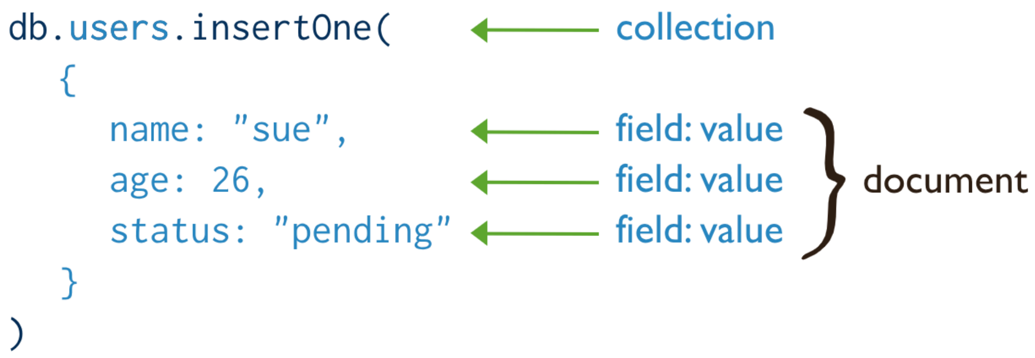
RDBMS	**MongoDB**	**Difference**
Table	Collection	In RDBMS, the table contains the columns and rows which are used to store the data where as, in MongoDB, this same structure is known as a collection. The collection contains documents which in turn contains Fields, which in turn are key-value pairs.
Row	Document	In RDBMS, the row represents a single, implicitly structured data item in a table. In MongoDB, the data is stored in documents.
Column	Field	In RDBMS, the column denotes a set of data values. These in MongoDB are known as Fields.
Joins	Embedded documents	In RDBMS, data is sometimes spread across various tables and in order to show a complete view of all data, a join is sometimes formed across tables to get the data. In MongoDB, the data is normally stored in a single collection, but separated by using Embedded documents. So there is no concept of joins in

MongoDB CRUD concepts:

We will see how to use CRUD operations in MongoDB and how they differ from SQL.

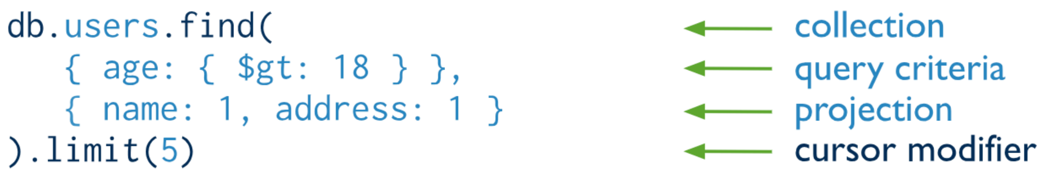


1) Insert



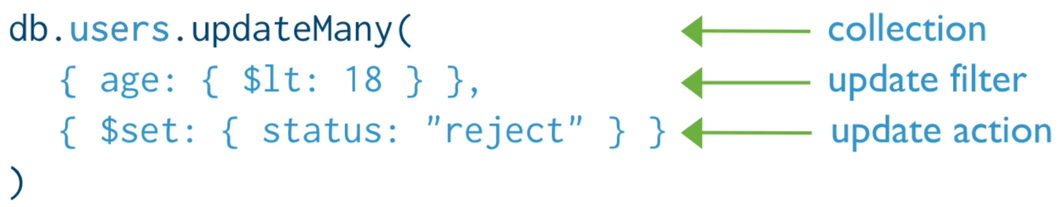
```
db.users.insertOne(
   {
      name: "sue",
      age: 26,
      status: "pending"
   }
)
```

2) Find



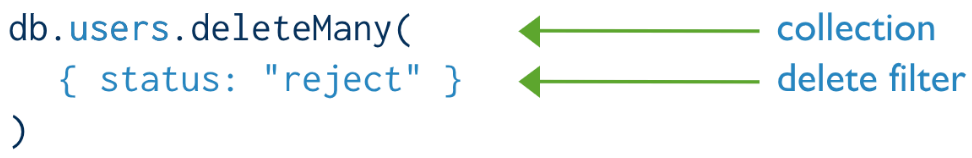
```
db.users.find(
   { age: { $gt: 18 } },
   { name: 1, address: 1 }
).limit(5)
```

3) Update



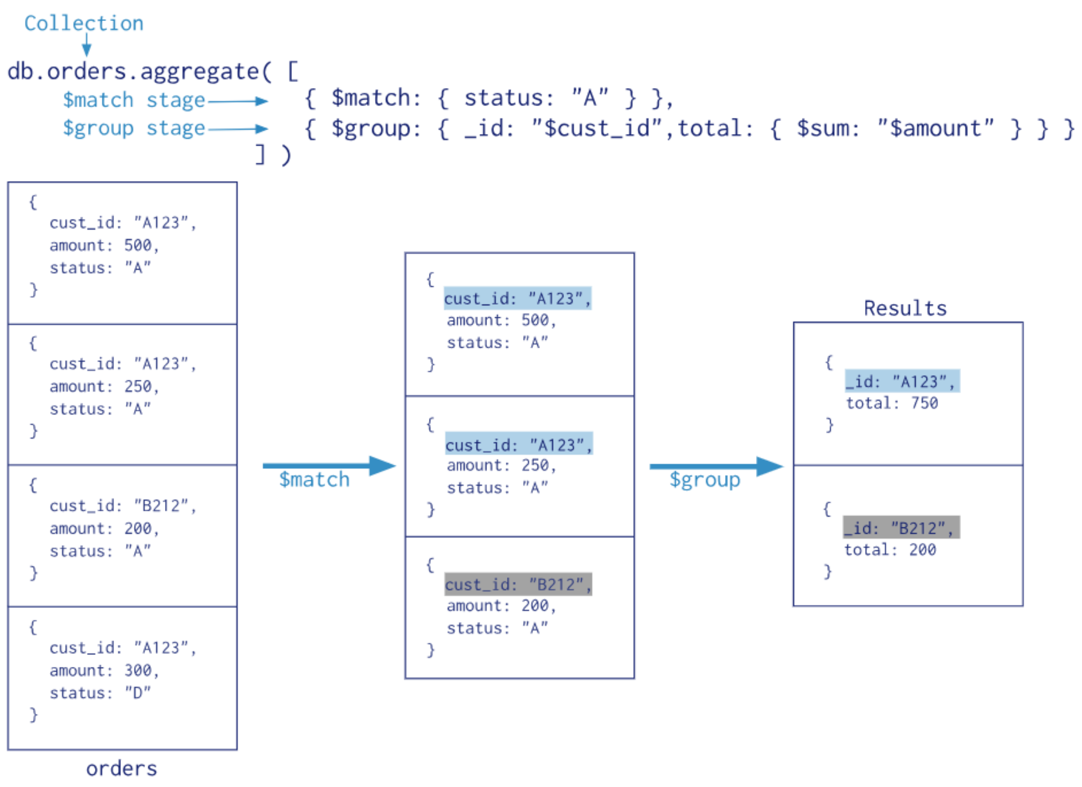
```
db.users.updateMany(
   { age: { $lt: 18 } },
   { $set: { status: "reject" } }
)
```

4) Remove



```
db.users.deleteMany(
   { status: "reject" }
)
```

Along with CRUD concepts we also cover some aggregations using MongoDB.

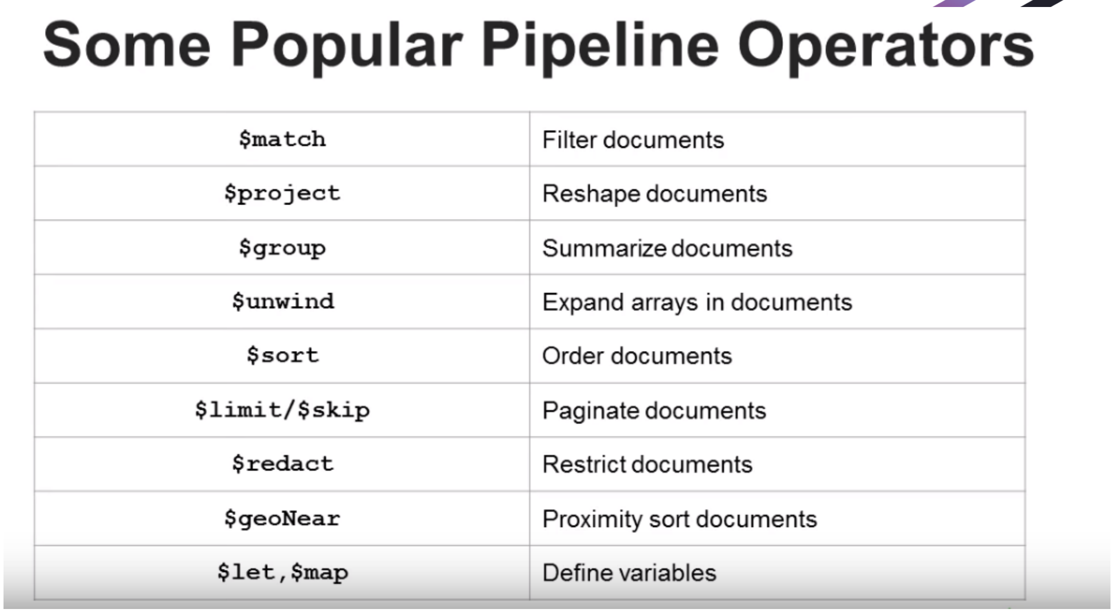


```
db.orders.aggregate( [
    { $match: { status: "A" } },
    { $group: { _id: "$cust_id",total: { $sum: "$amount" } } }
] )
```

```
{
  cust_id: "A123",
  amount: 500,
  status: "A"
}
{
  cust_id: "A123",
  amount: 250,
  status: "A"
}
{
  cust_id: "B212",
  amount: 200,
  status: "A"
}
{
  cust_id: "A123",
  amount: 300,
  status: "D"
}
```

```
{
  cust_id: "A123",
  amount: 500,
  status: "A"
}
{
  cust_id: "A123",
  amount: 250,
  status: "A"
}
{
  cust_id: "B212",
  amount: 200,
  status: "A"
}
```

```
{
  _id: "A123",
  total: 750
}
{
  _id: "B212",
  total: 200
}
```



Some Popular Pipeline Operators

`$match`	Filter documents
`$project`	Reshape documents
`$group`	Summarize documents
`$unwind`	Expand arrays in documents
`$sort`	Order documents
`$limit/$skip`	Paginate documents
`$redact`	Restrict documents
`$geoNear`	Proximity sort documents
`$let,$map`	Define variables

Best MongoDB GUI Tools:

Navicat for MongoDB
NoSQLBooster
MongoDB Compass
Studio 3T
Nucleon Database Master
NoSQL Manager
Mongo Management studio
MongoJs Query Analyzer
NoSQLClient