Lecture-18 /19

MongoDB Deep Dive & Mongoose

Coding Blocks - Kartik Mathur

Class Agenda

01	MongoDB		
02	Mongoose		
03	-	04	-
05	-		
06	-	07	-

MongoDB

1

MongoDB

- MongoDB installation
- Understanding CRUD operations on MongoDB
- Server Documentation vs Driver Documentation
- NodeJS and MongoDB: CRUD Operations

MongoDB

JOIN in mongoDB

MongoDB doesn’t have join like SQL

We can achieve them, using certain functions in mongoDB

LET US SEE IT IN ACTION.....

Orders JSON : https://gist.github.com/Kartik-Mathur/9c7508473587fd64408690c3b336245b
Customers JSON: https://gist.github.com/Kartik-Mathur/b6cd34b0ab5c233b5e9a1b83d9c3daba

You can add them to mongoDB compass and continue with the queries, ahead.

MongoDB

Consider this to make things easier:

Customer ID	Name	Email
c101	Kartik	kartik@cb.com
c102	Kanak	kanak@cb.com
c103	Abhishek	abhishek@cb.com
c104	Monu	monu@cb.com
c105	Aayush	aayush@cb.com

Order ID	Item	Customer Name (customerId)
1001	Laptop	Kartik (c101)
1002	Mouse	Kanak (c102)
1003	Keyboard	Kanak (c102)
1004	Monitor	Kanak (c102)
1005	USB Cable	Abhishek (c103)
1006	Charger	Abhishek (c103)

Joins in MongoDB

MongoDB doesn’t have Traditionals joins like SQL

Joins are:

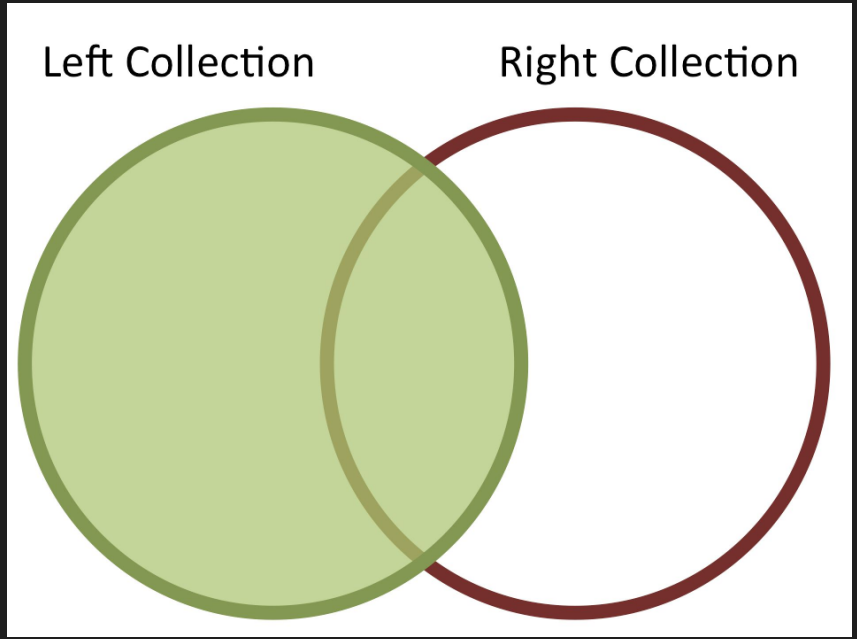
1. Left outer join: Returns **all documents** from the left collection, and matching documents from the right. If no match, right side will be empty array. THIS IS THE DEFAULT JOIN by MONGODB

```
db.orders.aggregate([
  {
    $lookup: {
      from: "customers",
      localField: "customerId",
      foreignField: "_id",
      as: "customerDetails"
    }
  }
]);
```

Left outer join

LEFT JOIN

[Customers] [Orders]
Kartik → Laptop
Kanak → Mouse, Keyboard, Monitor
Abhishek → USB Cable, Charger
Monu → ❌ No match (Still shown)
Aayush → ❌ No match (Still shown)



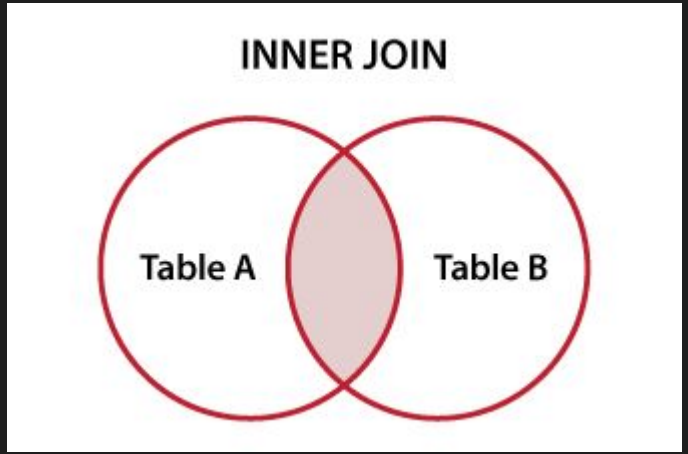
Inner JOIN

```
db.customers.aggregate([
  {
    $lookup: {
      from: "orders",
      localField: "_id",
      foreignField: "customerId",
      as: "orderDetails"
    }
  },
  { $unwind: "$orderDetails" } // removes empty arrays
]);
```

Inner JOIN

INNER JOIN

[Customers with Orders only]
Kartik → Laptop
Kanak → Mouse
Kanak → Keyboard
Kanak → Monitor
Abhishek → USB Cable
Abhishek → Charger
❌ Monu and Aayush are removed



Understanding $unwind

```
{
  name: "Kartik",
  hobbies: ["Coding", "Gaming", "Reading"]
}
```

```
db.users.aggregate([
  { $unwind: "$hobbies" }
]);
```

```
{ name: "Kartik", hobbies: "Coding" }
{ name: "Kartik", hobbies: "Gaming" }
{ name: "Kartik", hobbies: "Reading" }
```

How to unwind the LEFT outer join?

LEFT JOIN + UNWIND

Kartik → Laptop
Kanak → Mouse
Kanak → Keyboard
Kanak → Monitor
Abhishek → USB Cable
Abhishek → Charger
Monu → empty
Aayush → empty

```
db.customers.aggregate([
  {
    $lookup: {
      from: "orders",
      localField: "_id",
      foreignField: "customerId",
      as: "orderDetails"
    }
  },
  {
    $unwind: {
      path: "$orderDetails",
      preserveNullAndEmptyArrays: true
    }
  }
]);
```

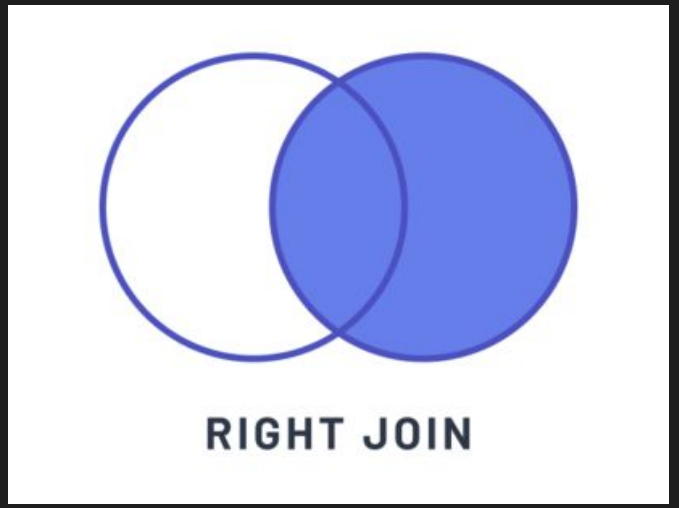
Right JOIN

```
db.orders.aggregate([
  {
    $lookup: {
      from: "customers",
      localField: "customerId",
      foreignField: "_id",
      as: "customerInfo"
    }
  },
  { $unwind: "$customerInfo" }
]);
```

Here we query on orders and not customers

RIGHT JOIN

Laptop → Kartik
Mouse → Kanak
Keyboard → Kanak
Monitor → Kanak
USB Cable → Abhishek
Charger → Abhishek
[No Monu or Aayush – since no orders]



Pipelines

In MongoDB, a pipeline is a sequence of stages that process documents step-by-step, used inside:

- **The main `aggregate()` function**
- **Operators like `$lookup` and `$unionWith`**

Pipelines Structure

```
db.collection.aggregate([
        { $stage1: { ... } },
        { $stage2: { ... } },
        ...
]);
```

Stage	Purpose
`$match`	Filter documents (like SQL `WHERE`)
`$project`	Pick or reshape fields (like SQL `SELECT`)
`$group`	Group by a field and perform aggregations
`$sort`	Sort the documents
`$limit`	Limit number of output documents
`$skip`	Skip documents (use with pagination)
`$lookup`	Join with another collection (like `JOIN`)
`$unwind`	Flatten array fields into separate docs
`$addFields`	Add new computed fields
`$count`	Count total documents after filtering
`$unionWith`	Combine documents from another collection

Pipelines: Consider the json

```
[
  { name: "DSA with C++", mentor: "Kartik", duration: 12 },

  { name: "Web Development", mentor: "Kanak", duration: 10 },

  { name: "Data Science", mentor: "Abhishek", duration: 14 },

  { name: "Android Development", mentor: "Monu", duration: 8 },

  { name: "Machine Learning", mentor: "Aayush", duration: 16 }
]
```

For json visit: https://gist.github.com/Kartik-Mathur/b51726b86aedcf0c5289215b9988386e

Pipelines: Consider the json

Task:

1. Show only the courses taught by **Kartik**, **Kanak**, and **Abhishek**
2. Show only the `name` and `mentor` (hide duration)
3. Add a field `platform: "Coding Blocks"`

Pipelines: Consider the json

Task:

1. Show only the courses taught by **Kartik**, **Kanak**, and **Abhishek**
2. Show only the `name` and `mentor` (hide duration)
3. Add a field `platform: "Coding Blocks"`

```
db.courses.aggregate([
  {
    $match: { mentor: { $in: ["Kartik", "Kanak", "Abhishek"] } }
  },
  {
    $project: { name: 1, mentor: 1 }
  },
  {
    $addFields: { platform: "Coding Blocks" }  // it will be added as extra field
  }
]);
```

Output to pipeline query

```
db.courses.aggregate([
  {
    $match: { mentor: { $in: ["Kartik", "Kanak", "Abhishek"] } }
  },
  {
    $project: { name: 1, mentor: 1 }
  },
  {
    $addFields: { platform: "Coding Blocks" }  // it will be added as extra field
  }
]);
```

```
[
  { name: "DSA with C++", mentor: "Kartik", platform: "Coding Blocks" },
  { name: "Web Development", mentor: "Kanak", platform: "Coding Blocks" },
  { name: "Data Science", mentor: "Abhishek", platform: "Coding Blocks" }
]
```

You are GOOD TO GO with MONGODB now

Mongoose

2

Mongoose

Mongoose is a Node.js-based Object Data Modeling (ODM) library for MongoDB, a popular NoSQL database. It provides a way to interact with MongoDB using a schema-based, object-oriented approach, making it easier for developers to manage data in their applications.

Let us see Mongoose in action!

Mongoose: Structure