UNIT V – Physical Storage and MongoDB

Data Storage and Indexes – RAID- File organization-Indexing and Hashing –Ordered Indices – B+ tree Index Files – B tree Index Files – Static Hashing – Dynamic Hashing. Query Processing Overview-Algorithms for Selection and Sorting Basics of MongoDB, Procedural Language

Algorithms for Selection and Sorting Basics of MongoDB

Sorting documents in MongoDB is the process of arranging data in a specific order based on field values, enhancing the efficiency and readability of query results.

Sorting

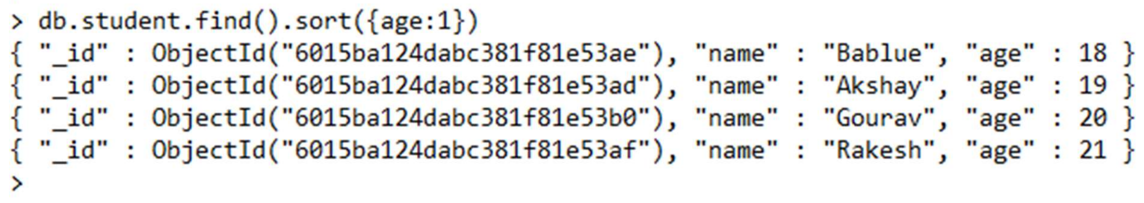
- Sorting documents in MongoDB refers to the process of arranging the documents in a specified order based on the values of one or more fields.
- This is typically done to make data retrieval more efficient and the resulting data set more useful and readable. MongoDB provides the sort() method to perform this operation, allowing users to specify the sort order as ascending (1) or descending (-1).

Syntax : db.Collection_name.sort({field_name : 1 or -1})

- Parameter: This method takes a document that contains a field: value pair. If the value of this field is 1 then this method sorts the documents in ascending order, or if the value of this field is -1 then this method sorts the documents in descending order.
- Return: This method return sorted documents.

Example : Return all the documents in ascending order of the age

db.student.find().sort({age:1})



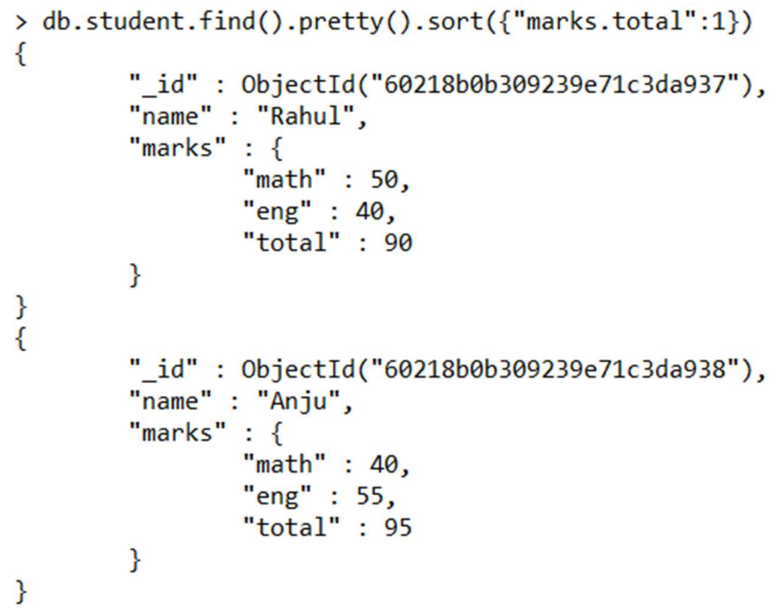
```
> db.student.find().sort({age:1})
{ "_id" : ObjectId("6015ba124dabc381f81e53ae"), "name" : "Bablue", "age" : 18 }
{ "_id" : ObjectId("6015ba124dabc381f81e53ad"), "name" : "Akshay", "age" : 19 }
{ "_id" : ObjectId("6015ba124dabc381f81e53b0"), "name" : "Gourav", "age" : 20 }
{ "_id" : ObjectId("6015ba124dabc381f81e53af"), "name" : "Rakesh", "age" : 21 }
>
```

Sorting embedded documents

Syntax : db.Collection_name.sort({“field_name.embed_field_name” : 1 or -1})

Sort documents in ascending order according to the total field of the marks document:

db.student.find().pretty().sort({"marks.total":1})



```
> db.student.find().pretty().sort({"marks.total":1})
{
        "_id" : ObjectId("60218b0b309239e71c3da937"),
        "name" : "Rahul",
        "marks" : {
                "math" : 50,
                "eng" : 40,
                "total" : 90
        }
}
{
        "_id" : ObjectId("60218b0b309239e71c3da938"),
        "name" : "Anju",
        "marks" : {
                "math" : 40,
                "eng" : 55,
                "total" : 95
        }
}
```

Sorting Multiple Documents

Syntax: db.teacher.find().pretty().sort({subject:1, age:1})

Searching

MongoDB **Text Search** allows searching for **specific text values** in documents stored within a collection. When we perform a text search query always remember that our collection must contain a text index and a collection can only contain one text index but this single text index covers multiple fields. To create a text index in MongoDB use the createIndex() method.

- Text indexes must be created on fields that store strings or arrays of strings.
- A collection can have only one text index, but this index can cover multiple fields.
- MongoDB’s text search ignores case and diacritics unless specified otherwise.

Syntax: db.collectionName.createIndex({ field: “text” })

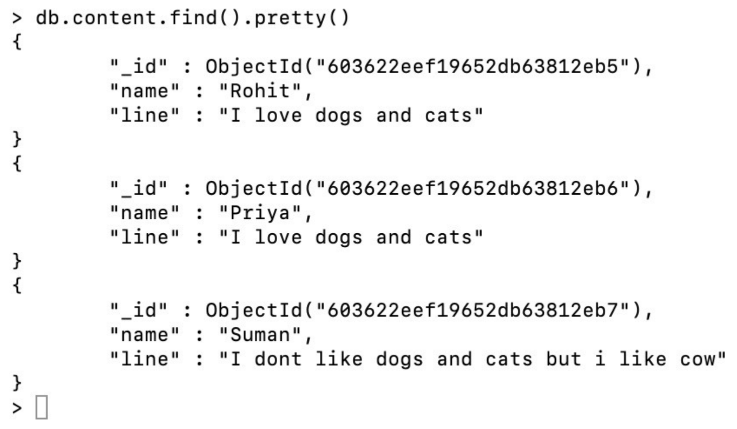
Syntax:

```
$text:
{
    $search: <string>,    $language: <string>,    $caseSensitive: <boolean>,
    $diacriticSensitive: <boolean>
}
```

Key Terms

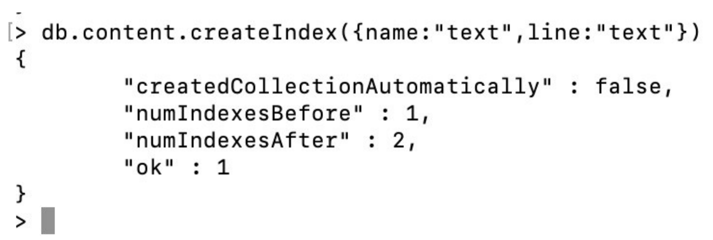
- **$search –** The text to search for.
- **$language –** (Optional) Specifies the language for tokenization.
- **$caseSensitive –** (Optional) Enables case-sensitive search.
- **$diacriticSensitive –** (Optional) Enables diacritic-sensitive search.

Create a Collection and Insert Documents



```
> db.content.find().pretty()
{
        "_id" : ObjectId("603622eef19652db63812eb5"),
        "name" : "Rohit",
        "line" : "I love dogs and cats"
}
{
        "_id" : ObjectId("603622eef19652db63812eb6"),
        "name" : "Priya",
        "line" : "I love dogs and cats"
}
{
        "_id" : ObjectId("603622eef19652db63812eb7"),
        "name" : "Suman",
        "line" : "I dont like dogs and cats but i like cow"
}
>
```

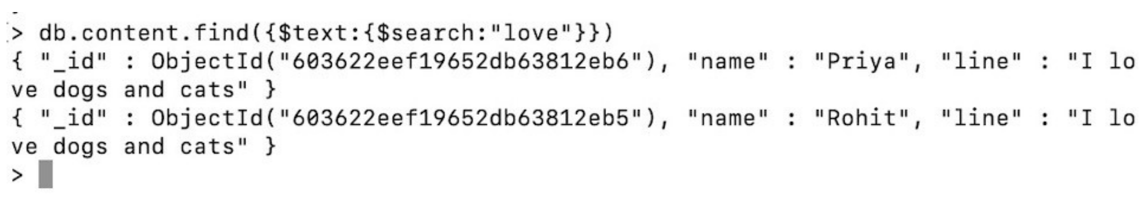
Create Index



```
> db.content.createIndex({name:"text",line:"text"})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
>
```

Search

db.content.find({$text:{$search:"love"}})



```
> db.content.find({$text:{$search:"love"}})
{ "_id" : ObjectId("603622eef19652db63812eb6"), "name" : "Priya", "line" : "I lo
ve dogs and cats" }
{ "_id" : ObjectId("603622eef19652db63812eb5"), "name" : "Rohit", "line" : "I lo
ve dogs and cats" }
>
```