How can I sort by multiple fields in MongoDB?

Authored by

stats writer

June 30, 2024

MongoDB is a popular NoSQL database management system that allows for efficient and flexible storage and retrieval of data. One useful feature of MongoDB is the ability to sort data by multiple fields, which can provide more comprehensive and specific results. To sort by multiple fields in MongoDB, the user can use the "sort" function and specify the fields to sort by in the desired order. This allows for the data to be organized and retrieved based on multiple criteria, providing a more tailored and efficient search experience. Sorting by multiple fields in MongoDB can greatly enhance the usability and effectiveness of the database for various applications and industries.

MongoDB: Sort by Multiple Fields

You can use the following syntax to sort documents in MongoDB by multiple fields:

db.myCollection.find().sort({ "field1": 1, "field2": -1 })

This particular code sorts the documents in the collection called myCollection first by field1 ascending and then by field2 descending.

The following examples show how to use this syntax with a collection teams with the following documents:

db.teams.insertOne({team: "Mavs", points: 30, rebounds: 8})
db.teams.insertOne({team: "Spurs", points: 30, rebounds: 12})
db.teams.insertOne({team: "Rockets", points: 20, rebounds: 7})

```
db.teams.insertOne({team: "Warriors", points: 25, rebounds: 5})
db.teams.insertOne({team: "Cavs", points: 25, rebounds: 9})
```

Example 1: Sort by Multiple Fields in MongoDB (Ascending)

We can use the following code to sort the documents in the teams collection first by "points" ascending and then by "rebounds" ascending:

```
db.teams.find().sort( { "points": 1, "rebounds": 1 } )
```

This query returns the following results:

```
{ _id: ObjectId("61f952c167f1c64a1afb203b"),
team: 'Rockets',
points: 20,
rebounds: 7 }
{ _id: ObjectId("61f952c167f1c64a1afb203c"),
team: 'Warriors',
points: 25,
rebounds: 5 }
{ _id: ObjectId("61f952c167f1c64a1afb203d"),
team: 'Cavs',
```

points: 25,
rebounds: 9 }
{ _id: ObjectId("61f952c167f1c64a1afb2039"),
team: 'Mavs',
points: 30,
rebounds: 8 }
{ _id: ObjectId("61f952c167f1c64a1afb203a"),
team: 'Spurs',
points: 30,
rebounds: 12 }

Notice that the documents are sorted by the "points" field ascending (smallest to largest) and then by the "rebounds" field ascending (smallest to largest).

Example 2: Sort by Multiple Fields in MongoDB (Descending)

We can use the following code to sort the documents in the teams collection first by "points" descending and then by "rebounds" descending:

```
db.teams.find().sort( { "points": -1, "rebounds": -1 } )
```

This query returns the following results:

{ _id: ObjectId("61f952c167f1c64a1afb203a"),
team: 'Spurs',
points: 30,
rebounds: 12 }
{ _id: ObjectId("61f952c167f1c64a1afb2039"),
team: 'Mavs',
points: 30,
rebounds: 8 }
{ _id: ObjectId("61f952c167f1c64a1afb203d"),
team: 'Cavs',
points: 25,
rebounds: 9 }
{ _id: ObjectId("61f952c167f1c64a1afb203c"),
team: 'Warriors',
points: 25,
rebounds: 5 }
{ _id: ObjectId("61f952c167f1c64a1afb203b"),
team: 'Rockets',
points: 20,
rebounds: 7 }

Notice that the documents are sorted by the "points" field descending (largest to smallest) and then by the "rebounds" field descending (largest to smallest).

Additional Resources

The following tutorials explain how to perform other common operations in MongoDB: