

How do I add a new field in a collection in MongoDB?

Authored by
stats writer

July 2, 2024

RECOMMENDED CITATION

stats writer (2024). *How do I add a new field in a collection in MongoDB?*.

PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=165627>

Adding a new field in a collection in MongoDB refers to the process of adding a new data field or attribute to an existing document within a collection. This can be achieved by using the "update" command and specifying the collection name, document to be updated, and the new field to be added with its corresponding value. This action allows for the storage and retrieval of additional information in the document, providing more flexibility and functionality to the database.

MongoDB: Add a New Field in a Collection

You can use the following methods to add a new field to every document in a collection in MongoDB:

Method 1: Add New Field Without Values

```
db.collection.updateMany({}, {$set:{"new_field": null}})
```

Method 2: Add New Field With Specific Value

```
db.collection.updateMany({}, {$set:{"new_field": 10}})
```

Method 3: Add New Field Using Values from Existing Fields

```
db.collection.updateMany(  
{  
}  
})
```

The following examples show how to use each method with a collection teams with the following documents:

```
db.teams.insertOne({team: "Mavs", position: "Guard",  
points: 31})db.teams.insertOne({team: "Spurs",  
position: "Guard", points:  
22})db.teams.insertOne({team: "Rockets", position:  
"Center", points: 19})db.teams.insertOne({team:  
"Warriors", position: "Forward", points:  
26})db.teams.insertOne({team: "Cavs", position:  
"Guard", points: 33})
```

Example 1: Add New Field Without Values

We can use the following code to add a new field called "rebounds" with a null value to every existing document in the collection:

```
db.teams.updateMany({}, {$set:{"rebounds": null}})
```

We can use the following query to view the first few updated documents:

```
db.teams.find().limit(3)
```

This query returns the following documents:

```
{_id: ObjectId("6189325896cd2ba58ce928e5"),  
team: 'Mavs',  
position: 'Guard',  
points: 31,  
rebounds: null }
```

```
{_id: ObjectId("6189325896cd2ba58ce928e6"),  
team: 'Spurs',  
position: 'Guard',  
points: 22,  
rebounds: null }
```

```
{_id: ObjectId("6189325896cd2ba58ce928e7"),  
team: 'Rockets',  
position: 'Center',  
points: 19,  
rebounds: null }
```

Notice that each document now has a field called "rebounds" with a null value.

Example 2: Add New Field With Specific Value

We can use the following code to add a new field called

"rebounds" with a value of 10 to every existing document in the collection:

```
db.teams.updateMany({}, {$set:{"rebounds": 10}})
```

We can use the following query to view the first few updated documents:

```
db.teams.find().limit(3)
```

This query returns the following documents:

```
{ _id: ObjectId("6189325896cd2ba58ce928e5"),  
team: 'Mavs',  
position: 'Guard',  
points: 31,  
rebounds: 10 }
```

```
{ _id: ObjectId("6189325896cd2ba58ce928e6"),  
team: 'Spurs',  
position: 'Guard',  
points: 22,  
rebounds: 10 }
```

```
{ _id: ObjectId("6189325896cd2ba58ce928e7"),
```

```
team: 'Rockets',  
position: 'Center',  
points: 19,  
rebounds: 10 }
```

Notice that each document now has a field called "rebounds" with a value of 10.

Example 3: Add New Field Using Values from Existing Fields

We can use the following code to add a field called "name" whose value is a concatenation of the existing fields "team" and "position":

```
db.teams.updateMany(  
{  
}},  
[  
)
```

We can use the following query to view the first few updated documents:

```
db.teams.find().limit(3)
```

This query returns the following documents:

```
{ _id: ObjectId("618934cb96cd2ba58ce928ea"),  
team: 'Mavs',  
position: 'Guard',  
points: 31,  
name: 'Mavs Guard' }
```

```
{ _id: ObjectId("618934cb96cd2ba58ce928eb"),  
team: 'Spurs',  
position: 'Guard',  
points: 22,  
name: 'Spurs Guard' }
```

```
{ _id: ObjectId("618934cb96cd2ba58ce928ec"),  
team: 'Rockets',  
position: 'Center',  
points: 19,  
name: 'Rockets Center' }
```

Notice that each document now has a field called "name" whose value is a concatenation of the "team" and "position" fields.

Note: You can find the complete documentation for the

updateMany() function .

Additional Resources

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

ARABPSYCHOLOGY.COM