MongoDB CRUD Operations > Insert Documents

Insert Documents

This page provides examples of insert operations in the `mongo` shell.

> **CREATING A COLLECTION:**
>
> If the collection does not currently exist, insert operations will create the collection.

Insert a Document via `db.collection.insertOne()`

New in version 3.2.

`db.collection.insertOne()` inserts a *single* document into a collection.

The following example inserts a new document into the `inventory` collection. Since the document does not specify an `_id` field, MongoDB adds the `_id` field with an ObjectId value to the new document. See Insert Behavior.

```
db.inventory.insertOne(
   { item: "canvas", qty: 100, tags: ["cotton"], size: { h: 28, w: 35.5, uom: "cm" } }
)
```

You can run this method in the web shell below:

MongoDB Web Shell

Click to connect

`insertOne()` will return a document that includes the newly inserted document's `_id` field value. For an example of a return document, see db.collection.insertOne() reference.

To retrieve the document that you just inserted, query the collection:

```
db.inventory.find( { item: "canvas" } )
```

For more information and examples, see `db.collection.insertOne()`.

Insert Multiple Documents via `db.collection.insertMany()`

New in version 3.2.

`db.collection.insertMany()` can insert *multiple* documents into a collection. Pass an array of documents to the method.

The following example inserts three new documents into the `inventory` collection. Since the documents do not specify an `_id` field, MongoDB adds the `_id` field with an ObjectId value to each document. See Insert Behavior.

```
db.inventory.insertMany([
   { item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } },
   { item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },
   { item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm" } }
])
```

You can run this method in the web shell below:

`insertMany()` will return a document that includes the newly inserted documents `_id` field values. See the reference for an example.

To retrieve the inserted documents, query the collection:

```
db.inventory.find()
```

For more information and examples, see `db.collection.insertMany()`.

Insert Document or Documents via `db.collection.insert()`

`db.collection.insert()` can insert a single document or multiple documents into a collection. To insert a single document, pass a document to the method; to insert multiple documents, pass an array of documents to the method.

The following example inserts a new document into the `inventory` collection. Since the document does not specify an `_id` field, MongoDB adds the `_id` field with an ObjectId value to the document. See Insert Behavior.

```
db.inventory.insert(
   { item: "notebook", qty: 50, tags: ["red", "hard cover", "plain"], size: { h: 8.5, w: 11, uom:
)
```

You can run this method in the web shell below:

MongoDB Web Shell

Click to connect

`db.collection.insert` returns a `WriteResult` object providing status information.

For example, a successful insert returns the following `WriteResult` object:

```
WriteResult({ "nInserted" : 1 })
```

The `nInserted` field specifies the number of documents inserted. If the operation encounters an error, the `WriteResult` object will contain the error information.

The following example inserts multiple documents into the `inventory` collection. Since the documents do not specify an `_id` field, MongoDB adds the `_id` field with an ObjectId value to each document. See Insert Behavior.

```
db.inventory.insert([
   { item: "paper", qty: 100, tags: ["red", "blank", "plain"], size: { h: 11.69 , w: 16.53, uom: "
   { item: "planner", qty: 75, tags: ["blank", "red"], size: { h: 22.85, w: 30, uom: "cm" } },
   { item: "postcard", qty: 45, tags: ["blue"], size: { h: 10, w: 15.25, uom: "cm" } }
])
```

You can run this method in the web shell below:

MongoDB Web Shell

Click to connect

The method returns a `BulkWriteResult` object with the status of the operation. A successful insert of the documents returns the following `BulkWriteResult` object:

```
BulkWriteResult({
   "writeErrors" : [ ],
   "writeConcernErrors" : [ ],
   "nInserted" : 3,
   "nUpserted" : 0,
   "nMatched" : 0,
   "nModified" : 0,
   "nRemoved" : 0,
   "upserted" : [ ]
})
```

For more information and examples, see `db.collection.insert()`.

Insert Behavior

Collection Creation

If the collection does not currently exist, insert operations will create the collection.

`_id` Field

In MongoDB, each document stored in a collection requires a unique _id field that acts as a primary key. If an inserted document omits the `_id` field, the MongoDB driver automatically generates an ObjectId for the `_id` field.

This also applies to documents inserted through update operations with upsert: true.

Atomicity

All write operations in MongoDB are atomic on the level of a single document. For more information on MongoDB and atomicity, see Atomicity and Transactions

Write Acknowledgement

With write concerns, you can specify the level of acknowledgement requested from MongoDB for write operations. For details, see Write Concern.

SEE ALSO:

- `db.collection.insert()`
- `db.collection.insertOne()`
- `db.collection.insertMany()`
- Additional Methods for Inserts