Delete Documents

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

The examples on this page use the `inventory` collection. To create and/or populate the `inventory` collection, run the following in the `mongo` shell:

```
db.inventory.insert( [
   { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },
   { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "P" },
   { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },
   { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },
   { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" },
]);
```

You can run the operation in the web shell below:

Delete All Documents

To remove all documents from a collection, pass an empty filter document `{}` to either the `db.collection.deleteMany()` or the `db.collection.remove()` method.

`db.collection.deleteMany()`

The following example uses the `db.collection.deleteMany()` method to delete *all* documents from the `inventory` collection:

```
db.inventory.deleteMany({})
```

The method returns a document with the status of the operation. For more information and examples, see `db.collection.deleteMany()`.

`db.collection.remove()`

Alternatively, the following example uses the `db.collection.remove()` method to delete *all* documents from the `inventory` collection:

```
db.inventory.remove({})
```

To delete all documents from a collection, it may be more efficient to use the `db.collection.drop()` method to drop the entire collection, including the indexes, and then recreate the collection and rebuild the indexes.

Delete All Documents that Match a Condition

You can specify criteria, or filters, that identify the documents to delete. These filters use the same syntax as read operations:

- To specify equality conditions, use `<field>:<value>` expressions in the query filter document:

  ```
  { <field1>: <value1>, ... }
  ```

- A query filter document can use the query operators to specify conditions in the following form:

```
{ <field1>: { <operator1>: <value1> }, ... }
```

To delete all documents that match a deletion criteria, pass a filter parameter to either `db.collection.deleteMany()` method or the `db.collection.remove()` method.

`db.collection.deleteMany()`

The following example uses `db.collection.deleteMany()` to remove all documents from the `inventory` collection where the `status` field equals `"A"`:

```
db.inventory.deleteMany({ status : "A" })
```

The method returns a document with the status of the operation.

`db.collection.remove()`

Alternatively, the following example uses `db.collection.remove()` to remove all documents from the `inventory` collection where the `status` field equals `"P"`:

```
db.inventory.remove( { status : "P" } )
```

For large deletion operations, it may be more efficient to copy the documents that you want to keep to a new collection and then use `db.collection.drop()` on the original collection.

Remove Only One Document that Matches a Condition

To delete at most a single document that match a specified filter,even though multiple documents may match the specified filter, use either the `db.collection.deleteOne()` method or the `db.collection.remove()` method with the `<justOne>` parameter set to `true` or `1`.

`db.collection.deleteOne()`

The following example uses `db.collection.deleteOne()` to delete the *first* document where `status` is `"D"`.

```
db.inventory.deleteOne( { status: "D" } )
```

`db.collection.remove()`

Alternatively, the following example uses the `db.collection.remove()` with the `<justOne>` parameter set to `1` to delete the *first* document where `status` is `"D"`:

```
db.inventory.remove({ status: "D" }, 1)
```

Delete Behavior

Indexes

Delete operations do not drop indexes, even if deleting all documents from a collection.

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.deleteMany()`
- `db.collection.deleteOne()`
- `db.collection.remove()`
- Additional Methods