# Collections

**On this page**



MongoDB stores documents in collections. If a collection does not exist, MongoDB creates the collection when you first insert a document in that collection.

You can also explicitly create a collection with various options, such as setting the maximum size or the documentation validation rules.

## Capped Collection

Capped collections have maximum size or document counts that prevent them from growing beyond maximum thresholds. All capped collections must specify a maximum size and may also specify a maximum document count. MongoDB removes older documents if a collection reaches the maximum size limit before it reaches the maximum document count.

To create a capped collection, use the `capped: true` option along with a `size` in bytes.

```
client = Mongo::Client.new([ '127.0.0.1:27017' ], :database => 'music')
client[:artists, capped: true, size: 10000].create
```

## Convert an Existing Collection to Capped

To convert an existing collection from non-capped to capped, use the `convertToCapped` command.

```
client = Mongo::Client.new([ '127.0.0.1:27017' ], :database => 'test')
db = client.database
db.command({ 'convertToCapped' => 'contacts', 'size' => 8192 })
```

## Document Validation

If you're using MongoDB version 3.2 or later, you can use document validation. Collections with validations compare each inserted or updated document against the criteria specified in the validator option. Depending on the `validationLevel` and `validationAction`, MongoDB either returns a warning, or refuses to insert or update the document if it fails to meet the specified criteria.

The following example creates a `contacts` collection with a validator that specifies that inserted or updated documents should match at least one of three following conditions:

- the `phone` field is a string
- the `email` field matches the regular expression
- the `status` field is either `Unknown` or `Incomplete`.

```
client = Mongo::Client.new([ '127.0.0.1:27017' ], :database => 'test')
client[:contacts,

    {
       'validator' => { '$or' =>
          [
             { 'phone' => { '$type' => "string" } },
             { 'email' => { '$regex' => /@mongodb\.com$/ } },
             { 'status' => { '$in' => [ "Unknown", "Incomplete" ] } }
          ]
       }
    }

  ].create
```

## Add Validation to an Existing Collection

To add document validation criteria to an existing collection, use the `collMod` command. The example below demonstrates how to add a validation to the `contacts` collection, ensuring that all new documents must contain an `age` field which is a number.

```
client = Mongo::Client.new([ '127.0.0.1:27017' ], :database => 'test')
db = client.database
db.command({ 'collMod' => 'contacts',
             'validator' =>
               { 'age' =>
                 { '$type' => "number" }
               }
           })
```