Create Operations

Create or insert operations add new [documents](https://docs.mongodb.com/manual/core/document/#std-label-bson-document-format) to a [collection](https://docs.mongodb.com/manual/core/databases-and-collections/#std-label-collections). If the collection does not currently exist, insert operations will create the collection.

[db.collection.insertOne()](https://docs.mongodb.com/manual/reference/method/db.collection.insertOne/#mongodb-method-db.collection.insertOne)

[db.collection.insertMany()](https://docs.mongodb.com/manual/reference/method/db.collection.insertMany/#mongodb-method-db.collection.insertMany)

const cursor = db.collection(**'inventory'**).find({});

SELECT \* FROM inventory

const cursor = db.collection(**'inventory'**).find({ status: **'D'** });

SELECT \* FROM inventory WHERE status = "D"

**where status equals either "A" or "D":**

|  |
| --- |
| const cursor = db.collection(**'inventory'**).find({ |
| status: { $in: [**'A'**, **'D'**] } |
| }); |

SELECT \* FROM inventory WHERE status in ("A", "D")

## Specify OR Conditions

Using the [$or](https://docs.mongodb.com/manual/reference/operator/query/or/#mongodb-query-op.-or) operator, you can specify a compound query that joins each clause with a logical OR conjunction so that the query selects the documents in the collection that match at least one condition.

|  |
| --- |
| const cursor = db.collection(**'inventory'**).find({ |
| $or: [{ status: **'A'** }, { qty: { $lt: 30 } }] |
| }); |

SELECT \* FROM inventory WHERE status = "A" OR qty < 30

## Specify AND Conditions

|  |
| --- |
| const cursor = db.collection(**'inventory'**).find({ |
| status: **'A'**, |
| qty: { $lt: 30 } |
| }); |

SELECT \* FROM inventory WHERE status = "A" AND qty < 30

### Specify AND as well as OR Conditions

|  |
| --- |
| const cursor = db.collection(**'inventory'**).find({ |
| status: 'D',        $or: [{ qty: { $lt: 30 } }, { title: 'Niketh' }] |
|  |
| }); |

SELECT \* FROM inventory WHERE status = "A" AND ( qty < 30 OR item LIKE "p%")

## Match an Embedded/Nested Document

|  |
| --- |
| const cursor = db.collection(**'inventory'**).find({ |
| size: { h: 14, w: 21, uom: **'cm'** } |
| }); |

### Specify Match using Query Operator

|  |
| --- |
| const cursor = db.collection(**'inventory'**).find({ |
| **'size.h'**: { $lt: 15 } |
| }); |

### **What is the Aggregation Framework?**

The aggregation framework allows you to analyze your data in real time. Using the framework, you can create an aggregation pipeline that consists of one or more  stages. Each stage transforms the documents and passes the output to the next stage.

The aggregation framework has a variety of stages available for you to use. Today, we’ll discuss the basics of how to use  $match,  $group,  $sort, and  $limit. Note that the aggregation framework has many other powerful stages including  $count,  $geoNear,  $graphLookup,  $project,  $unwind, and others.

## The Aggregation Pipeline

The aggregation pipeline is a framework for data aggregation modeled on the concept of data processing pipelines. Documents enter a multi-stage pipeline that transforms the documents into aggregated results.

# **Projections**

By default, queries in MongoDB return all fields in matching documents. To limit the amount of data that MongoDB sends to applications, you can include a projection document in the query operation.

The projection document limits the fields to return for all matching documents. The projection document can specify the inclusion of fields or the exclusion of field.

we use projection to display the name and age of the employee.

 Items.find({}, { "age": 6, "list": 'pass' })

* If the value of the field is set to 1 or true, then it means the field will include in the return document.
* If the value of the field is set to 0 or false, then it means the field will not include in the return document.
* There is no need to set \_id field to 1 to return \_id field, the find() method always return \_id unless you set a \_id field to 0.

**Indexing**

|  |
| --- |
| db.products.createIndex( |
| { item: 1, quantity: -1 } , |
| { name: **"query for inventory"** } |
| ) |