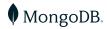
ANNOUNCEMENT: Voyage AI joins MongoDB to power more accurate and trustworthy AI applications on Atlas.





Docs Home / MongoDB Manual / CRUD Operations

Query Documents

On this page

Select All Documents in a Collection

Specify Equality Condition

Specify Conditions Using Query Operators

Specify AND Conditions

Specify OR Conditions

Specify AND as well as OR Conditions

Query Documents with MongoDB Atlas

Additional Query Tutorials

Behavior

Additional Methods and Options

To query documents, specify a query predicate indicating the documents you want to return. If you specify an empty query predicate ({ }), the query returns all documents in the collection.

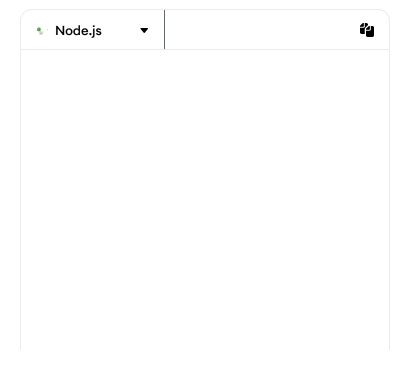
You can query documents in MongoDB by using the following methods:

- Your programming language's driver.
- The MongoDB Atlas UI. To learn more, see
 Query Documents with MongoDB Atlas.
- MongoDB Compass.

➤ Use the **Select your language** drop-down menu in the upper-right to set the language of the following examples or select MongoDB Compass.

This page provides examples of query operations using the Collection.find() method in the MongoDB Node.js Driver.

The examples on this page use the inventory collection. Connect to a test database in your MongoDB instance then create the inventory collection:



```
await db.collection('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: 'A'
  },
    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'
  }
]);
```

Select All Documents in a Collection

To select all documents in the collection, pass an empty document as the query filter parameter to the find method. The query filter parameter determines the select criteria:



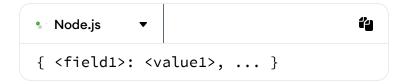
This operation uses a query predicate of {}}, which corresponds to the following SQL statement:



To see supported options for the [find()] method, see find().

Specify Equality Condition

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



The following example selects from the inventory collection all documents where the status equals "D":



This operation uses a query predicate of {
status: "D" }, which corresponds to the following SQL statement:



NOTE

The MongoDB Compass query bar autocompletes the current query based on the keys in your collection's documents, including keys in embedded sub-documents.

Specify Conditions Using Query Operators

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



The following example retrieves all documents from the inventory collection where status equals either "A" or "D":



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

NOTE

Although you can express this query using the sor operator, use the sin operator rather than the sor operator when performing equality checks on the same field.

The operation uses a query predicate of {
status: { \$in: ["A", "D"] } }, which
corresponds to the following SQL statement:

```
SELECT * FROM inventory WHERE statu
```

Refer to the Query and Projection Operators document for the complete list of MongoDB query operators.

Specify AND Conditions

A compound query can specify conditions for more than one field in the collection's documents. Implicitly, a logical AND conjunction connects the clauses of a compound query so that the query selects the documents in the collection that match all the conditions.

The following example retrieves all documents in the inventory collection where the status equals "A" and qty is less than (\$1t) 30:

```
Node.js ▼

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

The operation uses a query predicate of {
status: "A", qty: { \$lt: 30 } }, which
corresponds to the following SQL statement:

```
SELECT * FROM inventory WHERE statu 省
```

See comparison operators for other MongoDB comparison operators.

Specify OR Conditions

Singthment operator, you consispent of All 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.

The following example retrieves all documents in the collection where the status equals "A" or qty is less than (\$lt) 30:



MongoDB Manual

8.0 (current) ▼

- ▶ Introduction
- ▼ CRUD Operations
 - ▶ Insert
 - ▼ Query

Embedded Documents

Arrays

Arrays of Embedded
Documents

Drainat Paculta

C

C

רוטןכטו ועכטעונט

Null or Missing Fields

Timeouts

Long-Running Snapshots

Iterate a Cursor

- Update
- Remove

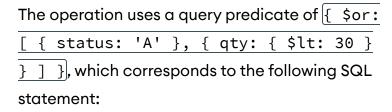
Bulk Write

Retryable Writes

Retryable Reads

SQL to MongoDB

- ▶ Text Search
- Geospatial Queries



```
SELECT * FROM inventory WHERE statu
```

NOTE

(

Queries that use comparison operators are subject to Type Bracketing.

Specify AND as well as OR Conditions

In the following example, the compound query document selects all documents in the collection where the status equals "A" and either qty is less than (\$1t) 30 or item starts with the character p:

```
Node.js ▼

const cursor = db.collection('inventory
  status: 'A',
  $or: [{ qty: { $lt: 30 } }, { item: {
});
```

The operation uses a query predicate of:

```
{
    status: 'A',
    $or: [
        { qty: { $lt: 30 } }, { item:
        }
}
```

which corresponds to the following SQL statement:

```
SELECT * FROM inventory WHERE statu
```

NOTE

MongoDB supports regular expressions [\$regex] queries to perform string pattern matches.

Query Documents with MongoDB Atlas

The example in this section uses the **sample** movies dataset. To learn how to load the sample dataset into your MongoDB Atlas deployment, see Load Sample Data.

To project fields to return from a query in MongoDB Atlas, follow these steps:

- 1 In the MongoDB Atlas UI, go to the Clusters page for your project.

Organizations menu in the navigation bar.

- b. If it's not already displayed, select your project from the **Projects** menu in the navigation bar.
- c. If it's not already displayed, click

 Clusters in the sidebar.

The Clusters page displays.

(2) Navigate to the collection

- a. For the cluster that contains the sample data, click **Browse** Collections.
- b. In the left navigation pane, select the sample_mflix database.
- c. Select the movies collection.

3 Specify the Filter field

Specify the query filter document in the **Filter** field. A query filter document uses **query operators** to specify search conditions.

Copy the following query filter document into the **Filter** search bar:

{ year: 1924 }



Click Apply

This query filter returns all documents in the sample_mflix.movies collection where the year field matches 1924.

Additional Query Tutorials

For additional query examples, see:

- Query on Embedded/Nested Documents
- Query an Array
- Query an Array of Embedded Documents
- Project Fields to Return from Query
- Query for Null or Missing Fields

Behavior

Cursor

The Collection.find() method returns a cursor.

Concurrent Updates While Using a Cursor

As a cursor returns documents, other operations may run in the background and affect the results, depending on the read concern level. For details, see Read Isolation, Consistency, and Recency.

Read Isolation

For reads to Replica sets and replica set shards, read concern allows clients to choose a level of isolation for their reads. For more information, see Read Concern.

Query Result Format

When you run a find operation with a MongoDB driver or mongosh, the command returns a cursor that manages query results. The query results are not returned as an array of documents.

To learn how to iterate through documents in a cursor, refer to your driver's documentation. If you are using mongosh, see Iterate a Cursor in mongosh.

Additional Methods and Options

The following can also read documents from a collection:

- Collection.findOne()[™]
- In aggregation pipeline, the smatch pipeline stage provides access to MongoDB queries. See the MongoDB Node.js Driver's aggregation tutorial.

NOTE

The Collection.findOne() ** method performs the same operation as the Collection.find() ** method with a limit of 1.

• English

About

Careers Investor Relations

Legal GitHub

Security Information Trust Center

Connect with Us

Support

Contact Us Customer Portal

Atlas Status Customer Support

Manage Cookies

Deployment Options

MongoDB Atlas Enterprise Advanced

Community Edition

© 2024 MongoDB, Inc.