### mongo

Allows you to browse MongoDB or transmit the input records to the file.

#### Syntax

mongo PROFILE [database=DB\_NAME] SUBCOMMANDmongo PROFILE [database=DB\_NAME] MONGODB\_METHOD

Required Parameter

**PROFILE**

MongoDB connect profile. You can configure the profile in the web console.

**SUBCOMMAND**

Command to be executed in the MongoDB session: dbs, cols

1. cols: Lists collections for the database specified by the database=DB\_NAME option. If no database is specified, all collections from all databases are shown.
2. dbs: List all databases in MongoDB. Used alone without any other arguments.

**MONGO\_METHOD**

MongoDB native method. For the collection specified by COL\_NAME, it receives an input record or expression as a parameter argument and executes the method.

Optional Parameter

**database=DB\_NAME**

Database on which to run the cols command.

#### Description

Output fields by SUBCOMMAND

The output fields of the dbs command are:

1. **name**: Database name
2. **disk\_usage**: Disk usage in bytes
3. **empty**: true if the database is empty and false otherwise.

The output fields of the cols command are:

1. **name**: Collection name
2. **type**: Type (collection)
3. **options**: Collection configuration options
4. **info**: Additional information such as whether it is read-only.
5. **idIndex**: \_id index specification

MONGO\_METHOD

The mongodb command supports the following MongoDB native methods. You can check the usage of each method in [MongoDB Reference Manual](https://www.mongodb.com/docs/manual/reference/method/).

1. [db.COL\_NAME.find()](https://www.mongodb.com/docs/manual/reference/method/db.collection.find/)
2. [db.COL\_NAME.insert()](https://www.mongodb.com/docs/manual/reference/method/db.collection.insert/)
3. [db.COL\_NAME.updateOne()](https://www.mongodb.com/docs/manual/reference/method/db.collection.updateOne/)
4. [db.COL\_NAME.updateMany()](https://www.mongodb.com/docs/manual/reference/method/db.collection.updateMany/)
5. [db.COL\_NAME.deleteOne()](https://www.mongodb.com/docs/manual/reference/method/db.collection.deleteOne/)
6. [db.COL\_NAME.deleteMany()](https://www.mongodb.com/docs/manual/reference/method/db.collection.deleteMany/)

find()

db.COL\_NAME.find("FILTER\_EXPR", ["PROJECTION\_EXPR"])

Selects documents specified by COL\_NAME in a collection and returns a cursor to the selected documents.

**"FILTER\_EXPR"**

A JSON expression that specifies search conditions (filters) If you specify it as null, it retrieves all documents.

**["PROJECTION\_EXPR"]**

A JSON expression that specifies the fields to be returned in the documents that match the search condition. If you omit the expression, it returns all fields in the documents that match the search condition. For more information, see ["Projection"](https://www.mongodb.com/docs/manual/reference/method/db.collection.find/) in MongoDB Documentation.

For information on query operators available in FILTER\_EXPR and PROJECTION\_EXPR, refer to the ["Query and Projection Operators"](https://www.mongodb.com/docs/manual/reference/operator/query/) in the MongoDB Documentation.

insert()

db.COL\_NAME.insert()

Inserts a document or documents into a collection specified by COL\_NAME. If the document contains an **\_id** value, it is used as a unique identification key (ObjectId). The **\_id** values must be unique. If the document does not specify an **\_id** field, then MongoDB adds a 12-byte hexadecimal unique identification key for the document before inserting. See usage #4, #5.

updateOne()

db.COL\_NAME.updateOne("KEY", "UPDATE")

Searches the collection specified by COL\_NAME with the value of the KEY field and modifies the value of the UPDATE field in the first document returned in the collection. See usage #6.

When using the **\_id** field as KEY, if the unique identifier is an automatically generated 12-digit binary, the \_id value MUST be set to the binary type.

updateMany()

db.COL\_NAME.updateMany("KEY\_LIST", "UPDATE\_LIST")

Searches the collection specified by COL\_NAME with the value of the KEY\_LIST field and modifies the value of the UPDATE\_LIST field in all the documents returned. Both KEY\_LIST and UPDATE\_LIST use a comma (,) as a separator. Unlike the db.COL\_NAME.updateOne() method, this modifies all searched documents.

deleteOne()

db.COL\_NAME.deleteOne("KEY")

Searches the collection specified by COL\_NAME with the value of the KEY field and deletes the first document returned in the collection.

When using the **\_id** field as KEY, if the unique identifier is an automatically generated 12-digit binary, the \_id value MUST be set to the binary type.

deleteMany()

db.COL\_NAME.deleteMany("KEY\_LIST")

Searches the collection specified by COL\_NAME with the value of the KEY\_LIST field and removes all documents. KEY\_LIST uses a comma (,) as a separator.

#### Usage

Retrieve all documents in the inventory collection.

mongo PROFILE db.inventory.find()

Retrieve documents with 3 or more stars from the restaurants collection.

mongo PROFILE db.restaurants.find("{stars: {$gte: 3}}")

Output only the name and stars fields from the restaurants collection.

mongo PROFILE db.restaurants.find(null, "{name: true, stars: true}")

Provide an arbitrary JSON document into the MongoDB restaurants collection.

json "{ name: Café Con Leche, contact: { phone: 228-555-0149, email: cafeconleche@example.com, location: [-73.92502, 40.8279556] }, stars:3, categories: [Bakery, Coffee, Pastries] }" | mongo PROFILE db.restaurants.insert()

Provide the lastest 10 items of the Logpresso inventory table into the MongoDB inventory collection.

table limit=10 inventory | mongo PROFILE db.inventory.insert()

Search the restaurants collection based on the name field and modify the stars value.

json "{}" | eval name="Café Con Leche", stars=4 | mongo PROFILE db.restaurants.updateOne("name", "stars")

Search 1 document with an ObjectId of 5982df1b7098262f64d4ffaf and then delete it.

json "{}" | eval \_id = fromhex("5982df1b7098262f64d4ffaf") | mongo stream2 db.restaurants.deleteOne("\_id")