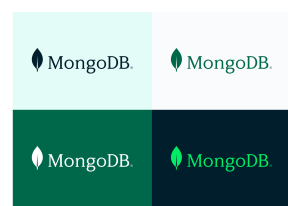
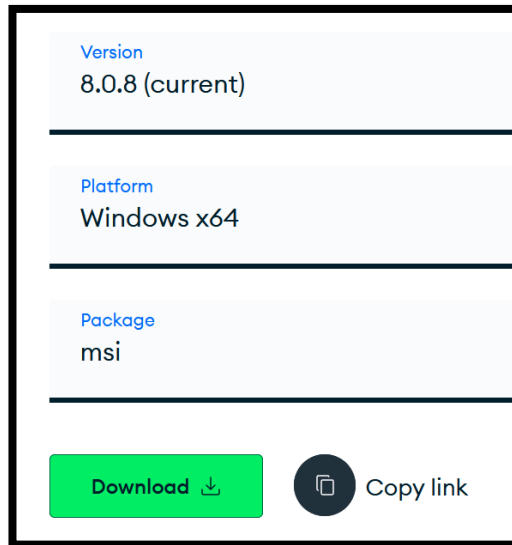


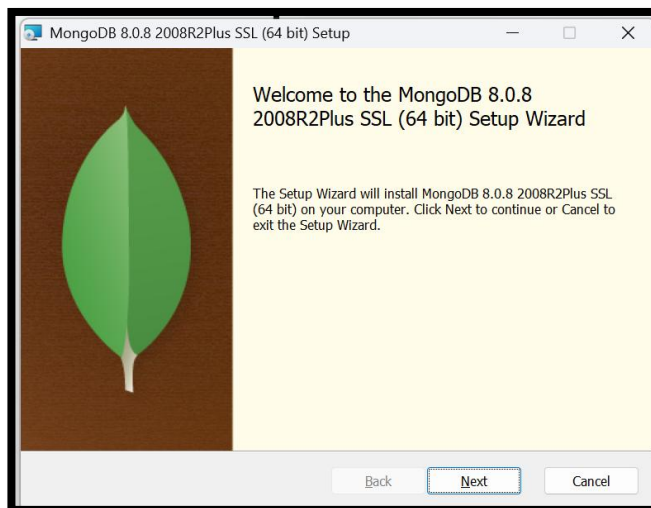
MongoDB Documentation



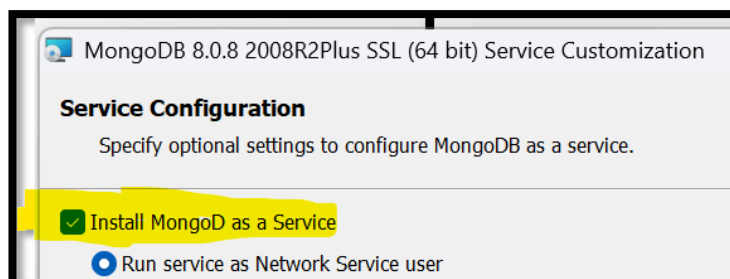
- Download the MongoDB Community Edition:
<https://www.mongodb.com/try/download/community>



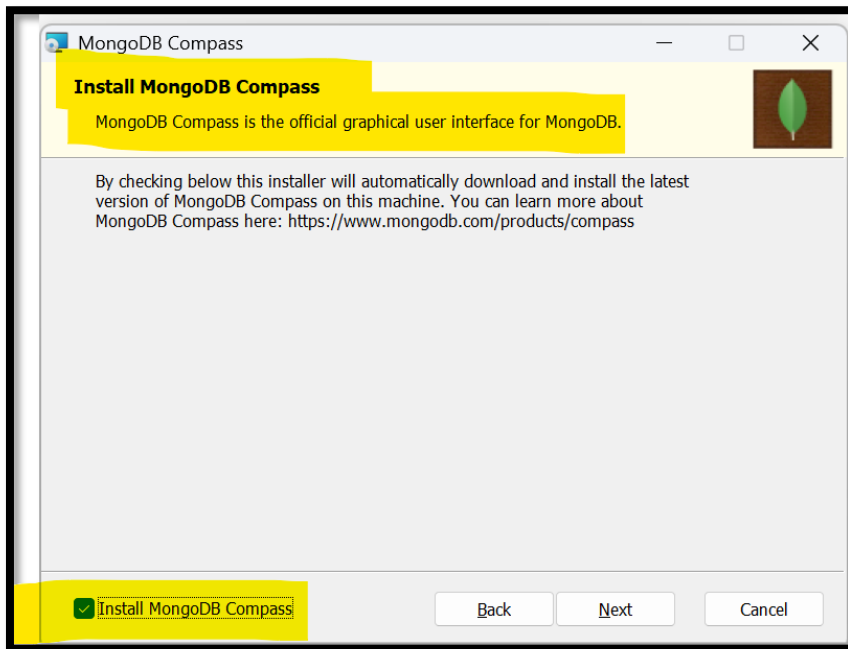
- Double click on the .msi file to install MongoDB:



- Install Complete version and Install MongoDB as a service:

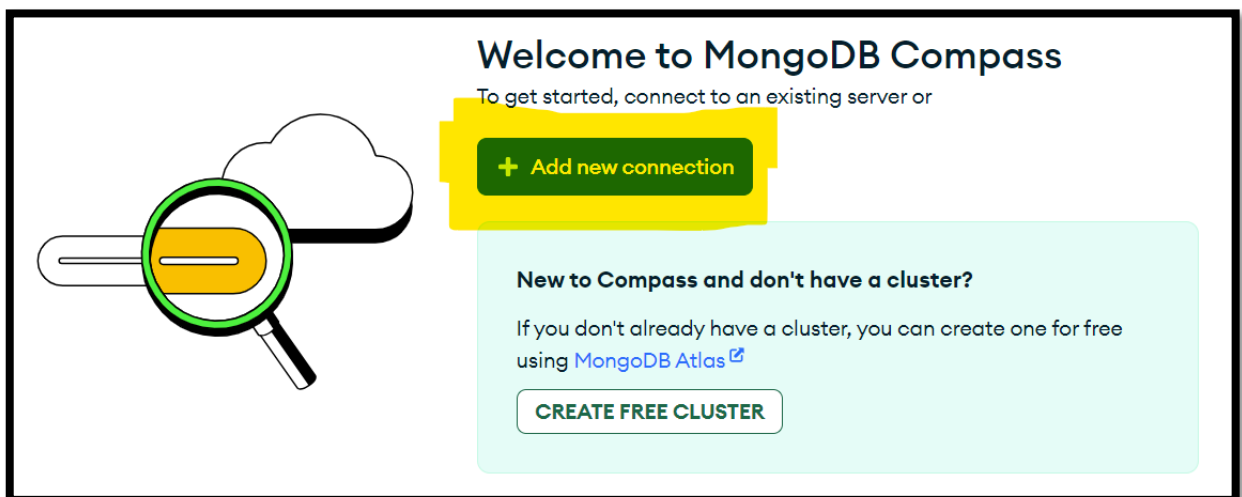


- Be sure to install “Compass” the MongoDB graphical user interface:



- Once MongoDB is finished installing, download a sample dataset:
<https://mavenanalytics.io/data-playground>

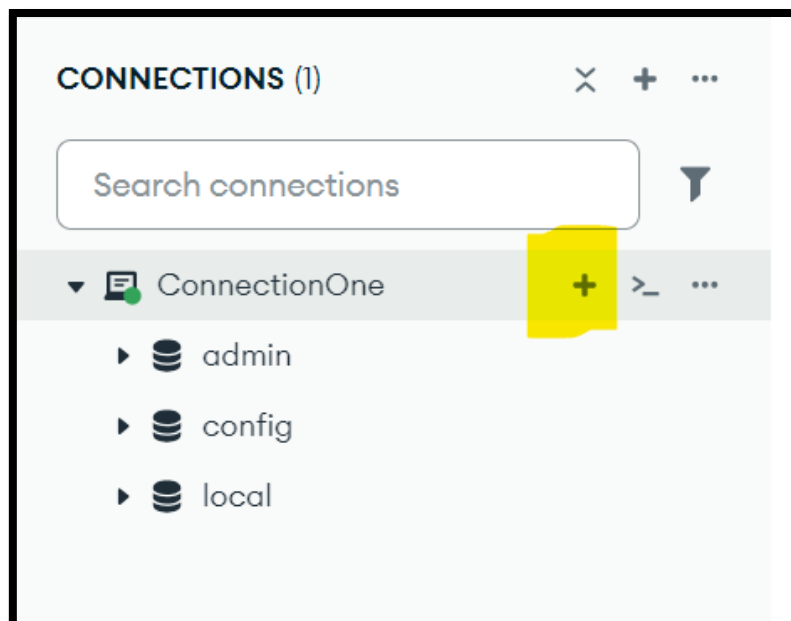
- When MongoDB has started, click on “Add new connection”:



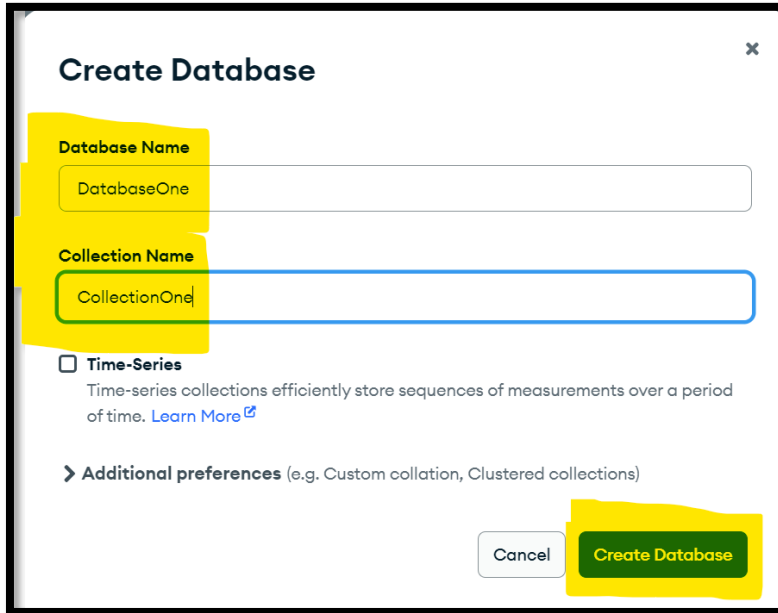
- In the New Connection box, under Name, type “ConnectionOne”, then click on “Save & Connect”:

The screenshot shows the 'New Connection' dialog box. The 'URI' field contains 'mongodb://localhost:27017/'. The 'Name' field is highlighted in yellow and contains 'ConnectionOne'. The 'Color' dropdown is set to 'No Color'. There are two informational boxes on the right: 'How do I find my connection string in Atlas?' and 'How do I format my connection string?'. At the bottom, there are three buttons: 'Cancel', 'Save', and 'Save & Connect'. The 'Save & Connect' button is highlighted in yellow.

- To the right of “ConnectionOne” click on the “+” symbol:



- In the Create Database window, type in “DatabaseOne” and “CollectionOne” then click on “Create Database”:



Create Database

Database Name
DatabaseOne

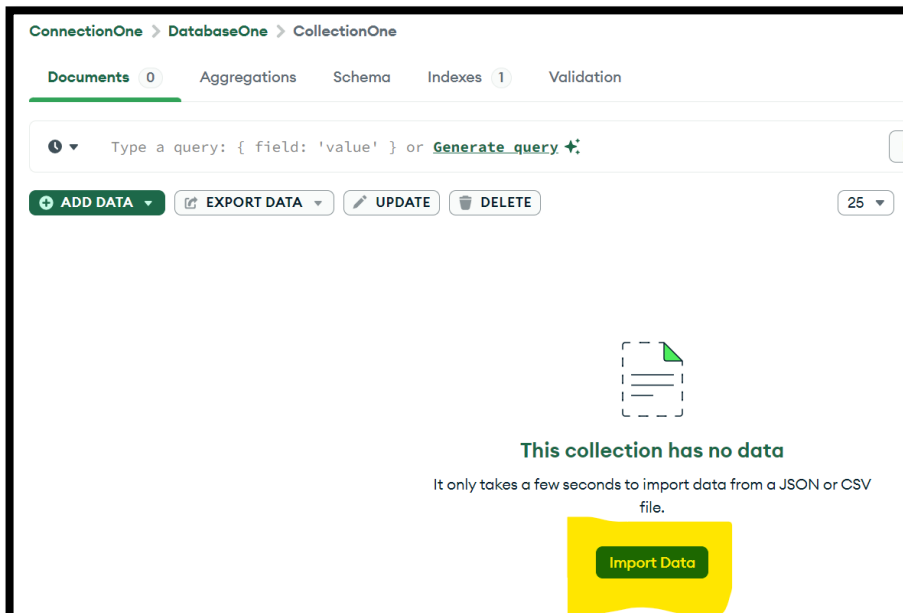
Collection Name
CollectionOne

☐ **Time-Series**
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> **Additional preferences** (e.g. Custom collation, Clustered collections)

Cancel Create Database

- To start the process of adding data, click on “Import Data”:




ConnectionOne > DatabaseOne > CollectionOne

Documents 0 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

ADD DATA EXPORT DATA UPDATE DELETE 25

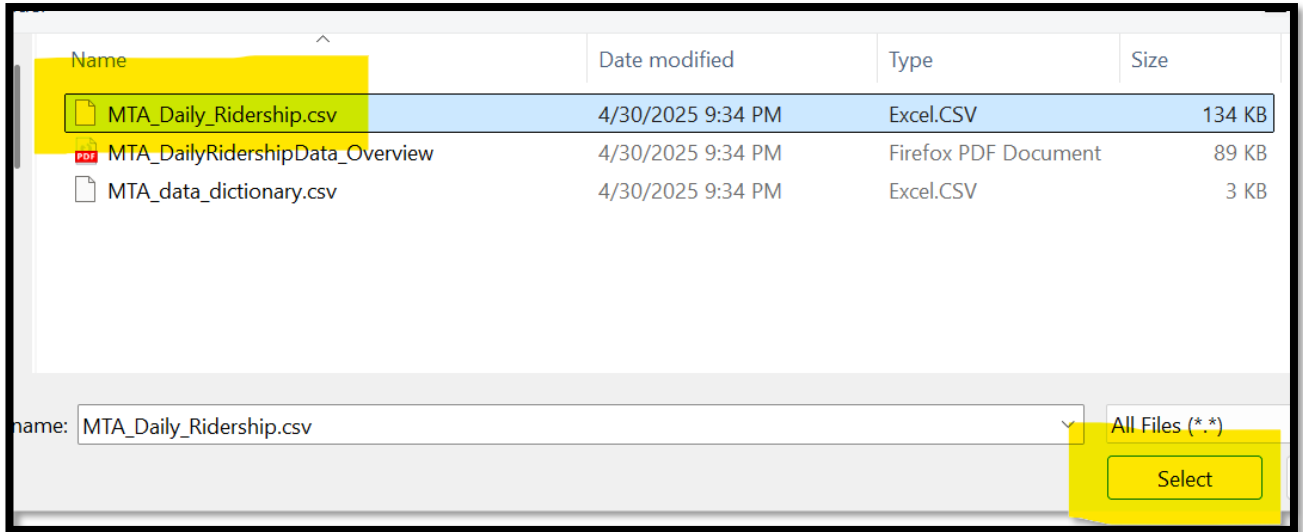


This collection has no data

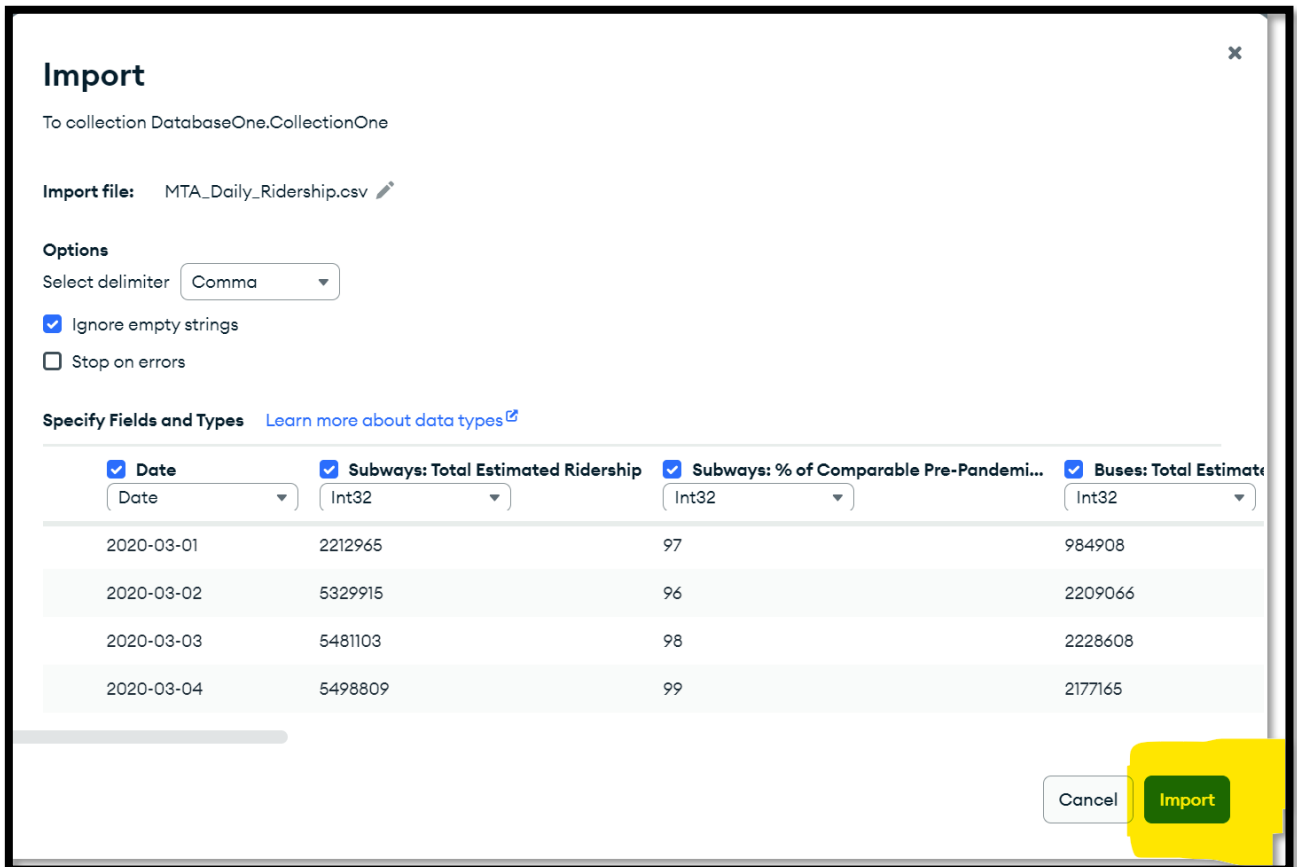
It only takes a few seconds to import data from a JSON or CSV file.

Import Data

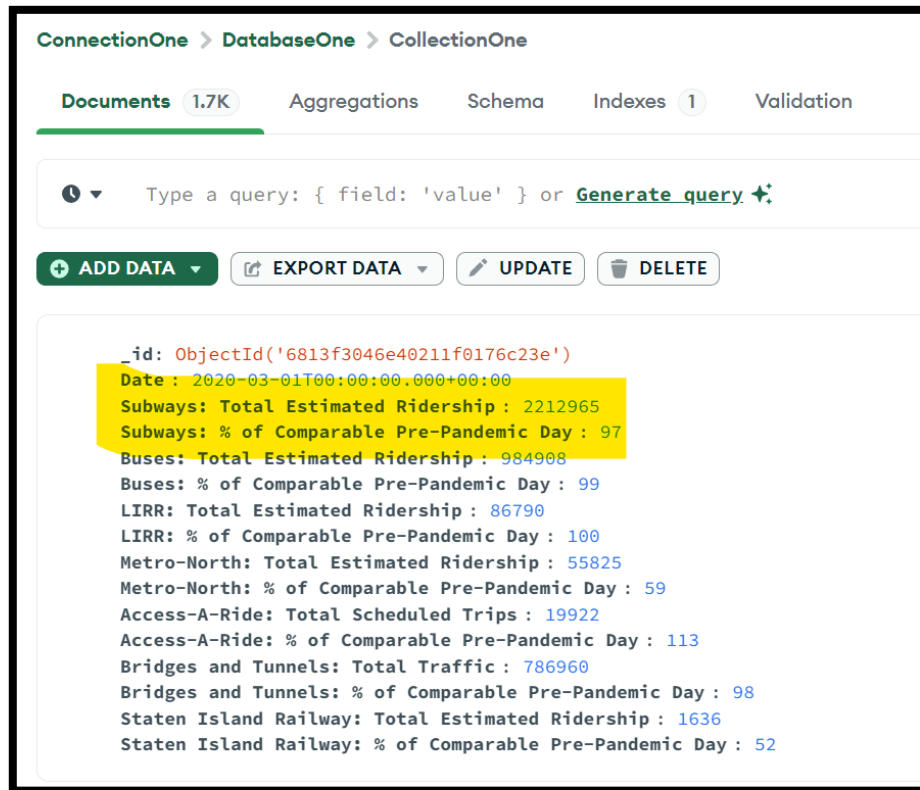
- Select CSV file:



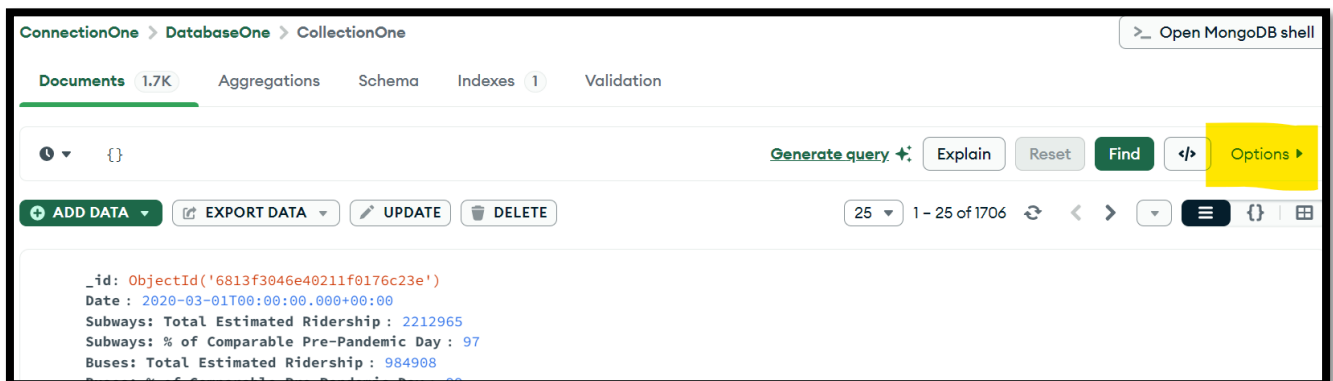
- Click on the “Import” button:



- After the import, the data will be available as a key-value pair:

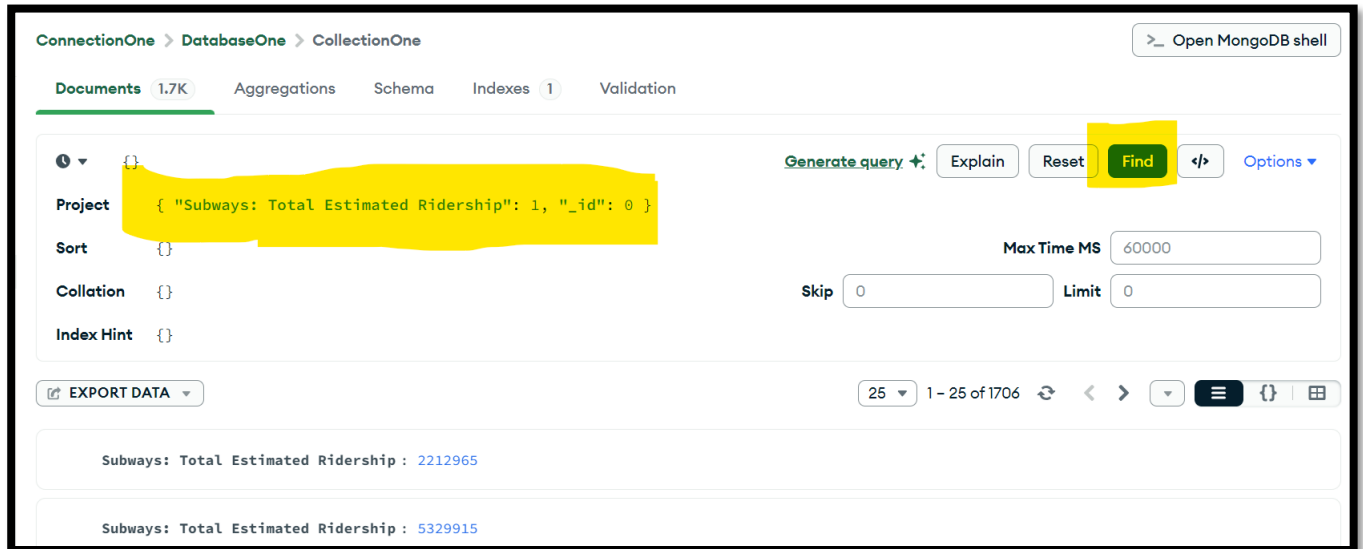


- To begin working with the data, click on “Options” on the far right hand side:

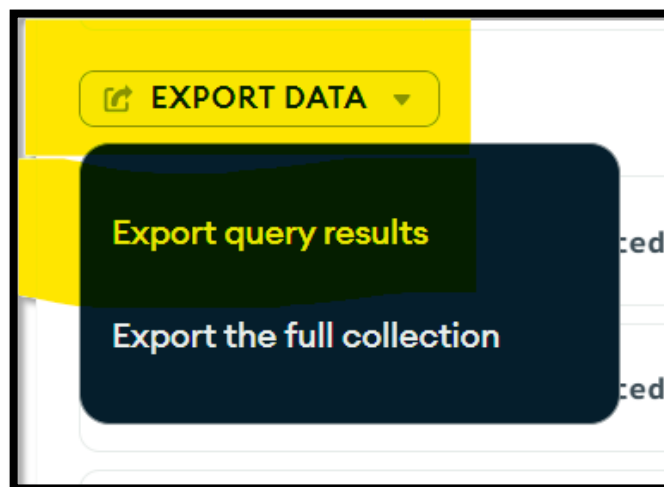


- To bring up a specific field [in this case subway ridership], type the following in the Project box and click “Find”:

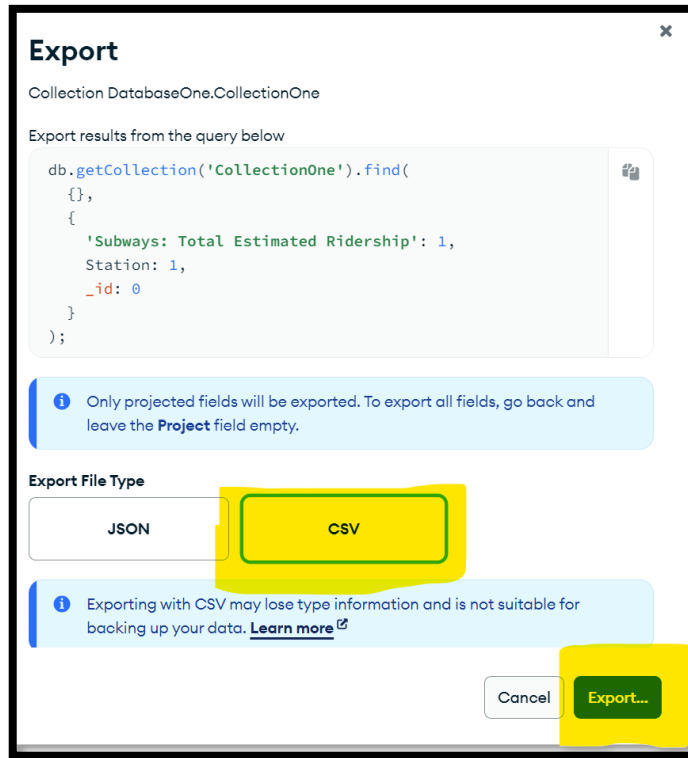
{ "Subways: Total Estimated Ridership": 1, "_id": 0 }



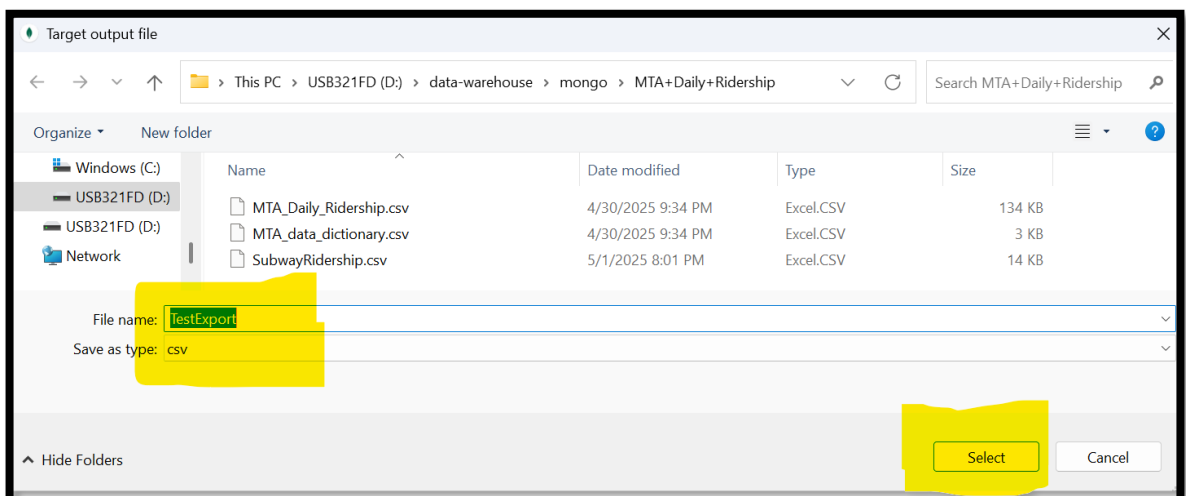
- To export the data, click on “Export Data”, then “Export query results”:



- Select “CSV”, then “Export”:



- Type in the name of the file and “Select”:



- In order to do a query to find subway ridership fields of over 5,000,000, type the following in the “Query” field above the Project field:

```
{ "Subways: Total Estimated Ridership": { $gt: 5000000 } }
```

- Be sure to type this in the “Project” field:

```
{ "Subways: Total Estimated Ridership": 1, "_id": 0 }
```

- Then click “Find”:

The screenshot shows the MongoDB Compass interface. The breadcrumb navigation at the top reads "ConnectionOne > DatabaseOne > CollectionOne". The "Documents" tab is selected, showing 1.7K documents. The "Query" field contains the query: `{ "Subways: Total Estimated Ridership": { $gt: 5000000 } }`. The "Project" field contains the projection: `{ "Subways: Total Estimated Ridership": 1, "_id": 0 }`. The "Find" button is highlighted in green. Below the query fields, the "Max Time MS" is set to 60000, "Skip" is 0, and "Limit" is 0. The "EXPORT DATA" button is visible. The results section shows a list of documents, with the first five highlighted in yellow. A blue arrow points from the "Query field" label to the query field. Another blue arrow points from the "Project field" label to the project field. A third blue arrow points from the "Results" label to the results list.

Query field

Project field

Results

Subways: Total Estimated Ridership
5329915
5481103
5498809
5496453
5189447

- The logic of the MongoDB is that first the Query retrieves Documents [also known as Rows] that have Subway Ridership above 5 million:

```
{ "Subways: Total Estimated Ridership": { $gt: 5000000 } }
```

- Then the Project retrieves only the Subway Ridership fields:

```
{ "Subways: Total Estimated Ridership": 1, "_id": 0 }
```

- Here we see 5 Documents [Rows] that have Subway Ridership over 5 million and the Subway Ridership fields:



A screenshot of a MongoDB query result. It shows a table with 5 rows. Each row contains a single field: "Subways: Total Estimated Ridership". The values for each row are 5329915, 5481103, 5498809, 5496453, and 5189447. The values are displayed in blue text on a light gray background.

Subways: Total Estimated Ridership : 5329915
Subways: Total Estimated Ridership : 5481103
Subways: Total Estimated Ridership : 5498809
Subways: Total Estimated Ridership : 5496453
Subways: Total Estimated Ridership : 5189447

- This separation between the Query and Field Selection increases efficiency and makes the system more flexible. This allows for only fields that are needed to be retrieved, thereby optimizing Read operations.

- MongoDB's logic operators:
 - * \$gt is "greater than"
 - * \$lt is "less than"
 - * \$eq is "equal to"
 - * \$in is "included in" which will locate values that match in a specified array

