Week 2 :Lab

Prepared by Dr Nadeem Qazi.

Important Note.

At the end of the lab manual, you are required to complete a short task. Once completed, write a reflection summarizing what you have learned. This reflection will form part of your portfolio.

What is MongoDB

MongoDB is an open-source database that uses a document-oriented data model. MongoDB is built on an architecture of collections. It is very useful when there is no specific database **structure** like RDBMS and varies data (i.e. columns) as per requirements. Terminologies used in MongoDB are Collection, Documents, Fields, Schema, and Models.

1. **Collections** in Mongo are equivalent to tables in relational databases. They can hold multiple JSON documents.

2. **Documents** are equivalent to records or rows of data in SQL. While a SQL row can reference data in other tables, Mongo documents usually combine that in a document.

3. **Fields** or attributes are similar to columns in a SQL table. However these are case sensitive ,so Age and age are two separate fields in the MongoDB collection.

4. **Schema**: A Mongoose 'schema' is a document data structure (or shape of the document) that is enforced via the application layer.

5. **Models:** are higher-order constructors that take a schema and create an instance of a document equivalent to records in a relational database.

Week 2 Lab tutorial has the following tasks:

1. You are provided with a CSV file named Peoples.csv. Your task is to download the file and perform the following MongoDB operations using MongoDB Compass.

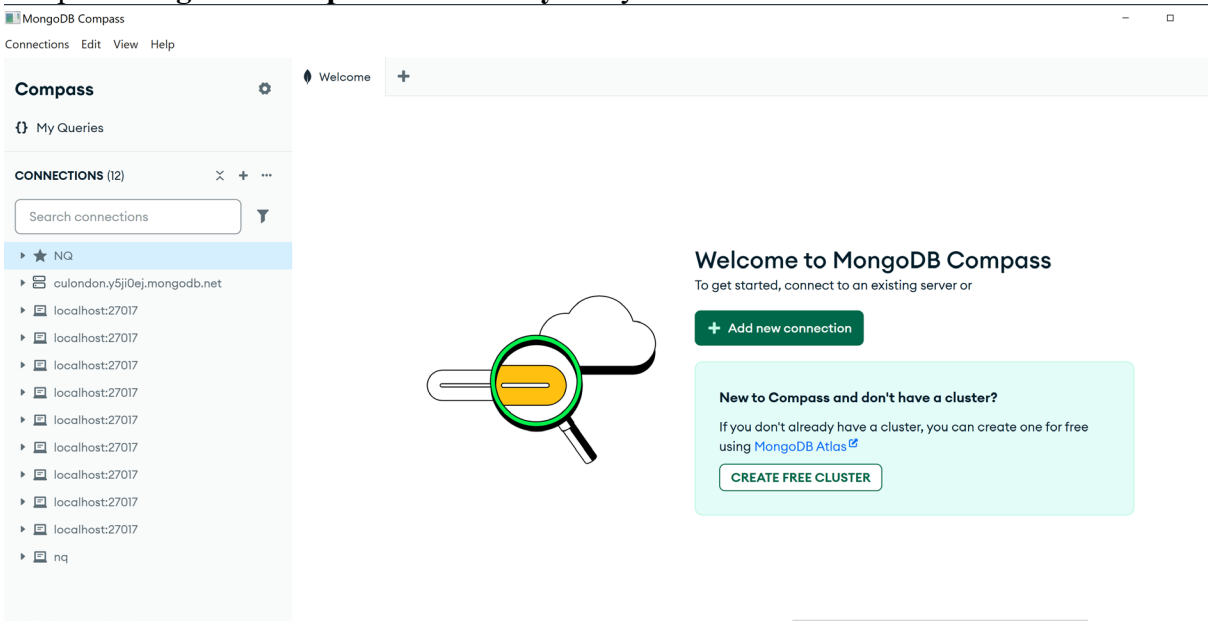
    a. **Inserting a document**
    b. **Updating a document**
    c. **Deleting a document**
    d. **Aggerate pipe line**

Managing Data With Compass (CRUD)

Creating a Database & Initial Collection

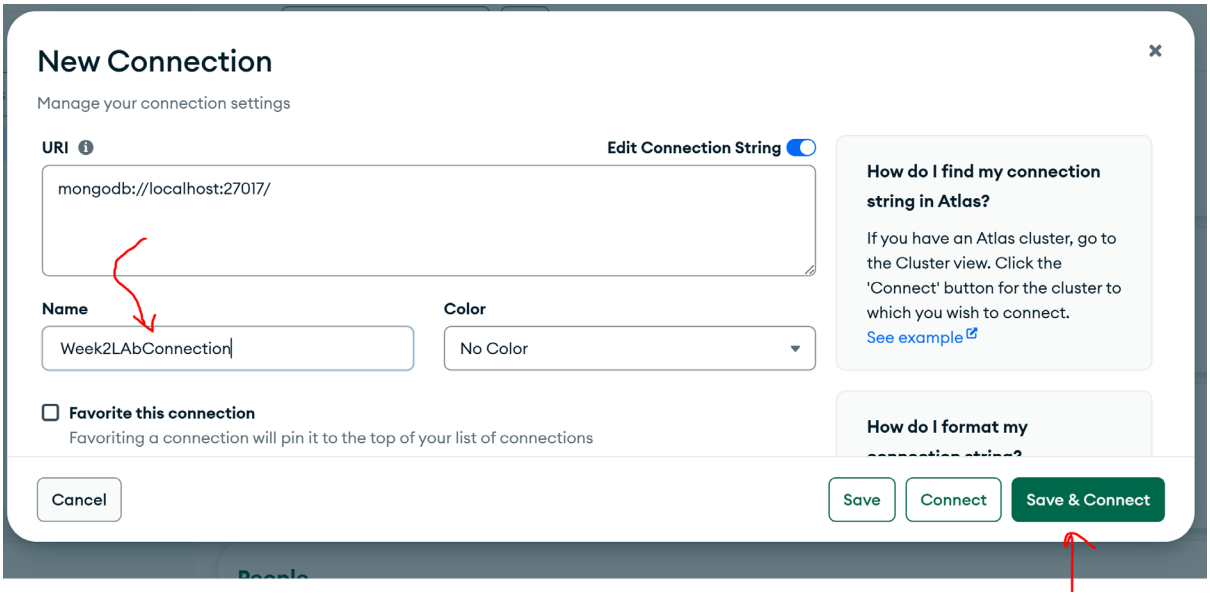
Step 1

- Open **MongoDB Compass Community** that you installed earlier.



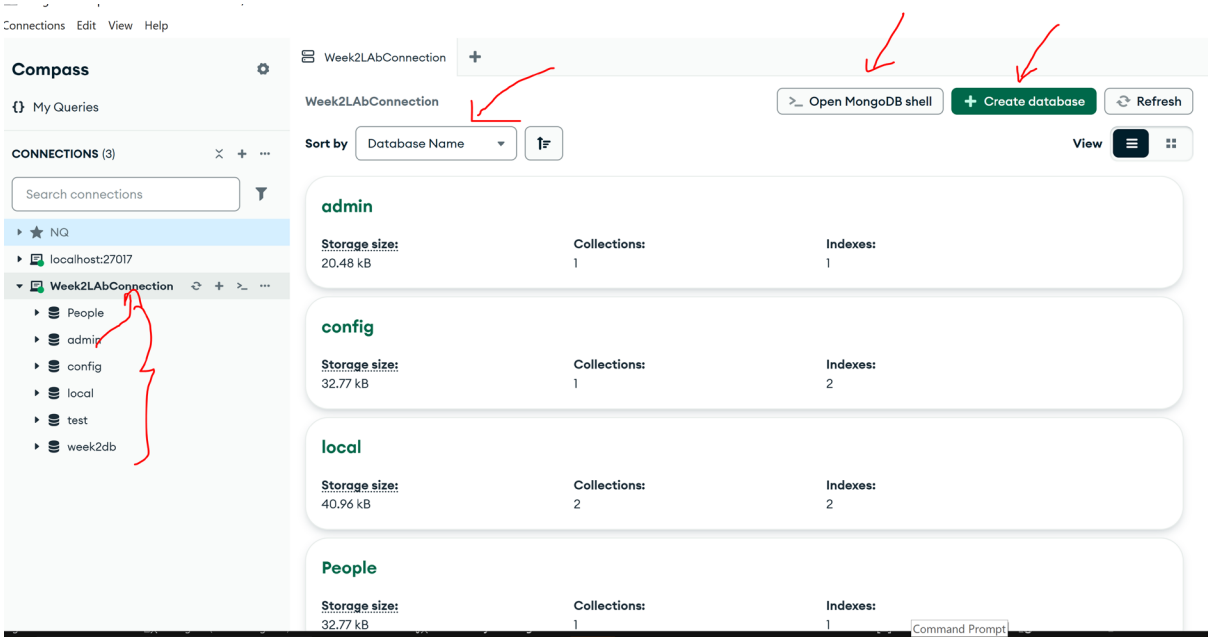
Click Add new connection

- Connect to the MongoDB server by entering the following parameters:
  - **Hostname**: `localhost`
  - **Port**: `27017`

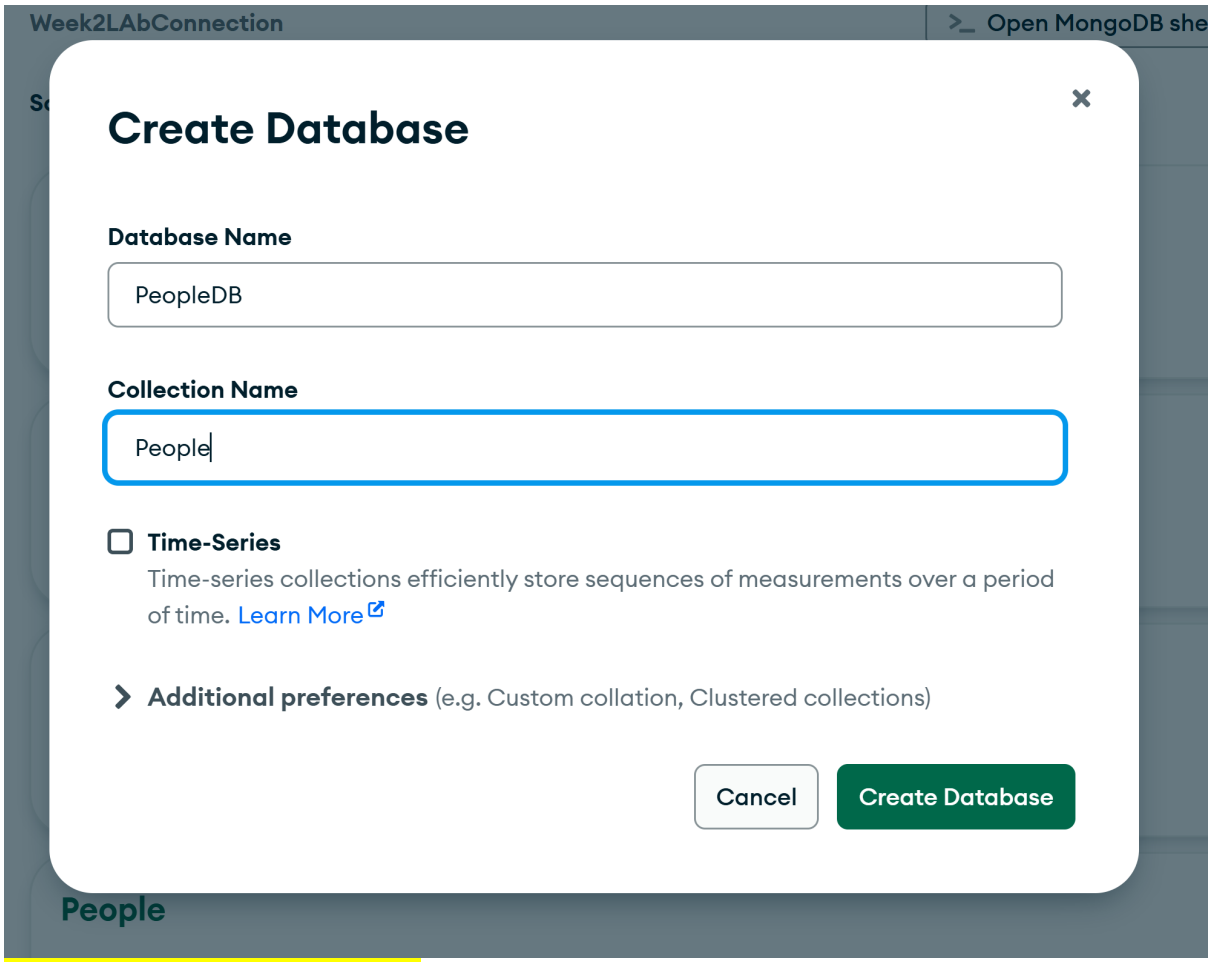


Once your click Save& connect the connection is saved and will be shown on the left :

You will all the databases and collection in your connection



- Now click Create database button



write PeopleDB in Database Name

people in the collection Name

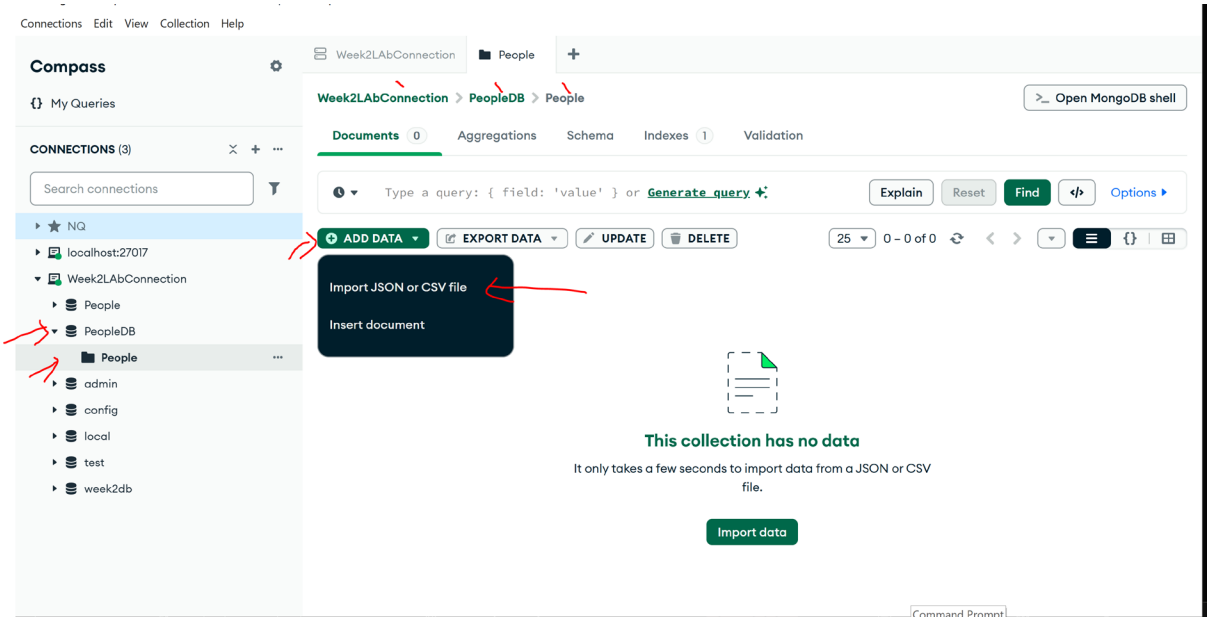
It will create a Database PeopleDB with a collection named people.

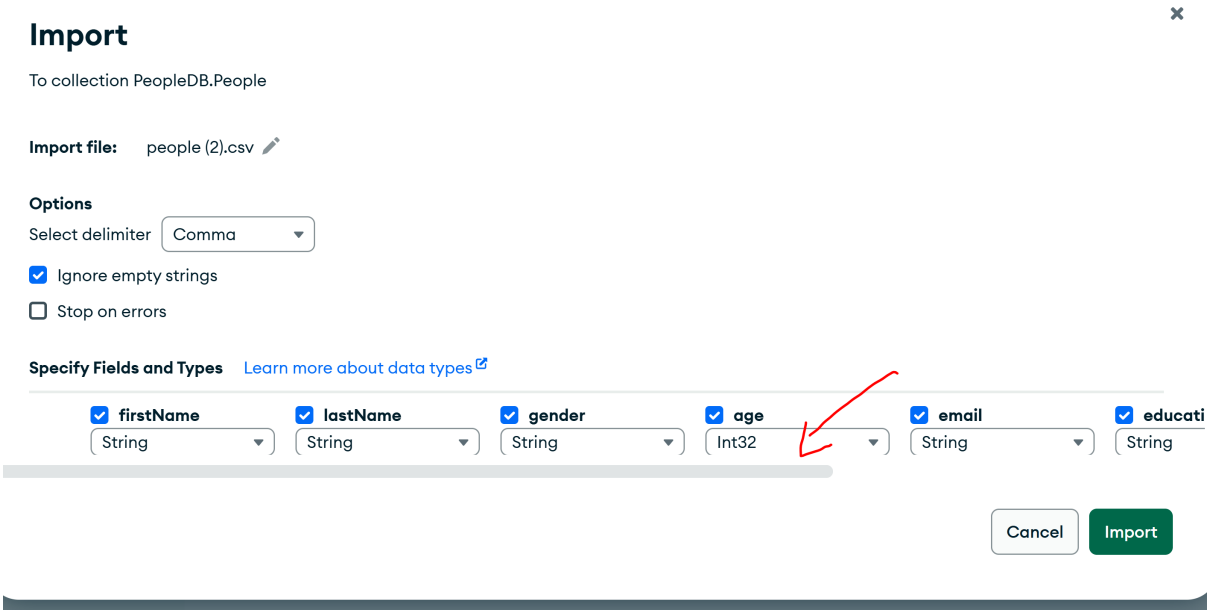
Adding Data to the collection either through

1. Importing a CSV File or
2. Insert Document

Adding Data through Importing a CSV File

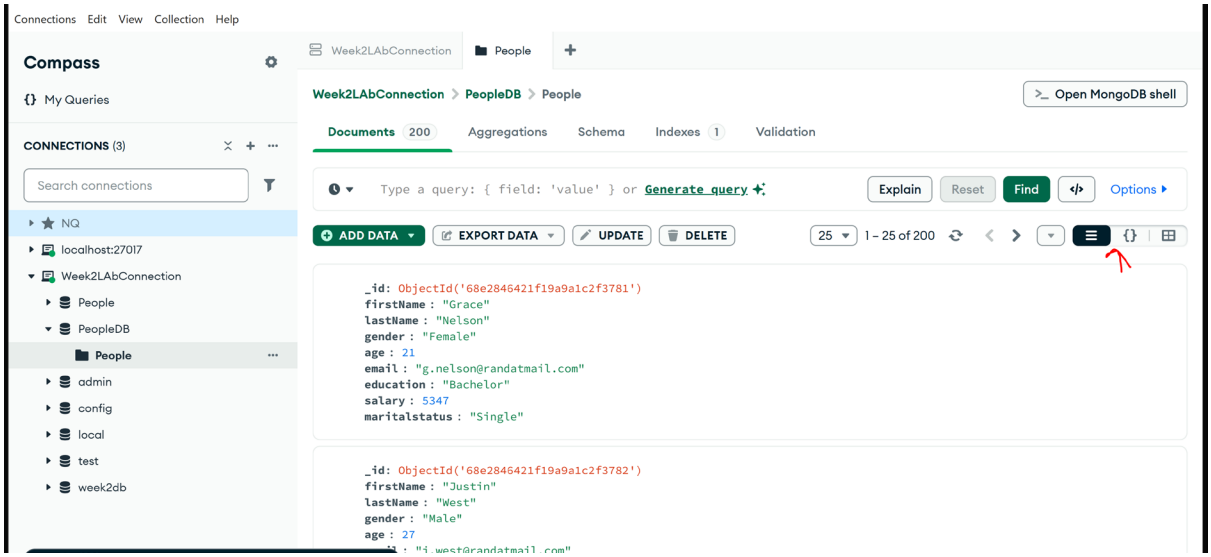
On the left-hand panel click PeopleDb and click people collection then click **ADD DATA**, and then click Import JSON or CSv file . following screen will be shown.





This will import the data in the people.csv file to the people collection of your **peopledb** database. Note the number of documents.

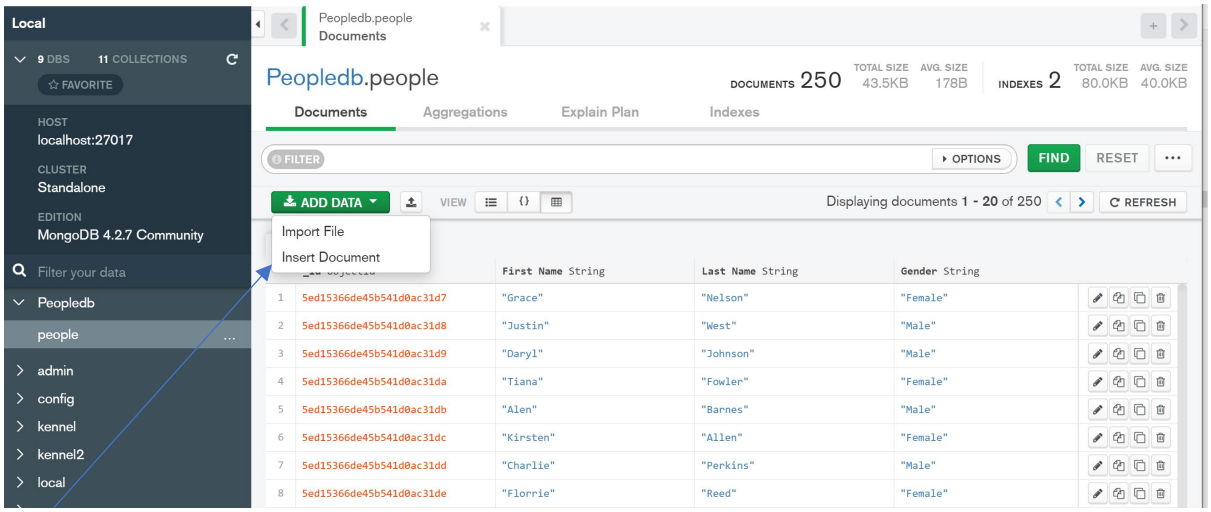
Once the data is imported it's time to perform some basic queries in MongoDB . Select Document Tab



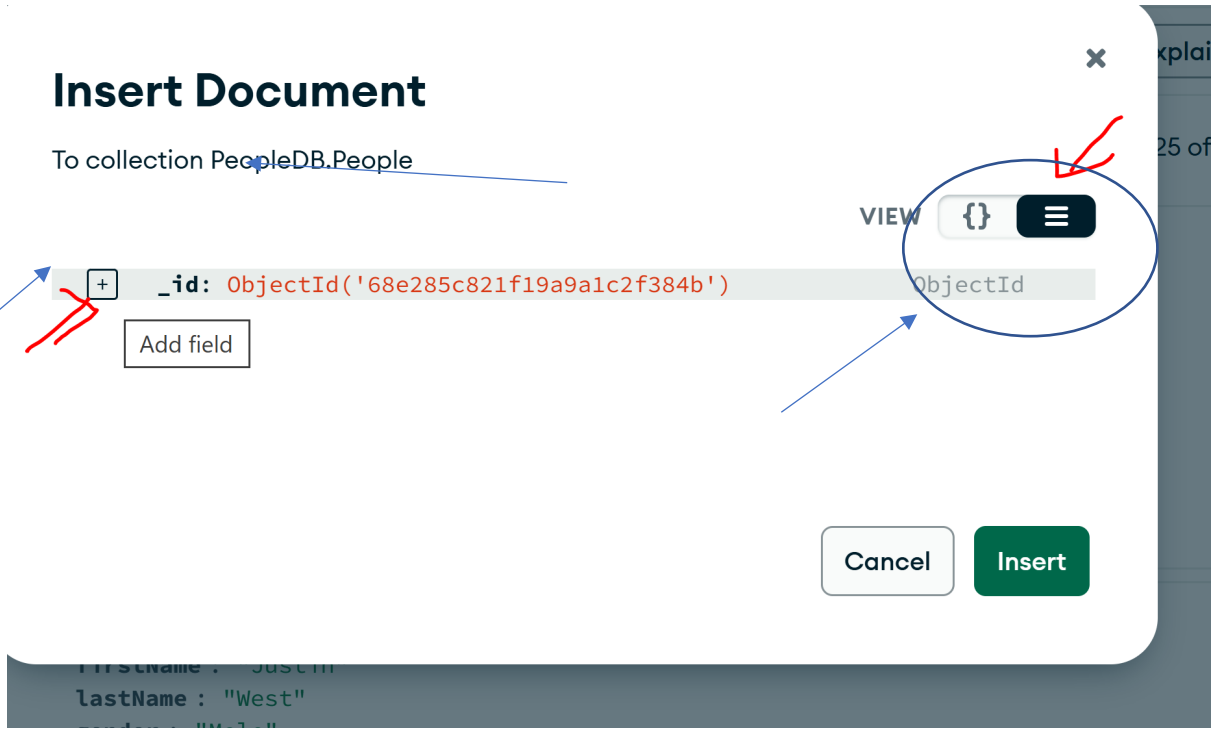
Next to View, you will observe three view settings list, JSON and table view. Click each of them to observe how data can be displayed in the collection. Note, , for each document there is a _id field associate DO NOT change it.

Inserting a document in the collection

Now is the time to insert a new document to the people collection. To do this click Add data but this time select Insert document

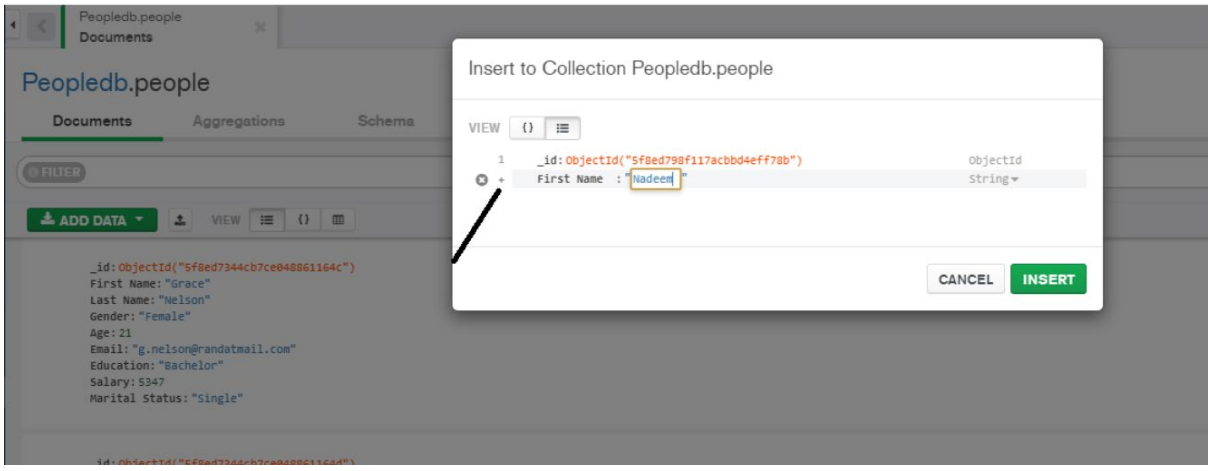


A new window will be displayed , select the list view as shown in the figure below. Note there is a default _id field created for you do not change it



For Inserting a new record:

click the + sign , it will add one field in the document for you to give a name and value in that field.



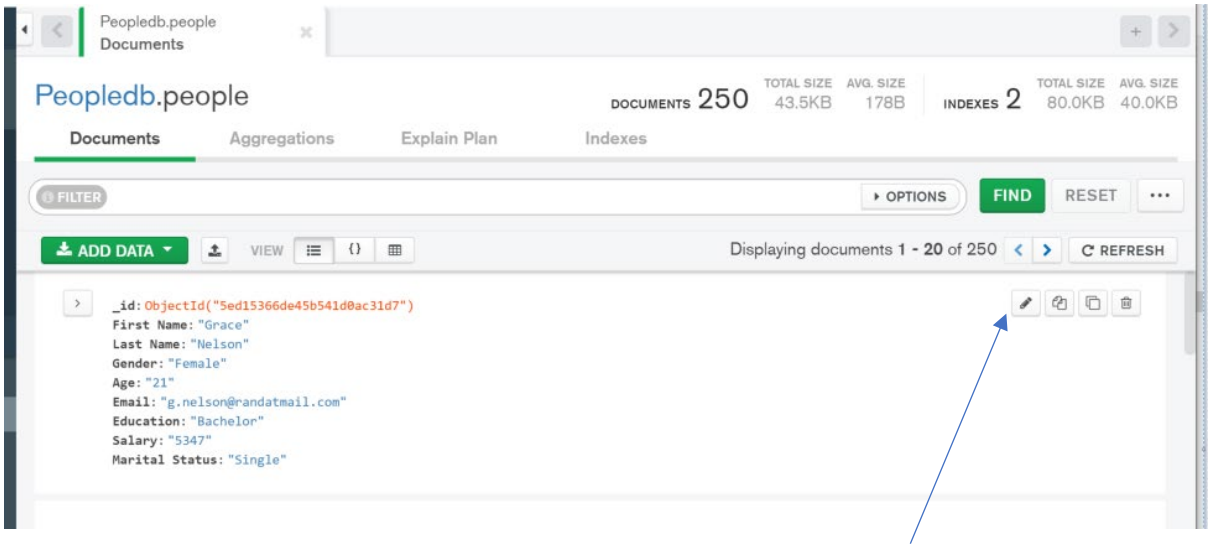
Add more fields to this document using + sign.

Remember for each of this fields you to click + sign on the left and select the type from the right.

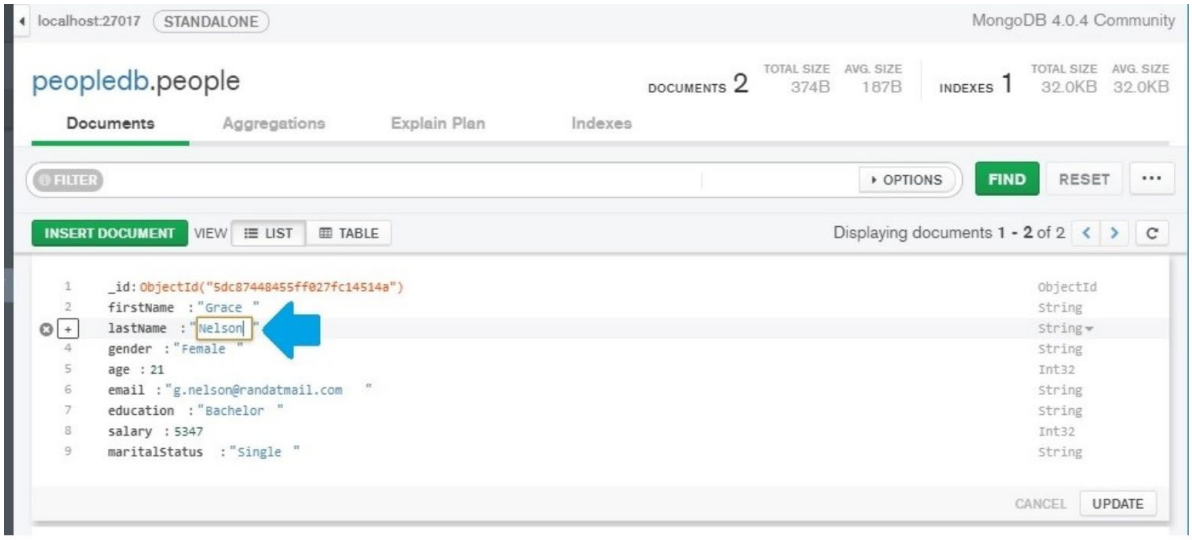
Once you filled all the fields from 2 to 9 then click **Insert** to insert the document in the collection.

Updating the document

1. Locate the document you want to update and hover over it until the tooltip pops up.
2. Click the edit document Pencil icon as shown in Figure This will enable the full modification of the document within the viewer.



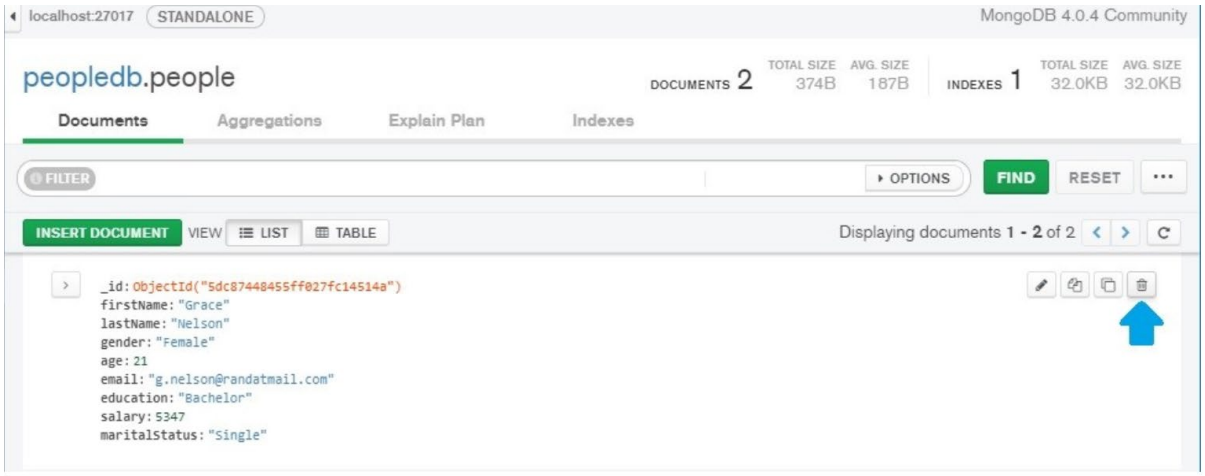
3. Double click the value field of the data you want to update, as shown in the Figure below
4. Change the data to your requirement and press enter.
5. You will notice a Document Modified notification will appear, click the update button on this notification to save changes. See Figure 17.



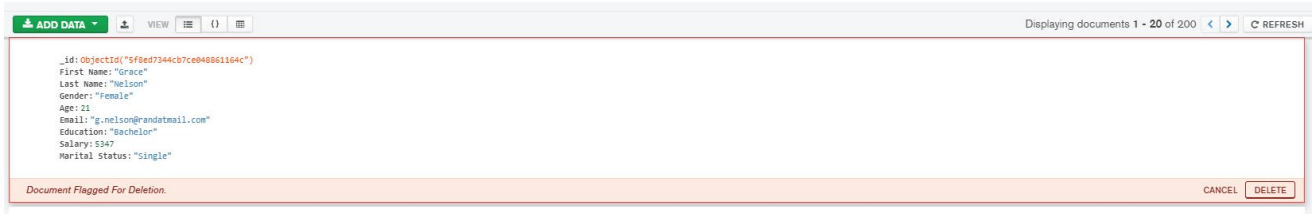
Deleting the documents

Deleting documents is another simple task with the MongoDB Compass software and can be done with two clicks of a button. Please follow the next steps to delete a document.

1. Locate the document you want to update and hover over it until the tooltip pops up.
2. Click the delete Trash Can icon as shown in the Figure below:



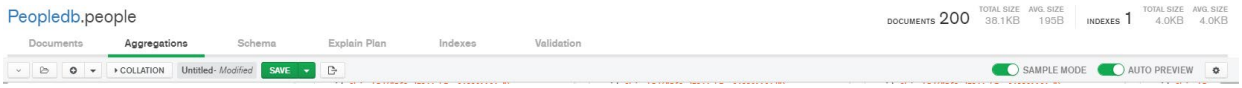
1. A notification will appear stating the document has been flagged for deletion in red, Click the delete button as shown in figure the document will be deleted.



Aggregations:

Aggregation operations process data records and return computed results. Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides three ways to perform aggregation: the aggregation pipeline, the map-reduce function, and single purpose aggregation methods. In this tutorial we will discuss the aggregation pipeline only.

You can find the aggregation tab just adjacent to the document tab as shown below:



Aggregation:

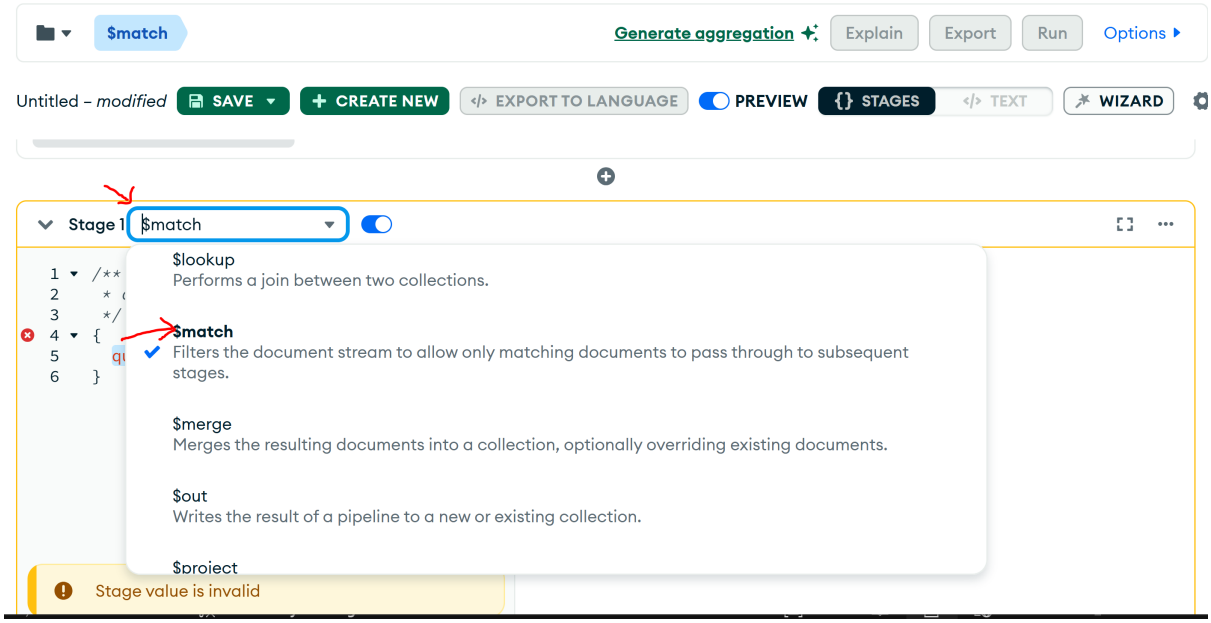
Aggregation is performed in multiple stages to produce aggregated results from the documents. Aggregation pipelines are a composition of various stages that transform and filter the data. For example suppose we in our collection people we want to see:

- How many people are there who has got the bachelor degree and are older than 21 years of age.
- what is average age of Female and male in this group
- what is the age salary of the Female and male in this group?
- What is max age of male and female in this group
- What is the min age of male and female in this group?

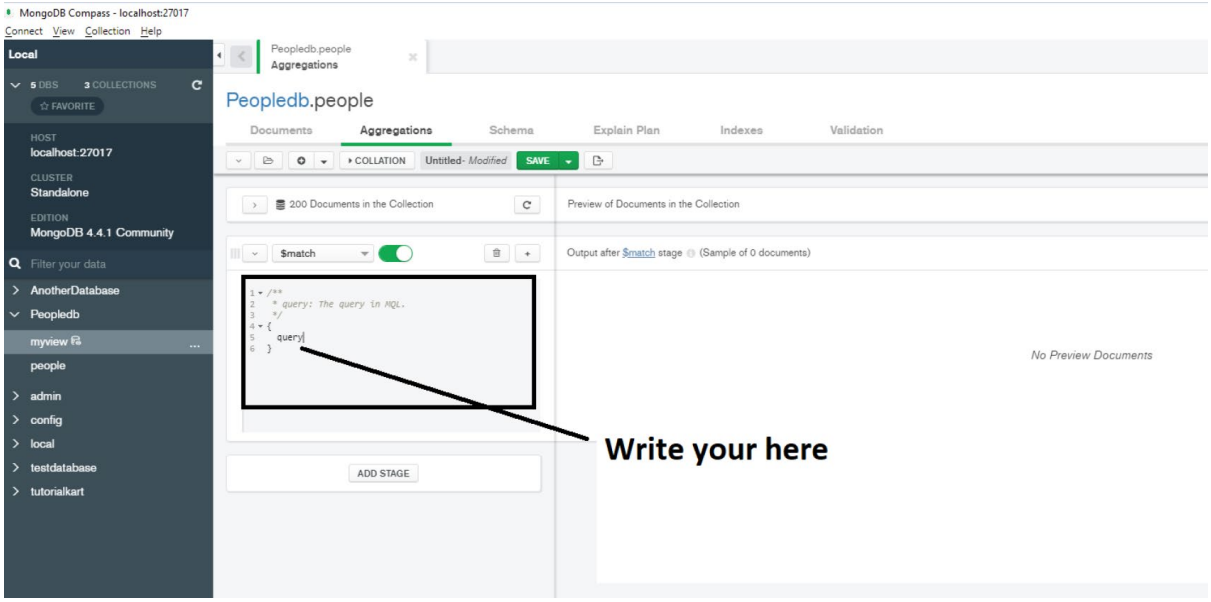
This will be performed in two stages :

1. $match — This stage is filtering the people has bachelor degree and are older than 21 years
2. $group — This stage is grouping the people by their gender and creating the aggregate functions.

Let's start the work: we will use the Aggregation tab. So, click the aggregation tab and then click Select



and chose $match as shown in the figure



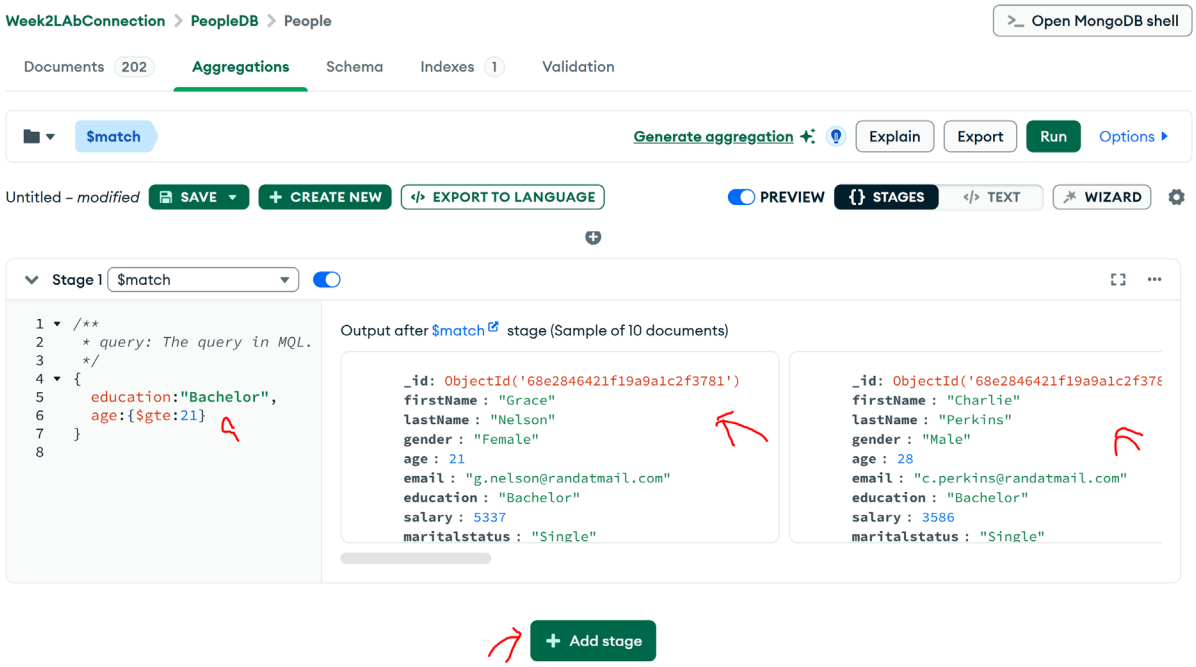
Task:
Write the following MongoDB query in JSON format as shown in the figure. This query retrieves documents where the `Education` field is `"Bachelor"` and the `Age` field is greater than `21`.

🔷 Important Notes:

- Use the exact field names as they appear in the CSV file. For example, if the field is named `"Education"` or starts with a lowercase `"e"` like `"education"`, use that exact name.
- The `$match` stage is used to filter documents that meet the specified criteria.

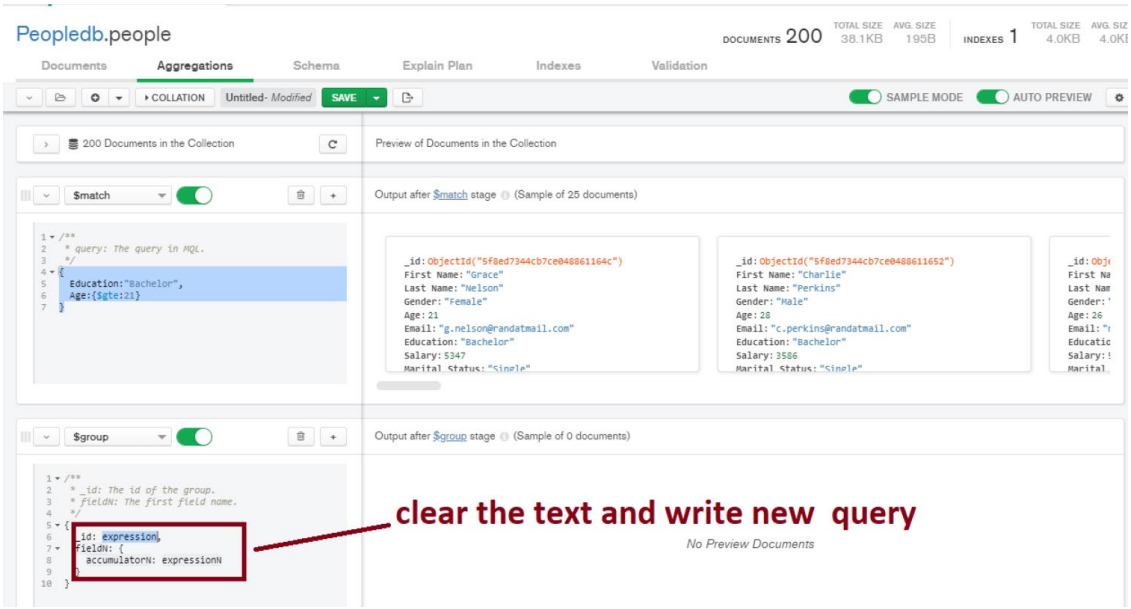
- Once you write the correct query syntax, the output will appear in the adjacent window as shown in the figure.

```
{
 education:"Bachelor",
 age:{$gte:21}
}
```



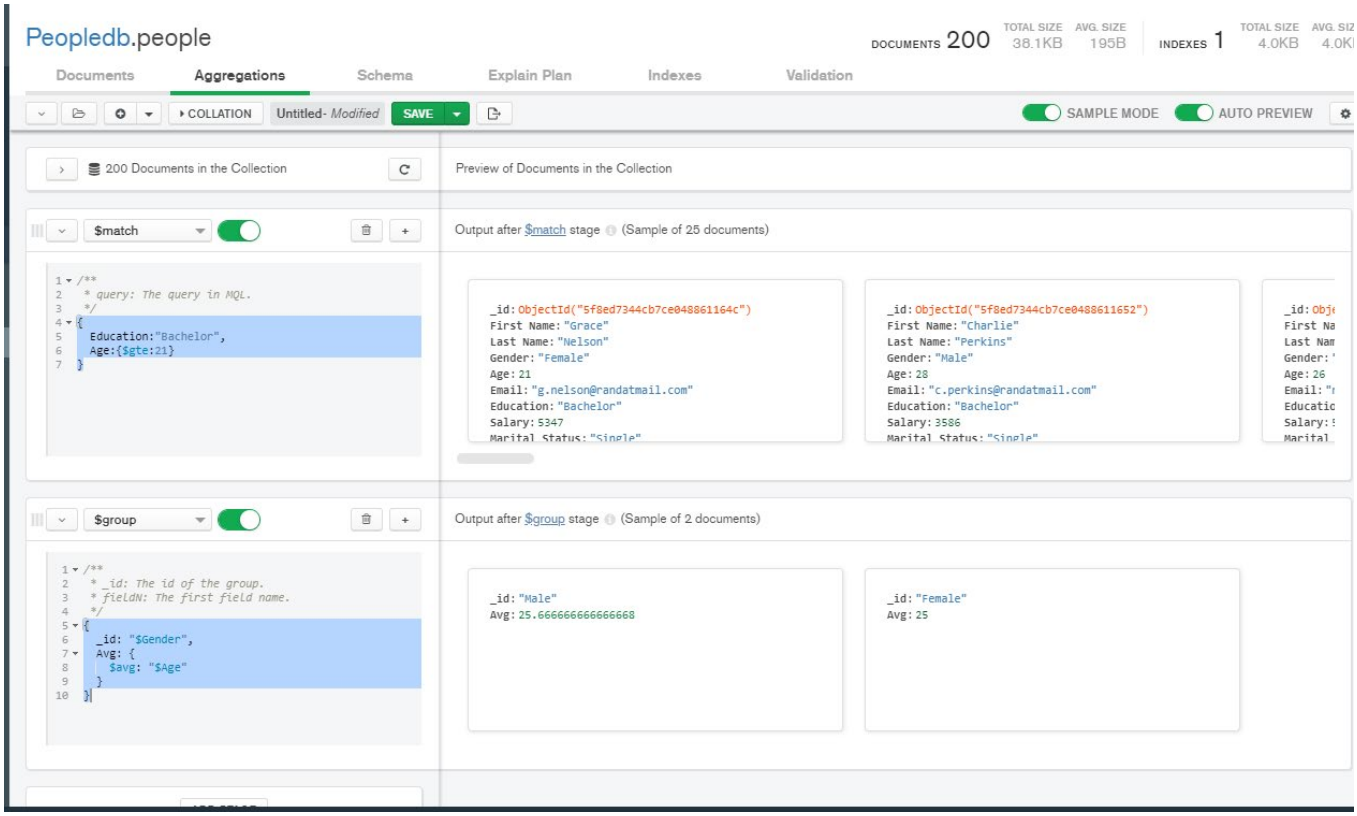
2nd Query :

- ***Now lets find out the*** what is average age of Female and male in this group. For this we will be using group aggregate.
- $group: Groups input documents by the specified _id expression and for each distinct grouping, outputs a document. The _id field of each output document contains the unique group by value. The output documents can also contain computed fields that hold the values of some accumulator expression.
- In order to do group, click add stage button and it will create another window , click select and then choose group as shown below :



Now write the following query in the space shown in red box in the above figure

```
{
 _id: "$gender",
 Avg: {
   $avg: "$age"
 }
}
```



Query Explanation: Note there is $ sign before the field Gender and it is enclosed in double quote

$avg is the aggregate function provided the mongo dB and in double quotes "$Age" is the name of the field in the documents over which this aggregate function is applied. So this query returns average age of female and male .

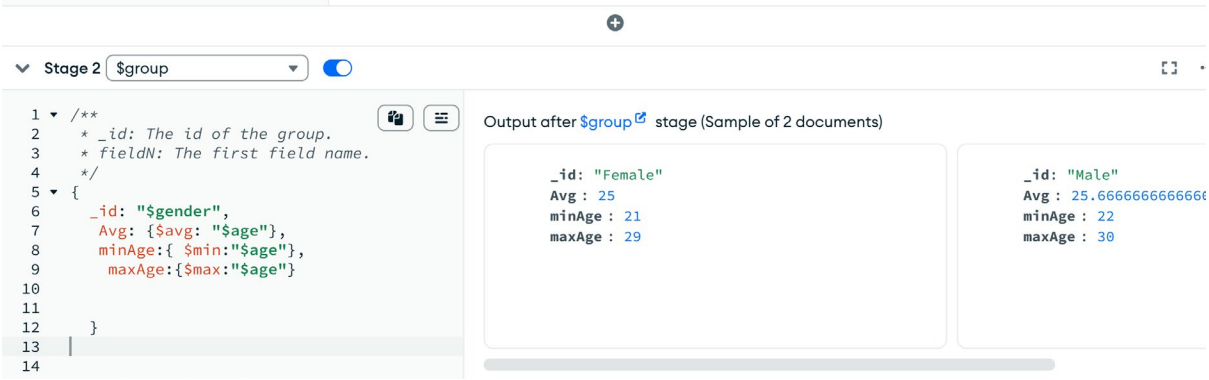
it is analogy of the SQL statement select avg(age) from people group by gender

Add another query for the max and min Age of males and females in this group, as shown in the figure below:

Query is :

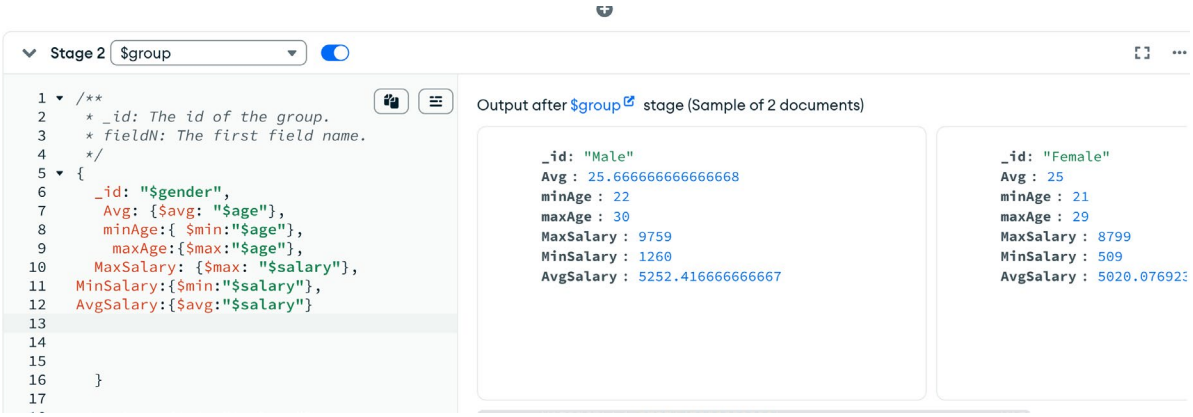
MinAge:{ $min:"$age"},
MaxAge:{$max:"$age"},

Your screen should look like this

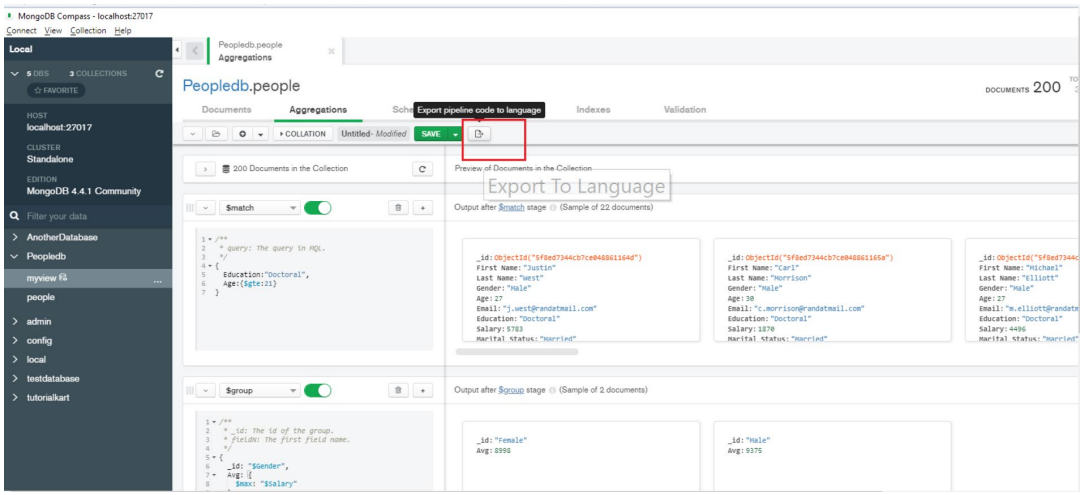


Next Query find the min, max and average salary of the male and female
Add the following query. Notice this time it is not age but salary in double quotes
MaxSalary: {$max: "$salary"},
MinSalary:{$min:"$salary"},
AvgSalary:{$avg:"$salary"}

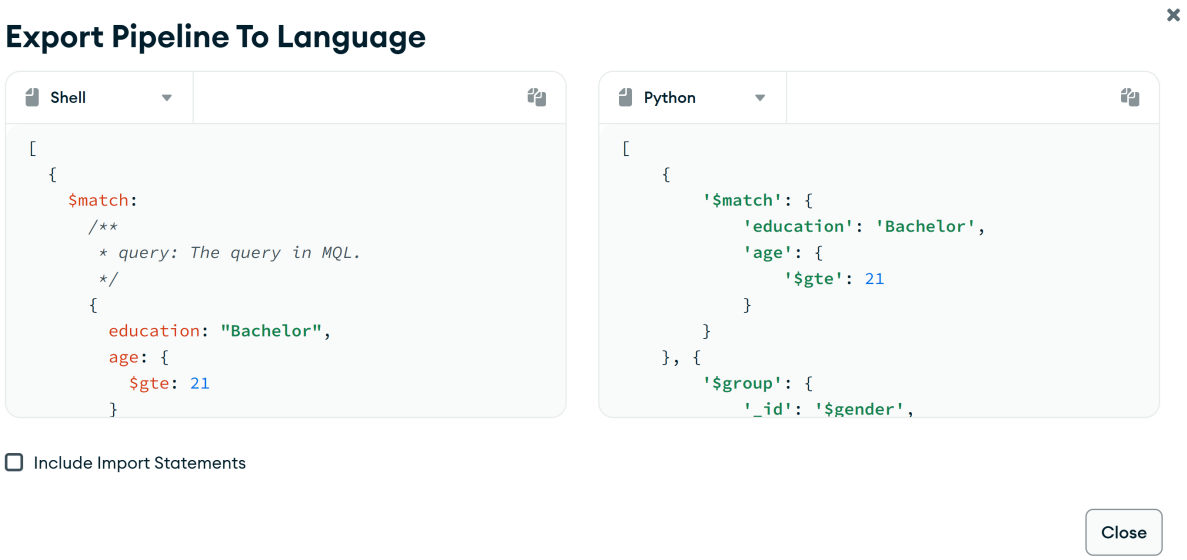
Your screen now should look like this



Lastly you can also save these queries that you have just written in to any language i.e. python or node. To do so click the -> button as shown in the figure



Then save the pipe line in Node language as shown in the figure :



Note once imported , you can use these code on mongodb shell.

Lab tasks

Write MongoDB queries for the following using either command shell:

1. **Repeat the same process to search education for Master and .Find the avg,min,max age and avg min max Salary of the people group by marital status.**
2. **find min,max average salary of each age group of female**
3. **find min,max average salary of each age group of male**
4. **Count married and unmarried females and males.**

After completing this task, write a reflective report summarizing your lab work for today. Include screenshots of the MongoDB queries you performed. Add these to your portfolio for Week 2. Inform your Lab tutor once you have finished the lab. Submit your complete portfolio, covering all weeks (Week 1 to Week 11), by the end of Week 11/