Week 2 :Lab

Important Note.

At the End of the lab manual, you have given a short task. Complete the task as mentioned and write a reflection for 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.

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 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**

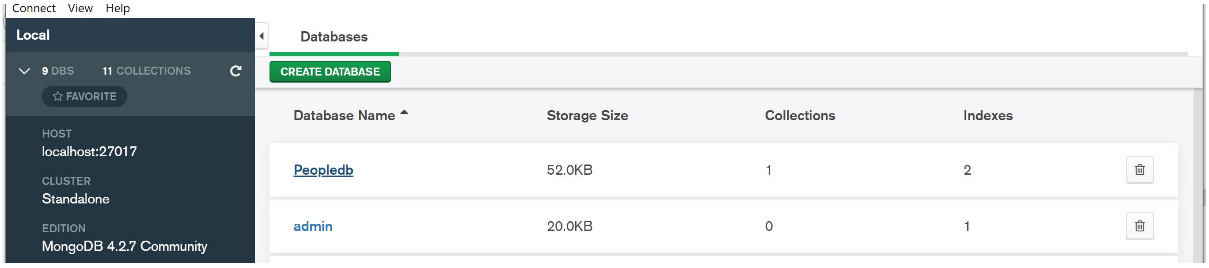
Managing Data With Compass (CRUD)

Creating a Database & Initial Collection

Step 1

- Open **MongoDB Compass Community** that you installed earlier.
- Connect to the MongoDB server by entering the following parameters:
  - **Hostname**: `localhost`
  - **Port**: `27017`
- Click **Connect**.
- From the dashboard, click the **Create Database** button.
- The **Create Database** helper will appear (as shown in Figure 7).
- Enter the following details:
  - **Database Name**: `peopledb`
  - **Collection Name**: `people`

1. • Click the **Create Database** button..



Local

9 DBS 11 COLLECTIONS

☆ FAVORITE

HOST
localhost:27017

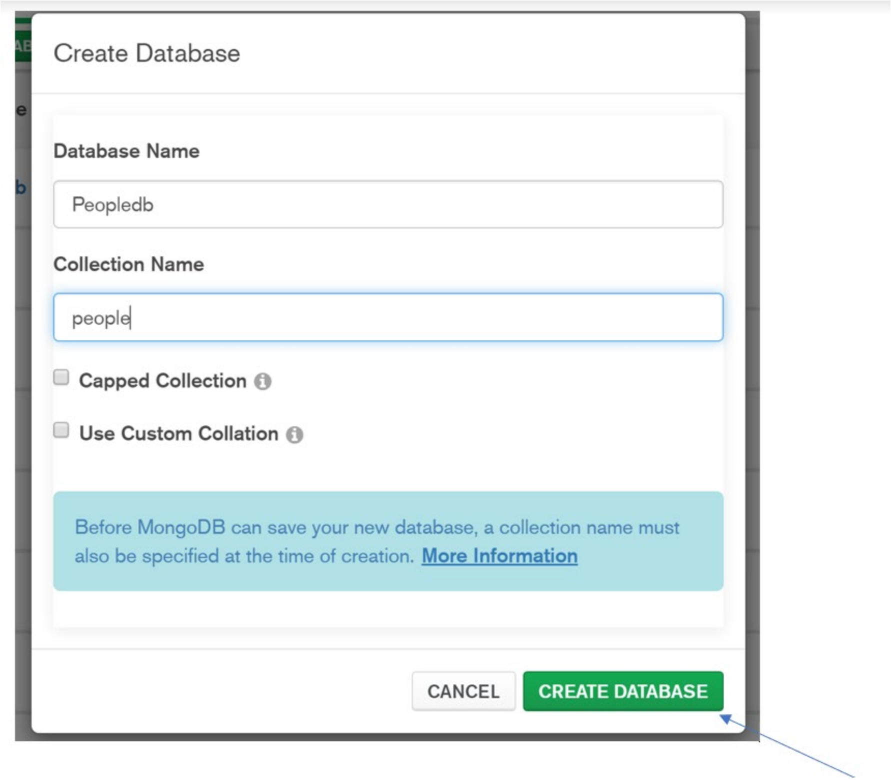
CLUSTER
Standalone

EDITION
MongoDB 4.2.7 Community

Database Name ▲	Storage Size	Collections	Indexes
Peopledb	52.0KB	1	2
admin	20.0KB	0	1

write Peopledb in Database Name

people in collection Name



Create Database

Database Name: Peopledb

Collection Name: people

□ Capped Collection

□ Use Custom Collation

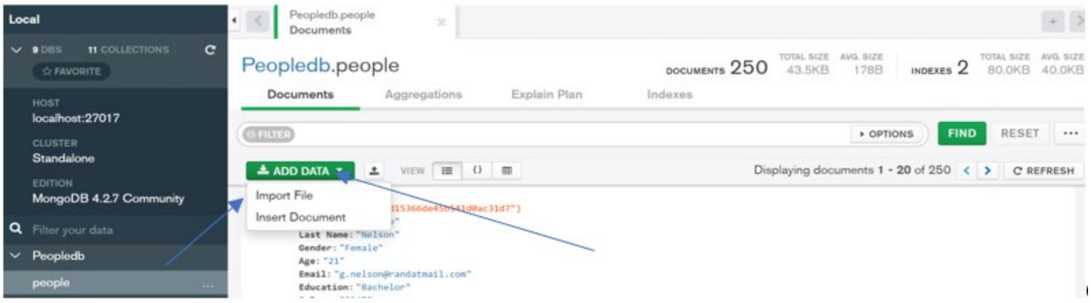
Before MongoDB can save your new database, a collection name must also be specified at the time of creation. More Information

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 .



Local

9 DBS 11 COLLECTIONS

HOST: localhost:27017

CLUSTER: Standalone

EDITION: MongoDB 4.2.7 Community

Peopledb.people

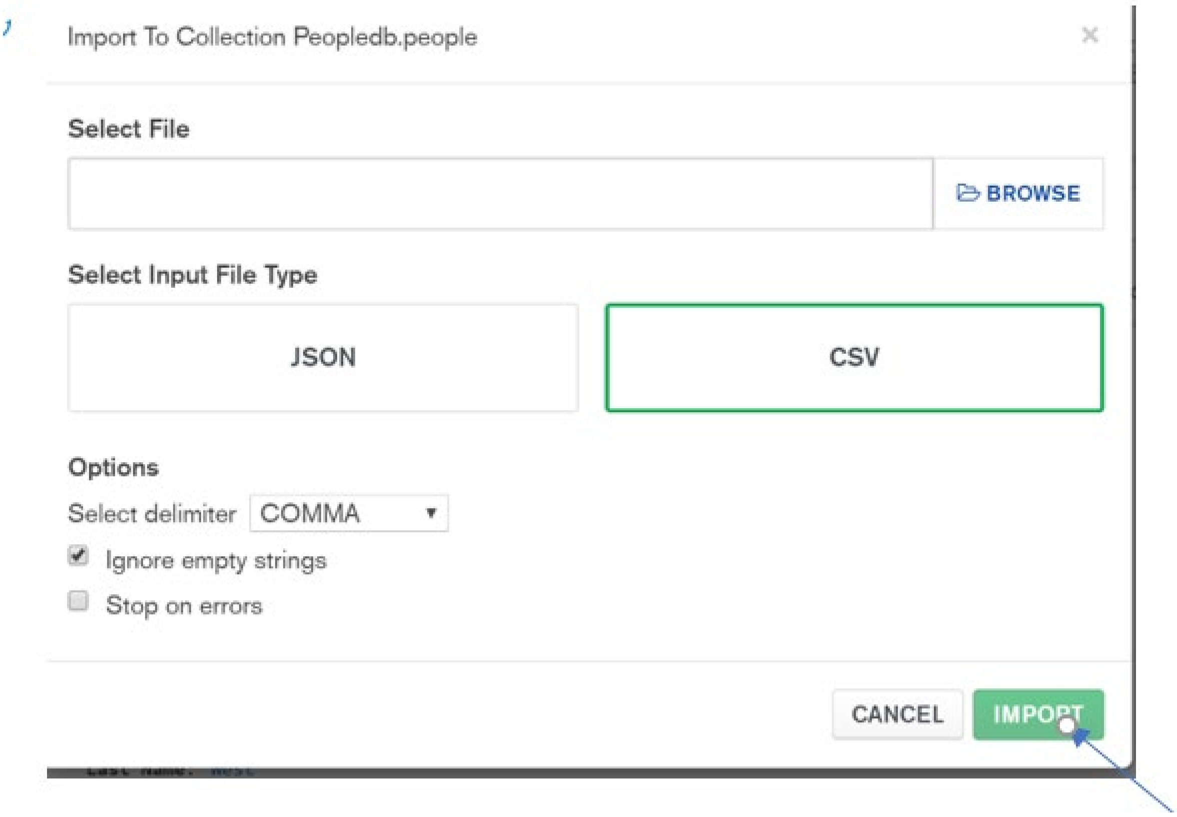
DOCUMENTS 250 TOTAL SIZE 43.5KB AVG. SIZE 178B

INDEXES 2 TOTAL SIZE 80.0KB AVG. SIZE 40.0KB

Displaying documents 1 - 20 of 250

```
Last Name: "Nelson"
Gender: "Female"
Age: "21"
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
```

The following screen will appear. Select CSV , Click browser to locate the file people.csv (provided to you for this week 's lab on Moodle) in your computer select the format CSV for the people.csv file and finally click import to import the selected file.



Import To Collection Peopledb.people

Select File

Select Input File Type

JSON | CSV

Options

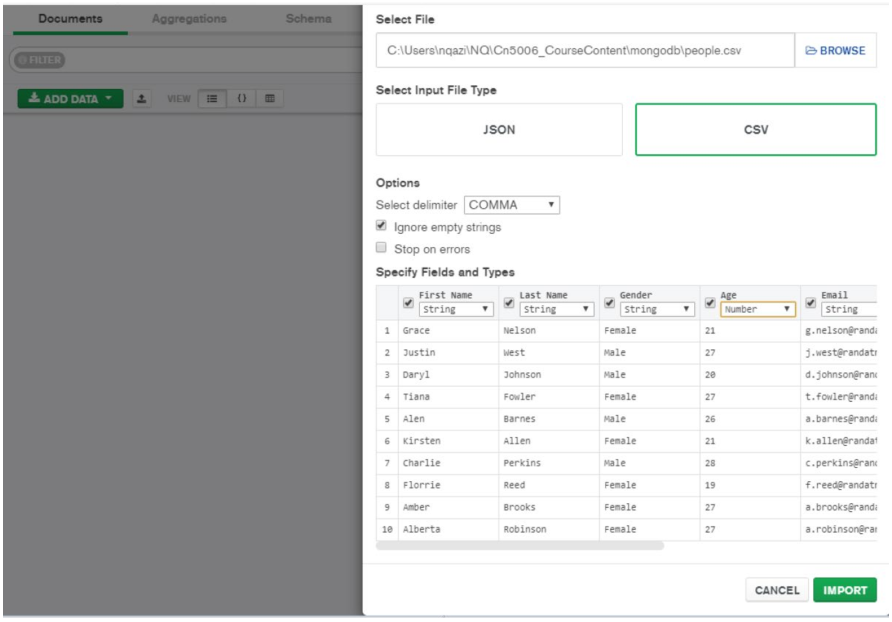
Select delimiter COMMA

☑ Ignore empty strings

☐ Stop on errors

Changing the Data type of the Column Field , while importing CSV data

Click the field or column you want to change the field. In this case, we select the Age field. Change the type to number from string.



Select File

C:\Users\nqazi\NQ\Cn5006_CourseContent\mongodb\people.csv

Select Input File Type

JSON | CSV

Options

Select delimiter COMMA

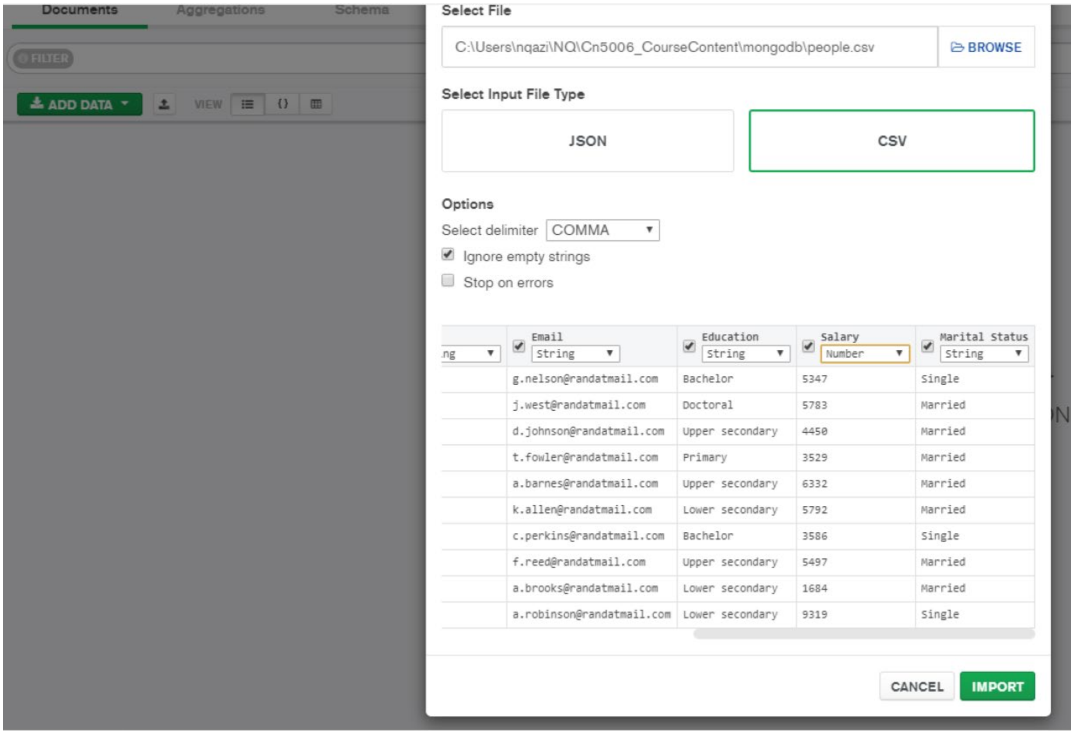
☑ Ignore empty strings

☐ Stop on errors

Specify Fields and Types

	☑ First Name String	☑ Last Name String	☑ Gender String	☑ Age Number	☑ Email String
1	Grace	Nelson	Female	21	g.nelson@randa
2	Justin	West	Male	27	j.west@randatr
3	Daryl	Johnson	Male	20	d.johnson@ranc
4	Tiana	Fowler	Female	27	t.fowler@randa
5	Alen	Barnes	Male	26	a.barnes@randa
6	Kirsten	Allen	Female	21	k.allen@randat
7	Charlie	Perkins	Male	28	c.perkins@ranc
8	Florrie	Reed	Female	19	f.reed@randatr
9	Amber	Brooks	Female	27	a.brooks@randa
10	Alberta	Robinson	Female	27	a.robinson@rar

Next change the type of the field Salary. Select Number and click the import button



Select File

C:\Users\nqazi\NQ\Cn5006_CourseContent\mongodb\people.csv

Select Input File Type

Options

Select delimiter COMMA

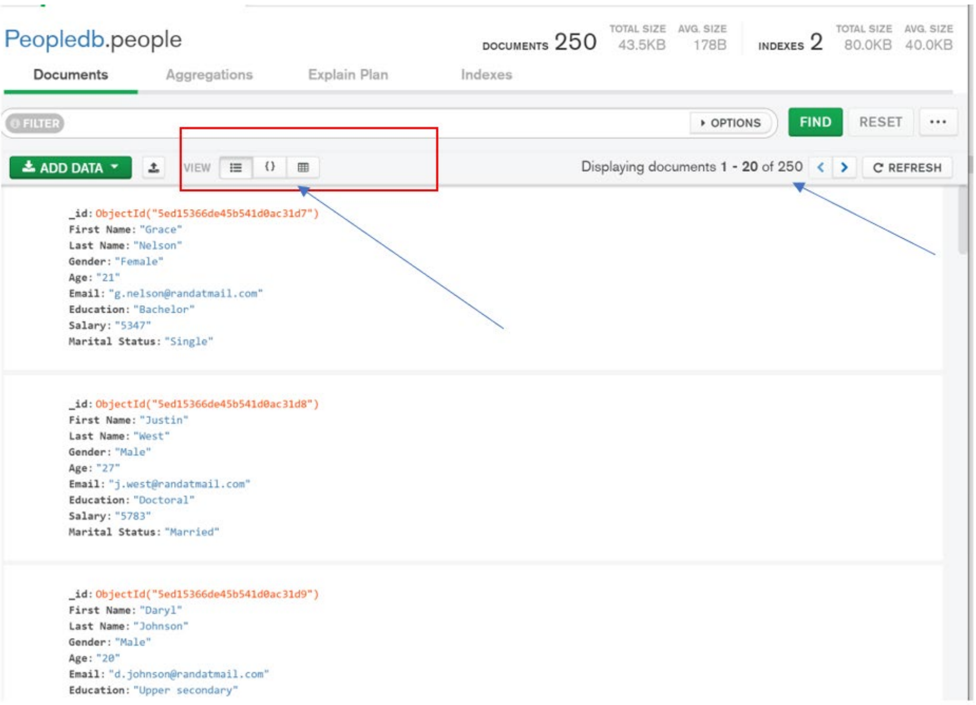
☑ Ignore empty strings

☐ Stop on errors

☑ ng	☑ Email	☑ Education	☑ Salary	☑ Marital Status
	String	String	Number	String
	g.nelson@randatmail.com	Bachelor	5347	Single
	j.west@randatmail.com	Doctoral	5783	Married
	d.johnson@randatmail.com	Upper secondary	4450	Married
	t.fowler@randatmail.com	Primary	3529	Married
	a.barnes@randatmail.com	Upper secondary	6332	Married
	k.allen@randatmail.com	Lower secondary	5792	Married
	c.perkins@randatmail.com	Bachelor	3586	Single
	f.reed@randatmail.com	Upper secondary	5497	Married
	a.brooks@randatmail.com	Lower secondary	1684	Married
	a.robinson@randatmail.com	Lower secondary	9319	Single

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



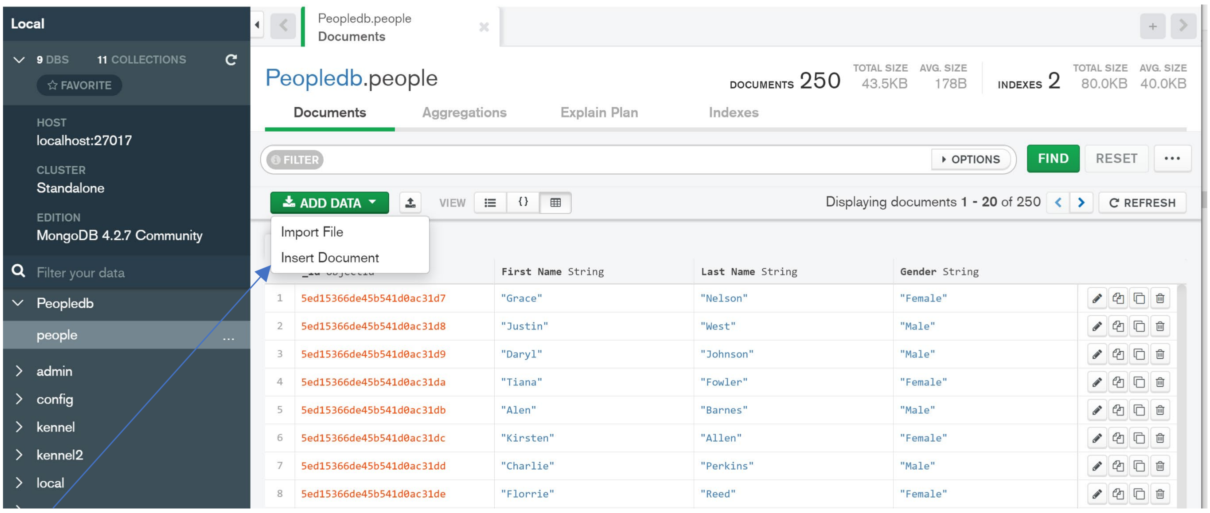
Displaying documents 1 - 20 of 250

```
_id: ObjectId("5ed15366de45b541d0ac31d7")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: "21"
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: "5347"
Marital Status: "Single"

_id: ObjectId("5ed15366de45b541d0ac31d8")
First Name: "Justin"
Last Name: "West"
Gender: "Male"
Age: "27"
Email: "j.west@randatmail.com"
Education: "Doctoral"
Salary: "5783"
Marital Status: "Married"

_id: ObjectId("5ed15366de45b541d0ac31d9")
First Name: "Daryl"
Last Name: "Johnson"
Gender: "Male"
Age: "20"
Email: "d.johnson@randatmail.com"
Education: "Upper secondary"
```

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



HOST
localhost:27017

CLUSTER
Standalone

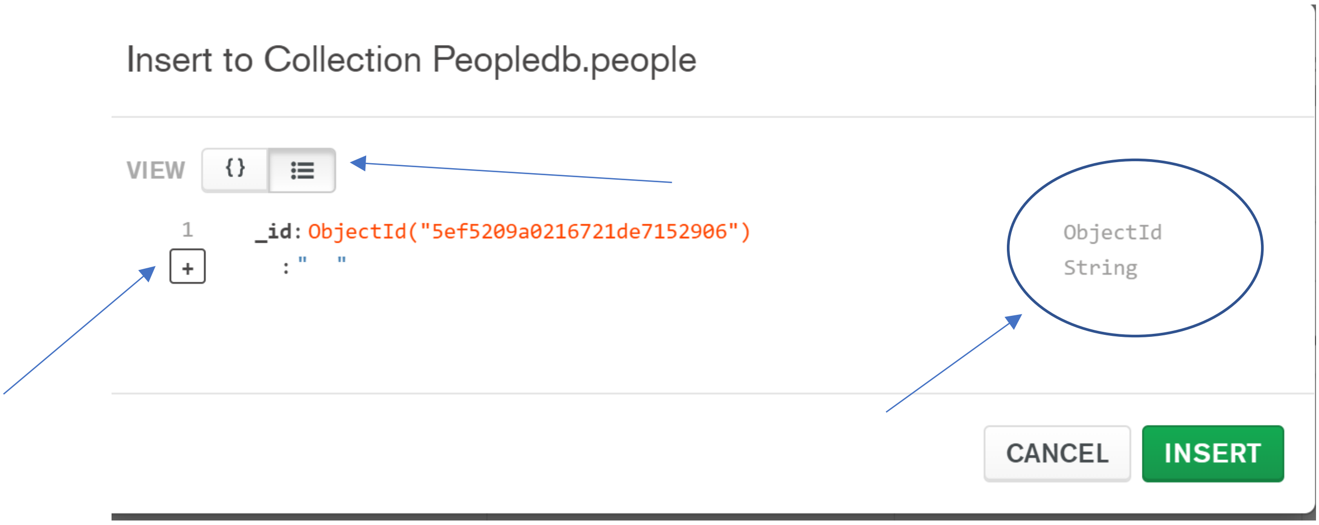
EDITION
MongoDB 4.2.7 Community

DOCUMENTS 250 TOTAL SIZE 43.5KB AVG. SIZE 178B INDEXES 2 TOTAL SIZE 80.0KB AVG. SIZE 40.0KB

Displaying documents 1 - 20 of 250

	_id ObjectId	First Name String	Last Name String	Gender String
1	5ed15366de45b541d0ac31d7	"Grace"	"Nelson"	"Female"
2	5ed15366de45b541d0ac31d8	"Justin"	"West"	"Male"
3	5ed15366de45b541d0ac31d9	"Daryl"	"Johnson"	"Male"
4	5ed15366de45b541d0ac31da	"Tiana"	"Fowler"	"Female"
5	5ed15366de45b541d0ac31db	"Alen"	"Barnes"	"Male"
6	5ed15366de45b541d0ac31dc	"Kirsten"	"Allen"	"Female"
7	5ed15366de45b541d0ac31dd	"Charlie"	"Perkins"	"Male"
8	5ed15366de45b541d0ac31de	"Florrie"	"Reed"	"Female"

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

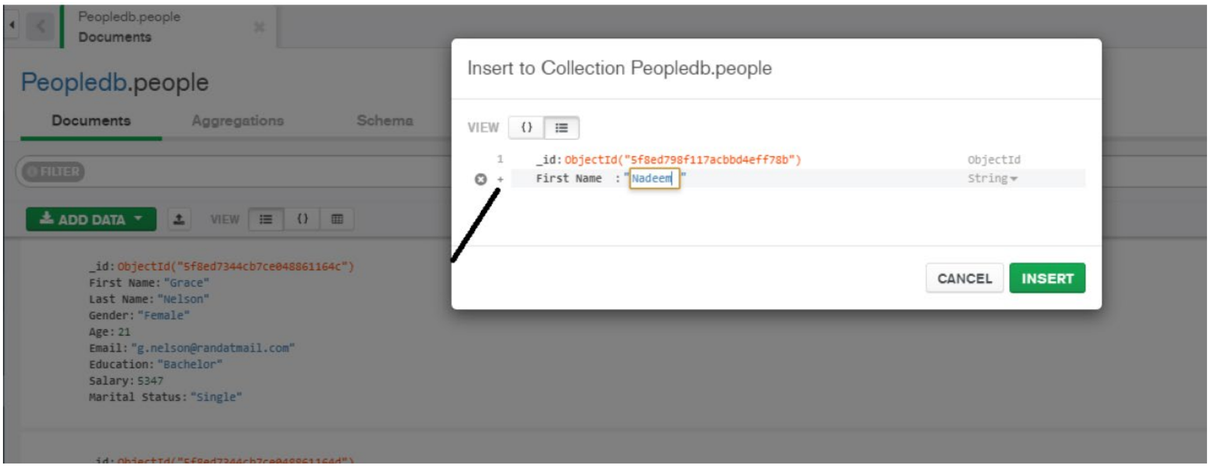


Insert to Collection Peopledb.people

```
1   _id: ObjectId("5ef5209a0216721de7152906")
    : "  "
```

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.



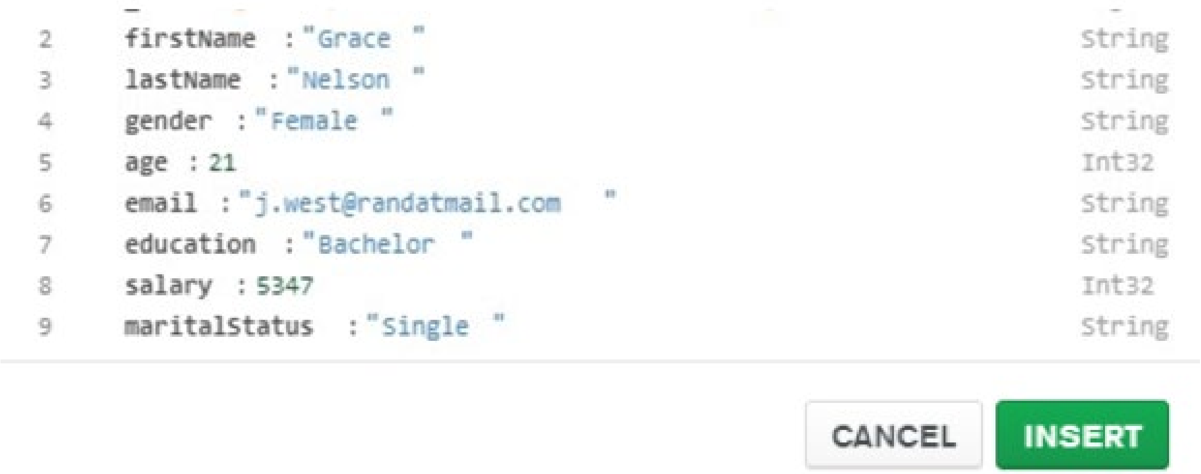
```
1   _id: ObjectId("5f8ed798f117acbbd4eff78b")        ObjectId
    First Name  : "Nadeem "                          String
```

```
_id: ObjectId("5f8ed7344cb7ce048861164c")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: 21
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: 5347
Marital Status: "Single"
```

<u>Add more fields to this document using + sign.</u>

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

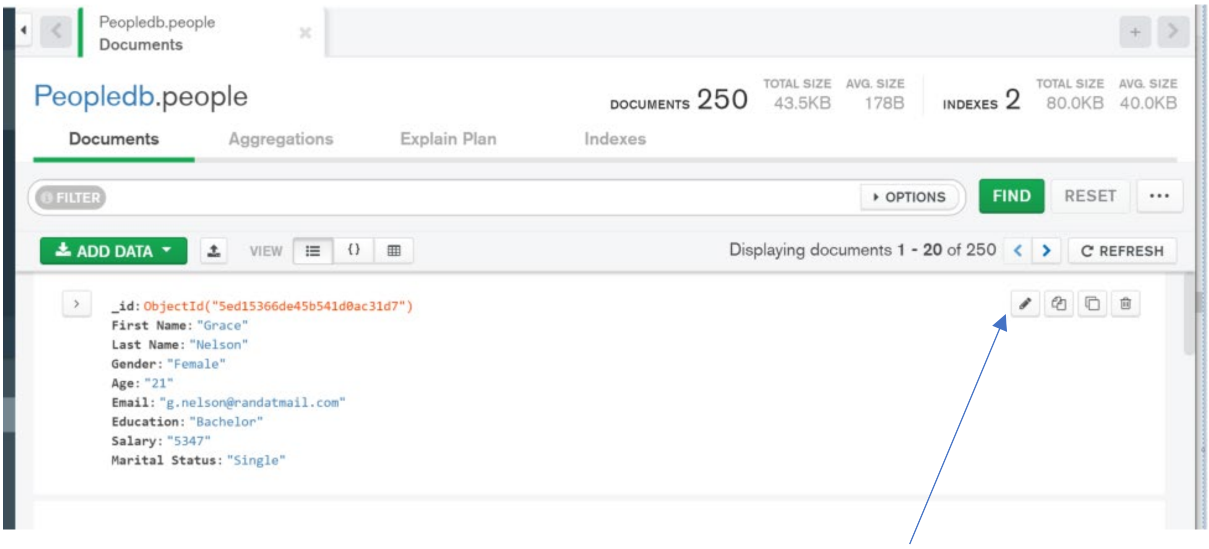
Once you filled all the fields from 2 to 9 then click **<u>Insert</u>** to insert the document in the collection.



```
2   firstName  : "Grace  "                                String
3   lastName  : "Nelson  "                                String
4   gender  : "Female  "                                  String
5   age  : 21                                             Int32
6   email  : "j.west@randatmail.com   "                   String
7   education  : "Bachelor  "                             String
8   salary  : 5347                                        Int32
9   maritalStatus  : "Single  "                           String
```

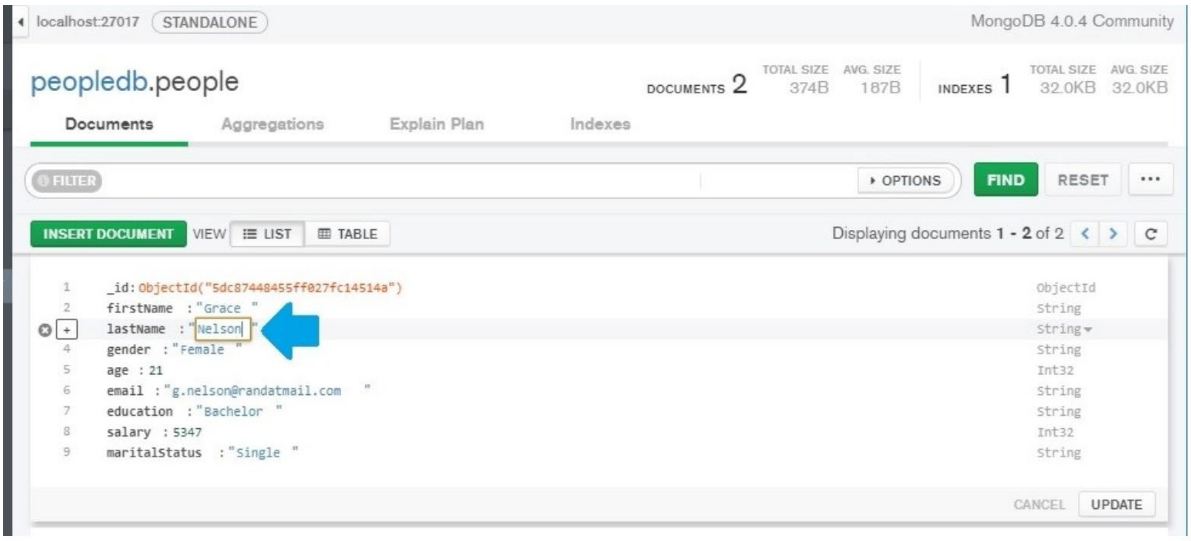
Updating the document

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.



```
_id: ObjectId("5ed15366de45b541d0ac31d7")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: "21"
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: "5347"
Marital Status: "Single"
```

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.

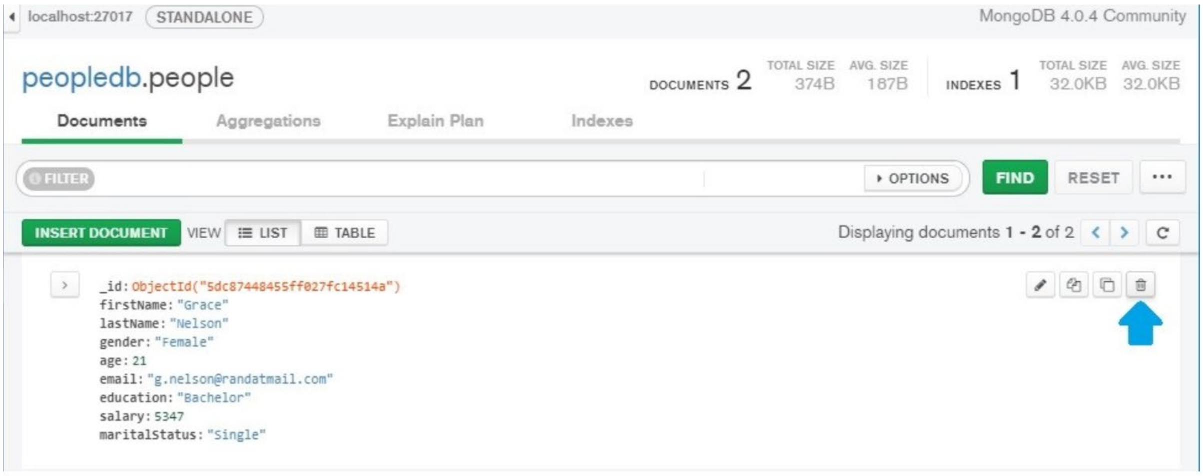


```
_id: ObjectId("5dc87448455ff027fc14514a")         ObjectId
firstName  : "Grace  "                            String
lastName  : "Nelson "                             String
gender  : "Female  "                              String
age  : 21                                         Int32
email  : "g.nelson@randatmail.com   "             String
education  : "Bachelor  "                         String
salary  : 5347                                    Int32
maritalStatus  : "Single  "                       String
```

Deleting the documents

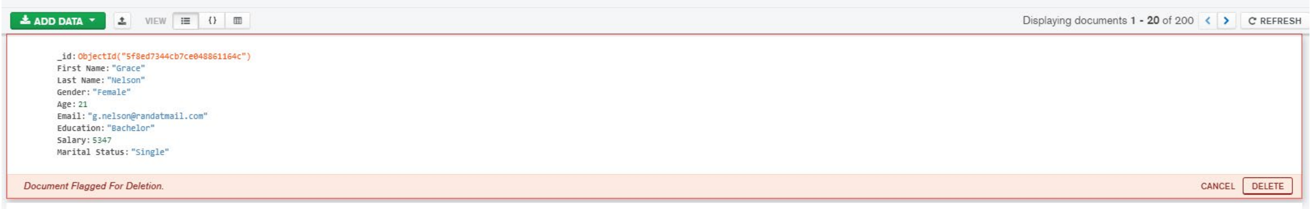
Deleting documents is another simple task with the MongoDB Compass software and can be done with two clicks of a button. Please follow the next steps to delete a document.

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:



```
_id: ObjectId("5dc87448455ff027fc14514a")
firstName: "Grace"
lastName: "Nelson"
gender: "Female"
age: 21
email: "g.nelson@randatmail.com"
education: "Bachelor"
salary: 5347
maritalStatus: "Single"
```

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.



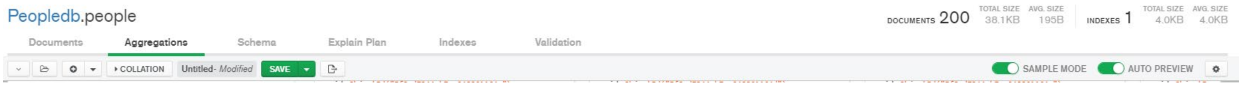
```
_id: ObjectId("5f8ed7344cb7ce048861164c")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: 21
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: 5347
Marital Status: "Single"
```

Document Flagged For Deletion.

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:

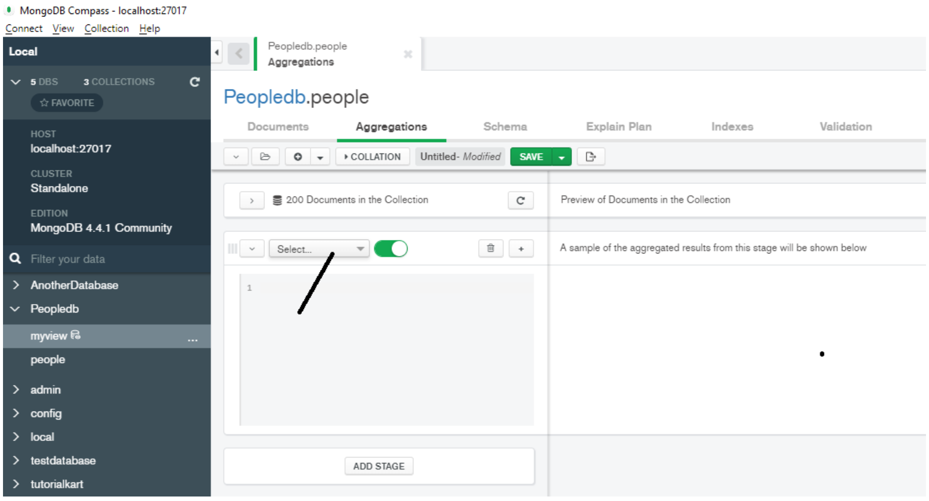
- How many people are there who has got the bachelor degree and are older than 21 years of age.
- what is average age of Female and male in this group
- what is the age salary of the Female and male in this group?
- What is max age of male and female in this group

- What is the min age of male and female in this group?

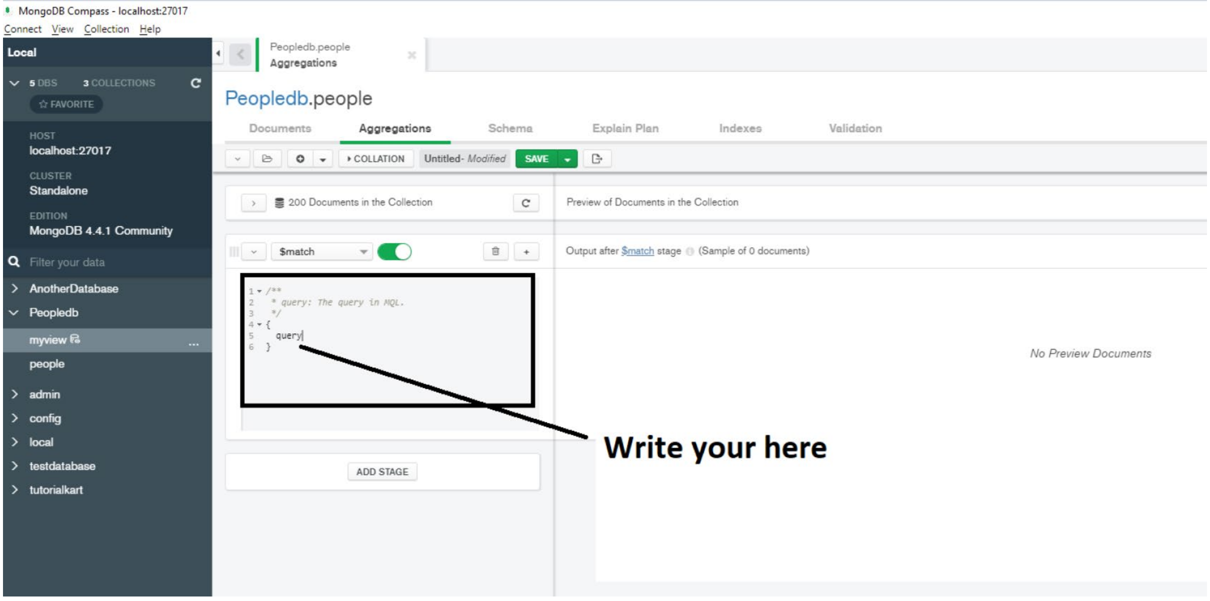
This will be performed in two stages :

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



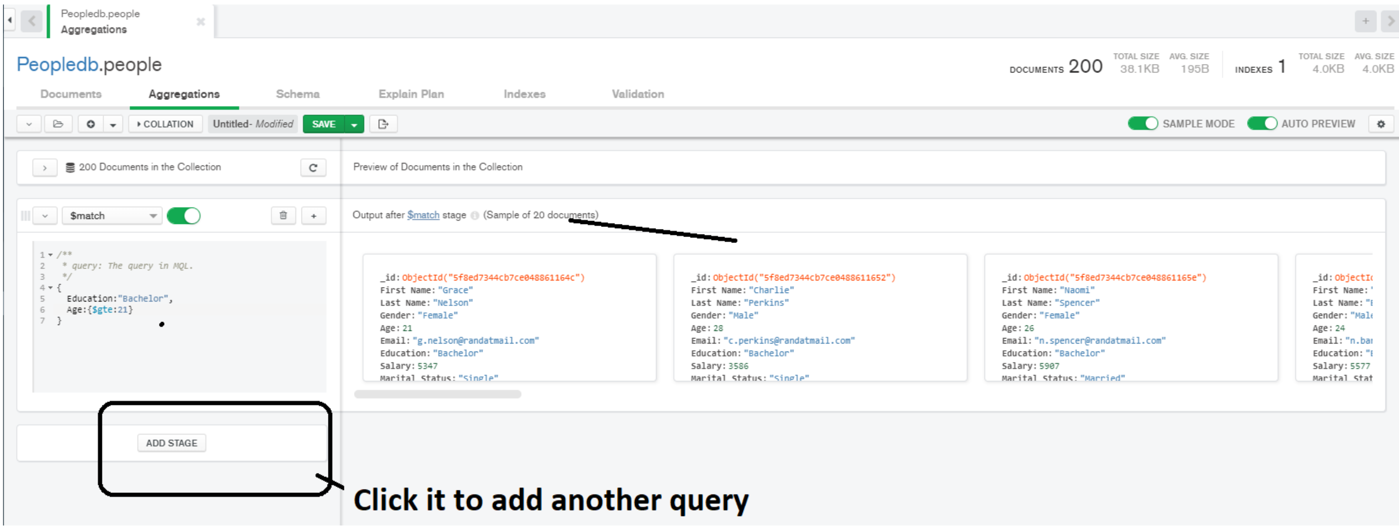
and chose $match as shown in the figure



```
/**
 * query: The query in MQL.
 */
{
  query
}
```

Now write the following Query as shown in the figure. This query will return the documents for all the bachelors age > 21. Note that the query is in JSON data.Education and Age are the names of the fields in the document and Bachelor" and 21 are their respective values. The $match retrieves the records that match this criteria. Note as soon as you finished the write syntax of the query you will see the output in the adjacent window as shown in the figure below

```
{
 Education:"Bachelor",
 Age:{$gte:21}
}
```



```
/**
 * query: The query in MQL.
 */
{
  Education:"Bachelor",
  Age:{$gte:21}
}
```

Output after $match stage (Sample of 20 documents)

```
_id: ObjectId("5f8ed7344cb7ce048861164c")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: 21
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: 5347
Marital Status: "Single"
```

```
_id: ObjectId("5f8ed7344cb7ce0488611652")
First Name: "Charlie"
Last Name: "Perkins"
Gender: "Male"
Age: 28
Email: "c.perkins@randatmail.com"
Education: "Bachelor"
Salary: 3586
Marital Status: "Single"
```

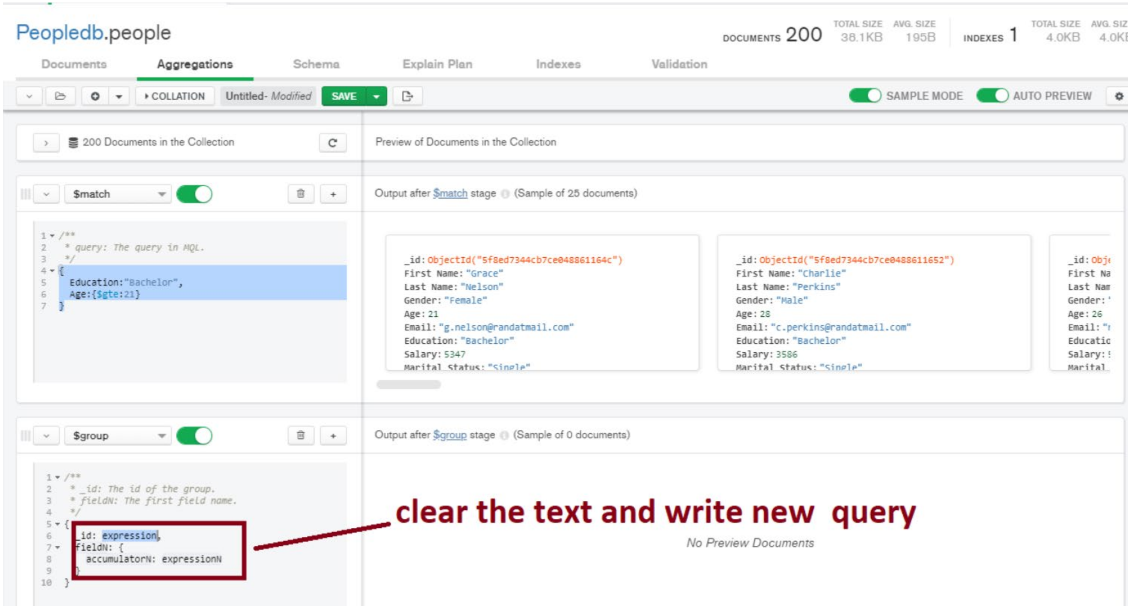
```
_id: ObjectId("5f8ed7344cb7ce048861165e")
First Name: "Naomi"
Last Name: "Spencer"
Gender: "Female"
Age: 26
Email: "n.spencer@randatmail.com"
Education: "Bachelor"
Salary: 5907
Marital Status: "Married"
```

```
_id: ObjectI
First Name: '
Last Name: "
Gender: "Mal
Age: 24
Email: "n.ba
Education: "
Salary: 5577
Marital Stat
```

Click it to add another query

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 :



Peopledb.people
DOCUMENTS 200 TOTAL SIZE 38.1KB AVG. SIZE 195B
INDEXES 1 TOTAL SIZE 4.0KB AVG. SIZ 4.0KI
Documents
Aggregations
Schema
Explain Plan
Indexes
Validation
COLLATION
Untitled- Modified
SAVE
SAMPLE MODE
AUTO PREVIEW
200 Documents in the Collection
Preview of Documents in the Collection
$match
$group

```
/**
 * query: The query in MQL.
 */
{
  Education:"Bachelor",
  Age:{$gte:21}
}
```

Output after $match stage (Sample of 25 documents)

```
_id: ObjectId("5f8ed7344cb7ce048861164c")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: 21
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: 5347
Marital Status: "Single"
```

```
_id: ObjectId("5f8ed7344cb7ce0488611652")
First Name: "Charlie"
Last Name: "Perkins"
Gender: "Male"
Age: 28
Email: "c.perkins@randatmail.com"
Education: "Bachelor"
Salary: 3586
Marital Status: "Single"
```

```
_id: Obj
First Na
Last Nam
Gender: '
Age: 26
Email: "
Educatic
Salary: !
Marital
```

```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: expression,
  fieldN: {
    accumulatorN: expressionN
  }
}
```

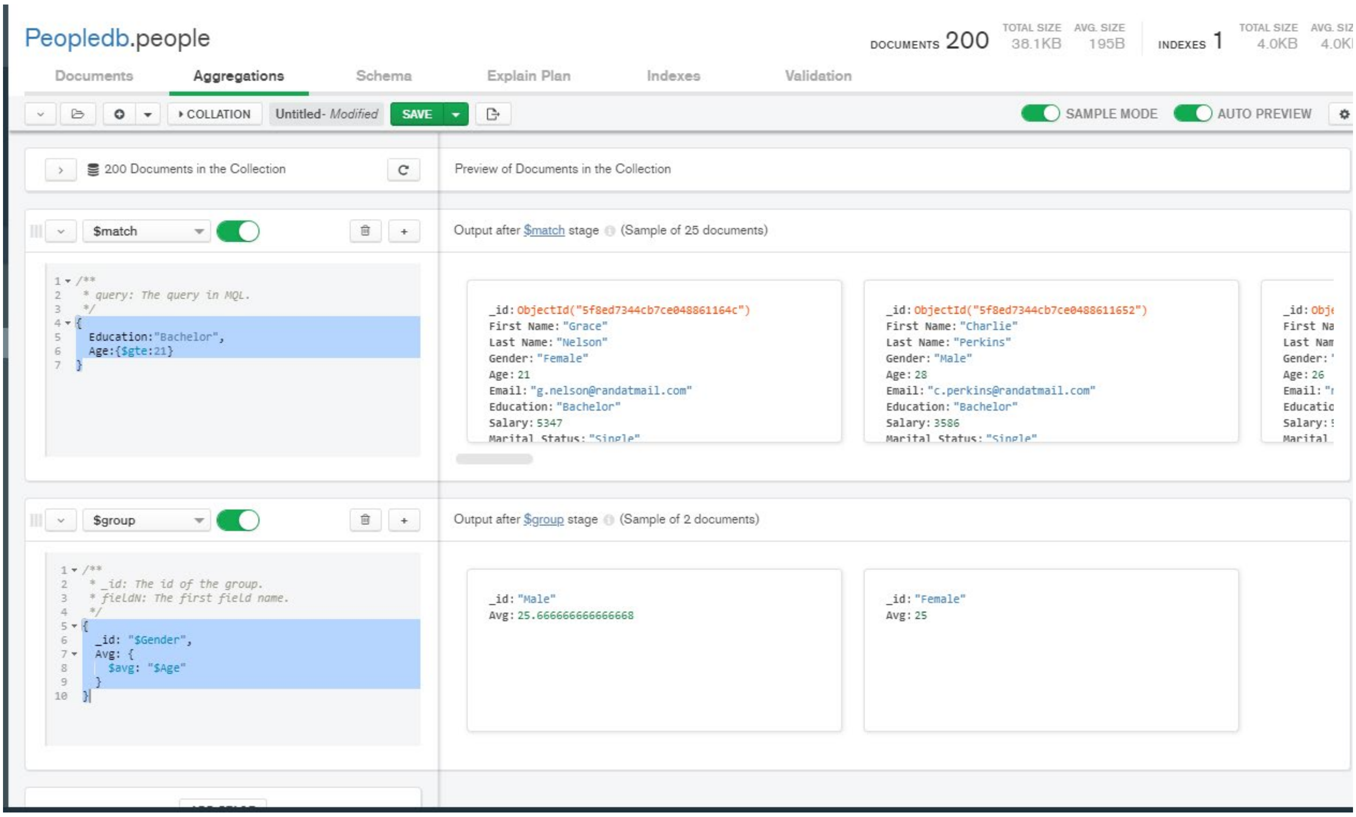
Output after $group stage (Sample of 0 documents)

clear the text and write new query

No Preview Documents

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

```
{
 _id: "$Gender",
 Avg: {
   $avg: "$Age"
 }
}
```



```
/**
 * query: The query in MQL.
 */
{
  Education:"Bachelor",
  Age:{$gte:21}
}
```

```
_id: ObjectId("5f8ed7344cb7ce048861164c")
First Name: "Grace"
Last Name: "Nelson"
Gender: "Female"
Age: 21
Email: "g.nelson@randatmail.com"
Education: "Bachelor"
Salary: 5347
Marital Status: "Single"
```

```
_id: ObjectId("5f8ed7344cb7ce0488611652")
First Name: "Charlie"
Last Name: "Perkins"
Gender: "Male"
Age: 28
Email: "c.perkins@randatmail.com"
Education: "Bachelor"
Salary: 3586
Marital Status: "Single"
```

```
_id: Obj
First Na
Last Nam
Gender: '
Age: 26
Email: "r
Educatic
Salary: !
Marital
```

```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: "$Gender",
  Avg: {
    $avg: "$Age"
  }
}
```

```
_id: "Male"
Avg: 25.666666666666668
```

```
_id: "Female"
Avg: 25
```

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 .

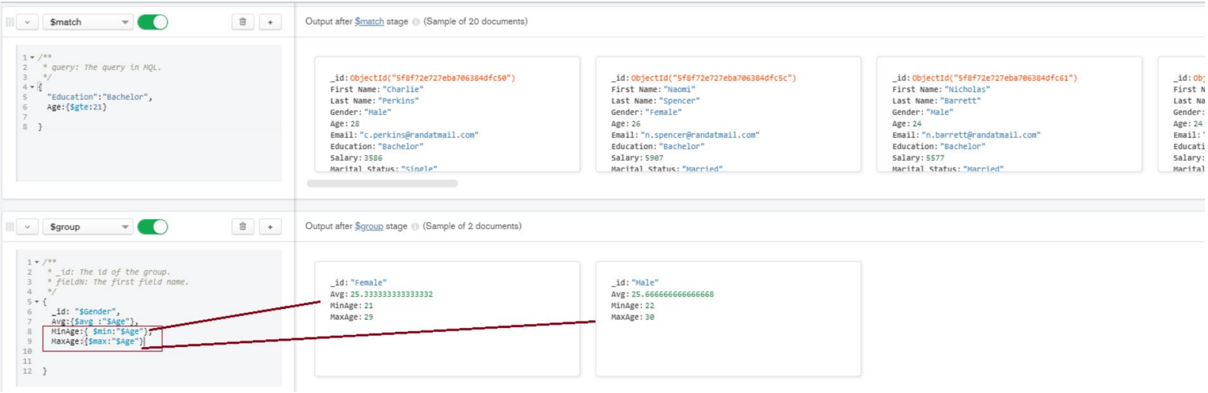
it is analogy of the SQL statement select avg(age) from people group by gender

Add another query for the max and min Age of male and female in this group as shown in the figure below:

Query is :

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

Your screen should look like this



Output after $match stage (Sample of 20 documents)

```
/**
 * query: The query in MQL.
 */
{
  "Education":"Bachelor",
  Age:{$gte:21}

}
```

```
_id: ObjectId("5f8f72e727eba706384dfc50")
First Name: "Charlie"
Last Name: "Perkins"
Gender: "Male"
Age: 28
Email: "c.perkins@randatmail.com"
Education: "Bachelor"
Salary: 3586
Marital Status: "Single"
```

```
_id: ObjectId("5f8f72e727eba706384dfc5c")
First Name: "Naomi"
Last Name: "Spencer"
Gender: "Female"
Age: 26
Email: "n.spencer@randatmail.com"
Education: "Bachelor"
Salary: 5907
Marital Status: "Married"
```

```
_id: ObjectId("5f8f72e727eba706384dfc61")
First Name: "Nicholas"
Last Name: "Barrett"
Gender: "Male"
Age: 24
Email: "n.barrett@randatmail.com"
Education: "Bachelor"
Salary: 5577
Marital Status: "Married"
```

Output after $group stage (Sample of 2 documents)

```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: "$Gender",
  Avg:{$avg :"$Age"},
  MinAge:{ $min:"$Age"},
  MaxAge:{$max:"$Age"}


}
```

```
_id: "Female"
Avg: 25.333333333333332
MinAge: 21
MaxAge: 29
```

```
_id: "Male"
Avg: 25.666666666666668
MinAge: 22
MaxAge: 30
```

Next Query find the min, max and average salary of the male and female

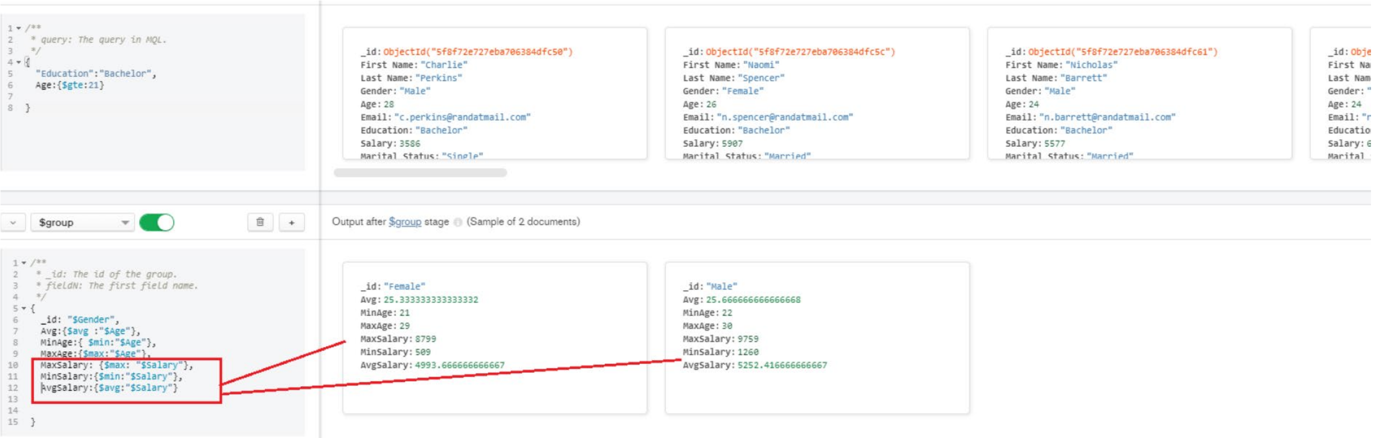
Add the following query. Notice this time it is not age but salary in double quotes

MaxSalary: {$max: "$Salary"},

MinSalary:{$min:"$Salary"},

AvgSalary:{$avg:"$Salary"}

Your screen now should look like this



```
/**
 * query: The query in MQL.
 */
{
  "Education":"Bachelor",
  Age:{$gte:21}

}
```

```
_id: ObjectId("5f8f72e727eba706384dfc50")
First Name: "Charlie"
Last Name: "Perkins"
Gender: "Male"
Age: 28
Email: "c.perkins@randatmail.com"
Education: "Bachelor"
Salary: 3586
Marital Status: "Single"
```

```
_id: ObjectId("5f8f72e727eba706384dfc5c")
First Name: "Naomi"
Last Name: "Spencer"
Gender: "Female"
Age: 26
Email: "n.spencer@randatmail.com"
Education: "Bachelor"
Salary: 5907
Marital Status: "Married"
```

```
_id: ObjectId("5f8f72e727eba706384dfc61")
First Name: "Nicholas"
Last Name: "Barrett"
Gender: "Male"
Age: 24
Email: "n.barrett@randatmail.com"
Education: "Bachelor"
Salary: 5577
Marital Status: "Married"
```

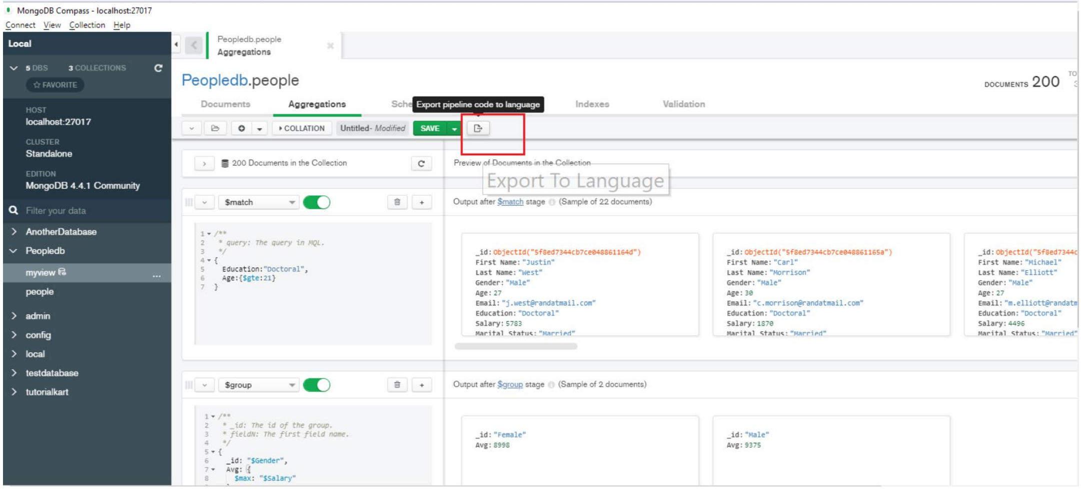
Output after $group stage (Sample of 2 documents)

```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: "$Gender",
  Avg:{$avg :"$Age"},
  MinAge:{ $min:"$Age"},
  MaxAge:{$max:"$Age"},
  MaxSalary: {$max: "$Salary"},
  MinSalary:{$min:"$Salary"},
  AvgSalary:{$avg:"$Salary"}


}
```

```
_id: "Female"
Avg: 25.333333333333332
MinAge: 21
MaxAge: 29
MaxSalary: 8799
MinSalary: 509
AvgSalary: 4993.666666666667
```

```
_id: "Male"
Avg: 25.666666666666668
MinAge: 22
MaxAge: 30
MaxSalary: 9759
MinSalary: 1260
AvgSalary: 5252.416666666667
```

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



MongoDB Compass - localhost:27017

Connect View Collection Help

Local — 5 DBS 3 COLLECTIONS

HOST localhost:27017

CLUSTER Standalone

EDITION MongoDB 4.4.1 Community

Peopledb.people — Aggregations

DOCUMENTS 200

Export pipeline code to language

200 Documents in the Collection

$match

```
/**
 * query: The query in MQL.
 */
{
  Education:"Doctoral",
  Age:{$gte:21}
}
```

Output after $match stage (Sample of 22 documents)

```
_id: ObjectId("5f8ed7344cb7ce048861164d")
First Name: "Justin"
Last Name: "West"
Gender: "Male"
Age: 27
Email: "j.west@randatmail.com"
Education: "Doctoral"
Salary: 5783
Marital Status: "Married"
```

```
_id: ObjectId("5f8ed7344cb7ce048861165a")
First Name: "Carl"
Last Name: "Morrison"
Gender: "Male"
Age: 30
Email: "c.morrison@randatmail.com"
Education: "Doctoral"
Salary: 1870
Marital Status: "Married"
```

```
_id: ObjectId("5f8ed7344c
First Name: "Michael"
Last Name: "Elliott"
Gender: "Male"
Age: 27
Email: "m.elliott@randatm
Education: "Doctoral"
Salary: 4496
Marital Status: "Married"
```

$group

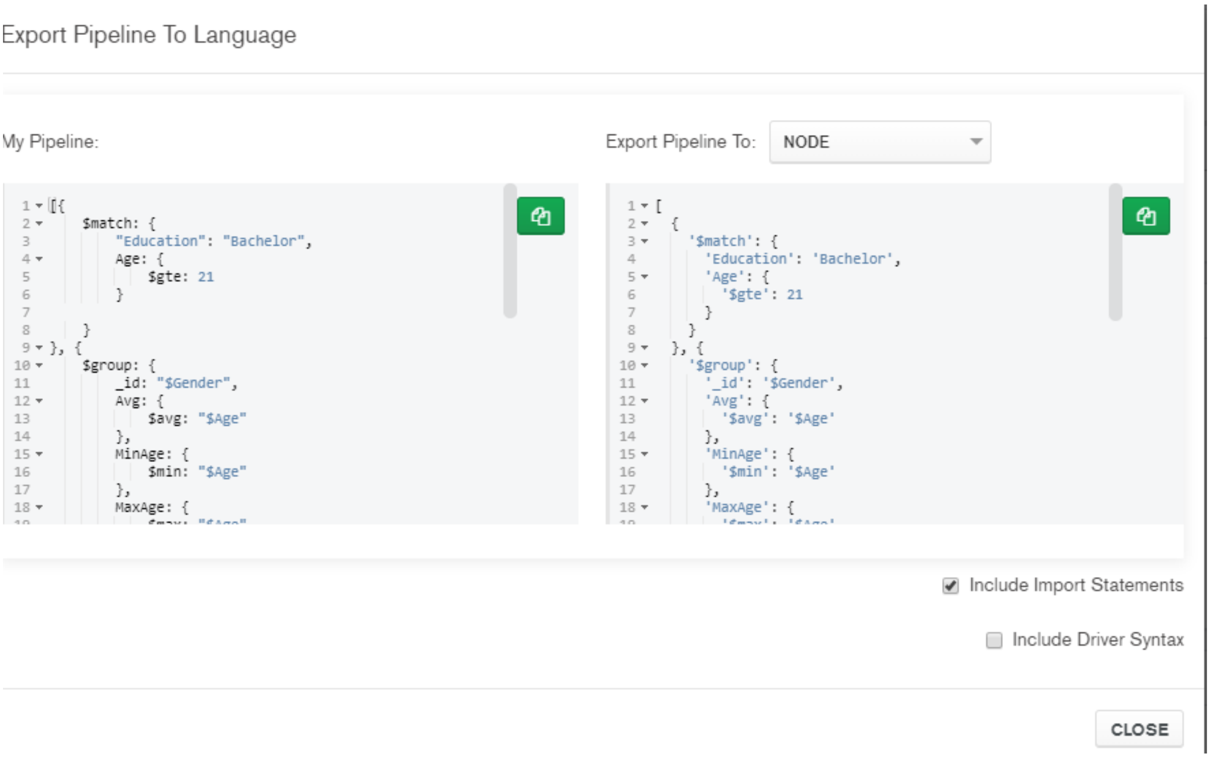
```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: "$Gender",
  Avg: {
    $max: "$Salary"
```

Output after $group stage (Sample of 2 documents)

```
_id: "Female"
Avg: 8998
```

```
_id: "Male"
Avg: 9375
```

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



Export Pipeline To Language

My Pipeline:

```
[{
    $match: {
        "Education": "Bachelor",
        Age: {
            $gte: 21
        }

    }
}, {
    $group: {
        _id: "$Gender",
        Avg: {
            $avg: "$Age"
        },
        MinAge: {
            $min: "$Age"
        },
        MaxAge: {
```

Export Pipeline To: NODE

```
[
  {
    '$match': {
      'Education': 'Bachelor',
      'Age': {
        '$gte': 21
      }
    }
  }, {
    '$group': {
      '_id': '$Gender',
      'Avg': {
        '$avg': '$Age'
      },
      'MinAge': {
        '$min': '$Age'
      },
      'MaxAge': {
```

☑ Include Import Statements

☐ Include Driver Syntax

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/