Where,AND,OR&CRUD

EXPERIMENT-1

WHERE

It is a collection that you want to filter a subset based on condition .That is the place we are going to use the WHERE

A screenshot of a computer code

Description automatically generated

Output:



Output explanation:

The code in the image is using MongoDB to query a database of students. Let's break it down step by step

Connection: The first line establishes a connection to the MongoDB server running locally on the machine. The mongodb://127.0.0.1:27017/ part of the URI specifies the connection string, which includes the protocol (mongodb), hostname (localhost), and port number (27017), which is the default port for MongoDB. The ?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies connection options.

Database: After connecting to the server, the code switches to the Student database using the db object. This database presumably stores information about students.

Query: The line db.Student.find({gpa:{$gt:3}}) is a query that retrieves documents from the Student collection where the student's GPA is greater than 3. The find method is used to find documents in a collection, and it takes a query document as an argument. The query document specifies the criteria for selecting documents. In this case, the query document is {gpa:{$gt:3}}, which means that the GPA field must be greater than ('>') 3.

Results: The query returns a cursor pointing to the matching documents. The cursor is used to iterate over the results. In the image, we can see that six documents were found that match the criteria. Each document contains information about a student, including their name, age, courses, GPA, home city, blood group, and whether they are a hotel resident.

Here are some additional points to note:

The \_id field is a special field that MongoDB automatically adds to each document. It is a unique identifier for the document.

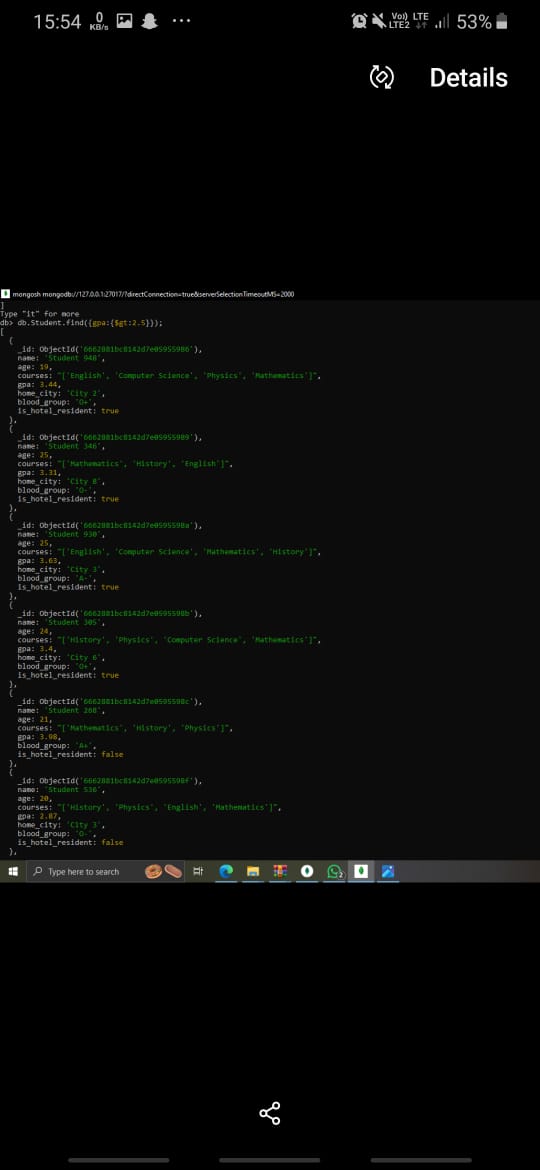
The courses field is an array of strings. This means that a student can be enrolled in multiple courses.

There seems to be a typo in the field name is hotel resident in two of the documents. It should be is\_hotel\_resident.

I hope this explanation helps!

pen\_spark

OUTPUT:



Output explanation:

Sure, the code in the image is using MongoDB to query a database of students. Let's break it down step by step:

Connection: The first line establishes a connection to the MongoDB server running locally on the machine. The mongodb://127.0.0.1:27017//directConnection-truediserverSelectionTimeoutMS-2000 part of the URI specifies the connection string, which includes the protocol (mongodb), hostname (localhost), and port number (27017), which is the default port for MongoDB. The ?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies connection options.

Database: After connecting to the server, the code switches to the Student database using the db object. This database presumably stores information about students.

Query: The line db.Student.find({gpa:($gt:2.5}}) is a query that retrieves documents from the Student collection where the student's GPA is greater than 2.5. The find method is used to find documents in a collection, and it takes a query document as an argument. The query document specifies the criteria for selecting documents. In this case, the query document is {gpa:{$gt:2.5}}, which means that the GPA field must be greater than ('>') 2.5.

Results: The query returns a cursor pointing to the matching documents. The cursor is used to iterate over the results. In the image, we can see that six documents were found that match the criteria. Each document contains information about a student, including their name, age, courses, GPA home city, blood group, and whether they are a hotel resident.

Here are some additional points to note:

The \_id field is a special field that MongoDB automatically adds to each document. It is a unique identifier for the document.

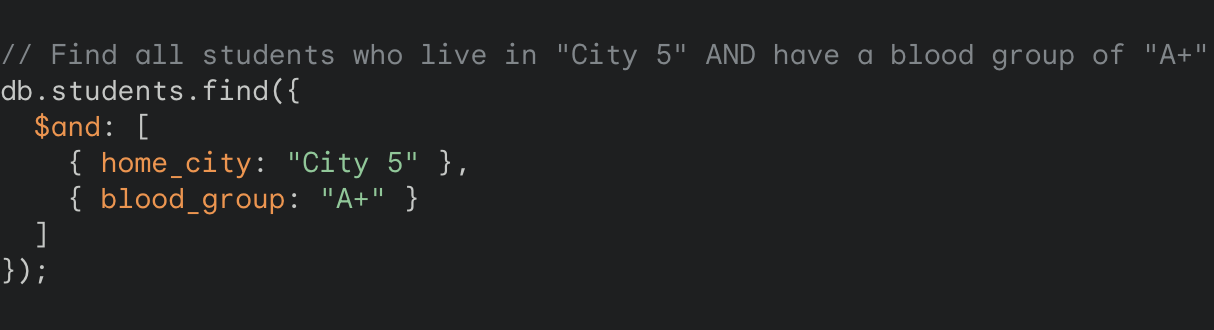
The courses field is an array of strings. This means that a student can be enrolled in multiple courses.

There seems to be a typo in the field name is hotel resident in two of the documents. It should be is\_hotel\_residnt.

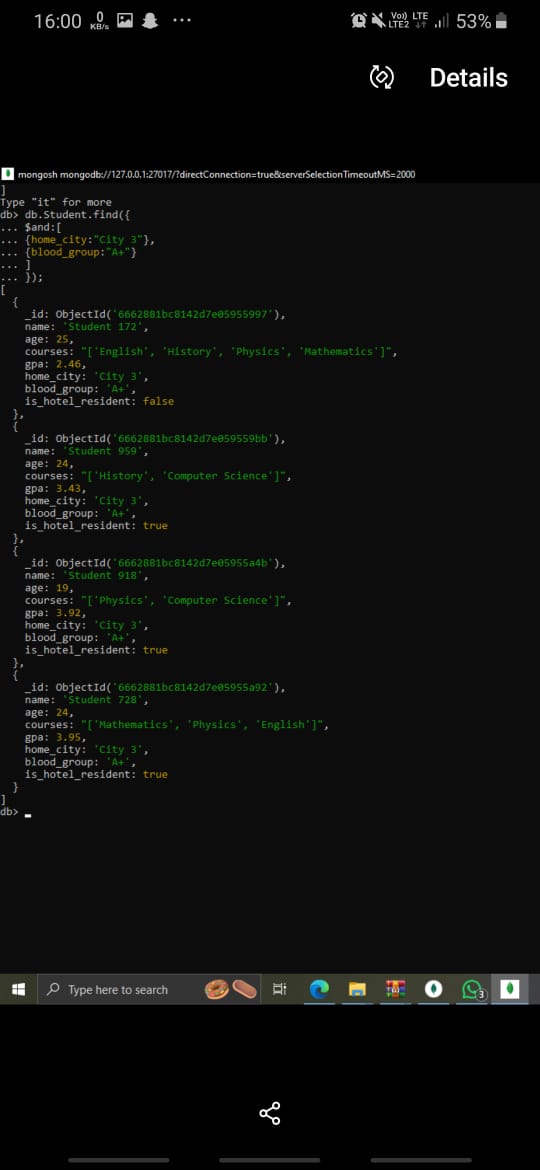
AND

TO FILTER A SUBSET BASED ON MULTIPLE CONDITION A COLLECTION THAT WE WANT

INPUT:



OUTPUT:

  
output explaination:

Sure, the code in the image is using MongoDB to query a database of students. Let's break it down step by step:

Connection: The first line establishes a connection to the MongoDB server running locally on the machine. The mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies the connection string, which includes the protocol (mongodb), hostname (localhost), and port number (27017), which is the default port for MongoDB. The ?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies connection options.

Database: After connecting to the server, the code switches to the Student database using the db object. This database presumably stores information about students.

Query: The line db.Student.find({$and: [{"home\_city":"City 3"}, {"blood\_group": "A+"}]})の説明 is a query that retrieves documents from the Student collection that meet two conditions: the student's home city must be "City 3" and their blood group must be "A+". The find method is used to find documents in a collection, and it takes a query document as an argument. The query document specifies the criteria for selecting documents. In this case, the query document is {$and: [{"home\_city":"City 3"}, {"blood\_group": "A+"}]}, which uses the logical AND operator ($and) to combine multiple conditions.

{"home\_city":"City 3"}: This part of the query document specifies that the home\_city field in the document must be equal to "City 3".

{"blood\_group": "A+"}: This part of the query document specifies that the blood\_group field in the document must be equal to "A+".

Explain: The .explain() part of the query helps to explain the query execution plan chosen by MongoDB to find the data. It's optional and useful for understanding how MongoDB finds the data.

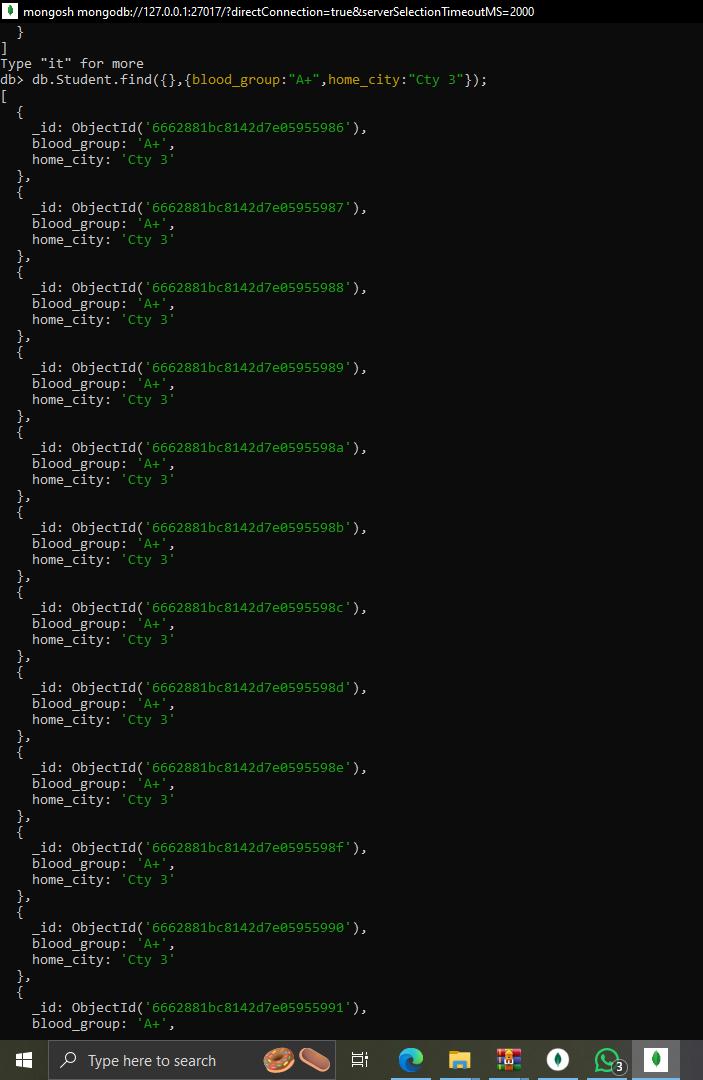
Results: The query returns a cursor pointing to the matching documents. The cursor is used to iterate over the results. In the image, we can see that three documents were found that match the criteria. Each document contains information about a student, including their name, age, courses, GPA, home city, blood group, and whether they are a hotel resident.

Here are some additional points to note:

The \_id field is a special field that MongoDB automatically adds to each document. It is a unique identifier for the document.

The courses field is an array of strings. This means that a student can be enrolled in multiple courses.

There seems to be a typo in the field name is hotel resident in two of the documents. It should be is\_hotel\_resident.



Output explaination:

Sure, the code in the image is using MongoDB to query a database of students. Let's break it down step by step:

Connection:

The first line establishes a connection to the MongoDB server running locally on the machine. The mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies the connection string, which includes the protocol (mongodb), hostname (localhost), and port number (27017), which is the default port for MongoDB. The ?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies connection options.

Database:

After connecting to the server, the code switches to the Student database using the db object. This database presumably stores information about students.

Query:

The line db.Student.find({}, {blood\_group: "A+", home\_city: "Cty 3"}) is a query that retrieves documents from the Student collection that meet two conditions: the student's blood group must be "A+" and their home city must be "Cty 3". The find method is used to find documents in a collection, and it takes two arguments:

A query document: This specifies the criteria for selecting documents. In this case, the query document is {} which means retrieve all documents.

A projection document: This specifies which fields to include or exclude in the returned documents. In this case, the projection document {blood\_group: "A+", home\_city: "Cty 3"} specifies that we only want to include the blood\_group and home\_city fields in the results.

Results:

The query returns a cursor pointing to the matching documents. The cursor is used to iterate over the results. In the image, we can see that several documents were found that match the criteria. Each document contains only the specified fields (blood\_group and home\_city), and their values.

Here are some additional points to note:

The \_id field is a special field that MongoDB automatically adds to each document. It is a unique identifier for the document, but it is not included in the results because it wasn't specified in the projection document.

There seems to be a typo in the field name home city in two of the documents. It should be home\_city.

OR

A COLLECTION WE WANT TO FILTER SUBSET BASED ON MULTIPLE CONDITION BUT ANY ONE IS SUFFICIENT

INPUT:



OUTPUT:

A screenshot of a computer program

Description automatically generated

**Output explanation:**

**Sure, the code in the image is using MongoDB to query a database of students. Let's break it down step by step:**

**Connection:**

**The first line establishes a connection to the MongoDB server running locally on the machine. The mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies the connection string, which includes the protocol (mongodb), hostname (localhost), and port number (27017), which is the default port for MongoDB. The ?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies connection options.**

**Database:**

**After connecting to the server, the code switches to the Student database using the db object. This database presumably stores information about students.**

**Query:**

**The line db.Student.find({}, {blood\_group: "A+", home\_city: "Cty 3"}) is a query that retrieves documents from the Student collection that meet two conditions: the student's blood group must be "A+" and their home city must be "Cty 3". The find method is used to find documents in a collection, and it takes two arguments:**

**A query document: This specifies the criteria for selecting documents. In this case, the query document is {} which means retrieve all documents.**

**A projection document: This specifies which fields to include or exclude in the returned documents. In this case, the projection document {blood\_group: "A+", home\_city: "Cty 3"} specifies that we only want to include the blood\_group and home\_city fields in the results.**

**Results:**

**The query returns a cursor pointing to the matching documents. The cursor is used to iterate over the results. In the image, we can see that several documents were found that match the criteria. Each document contains only the specified fields (blood\_group and home\_city), and their values.**

**Here are some additional points to note:**

**The \_id field is a special field that MongoDB automatically adds to each document. It is a unique identifier for the document, but it is not included in the results because it wasn't specified in the projection document**

**There seems to be a typo in the field name home city in two of the documents. It should be home\_city.**

**CRUD**

C - Create / Insert

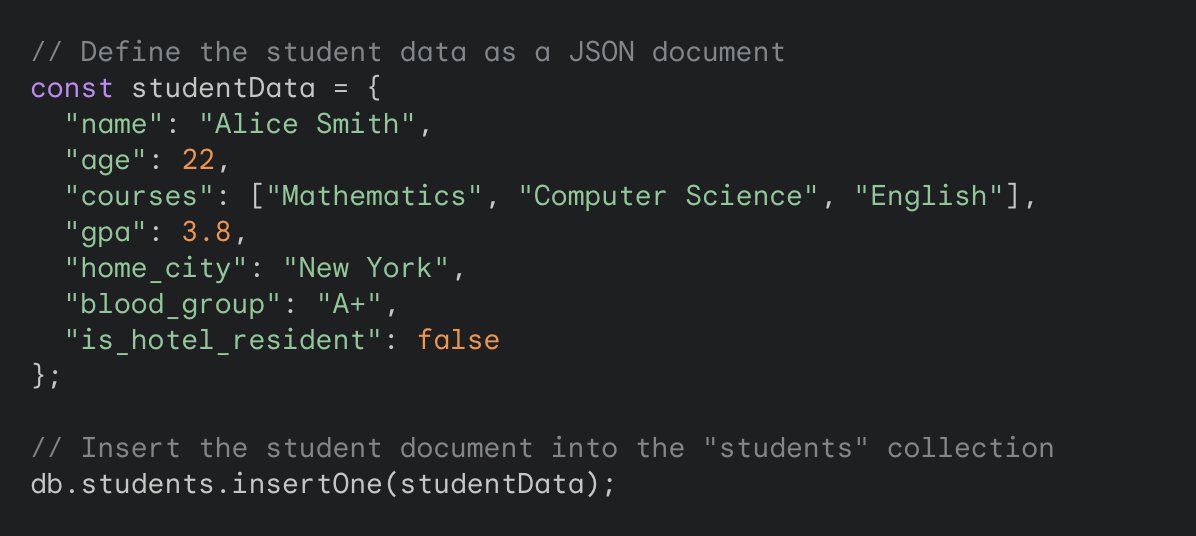
* R - Remove
* U - update
* D - Delete

This is applicable for a Collection (Table)  or a Document (Row)

**Insert**

**Insert is used to insert one or more document and returns a document containing the status of all inserts**

Input:



Output:



Output explaination:

Sure, the code in the image is using MongoDB to query a database of students. Let's break it down step by step:

Connection:

The first line establishes a connection to the MongoDB server running locally on the machine. The mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies the connection string, which includes the protocol (mongodb), hostname (localhost), and port number (27017), which is the default port for MongoDB. The ?directConnection=true&serverSelectionTimeoutMS=2000 part of the URI specifies connection options.

Database:

After connecting to the server, the code switches to the Student database using the db object. This database presumably stores information about students.

Query:

The line db.Student.find({ Sor:[{is\_hotel\_resident: true }, { gpa:{$lt:3.0} }] }) is a query that retrieves documents from the Student collection that meet two conditions:

The student must be a hotel resident (is\_hotel\_resident: true).

The student's GPA must be less than 3.0 (gpa:{$lt:3.0}).

The find method is used to find documents in a collection, and it takes a query document as an argument. The query document in this case uses the logical OR operator ($or) to combine multiple conditions.

{is\_hotel\_resident: true}: This part of the query document specifies that the is\_hotel\_resident field in the document must be equal to true.

{gpa:{$lt:3.0}}: This part of the query document specifies that the gpa field in the document must be less than (<) 3.0.

Results:

The query returns a cursor pointing to the matching documents. The cursor is used to iterate over the results. In the image, we can't see how many documents were found, but the output shows some of the matching documents. Each document contains information about a student, including their name, age, courses, GPA, home city, blood group, and whether they are a hotel resident.

Here are some additional points to note:

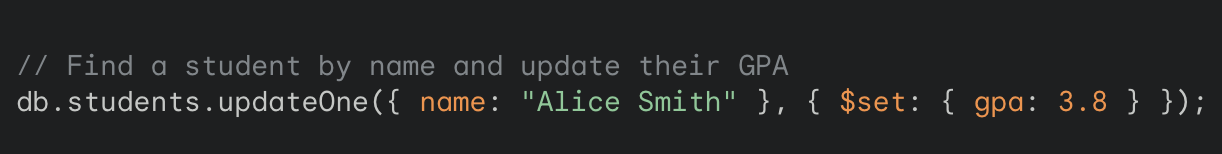
The \_id field is a special field that MongoDB automatically adds to each document. It is a unique identifier for the document, but it is not shown in the output.

The courses field is an array of strings. This means that a student can be enrolled in multiple courses.

There seems to be a typo in the field name is hotel resident in at least one of the documents. It should be is\_hotel\_resident.

UPDATE:

Method used to update document into a collection update(),and save()



Output:

db> db.Students.updateOne({name:"Alice Smith"},{$set:{gpa:3.5}});

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

**Delete**

**Input:**



Output:

db> db.Students.deleteOne({name:"John Doe"});

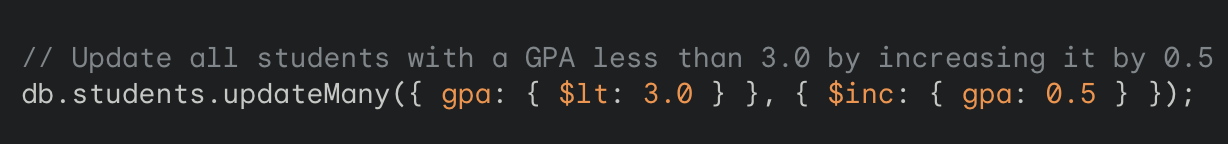
{ acknowledged: true, deletedCount: 0 }

db>

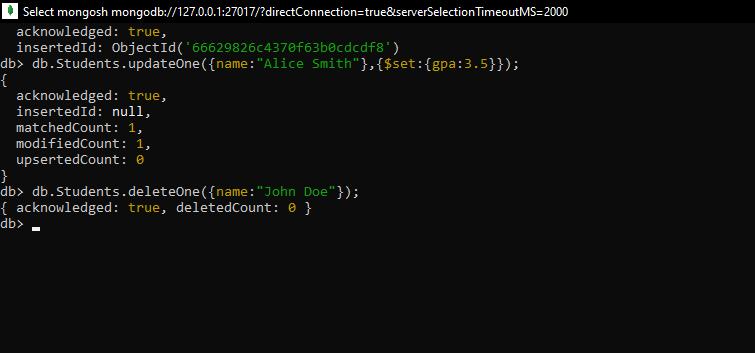
UPDATE MANY:

Update all the documents in mongo db collections that match the given query

Input:



Output:



Output explanation:

The code in the image is a MongoDB query that retrieves student information from a collection named "Students". Let's break down the code step by step:

Connection:

The first line establishes a connection to the MongoDB server running locally on the machine. The URI mongodb://127.0.0.1:27017 specifies the connection details including the hostname (localhost) and port (27017) which is the default port for MongoDB.

Database:

The line db is prepended to the subsequent commands, indicating that the following operations target a database. However, the database name isn't explicitly mentioned in the provided image. It's likely the shell is currently using a default database.

Find Query:

The line db.Students.find({},{name:2,gpa:2}); is the main query that retrieves data from the "Students" collection. Here's how it works:

db.Students.find(): This part targets the "Students" collection and initiates a find operation to retrieve documents.

{}: An empty curly brace {} following the find method indicates that all documents in the collection should be returned. You can use filters here to specify conditions for selecting particular documents.

,{name:2,gpa:2}: This part limits the fields returned in the query results. Here, it specifies that only the "name" and "gpa" fields should be included. The number 2 after each field name indicates that these fields should be included. A value of 0 would exclude the field.

Output:

The query results are displayed below the last line. Each document retrieved from the collection is displayed on a new line, with each field and its corresponding value indented. In this case, it shows all the student documents including their name and gpa.

**Delete Many**

**Method used to delete many multiple documents from a collection in mongo db**

**Input**



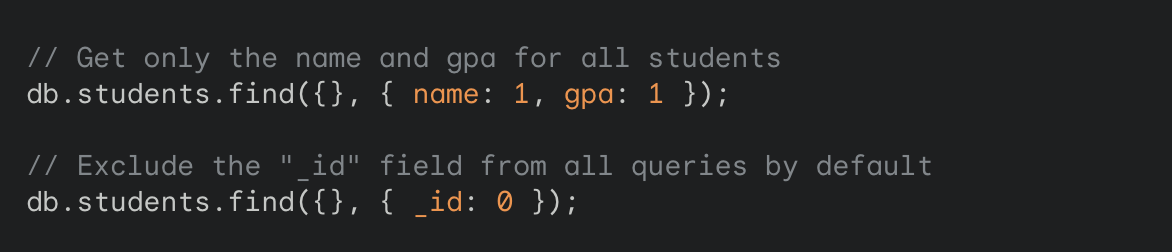
Output:



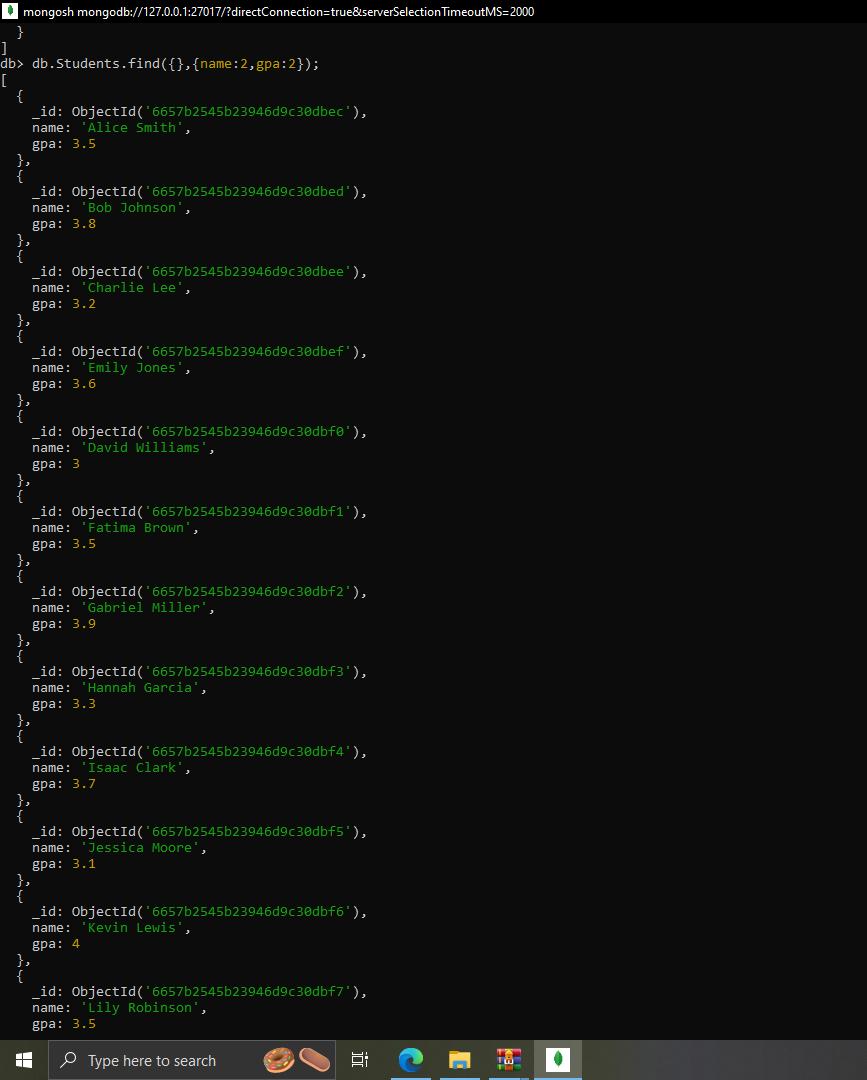
**Projection**

This is used when we don’t need all columns/attributes

Input



Output



Output explanation:

The code in the image is a MongoDB query that retrieves student information from a collection named "Students". Let's break down the code step by step:

Connection:

The first line establishes a connection to the MongoDB server running locally on the machine. The URI mongodb://127.0.0.1:27017 specifies the connection details including the hostname (localhost) and port (27017) which is the default port for MongoDB.

Database:

The line db is prepended to the subsequent commands, indicating that the following operations target a database. However, the database name isn't explicitly mentioned in the provided image. It's likely the shell is currently using a default database.

Find Query:

The line db.Students.find({},{name:2,gpa:2}); is the main query that retrieves data from the "Students" collection. Here's how it works:

db.Students.find(): This part targets the "Students" collection and initiates a find operation to retrieve documents.

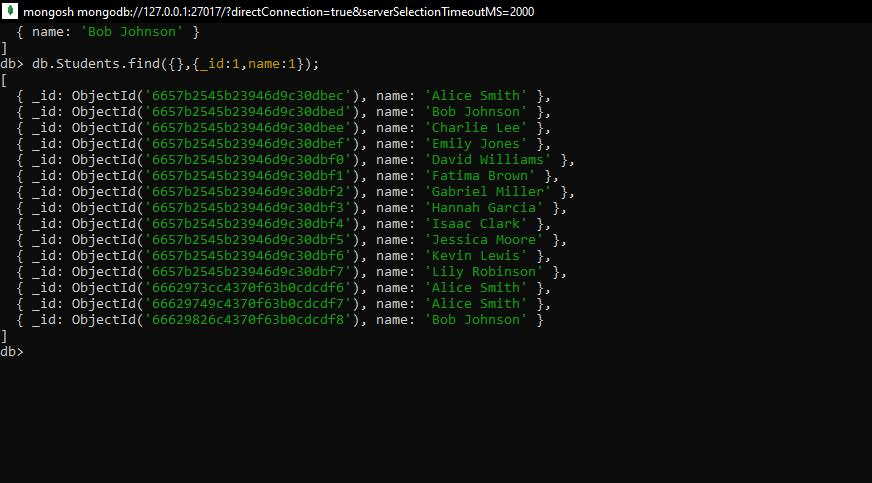
{}: An empty curly brace {} following the find method indicates that all documents in the collection should be returned. You can use filters here to specify conditions for selecting particular documents.

,{name:2,gpa:2}: This part limits the fields returned in the query results. Here, it specifies that only the "name" and "gpa" fields should be included. The number 2 after each field name indicates that these fields should be included. A value of 0 would exclude the field.

Output:

The query results are displayed below the last line. Each document retrieved from the collection is displayed on a new line, with each field and its corresponding value indented. In this case, it shows all the student documents including their name and gpa.

Output:



Output explanation:

Sure, the code in the image you sent is a series of MongoDB shell commands that retrieves and displays student data. Let's break down the code step by step:

Connection

The first line establishes a connection to the MongoDB server running locally on the machine. The URI mongodb://127.0.0.1:27017 specifies the connection details including the hostname (localhost) and port (27017) which is the default port for MongoDB.

Finding Students

The line db.Students.find({}) is the main query that retrieves data from the "Students" collection. Here's how it works:

db.Students.find(): This part targets the "Students" collection and initiates a find operation to retrieve documents.

{}: An empty curly brace {} following the find method indicates that all documents in the collection should be returned. You can use filters here to specify conditions for selecting particular documents.

Incomplete Output

The captured image cuts off part of the output, but it appears to show several student documents. Each document is displayed on a new line, with each field and its corresponding value indented. In a complete output, you would see all the student documents including their names and potentially other fields.