Logical Operators(Selectors)

Where , AND ,OR

WHERE:

In MongoDB, you don't use a specific **"where"** function to filter data. Instead, you leverage query documents to construct filtering criteria within the find method. These query documents express conditions that documents in your collection must meet to be included in the results.

Here's the process:

Specify the Collection: You start by indicating the collection you want to query from using the db.<collection_name> syntax (e.g., db.Students).

Construct the Query Document: The find method takes a query document as its argument. This document defines the filtering conditions using field names and comparison operators.

Comparison Operators: You use various operators like $gt (greater than), $lt (less than), $eq (equal to).

Retrieve Results: Once the query document is constructed, you call the find method, passing the query document as an argument. This retrieves the documents that match the specified criteria from the collection.

Example:

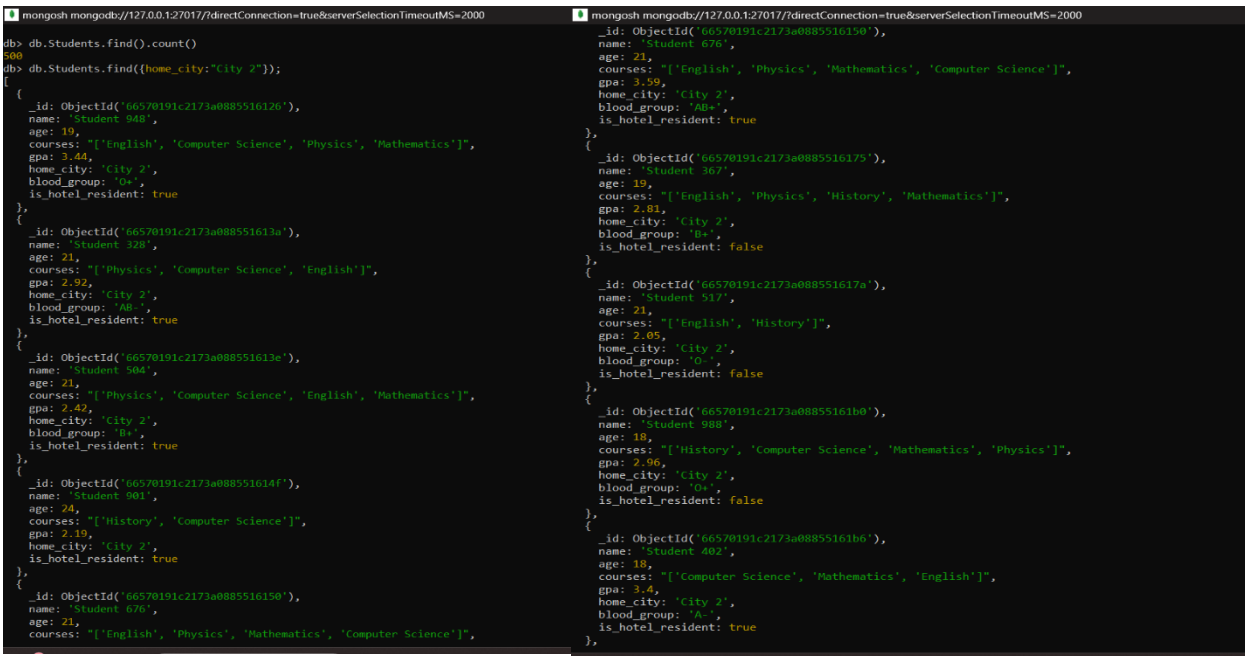
Total number of Students is **"500"**

Firstly, we will be finding the details of students who are from **"City 2"**

QUERY: "db.Students.find({ home_city: "City 2"});

"db.Students.find({ home_city: "City 2"}).count( );

Number of Students from "City 2" are "33".

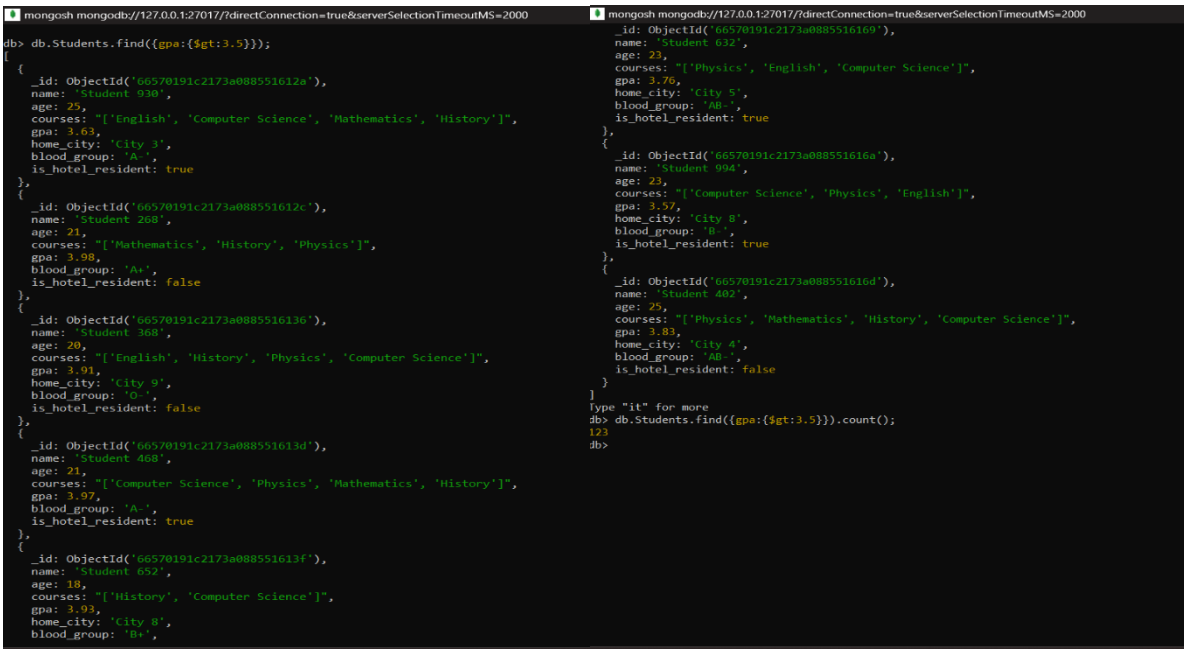


Next, we will be finding the details of students whose **“gpa”** is **“greater than($gt)” - “3.5”**

QUERY: “db.Students.find(gpa:{ $gt: 3.5});

“db.Students.find(gpa:{ $gt: 3.5}).count( );

Number of Students from “City 2” are “123”.



AND:

In MongoDB, the AND functionality isn't achieved through a standalone function like AND(). Instead, it's implemented using the logical operator $and within your query documents.

Here's a detailed explanation of how $and works for combining filtering conditions in MongoDB queries:

- **db.<collection_name>:** Specifies the collection you want to query from.
- **$and:** The logical operator used to combine conditions.
- **[condition1, condition2, ...]:** An array containing individual query document objects representing the filtering criteria. Each object defines conditions for a specific field using comparison operators.

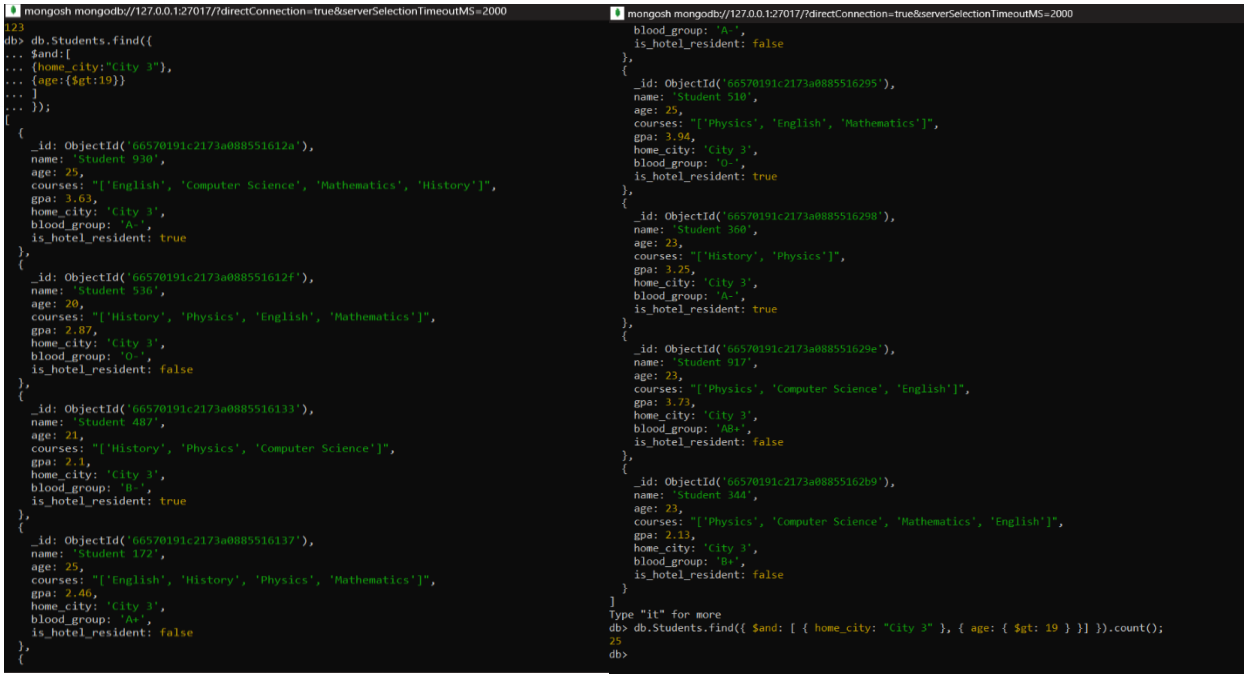
Example:

The $and operator combines two conditions:

{"home_city":"City 3"}: Filters for documents where the home_city field is equal to "City 3".

{"age": { $gt: 19}}: Filters for documents where the age field is greater than 19 using the $gt (greater than) operator.

Only customers who meet both conditions (City 3 resident and over 19) will be returned by the query.



Number of Students from “City 3” and “age greater than 19” are “25”.

OR:

MongoDB doesn't have a separate function named OR. Instead, you leverage the $or logical operator within your query documents to specify alternative filtering conditions.

Using $or for Alternative Conditions

The $or operator allows you to define multiple criteria, where at least one of them must be true for a document to be included in the query results. This is analogous to the logical concept of "OR."

Here's a detailed explanation of how $or works for combining filtering conditions in MongoDB queries:

db.<collection_name>: Specifies the collection you want to query from.

$or: The logical operator for combining alternative conditions.

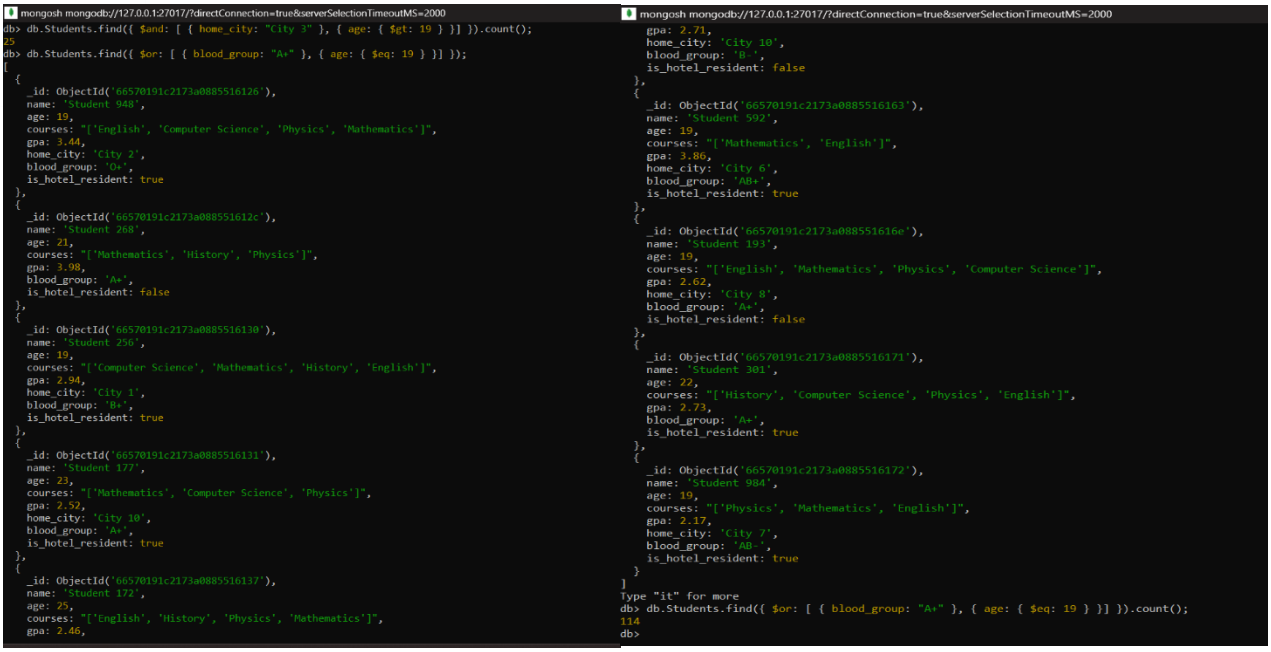
[condition1, condition2, ...]: An array containing individual query document objects representing alternative filtering criteria. Each object defines conditions for specific fields using comparison operators.

Example:

The $or operator combines two conditions:

{"blood_group":"A+"}: Filters for documents where the blood_group field is equal to "A+".

{"age": { $eq:18}}: Filters for documents where the age field is greater than 19 using the $eq (equal to) operator.



Number of Students from “A+” or “age equal 19” are “114”.

Additional Information

Name	Description
$eq	Matches values that are equal to a specified value.
$gt	Matches values that are greater than a specified value.
$gte	Matches values that are greater than or equal to a specified value.
$in	Matches any of the values specified in an array.
$lt	Matches values that are less than a specified value.
$lte	Matches values that are less than or equal to a specified value.
$ne	Matches all values that are not equal to a specified value.
$nin	Matches none of the values specified in an array.