PROJECTION,LIMIT AND SELECTORS:

PROJECTION :

MongoDB projection is a powerful tool that can be used to extract only the fields you need from a document—not all fields. It enables you to:

Project concise and transparent data

Filter the data set without impacting the overall database performance

Because MongoDB projection is built on top of the existing find() method, you can use any projection query without significant modifications to the existing functions. Plus, projection is a key factor when finding user-specific data from a given data set.

Basic Syntax of MongoDB Projection

db.collection_name.find({},{<field> : <value>})

MongoDB projection uses the same find syntax, but we also add a set of parameters to the find function. This set of parameters informs the MongoDB instance of which fields to be returned.

Consider the following collection called “vehicle information”

```
{
        "_id" : ObjectId("5faaf7291139223fe5c19a04"),
        "make" : "Audi",
        "model" : "RS3",
        "year" : 2018,
        "type" : "Sports",
        "reg_no" : "RFD 7866"
}
{
        "_id" : ObjectId("5faaf72f1139223fe5c19a05"),
        "make" : "Ford",
        "model" : "Transit",
        "year" : 2017,
        "type" : "Van",
        "reg_no" : "TST 9880"
}
{
        "_id" : ObjectId("5faaf7371139223fe5c19a06"),
        "make" : "Honda",
        "model" : "Gold Wing",
        "year" : 2018,
        "type" : "Bike",
        "reg_no" : "LKS 2477"
}
mongos>
```

```
mongos> db.vehicleinformation.find().pretty()
{
        "_id" : ObjectId("5faaf5541139223fe5c19a01"),
        "make" : "Nissan",
        "model" : "GTR",
        "year" : 2016,
        "type" : "Sports",
        "reg_no" : "EDS 5578"
}
{
        "_id" : ObjectId("5faaf7131139223fe5c19a02"),
        "make" : "BMW",
        "model" : "X5",
        "year" : 2020,
        "type" : "SUV",
        "reg_no" : "LLS 6899"
}
{
        "_id" : ObjectId("5faaf7221139223fe5c19a03"),
        "make" : "Toyota",
        "model" : "Yaris",
        "year" : 2019,
        "type" : "Compact",
        "reg_no" : "HXE 0153"
}
```

GET SELECTED ATTRIBUTES:

```
test> db.vehicle.find({},{model:1,year:1}).count();
```

IGNORED ATTRIBUTES:

```
test>db.vehicle.find({},{_id:0}).count();
```

RETRIEVING SPECIFIC FIELDS FROM NESTED OBJECTS:

$slice operator is used to retrieve specific field from nested objects.

```
db.findOne(
 {
   _id: "some_id",
   "array1._id": "level_1_id"
 },
 {
   fields: { _id: 0, "array1.$.array2": { $slice: [index, 1] }
 }
})
```

BENEFITS OF PROJECTION:

MongoDB projection is a powerful tool that can be used to extract only the fields you need from a document—not all fields. It enables you to:

Project concise and transparent data

Filter the data set without impacting the overall database performance.

LIMIT:

In MongoDB, the **limit()** method limits the number of records or documents that you want. It basically defines the max limit of records/documents that you want. Or in other words, this method uses on cursor to specify the maximum number of documents/ records the cursor will return. We can use this method after the find() method and find() will give you all the records or documents in the collection. You can also use some conditions inside the find to give you the result that you want.

In this method, we only pass numeric values.

This method is undefined for values which is less than -2^{31} and greater than 2^{31}.

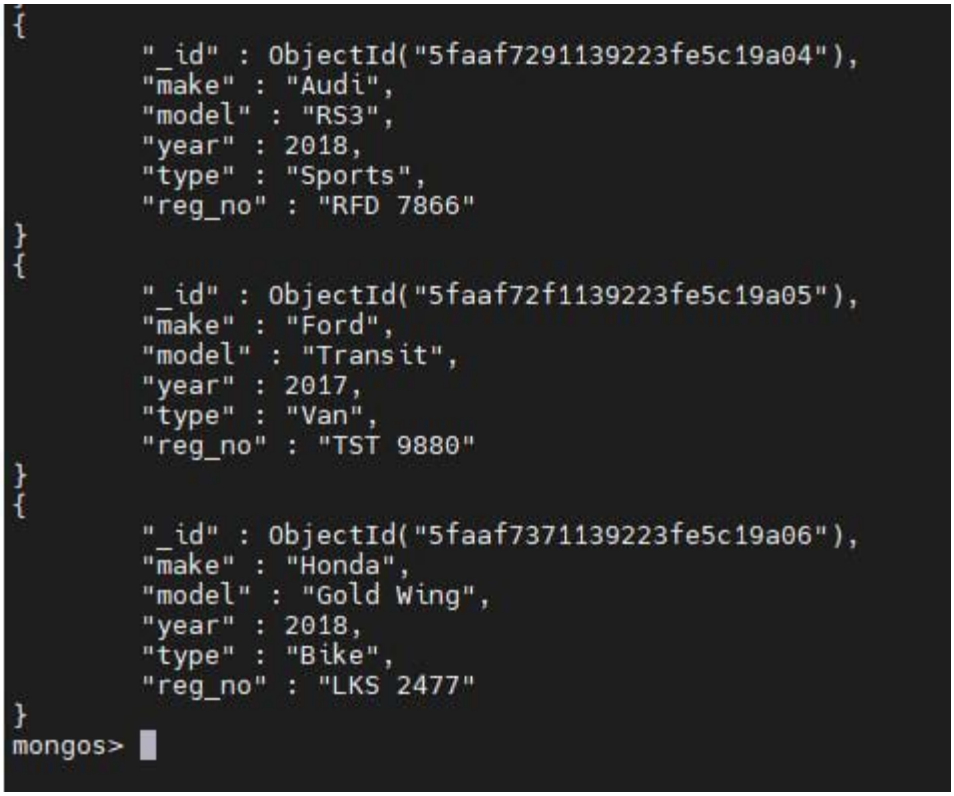
Passing 0 in this method(limit(0)) is equivalent to no limit.

Syntax:

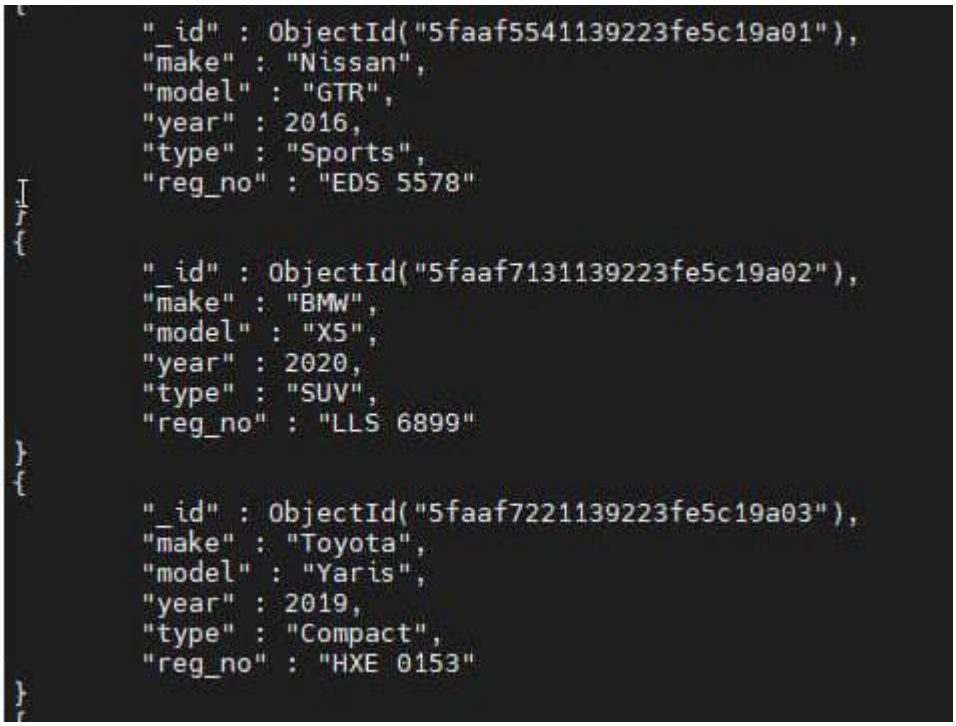
cursor.limit()

$limit takes a number,n,and returns the first n resulting documents:

test>db.vehicle.find({},{_id:0}).limit(3);



TOP 3 RESULTS:



SELECTORS:

COMPARISION:

$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.

OR & AND OPERATOR:

OR: The $or operator performs a logical OR operation on an array of one or more <expressions> and selects the documents that satisfy at least one of the <expressions>.

Syntax:

{ $or: [{ <expression1> }, { <expression2> }, ... , { <expressionN> }] }

AND: $and performs a logical AND operation on an array of one or more expressions (<expression1>, <expression2>, and so on) and selects the documents that satisfy all the expressions.

Syntax:

{ $and: [{ <expression1> }, { <expression2> } , ... , { <expressionN> }] }

BITWISE TYPES:

$bitsAllClear:	Matches numeric or binary values in which a set of bit positions *all* have a value of 0.
$bitsAllSet:	Matches numeric or binary values in which a set of bit positions *all* have a value of 1.
$bitsAnyClear	Matches numeric or binary values in which *any* bit from a set bit positions has a value of 0.
$bitsAnySet:	Matches numeric or binary values in which *any* bit from a set bit positions has a value of 1.

QUERY:

You can query documents in MongoDB by using the following methods:

Your programming language's driver.

1.The MongoDB Atlas UI To learn more, see Query Documents with MongoDB Atlas.

2.MongoDB Compass.