**> db.emp.find().pretty()**

```
{

"_id" : ObjectId("53bba5166a2e2e268459e6c4"),
        "eno" : 1,
        "name" : "sai",
        "age" : 21,
        "dept" : "comp"
}
{
        "_id" : ObjectId("53bba5206a2e2e268459e6c5"),
        "age" : 21,
        "dept" : "comp",
        "eno" : 2,
        "name" : "saiprasd"
}

{
        "_id" : ObjectId("53bba6716a2e2e268459e6c6"),
        "eno" : 3,
        "ename" : {
                "first" : "p",
                "middle" : "sai",
                "last" : "prasad"
        },
        "age" : 28,
        "dob" : "31-12-1985",
        "address" : [
                "vitalwadi",
                "shirdi"
        ]
}
{
        "_id" : ObjectId("53bba6886a2e2e268459e6c7"),
        "eno" : 4,
        "ename" : {
                "first" : "p",
                "middle" : "sai",
                "last" : "prasad"
        },

        "age" : 29,
        "dob" : "3-11-1985",
        "address" : [
                "vitalwadi",
                "shirdi"
]
}
```

## Update:

**systax:db.COLLECTION_NAME.update(SELECTIOIN_CRITERIA, UPDATED_DATA)**

```
> db.emp.find( {eno:2}).pretty()
{
        "_id" : ObjectId("53bba5206a2e2e268459e6c5"),
        "age" : 21,
        "dept" : "comp",
        "eno" : 2,
        "name" : "saiprasd"
```

```
> db.emp.update( {eno:2}, {$set:{name:"krishna"}} )

> db.emp.find( {eno:2}).pretty()
{
        "_id" : ObjectId("53bba5206a2e2e268459e6c5"),
        "age" : 21,
        "dept" : "comp",
        "eno" : 2,
        "name" : "krishna"
}
```

**Updating multiple fields**

```
> db.emp.update( {eno:2}, {$set:{dept:"civil",age:25}},{multi:true} )

> db.emp.find( {eno:2}).pretty()

{
        "_id" : ObjectId("53bba5206a2e2e268459e6c5"),
        "age" : 25,
        "dept" : "civil",
        "eno" : 2,
        "name" : "krishna"
}
```

## Remove

**syntax:**

db.COLLECTION_NAME.remove(DELETION_CRITERIA,1)

db.emp.remove()

It removes all Documents

**> db.emp.remove({eno:1})**

removes Document with eno 1
If there are multiple records and you want to delete only first record, then set justOne parameter in remove() method

**db.emp.remove({eno:2},1)**

## MongoDB Indexing

Indexes support the efficient resolution of queries. Without indexes, MongoDB must scan every document of a collection to select those documents that match the query statement. This scan is highly inefficient and require the mongod to process a large volume of data.

Indexes are special data structures, that store a small portion of the data set in an easy to traverse form. The index stores the value of a specific field or set of fields, ordered by the value of the field as specified in index.

To create an index you need to use **ensureIndex()** method of mongodb.

Basic syntax of ensureIndex() method is as follows()

>**db.COLLECTION_NAME.ensureIndex({KEY:1})**

Here key is the name of filed on which you want to create index and 1 is for ascending order. To create index in
descending order you need to use -1.

**Example**

>db.mycol.ensureIndex({"title":1})
In ensureIndex() method you can pass multiple fields, to create index on multiple fields.

>db.mycol.ensureIndex({"title":1,"description":-1})

## MongoDB Aggregation

Aggregation operations group the values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. In sql count(*) and with group by is an equivalent of mongodb aggregation.
For the aggregation in mongodb you should use **aggregate()** method.

**Syntax:**

Basic syntax of aggregate() method is as follows

>db.COLLECTION_NAME.aggregate(AGGREGATE_OPERATION)

**> db.zipcode.find().pretty()**

```
{ "_id" : "1001", "city" : "nasik", "state" : "MH", "pop" : 50000 }
{ "_id" : "1002", "city" : "mumbai", "state" : "MH", "pop" : 59000 }
{ "_id" : "1004", "city" : "vizag", "state" : "AP", "pop" : 53000 }
{ "_id" : "1005", "city" : "warangal", "state" : "TG", "pop" : 38000 }
{ "_id" : "1006", "city" : "noida", "state" : "ND", "pop" : 78999 }
{ "_id" : "1007", "city" : "banglore", "state" : "KA", "pop" : 62000 }
{ "_id" : "1008", "city" : "Hyderabad", "state" : "TG", "pop" : 55000}
{ "_id" : "1009", "city" : "vizayavada", "state" : "AP", "pop" : 45000 }
```

**Example:1**

**Return Each state population**

**> db.zipcode.aggregate( { $group : { _id : "$state", totalPop : { $sum : "$pop" } } } )**

**Output:**

```
{
        "result" : [
        {
                "_id" : "ND",
                "totalPop" : 78999
        },
        {
                "_id" : "KA",
                "totalPop" : 62000
        },
        {
                "_id" : "TG",
                "totalPop" : 93000
        },
        {
                "_id" : "AP",
                "totalPop" : 98000
        },
        {
                "_id" : "MH",
                "totalPop" : 109000
        }],"ok" : 1 }
```

**Retrun Avg population Of Each state**

```
> db.zipcode.aggregate( { $group : { _id : "$state", AveragePop : { $avg : "$pop" } } } )
```

**Retrun minimum population Of Each state**

```
> db.zipcode.aggregate( { $group : { _id : "$state", MinimumPop : { $min : "$pop" } } } )
```

**Retrun maximum population Of Each state**

```
>db.zipcode.aggregate( { $group : { _id : "$state", MaxPop : { $max : "$pop" } } } )
```

**Return all states with a population greater than 80000**

```
> db.zipcode.aggregate( {
 $group : { _id : "$state", totalPop : { $sum : "$pop" } } },
 { $match : {totalPop : { $gte : 80000 } } }
 )
```

**Output:**

```
{
        "result" : [
                {
                        "_id" : "TG",
                        "totalPop" : 93000
                },
                {
                        "_id" : "AP",
                        "totalPop" : 98000
                },
                {
                        "_id" : "MH",
                        "totalPop" : 109000
                }
        ],
        "ok" : 1
}
```

**To return the smallest and largest cities by population for each state**

```
db.zipcode.aggregate(
{ $group: { _id: { state: "$state", city: "$city" },
                pop: { $sum: "$pop" } } },
                        { $sort: { pop: 1 } },

 { $group: { _id : "$_id.state",
                biggestCity:  { $last: "$_id.city" },
                biggestPop:   { $last: "$pop" },
                smallestCity: { $first: "$_id.city" },
               smallestPop:  { $first: "$pop" } } }
)
```

**Output:**


```
{
"result" : [
        {
                "_id" : "ND",
                "biggestCity" : "noida",
                "biggestPop" : 78999,
        "smallestCity" : "noida",
                "smallestPop" : 78999
        },
        {
                "_id" : "MH",
        "biggestCity" : "mumbai",
                "biggestPop" : 59000,
                "smallestCity" : "nasik",
                "smallestPop" : 50000
    },
        {
                "_id" : "AP",
                "biggestCity" : "vizag",
                "biggestPop" : 53000,
        "smallestCity" : "vizayavada",
                "smallestPop" : 45000
        },
        {
        "_id" : "KA",
                "biggestCity" : "banglore",
                "biggestPop" : 62000,
                "smallestCity" : "banglore",
                "smallestPop" : 62000
        },
        {
                "_id" : "TG",
        "biggestCity" : "Hyderabad",
                "biggestPop" : 55000,

        "smallestCity" : "warangal",
        "smallestPop" : 38000
        }
    ],
    "ok" : 1

}
```


**$first**
:Gets the first document from the source documents according to the grouping.

**$last**
**:**Gets the last document from the source documents

These Two expressions are used with $sort Expression

```
db.zipcode.aggregate( { $group: { _id: { state: "$state", city: "$city" },  pop: { $sum: "$pop" } } },  { $sort: { pop: 1 } })
```

```
{

"result" : [
        {
        "_id" : {
                "state" : "TG",
                "city" : "warangal"
        },
        "pop" : 38000
},
{
        "_id" : {
                "state" : "AP",
                "city" : "vizayavada"
        },
        "pop" : 45000
},
{
        "_id" : {
                "state" : "MH",
                "city" : "nasik"
        },
        "pop" : 50000
},
{
        "_id" : {
                "state" : "AP",
                "city" : "vizag"
        },
        "pop" : 53000
},
{
        "_id" : {
                "state" : "TG",
                "city" : "Hyderabad"
        },
        "pop" : 55000
},
{
                "_id" : {
                "state" : "MH",
        "city" : "mumbai"
},
        "pop" : 59000
},
{
        "_id" : {
                "state" : "KA",
```

```
                        "city" : "banglore"
        },
                "pop" : 62000
        },
        {
                "_id" : {
                        "state" : "ND",
                                "city" : "noida"
                },
                "pop" : 78999
        }
],
        "ok" : 1
}
```