Update document

------ To update document

1. Update ----- it will update one or all matching documents
2. updateOne ---- It will update only first matching document
3. updateMany -----It will update all matching document

--------functions

1. $set ----- to overwrite the value of existing key
2. $unset ------- to remove a key value pair from existing document.
3. $inc ------ used for increase or decrease the value of existing key

SYNTAX----------------------

```
db.collection.update(
   <query>,
   <update>,
   {
     upsert: <boolean>,
     multi: <boolean>,

   }
)
```

Name	Description
$currentDate	Sets the value of a field to current date, either as a Date or a Timestamp.
$inc	Increments the value of the field by the specified amount.
$min	Only updates the field if the specified value is less than the existing field value.
$max	Only updates the field if the specified value is greater than the existing field value.
$mul	Multiplies the value of the field by the specified amount.
$rename	Renames a field.
$set	Sets the value of a field in a document.
$setOnInsert	Sets the value of a field if an update results in an insert of a document. Has no effect on update operations that modify existing documents.
$unset	Removes the specified field from a document.

Array

Operators

Name	Description
$	Acts as a placeholder to update the first element that matches the query

Name	Description
	condition.
$[]	Acts as a placeholder to update all elements in an array for the documents that match the query condition.
$[<identifier>]	Acts as a placeholder to update all elements that match the arrayFilters condition for the documents that match the query condition.
$addToSet	Adds elements to an array only if they do not already exist in the set.
$pop	Removes the first or last item of an array.
$pull	Removes all array elements that match a specified query.
$push	Adds an item to an array.
$pullAll	Removes all matching values from an array.
Name	Description
$each	Modifies the $push and $addToSet operators to append multiple items for array updates.
$position	Modifies the $push operator to specify the position in the array to add elements.
$slice	Modifies the $push operator to limit the size of updated arrays.
$sort	Modifies the $push operator to reorder documents stored in an array.

db.users.insert({ _id: 1, status: "a", lastModified: ISODate("2013-10-02T01:11:18.965Z") })

1. To update one field

```
db.Employee.update(
{"Employeeid" : 1},
{$set: { "EmployeeName" : "NewMartin"}});
```

```
-----change address of employee with name martin
>db.employee.update({empname:"martin"},
                    {$set:{empaddr:"Aundh"}},
                    {upsert:true,
                    multi:true})

------to change rating of movie whose name starts
With k to 4
Db.movie.update({name:/^k/},{$set:{rating:4}},{multi:true})
```

2. To update multiple value

```
db.Employee.updateMany
     (
             {
                     Employeeid : 1
             },
             {
                     $set :
                     {
                     "EmployeeName" : "NewMartin",
                    "Employeeid" : 22
                    }
            },{multi : true}

     )

----update rating of kahani movie collection
to 5
>db.movie.update({name:'kahani'},
                 {
                  $set:{rating:5,'modified.reason':'Public demand'},
                  $currentDate:{lastmodified:true,
                               'modified.time':{$type:'timestamp'}}
                  },
                  {multi:true,upsert:true})
```

3. Users

- {_id:1,status:'D',cancellation{date:ISODate(2018-10-01),reason:'user request}}

```
db.users.update(
  { _id: 1 },
  {
    $currentDate: {
      lastModified: true,
      "cancellation.date": { $type: "timestamp" }
    },
    $set: {
      status: "D",
      "cancellation.reason": "user request"
    }
  }
)
```

--------to change the rating to 5, assign current date to lastmodified key
assign type timestamp to cacellation:{ date: <timestamp>,reason:"user request"}
for movie lagan

```
db.movie.update(
  { name: 'lagaan' },
  {
    $currentDate: {
      lastModified: true,
      "cancellation.date": { $type: "timestamp" }
    },
    $set: {
      rating: 5,
      "cancellation.reason": "user request"
    }
  }
)
-----------------------------------------
```

--------to remove the rating key

>db.movie.update({name:”kahani”},{$unset:{rating:””}},{multi:true,upsert:true})

------ increase the price by 100 for kahani movie

>db.movie.update({name:’kahani’},{$inc:{price:-100}},{multi:true})

--------------------- To use $min

{name:’kahan’,rating:4,price:**350,**…….}

It will compare current price and 200, will keep the smallest

Db.movie.update({name:’kahani’},{$min:{price:200})

----------------to use $max function

{name:’kahan’,rating:4,price:**350,**…….}

It will compare current price and 200, will keep the maximum

Db.movie.update({name:’kahani’},{$max:{price:200})

Inventory examples

```
db.inventory.insertMany( [
   { item: "canvas", qty: 100, size: { h: 28, w: 35.5, uom: "cm" }, status:
"A" },
   { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status:
"A" },
   { item: "mat", qty: 85, size: { h: 27.9, w: 35.5, uom: "cm" }, status:
"A" },
   { item: "mousepad", qty: 25, size: { h: 19, w: 22.85, uom: "cm" },
status: "P" },
   { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status:
"P" },
   { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status:
"D" },
   { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" },
status: "D" },
   { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" },
status: "A" },
   { item: "sketchbook", qty: 80, size: { h: 14, w: 21, uom: "cm" },
status: "A" },
   { item: "sketch pad", qty: 95, size: { h: 22.85, w: 30.5, uom: "cm" },
status: "A" }
] );
```

Using updateone

```
db.inventory.updateOne(
   { item: "paper" },
   {
     $set: { "size.uom": "cm", status: "P" },
     $currentDate: { lastModified: true }
   }
)
-----update rating of all movies to 5 if
    the price > 300
```

```
>db.movie.update({price:{$gt:300}},{$set:{rating:5}},{upsert:true,multi:tru
e})
```

```
Using updatemany
db.inventory.updateMany(
   { "qty": { $lt: 50 } },
   {
     $set: { "size.uom": "in", status: "P" },
     $currentDate: { lastModified: true }
   }
)
```

To add data in the array
$push
$pop
$
$[]
$pull

db.movie.update({name:'padmavat'},{$push:{actor:"raza murad"}})

db.movie.update({name:'padmavat'},{$push:{actor:{$each:["raza murad",”aditi rao”]}}})

---

student : {_id:1,name:”revati”,hobbies:[“reading”,”swimming”]}
db.student.update({name:”revati”},{$push:{hobbies:{$each:[“drawing”,”riding”,”reading novels”],$position:0}}},{multi:true})

------will add at the beginning because given position is 0. $position should be used with $each function db.movie.update({name:'padmavat'},{$push:{actors:{$each:["raza murad",”aditi rao”],$position: 0}}})

-------write query to add grade (“B”,21-06-2018,89) object in grades array for all documents for cuisine is America or Chinese

>db.restaurants.update({“cuisine”:{$in:[“America”,”Chinese”]}},{
$push:{grades:”{grade:”A”,score:”89”,date:ISODate(“2018-06-21”)}”}},{multi:true})
>db.restarants.update({“cuisine”:{$in:[“America”,”Chinese”]},
{ $push:{grades:{$each:”[{ {grade:”A”,score:”89”,date:ISODate(“2018-06-21”)},
{grade:”A+”,score:”99”,date:ISODate(“2018-08-21”)}]”,$position:2} ,{}]}}
},{multi:true,upsert:true})

db.movie.update({name:”kahani”},{$push:{ actors:{ $each:[“aaaa”,”bbbb”],$position:0 }}},{multi:true})

```
db.movie.updateOne(
   { _id: 1, actor: ‘raza murad’ },
   { $set: { "actor.$" : ‘xxx’ } }
)
```

```
{
  _id: 4,
  grades: [
     { grade: 85, mean: 75, std: 8 },
     { grade: 80, mean: 90, std: 5 },
     { grade: 85, mean: 85, std: 8 }
  ]
}
```

$ indicates the matched record

```
db.students.updateOne(
  { _id: 4, "grades.grade": 85 },
  { $set: { "grades.$.std" : 6 } }
)
```

Coordinate:[34.5555,35.666]

{coordinate.0:12.0000}

---- increase the salary by 10000 for all employees.
>db.employee.update({},{$inc:{sal:10000}})

>db.movie.update({name:”padmavat”},{$inc:{rating:-2}})

$addToSet will add element if it is not there
otherwise no operation will happen

db.movie.update({actor:'cccccc',name:'kahani 2'},{$addToSet:{'actor':'cccccc'}})

$[] – all the values in the array $inc ---increament values

```
{ "_id" : 1, "grades" : [ 85, 82, 80 ] }
{ "_id" : 2, "grades" : [ 88, 90, 92 ] }
{ "_id" : 3, "grades" : [ 85, 100, 90 ] }
```

To increase all the values in grade array by 10

```
db.students.update(
   { },
   { $inc: { "grades.$[]": 10 } },
   { multi: true }
)

------write a query to increase only 85 values by 5
for all documents
>db.students.update({grades:85},{$inc:{“grades.$”:5}},{multi:true})



db.students.update(
   { grades:85},
   { $inc: { "grades.$": 10 } },
   { multi: true }
)
```

```
{
   "_id" : 1,
   "grades" : [
      { "grade" : 80, "mean" : 75, "std" : 8 },
      { "grade" : 85, "mean" : 90, "std" : 6 },
      { "grade" : 85, "mean" : 85, "std" : 8 }
   ]
}
{
   "_id" : 2,
   "grades" : [
      { "grade" : 90, "mean" : 75, "std" : 8 },
      { "grade" : 87, "mean" : 90, "std" : 5 },
      { "grade" : 85, "mean" : 85, "std" : 6 }
   ]
}
```

To decrease all values of std

```
db.students2.update(
   { },
   { $inc: { "grades.$[].std" : -2 } },
   { multi: true }
)
```

$pop – delete last element

```
db.students.update( { _id: 1 }, { $pop: { scores: -1 } } )
```

To remove 1 st element specify -1 and use 1 for deleting last element

```
db.students.update( { _id: 1 }, { $pop: { scores: -1 } } )
```

---- delete last value of actors array for movie kahani

```
>db.movie.update({name:"Kahani"},{$pop:{actor:1}},{multi:true})
```

```
The $pull operator removes from an existing array all instances of a value
or values that match a specified condition.

Remove all matching values
Removes apple and oranges from fruits and carrots from vegetables
remov
db.stores.update(
    { },
    { $pull: { fruits: { $in: [ "apples", "oranges" ] }, vegetables:
"carrots" } },
    { multi: true }
)


>db.movie.update({name:/^s/},{$pull:{actor:["vidya balan"]},{multi:true})
```

------ write a query to delete last object of grades array from all documents

>db.restaurants.update({},{$pop:{grades:1},{multi:true})

------to create index

```
Db.movie.ensureIndex({rating:1,price:-1})---deprecated
Db.movie.createIndex({rating:1,price:-1})
Db.movie.getIndexes()
Db.move.dropIndex('rating_2')
```

To create index on rating if it does not exists

```
rating:1 --- ascending     rating:-1   -----descending
db.movie.ensureIndex({rating:1})

To create composit index
db.movie.ensureIndex({rating:1,name:-1})



db.movie.createIndex({rating:1,name:-1})
types of indexes
   1. Simple index-→ single key index
   2. Compond index -→ multiple key index
   3. Multikey index -→ if it is on nested key
   4. Geospatial -→ used to find locations using latitude and longitude
   5. Full text -→ used to search huge data in a key



db.emp.ensureIndex({sal:1})—→it is deprecated


---All indexes are stored in system.indexes collections
----to delete index
Db.movie.dropIndex(name of index)

To view indexes
db.movie.getIndexes()
```

```
To remove documents
db.movie.remove()   ------remove all documents
db.movie.remove(criteria)   ----remove documents that match the criteria

-----delete all documents from employee collection whose
salary is < 10000

db.emp.remove({sal:{$lt:10000}})—delete all matching documents
db.movie.deleteOne({criteria})----delete the first matching document
db.movie.deleteMany({criteria}) --- delete all matching documents
db.movie.deleteMany({})----- delete all the documents
db.emp.remove({})----- delete all the documents
```
