

- 1) Create a Database called student
- 2) Create a collection called studentmarks

The screenshot shows a terminal window on the left with the following commands:

```
> use student
switched to db student
> db.createCollection("studentmarks")
{ "ok" : 1 }
> show collections
studentmarks
> use studentmarks
```

The main window displays a web browser with the title "MongoDB Exercise 2". It contains a table with student marks:

| name | maths_marks | english_marks | science_marks |
|---------|-------------|---------------|---------------|
| Mala | 45 | 53 | 72 |
| Vanu | 80 | 75 | 85 |
| Kala | 32 | 46 | 53 |
| Aruli | 78 | 85 | 80 |
| Shayu | 80 | 76 | 65 |
| Kumaran | 32 | 73 | 84 |
| Lucky | 66 | 90 | 45 |
| Gva | 71 | 75 | 56 |
| Raam | 41 | 65 | 88 |

Below the table, there is a list of exercises:

- 1) Create a Database called **student**
- 2) Create a collection called **studentmarks**
- 3) Create the documents listed in above table.
- 4) Increase the maths marks of Mala by 6 marks
- 5) List the names of students who got more than 50 marks in Maths Subject.
- 6) Add a new column(field) for Average for all students.
- 7) Update Marks_Science=75 to Lucky .
- 8) List the names who got more than 50 marks in all subjects.
- 9) List the names who got less than 50 marks in Maths subject and more than 50 marks in English

- 3) Create the documents listed in above table.

The screenshot shows a terminal window on the left with the following commands:

```
> use studentmarks
switched to db studentmarks
> db.insert({"name": "Aruli", "maths_marks": 78, "english_marks": 85, "science_marks": 80})
{ "insertedId" : "5e16bec8d399782711337c6f", "n": 1 }
> db.insert({"name": "Shayu", "maths_marks": 80, "english_marks": 76, "science_marks": 65})
{ "insertedId" : "5e16bef0d399782711337c70", "n": 1 }
> db.insert({"name": "Kumaran", "maths_marks": 32, "english_marks": 73, "science_marks": 84})
{ "insertedId" : "5e16bf0ad399782711337c71", "n": 1 }
> db.insert({"name": "Lucky", "maths_marks": 66, "english_marks": 90, "science_marks": 45})
{ "insertedId" : "5e16bf2fd399782711337c72", "n": 1 }
```

The main window displays a web browser with the title "MongoDB Exercise 2". It contains a table with student marks:

| name | maths_marks | english_marks | science_marks |
|---------|-------------|---------------|---------------|
| Shayu | 80 | 76 | 65 |
| Kumaran | 32 | 73 | 84 |
| Lucky | 66 | 90 | 45 |
| Gva | 71 | 75 | 56 |
| Raam | 41 | 65 | 88 |

Below the table, there is a list of exercises:

- 1) Create a Database called **student**
- 2) Create a collection called **studentmarks**
- 3) Create the documents listed in above table.
- 4) Increase the maths marks of Mala by 6 marks
- 5) List the names of students who got more than 50 marks in Maths Subject.
- 6) Add a new column(field) for Average for all students.
- 7) Update Marks_Science=75 to Lucky .
- 8) List the names who got more than 50 marks in all subjects.
- 9) List the names who got less than 50 marks in Maths subject and more than 50 marks in English
- 10) List the names who got less than 40 in both Maths and Science.
- 11) Remove Science column/field for Raam
- 12) Update John's Math mark as 87 and English mark as 23, if john not available upsert.
- 13) Rename the english_marks column/field for John to science_marks
- 14) Remove Kumaran's document from collection
- 15) Find Kala's or Aruli's math_marks and science_marks

- 4) Increase the maths marks of Mala by 6 marks

ukistu21@ukistu21-Latitude-E5420m: ~

```
> db.studentmarks.aggregate([{$addFields:{"average":{}}]).pretty()
```

```
{
  "_id" : ObjectId("5e16be8d399782711337c6c"),
  "name" : "Mal a",
  "maths_marks" : 51,
  "english_marks" : 53,
  "science_marks" : 72,
  "average" : ""
}
```

Answers

- Insert a node as a child before an existing child in JavaScript?
- Insert a specified HTML tag into a document?
- How to insert a date when logging exceptions to a file with JavaScript?
- Can we make static reference in non-static class?
- How to display the date and time of a document when it is last modified in JavaScript?

```
{
  "_id" : ObjectId("5e16be8d399782711337c6c"),
  "name" : "Kal a",
  "maths_marks" : 32,
  "english_marks" : 46,
  "science_marks" : 53,
  "average" : ""
}
```

You can achieve this with the help of \$addFields operator. To understand the concept, let us create a collection with the document. The query to create a collection with a document is as follows –

```
> db.addFieldDemo.insertOne({ "EmployeeId":101,"EmployeeName":"Larry","EmployeeDetails":{
  "EmployeeSalary":65000,"EmployeeCity":"New York","Message":"Hi"} });
```

```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5c7f654d8d10a061296a3c44")
}
```

Display all documents from a collection with the help of find() method. The query is as follows –

```
> db.addFieldDemo.find().pretty();
```

The following is the output –

```
{
  "_id" : ObjectId("5c7f654d8d10a061296a3c44"),
  "EmployeeId" : 101,
  "EmployeeName" : "Larry",
  "EmployeeDetails" : {
    "EmployeeSalary" : 65000,
    "EmployeeCity" : "New York",
    "Message" : "Hi"
  }
}
```

Here is the query to include all existing fields and add new fields to document in MongoDB –

```
> db.addFieldDemo.updateOne({ "_id" : ObjectId("5e16be8d399782711337c6c") },
```

7) Update Marks_Science=75 to Lucky .

ukistu21@ukistu21-Latitude-E5420m: ~

```
> db.studentmarks.update({'science_marks':66},{ $set: {'science_marks': 75}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
```

```
> db.studentmarks.update({'science_marks':45},{ $set: {'science_marks': 75}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Outline

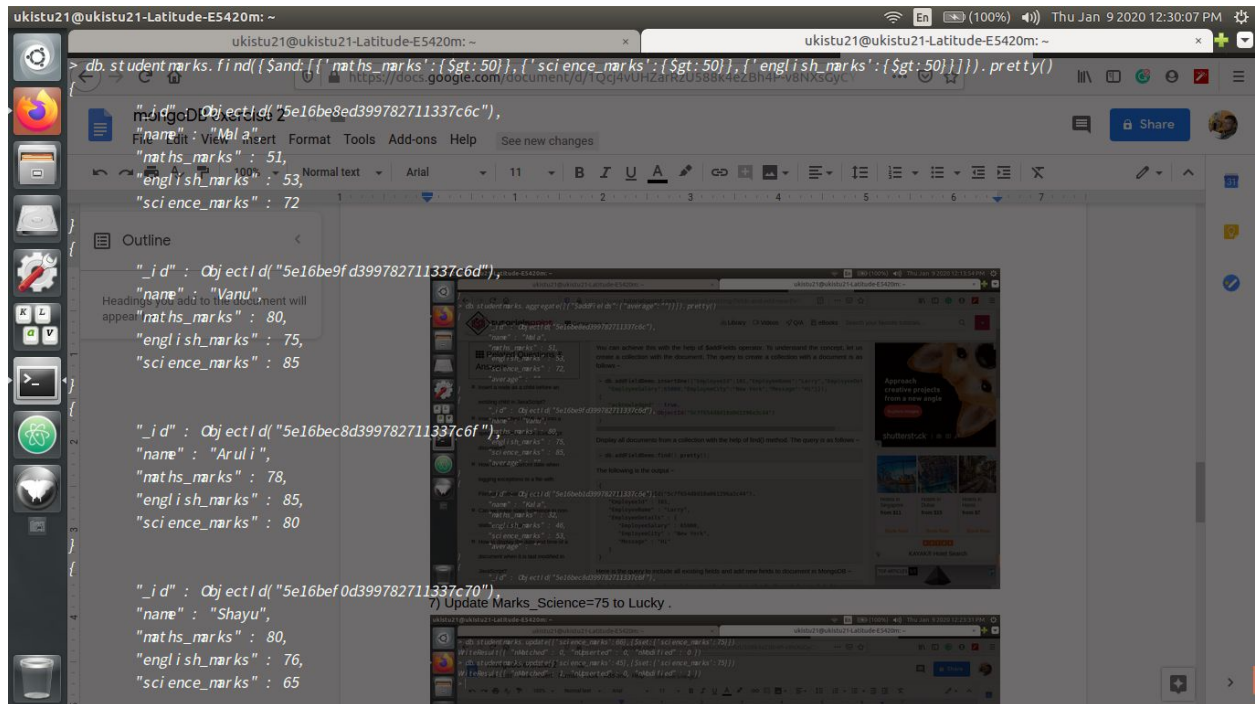
Headings you add to the document will appear here.

7) Update Marks_Science=75 to Lucky .

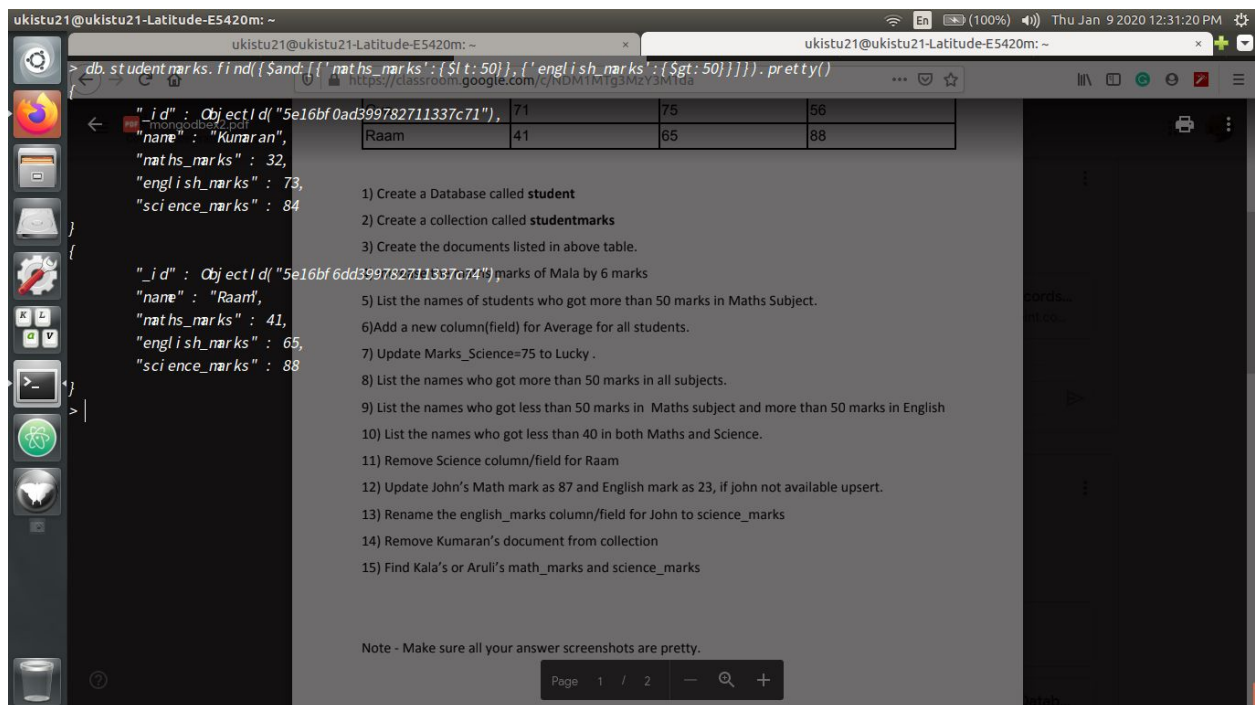
8) List the names who got more than 50 marks in all subjects.

9) List the names who got less than 50 marks in Maths subject and more than 50 marks in English

8) List the names who got more than 50 marks in all subjects.



9) List the names who got less than 50 marks in Maths subject and more than 50 marks in English



10) List the names who got less than 40 in both Maths and Science.

ukistu21@ukistu21-Latitude-E5420m: ~

ukistu21@ukistu21-Latitude-E5420m: ~

db.student_marks.find({'maths_marks':{'\$lt':40}}, {'science_marks':{'\$lt':40}}).pretty()

https://docs.google.com/document/d/1Q34UHZaHnU58dHnZ6H1Pv8KXdyCv

mongoDB exercise 2

File Edit View Insert Format Tools Add-ons Help See new changes

100% Normal text Arial 11 B I U A

Outline

Headings you add to the document will appear here.

10) List the names who got less than 40 in both Maths and Science.

11) Remove Science column/field for Raam

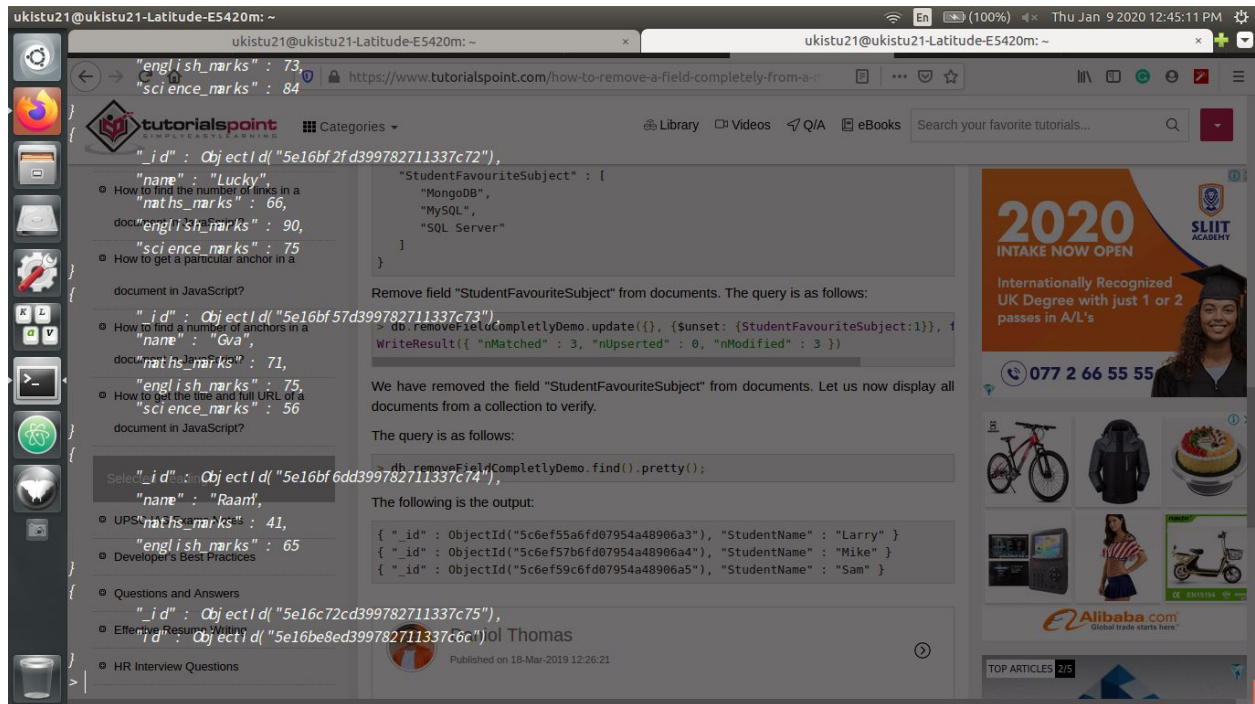
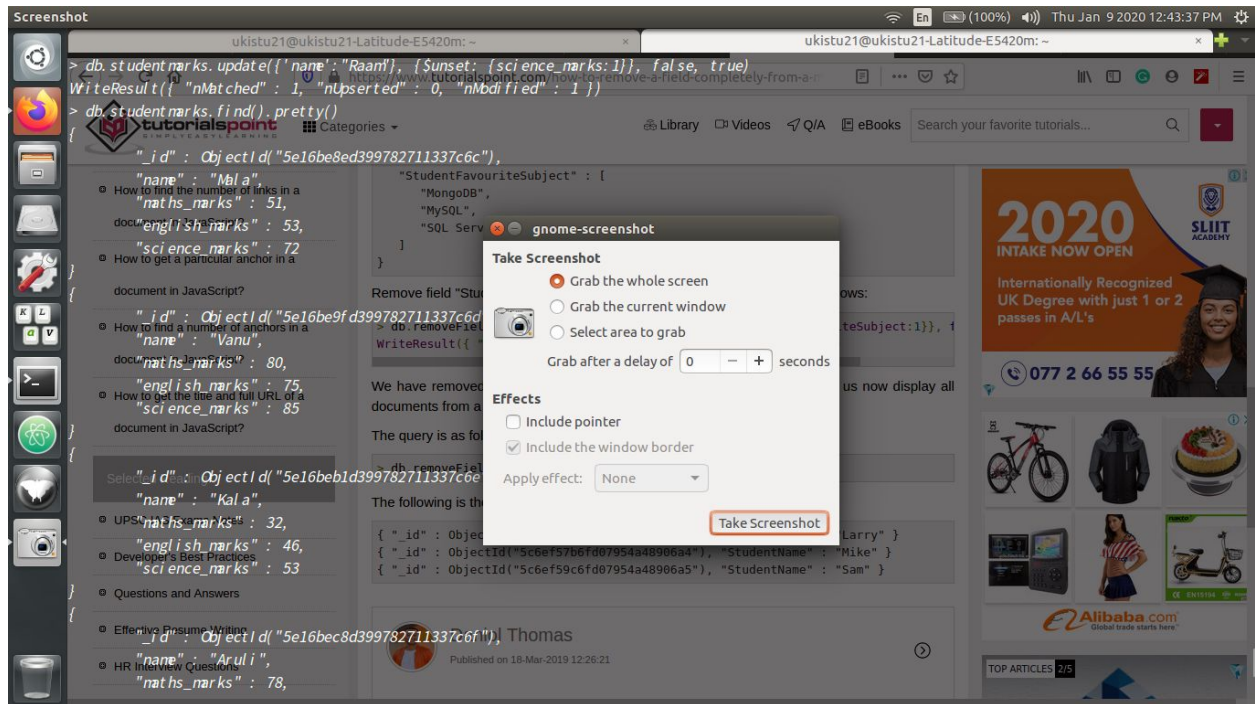
12) Update John's Math mark as 87 and English mark as 23, if john not available upsert.

13) Rename the english_marks column/field for John to science_marks

14) Remove Kumaran's document from collection

Note: Make sure all your answers are correct.

11) Remove Science column/field for Raam



12) Update John's Math mark as 87 and English mark as 23, if john not available upsert.

ukistu21@ukistu21-Latitude-E5420m: ~

```
> db.student_marks.insert({name: "John", maths_marks: 87, english_marks: 23}, {upsert: true})
WriteResult({
  "nInserted": 1
})
^[[A^[[A^[[BWriteResult({
  "nInserted": 1
})
^[[db.student_marks.find().pretty()
```

JournalDev

This operation first searches for the document if not present then inserts the new document into the database.

```
> db.car.update(
  ... { name: "Qualis" },
  ... {
    name: "Qualis",
    speed: 50,
    { upsert: true }
  }
)
WriteResult({
  "nMatched": 0,
  "nUpserted": 1,
  "nModified": 0,
  "id": ObjectId("548d3a955a5072e76925dc1c")
})
```

The car with the name Qualis is checked for existence and if not, a document with car name "Qualis" and speed 50 is inserted into the database. The nUpserted with value "1" indicates a new document is inserted.

Note that to avoid inserting the same document more than once, create an index on the name field thereby ensuring that the document will be inserted only once for the upsert option on every update specified. If the upsert fails because of duplicate index key error, retrying results in the successful update operation.

Click here for premium quality foods

Hotels in Singapore from \$11

Hotels in Dubai from \$15

ukistu21@ukistu21-Latitude-E5420m: ~

```
> db.student_marks.insert({name: "John", maths_marks: 87, english_marks: 23}, {upsert: true})
WriteResult({
  "nInserted": 1
})
^[[A^[[A^[[BWriteResult({
  "nInserted": 1
})
^[[db.student_marks.find().pretty()
```

JournalDev

This operation first searches for the document if not present then inserts the new document into the database.

```
> db.car.update(
  ... { name: "Qualis" },
  ... {
    name: "Qualis",
    speed: 50,
    { upsert: true }
  }
)
WriteResult({
  "nMatched": 0,
  "nUpserted": 1,
  "nModified": 0,
  "id": ObjectId("548d3a955a5072e76925dc1c")
})
```

The car with the name Qualis is checked for existence and if not, a document with car name "Qualis" and speed 50 is inserted into the database. The nUpserted with value "1" indicates a new document is inserted.

Note that to avoid inserting the same document more than once, create an index on the name field thereby ensuring that the document will be inserted only once for the upsert option on every update specified. If the upsert fails because of duplicate index key error, retrying results in the successful update operation.

Click here for premium quality foods

Hotels in Singapore from \$11

Hotels in Dubai from \$15

13) Rename the english_marks column/field for John to science_marks

Terminal Window: ukistu21@ukistu21-Latitude-E5420m: ~

```
> db.studentMarks.update({name: "John"}, {$rename: {"english_marks": "science_marks"}}, false, true)
WriteResult({ "nMatched": 1, "nUpserted": 0, "nModified": 1 })
> db.studentMarks.find().pretty()
```

Display all documents from a collection with the help of find() method. The query is as follows –

```
MySQL> db.studentMarks.find().pretty();
```

The following is the output –

```
{ "_id" : ObjectId("5e16be8ed399782711337c6c"),
  "name" : "Mbl a",
  "maths_marks" : 51,
  "english_marks" : 53 }
{ "_id" : ObjectId("5e16be9fd399782711337c6d"),
  "name" : "Vanu",
  "maths_marks" : 80,
  "english_marks" : 75 }
{ "_id" : ObjectId("5e16beb1d399782711337c6e"),
  "name" : "Kola",
  "maths_marks" : 32,
  "english_marks" : 46 }
{ "_id" : ObjectId("5e16bec8d399782711337c6f"),
  "name" : "Aruti",
  "maths_marks" : 78,
  "english_marks" : 85 }
```

Take Screenshot dialog box:

- Grab the whole screen
- Grab the current window
- Select area to grab
- Grab after a delay of 0 seconds
- Effects:
 - ☐ Include pointer
 - ☒ Include the window border
- Apply effect: None
- Take Screenshot

Terminal Window: ukistu21@ukistu21-Latitude-E5420m: ~

```
> db.studentMarks.update({name: "John"}, {$rename: {"english_marks": "science_marks"}}, false, true)
WriteResult({ "nMatched": 1, "nUpserted": 0, "nModified": 1 })
> db.studentMarks.find().pretty()
```

Display all documents from a collection with the help of find() method. The query is as follows –

```
MySQL> db.studentMarks.find().pretty();
```

The following is the output –

```
{ "_id" : ObjectId("5e16bf2fd399782711337c72"),
  "name" : "Lucky",
  "maths_marks" : 66,
  "english_marks" : 90 }
{ "_id" : ObjectId("5e16bf57d399782711337c73"),
  "name" : "Gya",
  "maths_marks" : 71,
  "english_marks" : 75 }
{ "_id" : ObjectId("5e16bf6dd399782711337c74"),
  "name" : "Raam",
  "maths_marks" : 41,
  "english_marks" : 65 }
{ "_id" : ObjectId("5e16c72cd399782711337c75"),
  "name" : "John",
  "maths_marks" : 87,
  "science_marks" : 23 }
```

Here is the query to rename field "StudentName" to "StudentFirstName" for all documents –

```
> db.renameFieldDemo.update({}, {$rename: {"StudentName": "StudentFirstName"}}, false, true);
WriteResult({ "nMatched": 5, "nUpserted": 0, "nModified": 5 })
```

Let us check all the documents from a collection. The query is as follows

```
> db.renameFieldDemo.find().pretty();
```

The following is the output –

```
{ "_id" : ObjectId("5c7ee6c7559dd2396cbfbfb"), "StudentFirstName" : "John" }
{ "_id" : ObjectId("5c7ee6cb559dd2396cbfbfb"), "StudentFirstName" : "Carol" }
{ "_id" : ObjectId("5c7ee6cf559dd2396cbfbfb"), "StudentFirstName" : "Bob" }
{ "_id" : ObjectId("5c7ee6d8559dd2396cbfbfb"), "StudentFirstName" : "David" }
{ "_id" : ObjectId("5c7ee6f559dd2396cbfbfb"), "StudentFirstName" : "Maxwell" }
```

14) Remove Kumaran's document from collection

ukistu21@ukistu21-Latitude-E5420m: ~

ukistu21@ukistu21-Latitude-E5420m: ~

https://www.tutorialspoint.com/mongodb/mongodb_delete_document.htm

Categories

Library Videos Q/A eBooks rename key mongodb

```
> db.studentmarks.remove({"name": "Kunaran"})
WriteResult({"nRemoved": 1})
> db.studentmarks.find().pretty()
{
  "name": "Kunaran",
  "maths_marks": 51,
  "english_marks": 53
}
{
  "name": "Aruli",
  "maths_marks": 78,
  "english_marks": 85
}
```

Example

Consider the mycol collection has the following data.

```
{ "_id" : ObjectId("5983548781331adf45ec5"), "title": "MongoDB Overview" }
{ "_id" : ObjectId("5983548781331adf45ec6"), "title": "NoSQL Overview" }
{ "_id" : ObjectId("5983548781331adf45ec7"), "title": "Tutorials Point Overview" }
```

Following example will remove all the documents whose title is 'MongoDB Overview'.

```
> db.mycol.remove({'title': 'MongoDB Overview'})
> db.mycol.find()
{ "_id" : ObjectId("5983548781331adf45ec6"), "title": "NoSQL Overview" }
{ "_id" : ObjectId("5983548781331adf45ec7"), "title": "Tutorials Point Overview" }
```

Remove Only One

If there are multiple records and you want to delete only the first record, then set **justOne** parameter in **remove()** method.

```
> db.COLLECTION_NAME.remove(DELETION_CRITERIA, 1)
```

Remove All Documents

If you don't specify deletion criteria, then MongoDB will delete whole documents from the collection. **This is equivalent of SQL's truncate command.**

```
> db.mycol.remove({})
```

Hotels in Singapore from \$11

Hotels in Hanoi from \$7

Hotels in Dubai from \$15

Book Now

KAYAK

Book early, save more

one world

QATAR AIRWAYS

TOP ARTICLES 1/5

15) Find Kala's or Aruli's math_marks and science_marks