

# MONGODB

## Basic Questions

Dataset – Students

```
"_id" : 1,  
"std_name" : "Mukesh",  
"Gender" : "Male",  
"class" : "VI",  
"age" : 11,  
"grd_point" : 33  
"_id" : 2,  
"std_name" : "Dechamma",  
"Gender" : "Female",  
"class" : "VI",  
"age" : 13,  
"grd_point" : 30  
"_id" : 3,  
"std_name" : "Akash",  
"Gender" : "Male",  
"class" : "V",  
"age" : 14,  
"grd_point" : 35.1257  
"_id" : 4,  
"std_name" : "Geetha",  
"Gender" : "Female",  
"class" : "X",  
"age" : 17,
```

"grd\_point" : 36.2514

"\_id" : 5,

"std\_name" : "Bhomika",

"Gender" : "Female",

"class" : "X",

"age" : 19,

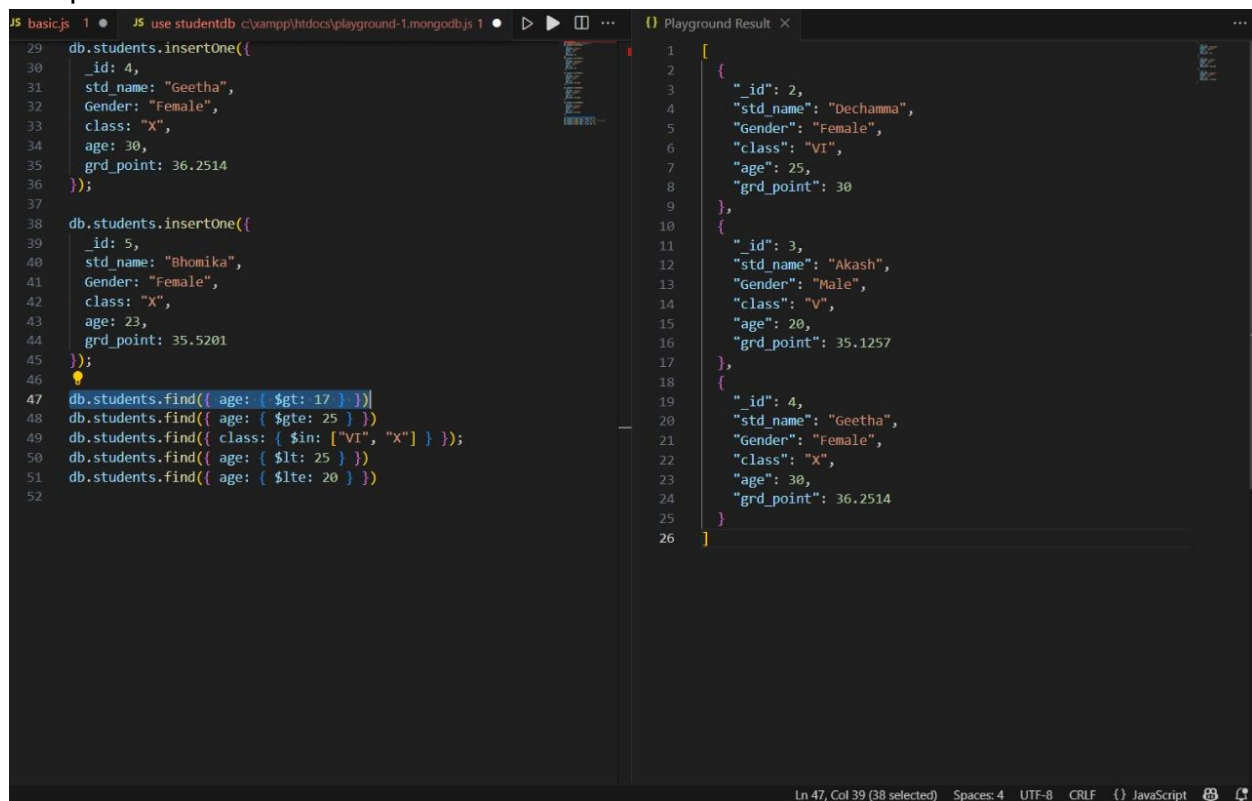
"grd\_point" : 35.5201

## >Using Comparison Operator perform the following questions

### 1.Display Student Details Where Age>17

```
db.students.find({ age: { $gt: 17 } })
```

Output:



The screenshot shows a MongoDB Playground interface. On the left, a JavaScript script is written in a text editor. It starts with two `insertOne` calls to add documents to the `students` collection. The first document has `_id: 4`, `std_name: "Geetha"`, `Gender: "Female"`, `class: "X"`, `age: 30`, and `grd_point: 36.2514`. The second document has `_id: 5`, `std_name: "Bhomika"`, `Gender: "Female"`, `class: "X"`, `age: 23`, and `grd_point: 35.5201`. Below these, a `find` query is executed: `db.students.find({ age: { $gt: 17 } })`. On the right, the 'Playground Result' pane shows the output of this query as a JSON array. It contains two objects: the first for Geetha (age 30, grd\_point 36.2514) and the second for Bhomika (age 23, grd\_point 35.5201). The status bar at the bottom indicates 'Ln 47, Col 39 (38 selected) Spaces: 4 UTF-8 CRLF {} JavaScript'.

```
29 db.students.insertOne({
30   _id: 4,
31   std_name: "Geetha",
32   Gender: "Female",
33   class: "X",
34   age: 30,
35   grd_point: 36.2514
36 });
37
38 db.students.insertOne({
39   _id: 5,
40   std_name: "Bhomika",
41   Gender: "Female",
42   class: "X",
43   age: 23,
44   grd_point: 35.5201
45 });
46
47 db.students.find({ age: { $gt: 17 } })
48 db.students.find({ age: { $gte: 25 } })
49 db.students.find({ class: { $in: ["VI", "X"] } })
50 db.students.find({ age: { $lt: 25 } })
51 db.students.find({ age: { $lte: 20 } })
52
```

```
1 [
2   {
3     "_id": 2,
4     "std_name": "Dechamma",
5     "Gender": "Female",
6     "class": "VI",
7     "age": 25,
8     "grd_point": 30
9   },
10  {
11    "_id": 3,
12    "std_name": "Akash",
13    "Gender": "Male",
14    "class": "V",
15    "age": 20,
16    "grd_point": 35.1257
17  }
18 ],
19 {
20   "_id": 4,
21   "std_name": "Geetha",
22   "Gender": "Female",
23   "class": "X",
24   "age": 30,
25   "grd_point": 36.2514
26 }
```

### 2.Display Student Details Where Age>=25

```
db.students.find({ age: { $gte: 25 } })
```

Output:

```
JS basic.js 1 • JS use studentdb c:\xampp\htdocs\playground-1.mongodbs.js 1 • Playground Result × ...
29 db.students.insertOne({
30   _id: 4,
31   std_name: "Geetha",
32   Gender: "Female",
33   class: "X",
34   age: 30,
35   grd_point: 36.2514
36 });
37
38 db.students.insertOne({
39   _id: 5,
40   std_name: "Bhomika",
41   Gender: "Female",
42   class: "X",
43   age: 23,
44   grd_point: 35.5201
45 });
46
47 db.students.find({ age: { $gt: 17 } })
48 db.students.find({ age: { $gte: 25 } })
49 db.students.find({ class: { $in: ["VI", "X"] } });
50 db.students.find({ age: { $lt: 25 } })
51 db.students.find({ age: { $lte: 20 } })
52

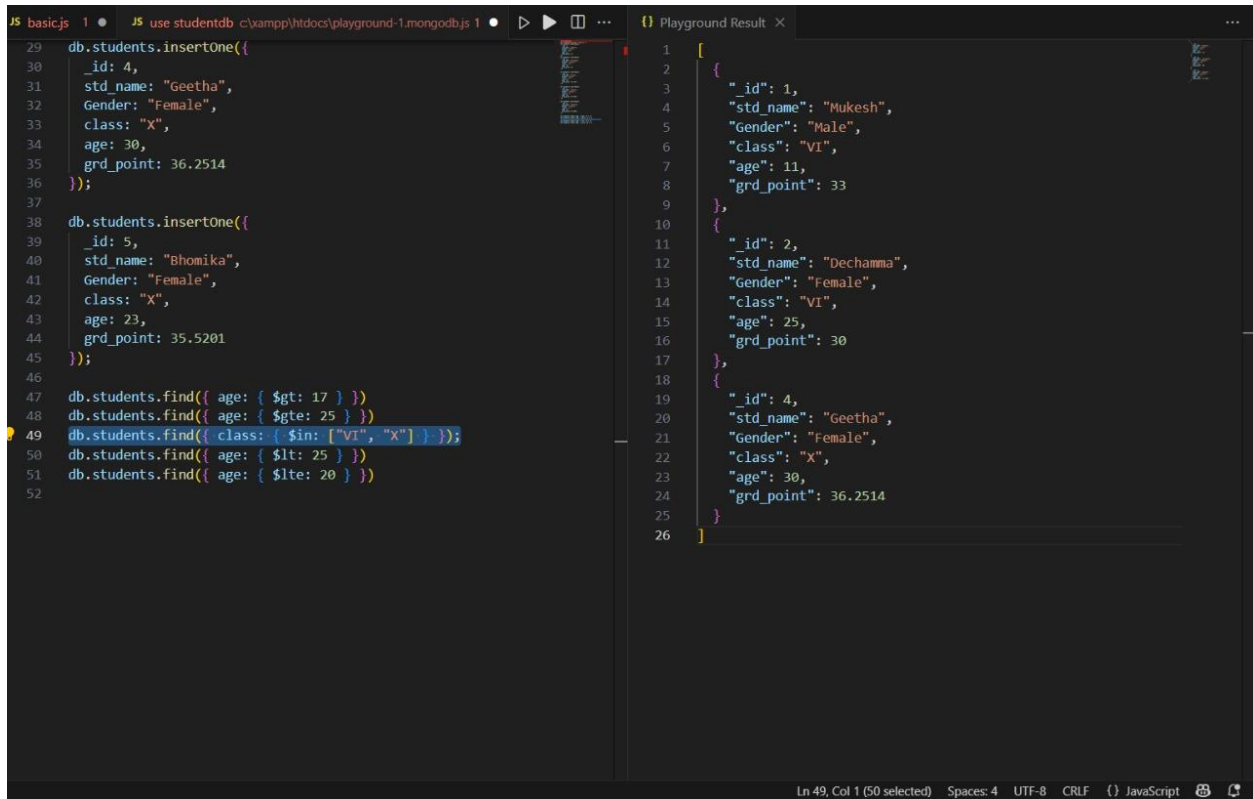
1 [
2   {
3     "_id": 2,
4     "std_name": "Dechamma",
5     "Gender": "Female",
6     "class": "VI",
7     "age": 25,
8     "grd_point": 30
9   },
10  {
11    "_id": 4,
12    "std_name": "Geetha",
13    "Gender": "Female",
14    "class": "X",
15    "age": 30,
16    "grd_point": 36.2514
17  }
18 ]

Ln 48, Col 1 (39 selected) Spaces: 4 UTF-8 CRLF {} JavaScript
```

3.Demonstrate Use of in Operation Along with in MongoDB with 2 Array Values

```
db.students.find({ class: { $in: ["VI", "X"] } });
```

Output:



```
JS basic.js 1 • JS use studentdb c:\xampp\htdocs\playground-1.mongodbs.js 1
29 db.students.insertOne({
30   _id: 4,
31   std_name: "Geetha",
32   Gender: "Female",
33   class: "X",
34   age: 30,
35   grd_point: 36.2514
36 });
37
38 db.students.insertOne({
39   _id: 5,
40   std_name: "Bhomika",
41   Gender: "Female",
42   class: "X",
43   age: 23,
44   grd_point: 35.5201
45 });
46
47 db.students.find({ age: { $gt: 17 } })
48 db.students.find({ age: { $gte: 25 } })
49 db.students.find({ class: { $in: ["VI", "X"] } });
50 db.students.find({ age: { $lt: 25 } })
51 db.students.find({ age: { $lte: 20 } })
52

1 [
2   {
3     "_id": 1,
4     "std_name": "Mukesh",
5     "Gender": "Male",
6     "class": "VI",
7     "age": 11,
8     "grd_point": 33
9   },
10  {
11    "_id": 2,
12    "std_name": "Dechamma",
13    "Gender": "Female",
14    "class": "VI",
15    "age": 25,
16    "grd_point": 30
17  },
18  {
19    "_id": 4,
20    "std_name": "Geetha",
21    "Gender": "Female",
22    "class": "X",
23    "age": 30,
24    "grd_point": 36.2514
25  }
26 ]
```

Ln 49, Col 1 (50 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

#### 4. Display Student Details Where Age<25

```
db.students.find({ age: { $lt: 25 } })
```

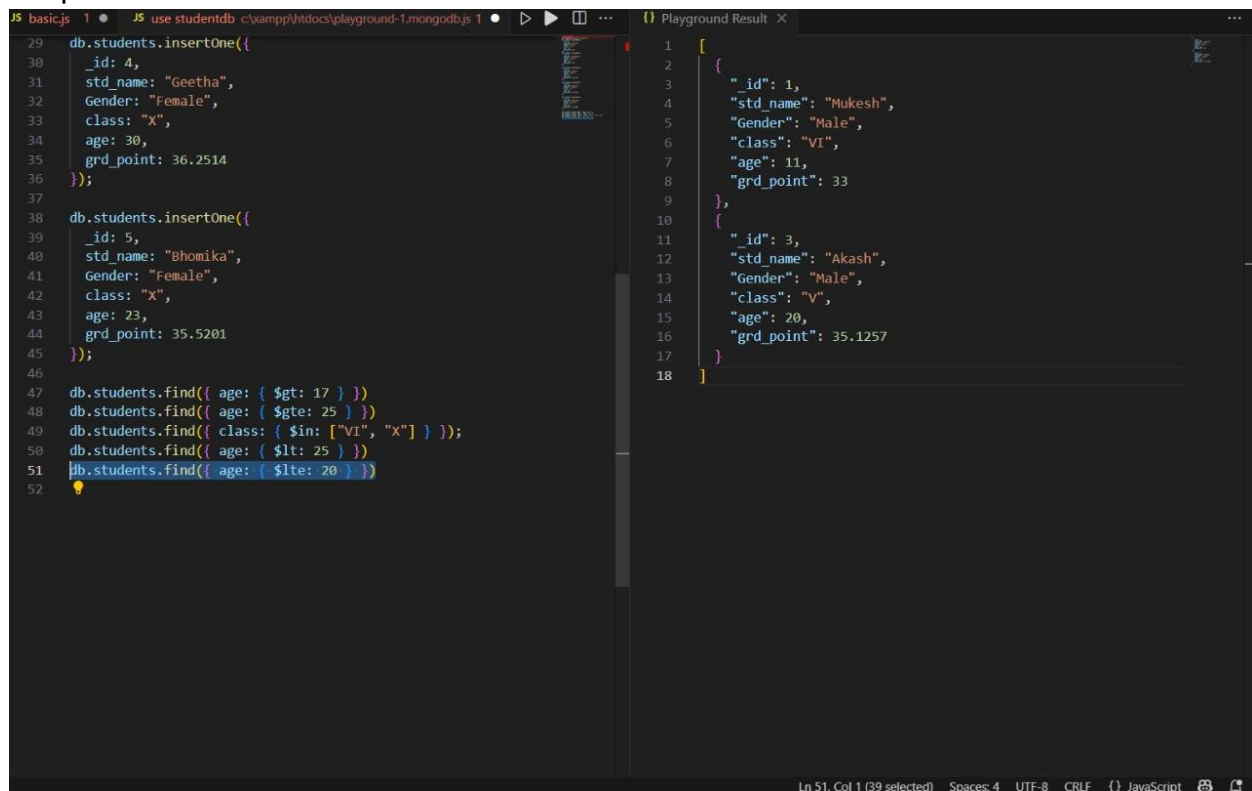
Output:

```
JS basicjs 1 • JS use studentdb c:\xampp\htdocs\playground-1.mongodbs 1 • Playground Result X ...
29 db.students.insertOne({
30   _id: 4,
31   std_name: "Geetha",
32   Gender: "Female",
33   class: "X",
34   age: 30,
35   grd_point: 36.2514
36 });
37
38 db.students.insertOne({
39   _id: 5,
40   std_name: "Bhomika",
41   Gender: "Female",
42   class: "X",
43   age: 23,
44   grd_point: 35.5201
45 });
46
47 db.students.find({ age: { $gt: 17 } })
48 db.students.find({ age: { $gte: 25 } })
49 db.students.find({ class: { $in: ["VI", "X"] } });
50 db.students.find({ age: { $lt: 25 } })
51 db.students.find({ age: { $lte: 20 } })
52
1 {
2   {
3     "_id": 1,
4     "std_name": "Mukesh",
5     "Gender": "Male",
6     "class": "VI",
7     "age": 11,
8     "grd_point": 33
9   },
10  {
11    "_id": 2,
12    "std_name": "Dechamma",
13    "Gender": "Female",
14    "class": "VI",
15    "age": 25,
16    "grd_point": 30
17  },
18  {
19    "_id": 4,
20    "std_name": "Geetha",
21    "Gender": "Female",
22    "class": "X",
23    "age": 30,
24    "grd_point": 36.2514
25  }
26 }
```

## 5.Display Student Details Where Age<=20

```
db.students.find({ age: { $lte: 20 } })
```

Output:



The screenshot shows a MongoDB Playground interface with two panels. The left panel contains JavaScript code for inserting two students and performing several queries. The right panel shows the results of the first query, which returns a single document for a student named Mukesh.

```
29 db.students.insertOne({
30   _id: 4,
31   std_name: "Geetha",
32   Gender: "Female",
33   class: "X",
34   age: 30,
35   grd_point: 36.2514
36 });
37
38 db.students.insertOne({
39   _id: 5,
40   std_name: "Bhomika",
41   Gender: "Female",
42   class: "X",
43   age: 23,
44   grd_point: 35.5201
45 });
46
47 db.students.find({ age: { $gt: 17 } })
48 db.students.find({ age: { $gte: 25 } })
49 db.students.find({ class: { $in: ["VI", "X"] } })
50 db.students.find({ age: { $lt: 25 } })
51 db.students.find({ age: { $lte: 20 } })
52
```

```
1 [
2   {
3     "_id": 1,
4     "std_name": "Mukesh",
5     "Gender": "Male",
6     "class": "VI",
7     "age": 11,
8     "grd_point": 33
9   },
10  {
11    "_id": 3,
12    "std_name": "Akash",
13    "Gender": "Male",
14    "class": "V",
15    "age": 20,
16    "grd_point": 35.1257
17  }
18 ]
```

Ln 51, Col 1 (39 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

## >Intermediate

### Dataset

```
"_id" : 1,
name:"Kavana",
"sem" : 1,
"marks" : [ 70, 87, 90 ]
"_id" : 2,
name:"Ayaan Sharief",
"sem" : 2,
```

"marks" : [ 90, 77, 80 ]

"\_id" : 3,

name:"Varsha",

"sem" : 3,

"marks" : [ 83, 67, 95 ]

"\_id" : 4,

name:"Shifa Banu",

"sem" : 4,

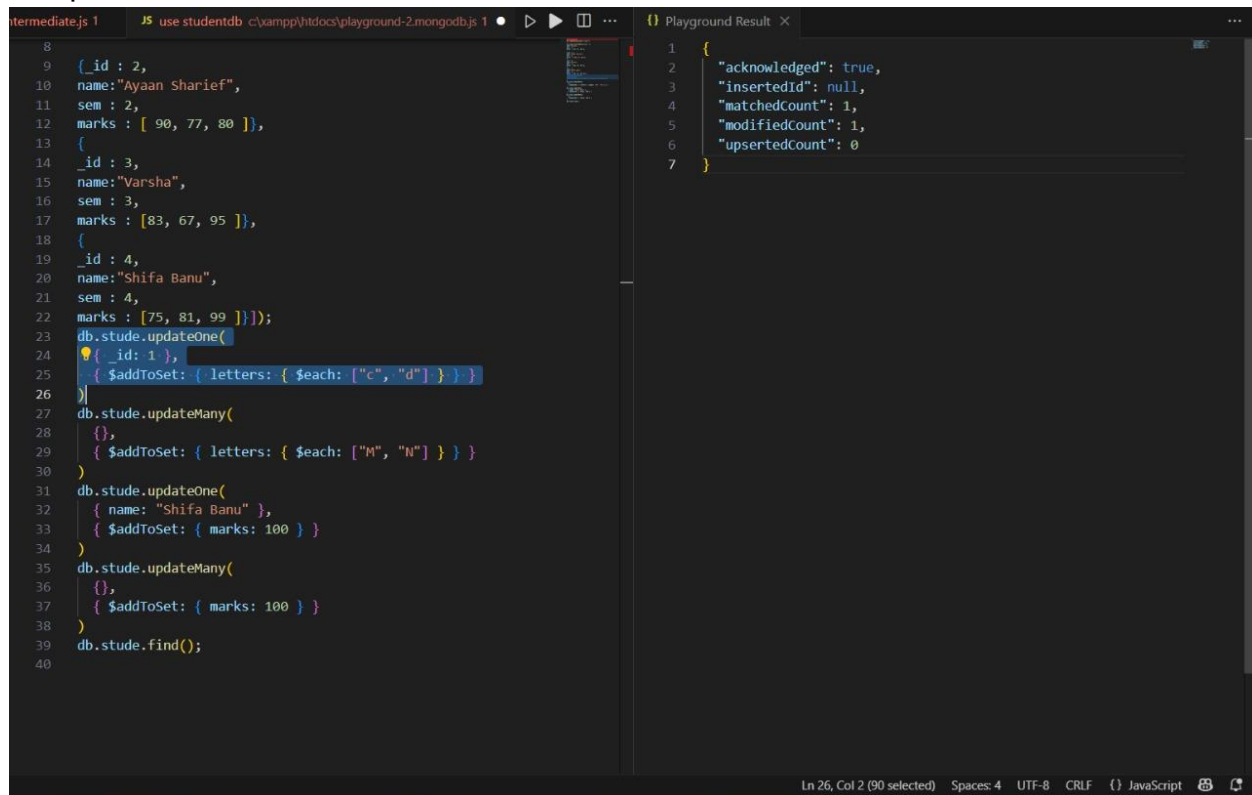
"marks" : [ 75, 81, 99 ]

### **>Using \$addToSet operator in document in MongoDB**

1.Query to Append c And d Letters To Array Letters Whose Id=1 Using updateOne and \$addTo set

```
db.stude.updateOne(  
  { _id: 1 },  
  { $addToSet: { letters: { $each: ["c", "d"] } } }  
)
```

Output:



The screenshot shows a MongoDB Playground interface with a dark theme. The left pane contains a JavaScript script for interacting with a database named 'studentdb'. The script defines an array of student documents and performs several update operations. The right pane shows the result of the final find operation, which returns a single document.

```
intermediate.js 1 JS use studentdb c:\xampp\htdocs\playground-2.mongodbs.js 1 ... Playground Result X ...
```

```
8
9  { _id : 2,
10    name:"Ayaan Sharief",
11    sem : 2,
12    marks : [ 90, 77, 80 ] },
13  {
14    _id : 3,
15    name:"Varsha",
16    sem : 3,
17    marks : [ 83, 67, 95 ] },
18  {
19    _id : 4,
20    name:"Shifa Banu",
21    sem : 4,
22    marks : [ 75, 81, 99 ] } ];
23 db.stude.updateOne(
24   { _id: 1 },
25   { $addToSet: { letters: { $each: ["c", "d"] } } }
26 )
27 db.stude.updateMany(
28   {},
29   { $addToSet: { letters: { $each: ["M", "N"] } } }
30 )
31 db.stude.updateOne(
32   { name: "Shifa Banu" },
33   { $addToSet: { marks: 100 } }
34 )
35 db.stude.updateMany(
36   {},
37   { $addToSet: { marks: 100 } }
38 )
39 db.stude.find();
40
```

```
1 {
2   "acknowledged": true,
3   "insertedId": null,
4   "matchedCount": 1,
5   "modifiedCount": 1,
6   "upsertedCount": 0
7 }
```

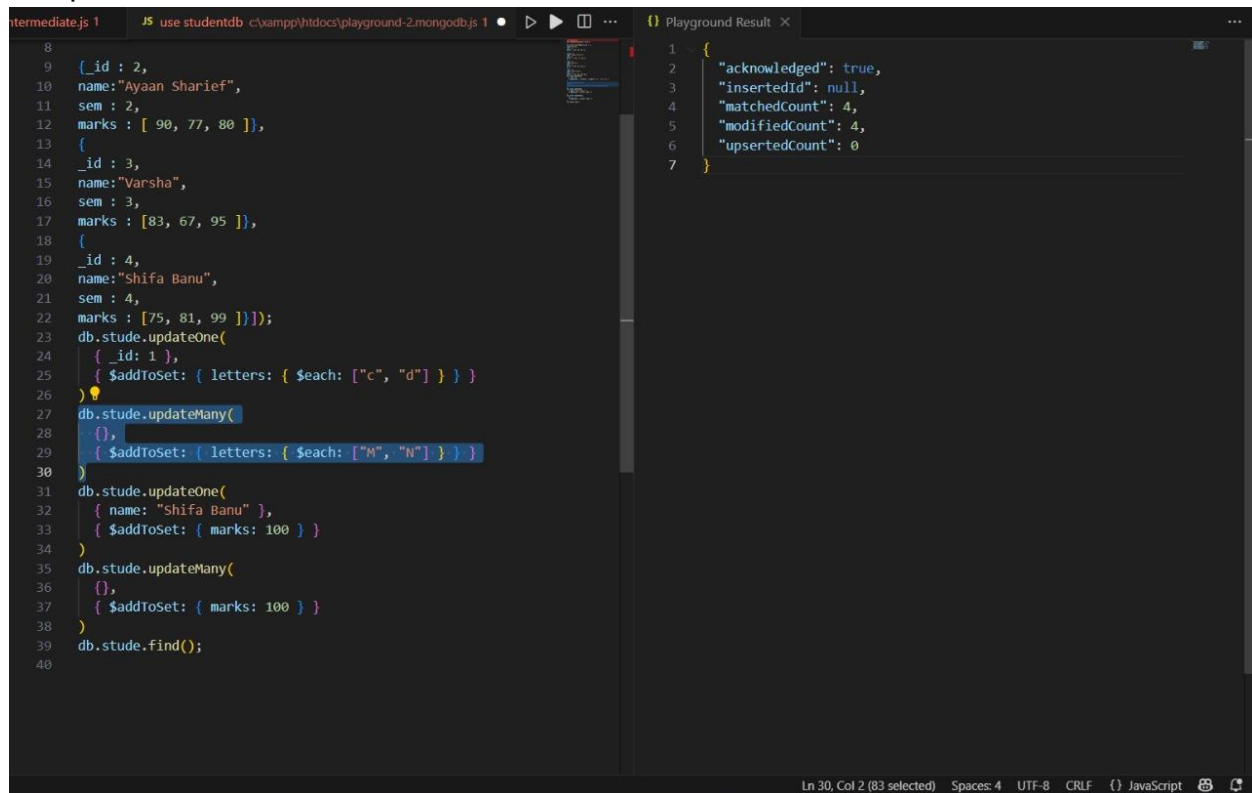
Ln 26, Col 2 (90 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

2. Query to Append M and N Letters to Array Letters Using updateMany And \$addToSet

```
db.stude.updateMany(
{
},
{ $addToSet: { letters: { $each: ["M", "N"] } } }
)
```



Output:



The screenshot shows a MongoDB Playground interface. On the left, a JavaScript script is written in a dark-themed editor. The script defines three students: Ayaan Sharief (sem 2, marks [90, 77, 80]), Varsha (sem 3, marks [83, 67, 95]), and Shifa Banu (sem 4, marks [75, 81, 99]). It then performs several updates: adds 'c' and 'd' to the letters of student 1; adds 'M' and 'N' to the letters of all students; adds 100 to the marks of Shifa Banu; and adds 100 to the marks of all students. Finally, it finds all documents in the 'stude' collection. On the right, the 'Playground Result' pane shows the output of the find() operation, which is an array of three documents. Each document includes an 'acknowledged' field (true), an 'insertedId' (null), a 'matchedCount' (4), a 'modifiedCount' (4), and an 'upsertedCount' (0).

```
8
9 { _id : 2,
10   name: "Ayaan Sharief",
11   sem : 2,
12   marks : [ 90, 77, 80 ] },
13 {
14   _id : 3,
15   name: "Varsha",
16   sem : 3,
17   marks : [ 83, 67, 95 ] },
18 {
19   _id : 4,
20   name: "Shifa Banu",
21   sem : 4,
22   marks : [ 75, 81, 99 ] } ]];
23 db.stude.updateOne(
24   { _id: 1 },
25   { $addToSet: { letters: { $each: [ "c", "d" ] } } }
26 )
27 db.stude.updateMany(
28   {},
29   { $addToSet: { letters: { $each: [ "M", "N" ] } } }
30 )
31 db.stude.updateOne(
32   { name: "Shifa Banu" },
33   { $addToSet: { marks: 100 } }
34 )
35 db.stude.updateMany(
36   {},
37   { $addToSet: { marks: 100 } }
38 )
39 db.stude.find();
40
```

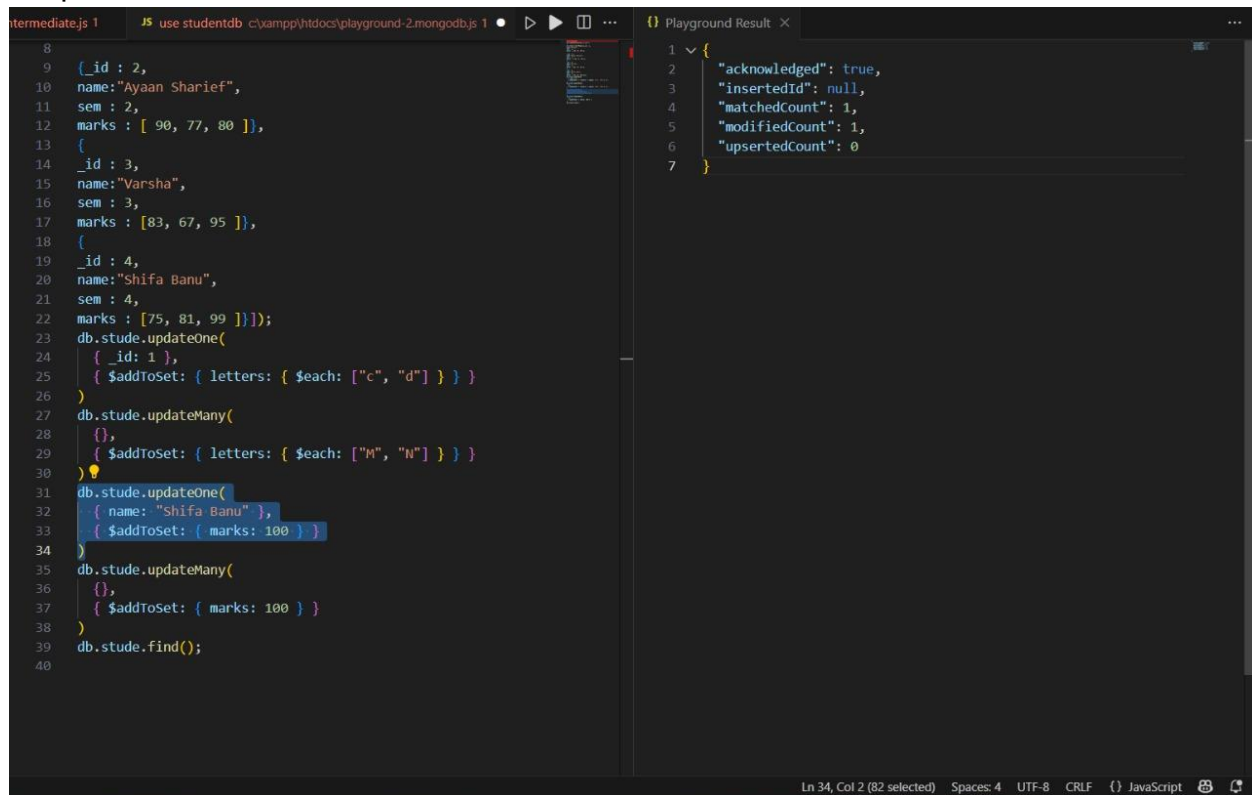
```
1 {
2   "acknowledged": true,
3   "insertedId": null,
4   "matchedCount": 4,
5   "modifiedCount": 4,
6   "upsertedCount": 0
7 }
```

Ln 30, Col 2 (83 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

3. Query to Append 100 to End of Array Using \$addToSet Where Name="Shifa Banu"

```
db.stude.updateOne(
  { name: "Shifa Banu" },
  { $addToSet: { marks: 100 } }
)
```

Output:



The screenshot shows a MongoDB Playground interface with a dark theme. The left pane contains a JavaScript script for interacting with a database named 'studentdb'. The script defines three students: Ayaan Sharief, Varsha, and Shifa Banu. It then performs several update operations: updating the first student's name to 'Ayaan Sharief', adding 'c' and 'd' to the letters of the first student, adding 'M' and 'N' to the letters of all students, updating the name of the third student to 'Shifa Banu' and adding 100 to her marks, and finally adding 100 to the marks of all students. The right pane shows the 'Playground Result' as a JSON object with metadata: 'acknowledged': true, 'insertedId': null, 'matchedCount': 1, 'modifiedCount': 1, and 'upsertedCount': 0. The status bar at the bottom indicates 'Ln 34, Col 2 (82 selected) Spaces: 4 UTF-8 CRLF {} JavaScript'.

```
8
9 { _id : 2,
10   name:"Ayaan Sharief",
11   sem : 2,
12   marks : [ 90, 77, 80 ] },
13 {
14   _id : 3,
15   name:"Varsha",
16   sem : 3,
17   marks : [ 83, 67, 95 ] },
18 {
19   _id : 4,
20   name:"Shifa Banu",
21   sem : 4,
22   marks : [ 75, 81, 99 ] } ]};
23 db.stude.updateOne(
24   { _id: 1 },
25   { $addToSet: { letters: { $each: [ "c", "d" ] } } }
26 )
27 db.stude.updateMany(
28   {},
29   { $addToSet: { letters: { $each: [ "M", "N" ] } } }
30 )
31 db.stude.updateOne(
32   { name: "Shifa Banu" },
33   { $addToSet: { marks: 100 } }
34 )
35 db.stude.updateMany(
36   {},
37   { $addToSet: { marks: 100 } }
38 )
39 db.stude.find();
40
```

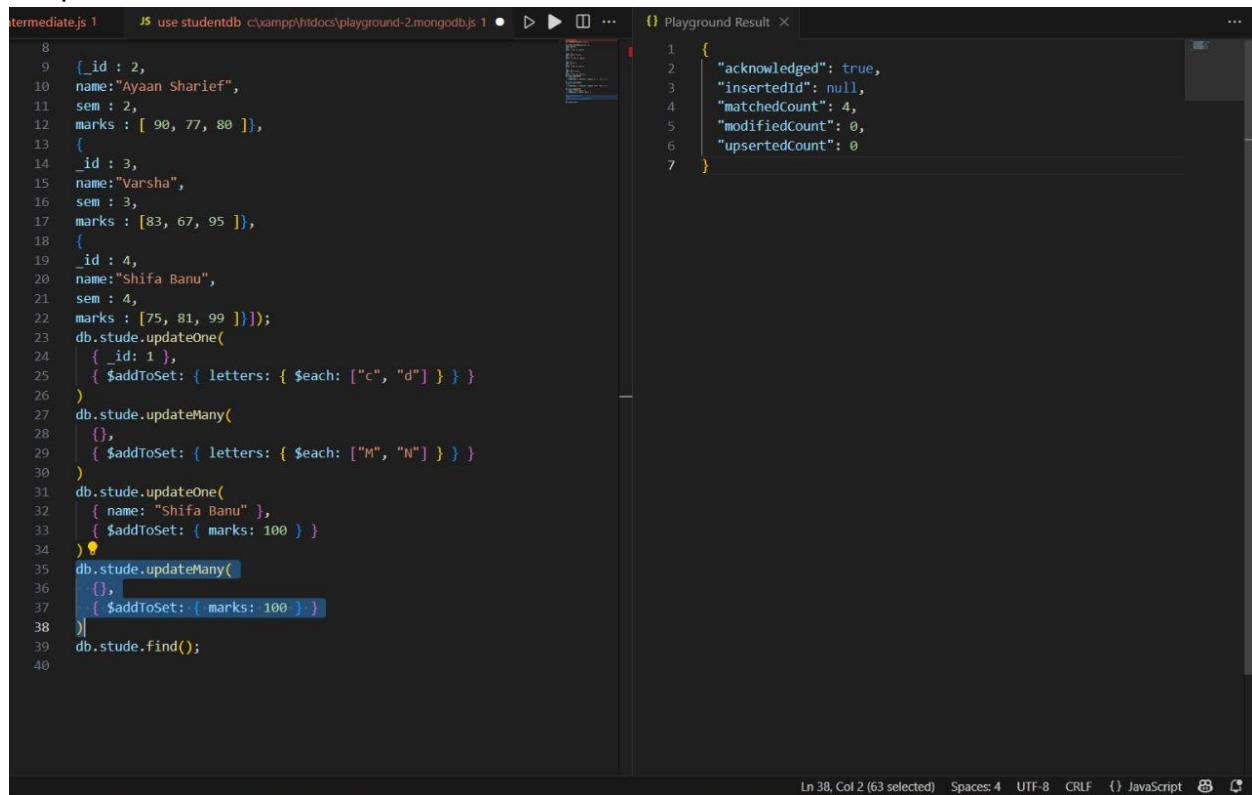
```
1 {
2   "acknowledged": true,
3   "insertedId": null,
4   "matchedCount": 1,
5   "modifiedCount": 1,
6   "upsertedCount": 0
7 }
```

Ln 34, Col 2 (82 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

4.Query To Append 100 To End of Array Using For All Students Using updateMany and \$ addToSet

```
Db.stude.updateMany(
{
},
{ $addToSet: { marks: 100 } }
)
```

Output:



The screenshot shows a MongoDB Playground interface. The left pane contains a JavaScript script for interacting with a database named 'studentdb'. The script defines an array of student documents and performs several update operations. The right pane shows the 'Playground Result' output, which displays a single document with update statistics.

```
8
9  { _id : 2,
10    name: "Ayaan Sharief",
11    sem : 2,
12    marks : [ 90, 77, 80 ] },
13  {
14    _id : 3,
15    name: "Varsha",
16    sem : 3,
17    marks : [ 83, 67, 95 ] },
18  {
19    _id : 4,
20    name: "Shifa Banu",
21    sem : 4,
22    marks : [ 75, 81, 99 ] } ];
23 db.stude.updateOne(
24   { _id: 1 },
25   { $addToSet: { letters: { $each: [ "c", "d" ] } } }
26 )
27 db.stude.updateMany(
28   {},
29   { $addToSet: { letters: { $each: [ "M", "N" ] } } }
30 )
31 db.stude.updateOne(
32   { name: "Shifa Banu" },
33   { $addToSet: { marks: 100 } }
34 )
35 db.stude.updateMany(
36   {},
37   { $addToSet: { marks: 100 } }
38 )
39 db.stude.find();
40
```

```
1 {
2   "acknowledged": true,
3   "insertedId": null,
4   "matchedCount": 4,
5   "modifiedCount": 0,
6   "upsertedCount": 0
7 }
```

Ln 38, Col 2 (63 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

To check the Dataset which is modified bu above operations

```
db.stude.find();
```

Output:

```
intermediate.js 1 JS use studentdb c:\xampp\htdocs\playground-2\mongodb.js 1 ... Playground Result X ...
8
9 { _id : 2,
10   name: "Ayaan Sharief",
11   sem : 2,
12   marks : [ 90, 77, 80 ] },
13 {
14   _id : 3,
15   name: "Varsha",
16   sem : 3,
17   marks : [ 83, 67, 95 ] },
18 {
19   _id : 4,
20   name: "Shifa Banu",
21   sem : 4,
22   marks : [ 75, 81, 99 ] } ];
23 db.stude.updateOne(
24   { _id: 1 },
25   { $addToSet: { letters: { $each: [ "c", "d" ] } } }
26 )
27 db.stude.updateMany(
28   {},
29   { $addToSet: { letters: { $each: [ "M", "N" ] } } }
30 )
31 db.stude.updateOne(
32   { name: "Shifa Banu" },
33   { $addToSet: { marks: 100 } }
34 )
35 db.stude.updateMany(
36   {},
37   { $addToSet: { marks: 100 } }
38 )
39 db.stude.find();
40
19 {
20   "letters": [
21   ],
22 },
23 {
24   "_id": 3,
25   "name": "Varsha",
26   "sem": 3,
27   "marks": [
28     83,
29     67,
30     95,
31     100
32   ],
33   "letters": [
34     "M",
35     "N"
36   ]
37 },
38 {
39   "_id": 4,
40   "name": "Shifa Banu",
41   "sem": 4,
42   "marks": [
43     75,
44     81,
45     99,
46     100
47   ],
48   "letters": [
49     "M",
50     "N"
51   ]
52 }
53 ]
54 }
```

```
intermediate.js 1 JS use studentdb c:\xampp\htdocs\playground-2\mongodb.js 1 ... Playground Result X ...
8
9 { _id : 2,
10   name: "Ayaan Sharief",
11   sem : 2,
12   marks : [ 90, 77, 80 ] },
13 {
14   _id : 3,
15   name: "Varsha",
16   sem : 3,
17   marks : [ 83, 67, 95 ] },
18 {
19   _id : 4,
20   name: "Shifa Banu",
21   sem : 4,
22   marks : [ 75, 81, 99 ] } ];
23 db.stude.updateOne(
24   { _id: 1 },
25   { $addToSet: { letters: { $each: [ "c", "d" ] } } }
26 )
27 db.stude.updateMany(
28   {},
29   { $addToSet: { letters: { $each: [ "M", "N" ] } } }
30 )
31 db.stude.updateOne(
32   { name: "Shifa Banu" },
33   { $addToSet: { marks: 100 } }
34 )
35 db.stude.updateMany(
36   {},
37   { $addToSet: { marks: 100 } }
38 )
39 db.stude.find();
40
19 {
20   "letters": [
21   ],
22 },
23 {
24   "_id": 3,
25   "name": "Varsha",
26   "sem": 3,
27   "marks": [
28     83,
29     67,
30     95,
31     100
32   ],
33   "letters": [
34     "M",
35     "N"
36   ]
37 },
38 {
39   "_id": 4,
40   "name": "Shifa Banu",
41   "sem": 4,
42   "marks": [
43     75,
44     81,
45     99,
46     100
47   ],
48   "letters": [
49     "M",
50     "N"
51   ]
52 }
53 ]
54 }
```