QUESTIONS

**1. Write a MongoDB query to display at least 10 details of the documents of book id, Btitle, aut**

**2. Write a MongoDB query to display the fields Btitle, authors and price but exclude the bookid**

**3. Write a MongoDB query to display the first 3 books based on similar authors.**

**4. Write a MongoDB query to find the book price more than 1500.**

**5. Write a MongoDB query to find the book title who's price is more than 800 but less than 2000.**

**6. Write a MongoDB query to arrange the book name of the bookstore in ascending order along with all the columns.**

**7.Write a MongoDB query to arrange the book name of the bookstore in ascending order along with all the columns.**

**8. Write a MongoDB query to know whether the author in the authors list.**

**9. Write a MongoDB query to update & increase the book price Rs.500 in all book name as "MongoDB Practice"**

**10. Write a MongoDB query to use Update modifiers ($Set, $push, $pull, $setOnInsert, $upsert) in bookstore**

ANSWERS

test> db.books.insertMany([{bookid: 1,B\_title: "Book 1",authors: ["Author 1", "Author 2"],price: 1000},{bookid: 2,B\_title: "Book 2",authors: ["Author 2", "Author 3"],price: 1500},{bookid: 3,B\_title: "Book 3",authors: ["Author 1", "Author 4"],price: 800},{bookid: 4,B\_title: "MongoDB Practice",authors: ["Author 5"],price: 2000},{bookid: 5,B\_title: "Book 5",authors: ["Author 1", "Author 3"],price: 500},{bookid: 6,B\_title: "Book 6",authors: ["Author 2", "Author 3"],price: 800},{bookid: 7,B\_title: "Book 7",authors: ["Author 1", "Author 4"],price: 300},{bookid: 8,B\_title: "MongoDB Practice",authors: ["Author 4"],price: 2500},{bookid: 9,B\_title: "Java Programming",authors: ["Author 3", "Author 6"],price: 1300},{bookid: 10,B\_title: "MySql Practice",authors: ["Author 1, Author 9"],price: 2800}])

{

acknowledged: true,

insertedIds: {

'0': ObjectId("653dfa0d3d5ae37f23a23f74"),

'1': ObjectId("653dfa0d3d5ae37f23a23f75"),

'2': ObjectId("653dfa0d3d5ae37f23a23f76"),

'3': ObjectId("653dfa0d3d5ae37f23a23f77"),

'4': ObjectId("653dfa0d3d5ae37f23a23f78"),

'5': ObjectId("653dfa0d3d5ae37f23a23f79"),

'6': ObjectId("653dfa0d3d5ae37f23a23f7a"),

'7': ObjectId("653dfa0d3d5ae37f23a23f7b"),

'8': ObjectId("653dfa0d3d5ae37f23a23f7c"),

'9': ObjectId("653dfa0d3d5ae37f23a23f7d")

}

}

1:test> db.books.find({}, { \_id: 0, bookid: 1, B\_title: 1, authors: 1 }).limit(10)

[

{ bookid: 1, B\_title: 'Book 1', authors: [ 'Author 1', 'Author 2' ] },

{ bookid: 2, B\_title: 'Book 2', authors: [ 'Author 2', 'Author 3' ] },

{ bookid: 3, B\_title: 'Book 3', authors: [ 'Author 1', 'Author 4' ] },

{ bookid: 4, B\_title: 'MongoDB Practice', authors: [ 'Author 5' ] },

{ bookid: 5, B\_title: 'Book 5', authors: [ 'Author 1', 'Author 3' ] },

{ bookid: 6, B\_title: 'Book 6', authors: [ 'Author 2', 'Author 3' ] },

{ bookid: 7, B\_title: 'Book 7', authors: [ 'Author 1', 'Author 4' ] },

{ bookid: 8, B\_title: 'MongoDB Practice', authors: [ 'Author 4' ] },

{

bookid: 9,

B\_title: 'Java Programming',

authors: [ 'Author 3', 'Author 6' ]

},

{

bookid: 10,

B\_title: 'MySql Practice',

authors: [ 'Author 1, Author 9' ]

}

2:test> db.books.find({}, { \_id: 0, B\_title: 1, authors: 1, price: 1 })

[

{

B\_title: 'Book 1',

authors: [ 'Author 1', 'Author 2' ],

price: 1000

},

{

B\_title: 'Book 2',

authors: [ 'Author 2', 'Author 3' ],

price: 1500

},

{

B\_title: 'Book 3',

authors: [ 'Author 1', 'Author 4' ],

price: 800

},

{ B\_title: 'MongoDB Practice', authors: [ 'Author 5' ], price: 2000 },

{

B\_title: 'Book 5',

authors: [ 'Author 1', 'Author 3' ],

price: 500

},

{

B\_title: 'Book 6',

authors: [ 'Author 2', 'Author 3' ],

price: 800

},

{

B\_title: 'Book 7',

authors: [ 'Author 1', 'Author 4' ],

price: 300

},

{ B\_title: 'MongoDB Practice', authors: [ 'Author 4' ], price: 2500 },

{

B\_title: 'Java Programming',

authors: [ 'Author 3', 'Author 6' ],

price: 1300

},

{

B\_title: 'MySql Practice',

authors: [ 'Author 1, Author 9' ],

price: 2800

}

]

test> bookstore> db.books.find({}, { \_id: 0, B\_title: 1, authors: 1, price: 1 })

ReferenceError: bookstore is not defined

test> bookstore> db.books.find({}, { \_id: 0, B\_title: 1, authors: 1, price: 1 })

ReferenceError: bookstore is not defined

test> db.books.find().sort({ authors: 1 }).limit(3)

[

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f78"),

bookid: 5,

B\_title: 'Book 5',

authors: [ 'Author 1', 'Author 3' ],

price: 500

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f76"),

bookid: 3,

B\_title: 'Book 3',

authors: [ 'Author 1', 'Author 4' ],

price: 800

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f74"),

bookid: 1,

B\_title: 'Book 1',

authors: [ 'Author 1', 'Author 2' ],

price: 1000

}

]

test> db.books.find({ price: { $gt: 1500 } })

[

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f77"),

bookid: 4,

B\_title: 'MongoDB Practice',

authors: [ 'Author 5' ],

price: 2000

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7b"),

bookid: 8,

B\_title: 'MongoDB Practice',

authors: [ 'Author 4' ],

price: 2500

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7d"),

bookid: 10,

B\_title: 'MySql Practice',

authors: [ 'Author 1, Author 9' ],

price: 2800

}

]

test> db.books.find({price:{$gt:800,$lt:2000}},{B\_title:1 })

[

{ \_id: ObjectId("653dfa0d3d5ae37f23a23f74"), B\_title: 'Book 1' },

{ \_id: ObjectId("653dfa0d3d5ae37f23a23f75"), B\_title: 'Book 2' },

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7c"),

B\_title: 'Java Programming'

}

]

test> db.books.find().sort({ B\_title: 1 })

[

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f74"),

bookid: 1,

B\_title: 'Book 1',

authors: [ 'Author 1', 'Author 2' ],

price: 1000

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f75"),

bookid: 2,

B\_title: 'Book 2',

authors: [ 'Author 2', 'Author 3' ],

price: 1500

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f76"),

bookid: 3,

B\_title: 'Book 3',

authors: [ 'Author 1', 'Author 4' ],

price: 800

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f78"),

bookid: 5,

B\_title: 'Book 5',

authors: [ 'Author 1', 'Author 3' ],

price: 500

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f79"),

bookid: 6,

B\_title: 'Book 6',

authors: [ 'Author 2', 'Author 3' ],

price: 800

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7a"),

bookid: 7,

B\_title: 'Book 7',

authors: [ 'Author 1', 'Author 4' ],

price: 300

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7c"),

bookid: 9,

B\_title: 'Java Programming',

authors: [ 'Author 3', 'Author 6' ],

price: 1300

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f77"),

bookid: 4,

B\_title: 'MongoDB Practice',

authors: [ 'Author 5' ],

price: 2000

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7b"),

bookid: 8,

B\_title: 'MongoDB Practice',

authors: [ 'Author 4' ],

price: 2500

},

{

\_id: ObjectId("653dfa0d3d5ae37f23a23f7d"),

bookid: 10,

B\_title: 'MySql Practice',

authors: [ 'Author 1, Author 9' ],

price: 2800

}

]

8:test> db.books.find({ authors: "AuthorName" }).count() > 0

false

9:test> db.books.update({ B\_title: "MongoDB Practice" }, { $inc: { price: 500 } }, { multi: true })

DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.

{

acknowledged: true,

insertedId: null,

matchedCount: 2,

modifiedCount: 2,

upsertedCount: 0

}

10:test> db.books.update({ B\_title: "Book 1" }, { $set: { price: 1200 } })

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

test> db.books.update({ B\_title: "Book 2" }, { $push: { authors: "New Author" } })

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

test> db.books.update({ B\_title: "Non-Existent Book" }, { $set: { price: 900 } }, { upsert: true })

{

acknowledged: true,

insertedId: ObjectId("653dfba2a936dbd4d7d7ead8"),

matchedCount: 0,

modifiedCount: 0,

upsertedCount: 1

}