Date : 9th - Dec- 2020
Morning Session : 9am – 11pm
By ~ Surya

Topics: Mongo Aggregation Queries and uploading file using express

Yesterday covered aggregation queries skip, limit, project,

[MongoDb.pdf - Google Drive]

Group:

```
MongoClient.connect(connectionUrl, {useNewUrlParser: true}, (err, client) => {
    if(err) {
        return console.log("Unable to connect to db");
    }
    const db = client.db(databaseName);
    db.collection('books').aggregate([
        {
            $group : {
                _id: "$genre" , books: {$push: "$title"}
            }
        }
    ]).toArray().then(data => console.log("data-1", data));

    db.collection('books').aggregate([
        {
            $group: {
                _id:"$genre", count: {$sum:2} //count = 0; count = count + 2
            }
        }
    ]).toArray().then(data => console.log("data-2", data));
    db.collection('books').aggregate([
        {
            $group: {
                _id:"$genre", books: {$first: "$title"}
            }
        }
    ]).toArray().then(data => console.log("data-3", data));
    console.log("connected to db");
});
```

Match:

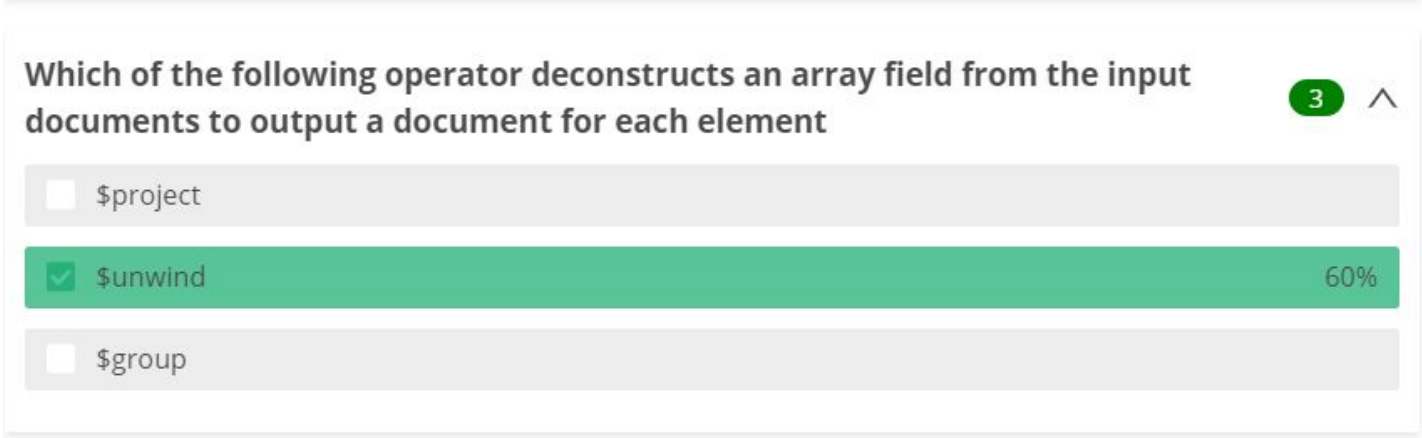
```
const mongodb = require("mongodb");
const MongoClient = mongodb.MongoClient;
const connectionUrl = 'mongodb://localhost:27017';
const databaseName = 'aggregation';

MongoClient.connect(connectionUrl, {useNewUrlParser: true}, (err, client) => {
    if(err) {
        return console.log("Unable to connect to db");
    }
    const db = client.db(databaseName);
    console.log("connected to db");
    db.collection('books').aggregate([
        {
            $match: {
                genre: "comedy"
            }
        },
        {$skip: 1},
        {$limit: 1}
    ]).toArray().then(data => console.log("data", data));

    db.collection('books').aggregate([
        {
            $match: {
                genre: "comic", _id :{$lt: 1}
            }
        }
    ]).toArray().then(data => console.log("data", data));
});
```

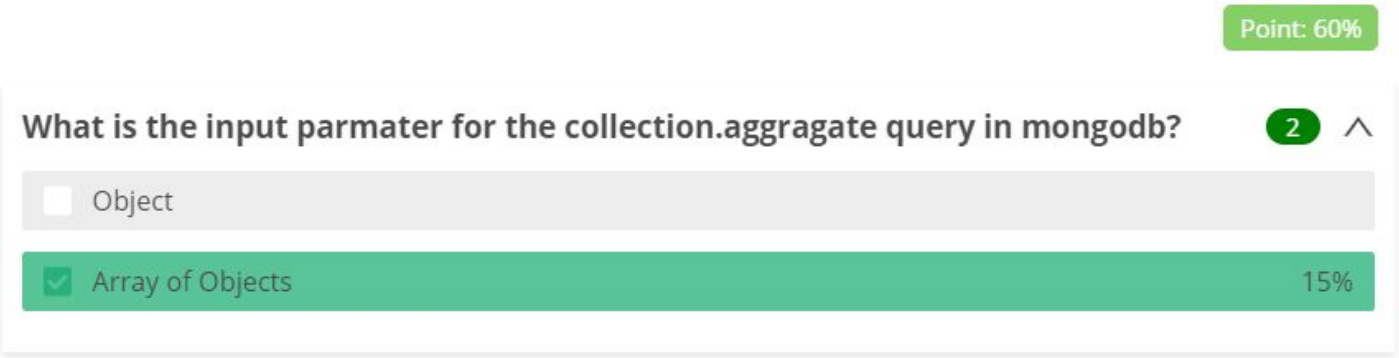
Sort:

```
const mongodb = require("mongodb");
const MongoClient = mongodb.MongoClient;
const connectionUrl = 'mongodb://localhost:27017';
const databaseName = 'aggregation';

MongoClient.connect(connectionUrl, {useNewUrlParser: true}, (err, client) => {
    if(err) {
        return console.log("Unable to connect to db");
    }
    const db = client.db(databaseName);
    console.log("connected to db");
    db.collection('books').aggregate([
        {
            $sort: {genre: 1, _id: -1}
        }
    ]).toArray().then(data=>console.log("data", data))
});
```

Unwind:

```
const mongodb = require("mongodb");
const MongoClient = mongodb.MongoClient;
const connectionUrl = 'mongodb://localhost:27017';
const databaseName = 'aggregation';

MongoClient.connect(connectionUrl, {useNewUrlParser: true}, (err, client) => {
    if(err) {
        return console.log("Unable to connect to db");
    }
    const db = client.db(databaseName);
    console.log("connected to db");
    db.collection('books').aggregate([
        {
            $group: {
                _id: "$genre",
                books: {$push: "$title"}
            }
        },
        {
           $unwind: "$books"
        }
    ]).toArray().then(data=>console.log("data", data))
});
```





Interview Questions:

When do we use Namespace in MongoDB?

What is aggregation in MOngoDB?

Which type of data MongoDb stores?

All the Aggregator operators?

File Upload:

Today we can cover frontend file upload part

For that create react app

File upload ui part code implementation please through the recorded lecture

10-12-2020 Recorded Lecture

Interview Questions:

We have class components, Now react introduced functional components which will maintain state and lifecycle methods and they are suggesting to use functional components instead of class components. Why??

1.we no longer needed 'this' in the functional component .

2.Binding is not required any more in functional component => constructor() {this.function = this.function.bind(this)} .

3.functional components have very less number of lines in code wise whereas class components have more number of lines it is easier to read and easier to test.

4.Class component takes more memory compared to functional components because it needs to initialise a constructor and inherit the properties from the parent(class extend React.component).

5.We can reuse the stateful logic in functional components.

Performance can be measured in the form of renders and memory usage when we use arrow function for every render a new instance will get created inside a memory

---