Indexing array and objects

Multikey Index cont’d

Multikey Indexes on array fields with Objects

We may also create multikey indexes on array fields that contain nested objects i.e. creating Index inside an array object in a document.

Let’s consider the collection stock having the documents as under:

```


```
This **stock** collection has an array **stock** which is an object having 3 fields "size", "color" and "qty".

Let's display only one document using `findOne()` from the stock collection.

```javascript
> db.store.findOne()
{    "id": ObjectId("5ebd64f24dd36225ea3a4076"),
    "item": "tshirt",
    "stock": [
        {           "size": "M",
            "color": "blue",
            "qty": 50
        },
        {            "size": "S",
            "color": "red",
            "qty": 43
        },
        {            "size": "M",
            "color": "blue",
            "qty": 25
        }
    ]
}
```

This clearly shows that the **stock** is an array object which has **size**, **color** and **qty** (quantity) fields. Store owner always want to query quantity available for each color and size. Therefore, they may use query command on finding **stock.qty** or **stock.size** data frequently. In order to meet the demand & supply ratio, it is good to create an index on these fields.

To create a simple multikey index on the array object field **stock.qty**, command will be

```javascript
db.store.createIndex({"stock.qty":1})
```
We can also create a compound multikey index on the stock.size and stock.qty fields on store collection using the following command:

```javascript
db.store.createIndex( { "stock.size": 1, "stock.qty": 1 } )
```

These kind of compound multikey index may support queries with predicates that include both indexed fields as well as predicates that include only the index prefix "stock.size".

**Example-1**

```javascript
db.store.find({"stock.size":"XL"})
```

This is to find all stock of size XL in the collection.

**Example-2**

```javascript
db.store.find({"stock.size":"S", "stock.qty":{$eq:15}})
```
Another use of the compound multikey index is to speedup sort operations.

Example-3
We are executing following two commands, we may see the difference in the output. Commands only differ in the sorting order of array field size ascending and descending

```javascript
db.store.find().sort({"stock.size":1, "stock.qty":1})
```

```javascript
db.store.find().sort({"stock.size":-1, "stock.qty":1})
```

Example-4
```javascript
db.store.find({"stock.size":"S"}).sort({ "stock.qty":1})
```

will first find Stock with M size then sort it on Qty in descending order.
Assignments

1. How to create index on Array Objects? What are their uses.

2. Give an example of sorting using Compound multikey index on array objects?