B Yaswanth 17 jun to 22 jun weekly tasks pdf

1. Find all products where the price field exists and is a number, the cpu speed is greater than 2 GHz and the storage array contains the value 512.

```
db.products.find({
  price: { $exists: true, $type: "number" },
  "spec.cpu": { $gt: 2 },
  storage: 512
})
```

2. Find products where both 128 and 256 gb options exist as storage options. Print their name and price.

```
db.products.find(
 {
   storage: { $all: [128, 256] }
 },
 {
   name: 1,
   price: 1
 }
)
```

3. Find the top 2 products with the highest cpu speed and where the price is between 600 and 1000.

```
db.products.aggregate([
 {
```

```
  $match: { price: { $gte: 600, $lte: 1000 } }
 },
 {
  $sort: { "spec.cpu": -1 }
 },
 {
  $limit: 2
 }
])
```

4. Update products that have orange as one of their color options, setting their price to the value of a particular storage option.

```
db.products.updateMany(
 {
  color: "orange"
 },
 {
  $set: { price: "$storage.0" }
 }
)
```

5. Update products where the ram is less than 8 gb and the screen is greater than 7 inches, setting their price to a particular value.

```
db.products.updateMany(
 {
  "spec.ram": { $lt: 8 },
  "spec.screen": { $gt: 7 }
```

```
 },
 {
   $set: { price: 499 }
 }
)
```

6. Update price of 'xReader' to 199 and display all documents in ascending order of price.

```
db.products.updateOne(
 {
   name: "xReader"
 },
 {
   $set: { price: 199 }
 }
);
db.products.find().sort({ price: 1 });
```

update the prices of the 3 recent mostly released products

```
{"price": {"$multiply": ["$price", 1.1]}}
```

update products where the spec ram is less than 8 and the spec screen is greater than 7 setting their price value to a specific value

```
{
  "spec.ram": { "$lt": 8 },
  "spec.screen": { "$gt": 7 }
```

}

uppdate products that have orange as one of their colour options setting thier price to a diff value based on storage options

```
{
  "$set": {
    "price": {
      "$switch": {
        "branches": [
          { "case": { "$eq": ["$storage", "32GB"] }, "then": 500 },
          { "case": { "$eq": ["$storage", "64GB"] }, "then": 700 },
          { "case": { "$eq": ["$storage", "128GB"] }, "then": 900 },
          { "case": { "$eq": ["$storage", "256GB"] }, "then": 1200 }
        ],
        "default": 1000  // Default price if storage option doesn't match any case
      }
    }
  }
}
```

set the first 3 procucts screen sice as 9 after sorting ascending order of price which exists

```
db.product.aggregate([
  { $sort: { price: 1 } },
  { $limit: 3 },
  { $set: { "spec.screen_size": 9 } }
```

]);

find the product with the highest price and update its price

```
[
 { "$sort": { "price": -1 } },
 { "$limit": 1 }
]
```

```
[
 { "$sort": { "price": -1 } },
 { "$limit": 1 }
]
```

find the top 2 product with thehighest cpu speed and where is betwween 600 and 1000

```
[
 {
  $match: {
   "spec.cpu_speed": { $gte: 600, $lte: 1000 }
  }
 },
 {
  $sort: {
   "spec.cpu_speed": -1
  }
 },
 {
```

```
    $limit: 2
  }
]
```

Mongosh code Saturday class :

```
use vit5
switched to db vit5
db.products.insertMany([
  { "_id" : 1, "name" : "xPhone", "price" : 799, "releaseDate" : ISODate("2011-05-14T00:00:00Z"), "spec" : { "ram" : 4, "screen" : 6.5, "cpu" : 2.66 }, "color" : [ "white", "black" ], "storage" : [ 64, 128, 256 ] },
  { "_id" : 2, "name" : "xTablet", "price" : 899, "releaseDate" : ISODate("2011-09-01T00:00:00Z"), "spec" : { "ram" : 16, "screen" : 9.5, "cpu" : 3.66 }, "color" : [ "white", "black", "purple" ], "storage" : [ 128, 256, 512 ] },
  { "_id" : 3, "name" : "SmartTablet", "price" : 899, "releaseDate" : ISODate("2015-01-14T00:00:00Z"), "spec" : { "ram" : 12, "screen" : 9.7, "cpu" : 3.66 }, "color" : [ "blue" ], "storage" : [ 16, 64, 128 ] },
  { "_id" : 4, "name" : "SmartPad", "price" : 699, "releaseDate" : ISODate("2020-05-14T00:00:00Z"), "spec" : { "ram" : 8, "screen" : 9.7, "cpu" : 1.66 }, "color" : [ "white", "orange", "gold", "gray" ], "storage" : [ 128, 256, 1024 ] },
  { "_id" : 5, "name" : "SmartPhone", "price" : 599, "releaseDate" : ISODate("2022-09-14T00:00:00Z"), "spec" : { "ram" : 4, "screen" : 9.7, "cpu" : 1.66 }, "color" : [ "white", "orange", "gold", "gray" ], "storage" : [ 128, 256 ] },
  { "_id" : 6, "name" : "xWidget", "spec" : { "ram" : 64, "screen" : 9.7, "cpu" : 3.66 }, "color" : [ "black" ], "storage" : [ 1024 ] },
  { "_id" : 7, "name" : "xReader", "price" : null, "spec" : { "ram" : 64, "screen" : 6.7, "cpu" : 3.66 }, "color" : [ "black", "white" ], "storage" : [ 128 ] }
```

```
])
{
  acknowledged: true,
  insertedIds: {
    '0': 1,
    '1': 2,
    '2': 3,
    '3': 4,
    '4': 5,
    '5': 6,
    '6': 7
  }
}
db.products.createIndex({price:1})
price_1
db.products.dropIndex({price:1})
{ nIndexesWas: 2, ok: 1 }
db.products.getIndexes()
[ { v: 2, key: { id: 1 }, name: '_id' } ]
db.products.createIndex({name: -1, price: 1})
name_-1_price_1
db.products.createIndex({"spec.ram": 1, "spec.cpu": -1})
spec.ram_1_spec.cpu_-1
db.blog.insertMany( [
  {
    _id: 1,
    content: "This morning I had a cup of coffee.",
    about: "beverage",
```

```
    keywords: [ "coffee" ]
  },
  {
    _id: 2,
    content: "Who likes chocolate ice cream for dessert?",
    about: "food",
    keywords: [ "poll" ]
  },
  {
    _id: 3,
    content: "My favorite flavors are strawberry and coffee",
    about: "ice cream",
    keywords: [ "food", "dessert" ]
  }
])
{
  acknowledged: true,
  insertedIds: {
    '0': 1,
    '1': 2,
    '2': 3
  }
}
db.blog.insertMany( [
  {
    _id: 1,
    content: "This morning I had a cup of coffee.",
    about: {author : "Jayanth", writer : "Jayanth Vellingiri"}
```

```
    keywords: [ "coffee" ]
  },
  {
    _id: 2,
    content: "Who likes chocolate ice cream for dessert?",
    about: {author:"Karthik", writer:"Karthik Raj"}
    keywords: [ "poll" ]
  },
  {
    _id: 3,
    content: "My favorite flavors are strawberry and coffee",
    about: {author:"Praveen", writer:"praveen kumar"}
    keywords: [ "food", "dessert" ]
  }
] )
SyntaxError: Unexpected token, expected "," (6:5)

  4 |     content: "This morning I had a cup of coffee.",
  5 |     about: {author : "Jayanth", writer : "Jayanth Vellingiri"}
> 6 |     keywords: [ "coffee" ]
    |     ^
  7 |   },
  8 |   {
  9 |     _id: 2,
db.blog.insertMany( [
  {
    _id: 1,
    content: "This morning I had a cup of coffee.",
```

```
    about: {author : "Jayanth", writer : "Jayanth Vellingiri"},
    keywords: [ "coffee" ]
  },
  {
    _id: 2,
    content: "Who likes chocolate ice cream for dessert?",
    about: {author:"Karthik", writer:"Karthik Raj"},
    keywords: [ "poll" ]
  },
  {
    _id: 3,
    content: "My favorite flavors are strawberry and coffee",
    about: {author:"Praveen", writer:"praveen kumar"},
    keywords: [ "food", "dessert" ]
  }
] )
{
  acknowledged: true,
  insertedIds: {
    '0': 1,
    '1': 2,
    '2': 3
  }
}
db.blog.createIndex({"content":"text"})
content_text
db.blog.getIndexes()
[
```

```
 { v: 2, key: { id: 1 }, name: '_id' },
 {
  v: 2,
  key: { _fts: 'text', _ftsx: 1 },
  name: 'content_text',
  weights: { content: 1 },
  default_language: 'english',
  language_override: 'language',
  textIndexVersion: 3
 }
]
db.blog.find({$text: { $search: "cream" }})
MongoServerError[IndexNotFound]: text index required for $text query
db.blog.find({$text: { $search: "cream" }})
{
 _id: 2,
 content: 'Who likes chocolate ice cream for dessert?',
 about: {
  author: 'Karthik',
  writer: 'Karthik Raj'
 },
 keywords: [
  'poll'
 ]
}
db.blog.createIndex({"$":"text"})
MongoServerError[IndexOptionsConflict]: An equivalent index already exists with a different name and options. Requested index: { v: 2, key: { _fts: "text", _ftsx: 1 }, name: "$_text", weights: { $: 1 }, default_language: "english", language_override: "language",
```

textIndexVersion: 3 }, existing index: { v: 2, key: { _fts: "text", _ftsx: 1 }, name: "content_text", weights: { content: 1 }, default_language: "english", language_override: "language", textIndexVersion: 3 }

db.blog.dropIndex("content_text")

{ nIndexesWas: 2, ok: 1 }

db.blog.createIndex({"$":"text"})

$_text

db.blog.find({$text: { $search: "Raj" }})

```
{
  _id: 2,
  content: 'Who likes chocolate ice cream for dessert?',
  about: {
    author: 'Karthik',
    writer: 'Karthik Raj'
  },
  keywords: [
    'poll'
  ]
}
```

db.blog.find({$text: { $search: "Raj coffee" }})

```
{
  _id: 1,
  content: 'This morning I had a cup of coffee.',
  about: {
    author: 'Jayanth',
    writer: 'Jayanth Vellingiri'
  },
  keywords: [
    'coffee'
```

```
  ]
}
{
  _id: 3,
  content: 'My favorite flavors are strawberry and coffee',
  about: {
    author: 'Praveen',
    writer: 'praveen kumar'
  },
  keywords: [
    'food',
    'dessert'
  ]
}
{
  _id: 2,
  content: 'Who likes chocolate ice cream for dessert?',
  about: {
    author: 'Karthik',
    writer: 'Karthik Raj'
  },
  keywords: [
    'poll'
  ]
}
db.blog.find({$text:{$search:"\"ice cream\""}})
{
  _id: 2,
```

```
content: 'Who likes chocolate ice cream for dessert?',
about: {
  author: 'Karthik',
  writer: 'Karthik Raj'
},
keywords: [
  'poll'
]
```