Activity 3.4 - Update and Delete

Perform the tasks to acquire the knowledge to update and delete documents, collections and databases in a MongoDB.

Instructions

From database m02uf4 and collections movies and moviestolkien from previous activity, perform the following tasks.
All tasks must include the necessary commands to verify the right behaviour of the solution proposed.
Estimated time: 2h.
No evaluation: solution in class

Task 1.

Add a field with the synopsis to the following documents:

Document - The Hobbit: An Unexpected Journey	
field	**value**
synopsis	A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug.

Document - The Hobbit: The Desolation of Smaug	
field	**value**
synopsis	The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of a mysterious and magical ring.

```
db.movies.update( { title : "The Hobbit: An Unexpected Journey" },
{ $set : { synopsis :  "A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug." } } )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title: "The Hobbit: An Unexpected Journey" }, {title:1, synopsis:1}).pretty()

db.movies.update( { title : "The Hobbit: The Desolation of Smaug" }, { $set : { synopsis :  "The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of a mysterious and magical ring." } } )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
db.movies.find({title: "The Hobbit: The Desolation of Smaug" }, {title:1,
synopsis:1}).pretty()
```

Task 2.

Add these actors to the following documents:

Document - Pulp Fiction			
name	age	children	hometown
Samuel L. Jackson	70	1	Los Angeles
Bruce Willis	65	5	Los Angeles

```
db.movies.update( { title : "Pulp Fiction" },
{ $push:
  { actors :
    {$each: [
          { name : "Samuel L. Jackson",  age : 70, children : 1, hometown : "Los Angeles"
},
         { name : "Bruce Willis",  age : 65, children : 5, hometown : "Los Angeles" }]
    }
   }
})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title : "Pulp Fiction"}, {title:true, actors:true}).pretty()
```

Also add this actress in the second position of the following document:

Document - Fight Club			
name	age	children	hometown
Helena Bonham Carter	52	2	Hampstead

```
db.movies.update( { title : "Pulp Fiction" },
{ $push:
  { actors :
    {$each: [
          { name : "Samuel L. Jackson",  age : 70, children : 1, hometown : "Los Angeles"
},
         { name : "Bruce Willis",  age : 65, children : 5, hometown : "Los Angeles" }]
    }
   }
})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
db.movies.find({title : "Pulp Fiction"}, {title:true, actors:true}).pretty()
```

Task 3.

Use the method ~~save~~ insert or insertOne to generate these documents using variables.

Document	JSON
Pee Wee Herman's Big Adventure	title : Pee Wee Herman's Big Adventure writer : [{ name : Phil Hartman, age : 49, children : 2, hometown : Brantford }, { name : Paul Reubens, age : 66, children : 0, hometown : Peekskill }, {name: Michael Varhol}] year : 1985 rating : 7
Avatar	title : Avatar writer : { name : James Cameron, age : 64, children : 4, hometown : Kapuskasing } year : 2009 rating : 8

```
var p1 = { title : "Pee Wee Herman's Big Adventure", writer : [ { name : "Phil Hartman", age : 49, children : 2, hometown : "Brantford" }, { name : "Paul Reubens", age : 66, children : 0, hometown : "Peekskill" } ], year : 1985, rating : 7 }

 var p2 = { title : "Avatar", writer : { name : "James Cameron", age : 64, children : 4, hometwon : "Kapuskasing" }, year : 2009, rating : 8 }

db.movies.insert([ p1, p2 ])
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 2,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})

db.movies.find().pretty()
```

Task 4.

Set the *rating* to 10 for all the movies where *Brad Pitt* acts.

```
db.movies.update( { "actors.name" : "Brad Pitt" },
      { $set : { rating :  10 } }, { multi : true } )
WriteResult({ "nMatched" : 2, "nUpserted" : 0, "nModified" : 2 })

db.movies.find({ "actors.name" : "Brad Pitt" }, {title:true, rating:true,
actors:true}).pretty()

db.movies.find({ "actors.name" : {$ne:"Brad Pitt"} }, {title:true, rating:true,
actors:true}).pretty()
```

Task 5.

Update the actors of *Inglourious Basterds* and set them in descending order by *name*. Then, show them in ascending order by *age*.

```
db.movies.update( {title : "Inglorious Basterds"},
 { $push: { "actors" : { $each : [ ], $sort: { "name": -1 } } } } )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title : "Inglorious Basterds"}).pretty()

db.movies.update( {title : "Inglorious Basterds"},
 { $push: { "actors" : { $each : [ ], $sort: { "age": 1 } } } } )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title : "Inglorious Basterds"}).pretty()
```

Task 6.

Add the following actors to these documents in a single sentence for each document:

Document - Pee Wee Herman's Big Adventure			
name	**age**	**children**	**hometown**
Paul Reubens	66	0	Peeksill
Elizabeth Ann Guttman	57	2	Los Angeles

```
db.movies.update( { title : "Pee Wee Herman's Big Adventure" },
{ $set :
  { actors : [
     { name : "Paul Reubens", age : 66, children : 0, hometwon : "Peeksill" },
     { name : "Elizabeth Ann Guttman", age : 57, children : 2, hometown : "Los
Angeles" } ]
   }
} )
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
db.movies.find({ title : "Pee Wee Herman's Big Adventure" }).pretty()
```

Document - Avatar			
name	**age**	**children**	**hometown**
Samuel Henry John Worthington	42	0	Los Angeles
Zoe Yadira Saldaña Nazario	40	3	Passaic

```
db.movies.update( { title : "Avatar" },
 { $set :
   { actors : [
        { name : "Samuel Henry John Worthington", age : 42, children : 0, hometwon :
"Los Angeles"},
            {name : "Zoe Yadira Saldaña Nazario", age : 40, children : 3, hometown :
"Passaic" }
     ] }
 })
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title : "Avatar"}).pretty()
```

Task 7.

Remove *John Travolta* from all movies.

```
db.movies.update(
   {},
   {$pull:
      {actors : {name : "John Travolta"}}
   },
   {multi:true}
)
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({“actors.name” : "John Travolta"}).count()
0
```

Task 8.

Remove *Edward Norton* from the movie *Fight Club*.

```
db.movies.update(
   {title: "Fight Club" },
   {$pull:
      {actors : {name : "Edward Norton"}}
   }
)
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title: "Fight Club"}).pretty()
```

Task 9.

Remove *Diane Kruger* and *Eli Roth* from the movie *Inglorious Basterds*.

```
db.movies.update(
   {title: "Inglorious Basterds" },
   {$pull:
      {actors:
        { $or: [{ name:"Diane Kruger" }, { name: "Eli Roth" } ]}
      }
   }
)
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.movies.find({title: "Inglorious Basterds"}).pretty()
```

Task 10.

Remove the movie *Pee Wee Herman's Big Adventure*.

```
db.movies.remove({title: "Pee Wee Herman's Big Adventure"})
WriteResult({ "nRemoved" : 1 })

db.movies.find({title: "Pee Wee Herman's Big Adventure"}).count()
0
```

Task 11.

Remove the first movie where the field *franchise* is equal to *The Hobbit*.

```
db.movies.remove({franchise: "The Hobbit"}, {justOne:true})
WriteResult({ "nRemoved" : 1 })

db.movies.find({franchise: "The Hobbit"}).count()
2
```

Task 12.

Remove the field franchise from the collection *movies*.

```
db.movies.update({}, {$unset : {franchise : 1}}, {multi: true})
WriteResult({ "nMatched" : 6, "nUpserted" : 0, "nModified" : 2 })

db.movies.find({franchise: "The Hobbit"}).count()
0
```

Resources

[MongoDB (official site)]