

Day -04

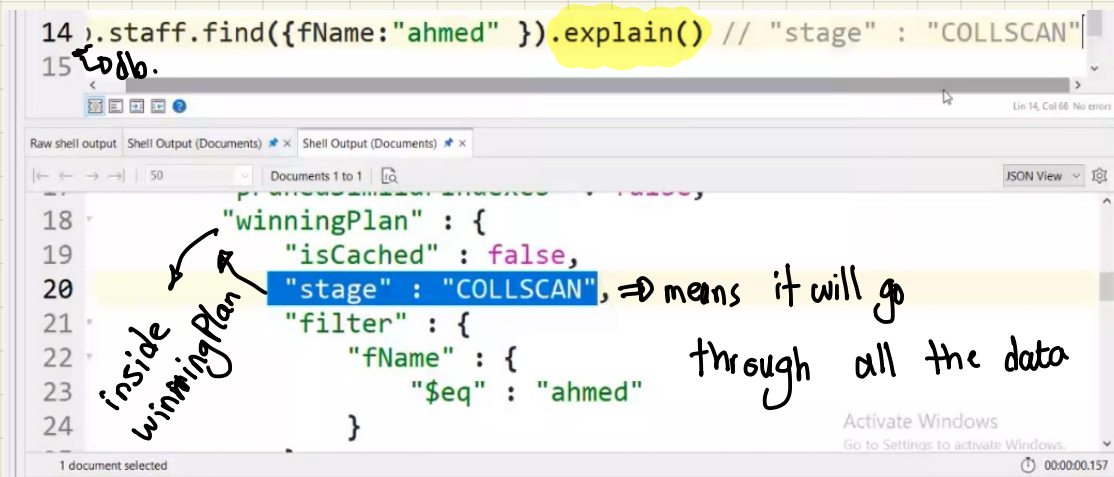
Index is a decision

↳ it helps faster data retrieval **BUT** it has disadvantages it have problems with insert, update and delete.

- when a Customer came to you to tell you that the system is very slow.
- you shouldn't just add indices because you need to understand the other disadvantages before take that decision.
- In that cases you can apply indices to some fields which caused the problem to can get the least of disadvantages

⇒ data page ⇒ output of find

⇒ when use Index it uses binary tree



The screenshot shows a MongoDB shell session. Line 14 contains the command `.staff.find({fName:"ahmed"}).explain() // "stage" : "COLLSCAN"`. Line 15 shows the output of the `explain()` command, which is a JSON document. The document has a `"winningPlan"` field, which is an object containing `"isCached"` (false), `"stage"` ("COLLSCAN"), and `"filter"` (an object with `"fName"` and `"$eq"` fields). A handwritten arrow points from the text "inside winningPlan" to the `"winningPlan"` field. Another handwritten note says "⇒ means it will go through all the data" pointing to the `"stage"` field.

```
14 .staff.find({fName:"ahmed"}).explain() // "stage" : "COLLSCAN"
15 {
  "winningPlan" : {
    "isCached" : false,
    "stage" : "COLLSCAN",
    "filter" : {
      "fName" : {
        "$eq" : "ahmed"
      }
    }
  }
}
```

.explain() ⇒ method explains what happen in the query

special case for the Pk (-id) because it uses IXSCAN

```
localhost:27017 > SW
db.staff.find({fName:"ahmed"}).explain() //"COLLSCAN"
db.staff.createIndex({fName : 1}) //fName_1
db.staff.find({fName:"ahmed"}).explain() | I
```

```
{
  "isCached" : false,
  "stage" : "FETCH",
  "inputStage" : {
    "stage" : "IXSCAN",
    "keyPattern" : {
      "fName" : 1
    }
  }
}
```

index scan

```
Quickstart x Day4_Index.js x Day4_Demo.js x IntelliShell: Local - imported on Aug 27, 2023* x
localhost:27017 > SW
db.staff.createIndex({fName : 1}) //fName_1
db.staff.getIndexes() => show all indexes
db.staff.find({fName:"ahmed"}).explain() //IXSCAN
```

```
{
  "v" : 2.0,
  "key" : {
    "fName" : 1.0
  },
  "name" : "fName_1"
}
```

index name
from 1 & first index

Quickstart x Day4_Index.js x Day4_Demo.js x IntelliShell: Local - imported on Aug 27, 2023* x

localhost:27017 > SW

New Load file Save file Enable Query Assist Change view

```
25 db.staff.createIndex({fName : 1},{name:"IX_Staff_FName"}) //fName
26 db.staff.getIndexes()
27 db.staff.dropIndex("fName_1") => drop index by its name
28 db.staff.find({fName:"ahmed"}).explain() //IXSCAN
29
30
```

Shell Output (x) Shell Output (x) Shell Output (x) Shell Output (x) Shell Output (x) Shell Output (x) Shell Output (x) Shell Output (x)

Documents 1 to 1 JSON View

```
11 {
12   "key" : {
13     "fName" : 1.0
14   },
15   "name" : "IX_Staff_FName"
16 }
```

Activate Windows
Go to Settings to activate Windows.

way of give it a name

=> drop index by its name

```
30 //both column (Componued ind)
31 db.staff.find({fName:"ahmed"}).explain()
32 db.staff.find({lName:"ahmed"}).explain()
33
34 db.staff.find({fName:"malak",lName:"mohamed"}).explain()
35
36 db.staff.createIndex({fName : 1 , lName:1} , {name:"IX_Staff_Nam
37
```

Raw shell output Shell Output (Documents) x Shell Output (Documents) x Shell Output (Documents) x Shell Output (Documents) x

Documents 1 to 1 JSON View

```
27 {
28   "winningPlan" : {
29     "isCached" : false,
30     "stage" : "COLLSCAN",
31     "filter" : {
32       "fName" : "malak",
33       "lName" : "mohamed"
34     }
35   }
36 }
```

Activate Windows
Go to Settings to activate Windows.

```
39 db.staff.find({fName:"malak",lName:"mohamed"}).explain()
40
41 db.staff.getIndexes()
42
```

Raw shell output Shell Output (Docume... x Shell Output (Docume... x Shell Output (Docume... x Shell Output (Docume... x Shell Output (Docume... x Shell Output (Array)

Documents 1 to 1 JSON View

```
9 {
10   "v" : 2.0,
11   "key" : {
12     "fName" : 1.0,
13     "lName" : 1.0
14   }
15 }
```

Displaying 1 document 00:00


```

38
39 db.staff.find({fName:"malak",lName:"mohamed"}).explain()//IX Scan
40 db.staff.find({fName:"malak"}).explain() //IXSCAN
41 db.staff.find({lName:"mohamed"}).explain()//COLLSCAN
42 db.staff.find({lName:"mohamed",fName:"malak"}).explain() //IXSCAN
43
44

```

```

29      "stage" : "FETCH",
30      "inputStage" : {
31        "stage" : "IXSCAN",
32        "keyPattern" : {
33          "fName" : 1.0,
34          "lName" : 1.0

```

Compound Index- Cases

- `db.employee.find({fName:"malak"}).explain() // IXSCAN`
- `db.employee.find({lName:"mohamed"}).explain() // COLLSCAN`
- The index is ordered by fName first, then by lName.
- This means the index is sorted primarily by fName, and within the same fName, it is further sorted by lName.

* we can have var containing all the data to be more reusable "test it in the lab"

example

```

1 var data =
2 [
3 {
4   _id:10 , name:"eman"
5 },
6 {
7   _id:11 , name:"mohamed"
8 }
9 ]
10 db.staff1.insertOne(data)
11 db.staff2.insertMany(data)
12

```

~~data~~ data is displayed as `{ "_id": 2, "fname": "israa", "lname": "abdelghany" }`

I want it to be displayed as
Employee Name : Israa
Employee Name : --- etc

we can do that by:-

looping

var name



`df.stuff.find([{}]).forEach(function(data)`

`{`

`print("Employee Name: " + data.fname)`

`print("Employee LName: " + data.lname)`

`}`

db.serverCmdLineOpts() → print the paths

log, storageetc

on burn data base → each one works on his machine

mongo restore --db Cairo --dir "C/..."

Coll name
I want it to
have after restore

mongo dump --db coll name --out "..."

name of
data I want
to have backup
for