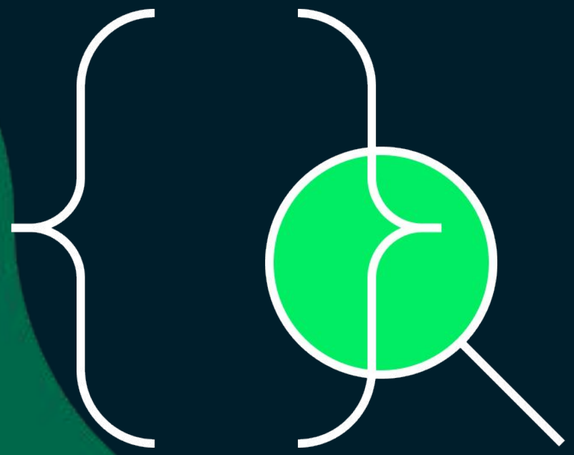
Theory & Demo

Kai Yong Lai (Vandyck)

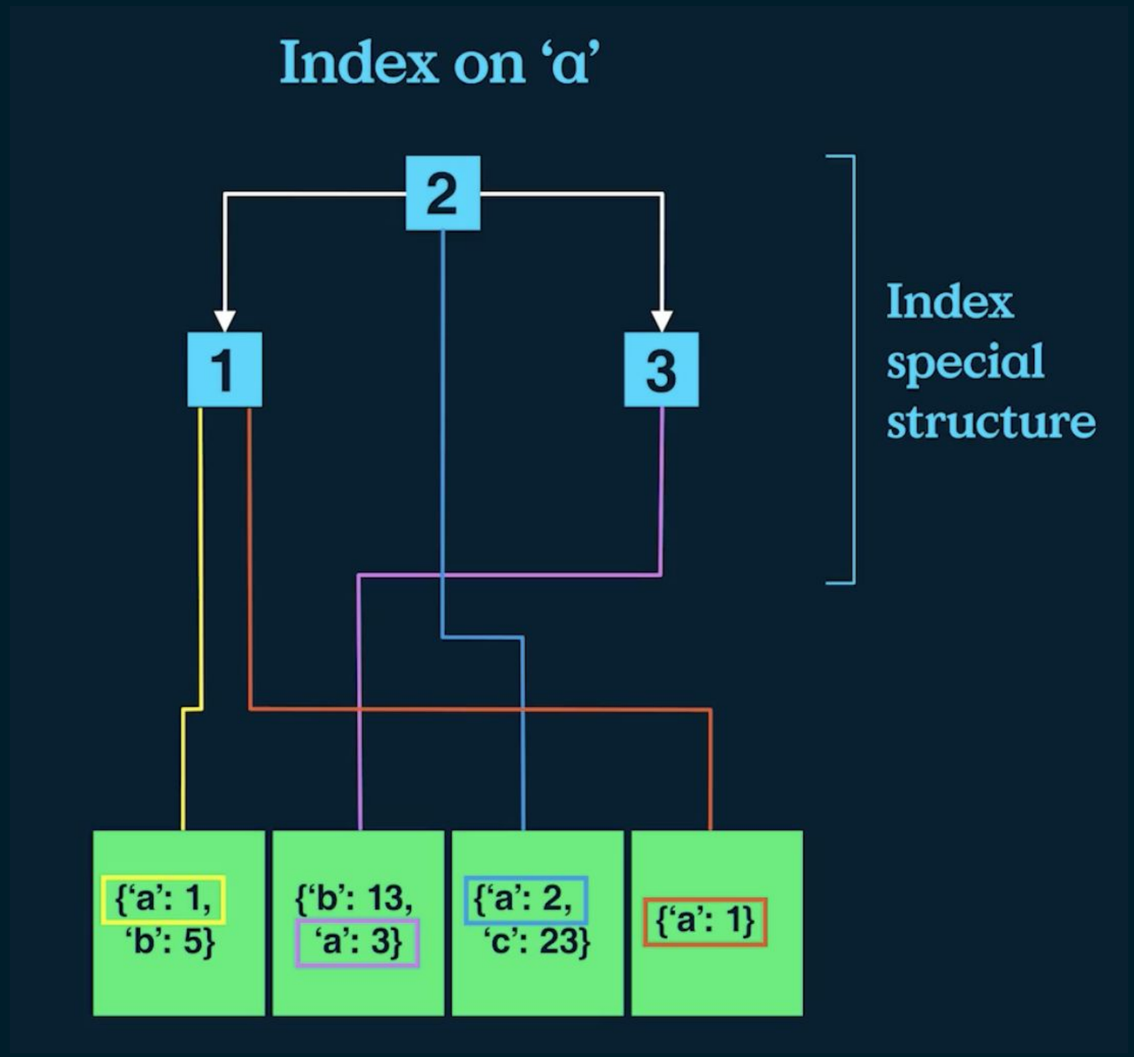
Is your index INDEXING?



MongoDB®

Community Creator + User Group Leader

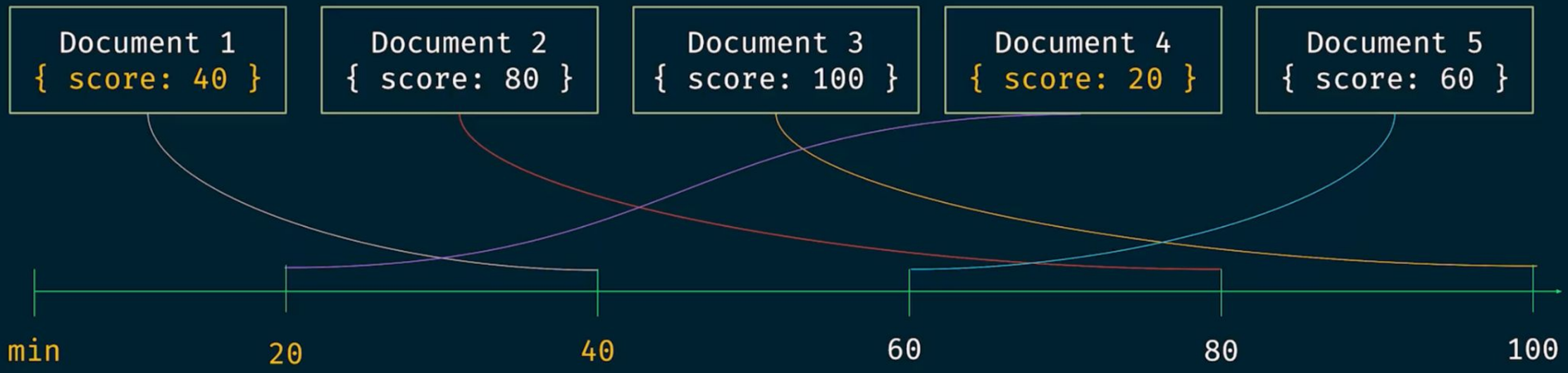
Basic of Indexing



Data structure for **fast RETRIEVAL**

How

Query: { score : { $lte: 40 } }



Index key: { score: 1 }

Why Index

📈 Speed

⬇️ Resource & Disk IO

Equality Operator
+ Range

Single Field

Example:

```
db.comments.createIndex({
  name: 1
})
```

MultiKey

```
_id: ObjectId('59b99db4cfa9a34dcd7885b6')
name : "Ned Stark"
email : "sean_bean@gameofthron.es"
password : "$2b$12$UREFwsRUoyF0CRqGNK0LzO0HM/jLhgUCNN…
```

Example:

```
db.users.createIndex({
  email: 1
})
```

Compound

```
_id: ObjectId('5a9427648b0beebeb69579e7')
name : "Mercedes Tyler"
email : "mercedes_tyler@fakegmail.com"
movie_id : ObjectId('573a1390f29313caabcd4323')
text : "Eius veritatis vero facilis quaerat fuga temporibus.
       Praesentium exped…"
date : 2002-08-18T04:56:07.000+00:00
```

Example:

```
db.comments.createIndex({
  name: 1,
  email: 1
})
```

WildCard

Example:

```
db.theaters.createIndex({
    "location.address.$**" : 1
})
```

Partial Index

```
_id: ObjectId('573a1390f29313caabcd5293')
plot : "Young Pauline is left a lot of money when her w
       Howe…"
▸ genres : Array (1)
runtime : 199
▸ cast : Array (4)
num_mflix_comments : 0
poster : "https://m.media-
         amazon.com/images/M/MV5BMzgxODk1Mzk2Ml5BMl5Ban
title : "The Perils of Pauline"
fullplot : "Young Pauline is left a lot of money when h
           dies. Howe…"
▸ languages : Array (1)
released : 1914-03-23T00:00:00.000+00:00
▸ directors : Array (2)
▸ writers : Array (5)
▸ awards : Object
lastupdated : "2015-09-12 00:01:18.647000000"
year : 1914
▸ imdb : Object
```

Example:

```
db.movies.createIndex(
    {
        “year”: 1
    },
    {
        “partialFilterExpression”:
        {
            "year”: {
                $gte: 1910
            }
        }
    }
)
```

Collation

```
_id: ObjectId('6778391eb518e48f1177c46e')
title : "Token Balance Validator"
description : "Create a smart contract that
              implements a basic token balance
              checking…"
difficulty : "Easy"
createdAt : 2025-01-03T19:23:10.400+00:00
testCases : "// SPDX-License-Identifier: MIT
                   pragma solidity ^0.8.0
                  …"
solution : "// SPDX-License-Identifier: MIT
           pragma solidity ^0.8.0;

           contract Toke…"
```

Example:

```
db.problems.createIndex(
    { “difficulty”: 1 },
    {
        "collation”: {
            “locale”: “en”,
            “strength”: 2
        }
    }
)
```

Index Golden Rule

Equality -> Sort -> Range

```
db.customers.find({
  birthdate: {
    $gte:ISODate("1977-01-01")
  },
  active:true
}).sort({
  name:1
})
```

```
db.customers.createIndex({
  active:1,
  name:1,
  birthdate: 1

})
```

Inspection Mode

- `queryPlanner`, which details the plan selected by the **query optimizer** and lists the rejected plans.
- `executionStats`, which details the execution of the winning plan and the rejected plans.
- `serverInfo`, which provides information on the MongoDB instance.

`allPlansExecution` Mode

By default, `explain` runs in `"allPlansExecution"` verbosity mode. The following `explain` command returns the `queryPlanner` and `executionStats` for all considered plans for an `update` command:

Stage

- `COLLSCAN` for a collection scan
- `IXSCAN` for scanning index keys
- `FETCH` for retrieving documents
- `GROUP` for grouping documents
- `SHARD_MERGE` for merging results from shards
- `SHARDING_FILTER` for filtering out orphan documents from shards
- `TS_MODIFY` for modifying a time series collection
- `BATCHED_DELETE` for multiple document deletions that are batched together internally (starting in MongoDB 6.1)
- `EXPRESS` stages for a limited set of queries that can bypass regular query planning to use optimized index scan plans *(New in version 8.0.)*

  `EXPRESS` stages can be one of the following:

  - `EXPRESS_CLUSTERED_IXSCAN`
  - `EXPRESS_DELETE`
  - `EXPRESS_IXSCAN`
  - `EXPRESS_UPDATE`

Code