

Indexes in MongoDB

Indexes

Indexes support the efficient execution of queries in MongoDB. Without indexes, MongoDB must perform a *collection scan*, i.e. scan every document in a collection, to select those documents that match the query statement. If an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.

```
school> db.students.find({name: "Name2"}).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'school.students',
    parsedQuery: { name: { '$eq': 'Name2' } },
    indexFilterSet: false,
    queryHash: 'F4DDDCDC',
    planCacheShapeHash: 'F4DDDCDC',
    planCacheKey: 'E45FBFA1',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { name: { '$eq': 'Name2' } },
      direction: 'forward'
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 1,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 9,
    executionStages: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { name: { '$eq': 'Name2' } },
      direction: 'forward'
    }
  }
}
```

```

nReturned: 1,
executionTimeMillisEstimate: 0,
works: 10,
advanced: 1,
needTime: 8,
needYield: 0,
saveState: 0,
restoreState: 0,
isEOF: 1,
direction: 'forward',
docsExamined: 9
}
},
queryShapeHash:
'61EF2197747739FAB45926D45CAE6F1E8A07DC4847027B8CF1E4AFDF426D2B56',
Comm...

```

docsExamined: 9 <- Meaning it has examined 9 documents(like linear search), If the search exceeds many documents.

We can speed up the lookup process by applying indexes.

To apply an index to the field:

Name: 1 <- for ascending order(a to z)

Name: -1 <- for descending order(z to a)

Name of the index:

```

school> db.students.createIndex({name: 1})
name_1 ←
school>

```

```

school> db.students.find({name: "Name2"}).explain("executionStats")

```

```

{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'school.students',
    parsedQuery: { name: { '$eq': 'Name2' } },
    indexFilterSet: false,
    queryHash: 'F4DDDCDC',
    planCacheShapeHash: 'F4DDDCDC',
    planCacheKey: '34C29116',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,

```

```

prunedSimilarIndexes: false,
winningPlan: {
  isCached: false,
  stage: 'FETCH',
  inputStage: {
    stage: 'IXSCAN',
    keyPattern: { name: 1 },
    indexName: 'name_1',
    isMultiKey: false,
    multiKeyPaths: { name: [] },
    isUnique: false,
    isSparse: false,
    isPartial: false,
    indexVersion: 2,
    direction: 'forward',
    indexBounds: { name: [ ["Name2", "Name2"] ] }
  }
},
rejectedPlans: []
},
executionStats: {
  executionSuccess: true,
  nReturned: 1,
  executionTimeMillis: 1,
  totalKeysExamined: 1,
  totalDocsExamined: 1,
  executionStages: {
    isCached: false,
    stage: 'FETCH',
    nReturned: 1,
    executionTimeMillisEstimate: 0,
    works: 2,
    advanced: 1,
    needTime: 0,
    needYield: 0,
    saveState: 0,
    restoreState: 0,
    isEOF: 1,
    docsExamined: 1,
    alreadyHasObj: 0,
    inputStage: {

```

Now the document examined is one.(like docsExamined: 1).

Which is lot less time consuming then examining 9 documents.

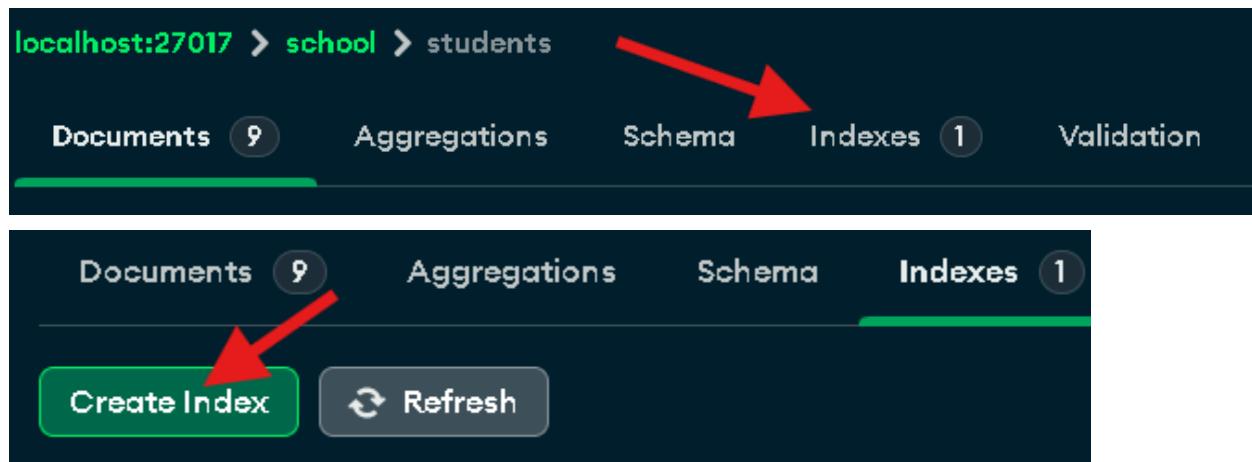
To Get all the indexes:

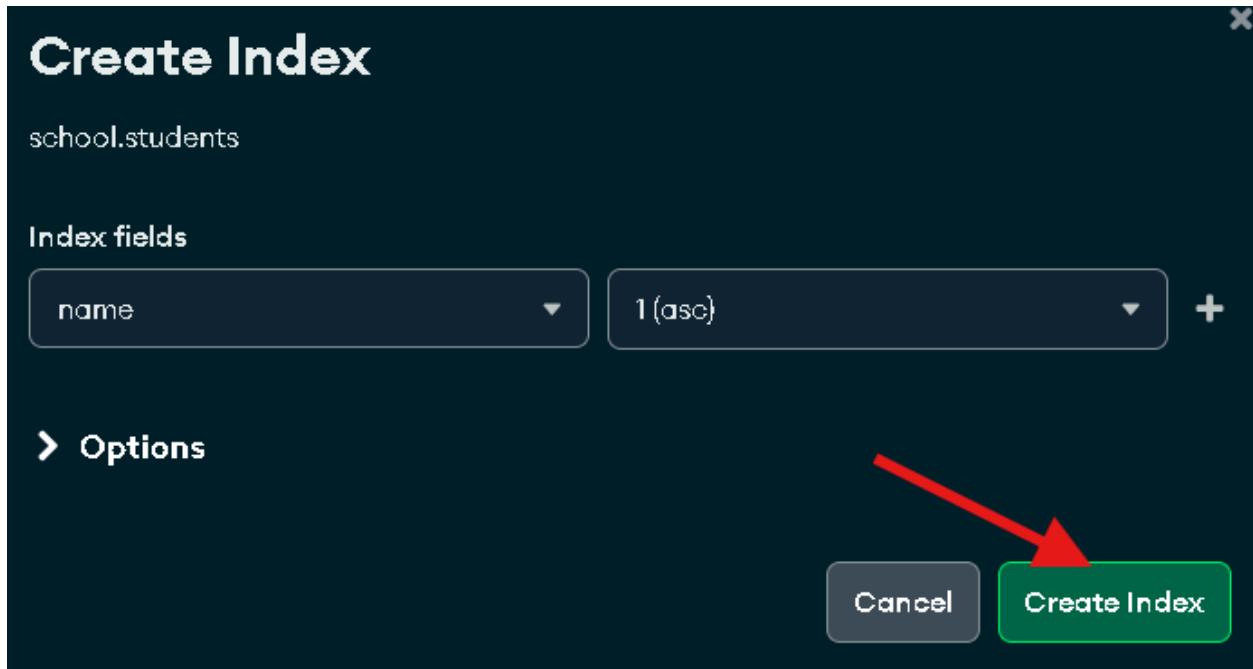
```
school> db.students.getIndexes()  
[  
  { v: 2, key: { _id: 1 }, name: '_id_' },  
  { v: 2, key: { name: 1 }, name: 'name_1' }  
]
```

To Drop an Index:

```
school> db.students.getIndexes()  
[  
  { v: 2, key: { _id: 1 }, name: '_id_' },  
  { v: 2, key: { name: 1 }, name: 'name_1' }  
]  
school> db.students.dropIndex("name_1")  
{ nIndexesWas: 2, ok: 1 }  
school> db.students.getIndexes()  
[ { v: 2, key: { _id: 1 }, name: '_id_' } ]
```

To create an Index in the Compass:





To drop the index: (click on trash icon)

The screenshot shows the 'Indexes' tab in the dashboard. It lists two indexes: '_id_' (Regular, 36.9 kB, 3 usage, Unique, Ready) and 'name_1' (Regular, 20.5 kB, 0 usage, Unique, Ready). A red arrow points to the trash icon next to the 'name_1' index entry, indicating where to click to drop it. A tooltip 'Drop Index name_1' is visible near the trash icon.

The screenshot shows a confirmation dialog titled 'Drop Index' with a warning icon. It asks 'Are you sure you want to drop index "name_1"?'. Below that, it says 'Type "name_1" to confirm your action'. A text input field contains 'name_1'. A red arrow points from the text 'Type "name_1" to confirm your action' to the 'Drop' button, which is highlighted with a red background.

Use indexes wisely, recommended do it while searching instead of updating.