

DB Optimization – Cubicles

1) Find companies by a user:






Query used: `db.companies.find({ userId: ObjectId('67f37b5d433182c2e3d08972') })`

For Route: `// Route to get all companies for the authenticated user (GET method)`
`router.route("/get").get(isAuthenticated, companyController.getCompany);`

- Before optimization:

```
{
  "isCached": false,
  "stage": "COLLSCAN",
  "filter": {
    "userId": {
      "$eq": "67f37b5d433182c2e3d08972"
    }
  },
  "nReturned": 2,
  "executionTimeMillisEstimate": 3,
  "works": 323,
  "advanced": 2,
  "needTime": 320,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "direction": "forward",
  "docsExamined": 322
}
```

Query Performance Summary






-  **2** documents returned
-  **323** documents examined
-  **3 ms** execution time
-  **Is not** sorted in memory
-  **0** index keys examined

 **No index available for this query.** 

- After Optimization:

```
{
  "isCached": false,
  "stage": "FETCH",
  "nReturned": 2,
  "executionTimeMillisEstimate": 0,
  "works": 3,
  "advanced": 2,
  "needTime": 0,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "docsExamined": 2,
  "alreadyHasObj": 0
}
```

Query Performance Summary

-  **2 documents returned**
-  **2 documents examined**
-  **0 ms execution time**
-  **Is not sorted in memory**
-  **2 index keys examined**

Query used the following index:

userId ↑

2) Find jobs created by a recruiter

Query used:

```
db.companies.find(
  { created_by: ObjectId('67f37b5d433182c2e3d08972') },
  { createdAt: -1 }
)
```

For Route: *// Route to get all companies for the authenticated user (GET method)*
`router.route("/get").get(isAuthenticated, companyController.getCompany);`

- Before optimization:

```
{
  "isCached": false,
  "stage": "COLLSCAN",
  "filter": {
    "userId": {
      "$eq": "67f37b5d433182c2e3d08972"
    }
  },
  "nReturned": 2,
  "executionTimeMillisEstimate": 0,
  "works": 324,
  "advanced": 2,
  "needTime": 321,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "direction": "forward",
  "docsExamined": 323}
```

Query Performance Summary

- 2 documents returned
- 323 documents examined
- 0 ms execution time
- Is not sorted in memory
- 0 index keys examined
- No index available for this query.**

- After Optimization:

```
{ "isCached": false,
  "stage": "FETCH",
  "nReturned": 0,
  "executionTimeMillisEstimate": 0,
  "works": 1,
  "advanced": 0,
  "needTime": 0,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "docsExamined": 0,
  "alreadyHasObj": 0}
```

Query Performance Summary

- 2 documents returned
- 2 documents examined
- 0 ms execution time
- Is not sorted in memory
- 2 index keys examined

Query used the following index:

userId ↑

3) Find all recruiters

Query used:








```
db.users.find({"role": "recruiter"})
```

For Route: *//Route to get all the recruiters*
router.get('/getAllRecruiters', getAllRecruiters);

- Before optimization:

```
{
  "isCached": false,
  "stage": "PROJECTION_SIMPLE",
  "nReturned": 301,
  "executionTimeMillisEstimate": 6,
  "works": 1004,
  "advanced": 301,
  "needTime": 702,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "transformBy": {
    "fullName": 1,
    "email": 1,
    "phoneNumber": 1
  }
}
```

Query Performance Summary

-  **301** documents returned
-  **1003** documents examined
-  **6 ms** execution time
-  **Is not** sorted in memory
-  **0** index keys examined
-  **No index available for this query.** 

- After Optimization:






```
{
  "isCached": false,
```

```

"stage": "PROJECTION_SIMPLE",
"nReturned": 301,
"executionTimeMillisEstimate": 0,
"works": 302,
"advanced": 301,
"needTime": 0,
"needYield": 0,
"saveState": 0,
"restoreState": 0,
"isEOF": 1,
"transformBy": {
  "fullName": 1,
  "email": 1,
  "phoneNumber": 1
}
}

```

Query Performance Summary

-  **301** documents returned
-  **301** documents examined
-  **0 ms** execution time
-  **Is not** sorted in memory
-  **301** index keys examined

Query used the following index:

role ↑

4) Generate report by finding the user and finding all his applications

Query used:

```

db.applications.find({ applicant: userId })
  .populate({
    path: 'job',
    populate: { path: 'company' } // Populate the company field
  });

```








For Route: // Route to generate PDF report for authenticated user

```
router.get('/report', isAuthenticated, generateUserReport);
```

- Before optimization:

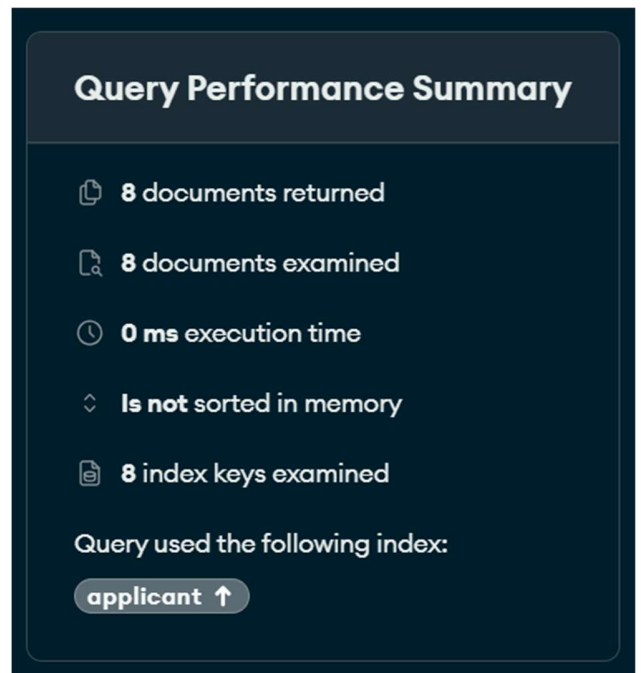
```
{
  "isCached": false,
  "stage": "COLLSCAN",
  "filter": {
    "applicant": {
      "$eq": "67f37b5d433182c2e3d08a78"
    }
  },
  "nReturned": 8,
  "executionTimeMillisEstimate": 4,
  "works": 5229,
  "advanced": 8,
  "needTime": 5220,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "direction": "forward",
  "docsExamined": 5228
}
```

Query Performance Summary

-  **8** documents returned
-  **5231** documents examined
-  **4 ms** execution time
-  **Is not** sorted in memory
-  **0** index keys examined
-  **No index available for this query.** 

- After Optimization:

```
{
  "isCached": false,
  "stage": "FETCH",
  "nReturned": 8,
  "executionTimeMillisEstimate": 0,
  "works": 9,
  "advanced": 8,
  "needTime": 0,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "docsExamined": 8,
  "alreadyHasObj": 0}
```



5) Checking if a company is already present before registering a company

Query used:

```
db.companies.findOne({ name: companyName })
```

For Route: // Route to register a company (POST method)

```
router.route("/register").post(isAuthenticated, companyController.registerCompany);
```








- Before optimization:

```

{
  "isCached": false,
  "stage": "COLLSCAN",
  "filter": {
    "name": {
      "$eq": "Vertex 452"
    }
  },
  "nReturned": 1,
  "executionTimeMillisEstimate": 0,
  "works": 323,
  "advanced": 1,
  "needTime": 321,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "direction": "forward",
  "docsExamined": 322
}

```

Query Performance Summary

-  **1** documents returned
-  **322** documents examined
-  **0 ms** execution time
-  **Is not** sorted in memory
-  **0** index keys examined
-  **No index available for this query.** 

- **After Optimization:**

```

{
  "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  
}
```

Query Performance Summary

- 1 documents returned
- 1 documents examined
- 0 ms execution time
- Is **not** sorted in memory
- 1 index keys examined

Query used the following index:

name ↑

6) Limiting blogs with pagination

Query used:

```
db.blogs.find()
```

For Route: *// Route to get all blogs available*
router.get("/getPosts", getAllBlogs);

- Before optimization:

```

{
  "isCached": false,
  "stage": "COLLSCAN",
  "nReturned": 1000,
  "executionTimeMillisEstimate": 0,
  "works": 1001,
  "advanced": 1000,
  "needTime": 0,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "direction": "forward",
  "docsExamined": 1000
}

```

Query Performance Summary

- 1000 documents returned
- 1000 documents examined
- 0 ms execution time
- Is not sorted in memory
- 0 index keys examined
- No index available for this query.**

Status: 200 OK Size: 750 KB Time: 1.62 s

Response

Headers 20 Cookies Results Docs

```

1  [
2  {
3    "_id": "67f37fbc97408ffeb77b7784",
4    "title": "How Social Media Trends is Revolutionizing the Industry",
5    "content": "The conversation around Social Media Trends is growing,
               with industry leaders sharing insights and forecasting trends
               that could redefine the future. Its influence spans multiple
               sectors and disciplines.\n\nIn today's rapidly evolving digital
               landscape, Social Media Trends has emerged as a pivotal area of
               innovation. Experts believe that the transformative potential of
               {subject} is only just beginning to be realized.",
6    "author": {
7      "_id": "67f37b5d433182c2e3d08959",
8      "fullname": "Amelia Allen",
9      "email": "ameliaallen14@example.com"
10   },
11   "tags": [
12     "Social Media Trends",
13     "User Experience Design"
14   ]

```

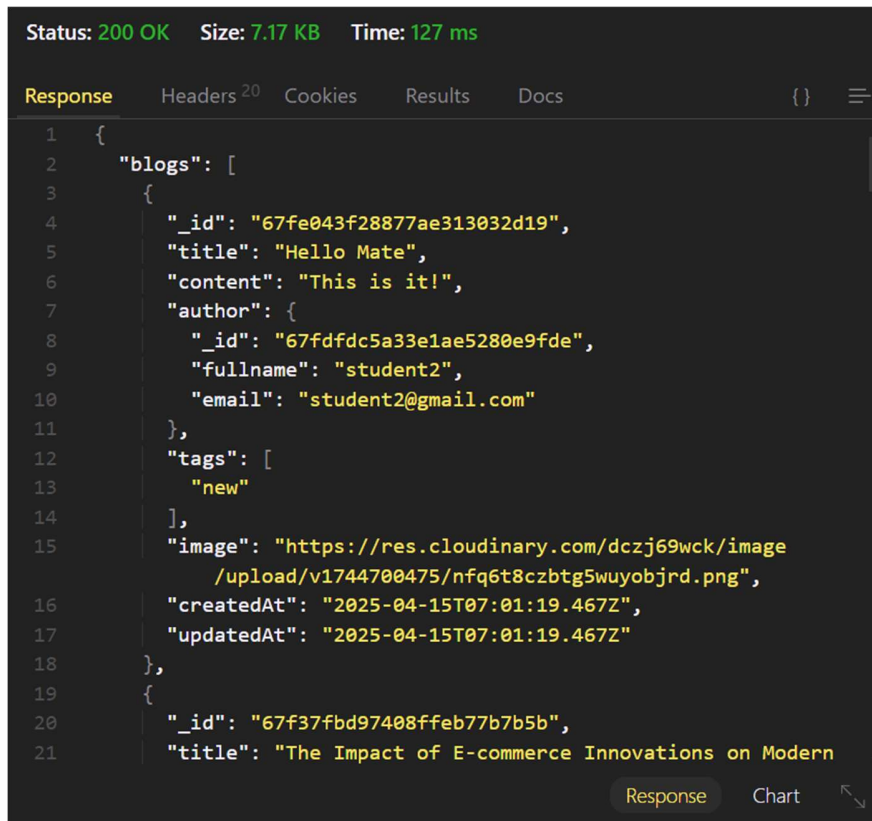
Response

Chart

- After Optimization:

Pagination implemented:

Time reduction: 1.62 seconds to 0.127 seconds !!



```
Status: 200 OK Size: 7.17 KB Time: 127 ms
Response Headers 20 Cookies Results Docs {}
1 {
2   "blogs": [
3     {
4       "_id": "67fe043f28877ae313032d19",
5       "title": "Hello Mate",
6       "content": "This is it!",
7       "author": {
8         "_id": "67fdfdc5a33e1ae5280e9fde",
9         "fullname": "student2",
10        "email": "student2@gmail.com"
11      },
12      "tags": [
13        "new"
14      ],
15      "image": "https://res.cloudinary.com/dczj69wck/image
16        /upload/v1744700475/nfq6t8czbtg5wuyobjrd.png",
17      "createdAt": "2025-04-15T07:01:19.467Z",
18      "updatedAt": "2025-04-15T07:01:19.467Z"
19    },
20    {
21      "_id": "67f37fbd97408ffeb77b7b5b",
22      "title": "The Impact of E-commerce Innovations on Modern
```

7) Get applicant count of a company

Query used:








```
db.companies.find({name: "Vertex 452"})
```

For Route: *// Get number of applicants of a company*
router.get('/getApplicantCountsOfEachCompany',
applicationController.getApplicantCountsOfEachCompany);

- Before optimization:

```
{
  "isCached": false,
  "stage": "COLLSCAN",
  "filter": {
    "name": {
      "$eq": "Vertex 452"
    }
  },
  "nReturned": 1,
  "executionTimeMillisEstimate": 0,
  "works": 324,
  "advanced": 1,
  "needTime": 322,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "direction": "forward",
  "docsExamined": 323}
```

Query Performance Summary

-  **1** documents returned
-  **323** documents examined
-  **0 ms** execution time
-  **Is not** sorted in memory
-  **0** index keys examined
-  **No index available for this query.** 

- After Optimization:

```
{
  "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}
```

Query Performance Summary

- 1 documents returned
- 1 documents examined
- 1 ms execution time
- Is not sorted in memory
- 1 index keys examined

Query used the following index:

name ↑

8) To get applied jobs of a user

Query used:

```
db.applications.find({ applicant: userId }).sort({ createdAt: -1 })
```

For Route: // Route to get applied jobs (GET method)

```
router.route("/get").get(isAuthenticated, applicationController.getAppliedJobs);
```

- Before optimization:

```
{
```

```

"stage": "COLLSCAN",
"filter": {
  "applicant": {
    "$eq": "67f37b5d433182c2e3d08a8e"
  }
},
"nReturned": 10,
"executionTimeMillisEstimate": 3,
"works": 5232,
"advanced": 10,
"needTime": 5221,
"needYield": 0,
"saveState": 0,
"restoreState": 0,
"isEOF": 1,
"direction": "forward",
"docsExamined": 5231
}

```

- After Optimization:

```

{
  "isCached": false,
  "stage": "FETCH",
  "nReturned": 10,
  "executionTimeMillisEstimate": 0,
  "works": 11,
  "advanced": 10,
  "needTime": 0,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "docsExamined": 10,
  "alreadyHasObj": 0
}

```

Query Performance Summary

- 10 documents returned
- 5231 documents examined
- 3 ms execution time
- Is sorted in memory
- 0 index keys examined
- No index available for this query.**

Query Performance Summary

- 10 documents returned
- 10 documents examined
- 0 ms execution time
- Is not sorted in memory
- 10 index keys examined

Query used the following index:

applicant ↑

9) Applying a job by finding applicant and job

Query used:

```
db.applications.find({ applicant:
  ObjectId("67f37b5d433182c2e3d08a79")}).sort({ createdAt: -1 })
```






For Route: *// Route to get applied jobs (GET method)*

```
router.route("/get").get(isAuthenticated, applicationController.getAppliedJobs);
```

- Before optimization:

```
{
  "isCached": false,
  "stage": "SORT",
  "nReturned": 10,
  "executionTimeMillisEstimate": 3,
  "works": 5243,
  "advanced": 10,
  "needTime": 5232,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "sortPattern": {
    "createdAt": -1
  },
  "memLimit": 33554432,
  "type": "simple",
  "totalDataSizeSorted": 1616,
  "usedDisk": false,
  "spills": 0,
  "spilledDataStorageSize": 0
}
```

Query Performance Summary

-  **10** documents returned
-  **5231** documents examined
-  **3 ms** execution time
-  **Is** sorted in memory
-  **0** index keys examined

 **No index available for this query.** 

After Optimization:

```
{
  "stage": "FETCH",
  "nReturned": 10,
  "executionTimeMillisEstimate": 0,
  "works": 11,
  "advanced": 10,
  "needTime": 0,
  "needYield": 0,
  "saveState": 0,
  "restoreState": 0,
  "isEOF": 1,
  "docsExamined": 10,
  "alreadyHasObj": 0
}
```

Query Performance Summary

- 10 documents returned
- 10 documents examined
- 0 ms execution time
- Is sorted in memory
- 10 index keys examined

Query used the following index:

applicant ↑

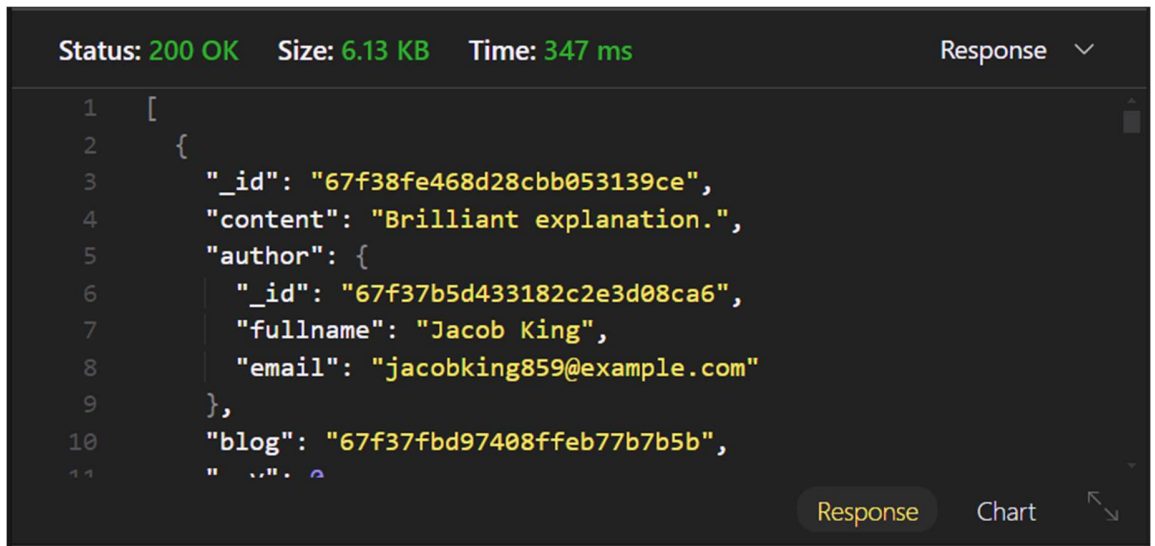
10) Limiting comments and optimizing number of comments fetched

Query used:

```
db.comments.find({ blog: blogId })
  .sort({ createdAt: -1 })
  .skip(skip)
  .limit(limit)
  .populate("author", "fullname email");
```

For Route: `router.get("/:blogId/comments", getCommentsByBlogId); // working`

- Before optimization:

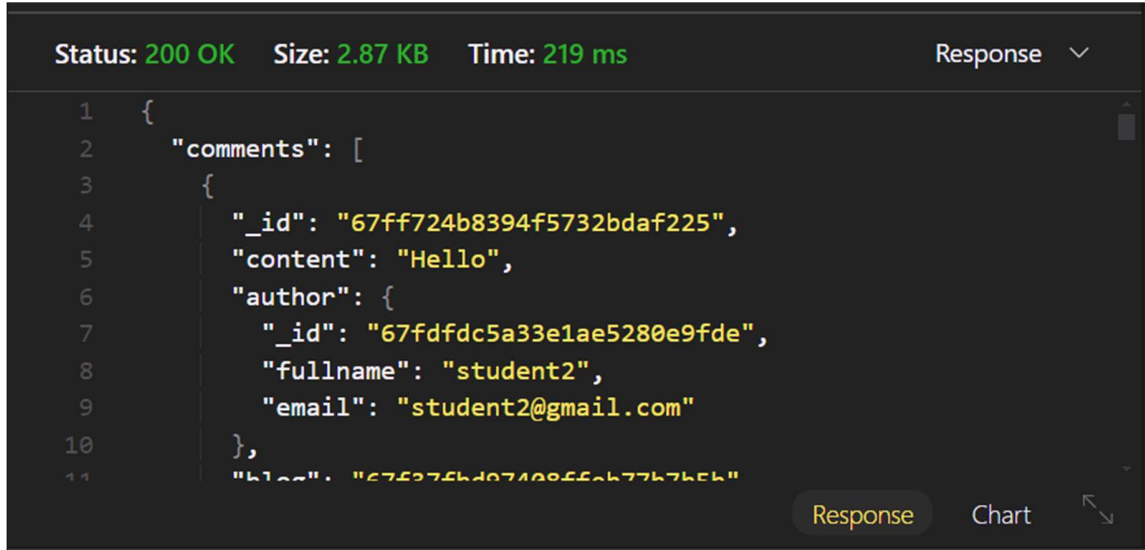


A screenshot of a REST client interface. At the top, it shows 'Status: 200 OK', 'Size: 6.13 KB', and 'Time: 347 ms'. The response is a JSON array with one object. The object has fields: '_id' (a long hex string), 'content' ('Brilliant explanation.'), 'author' (an object with '_id', 'fullname' 'Jacob King', and 'email' 'jacobking859@example.com'), and 'blog' (another long hex string). The interface includes a 'Response' dropdown and buttons for 'Response' and 'Chart'.

```
1  [  
2    {  
3      "_id": "67f38fe468d28cbb053139ce",  
4      "content": "Brilliant explanation.",  
5      "author": {  
6        "_id": "67f37b5d433182c2e3d08ca6",  
7        "fullname": "Jacob King",  
8        "email": "jacobking859@example.com"  
9      },  
10     "blog": "67f37fbd97408ffeb77b7b5b",  
11     " ..."
```

- After Optimization:

- ``${BLOG_API_END_POINT}/${id}/comments?skip=${skip}&limit=${COMMENTS_LIM
IT}``



A screenshot of a REST client interface. At the top, it shows 'Status: 200 OK', 'Size: 2.87 KB', and 'Time: 219 ms'. The response is a JSON object with a 'comments' array containing one object. This object has fields: '_id' (a long hex string), 'content' ('Hello'), 'author' (an object with '_id', 'fullname' 'student2', and 'email' 'student2@gmail.com'), and 'blog' (a long hex string). The interface includes a 'Response' dropdown and buttons for 'Response' and 'Chart'.

```
1  {  
2    "comments": [  
3      {  
4        "_id": "67ff724b8394f5732bdaf225",  
5        "content": "Hello",  
6        "author": {  
7          "_id": "67fdfdc5a33e1ae5280e9fde",  
8          "fullname": "student2",  
9          "email": "student2@gmail.com"  
10        },  
11        "blog": "67f37fbd97408ffeb77b7b5b"
```

@Cubicles IIITS 2025

END