

ServerLogsNoSQL

A REST API project in Flask & MongoDB for post-graduate class Database Management Systems

Authors

- [Papadopoulos Christos-Charalampos](#)
- [Papamichalopoulos Marios](#)

Tools Used

- Python 3.6.9
- Flask 1.1.1
- MongoDB Community Server 4.2.2
- Faker
- Postman

How to run

- populate the database by running the scripts:

```
populateAdmins.py
```

and

```
populateLogs.py
```

inside `logs` directory

- traverse inside the directory
- type in your terminal:

```
. flask-mongodb/bin/activate
```

or set up a virtual environment downloading:

```
flash flask-pymongo
```

- start the REST API:

```
flask run
```

- app is up and running on

Schema Design

Our database consists of two primary collections:

- log
- admin

The first one contains all types of logs. We felt it was not correct design principal to furtherly normalize our log collection, since the point of NoSQL is to keep normalization at a minimum. Another reason is that lookups are really costly and one wants to avoid it at all costs.

As a result we have merged all the types of logs into one collection, to avoid using joins for the queries.

The second collection, contains all the admin related data as mentioned in the requirements of the project. In more detail, each admin owns the following properties:

- username
- email
- telephone
- an array of the upvotes casted

All the admin data have been generated using Faker.

Queries

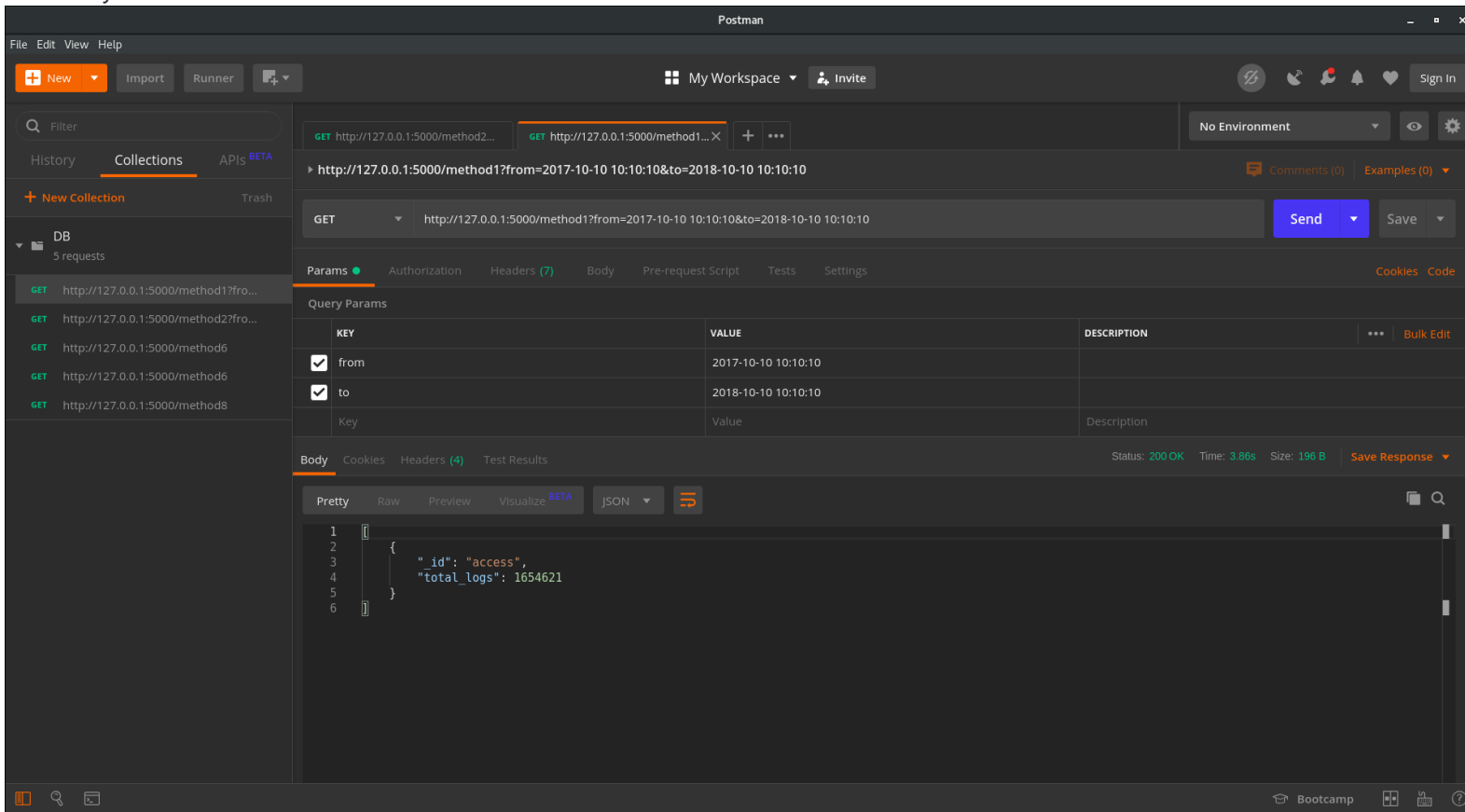
All the queries can be found at:

```
./ServerLogsFlask/ServerLogsNoSQL/methods.py
./ServerLogsFlask/ServerLogsNoSQL/insert.py
```

Sample Snapshots

We present some sample snapshots without the use of indeces:

• API Query 1



The screenshot shows the Postman application interface. On the left sidebar, under the 'Collections' tab, there is a collection named 'DB' containing 5 requests. The first request is selected, showing its details in the main panel. The request is a GET method to the URL 'http://127.0.0.1:5000/method1?from=2017-10-10 10:10:10&to=2018-10-10 10:10:10'. The 'Params' tab is active, displaying a table of query parameters. The response body is shown in the 'Body' tab, displaying a JSON object with 'id' and 'total_logs' fields. The status is 200 OK, time is 3.86s, and size is 196 B.

GET http://127.0.0.1:5000/method1?from=2017-10-10 10:10:10&to=2018-10-10 10:10:10

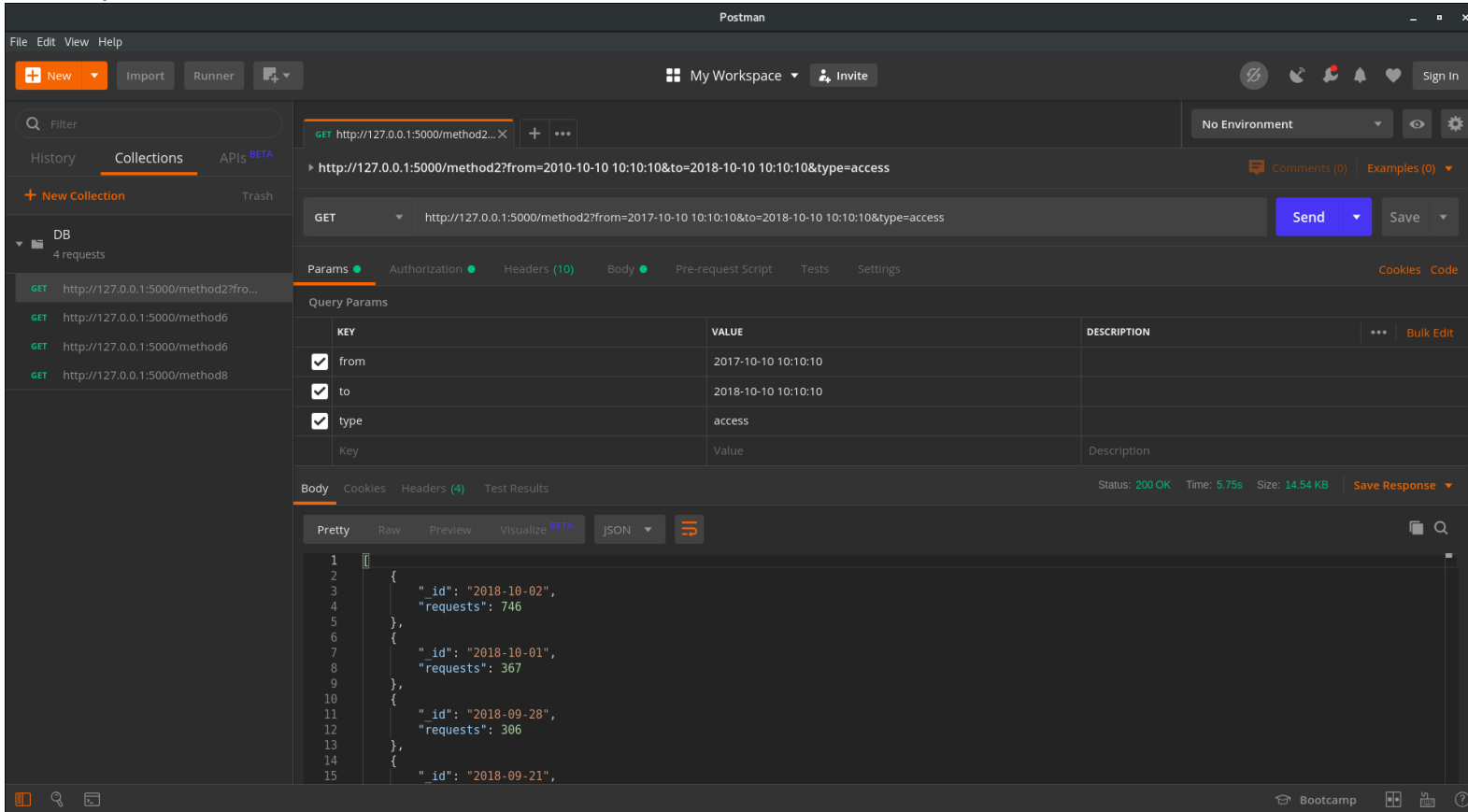
Params

KEY	VALUE	DESCRIPTION
from	2017-10-10 10:10:10	
to	2018-10-10 10:10:10	

Body

```
{  "id": "access",  "total_logs": 1654621}
```

• API Query 2



The screenshot shows the Postman application interface. On the left sidebar, under the 'Collections' tab, there is a collection named 'DB' containing 4 requests. The first request is selected, showing its details in the main panel. The request is a GET method to the URL 'http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access'. The 'Params' tab is active, displaying a table of query parameters. The response body is shown in the 'Body' tab, displaying a JSON array of objects with 'id' and 'requests' fields. The status is 200 OK, time is 5.75s, and size is 14.54 KB.

GET http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access

Params

KEY	VALUE	DESCRIPTION
from	2017-10-10 10:10:10	
to	2018-10-10 10:10:10	
type	access	

Body

```
[  {    "id": "2018-10-02",    "requests": 746  },  {    "id": "2018-10-01",    "requests": 367  },  {    "id": "2018-09-28",    "requests": 306  },  {    "id": "2018-09-21",    "requests": 306  }]
```

- API Query 4

The screenshot displays the Postman application interface. At the top, the menu bar includes File, Edit, View, and Help. Below the menu, there are buttons for 'New', 'Import', and 'Runner'. The main workspace shows 'My Workspace' and an 'Invite' button. On the right, there are icons for environment, history, and settings, along with a 'Sign In' button.

The left sidebar shows a 'Collections' tab with a search filter. Under 'Collections', there is a 'DB' folder containing 4 requests. The requests are listed as follows:

- GET method1
- GET method2
- GET method3
- GET http://127.0.0.1:5000/method8

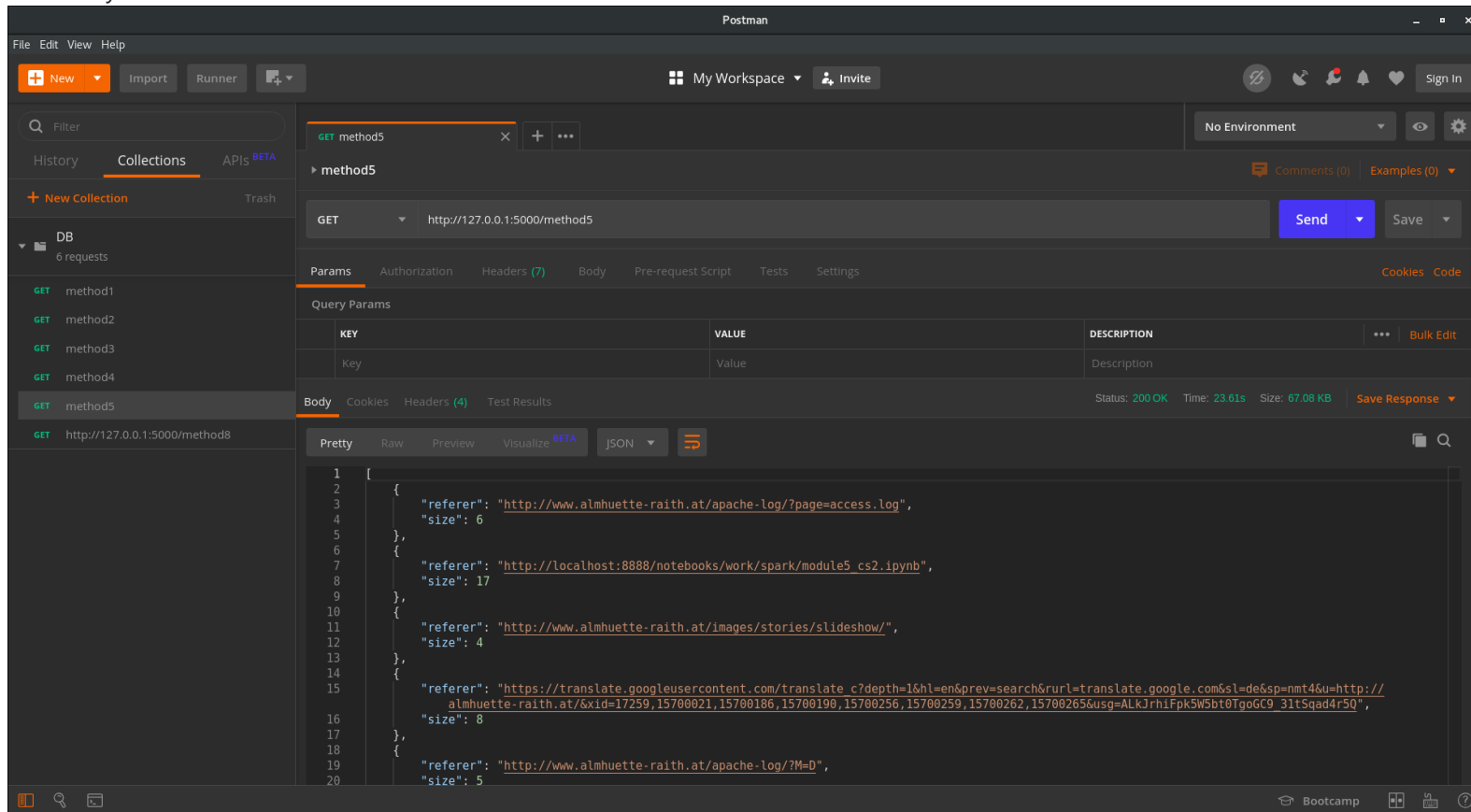
The main area shows an 'Untitled Request' for a GET method to the URL 'http://127.0.0.1:5000/method4?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10'. The 'Send' button is highlighted in blue. Below the URL bar, there are tabs for 'Params', 'Authorization', 'Headers (7)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Params' tab is active, showing a table of query parameters:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2010-10-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
Key	Value	Description

Below the table, there are tabs for 'Body', 'Cookies', 'Headers (4)', and 'Test Results'. The 'Body' tab is active, showing the response status '200 OK', time '10.74s', and size '213 B'. The response body is displayed in JSON format:

```
1 {
2   {
3     "id": "get",
4     "total": 1
5   },
6   {
7     "id": "SEARCH",
8     "total": 1
9   }
10 }
```

• API Query 5



The screenshot shows the Postman application interface. The left sidebar displays a collection named 'DB' with 6 requests. The main panel shows a GET request to 'http://127.0.0.1:5000/method5'. The response is a JSON array of 5 objects, each containing 'referer' and 'size' fields. The status is 200 OK, and the response size is 67.08 KB.

GET method5

URL: `http://127.0.0.1:5000/method5`

Method: GET

Params: Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

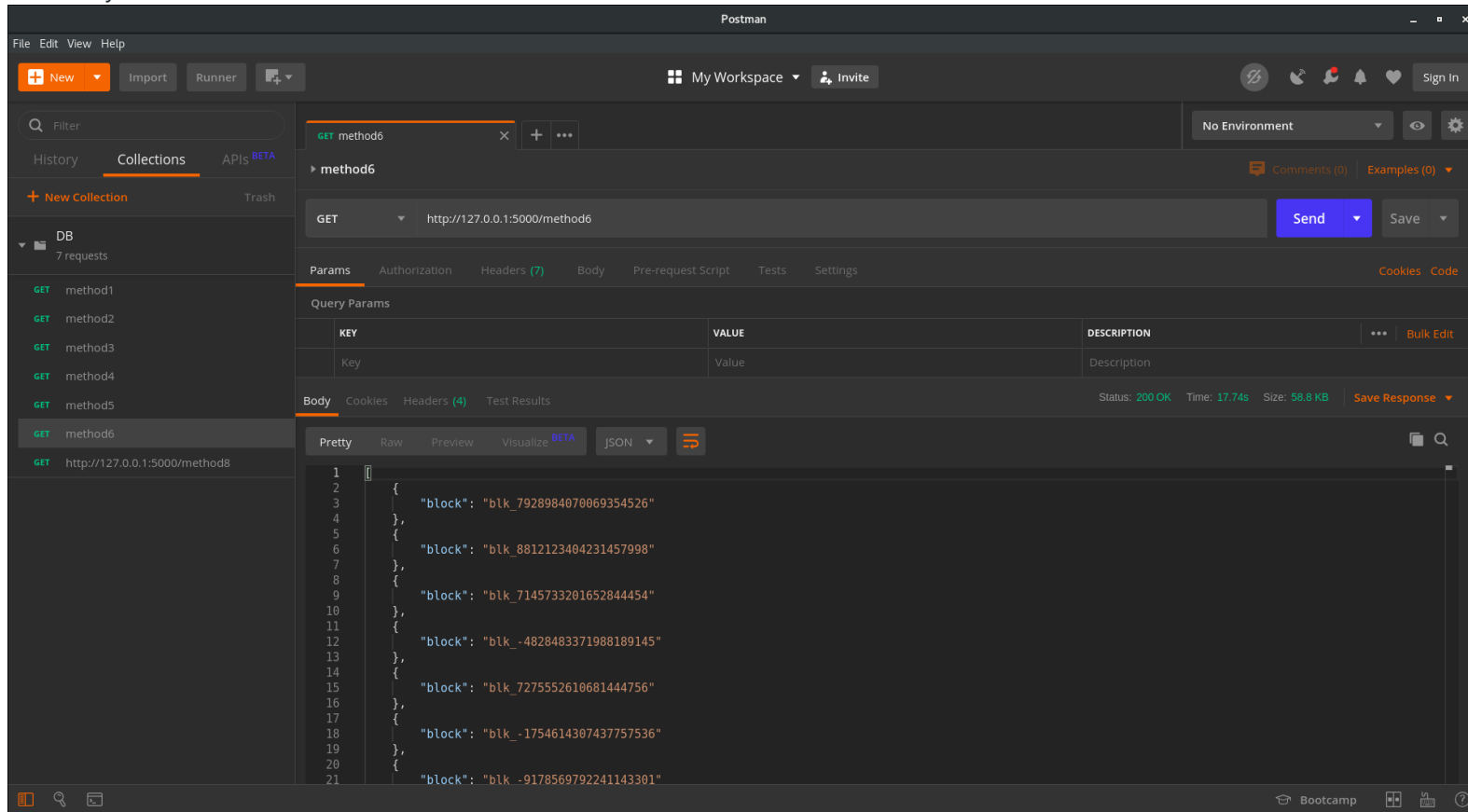
Body: Cookies Headers (4) Test Results

Status: 200 OK Time: 23.61s Size: 67.08 KB Save Response

Response (JSON):

```
1 [
2   {
3     "referer": "http://www.almhuetten-raith.at/apache-log/?page=access.log",
4     "size": 6
5   },
6   {
7     "referer": "http://localhost:8808/notebooks/work/spark/module5_cs2.ipynb",
8     "size": 17
9   },
10  {
11    "referer": "http://www.almhuetten-raith.at/images/stories/slideshow/",
12    "size": 4
13  },
14  {
15    "referer": "https://translate.googleusercontent.com/translate_c?depth=1&hl=en&prev=search&rurl=translate.google.com&sl=de&sp=nmt4&u=http://almhuetten-raith.at/6xid=17259,1570021,15700186,15700190,15700256,15700259,15700262,15700265&usq=ALKJrhIFpK5W5bt0TqoGc9_3ltSqad4r5Q",
16    "size": 8
17  },
18  {
19    "referer": "http://www.almhuetten-raith.at/apache-log/?M=D",
20    "size": 5
21  }
22 ]
```

• API Query 6



The screenshot shows the Postman application interface. The left sidebar displays a collection named 'DB' with 7 requests. The main panel shows a GET request to 'http://127.0.0.1:5000/method6'. The response is a JSON array of 6 objects, each containing a 'block' field. The status is 200 OK, and the response size is 58.8 KB.

GET method6

URL: `http://127.0.0.1:5000/method6`

Method: GET

Params: Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body: Cookies Headers (4) Test Results

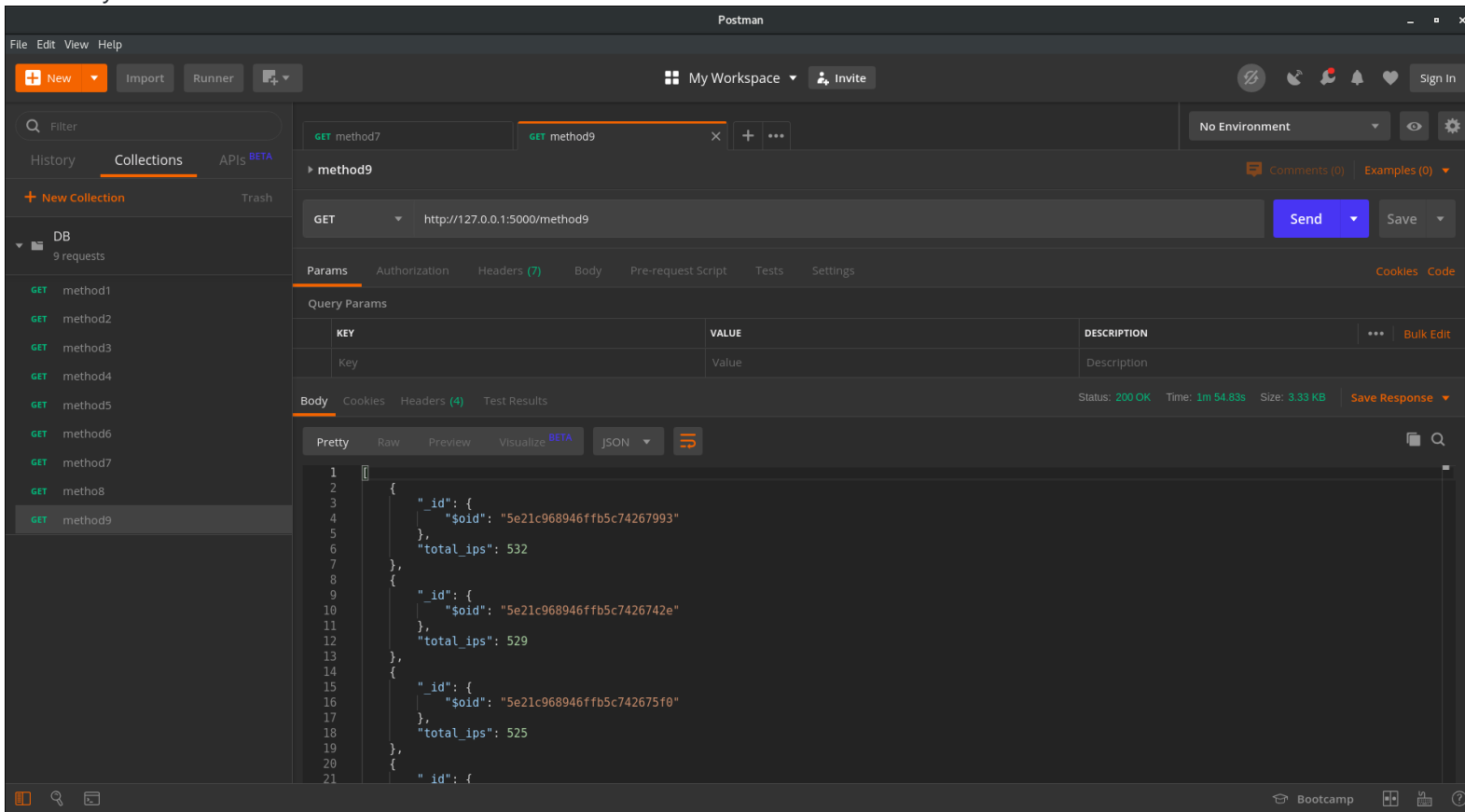
Status: 200 OK Time: 17.74s Size: 58.8 KB Save Response

Response (JSON):

```
1 [
2   {
3     "block": "blk_7928984070069354526"
4   },
5   {
6     "block": "blk_8812123404231457998"
7   },
8   {
9     "block": "blk_7145733201652844454"
10  },
11  {
12    "block": "blk_-4828483371988189145"
13  },
14  {
15    "block": "blk_7275552610681444756"
16  },
17  {
18    "block": "blk_-1754614307437757536"
19  },
20  {
21    "block": "blk_-9178569792241143301"
22  }
23 ]
```

- API Query 8

• API Query 9



The screenshot shows the Postman application interface. On the left sidebar, a collection named 'DB' is expanded, showing a list of requests from 'method1' to 'method9'. 'method9' is selected. The main panel displays the details for 'method9', which is a GET request to 'http://127.0.0.1:5000/method9'. The 'Body' tab is active, showing a JSON response in 'Pretty' format. The response is an array of three objects, each containing '_id', '\$oid', and 'total_ips'.

GET method9

http://127.0.0.1:5000/method9

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

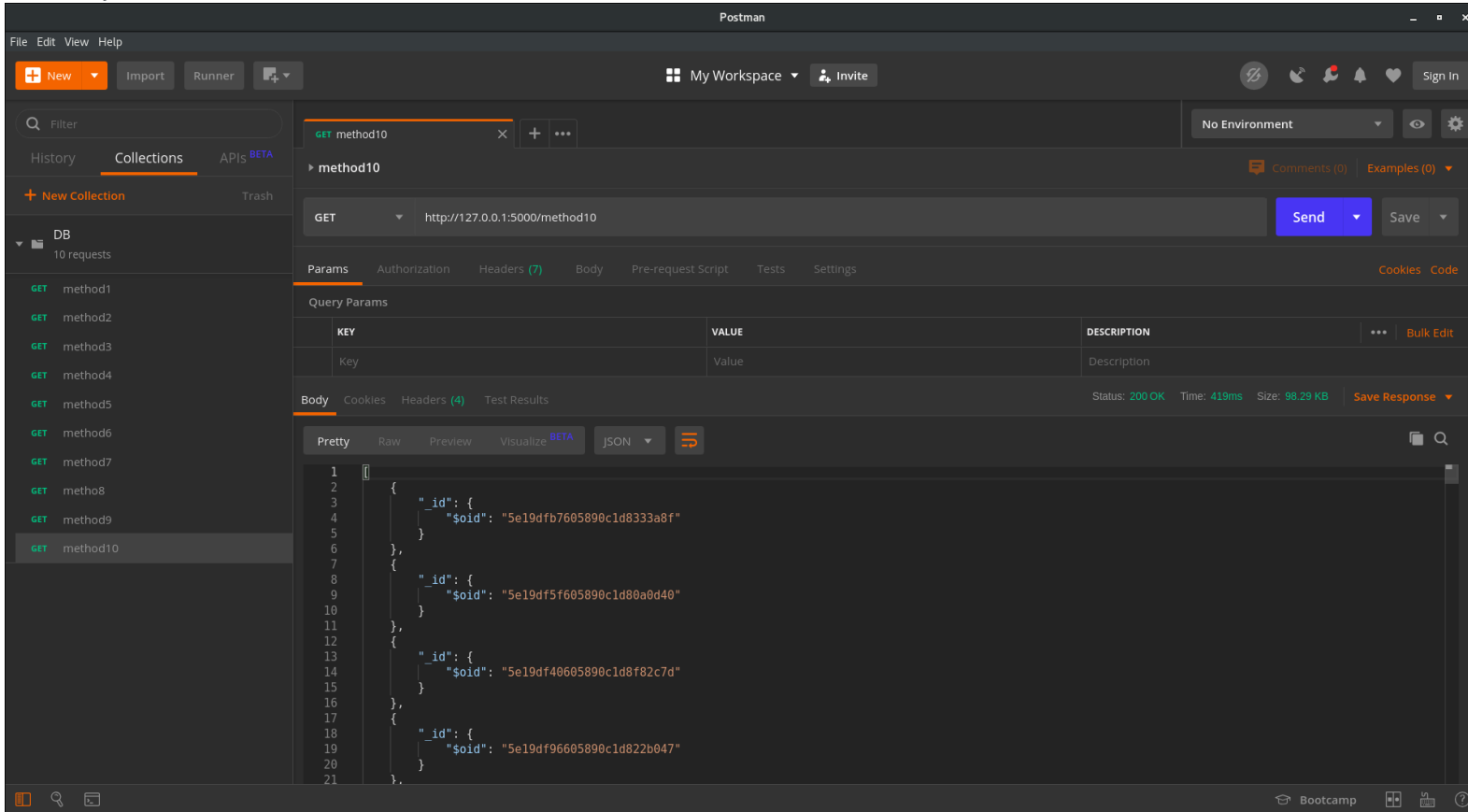
KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results Status: 200 OK Time: 1m 54.83s Size: 3.33 KB Save Response

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   {
3     "_id": {
4       "$oid": "5e21c968946ffb5c74267993"
5     },
6     "total_ips": 532
7   },
8   {
9     "_id": {
10      "$oid": "5e21c968946ffb5c7426742e"
11    },
12    "total_ips": 529
13  },
14  {
15    "_id": {
16      "$oid": "5e21c968946ffb5c742675f0"
17    },
18    "total_ips": 525
19  },
20  {
21    "_id": {
```

• API Query 10



The screenshot shows the Postman application interface. On the left sidebar, the 'DB' collection is expanded, showing requests from 'method1' to 'method10'. 'method10' is selected. The main panel displays the details for 'method10', which is a GET request to 'http://127.0.0.1:5000/method10'. The 'Body' tab is active, showing a JSON response in 'Pretty' format. The response is an array of four objects, each containing '_id' and '\$oid'.

GET method10

http://127.0.0.1:5000/method10

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results Status: 200 OK Time: 419ms Size: 98.29 KB Save Response

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   {
3     "_id": {
4       "$oid": "5e19dfb7605890c1d8333a8f"
5     }
6   },
7   {
8     "_id": {
9       "$oid": "5e19df5f605890c1d80a0d40"
10    }
11  },
12  {
13    "_id": {
14      "$oid": "5e19df40605890c1d8f82c7d"
15    }
16  },
17  {
18    "_id": {
19      "$oid": "5e19df96605890c1d822b847"
20    }
21  },
22  }
```

• API Query 11

GET

http://localhost:5000/method11?name=Jonathan_Lane

Send

Save

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

Code

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	name	Jonathan_Lane			
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 680ms

Size: 3.7 KB

Save Response

Pretty

Raw

Preview

Visualize BETA

JSON

1

2

3

4

5

6

7

8

9

10

11

12

[
 {
 "blocks": [
 "blk_-4671766229931495870"
],
 },
 {
 "blocks": [
 "blk_4284083803813138589"
],
 },
]

• Insert Log Examples

The screenshot shows the Postman application interface. On the left sidebar, under the 'Collections' tab, a collection named 'DB' is expanded, showing 12 requests. The request 'POST Insert access' is selected. The main panel displays the details of this request. The URL is 'http://127.0.0.1:5000/insert/access'. The 'Body' tab is active, showing a JSON payload:

```
1 - {
2   "source_ip": "1.1.1.1",
3   "log_timestamp": "2018-10-10 10:10:10",
4   "http_method": "GET",
5   "http_response": "200",
6   "resource": "index.php"
7 }
```

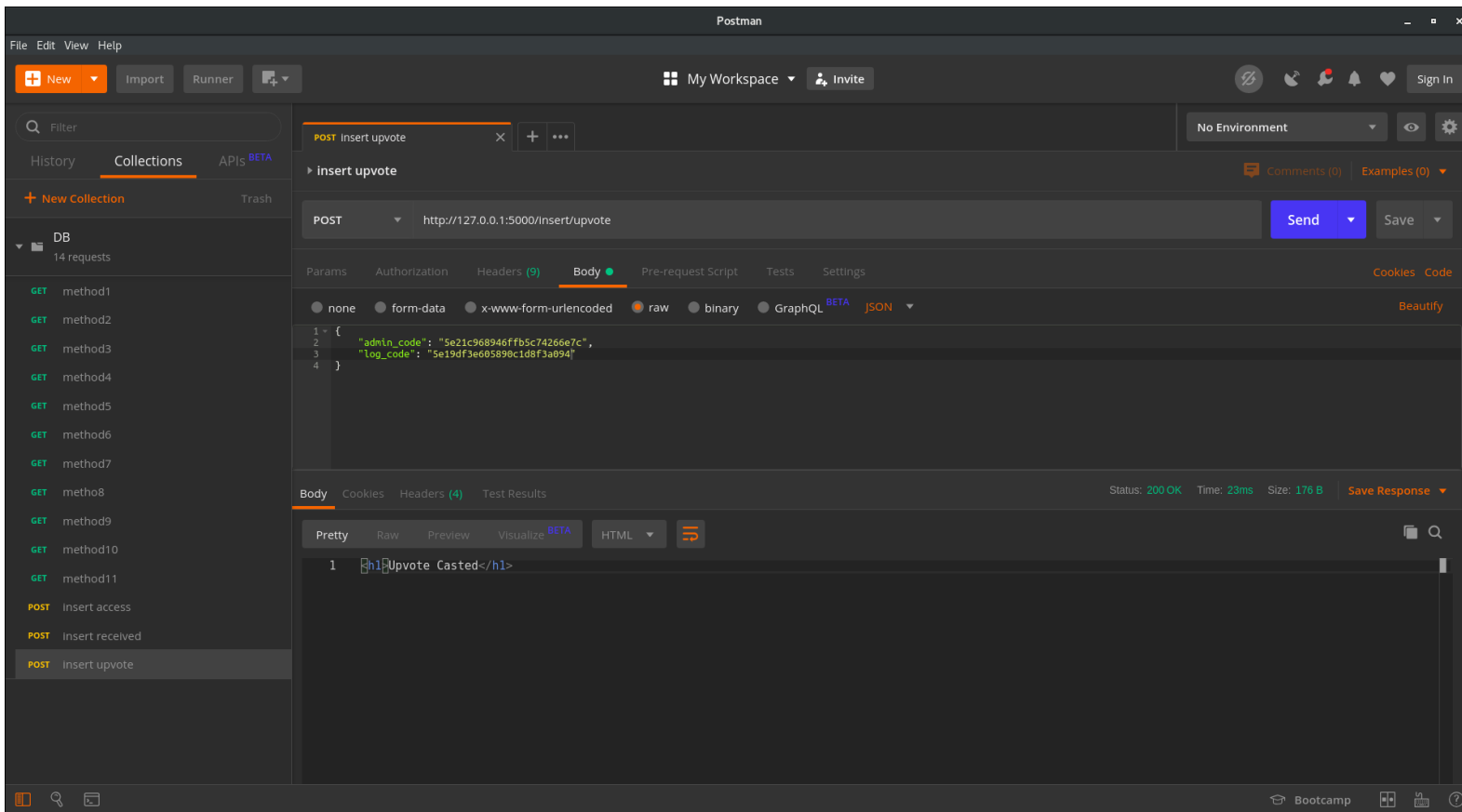
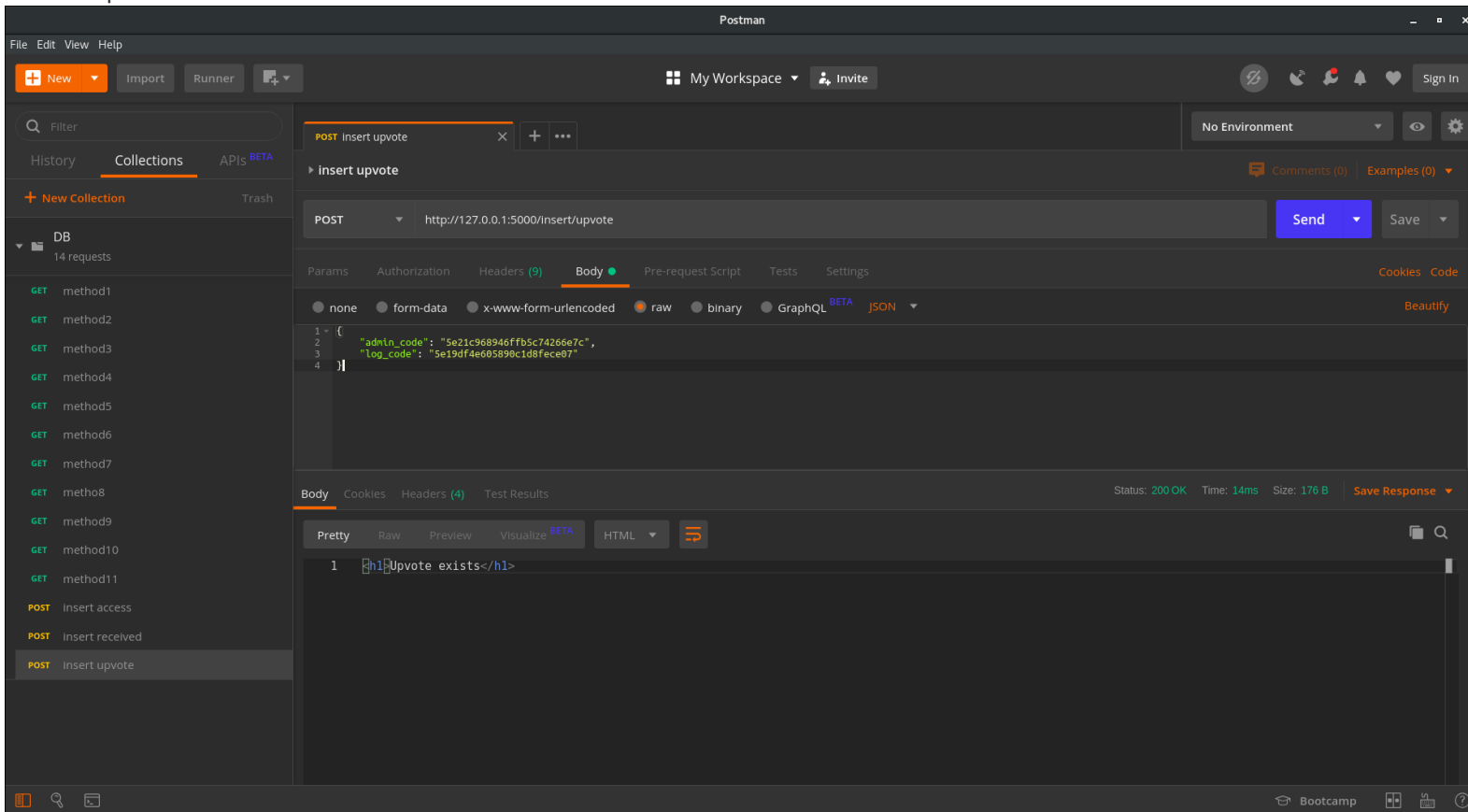
 The status bar at the bottom indicates a successful response: 'Status: 200 OK', 'Time: 9ms', 'Size: 183 B'. The response body is displayed in the 'Pretty' view as `<h1>Access Log Inserted!</h1>`.

The screenshot shows the Postman application interface. On the left sidebar, under the 'Collections' tab, a collection named 'DB' is expanded, showing 13 requests. The request 'POST Insert received' is selected. The main panel displays the details of this request. The URL is 'http://127.0.0.1:5000/insert/received'. The 'Body' tab is active, showing a JSON payload:

```
1 - {
2   "source_ip": "1.1.1.1",
3   "log_timestamp": "2018-10-10 10:10:10",
4   "blocks": ["block1", "block2"],
5   "destinations": ["dest1", "dest2"]
6 }
```

 The status bar at the bottom indicates a successful response: 'Status: 200 OK', 'Time: 10ms', 'Size: 184 B'. The response body is displayed in the 'Pretty' view as `<h1>Received Log Inserted</h1>`.

- Cast an upvote



Indices

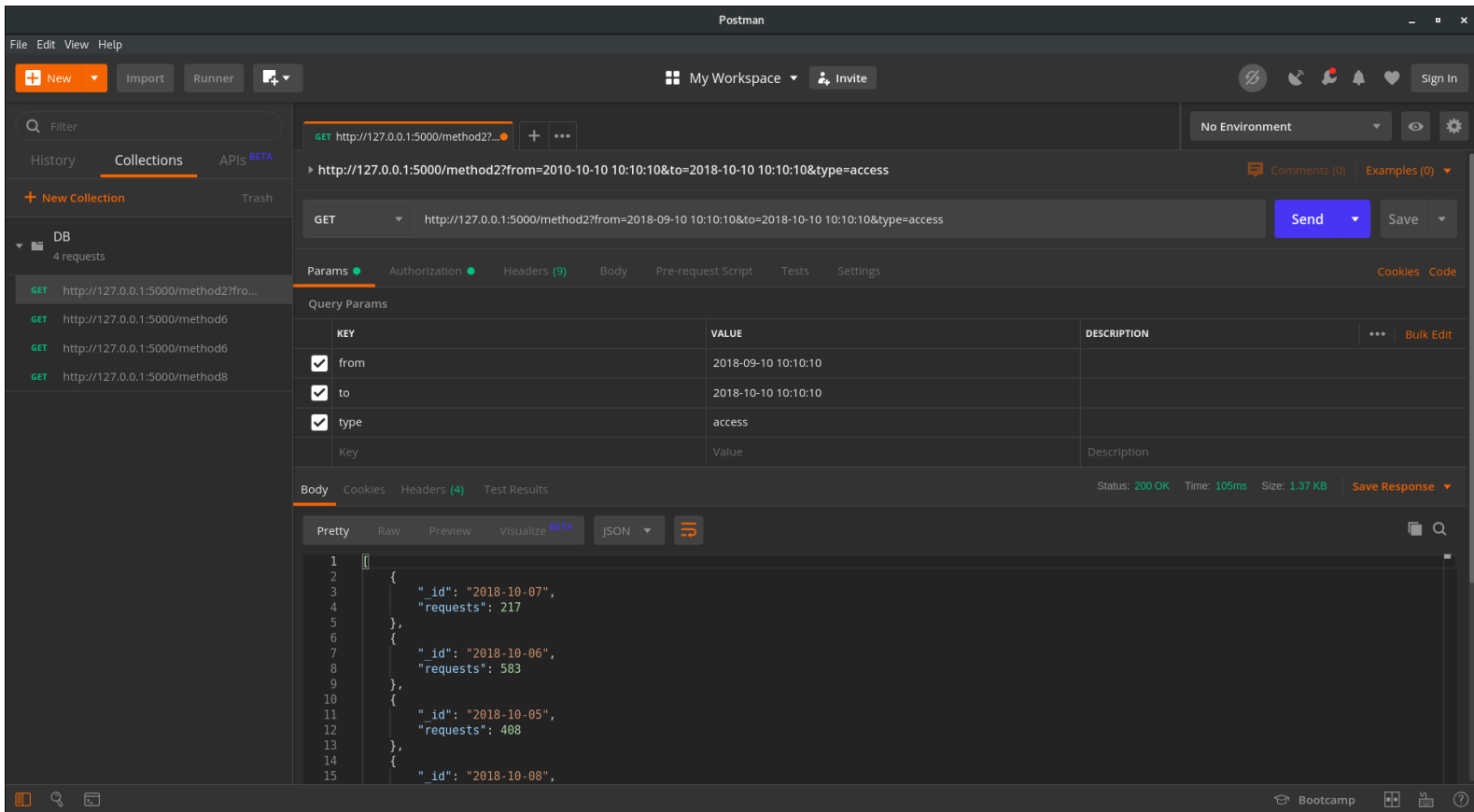
Log_timestamp Index

We found out that indices are situational since only two aggregation pipeline stages (sort, match) take indices into account. Group for example makes no use of it.

In addition to these, by running time range queries we observed that indices speed up the query when they search for a small time range. We can see that with the following snapshots based on method2:

1. For a small time range query using index makes it faster, to the point that is almost instant:

- Index



The screenshot shows the Postman REST client interface. The top bar includes the Postman logo, workspace name 'My Workspace', and a 'Sign In' button. The left sidebar shows a 'Collections' tab with a collection named 'DB' containing 4 requests. The main area displays a GET request to the endpoint `http://127.0.0.1:5000/method2?from=2018-09-10 10:10:10&to=2018-10-10 10:10:10&type=access`. The 'Params' tab is active, showing query parameters: 'from' (2018-09-10 10:10:10), 'to' (2018-10-10 10:10:10), and 'type' (access). The 'Body' tab is also active, showing a JSON response in 'Pretty' format. The response status is 200 OK, with a time of 105ms and a size of 1.37 KB. The JSON response is an array of documents, each with an '_id' and a 'requests' count.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2018-09-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

```
1  [
2    {
3      "_id": "2018-10-07",
4      "requests": 217
5    },
6    {
7      "_id": "2018-10-06",
8      "requests": 583
9    },
10   {
11     "_id": "2018-10-05",
12     "requests": 408
13   },
14   {
15     "_id": "2018-10-08",
```

- No Index

Postman interface showing a GET request to `http://127.0.0.1:5000/method2?from=2018-09-10 10:10:10&to=2018-10-10 10:10:10&type=access`. The response is a JSON array of 3 objects, each with an `_id` and `requests` count.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2018-09-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

```
1 {
2   {
3     "_id": "2018-10-07",
4     "requests": 217
5   },
6   {
7     "_id": "2018-10-06",
8     "requests": 583
9   },
10  {
11    "_id": "2018-10-05",
12    "requests": 408
13  },
14  {
15    "_id": "2018-10-08",
```

2. For a relatively large time range query we observe that the index gives only a slight boost in most cases:

- Index

Postman interface showing a GET request to `http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access`. The response is a JSON array of 4 objects, each with an `_id` and `requests` count.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2010-10-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

```
1 {
2   {
3     "_id": "2018-10-10",
4     "requests": 73
5   },
6   {
7     "_id": "2018-10-08",
8     "requests": 458
9   },
10  {
11    "_id": "2018-10-07",
12    "requests": 217
13  },
14  {
15    "_id": "2018-10-06",
```

- No Index

Postman

File Edit View Help

New Import Runner

My Workspace Invite

No Environment

Filter

History Collections APIs BETA

+ New Collection Trash

DB
4 requests

GET http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access

GET http://127.0.0.1:5000/method6

GET http://127.0.0.1:5000/method6

GET http://127.0.0.1:5000/method8

GET http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access

Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2010-10-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

Body Cookies Headers (4) Test Results Status: 200 OK Time: 9.63s Size: 40.71 KB Save Response

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   {
3     "id": "2018-10-10",
4     "requests": 73
5   },
6   {
7     "id": "2018-10-08",
8     "requests": 458
9   },
10  {
11    "id": "2018-10-07",
12    "requests": 217
13  },
14  {
15    "id": "2018-10-06",
```

3. Average case:

- Index

Postman

File Edit View Help

New Import Runner

My Workspace Invite

No Environment

Filter

History Collections APIs BETA

+ New Collection Trash

DB
4 requests

GET http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access

GET http://127.0.0.1:5000/method6

GET http://127.0.0.1:5000/method6

GET http://127.0.0.1:5000/method8

GET http://127.0.0.1:5000/method2?from=2017-10-10 10:10:10&to=2018-10-10 10:10:10&type=access

Send Save

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies Code

Query Params

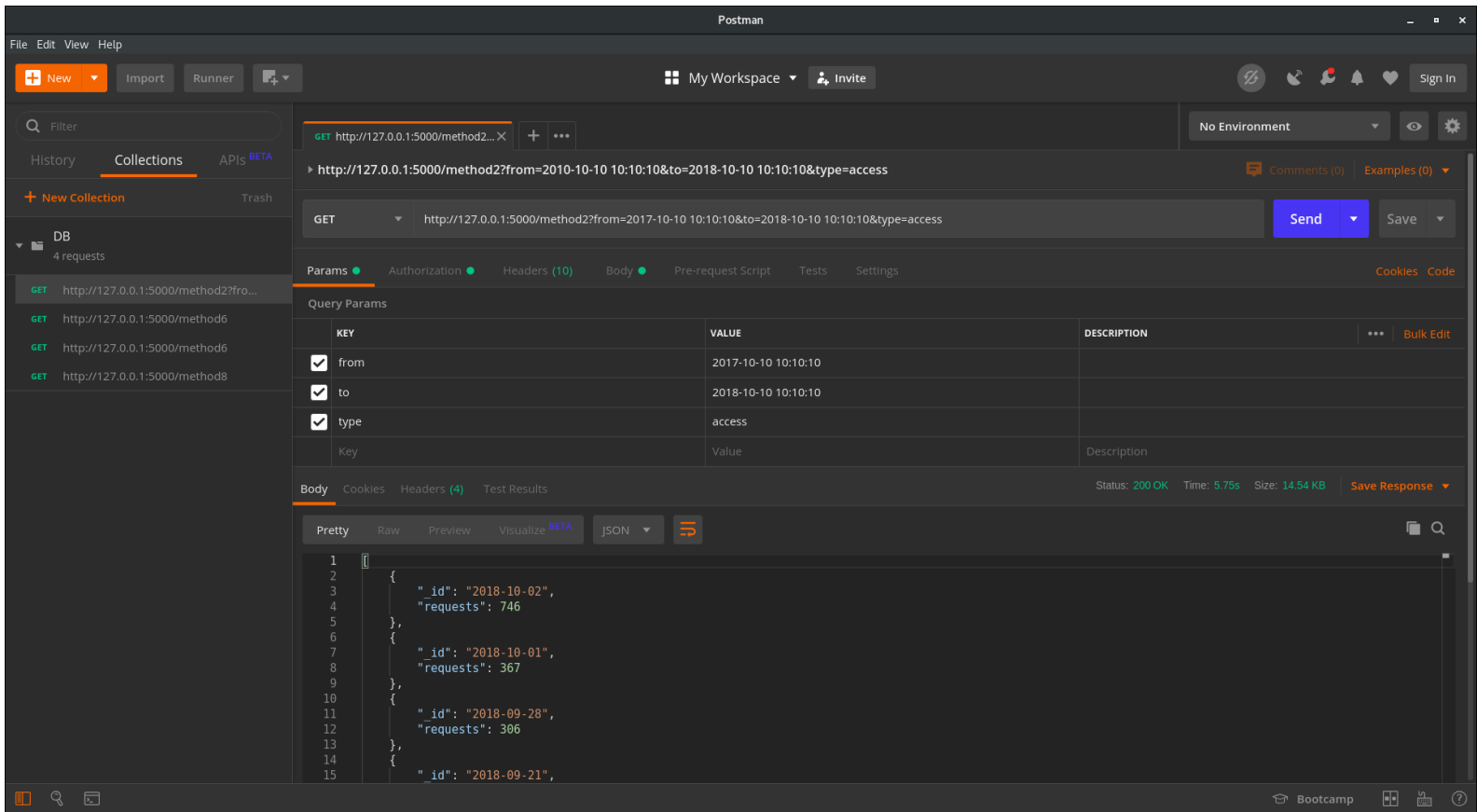
KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2017-10-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

Body Cookies Headers (4) Test Results Status: 200 OK Time: 4.72s Size: 14.54 KB Save Response

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   {
3     "id": "2018-10-02",
4     "requests": 746
5   },
6   {
7     "id": "2018-10-01",
8     "requests": 367
9   },
10  {
11    "id": "2018-09-28",
12    "requests": 306
13  },
14  {
15    "id": "2018-09-21",
```

- No Index



Despite the not so observable optimization, having a timestamp index is important since most of the queries involve a specific date or time range.

Type Index

As we see from the snapshot following adding an index on type field gives a slight boost to the query:

• Index

Postman interface showing a GET request to a method2 endpoint. The request URL is `http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access`. The response status is 200 OK, Time: 8.47s, Size: 40.71 KB. The response body is a JSON array of objects.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2010-10-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

```
1 {
2   {
3     "id": "2018-10-10",
4     "requests": 73
5   },
6   {
7     "id": "2018-10-08",
8     "requests": 458
9   },
10  {
11    "id": "2018-10-07",
12    "requests": 217
13  },
14  {
15    "id": "2018-10-06",
```

• No Index

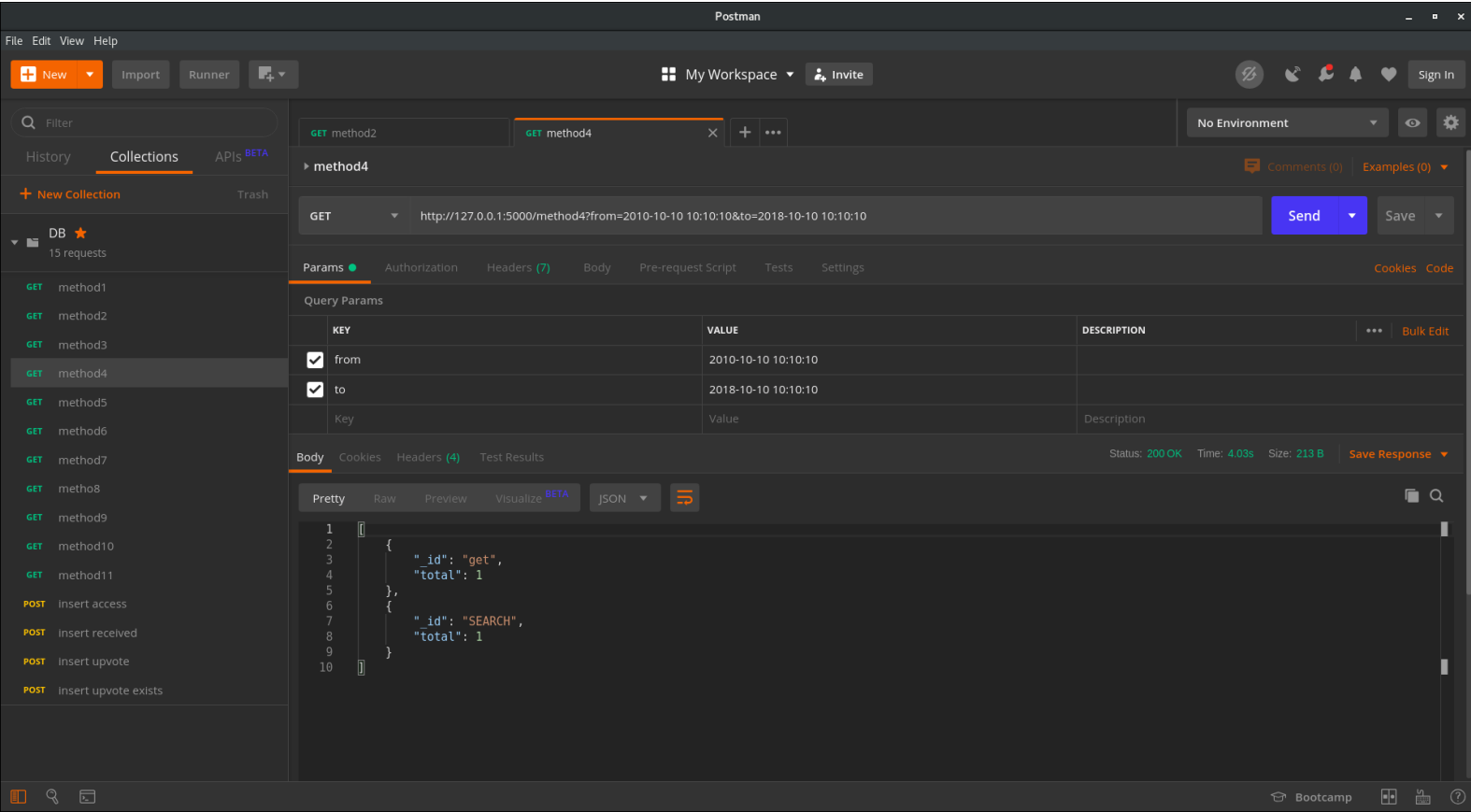
Postman interface showing a GET request to a method2 endpoint. The request URL is `http://127.0.0.1:5000/method2?from=2010-10-10 10:10:10&to=2018-10-10 10:10:10&type=access`. The response status is 200 OK, Time: 9.63s, Size: 40.71 KB. The response body is a JSON array of objects.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> from	2010-10-10 10:10:10	
<input checked="" type="checkbox"/> to	2018-10-10 10:10:10	
<input checked="" type="checkbox"/> type	access	
Key	Value	Description

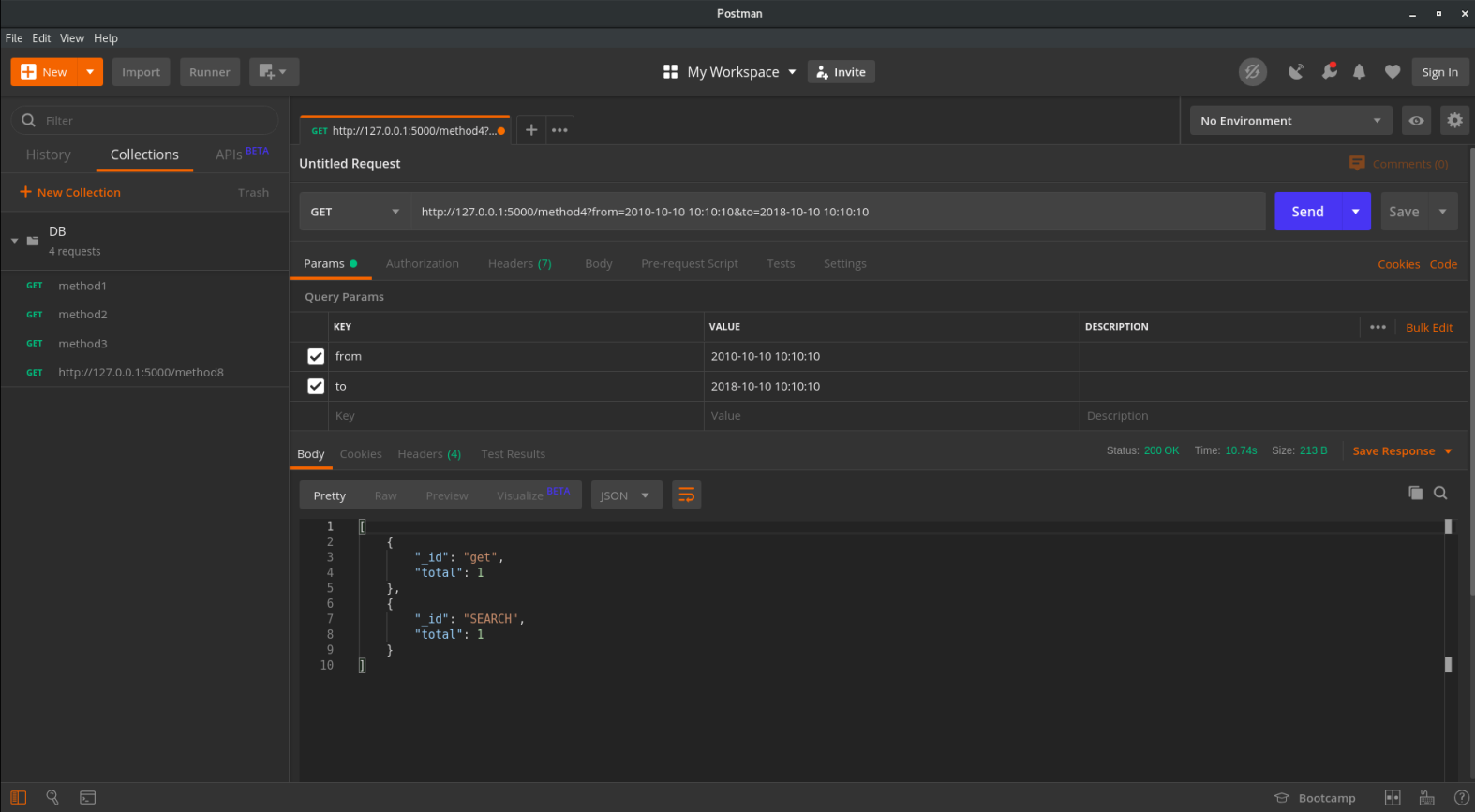
```
1 {
2   {
3     "id": "2018-10-10",
4     "requests": 73
5   },
6   {
7     "id": "2018-10-08",
8     "requests": 458
9   },
10  {
11    "id": "2018-10-07",
12    "requests": 217
13  },
14  {
15    "id": "2018-10-06",
```

HTTP Method Index

- Index



- No Index



Upvote Index

We also created an index for the upvote field of the admin collection which greatly increased query 7 execution speed

Index

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	day	2018-10-12			
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 22.07s

Size: 3.03 KB

Save Response

Pretty

Raw

Preview

Visualize BETA

HTML

1

[{"_id": {"\$oid": "5e122972abea1a1d560d3b36"}, "total": 3}, {"_id": {"\$oid": "5e122972abea1a1d560d39de"}, "total": 2},

2

{"_id": {"\$oid": "5e122972abea1a1d560d3ae1"}, "total": 2}, {"_id": {"\$oid": "5e122972abea1a1d560d3a36"}, "total": 2},

3

{"_id": {"\$oid": "5e122972abea1a1d560d39af"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39df"}, "total": 1},

4

{"_id": {"\$oid": "5e122972abea1a1d560d398c"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3932"}, "total": 1},

5

{"_id": {"\$oid": "5e122972abea1a1d560d3916"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3908"}, "total": 1},

6

{"_id": {"\$oid": "5e122972abea1a1d560d38fd"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39f6"}, "total": 1},

7

{"_id": {"\$oid": "5e122972abea1a1d560d39e6"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39b9"}, "total": 1},

8

{"_id": {"\$oid": "5e122972abea1a1d560d39d5"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3889"}, "total": 1},

9

{"_id": {"\$oid": "5e122972abea1a1d560d3934"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3909"}, "total": 1},

10

{"_id": {"\$oid": "5e122972abea1a1d560d3940"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d38e4"}, "total": 1},

11

{"_id": {"\$oid": "5e122972abea1a1d560d38d3"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d38be"}, "total": 1},

12

{"_id": {"\$oid": "5e122972abea1a1d560d387a"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d38ef"}, "total": 1},

13

{"_id": {"\$oid": "5e122972abea1a1d560d38b4"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d386d"}, "total": 1},

14

{"_id": {"\$oid": "5e122972abea1a1d560d3a03"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3a09"}, "total": 1},

15

{"_id": {"\$oid": "5e122972abea1a1d560d3a0e"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d389a"}, "total": 1},

16

{"_id": {"\$oid": "5e122972abea1a1d560d39c2"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39c4"}, "total": 1},

17

{"_id": {"\$oid": "5e122972abea1a1d560d399e"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3a15"}, "total": 1},

18

{"_id": {"\$oid": "5e122972abea1a1d560d3939"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d393d"}, "total": 1},

19

...

No Index

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	day	2018-10-12			
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 1m 6.53s

Size: 3.03 KB

Save Response

Pretty

Raw

Preview

Visualize BETA

HTML

1

[{"_id": {"\$oid": "5e122972abea1a1d560d3b36"}, "total": 3}, {"_id": {"\$oid": "5e122972abea1a1d560d39de"}, "total": 2},

2

{"_id": {"\$oid": "5e122972abea1a1d560d3ae1"}, "total": 2}, {"_id": {"\$oid": "5e122972abea1a1d560d3a36"}, "total": 2},

3

{"_id": {"\$oid": "5e122972abea1a1d560d39af"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39df"}, "total": 1},

4

{"_id": {"\$oid": "5e122972abea1a1d560d398c"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3932"}, "total": 1},

5

{"_id": {"\$oid": "5e122972abea1a1d560d3916"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3908"}, "total": 1},

6

{"_id": {"\$oid": "5e122972abea1a1d560d38fd"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39f6"}, "total": 1},

7

{"_id": {"\$oid": "5e122972abea1a1d560d39e6"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39b9"}, "total": 1},

8

{"_id": {"\$oid": "5e122972abea1a1d560d39d5"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3889"}, "total": 1},

9

{"_id": {"\$oid": "5e122972abea1a1d560d3934"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3909"}, "total": 1},

10

{"_id": {"\$oid": "5e122972abea1a1d560d3940"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d38e4"}, "total": 1},

11

{"_id": {"\$oid": "5e122972abea1a1d560d38d3"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d38be"}, "total": 1},

12

{"_id": {"\$oid": "5e122972abea1a1d560d387a"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d38ef"}, "total": 1},

13

{"_id": {"\$oid": "5e122972abea1a1d560d38b4"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d386d"}, "total": 1},

14

{"_id": {"\$oid": "5e122972abea1a1d560d3a03"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3a09"}, "total": 1},

15

{"_id": {"\$oid": "5e122972abea1a1d560d3a0e"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d389a"}, "total": 1},

16

{"_id": {"\$oid": "5e122972abea1a1d560d39c2"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d39c4"}, "total": 1},

17

{"_id": {"\$oid": "5e122972abea1a1d560d399e"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d3a15"}, "total": 1},

18

{"_id": {"\$oid": "5e122972abea1a1d560d3939"}, "total": 1}, {"_id": {"\$oid": "5e122972abea1a1d560d393d"}, "total": 1},

19

...

Indices were tested on other fields as well but no noticeable optimizations was indicated. We also created a compound index on log_timestamp and type but it actually made the query slower.

Github Link