Отчет по домашнему заданию №2 курса Highload Architect

Содержание отчета

[Mysql 2](#__RefHeading___Toc2251_678379071)

[DDL Таблицы users: 2](#__RefHeading___Toc2253_678379071)

[До добавления индекса 3](#__RefHeading___Toc2255_678379071)

[Explane запроса: 3](#__RefHeading___Toc2257_678379071)

[После добавления индекса 3](#__RefHeading___Toc2259_678379071)

[Добавление индекса: 3](#__RefHeading___Toc2261_678379071)

[Explane запроса: 3](#__RefHeading___Toc2263_678379071)

[Таблицы и графики 3](#__RefHeading___Toc2265_678379071)

[Выводы 4](#__RefHeading___Toc2267_678379071)

[Вопросы: 4](#__RefHeading___Toc2269_678379071)

[Приложение 1 – лог запросов до добавления индекса 4](#__RefHeading___Toc2271_678379071)

[Приложение 2 – лог запросов после добавления индекса 8](#__RefHeading___Toc2273_678379071)

[Postgres 12](#__RefHeading___Toc2275_678379071)

[DDL Таблицы users: 12](#__RefHeading___Toc2277_678379071)

[До добавления индекса 12](#__RefHeading___Toc2281_678379071)

[После добавления индекса 13](#__RefHeading___Toc2283_678379071)

[Добавление индекса: 13](#__RefHeading___Toc2285_678379071)

[Explane запроса: 13](#__RefHeading___Toc2287_678379071)

[Таблицы и графики 13](#__RefHeading___Toc2289_678379071)

[Выводы 14](#__RefHeading___Toc2291_678379071)

[Вопросы: 14](#__RefHeading___Toc2293_678379071)

[Приложение 1 – лог запросов до добавления индекса 14](#__RefHeading___Toc2295_678379071)

[Приложение 2 – лог запросов после добавления индекса 18](#__RefHeading___Toc2297_678379071)

# Mysql

## DDL Таблицы users:

SHOW CREATE TABLE users;

-------------------------------------------------------------------------------------------------------------------------+

| Table | Create Table |

-------------------------------------------------------------------------------------------------------------------------+

| users | CREATE TABLE `users` (

`user\_id` char(36) NOT NULL,

`first\_name` varchar(64) NOT NULL,

`second\_name` varchar(64) DEFAULT NULL,

`sex` enum('male','female') DEFAULT NULL,

`biography` text,

`city` varchar(64) DEFAULT NULL,

`birthdate` date DEFAULT NULL,

KEY `users\_user\_id\_index` (`user\_id`),

KEY `users\_first\_name\_second\_name\_index` (`first\_name`,`second\_name`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci |

+-------1 row in set (0.00 sec)

mysql> SHOW INDEX FROM users;

+-------+------------+------------------------------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+

| Table | Non\_unique | Key\_name | Seq\_in\_index | Column\_name | Collation | Cardinality | Sub\_part | Packed | Null | Index\_type | Comment | Index\_comment | Visible | Expression |

+-------+------------+------------------------------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+

| users | 1 | users\_user\_id\_index | 1 | user\_id | A | 922210 | NULL | NULL | | BTREE | | | YES | NULL |

| users | 1 | users\_first\_name\_second\_name\_index | 1 | first\_name | A | 139 | NULL | NULL | | BTREE | | | YES | NULL |

| users | 1 | users\_first\_name\_second\_name\_index | 2 | second\_name | A | 57115 | NULL | NULL | YES | BTREE | | | YES | NULL |

+-------+------------+------------------------------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+

3 rows in set (0.01 sec)

## До добавления индекса

### Explane запроса:

mysql> EXPLAIN select \* from users where first\_name like 'Иван%' and second\_name like 'Бес%';

+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

| EXPLAIN |

+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

| -> Filter: ((users.first\_name like 'Иван%') and (users.second\_name like 'Бес%')) (cost=107914 rows=12264)

-> Table scan on users (cost=107914 rows=993595)

|

+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

1 row in set (0.00 sec)

## После добавления индекса

### Добавление индекса:

mysql> create index users\_first\_name\_second\_name\_index

-> on users (first\_name, second\_name);

Query OK, 0 rows affected (3.05 sec)

Records: 0 Duplicates: 0 Warnings: 0

### Explane запроса:

mysql> EXPLAIN select \* from users where first\_name like 'Иван%' and second\_name like 'Бес%';

+----+-------------+-------+------------+-------+------------------------------------+------------------------------------+---------+------+-------+----------+----------------------------------+

| id | select\_type | table | partitions | type | possible\_keys | key | key\_len | ref | rows | filtered | Extra |

+----+-------------+-------+------------+-------+------------------------------------+------------------------------------+---------+------+-------+----------+----------------------------------+

| 1 | SIMPLE | users | NULL | range | users\_first\_name\_second\_name\_index | users\_first\_name\_second\_name\_index | 517 | NULL | 37052 | 11.11 | Using index condition; Using MRR |

+----+-------------+-------+------------+-------+------------------------------------+------------------------------------+---------+------+-------+----------+----------------------------------+

## Таблицы и графики

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Mysql | | | | | |
| Without index | | | With index | | |
| concurrently requests | rate (throughput) | latency(avg), sec | concurrently requests | rate | latency(avg), sec |
| 1 | 1 | 1,5 | 1 | 187 | 0,005 |
| 10 | 3 | 3 | 10 | 733 | 0,012 |
| 100 | 3 | 25 | 100 | 887 | 0,08 |
| 1000 | 3 | 220 | 1000 | 879 | 0,76 |

|  |  |
| --- | --- |
|  |  |

## Выводы

1. Поведение после добавления индекса ожидаемо, explane показывает использование индекса
2. Без использования индекса throughput составляет всего 3 запроса в секунду. После добавления индекса сервер без деградации latency держит примерно до 800 запросов в секунду. После наблюдается увеличение времени отклика.

## Вопросы:

1. Как можно увеличить throughput сервера. Вероятно, какими-то настройками, но какими?

## Приложение 1 – лог запросов до добавления индекса

hey -n 10 -c 1 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 15.1099 secs

Slowest: 1.5226 secs

Fastest: 1.4996 secs

Average: 1.5110 secs

Requests/sec: 0.6618

Response time histogram:

1.500 [1] |■■■■■■■■■■■■■■■■■■■■

1.502 [1] |■■■■■■■■■■■■■■■■■■■■

1.504 [1] |■■■■■■■■■■■■■■■■■■■■

1.506 [0] |

1.509 [0] |

1.511 [2] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

1.513 [0] |

1.516 [2] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

1.518 [1] |■■■■■■■■■■■■■■■■■■■■

1.520 [1] |■■■■■■■■■■■■■■■■■■■■

1.523 [1] |■■■■■■■■■■■■■■■■■■■■

Latency distribution:

10% in 1.4997 secs

25% in 1.5100 secs

50% in 1.5138 secs

75% in 1.5186 secs

90% in 1.5226 secs

0% in 0.0000 secs

0% in 0.0000 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0000 secs, 1.4996 secs, 1.5226 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0001 secs

req write: 0.0000 secs, 0.0000 secs, 0.0001 secs

resp wait: 1.5108 secs, 1.4995 secs, 1.5225 secs

resp read: 0.0001 secs, 0.0001 secs, 0.0001 secs

Status code distribution:

[200] 10 responses

hey -n 100 -c 10 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес' >>mysql.log

Summary:

Total: 32.6865 secs

Slowest: 6.5508 secs

Fastest: 1.6266 secs

Average: 3.0561 secs

Requests/sec: 3.0594

Response time histogram:

1.627 [1] |■

2.119 [34] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■

2.611 [0] |

3.104 [0] |

3.596 [48] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

4.089 [0] |

4.581 [0] |

5.074 [12] |■■■■■■■■■■

5.566 [0] |

6.058 [0] |

6.551 [5] |■■■■

Latency distribution:

10% in 1.6340 secs

25% in 1.6385 secs

50% in 3.2664 secs

75% in 3.2776 secs

90% in 4.9008 secs

95% in 6.5221 secs

99% in 6.5508 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0001 secs, 1.6266 secs, 6.5508 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0004 secs

req write: 0.0000 secs, 0.0000 secs, 0.0003 secs

resp wait: 3.0560 secs, 1.6265 secs, 6.5501 secs

resp read: 0.0000 secs, 0.0000 secs, 0.0001 secs

Status code distribution:

[200] 100 responses

hey -n 1000 -c 100 -t 0 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес' >>mysql.log

Summary:

Total: 337.3888 secs

Slowest: 297.5786 secs

Fastest: 1.5543 secs

Average: 25.2815 secs

Requests/sec: 2.9639

Response time histogram:

1.554 [1] |

31.157 [779] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

60.759 [121] |■■■■■■

90.362 [40] |■■

119.964 [20] |■

149.566 [11] |■

179.169 [11] |■

208.771 [7] |

238.374 [1] |

267.976 [6] |

297.579 [3] |

Latency distribution:

10% in 3.3414 secs

25% in 5.0195 secs

50% in 11.4449 secs

75% in 26.8221 secs

90% in 60.3494 secs

95% in 106.0476 secs

99% in 234.8641 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0006 secs, 1.5543 secs, 297.5786 secs

DNS-lookup: 0.0001 secs, 0.0000 secs, 0.0088 secs

req write: 0.0001 secs, 0.0000 secs, 0.0068 secs

resp wait: 25.2806 secs, 1.5542 secs, 297.5786 secs

resp read: 0.0000 secs, 0.0000 secs, 0.0004 secs

Status code distribution:

[200] 1000 responses

hey -n 3000 -c 1000 -t 0 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 987.2750 secs

Slowest: 982.3998 secs

Fastest: 1.5457 secs

Average: 220.2241 secs

Requests/sec: 3.0387

Response time histogram:

1.546 [1] |

99.631 [1234] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

197.717 [579] |■■■■■■■■■■■■■■■■■■■

295.802 [327] |■■■■■■■■■■■

393.887 [242] |■■■■■■■■

491.973 [203] |■■■■■■■

590.058 [143] |■■■■■

688.144 [101] |■■■

786.229 [83] |■■■

884.314 [51] |■■

982.400 [36] |■

Latency distribution:

10% in 10.0489 secs

25% in 43.0538 secs

50% in 141.9484 secs

75% in 339.4502 secs

90% in 564.8947 secs

95% in 705.6155 secs

99% in 906.5254 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0209 secs, 1.5457 secs, 982.3998 secs

DNS-lookup: 0.0066 secs, 0.0000 secs, 0.1053 secs

req write: 0.0068 secs, 0.0000 secs, 0.0609 secs

resp wait: 220.1956 secs, 1.5456 secs, 982.2938 secs

resp read: 0.0000 secs, 0.0000 secs, 0.0003 secs

Status code distribution:

[200] 3000 responses

## Приложение 2 – лог запросов после добавления индекса

hey -n 10 -c 1 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 0.0534 secs

Slowest: 0.0064 secs

Fastest: 0.0049 secs

Average: 0.0053 secs

Requests/sec: 187.4118

Response time histogram:

0.005 [1] |■■■■■■■■■■■■■

0.005 [3] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.005 [2] |■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.005 [0] |

0.005 [2] |■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.006 [0] |

0.006 [0] |

0.006 [0] |

0.006 [0] |

0.006 [0] |

0.006 [2] |■■■■■■■■■■■■■■■■■■■■■■■■■■■

Latency distribution:

10% in 0.0049 secs

25% in 0.0050 secs

50% in 0.0051 secs

75% in 0.0063 secs

90% in 0.0064 secs

0% in 0.0000 secs

0% in 0.0000 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0000 secs, 0.0049 secs, 0.0064 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0002 secs

req write: 0.0000 secs, 0.0000 secs, 0.0000 secs

resp wait: 0.0052 secs, 0.0048 secs, 0.0062 secs

resp read: 0.0000 secs, 0.0000 secs, 0.0001 secs

Status code distribution:

[200] 10 responses

hey -n 100 -c 10 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 0.1364 secs

Slowest: 0.0399 secs

Fastest: 0.0051 secs

Average: 0.0117 secs

Requests/sec: 733.3615

Response time histogram:

0.005 [1] |■

0.009 [26] |■■■■■■■■■■■■■■■■■■■■■■■■■

0.012 [42] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.016 [16] |■■■■■■■■■■■■■■■

0.019 [3] |■■■

0.022 [6] |■■■■■■

0.026 [3] |■■■

0.029 [1] |■

0.033 [1] |■

0.036 [0] |

0.040 [1] |■

Latency distribution:

10% in 0.0056 secs

25% in 0.0082 secs

50% in 0.0101 secs

75% in 0.0140 secs

90% in 0.0206 secs

95% in 0.0253 secs

99% in 0.0399 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0001 secs, 0.0051 secs, 0.0399 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0005 secs

req write: 0.0000 secs, 0.0000 secs, 0.0002 secs

resp wait: 0.0115 secs, 0.0050 secs, 0.0399 secs

resp read: 0.0000 secs, 0.0000 secs, 0.0001 secs

Status code distribution:

[200] 100 responses

hey -n 1000 -c 100 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 1.1261 secs

Slowest: 0.9258 secs

Fastest: 0.0050 secs

Average: 0.0839 secs

Requests/sec: 887.9856

Response time histogram:

0.005 [1] |

0.097 [775] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.189 [112] |■■■■■■

0.281 [42] |■■

0.373 [23] |■

0.465 [16] |■

0.557 [6] |

0.650 [6] |

0.742 [10] |■

0.834 [4] |

0.926 [5] |

Latency distribution:

10% in 0.0087 secs

25% in 0.0139 secs

50% in 0.0323 secs

75% in 0.0866 secs

90% in 0.2173 secs

95% in 0.3598 secs

99% in 0.7354 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0002 secs, 0.0050 secs, 0.9258 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0019 secs

req write: 0.0001 secs, 0.0000 secs, 0.0027 secs

resp wait: 0.0836 secs, 0.0049 secs, 0.9257 secs

resp read: 0.0000 secs, 0.0000 secs, 0.0003 secs

Status code distribution:

[200] 1000 responses

hey -n 3000 -c 1000 -t 0 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 3.4099 secs

Slowest: 3.3577 secs

Fastest: 0.0049 secs

Average: 0.7616 secs

Requests/sec: 879.7935

Response time histogram:

0.005 [1] |

0.340 [1186] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.675 [558] |■■■■■■■■■■■■■■■■■■■

1.011 [393] |■■■■■■■■■■■■■

1.346 [267] |■■■■■■■■■

1.681 [203] |■■■■■■■

2.017 [137] |■■■■■

2.352 [93] |■■■

2.687 [73] |■■

3.022 [58] |■■

3.358 [31] |■

Latency distribution:

10% in 0.0422 secs

25% in 0.1712 secs

50% in 0.5103 secs

75% in 1.1308 secs

90% in 1.9137 secs

95% in 2.4300 secs

99% in 3.0357 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0093 secs, 0.0049 secs, 3.3577 secs

DNS-lookup: 0.0062 secs, 0.0000 secs, 0.0862 secs

req write: 0.0014 secs, 0.0000 secs, 0.0529 secs

resp wait: 0.7445 secs, 0.0049 secs, 3.3091 secs

resp read: 0.0003 secs, 0.0000 secs, 0.0522 secs

Status code distribution:

[200] 3000 responses

# Postgres

## DDL Таблицы users:

\d users

Table "public.users"

Column | Type | Collation | Nullable | Default

-------------+-----------------------------+-----------+----------+---------

user\_id | character(36) | | not null |

first\_name | character varying(64) | | not null |

second\_name | character varying(64) | | |

sex | sex\_status | | |

biography | text | | |

city | character varying(64) | | |

birthdate | timestamp without time zone | | |

Indexes:

"users\_pkey" PRIMARY KEY, btree (user\_id)

"users\_first\_name\_second\_name\_index" btree (first\_name, second\_name)

Referenced by:

TABLE "token" CONSTRAINT "tokens\_user\_id\_fkey" FOREIGN KEY (user\_id) REFERENCES users(user\_id) ON DELETE CASCADE

TABLE "user\_credentials" CONSTRAINT "user\_credentials\_user\_id\_fkey" FOREIGN KEY (user\_id) REFERENCES users(user\_id) ON DELETE CASCADE

## До добавления индекса

snet=# EXPLAIN select \* from users where first\_name like 'Иван%' and second\_name like 'Бес%';

QUERY PLAN

---------------------------------------------------------------------------------------------------

Gather (cost=1000.00..22590.04 rows=133 width=128)

Workers Planned: 2

-> Parallel Seq Scan on users (cost=0.00..21576.74 rows=55 width=128)

Filter: (((first\_name)::text ~~ 'Иван%'::text) AND ((second\_name)::text ~~ 'Бес%'::text))

(4 rows)

## После добавления индекса

### Добавление индекса:

snet=# create index users\_first\_name\_second\_name\_index

on public.users (first\_name, second\_name);

CREATE INDEX

### Explane запроса:

\l snet

List of databases

Name | Owner | Encoding | Collate | Ctype | ICU Locale | Locale Provider | Access privileges

------+----------+----------+---------+-------+------------+-----------------+-------------------

snet | postgres | UTF8 | C | C | | libc |

(1 row)

snet=# EXPLAIN analyze select first\_name from users where first\_name like 'ИваН%' and second\_name like 'Бес%';

QUERY PLAN

---------------------------------------------------------------------------------------------------------------------------------------------------

Index Only Scan using users\_first\_name\_second\_name\_index on users (cost=0.42..411.94 rows=1 width=13) (actual time=0.077..0.188 rows=21 loops=1)

Index Cond: ((first\_name >= 'Иван'::text) AND (first\_name < 'Ивао'::text) AND (second\_name >= 'Бес'::text) AND (second\_name < 'Бет'::text))

Filter: ((first\_name ~~ 'Иван%'::text) AND (second\_name ~~ 'Бес%'::text))

Heap Fetches: 0

Planning Time: 0.129 ms

Execution Time: 0.230 ms

(6 rows)

## Таблицы и графики

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Postgres | | | | | |
| Without index | | | With index | | |
| concurrently requests | rate (throughput) | latency(avg), sec | concurrently requests | rate | latency(avg), sec |
| 1 | 22 | 0,04 | 1 | 1046 | 0,001 |
| 10 | 78 | 0,11 | 10 | 6632 | 0,014 |
| 100 | 93 | 0,78 | 100 | 10300 | 0,007 |
| 1000 | 93 | 7,2 | 1000 | 10300 | 0.07 |

|  |  |
| --- | --- |
|  |  |

## Выводы

1. Postgres использует созданный индекс, однако для того, чтобы он это сделал в необходимо для БД установить lc\_collate = C
2. До создания индекса запросы выполнялись на порядки быстрее чем с mysql(тоже без индекса)
3. Результаты намного превосходит результаты mysql(на порядок) в throughput 10000 vs 800 и latency
4. Деградация, выражающаяся в значительном увеличении latency в принципе не наблюдалась.
5. Оказывается для выражений like и операций сравнения необходимо устанавливать collation. lc\_collate = C как универсальный описан в блоге <https://simply.name/ru/pg-lc-collate.html>

## Приложение 1 – лог запросов до добавления индекса

hey -n 10 -c 1 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 0.4513 secs

Slowest: 0.1454 secs

Fastest: 0.0315 secs

Average: 0.0451 secs

Requests/sec: 22.1604

Response time histogram:

0.031 [1] |■■■■■

0.043 [8] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.054 [0] |

0.066 [0] |

0.077 [0] |

0.088 [0] |

0.100 [0] |

0.111 [0] |

0.123 [0] |

0.134 [0] |

0.145 [1] |■■■■■

Latency distribution:

10% in 0.0318 secs

25% in 0.0321 secs

50% in 0.0338 secs

75% in 0.0421 secs

90% in 0.1454 secs

0% in 0.0000 secs

0% in 0.0000 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0000 secs, 0.0315 secs, 0.1454 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0003 secs

req write: 0.0000 secs, 0.0000 secs, 0.0001 secs

resp wait: 0.0449 secs, 0.0313 secs, 0.1447 secs

resp read: 0.0001 secs, 0.0000 secs, 0.0001 secs

Status code distribution:

[200] 10 responses

hey -n 100 -c 10 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 1.2690 secs

Slowest: 0.3524 secs

Fastest: 0.0298 secs

Average: 0.1126 secs

Requests/sec: 78.8030

Response time histogram:

0.030 [1] |■

0.062 [19] |■■■■■■■■■■■■■■■■■■■■■■■■■

0.094 [31] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.127 [21] |■■■■■■■■■■■■■■■■■■■■■■■■■■■

0.159 [7] |■■■■■■■■■

0.191 [9] |■■■■■■■■■■■■

0.223 [6] |■■■■■■■■

0.256 [0] |

0.288 [3] |■■■■

0.320 [1] |■

0.352 [2] |■■■

Latency distribution:

10% in 0.0529 secs

25% in 0.0712 secs

50% in 0.0943 secs

75% in 0.1440 secs

90% in 0.2013 secs

95% in 0.2603 secs

99% in 0.3524 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0000 secs, 0.0298 secs, 0.3524 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0006 secs

req write: 0.0000 secs, 0.0000 secs, 0.0001 secs

resp wait: 0.1124 secs, 0.0297 secs, 0.3517 secs

resp read: 0.0001 secs, 0.0000 secs, 0.0001 secs

Status code distribution:

[200] 100 responses

hey -n 1000 -c 100 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 10.7142 secs

Slowest: 9.0241 secs

Fastest: 0.0318 secs

Average: 0.7837 secs

Requests/sec: 93.3339

Response time histogram:

0.032 [1] |

0.931 [768] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

1.830 [137] |■■■■■■■

2.729 [43] |■■

3.629 [14] |■

4.528 [6] |

5.427 [3] |

6.326 [7] |

7.226 [7] |

8.125 [10] |■

9.024 [4] |

Latency distribution:

10% in 0.0814 secs

25% in 0.1269 secs

50% in 0.3089 secs

75% in 0.8841 secs

90% in 1.7725 secs

95% in 2.8852 secs

99% in 7.5123 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0003 secs, 0.0318 secs, 9.0241 secs

DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0026 secs

req write: 0.0001 secs, 0.0000 secs, 0.0058 secs

resp wait: 0.7831 secs, 0.0317 secs, 9.0182 secs

resp read: 0.0001 secs, 0.0000 secs, 0.0002 secs

Status code distribution:

[200] 1000 responses

hey -n 3000 -c 1000 -t 0 -m GET 'http://localhost:8800/user/search?first\_name=Иван&second\_name=Бес'

Summary:

Total: 32.2619 secs

Slowest: 32.0809 secs

Fastest: 0.0443 secs

Average: 7.2458 secs

Requests/sec: 92.9890

Response time histogram:

0.044 [1] |

3.248 [1206] |■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■

6.452 [548] |■■■■■■■■■■■■■■■■■■

9.655 [377] |■■■■■■■■■■■■■

12.859 [267] |■■■■■■■■■

16.063 [188] |■■■■■■

19.266 [150] |■■■■■

22.470 [110] |■■■■

25.674 [78] |■■■

28.877 [48] |■■

32.081 [27] |■

Latency distribution:

10% in 0.4512 secs

25% in 1.5501 secs

50% in 4.6728 secs

75% in 11.0035 secs

90% in 18.3749 secs

95% in 22.5326 secs

99% in 28.8461 secs

Details (average, fastest, slowest):

DNS+dialup: 0.0314 secs, 0.0443 secs, 32.0809 secs

DNS-lookup: 0.0217 secs, 0.0000 secs, 0.1522 secs

req write: 0.0030 secs, 0.0000 secs, 0.1451 secs

resp wait: 7.2083 secs, 0.0437 secs, 31.9330 secs

resp read: 0.0001 secs, 0.0000 secs, 0.0083 secs

Status code distribution:

[200] 3000 responses

## Приложение 2 – лог запросов после добавления индекса

\l snet

List of databases

Name | Owner | Encoding | Collate | Ctype | ICU Locale | Locale Provider | Access privileges

------+----------+----------+---------+-------+------------+-----------------+-------------------

snet | postgres | UTF8 | C | C | | libc |

(1 row)

snet=# EXPLAIN analyze select first\_name from users where first\_name like 'ИваН%' and second\_name like 'Бес%';

QUERY PLAN

---------------------------------------------------------------------------------------------------------------------------------------------------

Index Only Scan using users\_first\_name\_second\_name\_index on users (cost=0.42..411.94 rows=1 width=13) (actual time=0.077..0.188 rows=21 loops=1)

Index Cond: ((first\_name >= 'Иван'::text) AND (first\_name < 'Ивао'::text) AND (second\_name >= 'Бес'::text) AND (second\_name < 'Бет'::text))

Filter: ((first\_name ~~ 'Иван%'::text) AND (second\_name ~~ 'Бес%'::text))

Heap Fetches: 0

Planning Time: 0.129 ms

Execution Time: 0.230 ms

(6 rows)