CSCI 5408 Data Management Warehousing and Analytics

Assignment 1

Search Query Implementation using Relational Database and Elastic Search

Date of Submission: May 24, 2018

Hemanth Kurra (B00784050)

Srisaichand Singamaneni (B00792835)

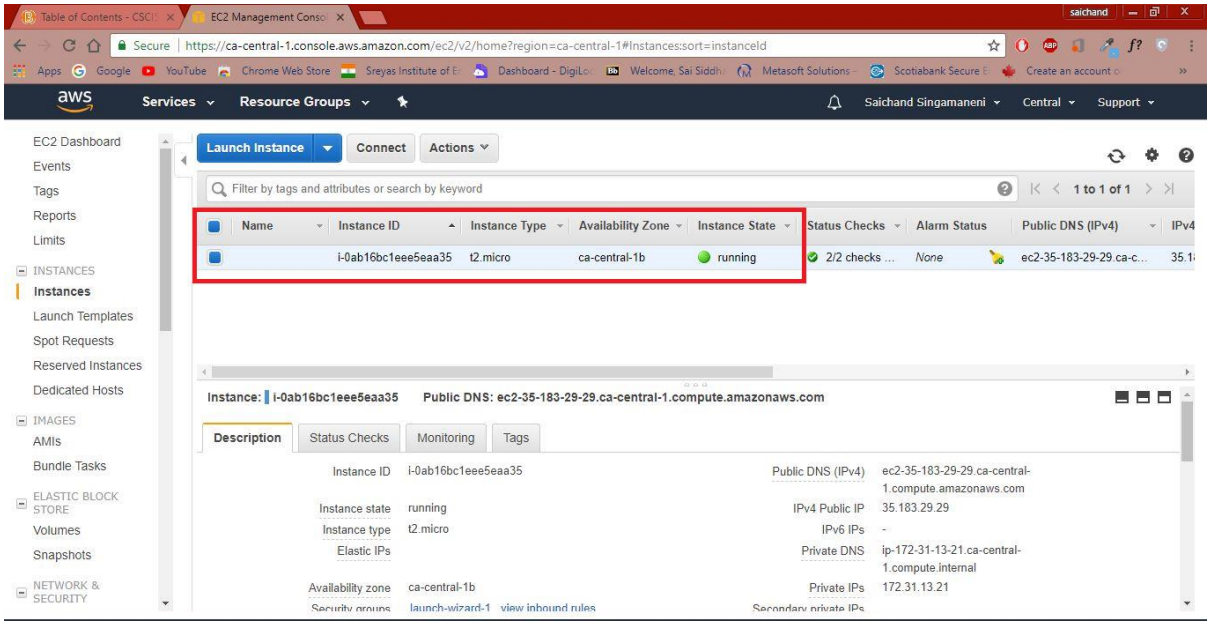
GitHub URL: https://github.com/singamanenisrisai/MySQL-and-Elastic-Search.git

1. TASK DESCRIPTION:

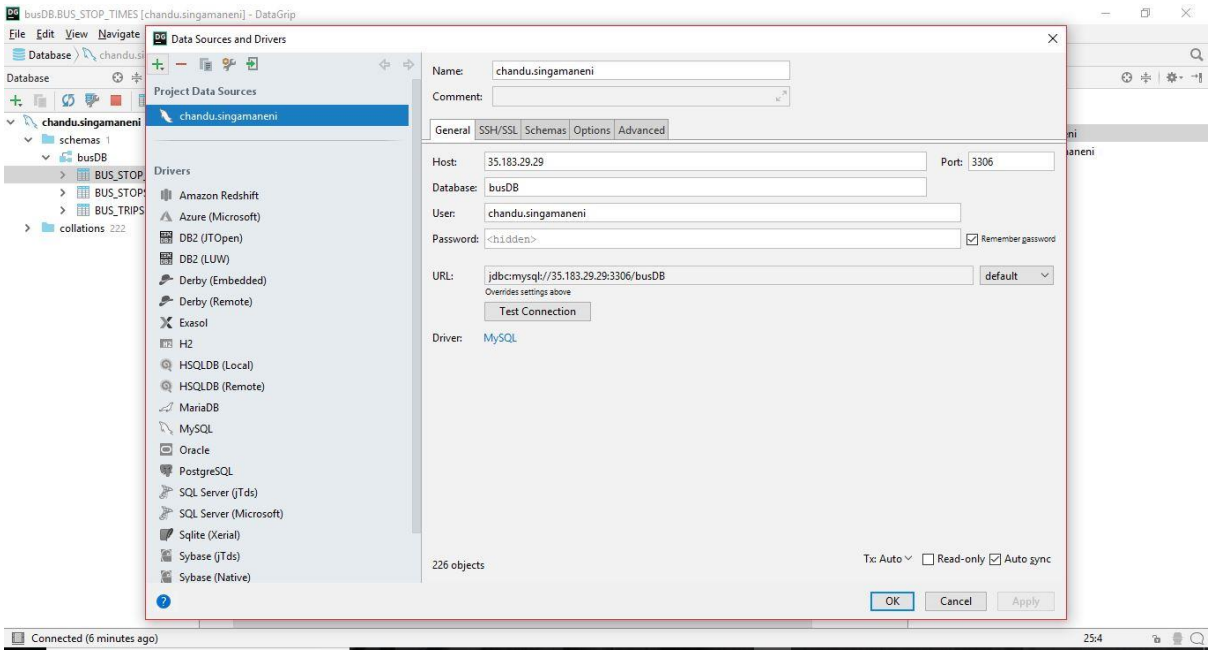
The purpose of the task is to analyse the most efficient data retrieval tool by comparing relational database system and cloud database. The Halifax transit data was used for the analysis.

Applications & Requirements

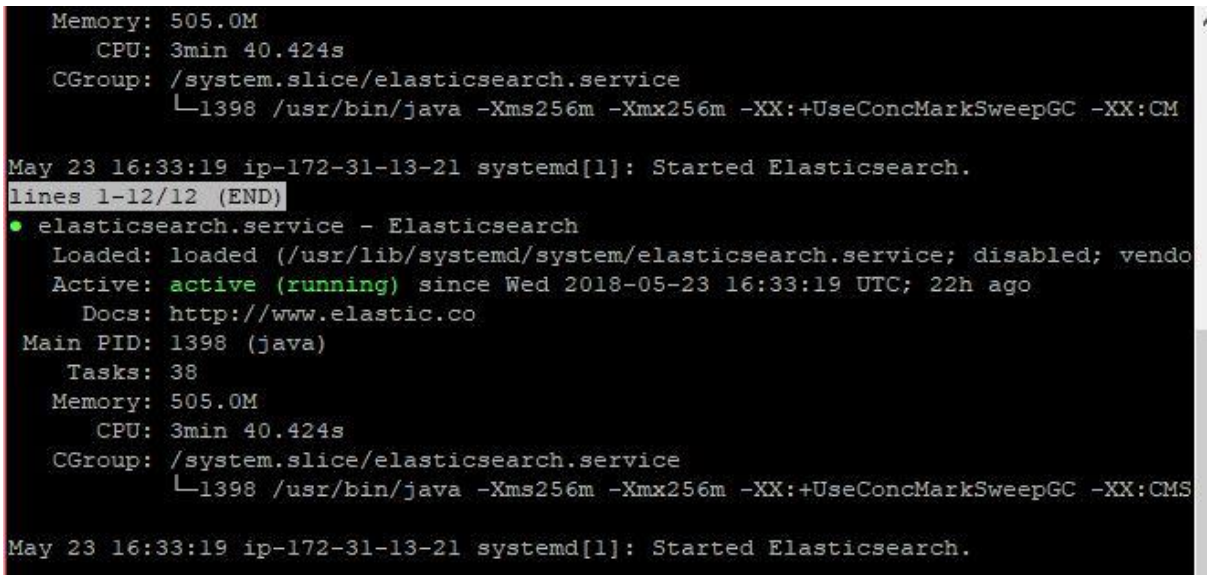
- Amazon AWS cloud service
- Virtual machine



- DataGrip 2018.1

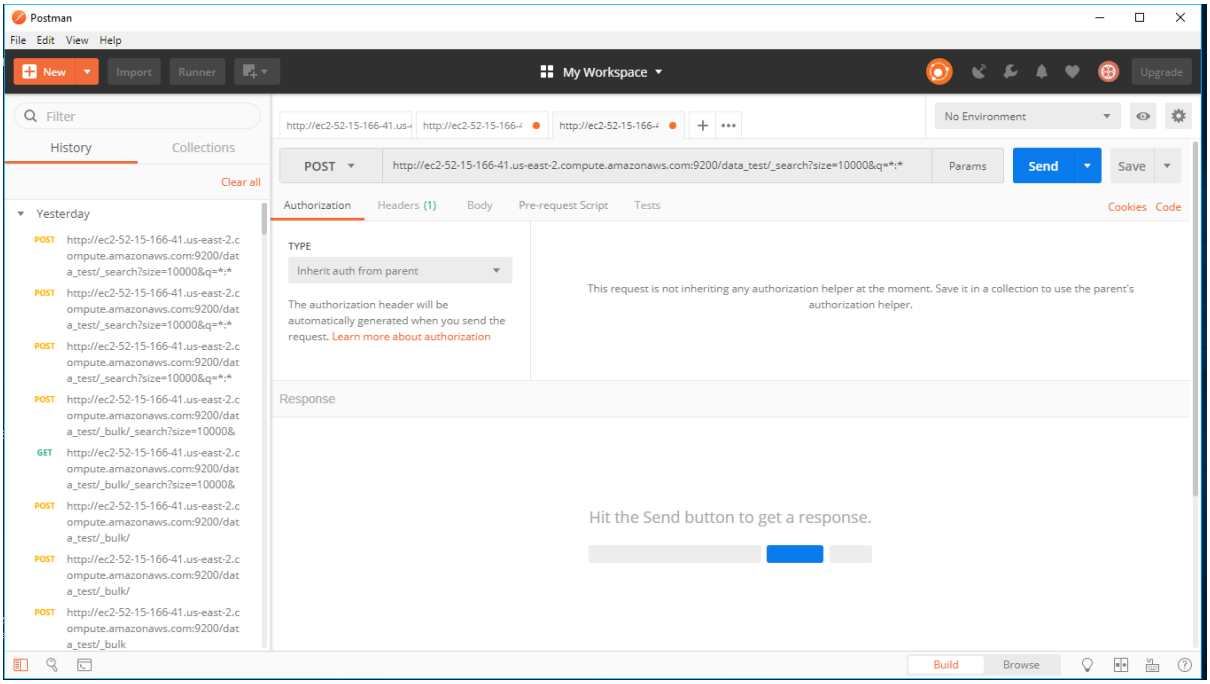


- Elastic Search



- PostMan

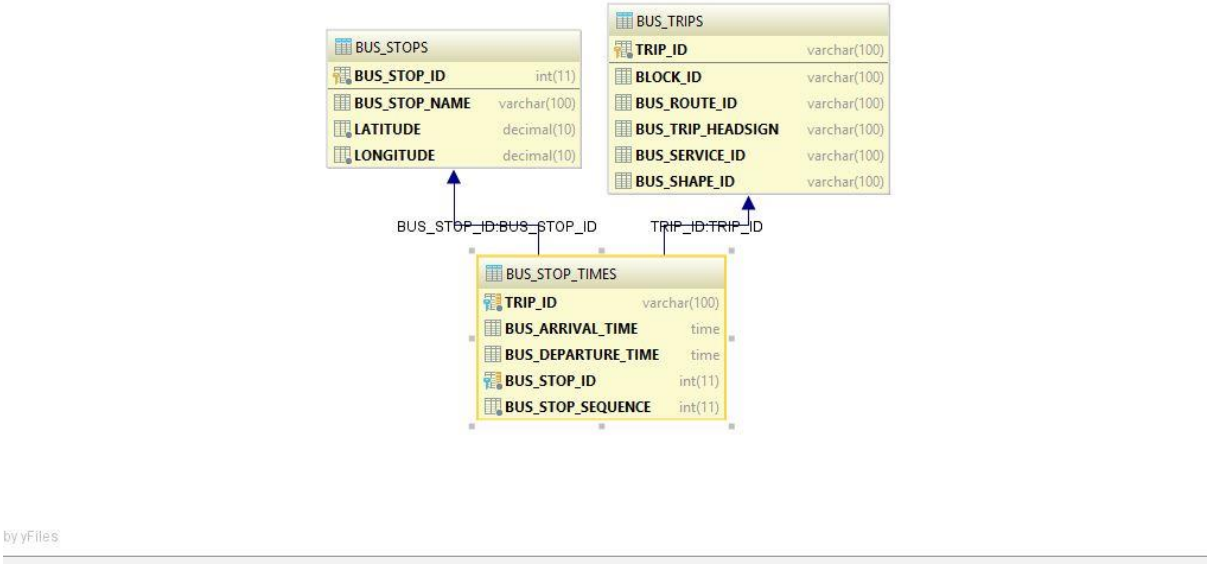
  The Postman API development application used to interact with elastic search in the AWS.



We also generated RSA keys by using PuTTYgen on Windows for secure SSH authentication with OpenSSH. This is used for secure SSH access to the cloud server by using public-private key pair.

2. RELATIONAL DATABASE DESIGN:

Data Grip 2018 is used for relational data base. As we encounter some issues while importing large csv file in MySQL workbench, we found that Data Grip provides a dedicated UI for importing csv and tsv files and better performance when compared to workbench.



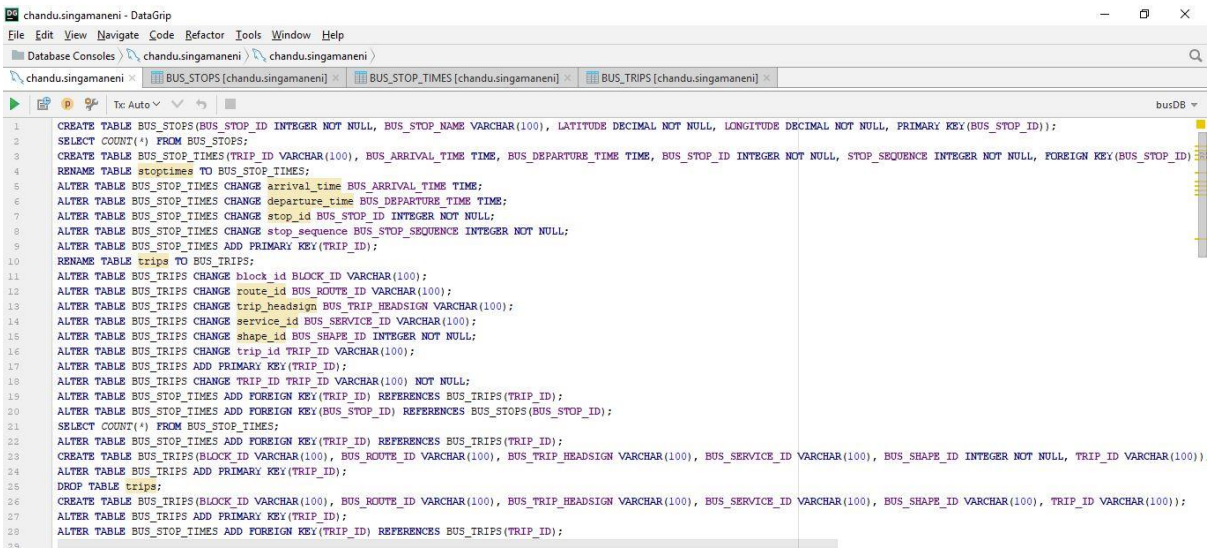
Screenshot: ER Diagram of the relational databases

Table1: BUS_STOPS (Primary Key: BUS_STOP_ID)

Table2: BUS_TRIPS (Primary Key: TRIP_ID)

Table3: BUS_STOP_TIMES (Foreign Keys: BUS_STOP_ID, TRIP_ID)

Data Formatting and setup



3. APPLICATION QUERIES:

The comparison is made between the RDBMS and Elastic search in Amazon services (Amazon AWS).

a. Find all buses for a particular Bus Stop

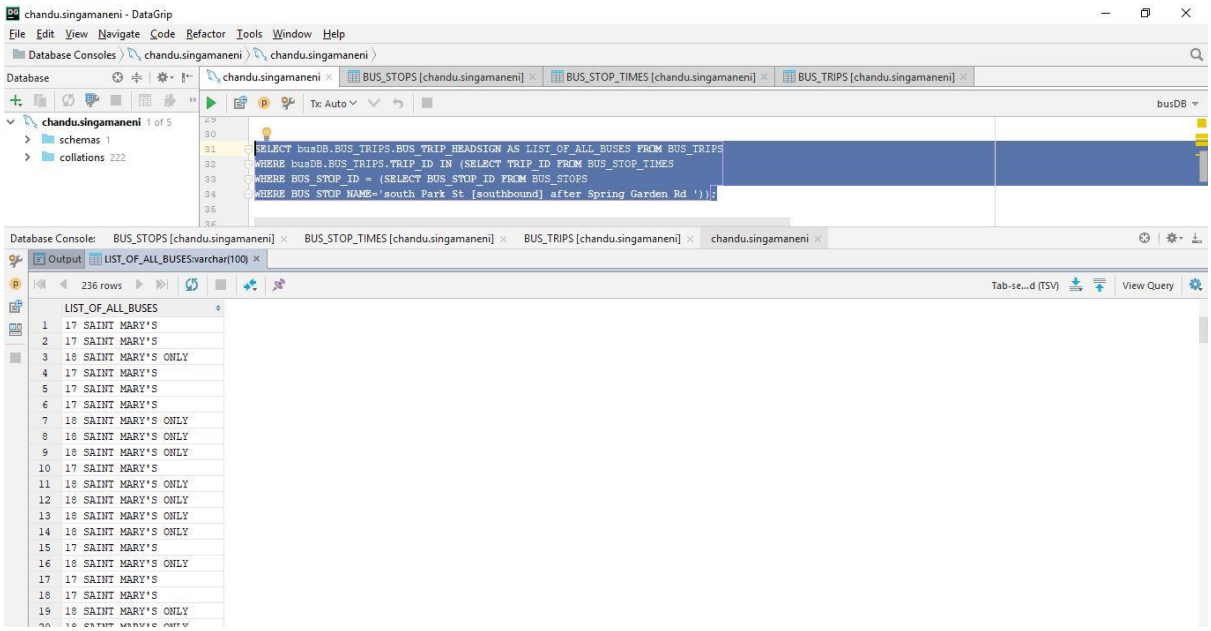
1.Input: Bus Stop Name

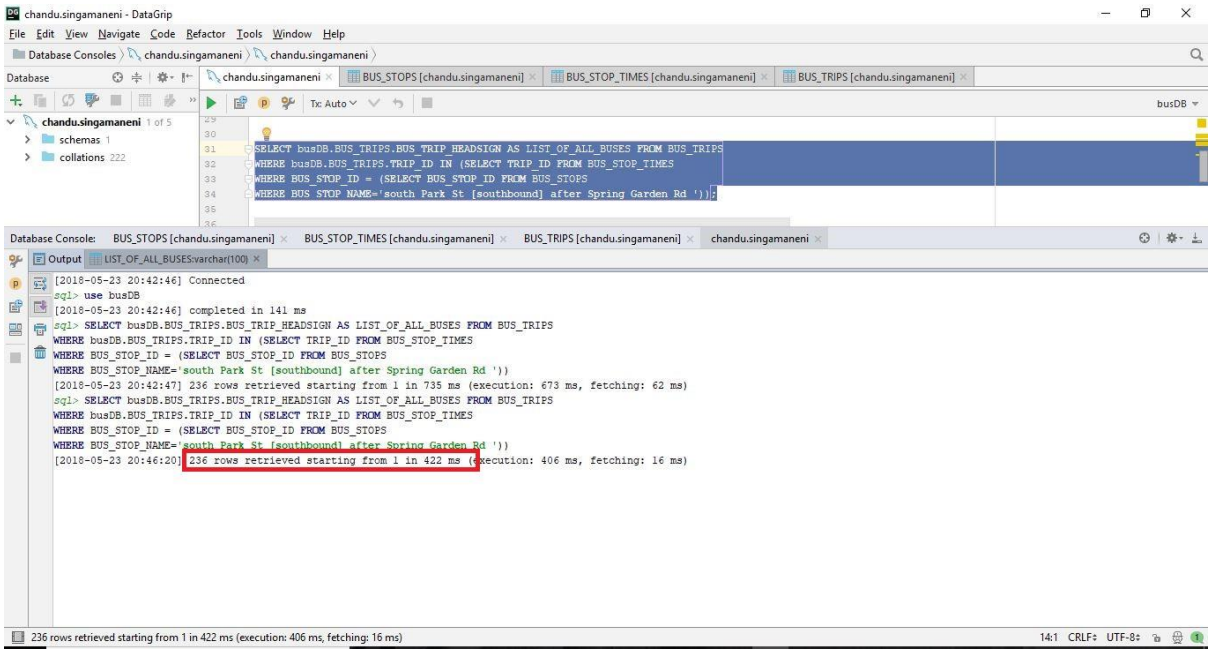
2.Output: List of all buses, response time for the search query

SQL Query

```
SELECT busDB.BUS_TRIPS.BUS_TRIP_HEADSIGN AS LIST_OF_ALL_BUSES FROM
BUS_TRIPS WHERE busDB.BUS_TRIPS.TRIP_ID IN (SELECT TRIP_ID FROM
BUS_STOP_TIMES WHERE BUS_STOP_ID = (SELECT BUS_STOP_ID FROM BUS_STOPS
WHERE BUS_STOP_NAME='south Park St [southbound] after Spring Garden Rd
'));
```

SQL Output





Response Time: 422ms

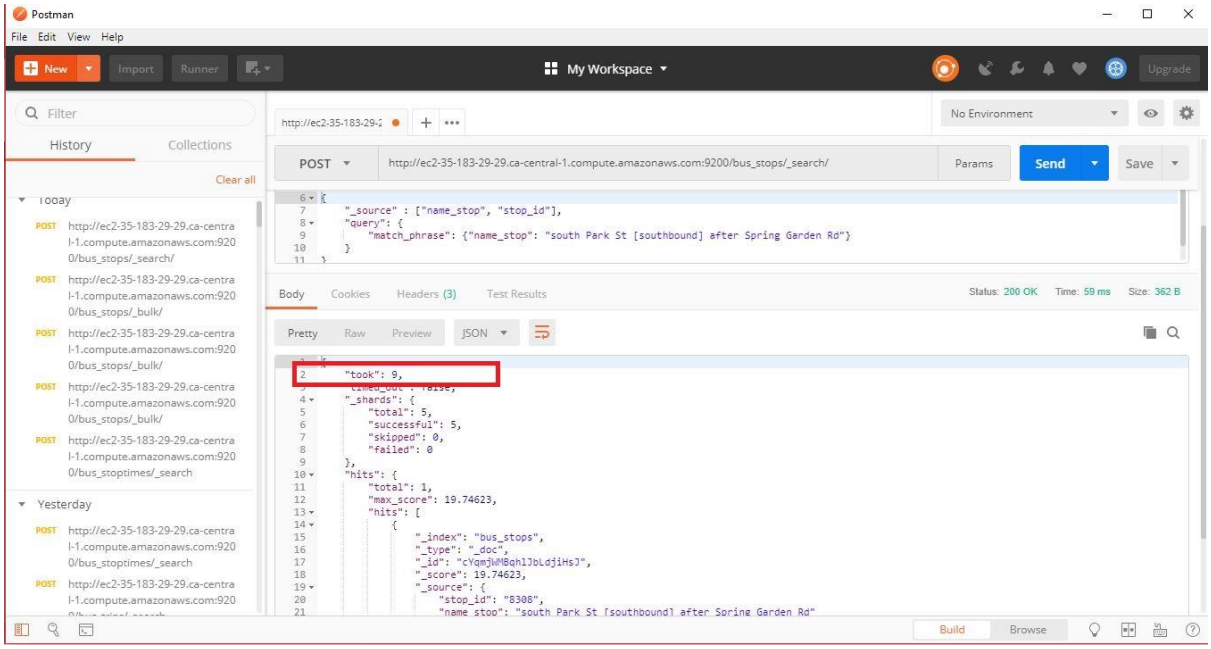
Elastic Search Query

Sub Query 1:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stops/_search

```
{
        "_source" : ["name_stop", "stop_id"],
   "query": {
      "match_phrase": {"name_stop": "south Park St [southbound] after Spring Garden Rd"}
   }
}
```

Output



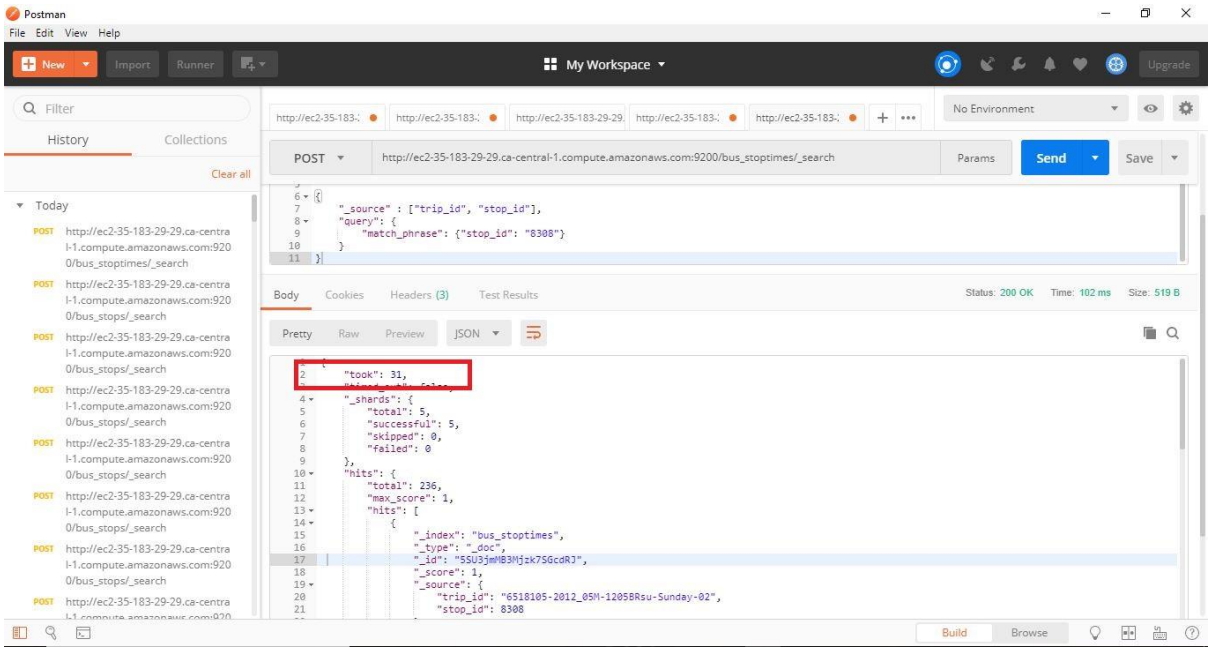
Response Time: 9ms

Sub Query 2:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stoptimes/_search

```
{
        "_source" : ["trip_id", "stop_id"],
   "query": {
      "match_phrase": {"stop_id": "8308"}
   }
}
```

Output



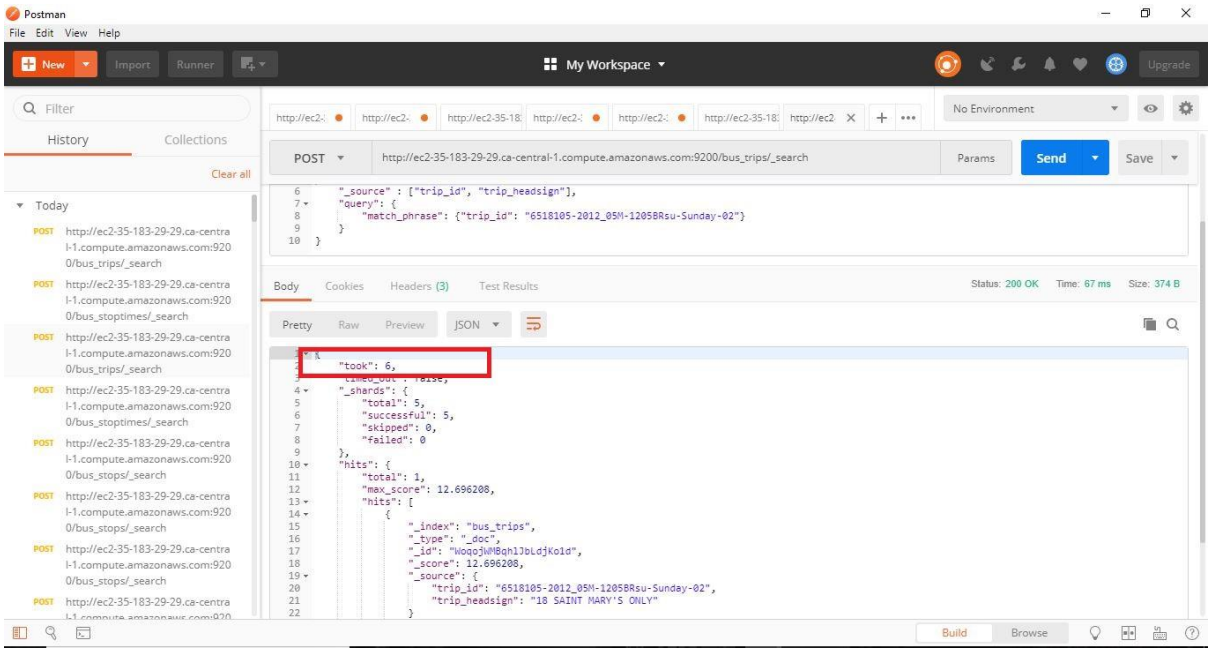
Response Time: 31ms

Sub Query 3:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_trips/_search

```
{
        "_source" : ["trip_id", "trip_headsign"],
   "query": {
      "match_phrase": {"trip_id": "6518105-2012_05M-1205BRsu-Sunday-02"}
   }
}
```

Output



Response Time: 6ms

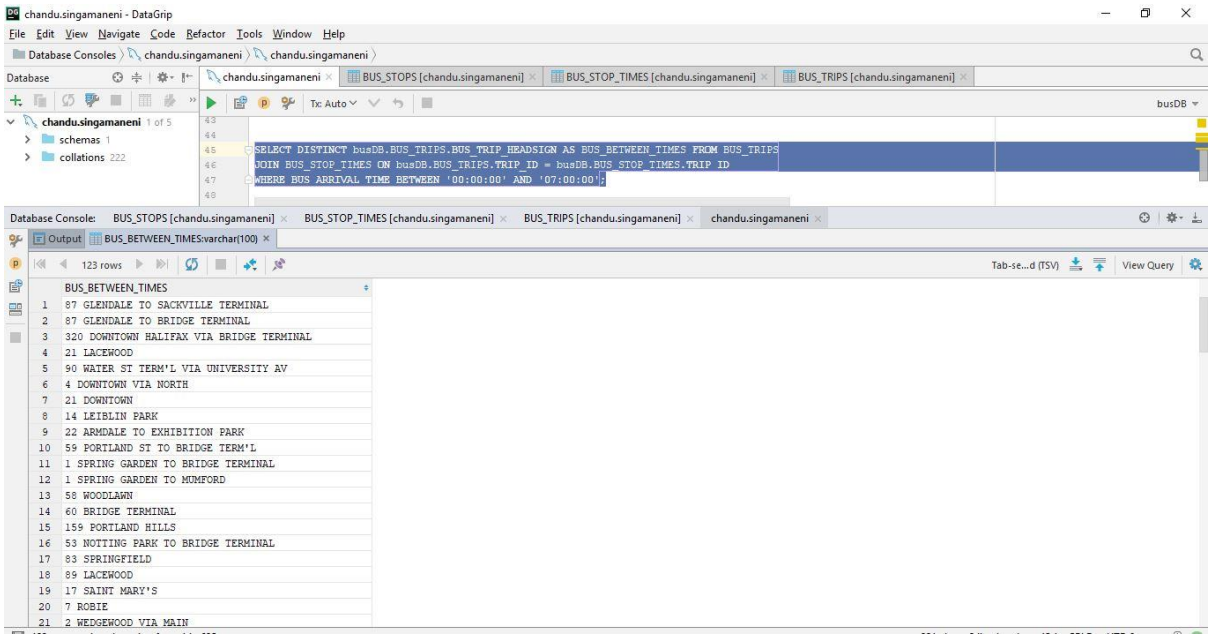
When compared both SQL and Elastic search response times (422ms, 46ms), the elastic search is more time efficient.

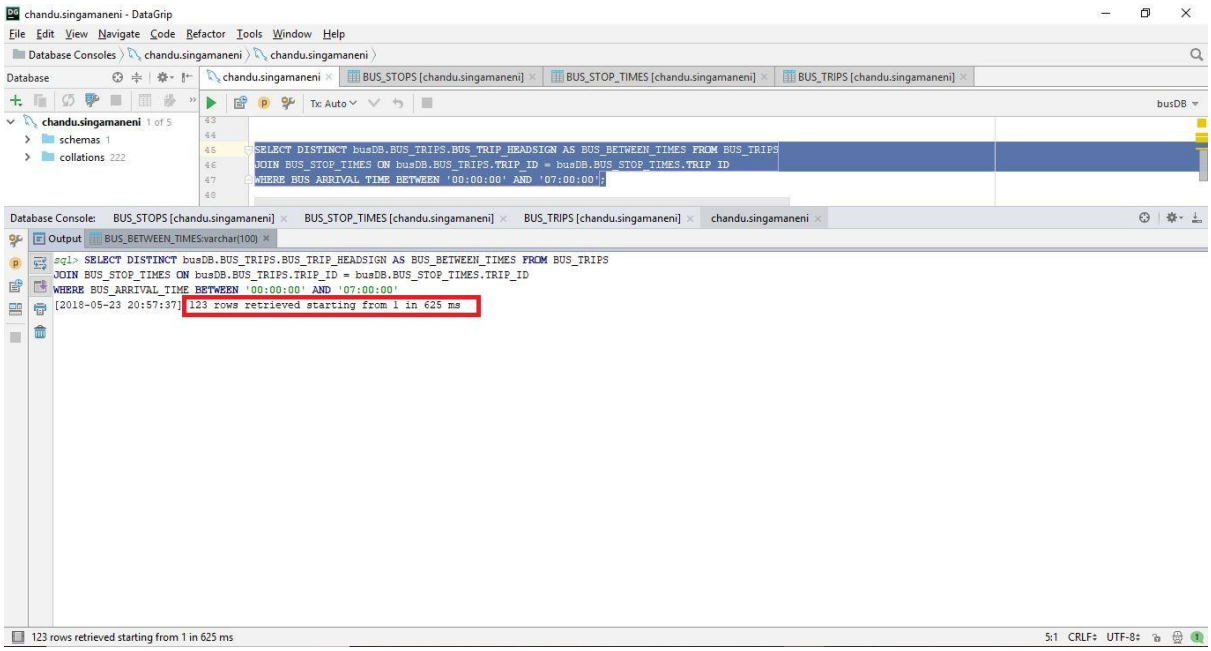
b. Find buses between two-time ranges
1.Input: Time Range 1 (hh:mm:ss), Time Range 2 (hh:mm:ss)
2.Output: List of all buses, response time for the search query

SQL Query

```
SELECT DISTINCT busDB.BUS_TRIPS.BUS_TRIP_HEADSIGN AS BUS_BETWEEN_TIMES
FROM BUS_TRIPS JOIN BUS_STOP_TIMES ON busDB.BUS_TRIPS.TRIP_ID =
busDB.BUS_STOP_TIMES.TRIP_ID WHERE BUS_ARRIVAL_TIME BETWEEN '00:00:00'
AND '07:00:00';
```

SQL Output





Response Time: 625ms

Elastic Search Query

Sub Query 1:

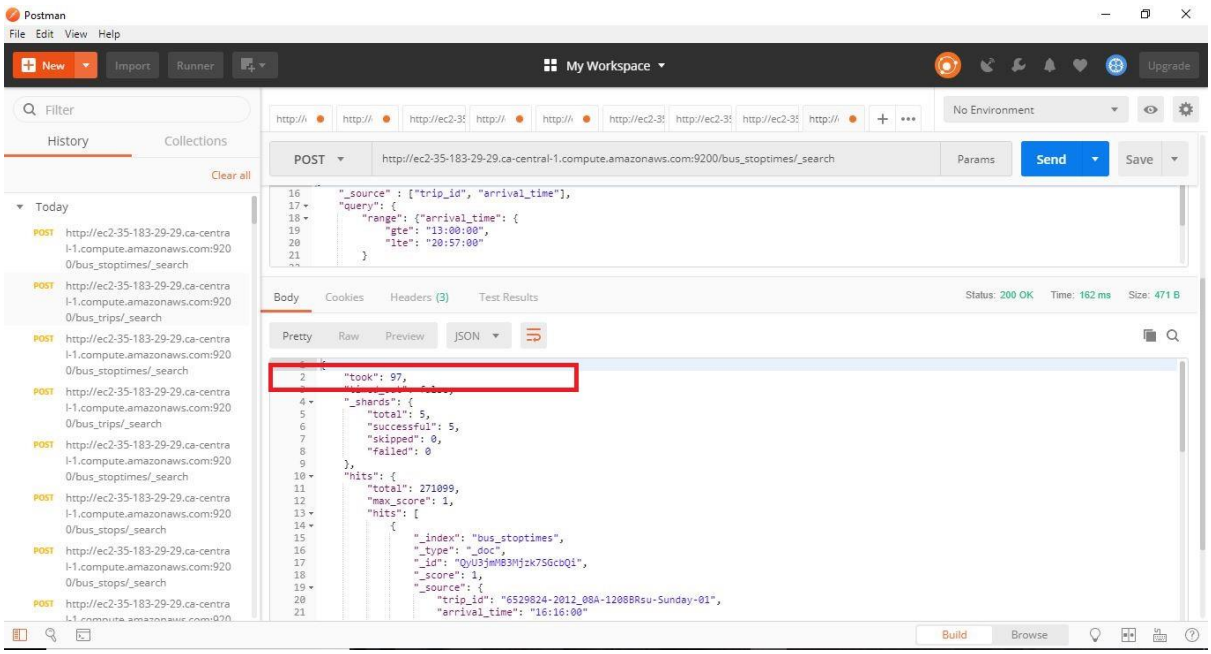
URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stoptimes/_search

```
{
        "_source" : ["trip_id", "arrival_time"],
   "query": {
```

```
        "range": {"arrival_time": {
            "gte": "13:00:00",
            "lte": "20:57:00"
        }

        }
    }
}
```

Output



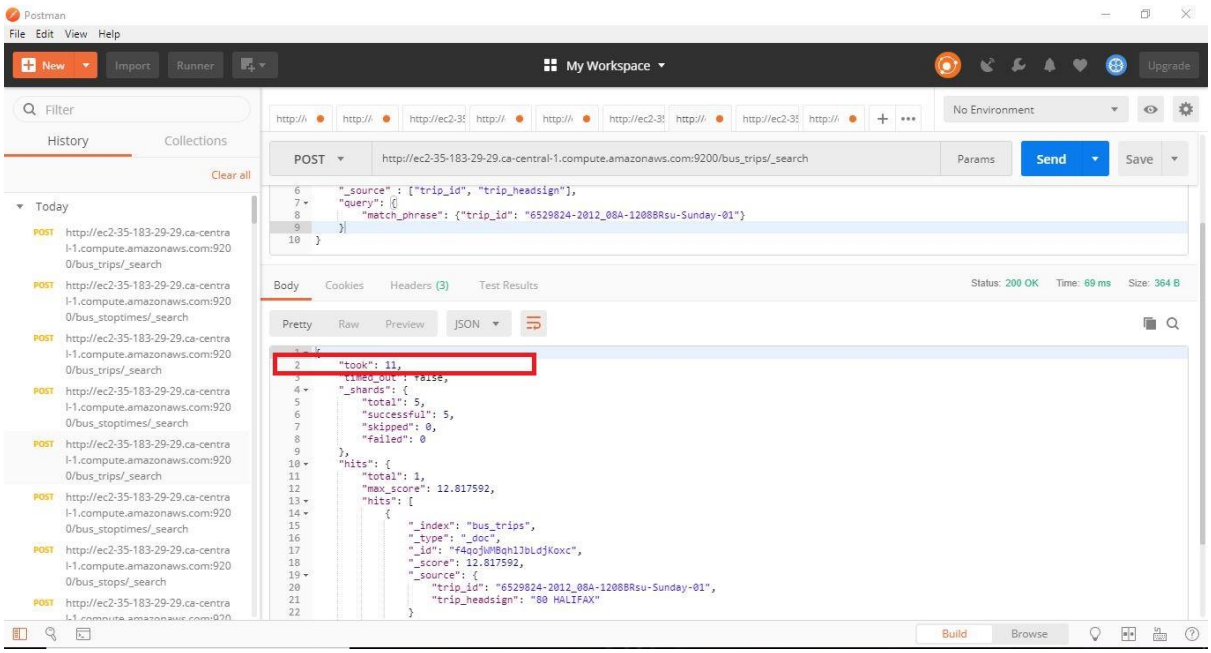
Response Time: 97ms

Sub Query 2:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_trips/_search

```
{       "_source" : ["trip_id", "trip_headsign"],
    "query": {
        "match_phrase": {"trip_id": "6529824-2012_08A-1208BRsu-Sunday-01"}
    }
}
```

Output



Response Time: 11ms

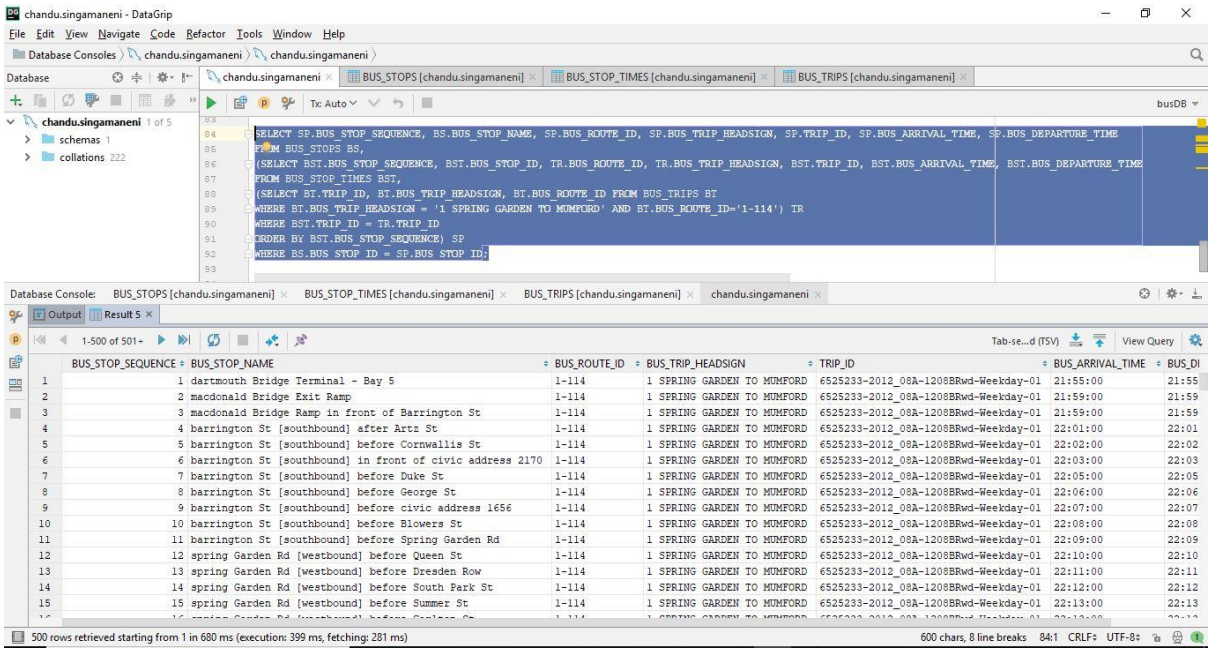
When compared both SQL and Elastic search response times (625ms, 108ms), the elastic search is more time efficient.

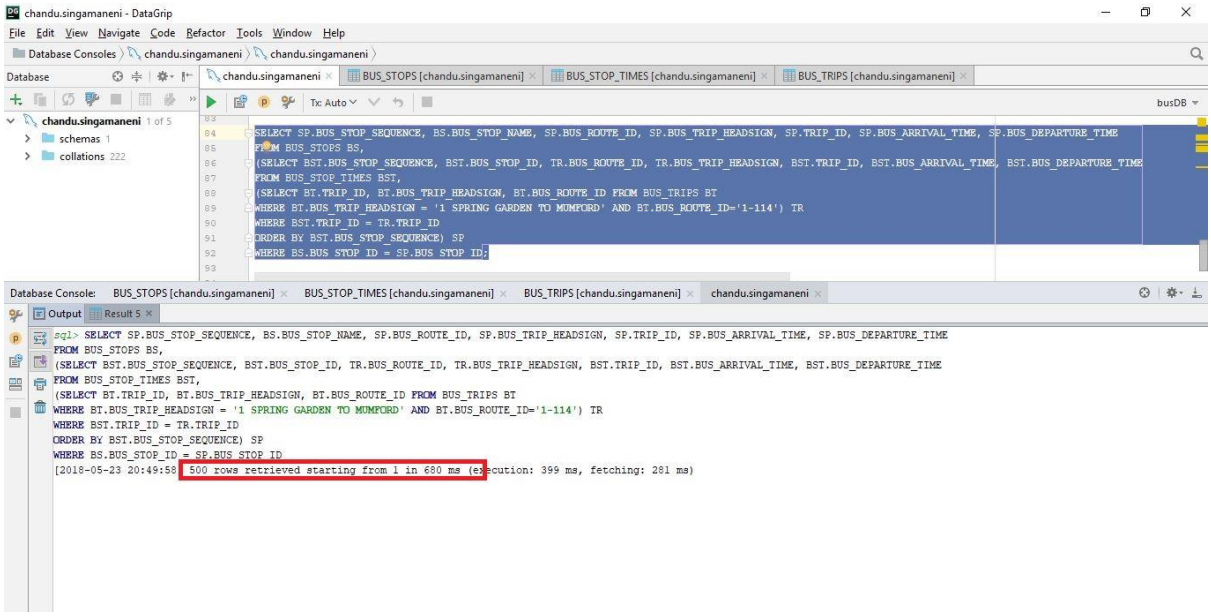
c. Find route information of a particular bus on a particular route
1.Input: Bus Name, Route Name
2.Output: List of all routes, response time for the search query

SQL Query

```
SELECT SP.BUS_STOP_SEQUENCE, BS.BUS_STOP_NAME, SP.BUS_ROUTE_ID,
SP.BUS_TRIP_HEADSIGN, SP.TRIP_ID, SP.BUS_ARRIVAL_TIME,
SP.BUS_DEPARTURE_TIME FROM BUS_STOPS BS,(SELECT BST.BUS_STOP_SEQUENCE,
BST.BUS_STOP_ID, TR.BUS_ROUTE_ID, TR.BUS_TRIP_HEADSIGN, BST.TRIP_ID,
BST.BUS_ARRIVAL_TIME, BST.BUS_DEPARTURE_TIME FROM BUS_STOP_TIMES
BST,(SELECT BT.TRIP_ID, BT.BUS_TRIP_HEADSIGN, BT.BUS_ROUTE_ID FROM
BUS_TRIPS BT WHERE BT.BUS_TRIP_HEADSIGN = '1 SPRING GARDEN TO MUMFORD'
AND BT.BUS_ROUTE_ID='1-114') TR WHERE BST.TRIP_ID = TR.TRIP_ID ORDER BY
BST.BUS_STOP_SEQUENCE) SP WHERE BS.BUS_STOP_ID = SP.BUS_STOP_ID;
```

SQL Output





Response Time: 680ms

Elastic Search Query

Sub Query 1:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_trips/_search

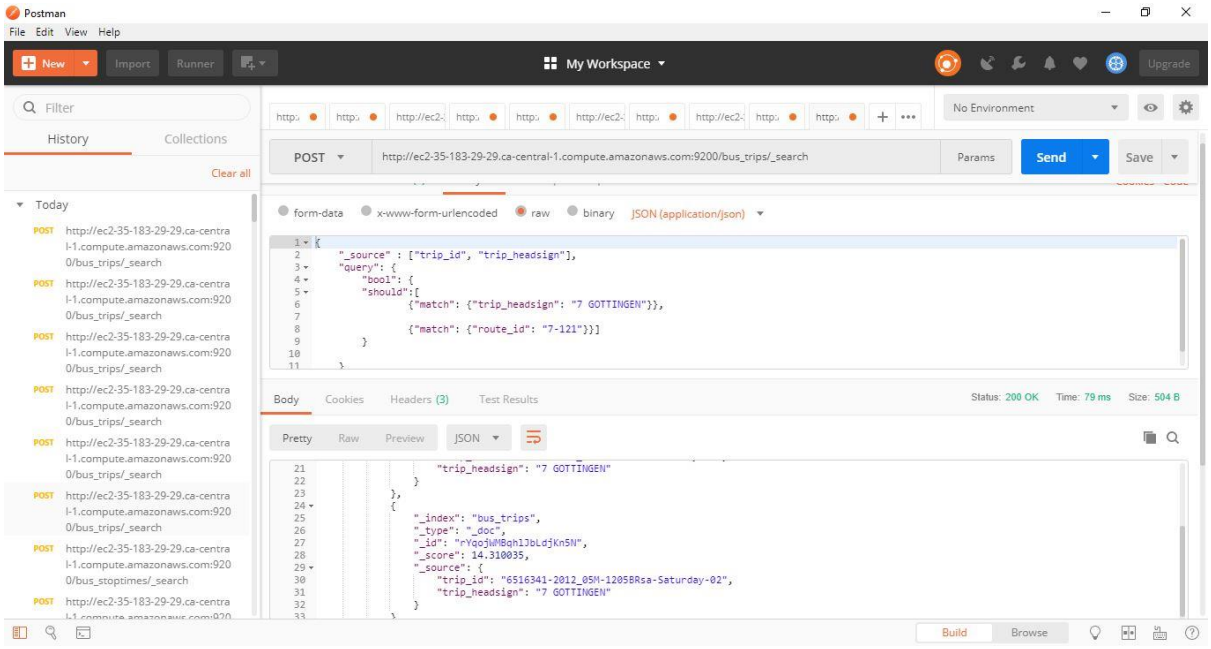
```
{
        "_source" : ["trip_id", "trip_headsign"],
   "query": {
```

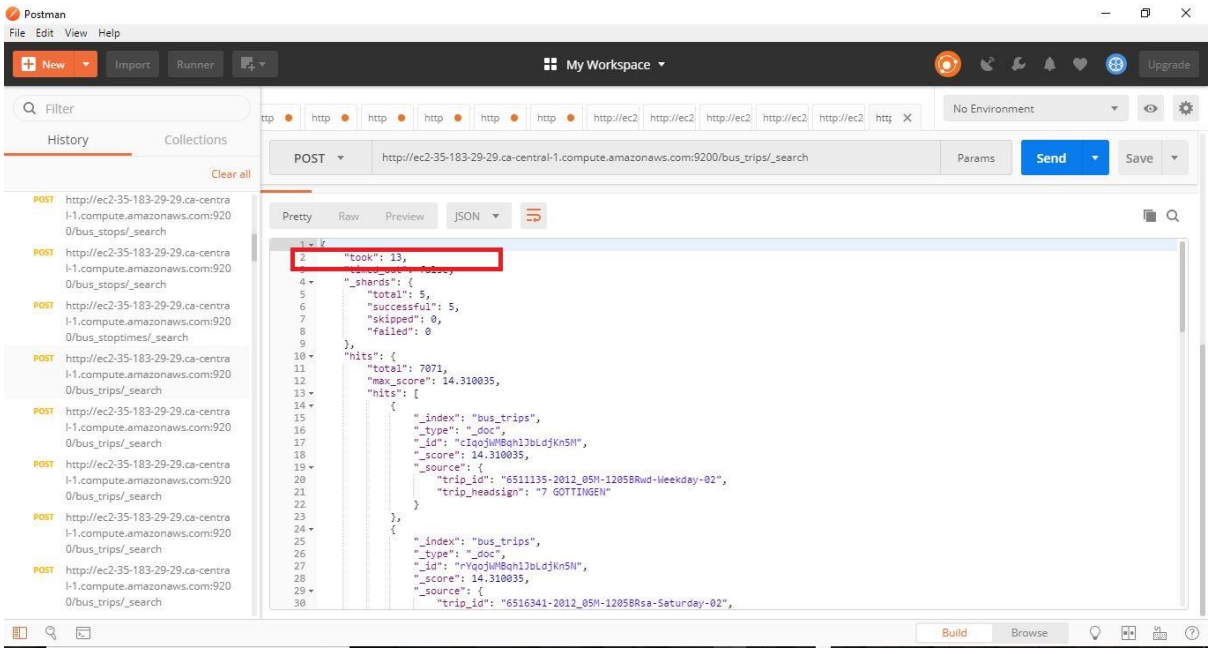
```
        "bool": {
                "should":[
                        {"match": {"trip_headsign": "7 GOTTINGEN"}},

                        {"match": {"route_id": "7-121"}}]
        }

    }
}
```

Output





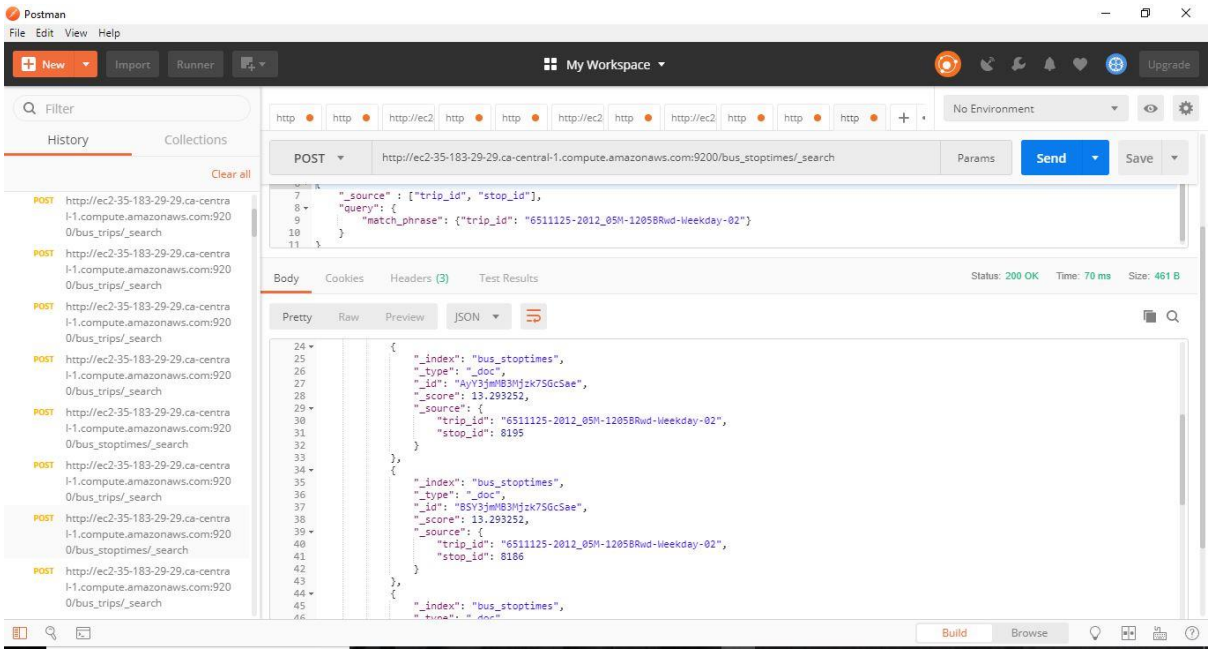
Response Time: 13ms

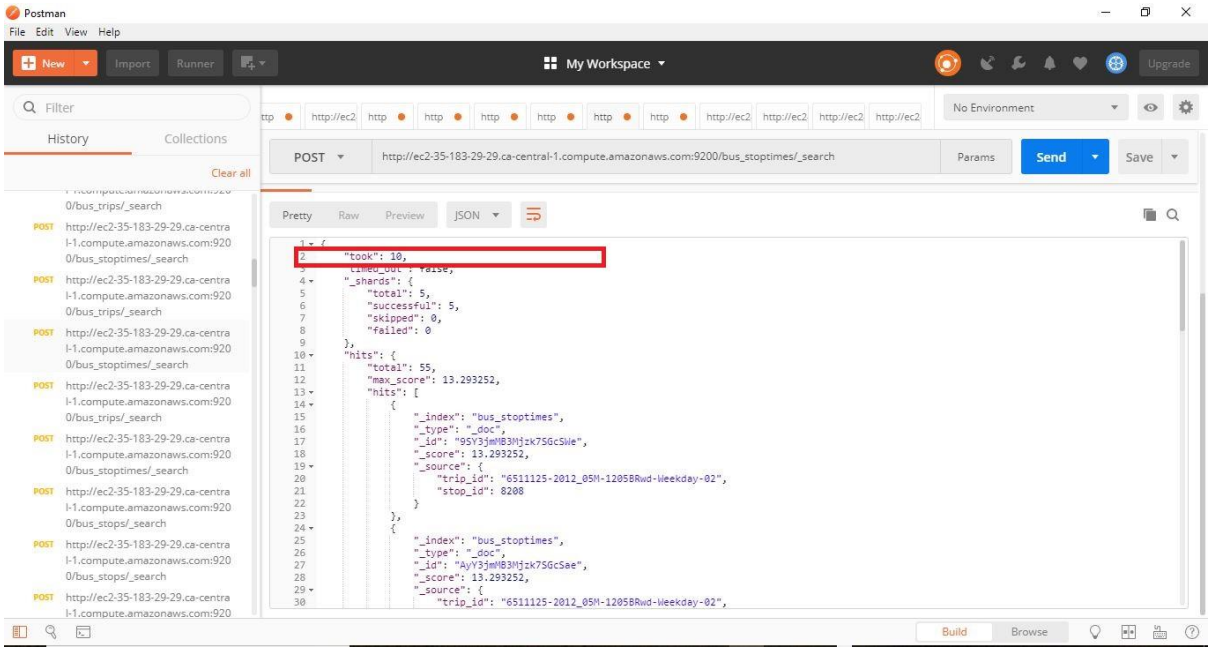
Sub Query 2:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stoptimes/_search

```
{
        "_source" : ["trip_id", "stop_id"],
   "query": {
      "match_phrase": {"trip_id": "6511125-2012_05M-1205BRwd-Weekday-02"}
   }
}
```

Output





Response Time: 10ms

Sub Query 3:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stops/_search

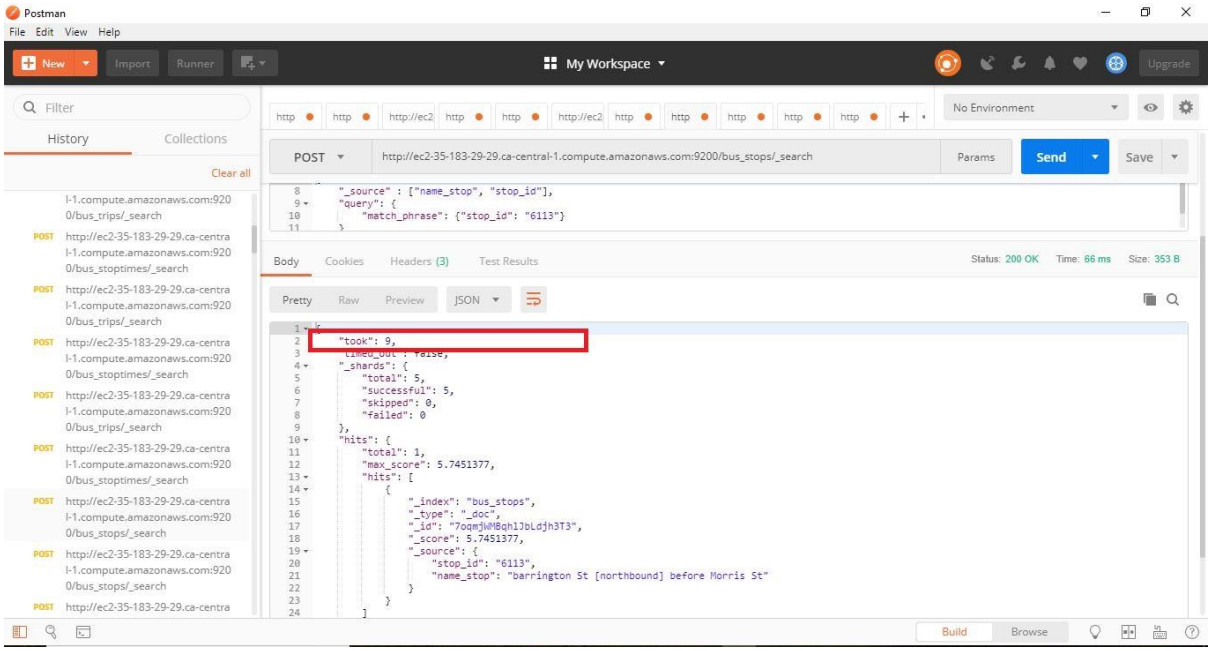
{

"_source" : ["name_stop", "stop_id"],

"query": {

```
        "match_phrase": {"stop_id": "6113"}
    }
}
```

Output



Response Time: 9ms

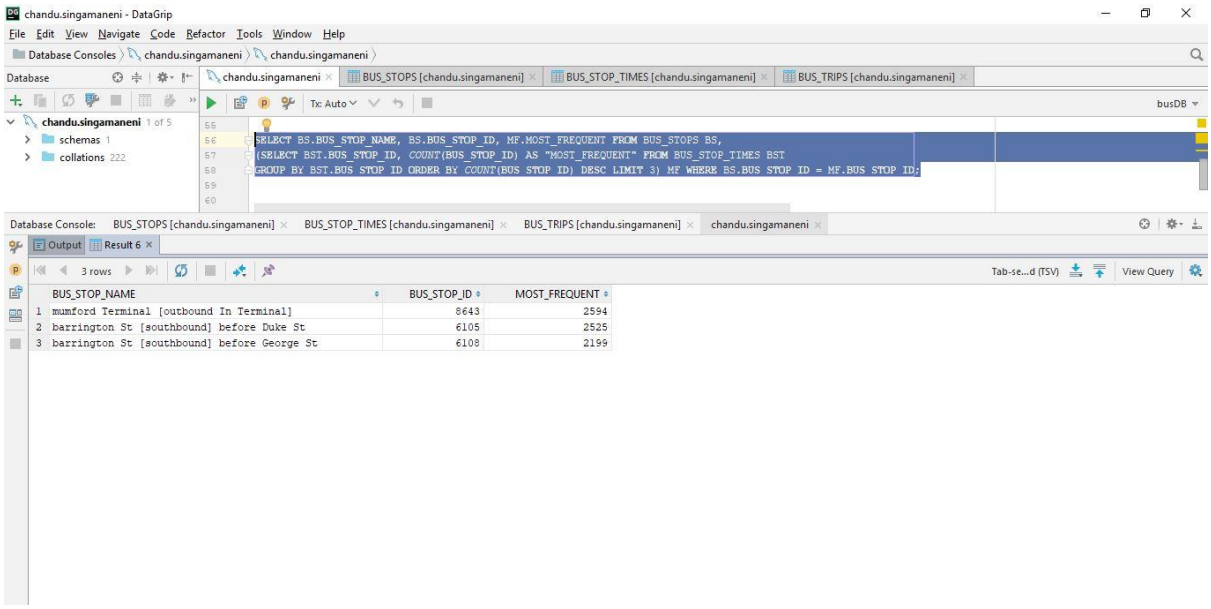
When compared both SQL and Elastic search response times (680ms, 32ms), the elastic search is more time efficient.

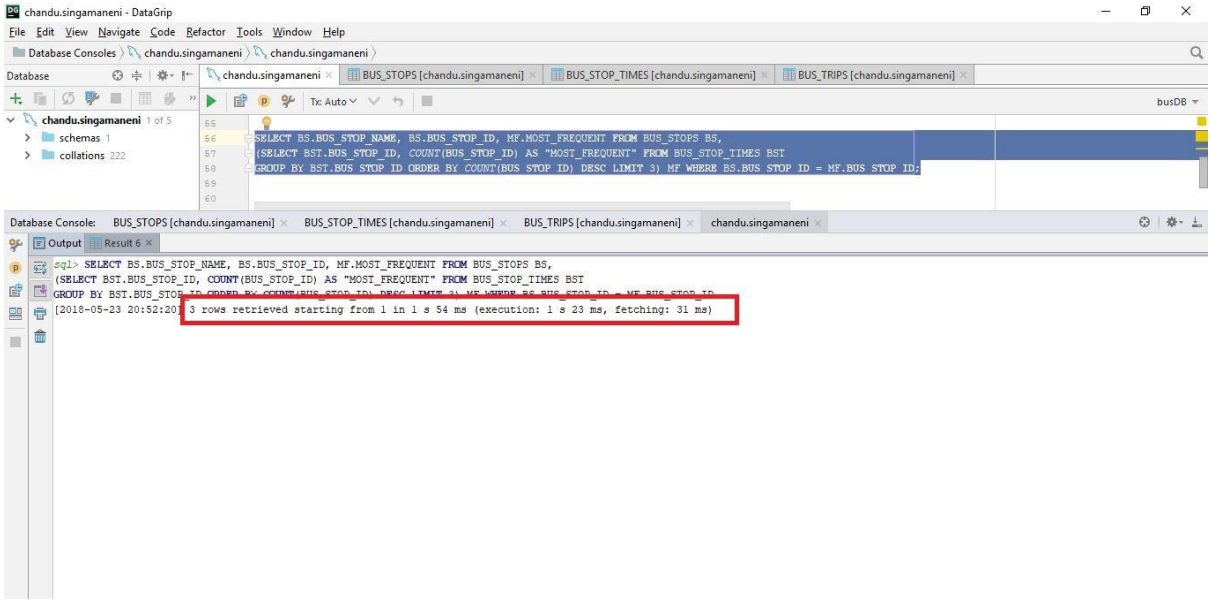
d. Find top 3 bus stops that are the busiest throughout the day in terms of bus routes. (Hint: The bus stops with high volume of bus routes and close time gaps would be considered as busiest).
1.Input: None
2.Output: List of Bus Name, response time for the search query

SQL Query

```
SELECT BS.BUS_STOP_NAME, BS.BUS_STOP_ID, MF.MOST_FREQUENT FROM
BUS_STOPS BS,(SELECT BST.BUS_STOP_ID, COUNT(BUS_STOP_ID) AS
"MOST_FREQUENT" FROM BUS_STOP_TIMES BST GROUP BY BST.BUS_STOP_ID ORDER
BY COUNT(BUS_STOP_ID) DESC LIMIT 3) MF WHERE BS.BUS_STOP_ID =
MF.BUS_STOP_ID;
```

SQL Output





Response Time: 1054ms

Elastic Search Query

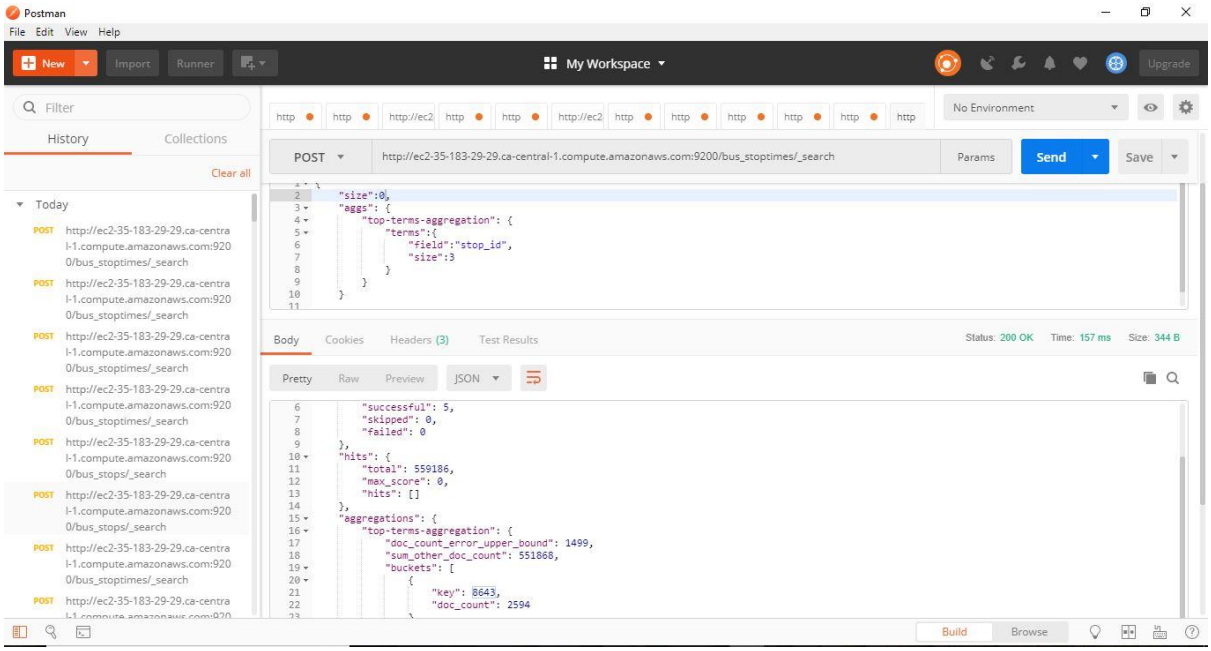
Sub Query 1:

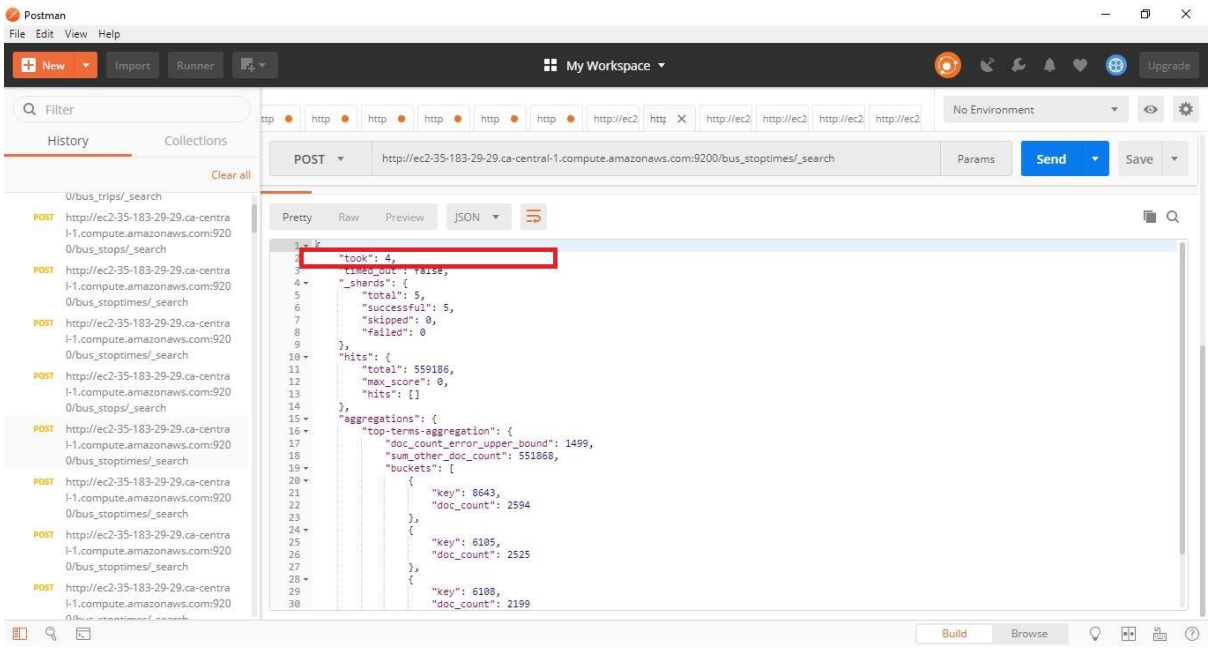
URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stoptimes/_search

```
{
        "size":0,
        "aggs": {
                "top-terms-aggregation": {
```

```
                "terms":{
                        "field":"stop_id",
                        "size":3
                }
            }
        }
}
```

Output





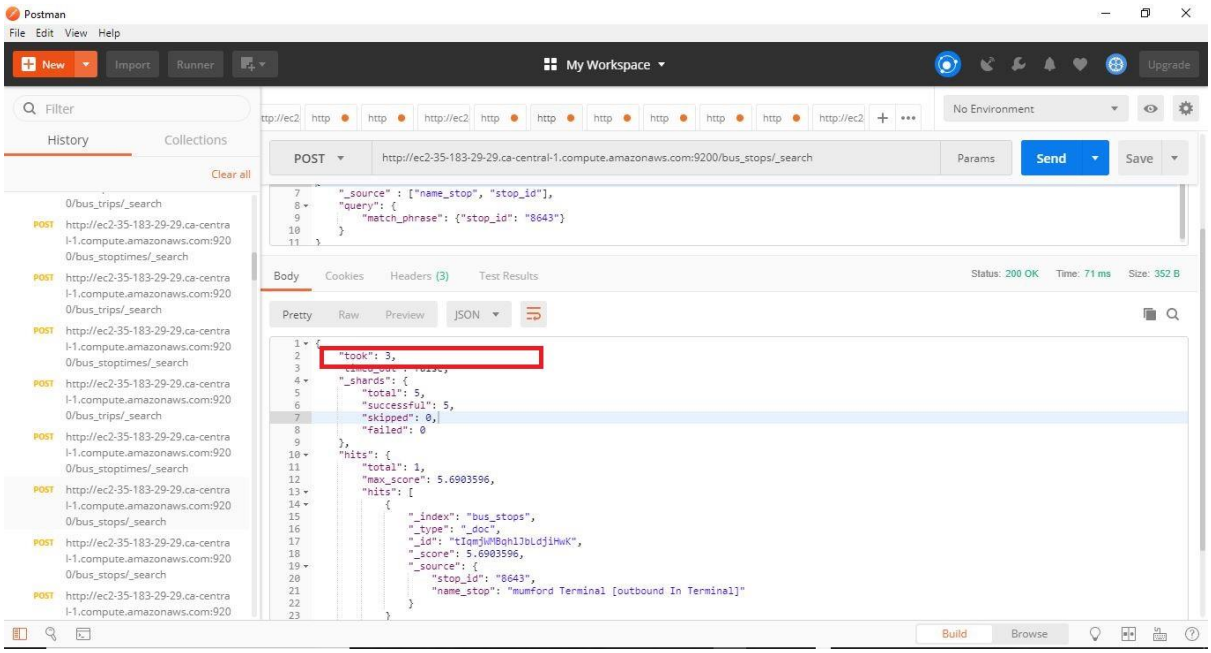
Response Time: 4ms

Sub Query 2:

URL: http://ec2-35-183-29-29.ca-central-1.compute.amazonaws.com:9200/bus_stops/_search

```
{
        "_source": ["name_stop", "stop_id"],
   "query": {
      "match_phrase": {"stop_id": "8643"}
   }
}
```

Output

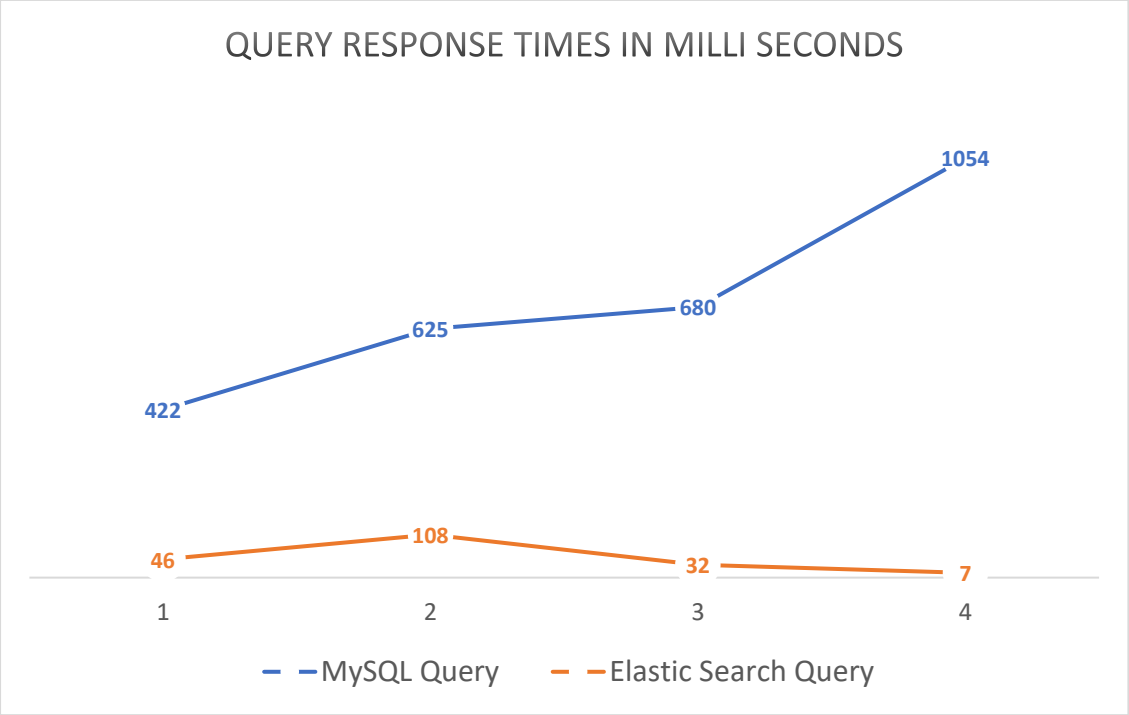


Response Time: 3ms

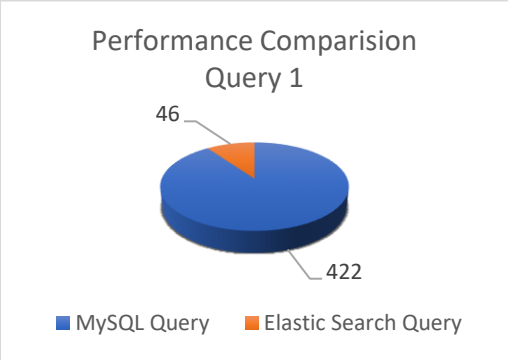
When compared both SQL and Elastic search response times (1054ms, 7ms), the elastic search is more time efficient.

4. TEST RESULTS:

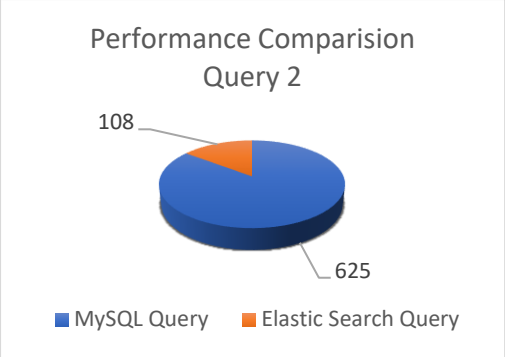
The below diagram is the timeline for the time taken by all the queries to execute and fetch the data from the server.



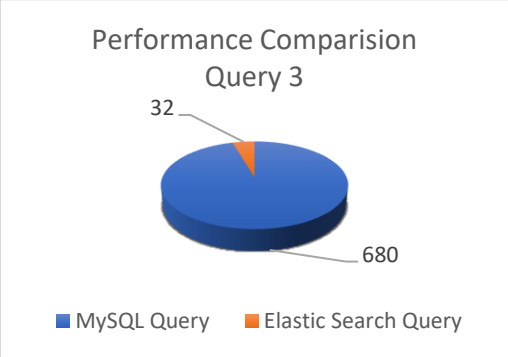
The below diagrams are the comparison of the performance of MySQL and Elastic Search query on the basis execution and fetch time



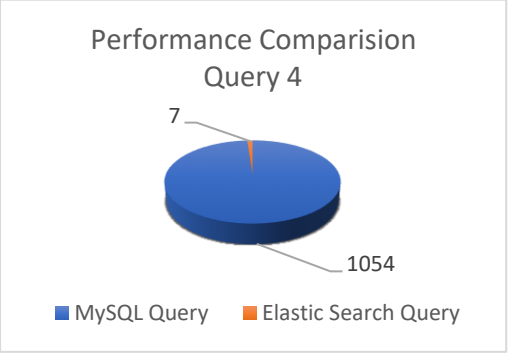
ELASTIC SEARCH QUERY IS 9 FASTER THAN SQL QUERY



ELASTIC SEARCH QUERY IS 6 FASTER THAN SQL QUERY



ELASTIC SEARCH QUERY IS 21 FASTER THAN SQL QUERY



ELASTIC SEARCH QUERY IS 150 FASTER THAN SQL QUERY

5. SUMMARY:

This assignment helped us to explore different applications such as Amazon could services, MySQL Workbench, Data Grip, Elastic search and Postman. The Halifax transit data was used in classical relational database and compared with the elastic search i.e., NoSQL database to analyse the performance and execution of queries in retrieving data. After using all these tools, we had an opportunity to learn how to use the infrastructure services on amazon cloud and implementation and connectivity of different databases on cloud. Finally, comparing both the response times of MySQL and Elastic search, we analysed that queries in elastic search executed more rapidly and is better and faster mechanism to retrieve and store data. Although, the implementation in traditional databases are easier than the NoSQL databases.

References

“Amazon Web Services (AWS) - Cloud Computing Services,” Amazon. [Online]. Available: https://aws.amazon.com/. [Accessed: 23-May-2018].

“DataGrip: Cross-Platform IDE for Databases & SQL by JetBrains,” JetBrains. [Online]. Available: https://www.jetbrains.com/datagrip/. [Accessed: 23-May-2018].

“Must match multiple values,” Stack Overflow. [Online]. Available: https://stackoverflow.com/questions/35583781/must-match-multiple-values?utm_medium=organic&utm_source=google_rich_qa&utm_campaign=google_rich_qa. [Accessed: 23-May-2018].

“MySQL WorkbenchDownload Now »,” MySQL. [Online]. Available: https://www.mysql.com/products/workbench/. [Accessed: 23-May-2018].

“Open Source Search & Analytics · Elasticsearch,” Open Source Search & Analytics · Elasticsearch. [Online]. Available: https://www.elastic.co/. [Accessed: 23-May-2018].

“Postman,” Debugging and logs. [Online]. Available: https://www.getpostman.com/. [Accessed: 24-May-2018].