



---

# CSCI 5408 ASSIGNMENT 1

---

By: Bhargav Dalal (B00785773) & Hardik Galiawala (B00777450)



**DALHOUSIE  
UNIVERSITY**

MAY 23, 2018

DALHOUSIE UNIVERSITY  
Submitted to: Suhaib Qaiser

## Table of Contents

1. Objective: .....	2
2. Source code:.....	2
3. Task Description:.....	2
4. Relational Database Design: .....	3
5. Application Queries:.....	4
5.1. MySQL Queries:.....	4
5.2. Elastic Search: .....	6
6. Summary: .....	12
6.1. Data extraction and manipulation: .....	12
6.2. Cloud computing power: .....	13
7. Conclusion:.....	14

### 1. Objective:

The main purpose of this assignment is to review conventional RDBMS, learn Infrastructure Services on a cloud system, learn Elastic Search concepts and implementation, and learn installation of MySQL (RDBMS) on a Cloud system.

### 2. Source code:

[Link to github account.](#)

### 3. Task Description:

The task was to compare the performance of the RDBMS structure database and NOSQL structure database. The data was obtained from [website](#). The data includes information about the number of bus stops(stops), trips of each bus(trips), bus names and routes(stoptimes). Four tasks were given to assess the performance. These four tasks were first performed on MYSQL (RDBMS structure-based database). Later, these tasks were performed on Elastic Search (NOSQL structure-based database).

We have installed conventional RDBMS (MySQL) in an instance which is created on EC2 AWS cloud infrastructure. Then we created a schema in MySQL where we created the structure of the tables along with the relationship. The dataset was given in the form of insert statements as well as .csv files. We used insert statements to populate the data in tables. So, by now our relational database is ready with the required data. Then we used MySQL Workbench as a client to interact with our database in the cloud. We fired the below-mentioned queries and captured their response time.

## 4. Relational Database Design:

As we have already mentioned in the above section about the database being ready, it is a good idea to understand the relationship of the database objects in detail. This scenario consists of three database objects namely STOPS, STOPTIMES, and TRIPS.

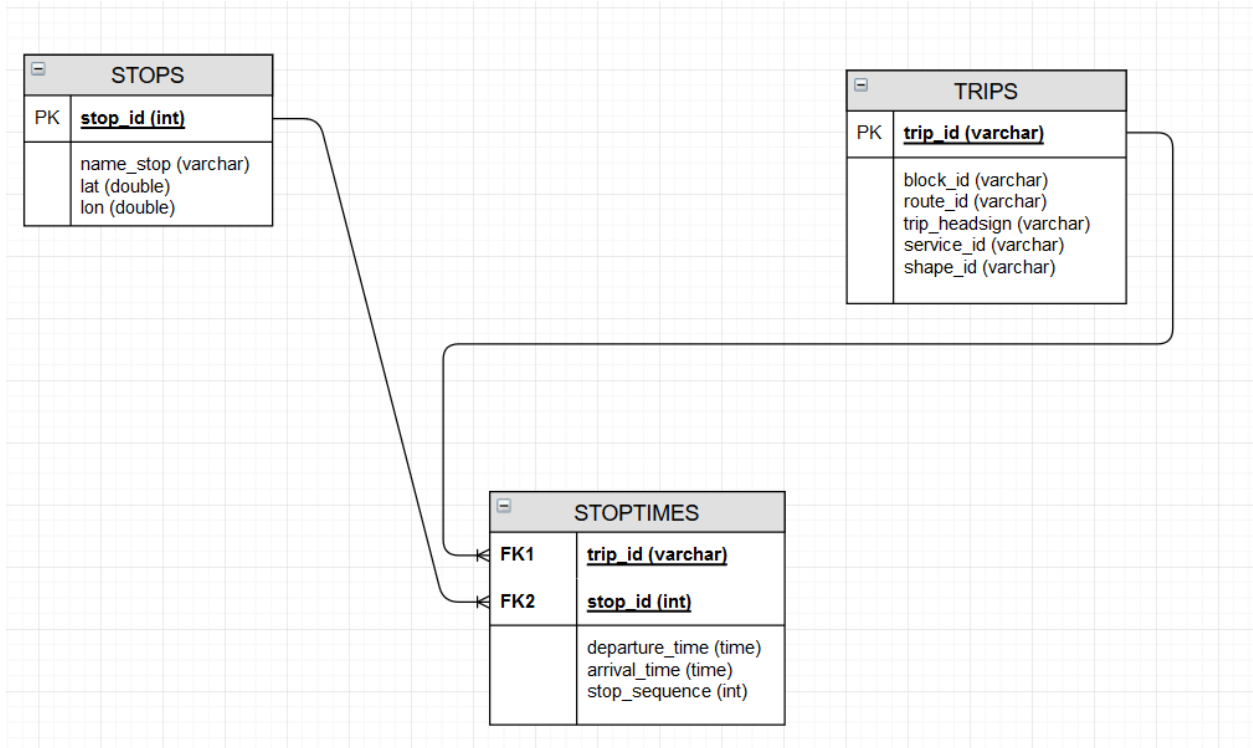


Figure: 3.1 ER Diagram

As from the Figure 3.1, it is clear that STOPS.stop\_id is the primary key and is referenced to STOPTIMES.stop\_id as a foreign key. Similarly, TRIPS.trip\_id is a primary key which is referenced to STOPTIMES.trip\_id. STOPTIMES.trip\_id is a foreign key. It is important to note that both the columns with primary key have an index on them. The storage engine creates it by default.

## 5. Application Queries:

### 5.1. MySQL Queries:

#### 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

Query 1

```

1 select
2   distinct tr.trip_headsign
3   from
4   trips tr
5   join stopTimes sti on (tr.trip_id = sti.trip_id)
6   join stops st on (sti.stop_id = st.stop_id)
7   where
8   st.name_stop = 'Ferry Stop - Alderney';
9

```

Result Grid

trip_headsign
FERRY TO HALIFAX
FERRY TO DARTMOUTH

Result 9 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:50:32	select distinct tr.trip_headsign from trips tr join stopTimes sti on (tr.trip_id = sti.trip_id) join stops st on (sti.stop_id = ...	2 row(s) returned	0.078 sec / 0.000 sec

Ans:

1. Ferry to Dartmouth

2. Ferry to Halifax

Total time taken –78 ms

### b. Find buses between two range

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

The screenshot shows a SQL query editor with the following query:

```

1 select
2     distinct tr.trip_headsign
3 from
4     trips tr
5 join stopTimes sti on (tr.trip_id = sti.trip_id)
6 where
7     sti.arrival_time between '20:57:00' and '20:57:00';
8

```

The result grid displays a list of trip\_headsigns:

trip_headsign
53 NOTTING PARK TO HIGHFIELD TERMINAL
60 EASTERN PASSAGE
FERRY TO DARTMOUTH
66 COBEQUID
59 DOWNTOWN HALIFAX TO SUMMER ST
10 DALHOUSIE
21 LACEWOOD
10 WESTPHAL
1 SPRING GARDEN TO BRIDGE TERMINAL
72 PORTLAND HILLS VIA WOODLAWN
9 BARRINGTON TO TOWER RD LOOP
7 ROBIE
62 WILDWOOD
80 SACKVILLE
6 STONEHAVEN
68 PORTLAND ST TO CHERRYBROOK
80 HALIFAX
14 DOWNTOWN
20 HERRING COVE
52 CROSTOWN TO BAYERS LAKE
19 GREYSTONE
22 ARMDALE TO MUMFORD

The output section shows the query execution details:

#	Time	Action	Message	Duration / Fetch
1	22:56:20	select distinct tr.trip_headsign from trips tr join stopTimes sti on (tr.trip_id = sti.trip_id) where sti.arrival_time between '20:57:00' and '20:57:00';	59 row(s) returned	0.312 sec / 0.000 sec

Ans:

Total time taken – 312 ms

### 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.

The screenshot shows a SQL query editor with the following query:

```

1 select
2     distinct st.name_stop
3 from
4     trips tr
5 join stopTimes sti on (tr.trip_id = sti.trip_id)
6 join stops st on (sti.stop_id = st.stop_id)
7 where
8     tr.trip_headsign = 'FERRY TO HALIFAX'
9     and tr.route_id = 'FerD-116';

```

The result grid displays a list of name\_stop values:

name_stop
Ferry Stoo - Alderney
Ferry Stoo - Halifax

The output section shows the query execution details:

#	Time	Action	Message	Duration / Fetch
1	23:04:24	select distinct st.name_stop from trips tr join stopTimes sti on (tr.trip_id = sti.trip_id) join stops st on (sti.stop_id = st.stop_id) where tr.trip_headsign = 'FERRY TO HALIFAX' and tr.route_id = 'FerD-116';	2 row(s) returned	0.079 sec / 0.000 sec

Ans:

1. Ferry to Dartmouth
2. Ferry to Halifax

Total time taken – 79 ms

- 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

Query 1

```

1 select
2   st.name_stop,
3   count(sti.trip_id) AS coun
4 from
5   trips tr
6 join stopTimes sti on (tr.trip_id = sti.trip_id)
7 join stops st on (sti.stop_id = st.stop_id)
8 group by
9   st.name_stop
10 order by 2 desc limit 3;
11

```

name_stop	coun
mumford Terminal [outbound In Terminal]	2594
barrington St [southbound] before Duke St	2525
barrington St [southbound] before George St	2199

Result 15 x

Output

#	Time	Action	Message	Duration / Fetch
1	23:07:39	select st.name_stop, count(sti.trip_id) AS coun from trips tr join stopTimes sti on (tr.trip_id = sti.trip_id) join stops ...	3 row(s) returned	1.625 sec / 0.000 sec

Ans.

- mumford Terminal [outbound In Terminal]
- barrington St [southbound] before Duke St
- barrington St [southbound] before George St

Total time taken – 1625 ms

## 5.2. Elastic Search:

In our approach, we have created separate index and document for each schema (tripdb/trips, stopdb/stops, and stopptimesdb/stoptimes). We have also used multiple features of elastic search such as range, match, filters, etc for searching the document. The execution of the elastic search query is as follows:

The output of the first query is taken as input for the second query and is continued until we get the final output.

### 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

Query 1: POST [http://40.86.219.24:9200/stopdb/stops/\\_search](http://40.86.219.24:9200/stopdb/stops/_search)

The screenshot shows a REST client interface with a POST request to `http://40.86.219.24:9200/stopdb/stops/_search`. The request body is a JSON object:

```
1 {
2   "query": {
3     "match_phrase": {
4       "name_stop": "Ferry Stop - Alderney"
5     }
6   },
7   "_source": ["stop_id"]
8 }
9
```

The response status is 200 OK, with a time of 51 ms and a size of 312 B. The response body is a JSON object:

```
1 {
2   "took": 5,
3   "timed_out": false,
4   "_shards": {
5     "total": 5,
6     "successful": 5,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 1,
12    "max_score": 19.865513,
13    "hits": [
14      {
15        "_index": "stopdb",
16        "_type": "stops",
17        "_id": "MYwOeWMBjzk4o8qNUZHx",
18        "_score": 19.865513,
19        "_source": {
20          "stop_id": "1074"
21        }
22      }
23    ]
24  }
25}
```

Query 2: POST [http://40.86.219.24:9200/stoptimesdb/stoptimes/\\_search](http://40.86.219.24:9200/stoptimesdb/stoptimes/_search)

The screenshot shows a REST client interface with a POST request to `http://40.86.219.24:9200/stoptimesdb/stoptimes/_search`. The request body is a JSON object:

```
1 {
2   "query": {
3     "match_phrase": {
4       "stop_id": "1074"
5     }
6   },
7   "_source": ["trip_id"]
8 }
9
```

The response status is 200 OK, with a time of 77 ms and a size of 534 B. The response body is a JSON object:

```
{
  "took": 8,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 542,
    "max_score": 1.0,
    "hits": [
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "14wfeWMBjzk4o8qNc_Yu",
        "_score": 1.0,
        "_source": {
          "trip_id": "6219323-2012_08A-12AferSU-Sunday-01"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "AYwfeWMBjzk4o8qNc_0w",
        "_score": 1.0,
        "_source": {
          "trip_id": "5808147-2012_05M-12MferSU-Sunday-00"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "YY0feWMBjzk4o8qNc1hP",
        "_score": 1.0,
        "_source": {
          "trip_id": "6219296-2012_08A-12AferSA-Saturday-01"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "zI0feWMBjzk4o8qNc11X",
        "_score": 1.0,
        "_source": {
          "trip_id": "6302519-2012_05M-12MferHO-Sunday-00"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "To0feWMBjzk4o8qNc2VY",
        "_score": 1.0,
        "_source": {
          "trip_id": "6219359-2012_08A-12AferSU-Sunday-01"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "3I0feWMBjzk4o8qNc3Rb",
        "_score": 1.0,
        "_source": {
          "trip_id": "5807986-2012_05M-12MferWD-Weekday-00"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "m40feWMBjzk4o8qNc4Fd",
        "_score": 1.0,
        "_source": {
          "trip_id": "5808091-2012_05M-12MferSA-Saturday-00"
        }
      },
      {
        "_index": "stoptimesdb",
        "_type": "stoptimes",
        "_id": "o0feWMBjzk4o8qNc4If",
        "_score": 1.0,
        "_source": {
          "trip_id": "5808091-2012_05M-12MferSA-Saturday-00"
        }
      }
    ]
  }
}
```



## Query 3: POST http://40.86.219.24:9200/tripdb/trips/\_search

POST http://40.86.219.24:9200/tripdb/trips/\_search

Status: 200 OK Time: 84 ms Size: 465 B

Body (application/json)

```

1 {
2   "query": {
3     "match": {
4       "trip_id": "6219323-2012_08A-12AferSU-Sunday-01"
5     }
6   },
7   "_source": "trip_headsign"
8 }
9

```

Body (Pretty)

```

{"took":19,"timed_out":false,"_shards":{"total":5,"successful":5,"skipped":0,"failed":0},"hits":{"total":8956,"max_score":16.213545,"hits":[{"_index":"tripdb","type":"trips","id":"IowSewMBjk408qNmpvr","score":16.213545,"source":{"trip_headsign":"FERRY TO DARTMOUTH"}}, {"_index":"tripdb","type":"trips","id":"FYwSewMBjk408qNmzf2","score":8.774479,"source":{"trip_headsign":"FERRY TO HALIFAX"}}, {"_index":"tripdb","type":"trips","id":"tYwSewMBjk408qNmz8","score":8.774479,"source":{"trip_headsign":"FERRY TO DARTMOUTH"}}, {"_index":"tripdb","type":"trips","id":"ZIwSewMBjk408qNm78","score":8.774479,"source":{"trip_headsign":"FERRY TO HALIFAX"}}, {"_index":"tripdb","type":"trips","id":"fowSewMBjk408qNmst","score":8.774479,"source":{"trip_headsign":"FERRY TO HALIFAX"}}, {"_index":"tripdb","type":"trips","id":"N4wSewMBjk408qNmsf","score":8.774479,"source":{"trip_headsign":"FERRY TO DARTMOUTH"}}, {"_index":"tripdb","type":"trips","id":"tYwSewMBjk408qNm8oA","score":8.774479,"source":{"trip_headsign":"FERRY TO HALIFAX"}}, {"_index":"tripdb","type":"trips","id":"IIwSewMBjk408qNm9AB","score":8.774479,"source":{"trip_headsign":"FERRY TO HALIFAX"}}, {"_index":"tripdb","type":"trips","id":"fYwSewMBjk408qNmp7z","score":8.648615,"source":{"trip_headsign":"FERRY TO HALIFAX"}}, {"_index":"tripdb","type":"trips","id":"p4wSewMBjk408qNmRH5","score":8.648615,"source":{"trip_headsign":"FERRY TO DARTMOUTH"}}]}}

```

Ans:

1. Ferry to Dartmouth
2. Ferry to Halifax

Total time taken – 212 ms.

## b. Find buses between two range

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

## Query 1: POST http://40.86.219.24:9200/stoptimesdb/stoptimes/\_search

POST http://40.86.219.24:9200/stoptimesdb/stoptimes/\_search

Status: 200 OK Time: 61 ms Size: 512 B

Body (application/json)

```

1 {
2   "query": {
3     "range": {
4       "arrival_time": {
5         "gte": "20:57:00",
6         "lte": "21:00:00",
7         "format": "HH:mm:ss"
8       }
9     }
10  },
11  "_source": ["trip_id"]
12 }

```

Body (Pretty)

```

{"took":14,"timed_out":false,"_shards":{"total":5,"successful":5,"skipped":0,"failed":0},"hits":{"total":33505,"max_score":1.0,"hits":[{"_index":"stoptimesdb","type":"stoptimes","id":"jYwfeWMBjk408","trip_id":"6530005-2012_08A-1208BRsu-Sunday-01"}, {"_index":"stoptimesdb","type":"stoptimes","id":"vIwfeWMBjk408q","trip_id":"6525509-2012_08A-1208BRwd-Weekday-01"}, {"_index":"stoptimesdb","type":"stoptimes","id":"N4wfeWMBjk408","trip_id":"6527180-2012_08A-1208BRsa-Saturday-01"}, {"_index":"stoptimesdb","type":"stoptimes","id":"jYwfeWMBjk408","trip_id":"6512207-2012_05M-1205BRwd-Weekday-02"}, {"_index":"stoptimesdb","type":"stoptimes","id":"WowfeWMBjk408","trip_id":"6518455-2012_05M-1205BRsu-Sunday-02"}, {"_index":"stoptimesdb","type":"stoptimes","id":"Q4wfeWMBjk408","trip_id":"6517046-2012_05M-1205BRsa-Saturday-02"}, {"_index":"stoptimesdb","type":"stoptimes","id":"AowfeWMBjk408","trip_id":"6530325-2012_08A-1208BRsu-Sunday-01"}, {"_index":"stoptimesdb","type":"stoptimes","id":"z4wfeWMBjk408q"}]}}

```

## Query 2: POST http://40.86.219.24:9200/tripdb/trips/\_search

The screenshot shows a REST client interface with a POST request to `http://40.86.219.24:9200/tripdb/trips/_search`. The request body is a JSON object:

```

{
  "query": {
    "match": {
      "trip_id": "6530005-2012_08A-1208BRSu-Sunday-01"
    }
  },
  "_source": "trip_headsign"
}

```

The response status is 200 OK, with a time of 106 ms and a size of 561 B. The response body is a JSON object containing search results:

```

{
  "took": 49,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 8956,
    "max_score": 12.766067,
    "hits": [
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "0IwSewMBjzk4o8qNmprq",
        "_score": 12.766067,
        "_source": {
          "trip_headsign": "54 MONTEBELLO"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "a4wSewMBjzk4o8qNmpr",
        "_score": 5.2631664,
        "_source": {
          "trip_headsign": "68 PORTLAND ST TO CHERRYBROOK"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "b4wSewMBjzk4o8qNmpr",
        "_score": 5.2631664,
        "_source": {
          "trip_headsign": "60 EASTERN PASSAGE"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "gowSewMBjzk4o8qNmpr",
        "_score": 5.2631664,
        "_source": {
          "trip_headsign": "59 PORTLAND ST TO COLBY"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "v4wSewMBjzk4o8qNmpr",
        "_score": 5.2631664,
        "_source": {
          "trip_headsign": "14 LEIBLIN PARK"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "04wSewMBjzk4o8qNmpr",
        "_score": 5.2631664,
        "_source": {
          "trip_headsign": "66 PENHORN TO GASTON RD"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "UYwSewMBjzk4o8qNmpvr",
        "_score": 5.2631664,
        "_source": {
          "trip_headsign": "7 GOTTINGEN"
        }
      }
    ]
  }
}

```

Ans.

Total time taken – 61+106 = 167 ms

### 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.

Query 1: POST http://40.86.219.24:9200/tripdb/trips/\_search?q=route\_id:FerD-116

The screenshot shows a REST client interface with a POST request to `http://40.86.219.24:9200/tripdb/trips/_search?q=route_id:FerD-116`. The request body is a JSON object:

```

{
  "query": {
    "match_phrase": {
      "trip_headsign": "FERRY TO HALIFAX"
    }
  },
  "_source": ["trip_id"]
}

```

The response status is 200 OK, with a time of 101 ms and a size of 477 B. The response body is a JSON object containing search results for the route 'FERRY TO HALIFAX':

```

{
  "took": 42,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 570,
    "max_score": 7.479691,
    "hits": [
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "F4wSewMBjzk4o8qNmpr",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5807969-2012_05M-12MferWD-Weekday-00"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "L4wSewMBjzk4o8qNmpzt",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5808073-2012_05M-12MferSA-Saturday-00"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "hIwSewMBjzk4o8qNmp7z",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5807972-2012_05M-12MferWD-Weekday-00"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "nYwSewMBjzk4o8qNmp7z",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5808091-2012_05M-12MferSA-Saturday-00"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "P4wSewMBjzk4o8qNmqP0",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5808087-2012_05M-12MferSA-Saturday-00"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "CowSewMBjzk4o8qNmqT1",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5807997-2012_05M-12MferWD-Weekday-00"
        }
      },
      {
        "_index": "tripdb",
        "_type": "trips",
        "_id": "fIwSewMBjzk4o8qNmqf2",
        "_score": 7.479691,
        "_source": {
          "trip_id": "5808150-2012_05M-12MferSU-Sunday-00"
        }
      }
    ]
  }
}

```

## Query 2: POST http://40.86.219.24:9200/stopptimesdb/stoptimes/\_search

POST http://40.86.219.24:9200/stopptimesdb/stoptimes/\_search

Status: 200 OK Time: 76 ms Size: 337 B

Auth Headers (1) Body Pre-req. Tests Cookies Code

form-data x-www-form-urlencoded raw binary

JSON (application/json)

```

1 {
2   "query": {
3     "match_phrase": {
4       "trip_id": "5807969-2012_05M-12MferWD-Weekday-00"
5     }
6   }, "_source": ["stop_id"]
7 }

```

Pretty Raw Preview

```

{"took":7,"timed_out":false,"_shards":
{"total":5,"successful":5,"skipped":0,"failed":0},"hits":
{"total":2,"max_score":27.298138,"hits":
[{"_index":"stopptimesdb","_type":"stoptimes","_id":"8owfeWMBjzk4o8qNUZ",
{"stop_id":1073}},
{"_index":"stopptimesdb","_type":"stoptimes","_id":"84wfeWMBjzk4o8qNUZ",
{"stop_id":1074}}]}

```

## Query 3: POST http://40.86.219.24:9200/stopdb/stops/\_search

POST http://40.86.219.24:9200/stopdb/stops/\_search

Status: 200 OK Time: 211 ms Size: 344 B

Auth Headers (1) Body Pre-req. Tests Cookies Code

form-data x-www-form-urlencoded raw binary

JSON (application/json)

```

1 {
2   "query": {
3     "range": {
4       "stop_id": {
5         "gte": "1073",
6         "lte": "1074"
7       }
8     }, "_source": ["name_stop"]
9   }
10 }

```

Pretty Raw Preview

```

{"took":3,"timed_out":false,"_shards":
{"total":5,"successful":5,"skipped":0,"failed":0},"hits":
{"total":2,"max_score":1.0,"hits":
[{"_index":"stopdb","_type":"stops","_id":"MYwOeWMBjzk4o8qNUZ",
{"name_stop":"Ferry Stop - Alderney"}},
{"_index":"stopdb","_type":"stops","_id":"MlIwOeWMBjzk4o8qNUZ",
{"name_stop":"Ferry Stop - Halifax"}]}]}

```

Ans.

Stops for route id = FerD-116 and bus name = Ferry to Halifax are:

1. Ferry Stop – Alderney
2. Ferry Stop – Halifax

Total time taken – 101+76+211 = 388 ms

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

Query 1: POST [http://40.86.219.24:9200/stoptimesdb/stoptimes/\\_search](http://40.86.219.24:9200/stoptimesdb/stoptimes/_search)

POST [http://40.86.219.24:9200/stoptimesdb/stoptimes/\\_search](http://40.86.219.24:9200/stoptimesdb/stoptimes/_search) Params Send Save Status: 200 OK Time: 80 ms Size: 343 B

Auth Headers (1) Body Pre-req. Tests Cookies Code Body Cookies Headers (3) Test Results

form-data x-www-form-urlencoded raw binary JSON (application/json)

```

1 {
2   "size": 0,
3   "aggs": {
4     "counts_stop_id": {
5       "terms": {
6         "field": "stop_id",
7         "order": { "_count": "desc" },
8         "size": 3
9       }
10    }
11  }
12 }

```

```

{"took":28,"timed_out":false,"_shards":
{"total":5,"successful":5,"skipped":0,"failed":0},"hits":
{"total":559186,"max_score":0.0,"hits":[]},"aggregations":
{"counts_stop_id":
{"doc_count_error_upper_bound":1476,"sum_other_doc_count":551868,
[{"key":8643,"doc_count":2594},{key":6105,"doc_count":2525},
{"key":6108,"doc_count":2199}]}}

```

Query 2: POST [http://40.86.219.24:9200/stopdb/stops/\\_search](http://40.86.219.24:9200/stopdb/stops/_search)

POST [http://40.86.219.24:9200/stopdb/stops/\\_search](http://40.86.219.24:9200/stopdb/stops/_search) Params Send Save Status: 200 OK Time: 59 ms Size: 345 B

Auth Headers (1) Body Pre-req. Tests Cookies Code Body Cookies Headers (3) Test Results

form-data x-www-form-urlencoded raw binary JSON (application/json)

```

1 {
2   "query": {
3     "match": {
4       "stop_id": "8643"
5     }
6   }, "_source": ["name_stop"]
7 }

```

```

{"took":6,"timed_out":false,"_shards":
{"total":5,"successful":5,"skipped":0,"failed":0},"hits":
{"total":1,"max_score":5.7807436,"hits":
[{"_index":"stopdb","_type":"stops","_id":"aIwOeWMBjzk4o8qNUZn
{"name_stop":"mumford Terminal [outbound In Terminal]}"}]}

```

Query 3: POST [http://40.86.219.24:9200/stopdb/stops/\\_search](http://40.86.219.24:9200/stopdb/stops/_search)

POST [http://40.86.219.24:9200/stopdb/stops/\\_search](http://40.86.219.24:9200/stopdb/stops/_search) Params Send Save Status: 200 OK Time: 59 ms Size: 345 B

Auth Headers (1) Body Pre-req. Tests Cookies Code Body Cookies Headers (3) Test Results

form-data x-www-form-urlencoded raw binary JSON (application/json)

```

1 {
2   "query": {
3     "match": {
4       "stop_id": "6105"
5     }
6   }, "_source": ["name_stop"]
7 }

```

```

{"took":3,"timed_out":false,"_shards":
{"total":5,"successful":5,"skipped":0,"failed":0},"hits":
{"total":1,"max_score":5.786897,"hits":
[{"_index":"stopdb","_type":"stops","_id":"mowOeWMBjzk4o8qNUZ
{"name_stop":"barrington St [southbound] before Duke St"}]}

```

## Query 4: POST http://40.86.219.24:9200/stopdb/stops/\_search

The screenshot shows a REST client interface with a POST request to `http://40.86.219.24:9200/stopdb/stops/_search`. The request body is in JSON format, and the response is also in JSON format. The response status is 200 OK, with a time of 57 ms and a size of 346 B.

**Request Body (JSON):**

```

1 {
2   "query" : {
3     "match" : {
4       "stop_id": "6108"
5     }
6   }, "_source": ["name_stop"]
7 }

```

**Response Body (JSON):**

```

{"took":4,"timed_out":false,"_shards":
{"total":5,"successful":5,"skipped":0,"failed":0},"hits":
{"total":1,"max_score":5.710427,"hits":
[{"_index":"stopdb","_type":"stops","_id":"nYwOeWMBjzk4o8qNUZl
{"name_stop":"barrington St [southbound] before George St"}]}}

```

Ans.

1. mumford Terminal [outbound In Terminal]
  2. barrington St [southbound] before Duke St
  3. barrington St [southbound] before George St
- Total time taken – 80+59+60+57 = 256 ms.

## 6. Test results and summary:

As we observed two different databases with same datasets, it is a good idea to compare their performance with few metrics as given below.

### 6.1. Data extraction and manipulation:

While extracting the data, we can see that there is not a significant difference between the time taken by both the databases except for question “d”.



Figure: 6.1. SQL v/s Elastic Search

From the above figure, one can easily understand that we have Time (in milliseconds) on the Y-axis and all the questions are on the X-axis with blue colored bars representing SQL and green ones representing Elastic.

It is also a good idea to discuss another metric i.e., the time taken for manipulation. The data for MYSQL was inserted remotely from MYSQL workbench. For Elastic Search data was first converted into JSON format and then using REST client(POSTMAN) data was imported. Elastic Search took around twenty minutes to import the data while MYSQL took around thirty minutes to import it. But it is worth noting that both the databases are installed on different clouds with different hardware configurations. We have emphasized on this point later in the next part.

## 6.2. Cloud computing power:

MYSQL is based on RDBMS while Elastic Search is based on NoSQL database. The environment setup for MYSQL as well as for elastic search was easy. But performance-wise, the server was getting freeze due to the shortage of memory (RAM) on Amazon Web Service. While MYSQL was performing well on the same server. Due to this memory issue, we used another cloud service provider Azure to get better performance for elastic search. Elastic Search was then performing better at Azure. For the record, Amazon Web has one GB of RAM allocated per instance, while Azure has four GB. Hence, the Elastic search is not lightweight in terms of memory usage as compared to MYSQL.

## 7. Conclusion:

Based on the points emphasized in section 5. of this report we conclude that given the dataset, it is a fair choice of using traditional database over NoSQL database (Elastic Search). Even though non-traditional databases may be good at handling large data, but at the same time, it demands more computing resources. Given the resources provided (as we shifted to Microsoft Azure), we did not observe any strong improvement in data extraction process.