

Data Management, Warehousing, and Analytics Assignment 1

TEAM MEMBERS:

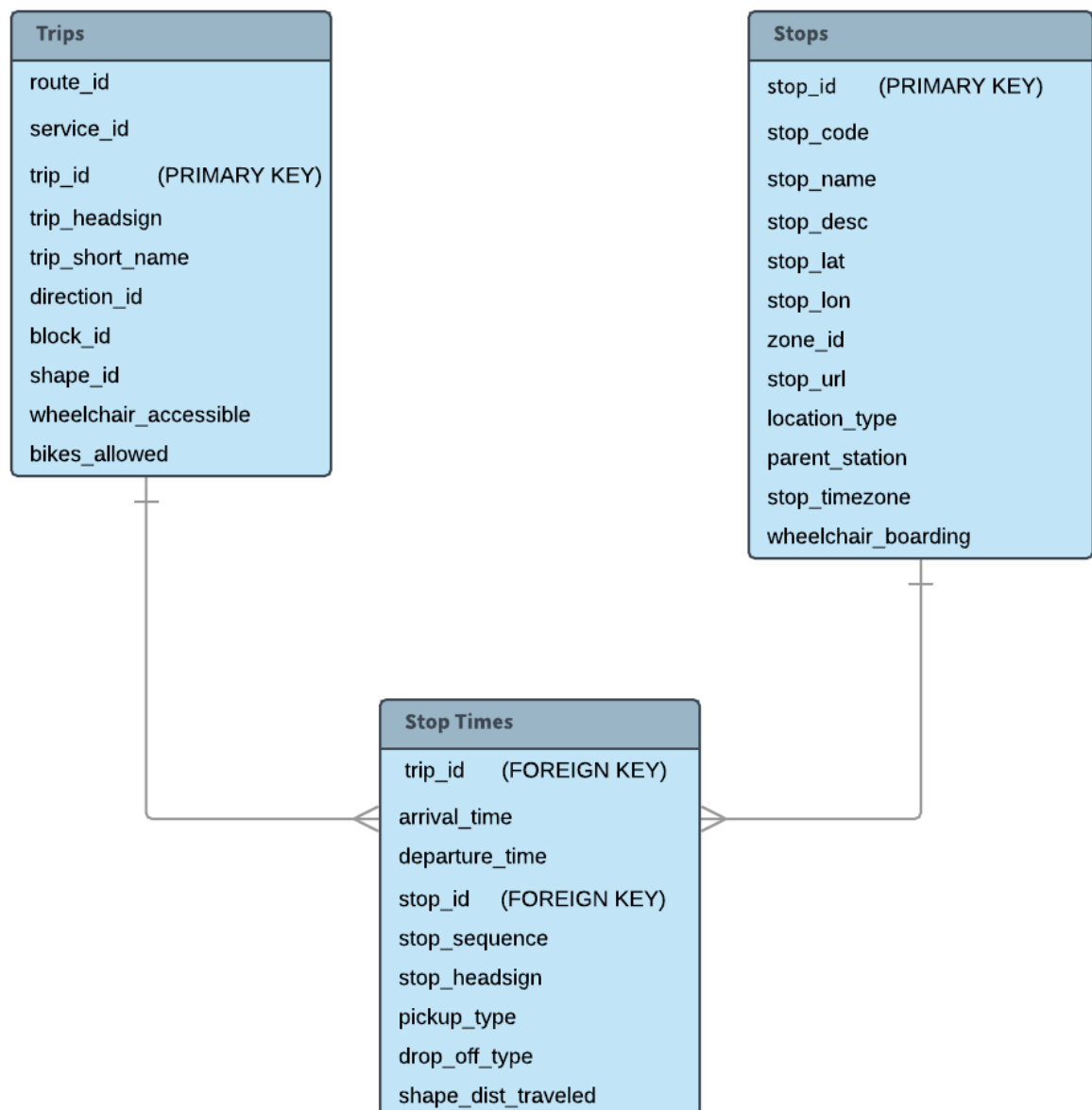
Aravind Govindarajan (B00803336)

Haritha Desikan (B00801722)

TASK DESCRIPTION

To get ourselves acquainted with a cloud computing platform and database concepts by running queries for a Transit dataset in local RDBMS (MySQL), IBM cloud RDBMS and performing Elasticsearch on Cloud using the Postman software and then compare the efficiency of the three methods with the execution time taken for each query.

RELATIONAL DATABASE DESIGN (ENTITY RELATIONSHIP DIAGRAM)



QUERIES

1. We are asked to identify all routes and buses that goes via a specific stop from the Transit database in this question. For this, we have given a stop id as input with the where clause and obtained trip id, trip head sign along with the stop id. We have made use of joins to combine trips and stop_times tables. The Response time taken for this query to run in local RDBMS and cloud are 172 millisecs and 53 millisecs.

USING LOCAL RDBMS

The screenshot shows a local RDBMS interface with a query editor and a result grid. The query is:

```
select trips.trip_id, trips.trip_headsign, stop_times.stop_id from trips join stop_times
on trips.trip_id=stop_times.trip_id where stop_times.stop_id= 7605;
```

The result grid displays the following data:

trip_id	trip_headsign	stop_id
17182630	1 SPRING GARDEN TO MUMFORD TERM	7605
17182635	1 SPRING GARDEN TO MUMFORD TERM	7605
17182641	1 SPRING GARDEN TO MUMFORD TERM	7605
17182647	1 SPRING GARDEN TO MUMFORD TERM	7605
17182653	1 SPRING GARDEN TO MUMFORD TERM	7605
17182632	1 SPRING GARDEN TO MUMFORD TERM	7605
17182638	1 SPRING GARDEN TO MUMFORD TERM	7605
17182644	1 SPRING GARDEN TO MUMFORD TERM	7605
17182650	1 SPRING GARDEN TO MUMFORD TERM	7605
17182656	1 SPRING GARDEN TO MUMFORD TERM	7605
17182707	1 SPRING GARDEN TO MUMFORD TERM	7605
17182636	1 SPRING GARDEN TO MUMFORD TERM	7605
17182642	1 SPRING GARDEN TO MUMFORD TERM	7605
17182648	1 SPRING GARDEN TO MUMFORD TERM	7605

The output pane shows the execution details:

#	Time	Action	Message	Duration / Fetch
89	16:41:15	select trips.route_id, count(*) from trips join stop_times on trips.trip_id = stop_times.trip_id and stop_times departu...	0 row(s) returned	0.265 sec / 0.000 sec
90	16:41:45	select trips.route_id, count(*) from trips join stop_times on trips.trip_id = stop_times.trip_id and stop_times departu...	0 row(s) returned	0.312 sec / 0.000 sec
91	16:49:14	select trips.trip_id, trips.trip_headsign, stop_times.stop_id from trips join stop_times on trips.trip_id=stop_times.tri...	495 row(s) returned	0.172 sec / 0.156 sec

USING CLOUD RDBMS (OUTPUT SCREEN)

The screenshot shows a cloud RDBMS interface with a query editor and a result set. The query is:

```
select trips.trip_id, trips.trip_headsign, stop_times.stop_id from trips join stop_times
on trips.trip_id=stop_times.trip_id where stop_times.stop_id= 7605 ;
```

The result set displays the following data:

TRIP_ID	TRIP_HEADSIGN	STOP_ID
17174781	1 SPRING GARDEN TO MUMFORD TERM	7605
17174855	1 SPRING GARDEN TO MUMFORD TERM	7605
17174893	1 SPRING GARDEN TO MUMFORD TERM	7605
17174744	1 SPRING GARDEN TO MUMFORD TERM	7605
17174890	1 SPRING GARDEN TO MUMFORD TERM	7605

Total rows: 495

USING CLOUD RDBMS (RESPONSE TIME)

The screenshot displays a cloud RDBMS interface. At the top, a SQL query is entered in a text area:

```
18 select trips.trip_id, trips.trip_headsign, stop_times.stop_id from trips join stop_times
19 on trips.trip_id=stop_times.trip_id where stop_times.stop_id= 7605 ;
```

Below the query editor, the interface is divided into two main sections: "Saved scripts" and "Result".

The "Saved scripts" section on the left shows a list of scripts. The first script, "select trips.trip_id, trips.trip_headsign, stop_times.stop_id from trips join stop_times ...", is highlighted with a green checkmark and the status "All(1), Failed(0)".

The "Result" section on the right shows the execution status. A message box states: "The statement ran successfully." Below this, the "SQL statement" is repeated:

```
select trips.trip_id, trips.trip_headsign, stop_times.stop_id from trips join s
top_times
on trips.trip_id=stop_times.trip_id where stop_times.stop_id= 7605
```

To the right of the SQL statement, the "Execution log" shows the following details:

- Run time: 0.053 s
- Status: SUCCEEDED
- Database: BLUDB

2. We are asked to find the total number of trips on each route in this question. We have used trips table in the query and obtained the count of total trips in each route. Response time taken to run this query in local RDBMS and cloud RDBMS are 160 millisecs and 18 millisecs.

USING LOCAL RDBMS

The screenshot shows a local RDBMS interface. The SQL query entered is:

```
select route_id, count(trip_id) From trips group by route_id;
```

The "Result Grid" displays the following data:

route_id	count(trip_id)
1	495
10	249
11	7
123	19
135	14
136	16
137	12
138	14
14	147
15	82
159	80
185	87
194	8
2	332

The "Output" section at the bottom shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	15:27:19	select route_id, count(trip_id) From trips group by route_id LIMIT 0, 1000	67 row(s) returned	0.016 sec / 0.000 sec

USING CLOUD RDBMS (OUTPUT SCREEN)

```
select route_id, count(trip_id) from trips group by route_id;
```

red scripts

Result

Filter by status: All Delete All

1(1), Failed(0)

select route_id, count(trip_id) from trips group by route_id

Result set

Log

ROUTE_ID	
1	495
10	249
11	7
123	19
135	14

Total rows: 67

USING CLOUD RDBMS (RESPONSE TIME)

```
93 select route_id, count(trip_id) from trips group by route_id;
94
95
96
97
98
99
100
```

Saved scripts

Result

Filter by status: All Delete All

1(1), Failed(0)

select route_id, count(trip_id) from trips group by route_id

Result set

Log

The statement ran successfully.

SQL statement

select route_id, count(trip_id) from trips group by route_id

Execution log

Run time: 0.018 s

Status: SUCCEEDED

Database: BLUDB

- We are asked to find the most and least busy bus stops in a day in this question. For this, we sort the stop names in descending order of the count of trips. We use join in this query to combine stops and stop_times tables. Response time for executing this query in local RDBMS and cloud RDBMS are 853.2 millisecs and 63 millisecs.

USING LOCAL RDBMS

Query 1

128
129
130
131
132
133
134
135

```
select stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id
group by stops.stop_name order by count(trip_id) desc;
```

Result Grid

stop_name

Barrington St Before Duke St Bay 2 (6107)
Barrington St After Duke St Bay 1 (6087)
Barrington St After Cornwallis St (6125)
Barrington St Before Sackville St (6121)
Barrington St Before Duke St (6106)
Barrington St Before Prince St (6084)
Barrington St Before Prince St (6083)
Barrington St After Prince St (6103)
Barrington St Before Blowers St (6102)
Mumford Terminal Bay 3 (8641)
Mumford Terminal Bay 2 (8640)
Mumford Rd After Chebucto Rd (7275)
Mumford Rd Opposite St Agnes School (72...
Mumford Rd Before Leppert St (7274)
North St Before Brunswick St (7351)
Spring Garden Rd Before Dresden Row (8...
Spring Garden Rd Before Barrington St (...
Barrington St Before Spring Garden Rd (...
Spring Garden Rd After Dresden Row (8336)
Barrington St Before Cornwallis St (6104)

Output

Action Output

#	Time	Action	Message	Duration / Fetch
79	15:43:41	select stops.stop_names from stops join stop_times on stops.stop_id = stop_times.stop_id group by stops.stop_...	Error Code: 1054. Unknown column 'stops.stop_names' in 'field list'	0.000 sec
80	15:44:14	select stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id group by stops.stop_n...	2374 row(s) returned	8.532 sec / 0.000 sec

USING CLOUD RDBMS (OUTPUT SCREEN)

95
96
97
98
99
100

```
select stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id group by stops.stop_name order by count(trip_id) desc;
```

Saved scripts

Result

Filter by status: All

Delete All

✓ All(1), Failed(0)

select stops.stop_name from stops join stop_times on stops.stop_id = stop_times.st...

✓ All(1), Failed(1)

select stops.stop_names from stops join stop_times on stops.stop_id = stop_times.st...

✓ All(1), Failed(0)

select route_id, count(trip_id) from trips group by route_id

Result set

Log

STOP_NAME

Barrington St Before Duke St Bay 2 (6107)
Barrington St After Duke St Bay 1 (6087)
Barrington St After Cornwallis St (6125)
Barrington St Before Duke St (6106)
Barrington St Before Sackville St (6121)

Total rows: 2374

USING CLOUD RDBMS (RESPONSE TIME)

```
95
96 select stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id group by stops.stop_name order by count(trip_id) desc;
97
98
99
100
```

Saved scripts **Result**

Filter by status: All Delete All

▼ All(1), Failed(0) 🗑️

✓ select stops.stop_name from stops join stop_times on stops.stop_id = stop_times.st...

▼ All(1), Failed(1) 🗑️

✗ select stops.stop_names from stops join stop_times on stops.stop_id = stop_times.st...

▼ All(1), Failed(0) 🗑️

✓ select route_id, count(trip_id) from trips group by route_id

Result set **Log**

The statement ran successfully.

SQL statement

```
select stops.stop_name from stops join stop_times on stops.stop_id = stop_time
s.stop_id group by stops.stop_name order by count(trip_id) desc
```

Execution log

Run time: 0.063 s

Status: SUCCEEDED

Database: BLUDB

4. We must find the frequency of a bus service over a time range in this question. For this purpose, we give a route identifier and time range as input and get the number of times a bus route service is available in that time range. The query is written with a join combining trips and stop_times tables. Response time for running this query in local RDBMS and cloud RDBMS are 31 millisecs and 26 millisecs.

USING LOCAL RDBMS

```
209 • select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on
210 trips.trip_id = stop_times.trip_id where stop_times.departure_time
211 between '5:45:00' and '6:45:00' group by trips.route_id;
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

route_id	frequency
1	580
10	345
14	247
2	360
21	360
22	189
29	378
3	466
30A	128
30B	134
320	85
39	241
4	333
51	128
53	116

Result 73 x

Read Only Context Help Snippets

Output

#	Time	Action	Message	Duration / Fetch
81	14:18:46	select trips.route_id, distinct stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id j...	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL ser...	0.016 sec
82	14:19:08	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.078 sec / 0.000 sec
83	14:19:52	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.047 sec / 0.000 sec
84	14:20:00	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.047 sec / 0.000 sec
85	14:22:34	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	105 row(s) returned	0.047 sec / 0.000 sec
86	14:42:10	select trips.route_id , count(*) as frequency from trips join stop_times on trips.trip_id = stop_times.trip_id where st...	63 row(s) returned	0.266 sec / 0.000 sec
87	14:50:32	select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on trips.trip_id = stop_times.trip_j...	63 row(s) returned	0.187 sec / 0.000 sec

USING CLOUD RDBMS (OUTPUT SCREEN)

112
113
114 select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on trips.trip_id = stop_times.trip_id where stop_times.departure_time between '5:45:00' and '6:45:00' group by trips.route_id;
115
116

Saved scripts

Result

Filter by status: All Delete All

All(1), Failed(0)

select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on...

All(1), Failed(0)

select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on ...

All(1), Failed(0)

select trips.route_id , count(*) as frequency from trips join stop_times on trips.trip_i...

All(1), Failed(0)

select trips.route_id , count(*) from trips join stop_times on trips.trip_id = stop_time...

Result set

Log

ROUTE_ID	FREQUENCY
1	580
10	345
11	10
123	60
135	26

Total rows: 63

USING CLOUD RDBMS (RESPONSE TIME)

1
2 select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on trips.trip_id = stop_times.trip_id where stop_times.departure_time between '5:45:00' and '6:45:00' group by trips.route_id;
3

ved scripts

Result

er by status: All Delete All

ll(1), Failed(0)

select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on...

ll(1), Failed(0)

select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on ...

ll(1), Failed(0)

select trips.route_id , count(*) as frequency from trips join stop_times on trips.trip_i...

ll(1), Failed(0)

Result set

Log

The statement ran successfully.

SQL statement

select trips.route_id , count(trips.trip_id) as frequency from trips join stop_ times on trips.trip_id = stop_times.trip_id where stop_times.departure_time bet ween '5:45:00' and '6:45:00' group by trips.route_id

Execution log

Run time: 0.026 s

Status: SUCCEEDED

Database: BLUDB

5. We are to find the actual route of a bus service in this question. For this purpose, we get route identifier as input dynamically and display stop names along with their route identifier with the help of joins to combine stops with stop_times table and another join to combine trips and stop_times. Response time for running this query in local RDBMS and cloud RDBMS are 47 milliseconds and 54 milliseconds.

USING LOCAL RDBMS

The screenshot shows a database IDE interface. At the top, a SQL query is entered in a text editor:

```
select distinct route_id, stops.stop_name from stops
join stop_times on stops.stop_id = stop_times.stop_id join trips on trips.trip_id = stop_times.trip_id
where trips.route_id=10;
```

Below the query editor, the "Result Grid" displays the results of the query. The results are as follows:

route_id	stop_name
10	Raymoor Dr Before Main St (8162)
10	Weyburn Rd After Main St (8483)
10	Weyburn Rd Before Athabaskan Ln (8484)
10	Weyburn Rd Before Spikenard St (8482)
10	Spikenard St Before Valleyfield Rd (8319)
10	Hartlen St After Tacoma Dr (6835)
10	Main St Before Major St (7175)
10	Micmac Blvd After Brookdale Cr (7209)
10	Micmac Terminal Bay 2 (8172)
10	Micmac Blvd Before Horizon Ct (7211)
10	Micmac Blvd Before Woodland Ave (7215)
10	Woodland Ave After Laurier St (8581)
10	Woodland Ave After Pine Hill Rd (8583)

Below the result grid, the "Output" pane shows the execution log. The log entries are as follows:

#	Time	Action	Message	Duration / Fetch
79	13:52:18	select trips.route_id, min(stop_times.departure_time) as firsttrip, max(stop_times.departure_time) as lasttrip from ...	1 row(s) returned	0.141 sec / 0.016 sec
80	14:17:43	select trips.route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join trips ...	18000 row(s) returned	0.047 sec / 0.063 sec
81	14:18:46	select trips.route_id, distinct stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join ...	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL ser...	0.016 sec
82	14:19:08	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.078 sec / 0.000 sec
83	14:19:52	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.047 sec / 0.000 sec
84	14:20:00	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.047 sec / 0.000 sec
85	14:22:34	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	105 row(s) returned	0.047 sec / 0.000 sec

USING CLOUD RDBMS (OUTPUT SCREEN)

107 select distinct route_id, stops.stop_name from stops
108 join stop_times on stops.stop_id = stop_times.stop_id join trips on trips.trip_id = stop_times.trip_id
109 where trips.route_id='10';
110

Saved scripts

Result

Filter by status: All Delete All

All(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

All(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

All(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

All(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

Result set

Log

ROUTE_ID	STOP_NAME
10	Barrington St After Artz St (6088)
10	Barrington St After Cornwallis St (6125)
10	Barrington St After Duke St Bay 1 (6087)
10	Barrington St After Prince St (6103)
10	Barrington St Before Blowers St (6102)

Total rows: 105

USING CLOUD RDBMS (RESPONSE TIME)

select distinct route_id, stops.stop_name from stops
join stop_times on stops.stop_id = stop_times.stop_id join trips on trips.trip_id = stop_times.trip_id
where trips.route_id='10';

red scripts

Result

Filter by status: All Delete All

l(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

l(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

l(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

l(1), Failed(0)

select distinct route_id, stops.stop_name from stops join stop_times on stops.sto...

Result set

Log

The statement ran successfully.

SQL statement

select distinct route_id, stops.stop_name from stops
join stop_times on stops.stop_id = stop_times.stop_id join trips on trips.trip_id = stop_times.trip_id
where trips.route_id='10'

Execution log

Run time: 0.054 s

Status: SUCCEEDED

Database: BLUDB

6. In this question, we must get the first and last trip for a particular route. By giving route identifier as input, we run a query to find the minimum and maximum of departure times for that route. We have combined stop_times and trips tables using join. Response time for this query to execute in local RDBMS and cloud RDBMS are 2500 millisecs and 33 millisecs.

USING LOCAL RDBMS

217 `select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times.departure_time) as`

218 `lasttrip from stop_times join trips on trips.trip_id = stop_times.trip_id group by trips.route_id;`

Result Grid

	route_id	firsttrip	lasttrip
1	05:49:00	25:11:00	
10	05:30:00	25:28:00	
11	06:08:00	16:33:00	
123	06:00:00	19:02:00	
135	06:25:00	19:00:00	
136	06:14:00	18:50:00	
137	06:20:00	18:39:00	
138	06:04:00	19:05:00	
14	05:46:00	25:00:00	
15	06:08:00	20:31:00	
159	06:00:00	19:20:00	
185	06:00:00	22:24:00	
194	06:36:00	17:50:00	
2	05:25:00	25:01:00	
21	05:26:00	24:21:00	

Output

#	Time	Action	Message	Duration / Fetch
82	14:19:08	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.078 sec / 0.000 sec
83	14:19:52	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.047 sec / 0.000 sec
84	14:20:00	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	72 row(s) returned	0.047 sec / 0.000 sec
85	14:22:34	select distinct route_id, stops.stop_name from stops join stop_times on stops.stop_id = stop_times.stop_id join tr...	105 row(s) returned	0.047 sec / 0.000 sec
86	14:42:10	select trips.route_id , count(*) as frequency from trips join stop_times on trips.trip_id = stop_times.trip_id where st...	63 row(s) returned	0.266 sec / 0.000 sec
87	14:50:32	select trips.route_id , count(trips.trip_id) as frequency from trips join stop_times on trips.trip_id = stop_times.trip_id	63 row(s) returned	0.187 sec / 0.000 sec
88	00:17:49	select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times.departure_time) as lasttrip from ...	67 row(s) returned	2.500 sec / 0.000 sec

USING CLOUD RDBMS (OUTPUT SCREEN)

97 `select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times.departure_time) as`

98 `lasttrip from stop_times join trips on trips.trip_id = stop_times.trip_id group by trips.route_id;`

100

Result

Filter by status: All

Delete All

All(1), Failed(0)

select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times...

All(1), Failed(1)

select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times.d...

All(1), Failed(0)

select stops.stop_name from stops join stop_times on stops.stop_id = stop_times...

All(1), Failed(1)

select stops.stop_names from stops join stop_times on stops.stop_id = stop_times...

ROUTE_ID

FIRSTTRIP

LASTTRIP

1

10:00:00

9:59:00

10

10:00:00

9:59:00

11

15:10:00

7:28:00

123

15:20:00

9:29:00

135

15:15:00

8:51:00

Total rows: 67

USING CLOUD RDBMS (RESPONSE TIME)

98 `select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times.departure_time) as`

99 `lasttrip from stop_times join trips on trips.trip_id = stop_times.trip_id group by trips.route_id;`

100

Result

Filter by status: All

Delete All

All(1), Failed(0)

select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times...

All(1), Failed(1)

select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_times.d...

All(1), Failed(0)

select stops.stop_name from stops join stop_times on stops.stop_id = stop_times...

All(1), Failed(1)

select stops.stop_names from stops join stop_times on stops.stop_id = stop_times...

The statement ran successfully.

SQL statement

select trips.route_id, min(stop_times.departure_time) as firsttrip , max(stop_t
imes.departure_time) as
lasttrip from stop_times join trips on trips.trip_id = stop_times.trip_id group
by trips.route_id

Execution log

Run time: 0.033 s

Status: SUCCEEDED

Database: BLUDB

CARRYING OUT QUERIES IN ELASTICSEARCH

We use Elasticsearch to run the same set of queries in NOSQL. Elastic search is a search engine which makes use of indices to search for a specific pattern in the dataset. Below are screenshots of queries taken with Elasticsearch.

LOADING BULK DATA INTO ELASTICSEARCH

[illegible]

1. We are to identify routes and busses that pass through a specific stop from the Transit database in this question.

https://f https://f https://f https://f https://portal- https:// https://f https://f https://portal- https://f + ... No Environment

POST https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search Params Send Save

```
1 {
2   "query": {
3     "match_phrase": {
4       "stop_id": "1073"
5     }
6   },
7   "_source": "trip_id"
8 }
```

Body Cookies (1) Headers (3) Test Results Status: 200 OK Time: 495 ms Size: 5

Pretty Raw Preview JSON

```
1 {
2   "took": 22,
3   "timed_out": false,
4   "_shards": {
5     "total": 3,
6     "successful": 3,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 711,
12    "max_score": 6.868556,
13    "hits": [
14      {
15        "_index": "bus",
16        "_type": "stops",
17        "_id": "FuvFBGYBkzuw-Siejm9q",
18        "_score": 6.868556,
19        "_source": {
20          "trip_id": "5808164-2012_05M-12MferSU-Sunday-00"
21        }
22      },
23    ]
24  }
25 }
```

https://f https://f https://f https://f https://portal- https://f https://f https://f https://portal- https://f + ... No Environment

POST https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search Params Send Save

```
1 {
2   "query": {
3     "match_phrase": {
4       "trip_id": "5808164-2012_05M-12MferSU-Sunday-00"
5     }
6   },
7   "_source": "trip_headsign"
8 }
```

Body Cookies (1) Headers (3) Test Results Status: 200 OK Time: 572 ms Size: 395 B

Pretty Raw Preview JSON

```
1 {
2   "took": 14,
3   "timed_out": false,
4   "_shards": {
5     "total": 3,
6     "successful": 3,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 5,
12    "max_score": 28.340418,
13    "hits": [
14      {
15        "_index": "bus",
16        "_type": "stops",
17        "_id": "kuvEBGYBkzuw-SietS9o",
18        "_score": 28.340418,
19        "_source": {
20          "trip_headsign": "FERRY TO HALIFAX"
21        }
22      },
23    ]
24  }
25 }
```

Build Browse ?

2. We are to find the first and last trip for a specific route here.

The screenshot shows a REST client interface with a POST request to `https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search`. The request body is a JSON object:

```
{  "query": {    "range": {      "arrival_time": {        "gte": "05:00:00",        "lte": "06:40:00",        "format": "HH:MM:SS"      }    }  },  "_source": "trip_id"}
```

The response status is 200 OK, with a time of 356 ms and a size of 467 B. The response body is a JSON array of two objects:

```
[  {    "_index": "bus",    "_type": "stops",    "_id": "q105CmYBzFUEPa9BZ3Xn",    "_score": 1,    "_source": {      "trip_id": "6512781-2012_05M-12058Rwd-Weekday-02"    }  },  {    "_index": "bus",    "_type": "stops",    "_id": "N105CmYBzFUEPa9BZ3bo",    "_score": 1,    "_source": {      "trip_id": "6522060-2012_08A-12088Rwd-Weekday-01"    }  }]
```

The screenshot shows a REST client interface with a POST request to `https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search`. The request body is a JSON object:

```
{  "query": {    "match_phrase": {      "trip_id": "6522060-2012_08A-12088Rwd-Weekday-01"    }  },  "_source": "route_id"}
```

The response status is 200 OK, with a time of 335 ms and a size of 369 B. The response body is a JSON object:

```
{  "hits": {    "total": 39,    "max_score": 11.017594,    "hits": [      {        "_index": "bus",        "_type": "stops",        "_id": "1104CmYBzFUEPa9BUTx6",        "_score": 11.017594,        "_source": {          "route_id": "87-114"        }      },      {        "_index": "bus",        "_type": "stops",        "_id": "N105CmYBzFUEPa9BZ3bo"      }    ]  } }
```

https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... + ... No Environment

POST https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search Params Send Save

Authorization Headers (2) Body Pre-request Script Tests Cookies Code

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

```
1 {
2   "query": {
3     "match_phrase": {
4       "route_id": "87-114"
5     }
6   }
7 }
```

Body Cookies (1) Headers (3) Test Results Status: 200 OK Time: 114 ms Size: 673 B

Pretty Raw Preview JSON

```
1 {
2   "took": 9,
3   "timed_out": false,
4   "_shards": {
5     "total": 3,
6     "successful": 3,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 150,
12    "max_score": 4.6413827,
13    "hits": [
14      {
15        "_index": "bus",
16        "_type": "stops",
17        "_id": "1FOUCwYBzFUEPa9BUU87",
18        "score": 4.6413827,
```

3. We are to find the actual route of a bus service.

https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... + ... No Environment

POST https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search Params Send Save

Authorization Headers (2) Body Pre-request Script Tests Cookies Code

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

```
1 {
2   "query": {
3     "match_phrase": {
4       "route_id": "80-121"
5     }
6   },
7   "_source": "trip_id"
8 }
```

Body Cookies (1) Headers (3) Test Results Status: 200 OK Time: 385 ms Size: 474 B

Pretty Raw Preview JSON

```
12 "max_score": 4.2674046,
13 "hits": [
14   {
15     "_index": "bus",
16     "_type": "stops",
17     "_id": "0104CmYBzFUEPa9BUT16",
18     "score": 4.2674046,
19     "_source": {
20       "trip_id": "6513451-2012_05M-1205BRwd-Weekday-02"
21     }
22   },
23   {
24     "_index": "bus",
25     "_type": "stops",
26     "_id": "UV04CmYBzFUEPa9BUT56",
27     "score": 4.2674046,
28     "_source": {
29       "trip_id": "6513412-2012_05M-1205BRwd-Weekday-02"
```

https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search

POST

Authorization Headers (2) Body Pre-request Script Tests

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

```

1 {
2   "query": {
3     "match_phrase": {
4       "trip_id": "6513451-2012_05M-12058Rwd-Weekday-02"
5     }
6   },
7   "_source": "stop_id"
8 }

```

Body Cookies (1) Headers (3) Test Results

Status: 200 OK Time: 395 ms Size: 390 B

Pretty Raw Preview JSON

```

5     "total": 3,
6     "successful": 3,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 87,
12    "max_score": 10.126931,
13    "hits": [
14      {
15        "_index": "bus",
16        "_type": "stops",
17        "_id": "RF05CmYBzfUEPa98Z4_w",
18        "_score": 10.126931,
19        "_source": {
20          "stop_id": 7309
21        }
22      }
23    ]
24  }

```

https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search

POST

Authorization Headers (2) Body Pre-request Script Tests

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

```

1 {
2   "query": {
3     "match_phrase": {
4       "stop_id": "7309"
5     }
6   },
7   "_source": "name_stop"
8 }

```

Body Cookies (1) Headers (3) Test Results

Status: 200 OK Time: 428 ms Size: 440 B

Pretty Raw Preview JSON

```

27   },
28   {
29     "_index": "bus",
30     "_type": "stops",
31     "_id": "KV04CmYBzfUEPa98CDhf",
32     "_score": 8.682843,
33     "_source": {
34       "name_stop": "sackville Dr in front of civic address 614"
35     }
36   },
37   {
38     "_index": "bus",
39     "_type": "stops",
40     "_id": "S105CmYBzfUEPa98agY",
41     "_score": 8.682843,
42     "_source": {}
43   },
44   {

```

TEST RESULTS

Comparing the response time taken for the first query to run in local MySQL, cloud and Elasticsearch, we notice that Elasticsearch is more efficient of the three methods. This could be understood by the below screenshot where it has taken $9+10 = 19$ milliseconds for the query.

https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... + ...

POST https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search Params Send Save

Authorization Headers (2) Body Pre-request Script Tests Cookies Code

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

```
1 {
2   "query" : {
3     "match_phrase" : {
4       "stop_id" : "1073"
5     }
6   },
7   "_source" : "route_id"
8 }
```

Body Cookies (1) Headers (3) Test Results Status: 200 OK Time: 265 ms Size: 384 B

Pretty Raw Preview JSON

```
1 {
2   "took": 9,
3   "timed_out": false,
4   "_shards": {
5     "total": 3,
6     "successful": 3,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 43,
12    "max_score": 7.0770593,
13    "hits": [
14      {
15        "_index": "bus",
16        "_type": "stops",
17        "_id": "Q105CmYBzFUEPa9BZ47w",
18        "score": 7.0770593,
```

https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... https://portal-ssl1537-... + ...

POST https://portal-ssl1537-8.bmix-dal-yp-d3361a26-bcb4-46bf-aeaf-dedc77cfef32.443324286.composedb.com:58055/bus/stops/_search Params Send Save

Authorization Headers (2) Body Pre-request Script Tests Cookies Code

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

```
1 {
2   "query" : {
3     "match_phrase" : {
4       "trip_id" : "5808164-2012_05M-12MferSU-Sunday-00"
5     }
6   },
7   "_source" : "trip_headsign"
8 }
```

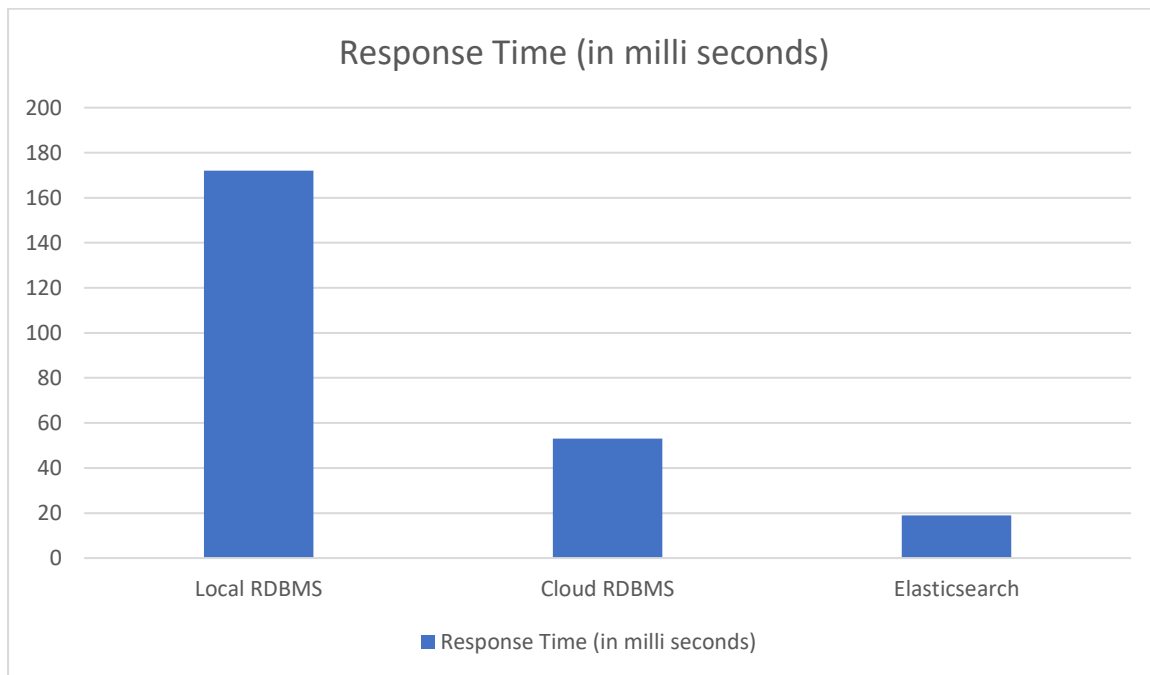
Body Cookies (1) Headers (3) Test Results Status: 200 OK Time: 112 ms Size: 369 B

Pretty Raw Preview JSON

```
1 {
2   "took": 10,
3   "timed_out": false,
4   "_shards": {
5     "total": 3,
6     "successful": 3,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": 3,
12    "max_score": 22.925009,
13    "hits": [
14      {
15        "_index": "bus",
16        "_type": "stops",
17        "_id": "Q105CmYBzFUEPa9BZ3zq",
18        "score": 22.925009,
```

Build Browse

The response time for the first query in local RDBMS, cloud RDBMS and elastic search is represented as a chart:



DISCUSSION AND SUMMARY

We used MySQL Workbench for querying on our local machine. Since the software is a GUI version, we found it to be more appealing and easier to use unlike the command line MySQL server. Loading bulk data into the local database using MySQL workbench also took only a decent amount of time. Since the server runs locally we had no issues with its availability as well. On the contrary, we faced a lot of availability issues with IBM DB2 cloud service as services were down most of the time due to a massive number of users trying to reach the cloud. Apart from this only issue we found the cloud RDBMS to be more efficient than local RDBMS in terms of response time taken for each query. Also, loading huge data sets was comparatively fast.

Elasticsearch was very efficient for queries that search for a matching data/pattern from the database. But loading bulk data into Elasticsearch using Postman REST API was quite tedious as most of the time there were gateway failures. To sum up, elastic search has outplayed both the local and cloud RDBMS in terms of efficiency.