1 Queries in SQL and MongoDB

The given table schemas were:

```
A(A1 integer, A2 string, primary-key(A1))
```

```
B(B1 integer, B2 integer(foreign-key = A1), B3 string, primary-key(B1))
```

The given queries were:

- Find all tuples of A with $A1 \leq 50$.
- Find all tuples of B in sorted order of B3.
- Find average number of values per A1 by using only B table.
- Find all A2 that corresponds to B by using B2 (output the fields of B and A2).

The queries for the above in SQL are given below:

- `select * from A where A1 <= 50;`
- `select * from B order by B3;`
- `select count(*) * 1.00 / count(distinct B2) from B;`
- `select B1, B2, B3, A2 from B, A where B2 = A1;`

The queries in MongoDB are given below:

- `db.A.find({ A1 : { $lte : 50 } })`
- `db.B.aggregate( [ {$sort : {B3:1}} ], {allowDiskUse: true})`
- `db.B.count() / db.B.distinct("B2").length`
- ```
  db.B.aggregate ([
  {
      $lookup : {
          from : "A",
          localField : "B2",
          foreignField : "A1",
          as : "A"
      }
  },
  {
      $unwind : "$A"
  },
  {
      "$project": {
          "_id":0,
  ```
  ```

```
        "B1": 1,
        "B2": 1,
        "B3": 1,
        "A.A2": 1,
    }
}
])
```

2 Times Taken

The data files assigned to me according to the last three digits of my roll number are as follows:

1. A-100.csv , B-100-3-4.csv
2. A-100.csv , B-100-5-2.csv
3. A-100.csv , B-100-10-1.csv
4. A-1000.csv , B-1000-5-2.csv
5. A-1000.csv , B-1000-10-1.csv
6. A-1000.csv , B-1000-50-3.csv
7. A-10000.csv , B-10000-5-1.csv
8. A-10000.csv , B-10000-50-3.csv
9. A-10000.csv , B-10000-500-4.csv

All the times are measured using bash `time` command at different times. 8 runs were taken and two outliers were removed.

DBMS	Queries	A-100.csv , B-100-3-4.csv		A-100.csv , B-100-5-2.csv	
		Average	Std.Deviation	Average	Std.Deviation
SQLite	Q1	0.003	0.001	0.002	0.001
	Q2	0.006	0.001	0.004	0.002
	Q3	0.003	0.001	0.002	0.0
	Q4	0.006	0.003	0.003	0.0
MariaDB with Indexes	Q1	0.005	0.001	0.005	0.001
	Q2	0.008	0.001	0.006	0.003
	Q3	0.006	0.001	0.005	0.0
	Q4	0.009	0.002	0.006	0.001
MariaDB without Indexes	Q1	0.005	0.002	0.005	0.001
	Q2	0.008	0.001	0.007	0.004
	Q3	0.005	0.002	0.005	0.001
	Q4	0.012	0.004	0.01	0.002
MongoDB	Q1	0.057	0.011	0.055	0.006
	Q2	0.081	0.01	0.077	0.008
	Q3	0.052	0.01	0.049	0.002
	Q4	0.115	0.024	0.105	0.01

Table 1: Times for DB 1,2 in seconds

DBMS	Queries	A-100.csv , B-100-10-1.csv		A-1000.csv , B-1000-5-2.csv	
		Average	Std.Deviation	Average	Std.Deviation
SQLite	Q1	0.003	0.001	0.003	0.0
	Q2	0.004	0.003	0.006	0.002
	Q3	0.003	0.001	0.003	0.0
	Q4	0.003	0.0	0.005	0.001
MariaDB with Indexes	Q1	0.005	0.001	0.005	0.0
	Q2	0.008	0.005	0.013	0.003
	Q3	0.006	0.001	0.006	0.0
	Q4	0.006	0.001	0.01	0.002
MariaDB without Indexes	Q1	0.005	0.0	0.005	0.0
	Q2	0.007	0.004	0.013	0.003
	Q3	0.006	0.001	0.006	0.0
	Q4	0.01	0.002	0.272	0.017
MongoDB	Q1	0.054	0.003	0.054	0.007
	Q2	0.092	0.009	0.312	0.026
	Q3	0.055	0.01	0.053	0.01
	Q4	0.129	0.008	1.437	0.044

Table 2: Times for DB 3,4 in seconds

DBMS	Queries	A-1000.csv , B-1000-10-1.csv		A-1000.csv , B-1000-50-3.csv	
		Average	Std.Deviation	Average	Std.Deviation
SQLite	Q1	0.002	0.0	0.002	0.001
	Q2	0.008	0.0	0.027	0.001
	Q3	0.003	0.0	0.006	0.0
	Q4	0.008	0.002	0.02	0.001
MariaDB with Indexes	Q1	0.005	0.001	0.005	0.0
	Q2	0.017	0.001	0.062	0.001
	Q3	0.006	0.0	0.012	0.0
	Q4	0.014	0.001	0.034	0.002
MariaDB without Indexes	Q1	0.005	0.001	0.005	0.0
	Q2	0.018	0.0	0.078	0.013
	Q3	0.007	0.001	0.015	0.001
	Q4	0.46	0.011	2.029	0.043
MongoDB	Q1	0.066	0.021	0.081	0.022
	Q2	0.483	0.011	1.957	0.114
	Q3	0.064	0.021	0.105	0.019
	Q4	2.513	0.079	11.246	0.186

Table 3: Times for DB 5,6 in seconds

DBMS	Queries	A-10000.csv, B-10000-5-1.csv		A-10000.csv, B-10000-50-3.csv	
		Average	Std.Deviation	Average	Std.Deviation
SQLite	Q1	0.003	0.0	0.002	0.0
	Q2	0.036	0.004	0.257	0.015
	Q3	0.009	0.0	0.036	0.001
	Q4	0.027	0.001	0.181	0.013
MariaDB with Indexes	Q1	0.005	0.0	0.005	0.0
	Q2	0.078	0.003	0.69	0.05
	Q3	0.017	0.001	0.084	0.001
	Q4	0.048	0.002	0.29	0.013
MariaDB without Indexes	Q1	0.008	0.001	0.008	0.001
	Q2	0.087	0.006	0.754	0.098
	Q3	0.022	0.004	0.106	0.002
	Q4	25.942	0.806	198.116	9.495
MongoDB	Q1	0.096	0.025	0.073	0.022
	Q2	2.597	0.319	18.209	1.097
	Q3	0.107	0.016	0.206	0.022
	Q4	109.824	3.635	822.083	31.48

Table 4: Times for DB 7,8 in seconds

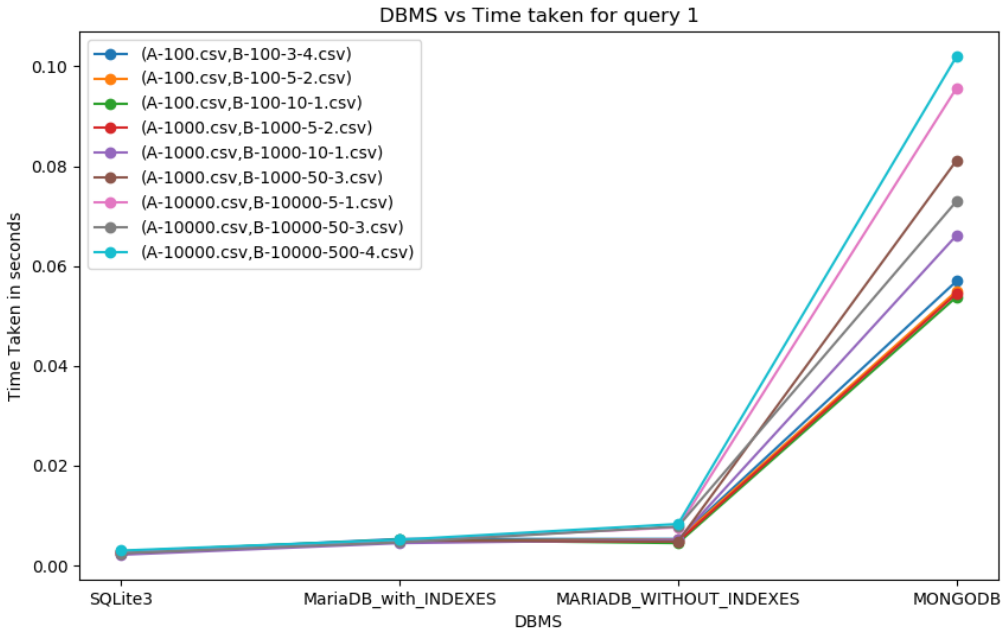
DBMS	Queries	A-10000.csv , B-10000-500-4.csv	
		Average	Std.Deviation
SQLite	Q1	0.003	0.001
	Q2	2.695	0.107
	Q3	0.311	0.014
	Q4	1.765	0.082
MariaDB with Indexes	Q1	0.005	0.0
	Q2	11.712	2.56
	Q3	0.852	0.022
	Q4	13.219	0.983
MariaDB without Indexes	Q1	0.008	0.001
	Q2	12.31	1.296
	Q3	1.064	0.035
	Q4	1908.729	48.294
MongoDB	Q1	0.102	0.015
	Q2	171.175	9.505
	Q3	1.238	0.032
	Q4	7920.63	233.188

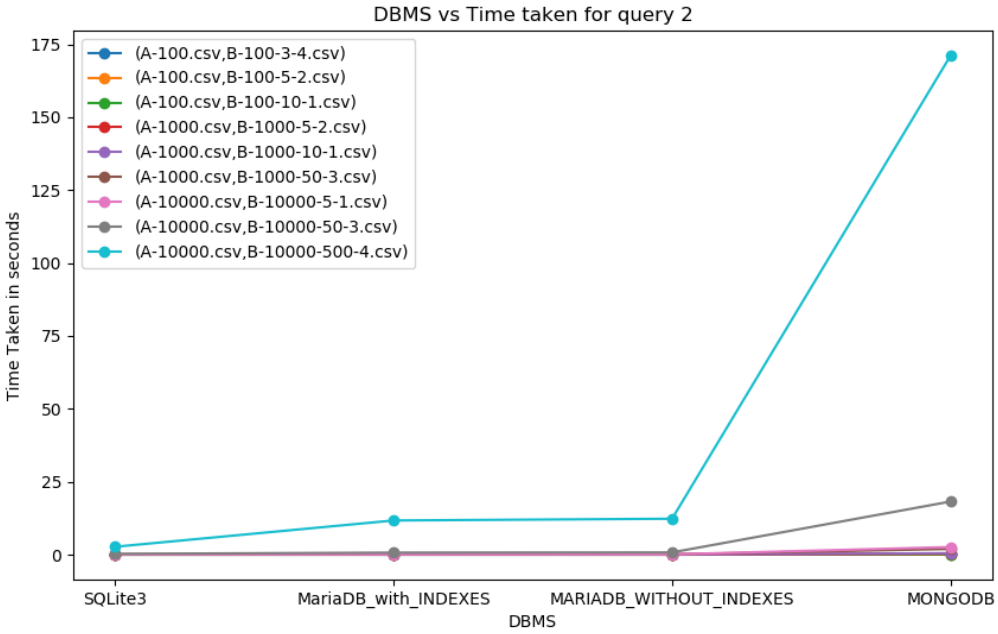
Table 5: Times for DB 9 in seconds

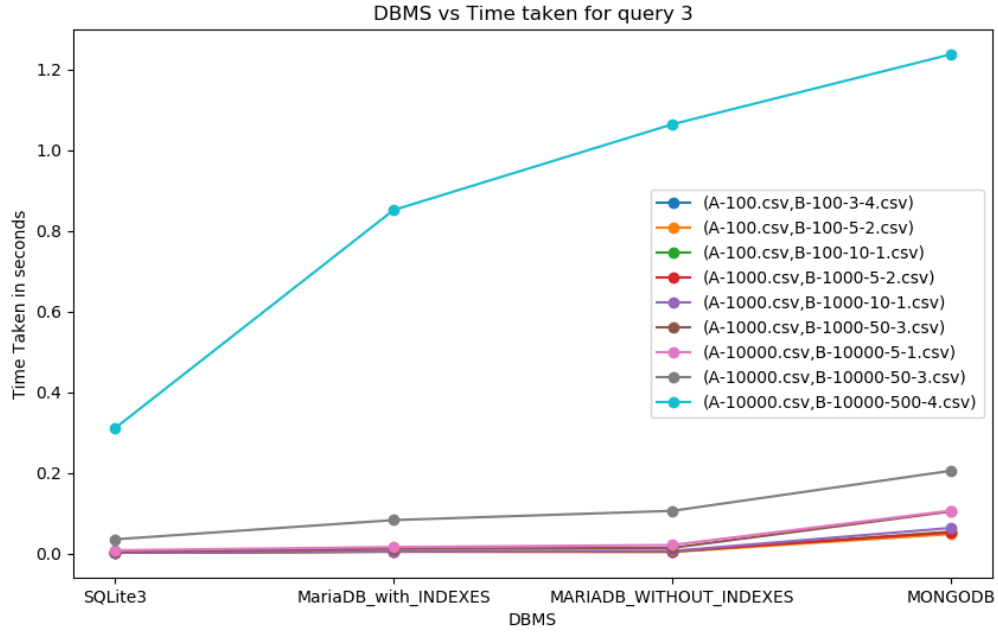
3 Graphs

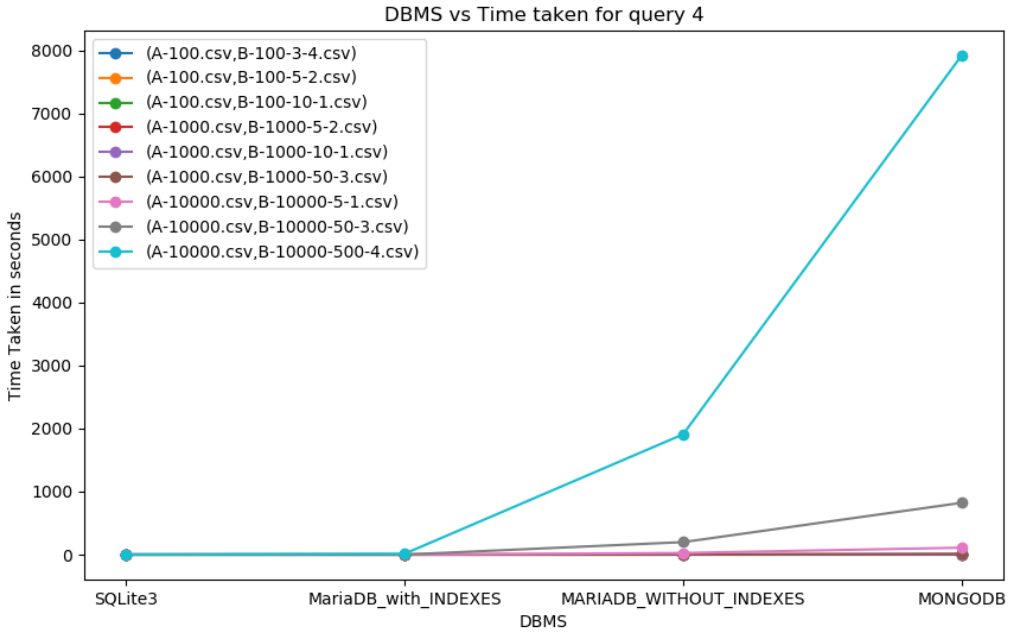
Some of the graphs below may be difficult to interpret due to wide range of values for Time. For more detailed graphs refer to the directory `170268/graphs/extra`.

3.1 DBMS vs Time Taken for each Query

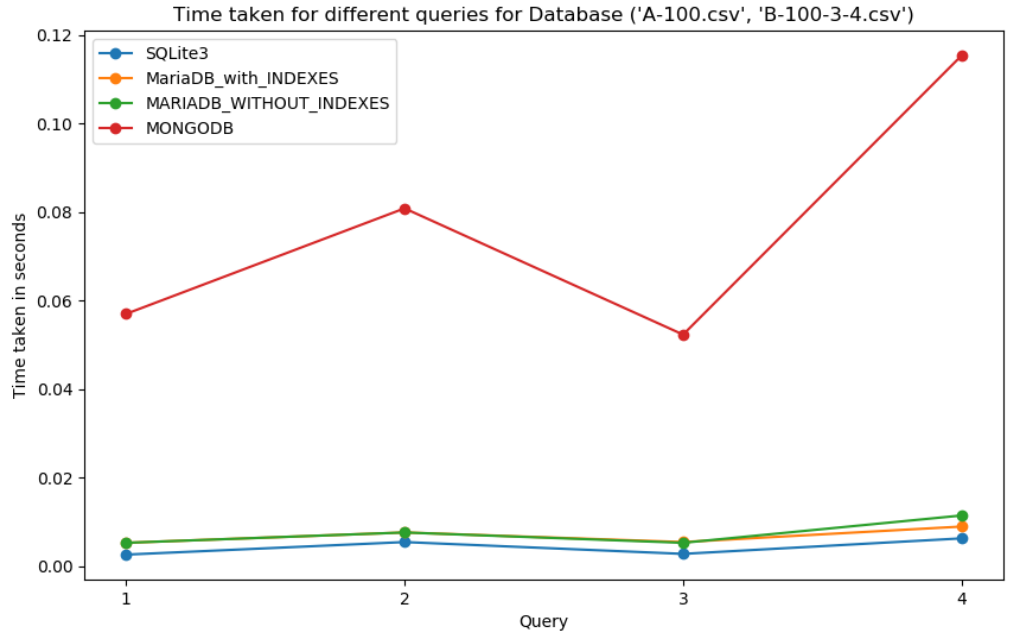


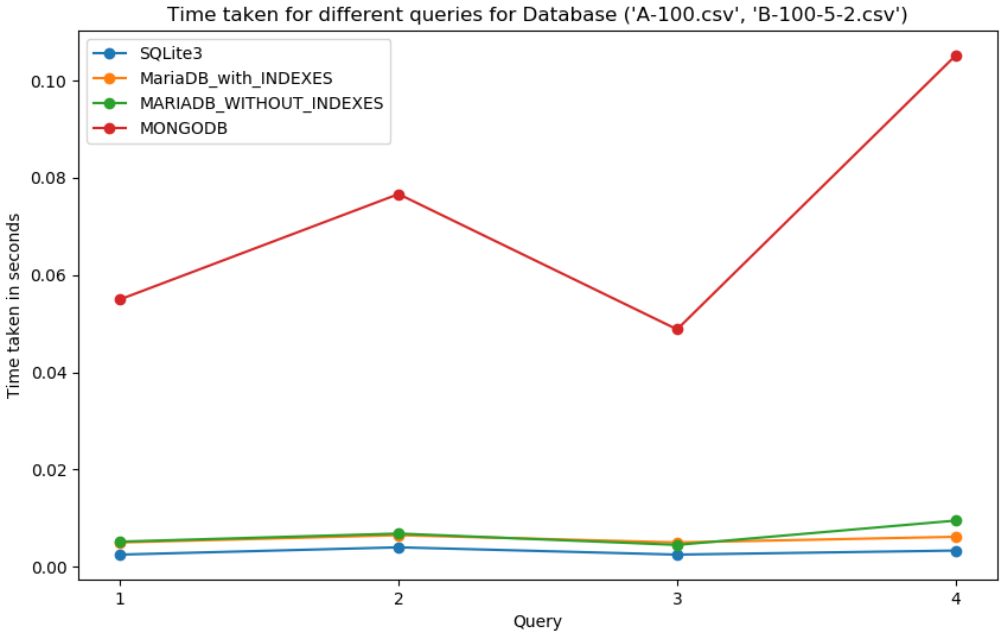

DBMS vs Time taken for query 2
Time Taken in seconds
DBMS
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
0
25
50
75
100
125
150
175
(A-100.csv,B-100-3-4.csv)
(A-100.csv,B-100-5-2.csv)
(A-100.csv,B-100-10-1.csv)
(A-1000.csv,B-1000-5-2.csv)
(A-1000.csv,B-1000-10-1.csv)
(A-1000.csv,B-1000-50-3.csv)
(A-10000.csv,B-10000-5-1.csv)
(A-10000.csv,B-10000-50-3.csv)
(A-10000.csv,B-10000-500-4.csv)

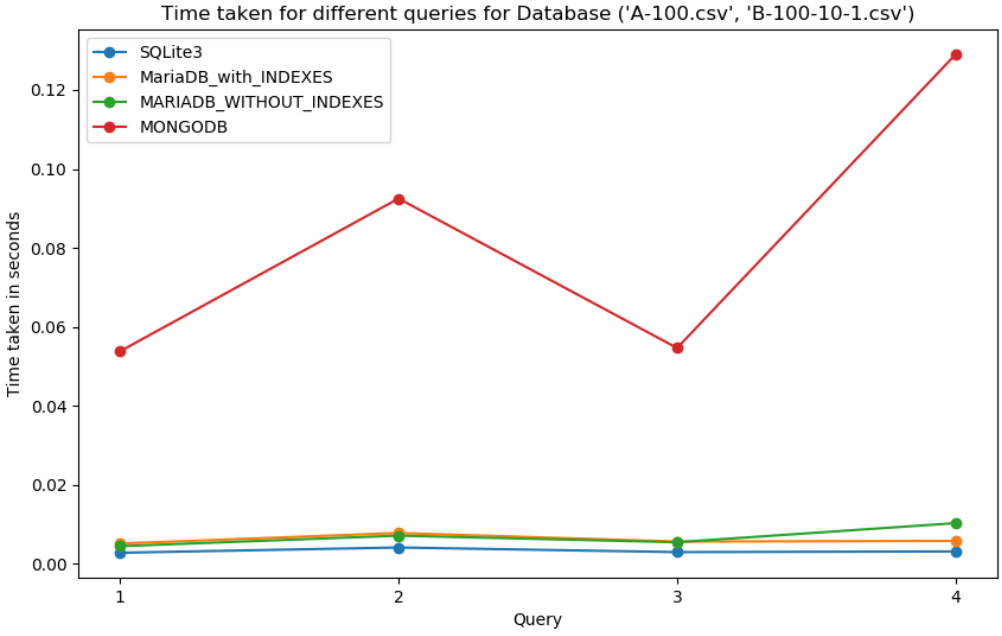

DBMS vs Time taken for query 3
Time Taken in seconds
DBMS
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
0.0
0.2
0.4
0.6
0.8
1.0
1.2
(A-100.csv,B-100-3-4.csv)
(A-100.csv,B-100-5-2.csv)
(A-100.csv,B-100-10-1.csv)
(A-1000.csv,B-1000-5-2.csv)
(A-1000.csv,B-1000-10-1.csv)
(A-1000.csv,B-1000-50-3.csv)
(A-10000.csv,B-10000-5-1.csv)
(A-10000.csv,B-10000-50-3.csv)
(A-10000.csv,B-10000-500-4.csv)

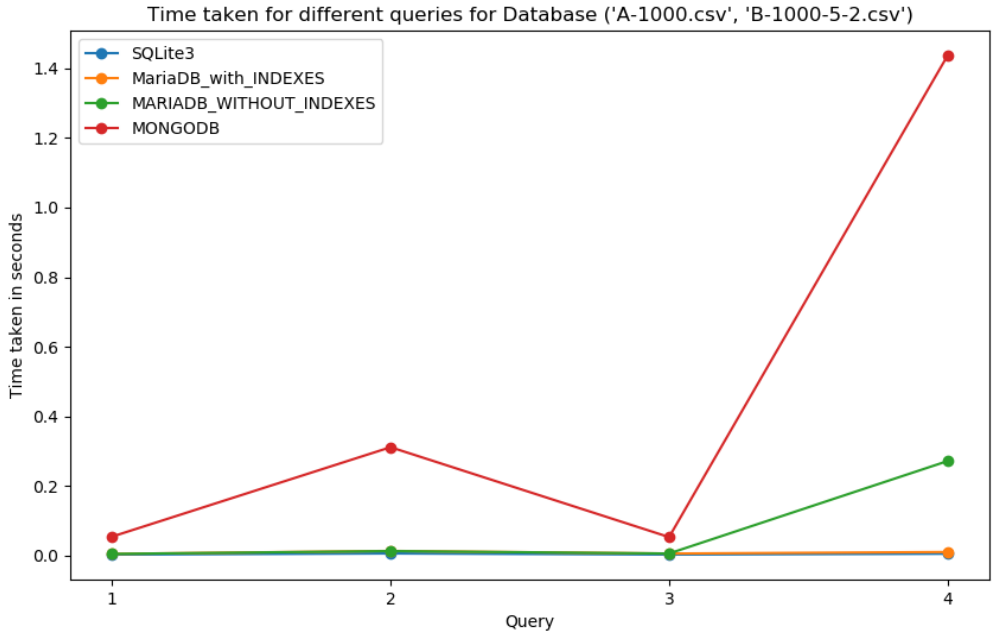


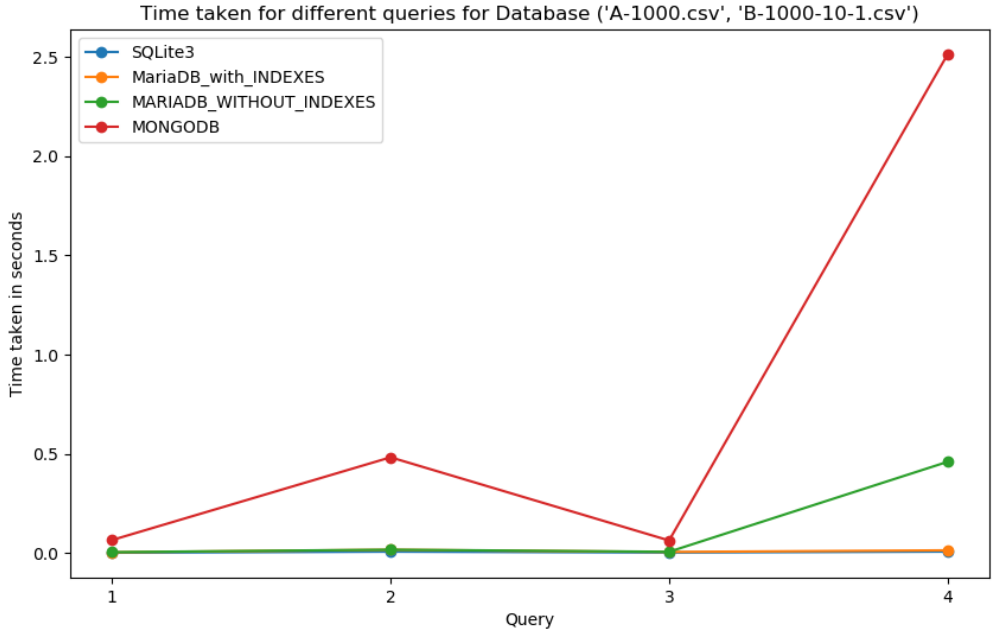
3.2 Query vs Time Taken for each Database

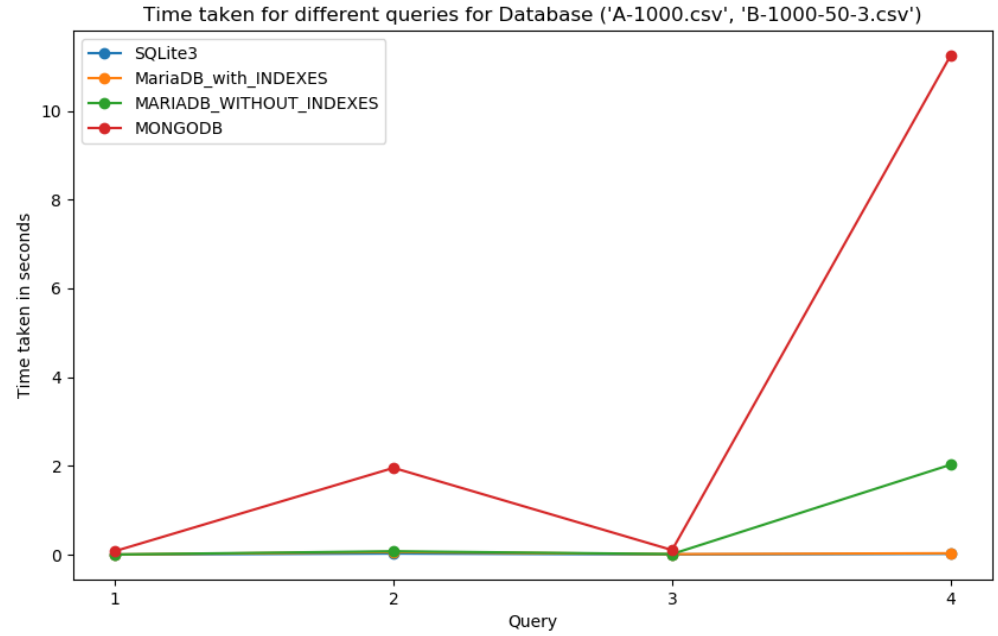


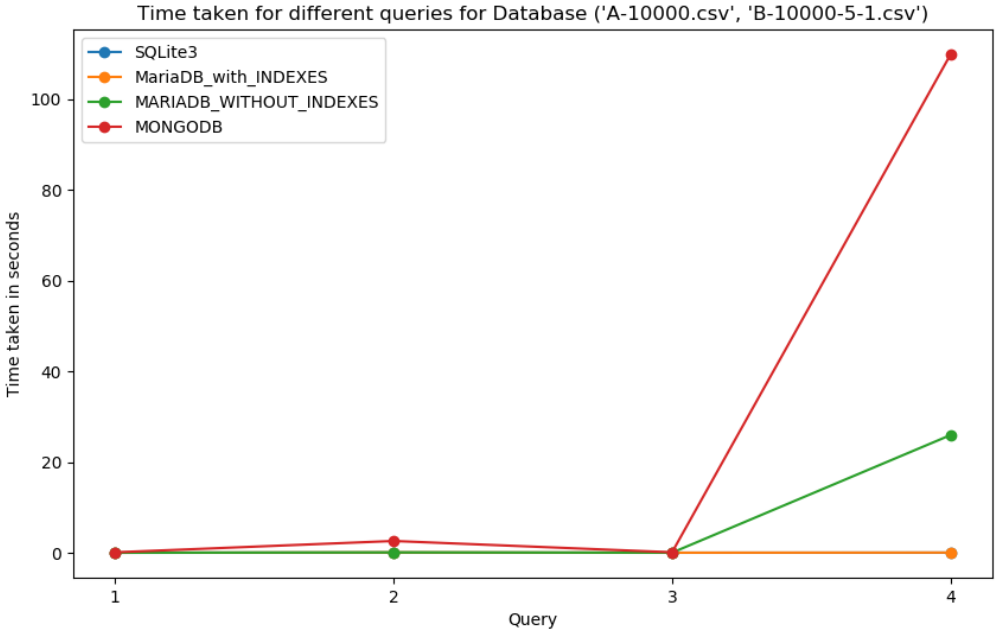

Time taken for different queries for Database ('A-100.csv', 'B-100-5-2.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0.00
0.02
0.04
0.06
0.08
0.10
1
2
3
4
Query

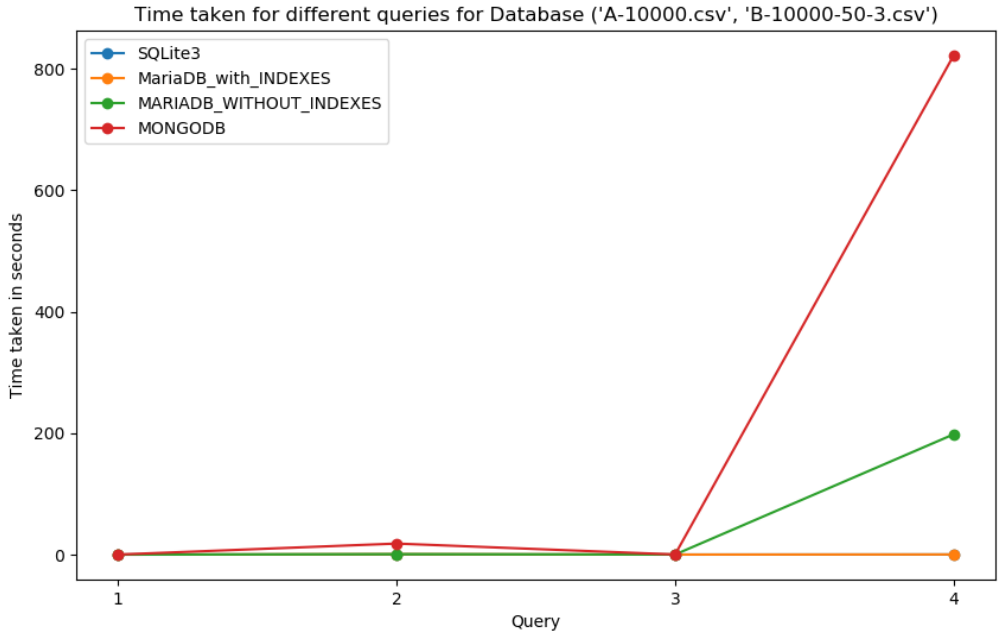

Time taken for different queries for Database ('A-100.csv', 'B-100-10-1.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0.00
0.02
0.04
0.06
0.08
0.10
0.12
1
2
3
4
Query

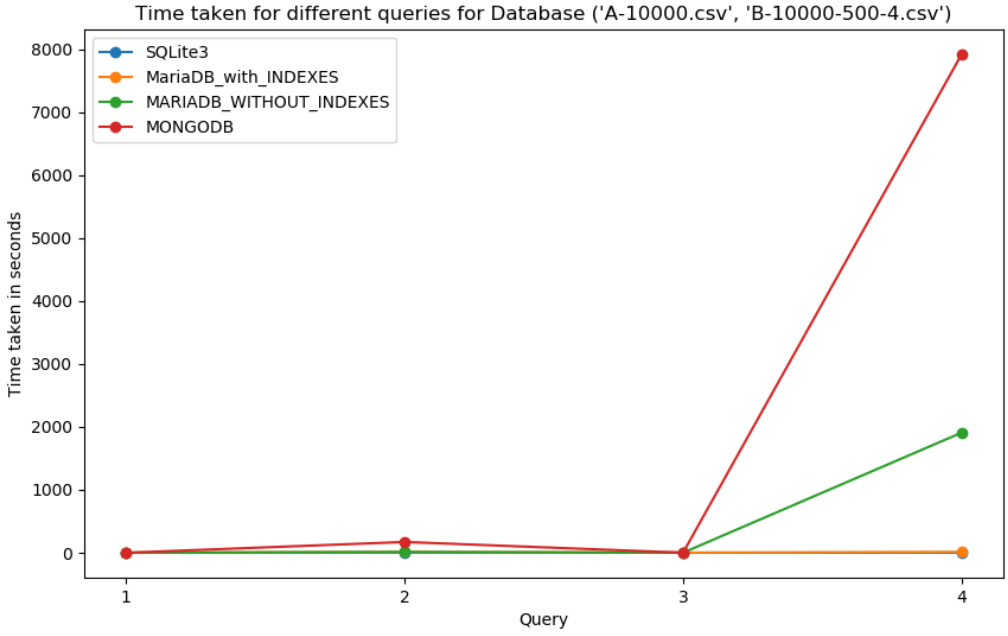

Time taken for different queries for Database ('A-1000.csv', 'B-1000-5-2.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1
2
3
4
Query


Time taken for different queries for Database ('A-1000.csv', 'B-1000-10-1.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
Query

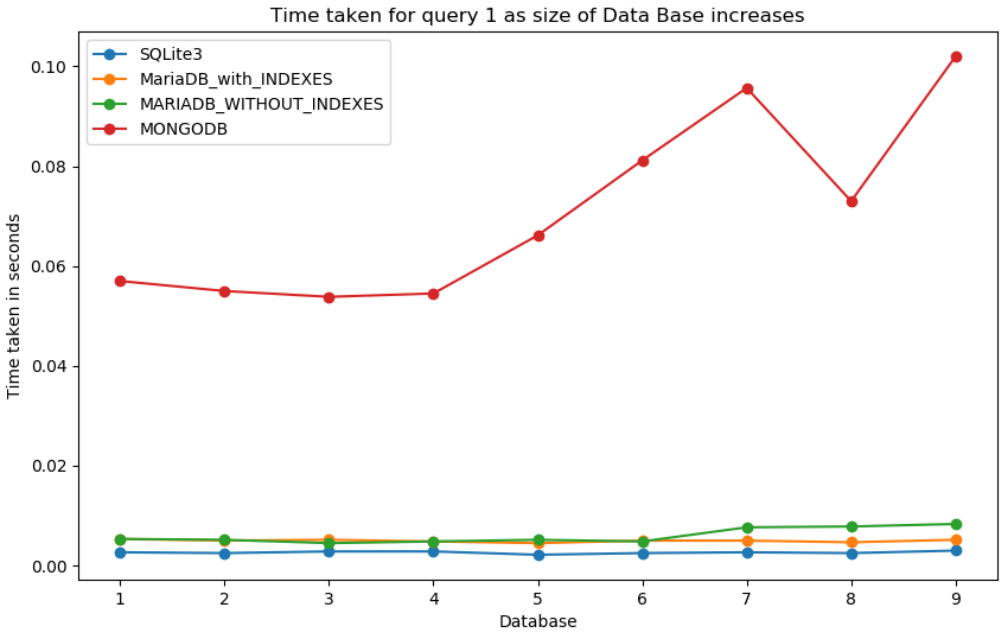

Time taken for different queries for Database ('A-1000.csv', 'B-1000-50-3.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0
2
4
6
8
10
1
2
3
4
Query

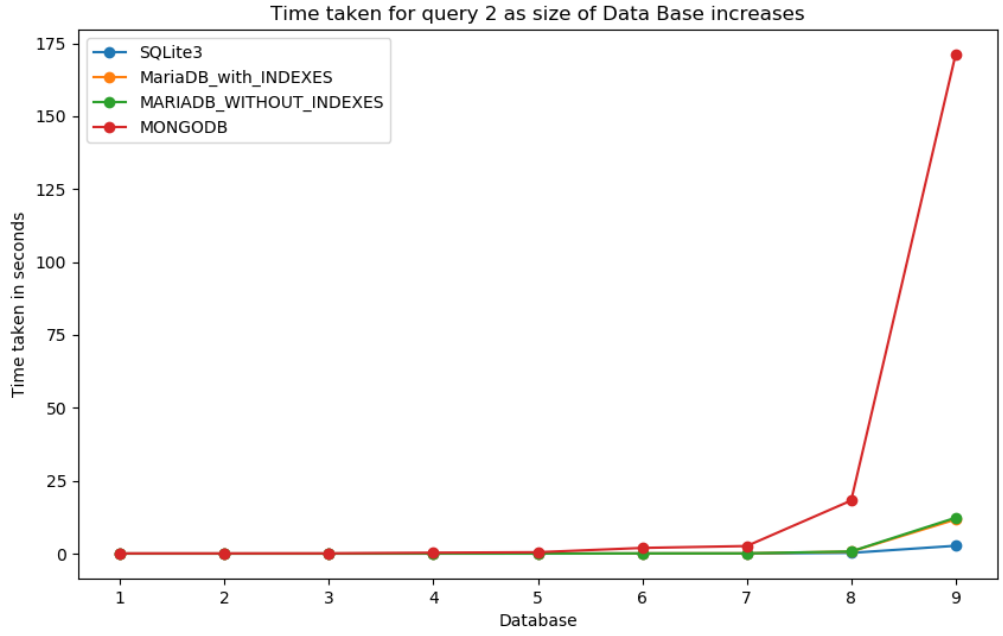

Time taken for different queries for Database ('A-10000.csv', 'B-10000-5-1.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0
20
40
60
80
100
1
2
3
4
Query

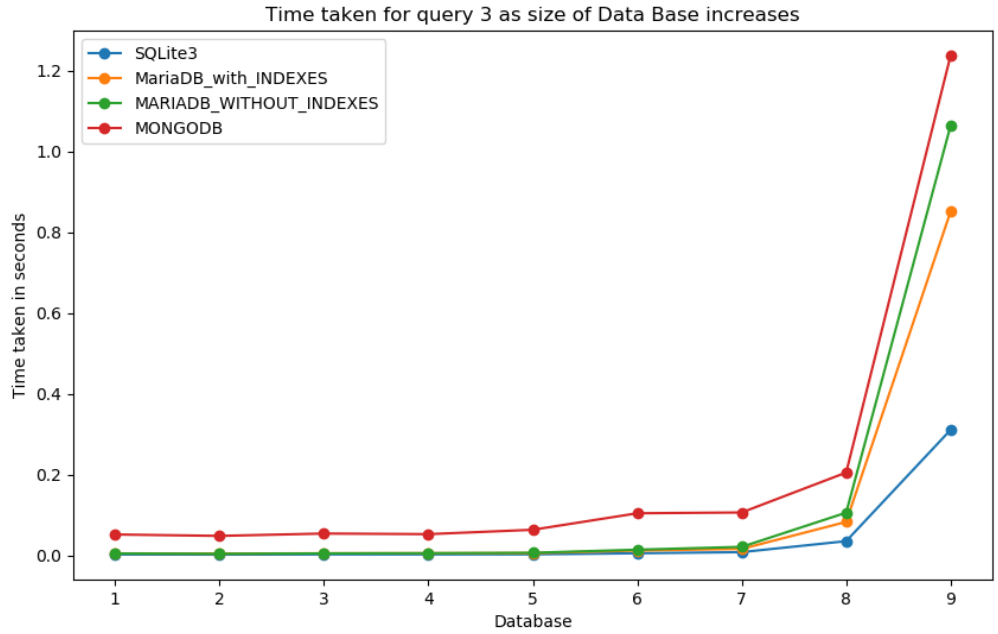

Time taken for different queries for Database ('A-10000.csv', 'B-10000-50-3.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0
200
400
600
800
1
2
3
4
Query

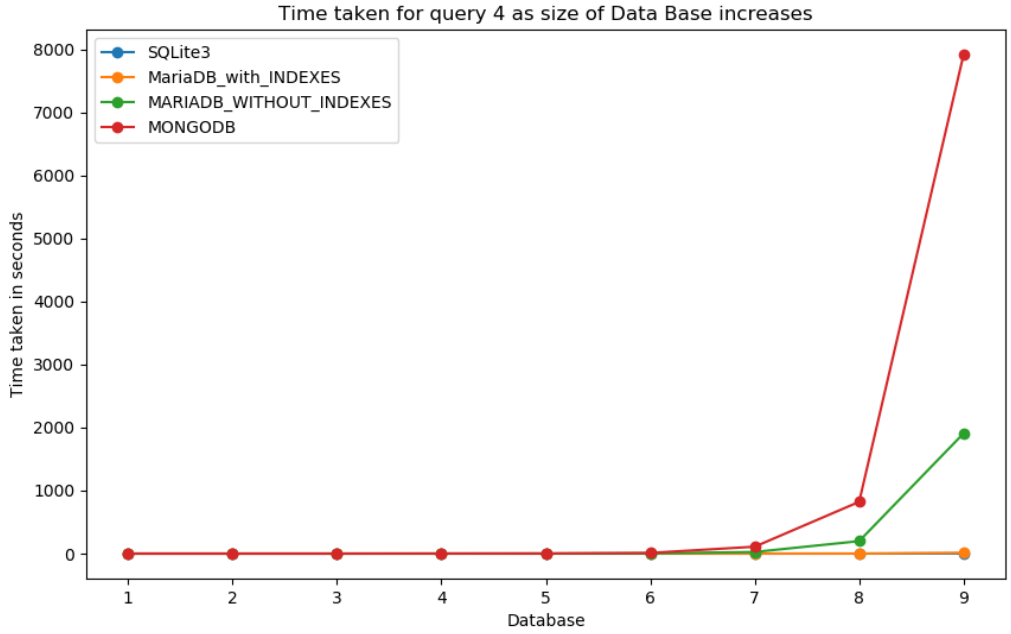

Time taken for different queries for Database ('A-10000.csv', 'B-10000-500-4.csv')
SQLite3
MariaDB_with_INDEXES
MARIADB_WITHOUT_INDEXES
MONGODB
Time taken in seconds
0
1000
2000
3000
4000
5000
6000
7000
8000
1
2
3
4
Query

3.3 Database Size vs Time Taken for each Query









4 Conclusions

Specifications of the Machine on which the above measurements are as follows:

```
OS: Ubuntu 20.04 focal
```

```
RAM: 16GB
CPU: Intel Core i7-8750H @ 12x 4.1GHz
Disk:  1TB HDD/256 GB SSD
SQLite Version :  3.31.1
MariaDB Version :  10.3.25-MariaDB-0ubuntu0.20.04.1
MongoDB Version :  4.4.4
```

- **Scaling of Queries:** As size of Database increases, the time taken also increases for all the Database management systems. The difference in time is more significant for Queries 2 and 4 compared to others, it may be since query 2 involves comparison of strings and Q4 is a computationally heavy join operation which takes time proportional to product of entries in A and B. Also among the database management systems, the difference is more significant in MariaDB without Indexes and MongoDB compared to others. This is because of absence of indexes and non-relational nature of MongoDB.

- **Comparison among Queries:** The order of Times taken for the queries is $Q4 > Q2 > Q3 > Q1$. The trend is self explanatory, Q1 involves a simple select operation on A and also A has fewer entries compared to B hence it is significantly faster than other queries, Q2 and Q3 are similar except that Q2 involves comparison of Strings(B3), while Q3 involves comparison of integers(B2) so Q3 is faster than Q2. Q4 is slower than all other queries since it involves join of two tables.

- **Comparison among DBMS:** The order of Times taken for all the four queries was found to be MongoDB > MariaDB_without_Indexes > MariaDB_with_indexes > SQLite. Since MongoDB is a no sql database and others are relational databases, the queries run slower in MongoDB compared to others. As expected, MariaDB with indexes runs faster than MariaDB without indexes.SQLite and MariaDB with Indexes take almost same time which is expected because SQLite builds indexes on primary and foreign keys by default. An interesting observation is that for Q2, MariaDB takes almost same time with and without index on B3, comparison among strings might be the main reason for this.

- **Effect of writing Output:** Writing the outputs to the terminal/file constitute a fair amount of share in the execution time of a query. The difference in execution times with and without writing the outputs was found to be significant.

- **System Issues:** There is a slight irregularity in the trend of time taken for query 1 as size of database increases which can be seen in the graphs above. This might be due to external factors like cpu load, cpu temperature, bash time overhead etc. Since Q1 is the least expensive query computationally, the effect of external factors on Q1 is higher compared to other queries.

5 Running the scripts

The directory `170268` contains a script named `script.sh` which runs all the queries (9*4*4) once and saves the execution times in relavant files. This script was run 8 times during

different periods to obtain the data. The directory also contains a python script named `analyse.py` which takes the data produced by `script.sh` and produces the relavant graphs. There are also folders corresponding to each DBMS(sqlite,mariadb_with_index, mariadb_without_index) which contain the files for setting up the database, files for the queries and also the outputs of `script.sh`. There is also a folder named `graphs` which contains the graphs produced by `analyse.py`. To run the script locally, one must ensure that databases have a default user, if not the commands for running the queries must be changed accordingly to contain the name and password for the user in the script `script.sh`. Also make sure that the database files mentioned in section 2 are placed in a folder named dbs inside the 170268 directory.

More detailed graphs can be seen in `170268/graphs/extra`. The folder `dbms_time` contains the graphs for dbms vs time for a given query and database. The folder `db_size_time` contains graphs for database size vs time for a given query and dbms. The folder `db_qry_time` contains the graphs for query vs time for a given dbms and database.