Assignment: NoSQL Schema Design and Query Workload Implementation

COMP 5338 Group 12

Contents

1. Introduction

NoSQL systems have become a popular choice as a database for applications with respect to the high performance, scalability, and availability that they perform (Mior, Salem, Aboulnaga & Liu, 2017). In this project, the data set contains the posts, users, votes, and tags. This report will demonstrate that the suitable NoSQL schema design with both MongoDB and Neo4j, and then briefly illustrate the strength and weakness of the query workload about the selection criteria of a schema. The following is the schema design that how to filter and collect data explaining the MongoDB schema and Neo4j schema respectively. In addition, the report makes a comparison in terms of queries and performance after implementing the query design and execution. The elaboration will be interpreted as below.

2. Query Workload

2.1 MongoDB

Selection criteria: Except for the two compulsory queries (SQ1, SQ2), we select AQ1, AQ2, AQ4 as the target queries for MongoDB. The reasons show in the below.

- MongoDB is a document-based database, which is more suitable for date range search, so we choice AQ1, AQ2, AQ4.
- AQ1, AQ2, AQ4 need many common properties. Therefore, we can minimize the unnecessary data when cleaning the data, and the remaining selected data is efficient.
- MongoDB is suitable for queries that require complex aggregation operations.

2.2 Neo4j

Selection criteria: Except for the two compulsory queries (SQ1, SQ2), we select AQ3, AQ5, AQ6 as the target queries for MongoDB. The reasons show in the below.

- Neo4j is suitable for querying and storing graphical data, such as interpersonal relationships, so we chose AQ3, AQ6, AQ5. Both of these problems are more suitable for graphical data since the data can be queried through the traversal of the path.

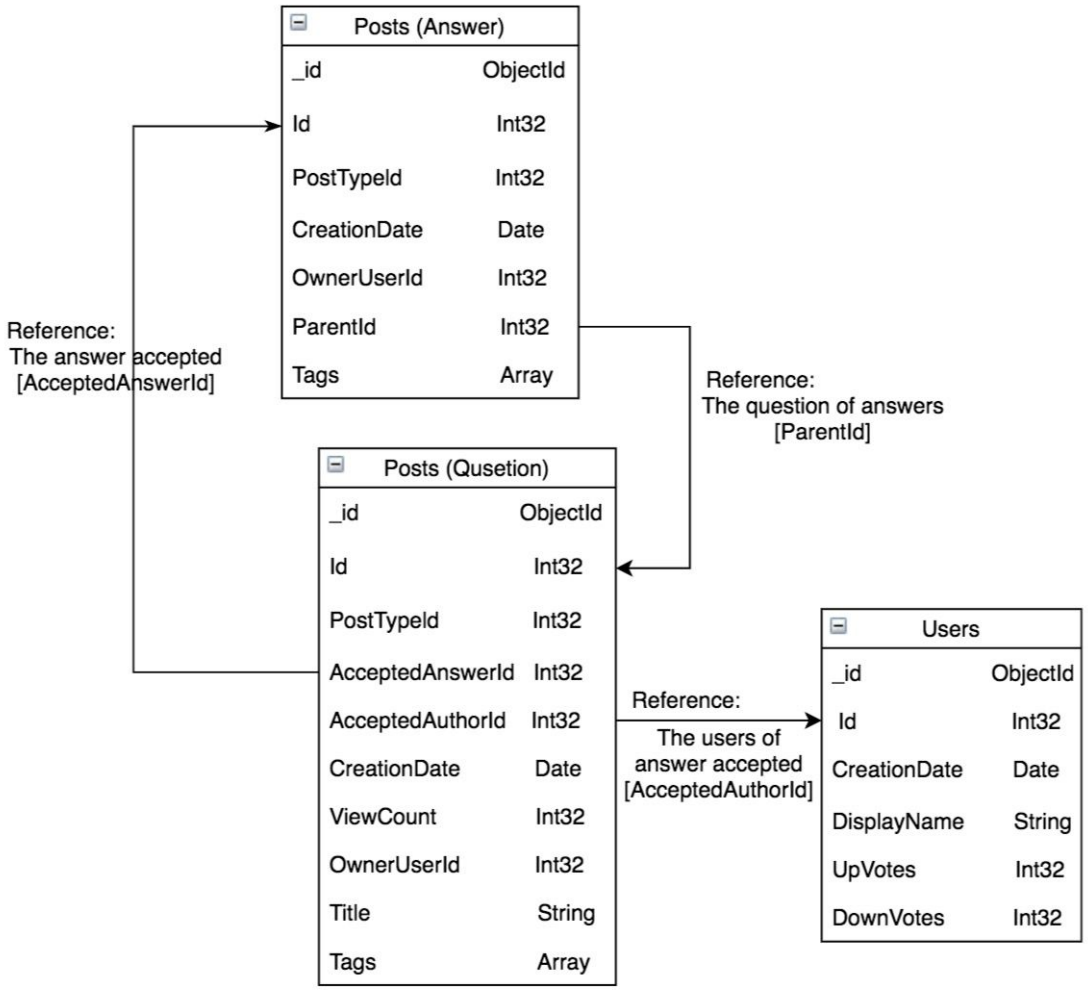
3. Schema Design

The following diagrams briefly describe the data model schema and their relationships in both MongoDB and neo4j.

3.1 MongoDB Schema Design

3.1.1 MongoDB Data Model

The schema design of MongoDB is shown as the below picture (Picture 1).



Picture 1

***Explanation:**

There is only one Posts collection in the schema. Just use "two collections" in the above picture to represent the different properties of the two types of posts (Questions & Answers).

The total collections for Query {SQ1,SQ2,AQ1,AQ2,AQ4} used are Posts and Users.

For Posts Collection:

- **For The Whole Collection:**
  In order to optimize the query about the "Tags" property, the data type of Tags is changed from **String to String Array**, and it is partially embedded into all documents(only TagName is embedded). By this way, the grouping query and the "unwind" stage based on the "Tag" property can be more efficient.

- **For Question Documents**:
  Using the "PostTypeId:1" to represent the questions in post collection. Based on the analysis of the selected queries for MongoDB, these properties are removed from the collection: score, ClosedDate, OwnerDisplayName, FavoriteCount, CommentCount, and AnswerCount. Furthermore, we added the new field "AcceptedAuthorId" as the reference from the answer collection and this new field will not increase too much query cost, since **the original collection for question already has "AcceptedAnswerId" as one property, and "AcceptedAuthorId" field just the reference of "OwnerUserId" the accepted answer. Thus, both "AcceptedAnswerId" and the original "AcceptedAnswerId" come from the same answer document, which avoids too many costs when a new Write operation executed.**

- **For Answer Documents:**
  Using "PostTypeId:2" to represent answer in one posts collection. As mentioned before, adding "Tags" array in all Answer documents. By this way, **the query, such as AQ2, AQ4, can avoid the more join operation ("lookup stage") which could be the great query cost.** In addition, there are one-to-many relationships with partially embedded methods.

For Users Collection:

Deleted some fields without using in selected queries, which is Reputation LastAccessDate, Location Views, and AccountId.

3.1.2 Performance Plan

The performance plan illustrates the method of query performance analysis. To realize the optimized performance, using a **system profile** command can monitor the performance of the query on MongoDB. We set “db.setProfilingLevel(2)” to monitor all query execution. By this way, the MongoDB system will automatically generate the log documents when a query is executed. Secondly, executing the command “db.system.profile.find({millis:{$gte:1}})” to return the query histories, which the query time is more than 1 milliseconds. The following shows some fields using for the performance analysis.

- KeysExamined: the number of index scanning.
- docsExamined: the number of documents scanning.
- stages: According to index, stages retrieves the time when the document

3.1.3 Sample Document

- Posts (Qusestion)


```
{
    "_id" : ObjectId("5baf3bce11ae17bbac4d5feb"),
    "Id" : 4,
    "PostTypeId" : 1,
    "AcceptedAnswerId" : 12,
    "CreationDate" : ISODate("2016-08-02T15:41:22.020Z"),
    "ViewCount" : 719,
    "OwnerUserId" : 8,
    "Title" : "How to find the optimal number of neurons per layer?",
    "Tags" : [
        "deep-network",
        "neurons"
    ],
    "AcceptedAuthorId" : 4
}
```


- Posts (Answer)


```
{
    "_id" : ObjectId("5baf3bce11ae17bbac4d5fee"),
    "Id" : 14,
    "PostTypeId" : 2,
    "CreationDate" : ISODate("2016-08-02T15:52:24.380Z"),
    "OwnerUserId" : 52,
    "ParentId" : 5,
    "Tags" : [
        "mindstorms"
    ]
}
```

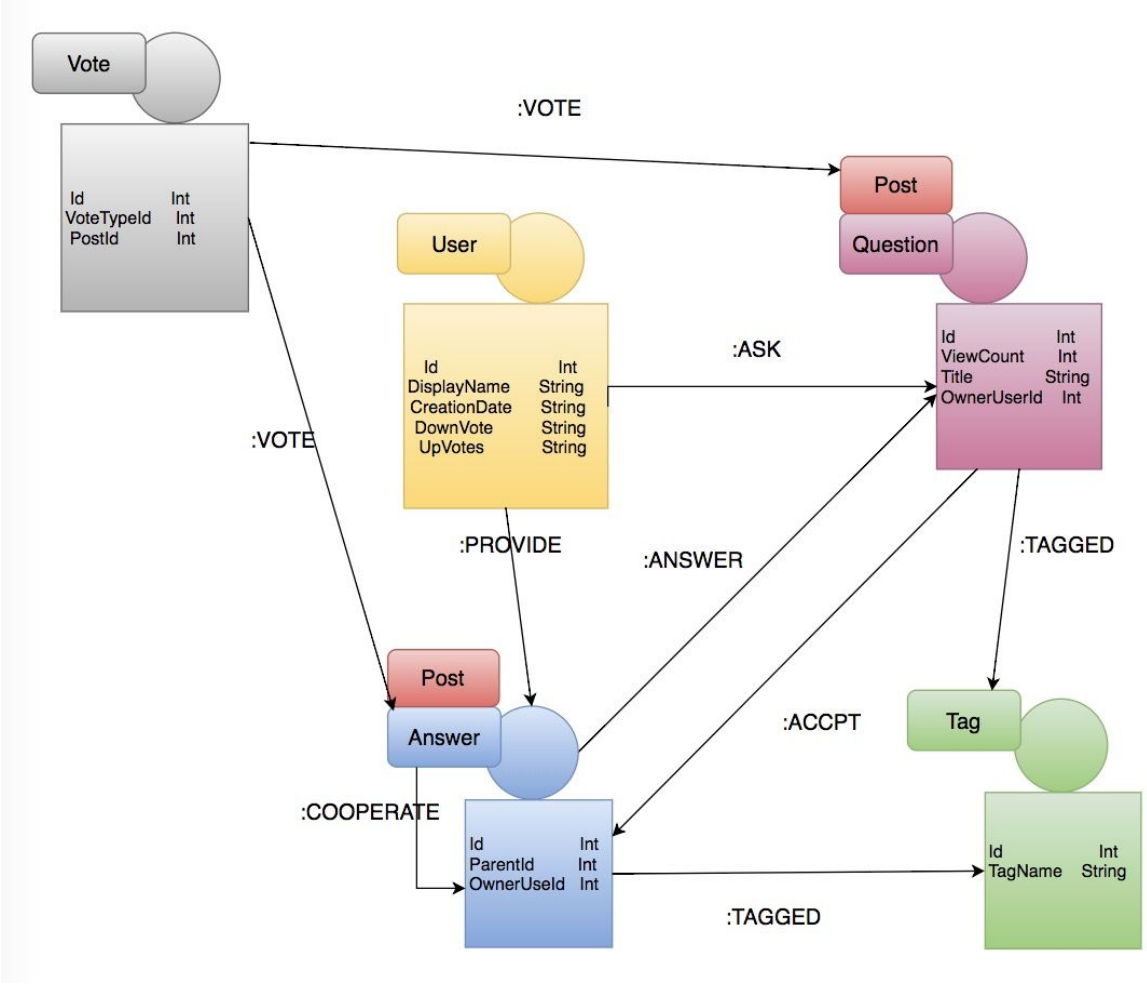

- Users


```
{
    "_id" : ObjectId("5bb45f210238e906f28f4bb8"),
    "CreationDate" : ISODate("2016-08-02T15:38:01.633Z"),
    "DisplayName" : "Geoff Dalgas",
    "UpVotes" : 0,
    "DownVotes" : 0,
    "Id" : 3
}
```

3.2 Neo4j Schema Design

3.2.1 Neo4j Data Model

The schema design of Neo4j is shown as the below picture (Picture 2).



Picture 2

The properties of relationships:

:PROVIDE: QuestionId **:ASK:** QuestionId

***Explanation of the schema:**

:VOTE: this relationship describes a vote to a post node.

:ASK: this relationship describes an user posts a question.

:PROVIDE: this relationship describes an user posts an answer.

:ANSWER: this relationship describes which question is an answer answers.

:ACCEPT: this relationship describes a question accept an answer.

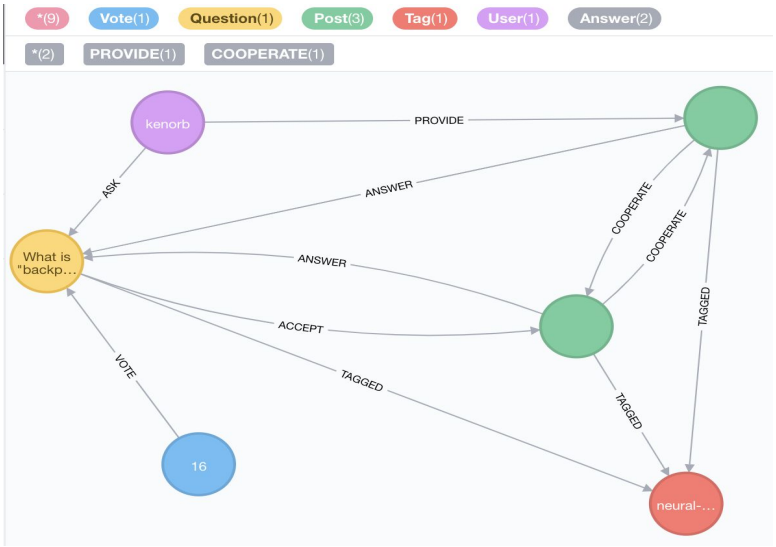
:TAGGED: this relationship describes a post belongs to what types of tag.

:COOPERATE: this relationship describes answers that answer the same question.

3.2.2 Performance Plan

For Neo4j, we use neo4j-shell application to monitor the performance of the query. The monitor of the performance focus on the space performance and its time space. For the space performance we use neo4j-shell to monitor the estimated rows of the stages of each query. To further monitor the performance, we will compare the space and time performance of a query with index and without index.

3.2.3 Sample Graph



Two green circles and yellow circles represent the answer and question respectively, which both belongs to the Posts collection. It is noted that the yellow one and green ones are question and answers respectively.

4. Query Design and Execution

4.1 MongoDB

4.1.1 Query for [SQ1]

- Query Requirement
  Find all users involved in a given question (identified by id) and their respective profiles including the creationDate, DisplayName, upVote and DownVote.
- Query Design
  To avoid join operation (“lookup”) stage, we use two subqueries to get the ultimate results.

Sub-query 1: find the all user id involved in a given question.

```
query = [
    {'$match': {'$or': [{'ParentId': question_id}, {'Id': question_id}],'OwnerUserId':{'$ne':""}}},
    {'$project': {'_id': 0, 'OwnerId': '$OwnerUserId'}}
        ]
```

Query Design	
Input Parameter	question_id (data type: int)
Output Result	the set of user id

Sub-query 2: find personnal information of involved users.

```
query = [
    {'$match':{'Id': owner_id}},
    {'$project':{'_id':0,'DisplayName':1,'CreationDate':1,'UpVotes':1,'DownVotes':1}}
        ]
```

Query Design	
Input Parameter	user_id (data type: int)
Output Result	CreationDate, DisplayName, UpVotes, DownVotes

- Query Execution

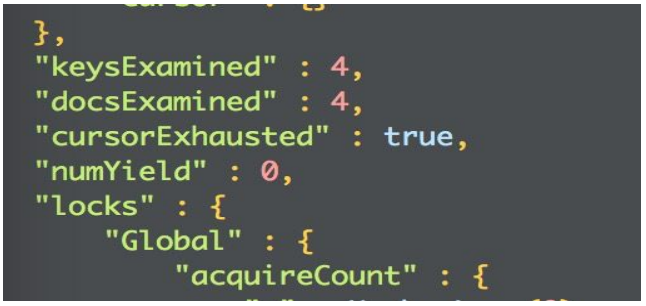
```
1. bash
vlan-2631-10-19-0-130:src mistletoe$ python3 main.py -type mongo -query sq1 -sq1 2
[{'CreationDate': '2016-08-02T15:38:21.100', 'DisplayName': 'Franck Dernoncourt', 'UpVotes': 15, 'Dow
nVotes': 2}]
[{'CreationDate': '2016-08-02T15:38:37.680', 'DisplayName': 'Matthew Graves', 'UpVotes': 64, 'DownVot
es': 25}]
[{'CreationDate': '2017-05-31T09:46:16.857', 'DisplayName': 'Tshilidzi Mudau', 'UpVotes': 24, 'DownVo
tes': 1}]
[{'CreationDate': '2016-08-02T15:38:36.723', 'DisplayName': 'kenorb', 'UpVotes': 559, 'DownVotes': 3}
]
```

- Query Performance Analysis

  Index optimization: creating the {ParentId:1} index and {Id:1} index separately, the system selects both indexes as the optimized options. The Id index scans the posts collection related with questions, whereas ParentId index is used for scanning the posts collection related with answers. As the graph shows, the optimized combines “FETCH” and “IXSCAN” in the winningPlan after using the index. “keysExamined : 4 = docsExamined: 4”, which means the number of index scan is equal with the number of documents scan. The time of operation is approximately 0 millisecond. The actual performance shows as the below pictures.


"winningPlan" : {
"stage" : "CACHED_PLAN",
"inputStage" : {
"stage" : "FETCH",
"filter" : {
"$nor" : [
{
"OwnerUserId" : {
"$eq" : ""
}
}
]
},
"inputStage" : {
"stage" : "OR",
"inputStages" : [
{
"stage" : "IXSCAN",
"keyPattern" : {
"ParentId" : 1.0
},
"indexName" : "ParentId_1",
"isMultiKey" : false,
"multiKeyPaths" : {
"ParentId" : []


"isUnique" : false,
"isSparse" : false,
"isPartial" : false,
"indexVersion" : 2,
"direction" : "forward",
"indexBounds" : {
"ParentId" : [
"[2.0, 2.0]"
]
}
},
{
"stage" : "IXSCAN",
"keyPattern" : {
"Id" : 1.0
},
"indexName" : "Id_1",
"isMultiKey" : false,
"multiKeyPaths" : {
"Id" : []


},
"keysExamined" : 4,
"docsExamined" : 4,
"cursorExhausted" : true,
"numYield" : 0,
"locks" : {
"Global" : {
"acquireCount" : {


"responseLength" : 173,
"protocol" : "op_command",
"millis" : 0,
"planSummary" : "IXSCAN { ParentId: 1 }, IXSCAN { Id: 1 }",
"ts" : ISODate("2018-10-05T12:04:32.772Z"),
"client" : "127.0.0.1",

4.1.2 Query for [SQ2]

- Query Requirement
  Assuming each tag represents a topic, find the most viewed question in a given topic.

- Query Design

```
query = [{'$match': {'Tags': topic,'PostTypeId':1}},{'$sort':{'ViewCount':-1}},{'$limit':1}]
```

Query Design	
Input Parameter	topic (data type: String)
Output Result	_id, Id,PostTypeId, AcceptedAnswerId, CreationDate, ViewCount, OwnerUserId, Title, Tags, AcceptedAuthorId

- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type mongo -query sq2 -sq2 n
eural-networks
{'_id': ObjectId('5baf3bce11ae17bbac4d60df'), 'Id': 1294, 'PostTypeId': 1, 'Acce
ptedAnswerId': 1313, 'CreationDate': datetime.datetime(2016, 8, 4, 8, 28, 6, 277
000), 'ViewCount': 19807, 'OwnerUserId': 144, 'Title': 'How does Hinton\'s "caps
ules theory" work?', 'Tags': ['neural-networks'], 'AcceptedAuthorId': 10}
vlan-2631-10-19-6-0:src mistletoe$
```

- Query Performance Analysis

  Index optimization: After using Tags index and PostTypeId index, the system selects both indexes as the optimized options that is {Tags:1,PostTypeId:1}. It is better than the performance that using “Tags”. The following picture shows the optimized results that combine “FETCH” and “IXSCAN” in the winningPlan, and keysExamined number is equal with docsExamined number”. The time of operation is 5 milliseconds.

4.1.3 Query for [AQ1]

- Query Requirement

  Given a list of topics (tags), find the question easiest to answer in each topic.

- Query Design
  According to Aggregation Pipeline Optimization (2018), in order to increase the efficiency, the first stage is *$match,* which can reduce the number of documents scanned for the next stages. When *$unwind* follows *$ lookup* immediately and the *$unwind* operates on the as field of the *$lookup*, the optimizer can merge *$unwind* into the *$ lookup* phase. This avoids creating large intermediate documents.

```
query = lambda topic:[
    {'$match':{'Tags':topic}},
    {'$lookup': {'from': 'posts', 'localField': 'AcceptedAnswerId', 'foreignField': 'Id', 'as': 'Answers'}},
    {'$unwind': '$Answers'},
    {'$project': {'_id': 0, 'Id':1,'Title': 1, 'Interval': {'$subtract': ['$Answers.CreationDate', '$CreationDate']}}},
    {'$sort':{'Interval':1}},
    {'$limit':1}
        ]
```

Query Design	
Input Parameter	topics (data type: String Array)
Output result	Id, Title, Interval

- Query Execution

```
vlan-2631-10-19-0-130:src mistletoe$  python3 main.py -type mongo -query aq1 -aq1 neural-networks
[{'Id': 1, 'Title': 'What is "backprop"?', 'Interval': 69873}]
vlan-2631-10-19-0-130:src mistletoe$
```

- Query Performance Analysis
  Index optimization:
  For this query, we create an index of “{Tags:1}”. The picture below shows the use of the index. Thus, by adding this index, this “winningPlan” select “FETCH” as the stage and “IXSCAN” as the inputStage. The pictures below shows the performance of this query by adding the index.

4.1.4 Query for [AQ2]

- Query Requirement
  Given a time period as indicated by starting and ending date, find the top 5 topics in that period.
- Query Design

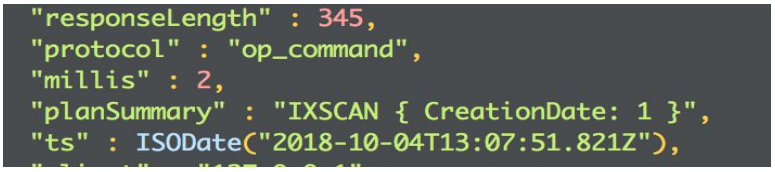
```
query = [
    {'$match': {'CreationDate': {'$gte': iso_start_date, '$lte':iso_end_date}}},
    {'$unwind': '$Tags'},
    {'$group': {'_id': '$Tags', 'Users': {'$addToSet': '$OwnerUserId'}}},
    {'$project': {'_id':0,'Topic': '$_id', 'Num': {'$size': '$Users'}}},
    {'$sort': {'Num': -1}},
    {'$limit': 5}
]
```

Query Design	
Input Parameter	start_date (data type: String), end_date (date type: String)
Output result	Topic, Num

- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type mongo -query aq2 -aq2 2
018-08-01T00:00:00 2018-08-31T00:00:00
{'Topic': 'neural-networks', 'Num': 65}
{'Topic': 'machine-learning', 'Num': 44}
{'Topic': 'deep-learning', 'Num': 39}
{'Topic': 'reinforcement-learning', 'Num': 28}
{'Topic': 'convolutional-neural-networks', 'Num': 24}
vlan-2631-10-19-6-0:src mistletoe$
```

- Query Performance Analysis
  Index optimization: creating the index of {CreationDate:1}can enhance the result. As the picture shows, queryplanner combines "FETCH" and "IXSCAN" in the winningPlan after using the index. The optimized results that combine "FETCH" and "IXSCAN" in the winningPlan, and keysExamined number is equal with docsExamined number". The time of operation is 2 milliseconds.



4.1.5 Query for [AQ4]

- Query Requirement
  Recommend unanswered questions to a given user.
- Query Design

```
query = [
    {'$match':{'AcceptedAuthorId':user_id}},
    {'$unwind':'$Tags'},
    {'$group':{'_id':'$Tags','num':{'$sum':1}}},
    {'$sort':{'num':-1}},
    {'$match':{'num':{'$gte':alpha}}}
]
```

Query Design	
Input Parameter	user_id (data type: int), alpha (data type: int)
Output result	_id, Id,PostTypeId, AcceptedAnswerId, CreationDate, ViewCount, OwnerUserId, Title, Tags, AcceptedAuthorId

```
query = [
    {'$match':{'PostTypeId':1,'AcceptedAuthorId':0,'Tags':{'$in':topics},'CreationDate':{'$lte':iso_date}}},
    {'$project':{'_id':0,'Id':1,'Title':1,'CreationDate':1}},
    {'$sort':{'CreationDate':-1}},
    {'$limit':5}
]
```

Query Design	
Input Parameter	post_id (data type: int), date (data type: String)
Output result	Id, Title, CreationDate

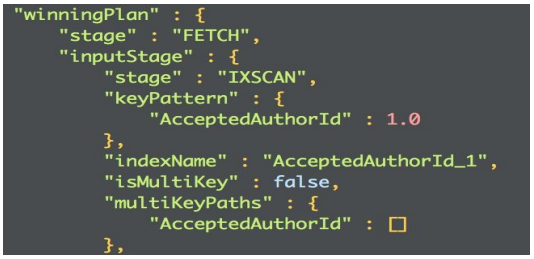
- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type mongo -query aq4 -aq4 4
398 4 2018-08-30T00:00:00
{'Id': 7755, 'CreationDate': datetime.datetime(2018, 8, 29, 16, 4, 16, 113000),
'Title': 'How to implement a constrained action space in reinforcement learning?
'}
{'Id': 7737, 'CreationDate': datetime.datetime(2018, 8, 28, 0, 17, 55, 907000),
'Title': 'creating application to transform human computer experience into physi
cal activity'}
{'Id': 7736, 'CreationDate': datetime.datetime(2018, 8, 27, 18, 41, 56, 223000),
 'Title': 'In imitation learning, do you simply inject optimal (state, action, r
eward, s(t+1)) experiences into your experience replay buffer?'}
{'Id': 7734, 'CreationDate': datetime.datetime(2018, 8, 27, 16, 13, 16, 433000),
 'Title': 'AI composing music'}
{'Id': 7727, 'CreationDate': datetime.datetime(2018, 8, 27, 10, 18, 17, 893000),
 'Title': 'How is it possible to teach a neural network to perform addition?'}
vlan-2631-10-19-6-0:src mistletoe$
```

- Query Performance Analysis

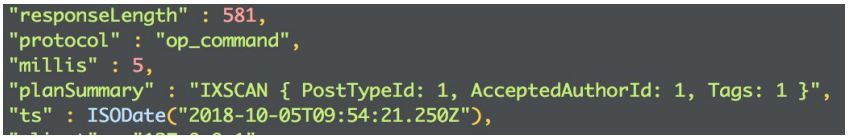
  AQ4-1：

  Index optimization: creating the index of “AcceptedAuthorID” is the optimized results. As the graph shows, we carry out the optimized combination “FETCH+IXSCAN” stage in the winningPlan after using the index of “AcceptedAuthorID”. As the picture below, “keysExamined : 27 = docsExamined: 27” , the number of index scan is equal with the number of documents scan. It is the ideal result, which means no extra index scan and document scan.The following picture shows the time of operation nearly: “millis:0”.



AQ4-2

Index optimization: creating the index of {PostTypeId:1, AcceptedAuthorId:1,Tags:1} is the results. As the graph shows, the optimized combines "FETCH" and "IXSCAN" in the winningPlan after using index. "keysExamined : 329 = docsExamined: 329", which is also the optimized results, means no extra index scan and document scan. The time of operation is 5 milliseconds.



4.2 Neo4j

4.1.1 Query for [SQ1]

- Query Requirement
  Find all users involved in a given question (identified by id) and their respective profiles including the creationDate, DisplayName, upVote, and DownVote.
- Query Design

```
query = "match (user:User)-[r:PROVIDE|ASK]-() where r.QuestionId="+str(question_id)+" return user"
```

Query Design	
Input Parameter	question_id (data type: int)
Output result	CreationDate, DisplayName, UpVotes, DownVotes

- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query sq1 -sq1 2
{'user': (_21013:User {CreationDate: '2016-08-02T15:38:21.100', DisplayName: 'Fr
anck Dernoncourt', DownVotes: '2', Id: 4, UpVotes: '15'})}
{'user': (_21048:User {CreationDate: '2017-05-31T09:46:16.857', DisplayName: 'Ts
hilidzi Mudau', DownVotes: '1', Id: 7550, UpVotes: '24'})}
{'user': (_21090:User {CreationDate: '2016-08-02T15:38:37.680', DisplayName: 'Ma
tthew Graves', DownVotes: '25', Id: 10, UpVotes: '64'})}
{'user': (_21126:User {CreationDate: '2016-08-02T15:38:36.723', DisplayName: 'ke
norb', DownVotes: '3', Id: 8, UpVotes: '559'})}
vlan-2631-10-19-6-0:src mistletoe$
```

- Query Performance Analysis
  The space performance:

```
+-----------------+---------------+--------------------+-----------------------------+
| Operator        | Estimated Rows | Variables          | Other                       |
+-----------------+---------------+--------------------+-----------------------------+
| +ProduceResults |           526 | anon[35], r, user  |                             |
| |               +---------------+--------------------+-----------------------------+
| +Filter         |           526 | anon[35], r, user  | r.QuestionId = $`  AUTOINT0` |
| |               +---------------+--------------------+-----------------------------+
| +Expand(All)    |          5263 | anon[35], r -- user | (user)-[r:PROVIDE|:ASK]-()  |
| |               +---------------+--------------------+-----------------------------+
| +NodeByLabelScan |         15541 | user               | :User                       |
+-----------------+---------------+--------------------+-----------------------------+
```

The time performance:

```
neo4j-sh (?)$ match (user:User)-[r:PROVIDE|ASK]-() where r.QuestionId=2 return user;
+-------------------------------------------------------------------------------------------------------------------------------+
| user                                                                                                                          |
+-------------------------------------------------------------------------------------------------------------------------------+
| Node[21013]{DisplayName:"Franck Dernoncourt",UpVotes:"15",DownVotes:"2",Id:4,CreationDate:"2016-08-02T15:38:21.100"}          |
| Node[21048]{DisplayName:"Tshilidzi Mudau",UpVotes:"24",DownVotes:"1",Id:7550,CreationDate:"2017-05-31T09:46:16.857"}          |
| Node[21090]{DisplayName:"Matthew Graves",UpVotes:"64",DownVotes:"25",Id:10,CreationDate:"2016-08-02T15:38:37.680"}            |
| Node[21126]{DisplayName:"kenorb",UpVotes:"559",DownVotes:"3",Id:8,CreationDate:"2016-08-02T15:38:36.723"}                     |
+-------------------------------------------------------------------------------------------------------------------------------+
4 rows
19 ms
```

We also designed another query before designing this query, as shown in the following figure.By comparison, we found that this query has higher space efficiency after adding the relevant index, but the time is not very well under given data.

```
0 rows available after 2 ms, consumed after another 0 ms
neo4j> explain match (user:User)-[:ASK|PROVIDE]->(p:Post) where p.Id=2 or p.ParentId=2 return user;
+-----------------------------------------------------------------------+
| Plan      | Statement   | Version      | Planner | Runtime   | Time |
+-----------------------------------------------------------------------+
| "EXPLAIN" | "READ_ONLY" | "CYPHER 3.4" | "COST"  | "SLOTTED" | 205  |
+-----------------------------------------------------------------------+

+------------------+----------------+------------------+-------------------------------------+
| Operator         | Estimated Rows | Identifiers      | Other                               |
+------------------+----------------+------------------+-------------------------------------+
| +ProduceResults  |              3 | anon[18], p, user | 3.4; 3.4                           |
| |                +----------------+------------------+-------------------------------------+
| +Filter          |              3 | anon[18], p, user | user:User                          |
| |                +----------------+------------------+-------------------------------------+
| +Expand(All)     |              3 | anon[18], p, user | (p)<-[anon[18]:ASK|:PROVIDE]-(user) |
| |                +----------------+------------------+-------------------------------------+
| +Distinct        |              3 | p                | p                                   |
| |                +----------------+------------------+-------------------------------------+
| +Union           |              1 | p                |                                     |
| |\               +----------------+------------------+-------------------------------------+
| | +NodeIndexSeek |              2 | p                | :Post(ParentId)                     |
| |                +----------------+------------------+-------------------------------------+
| +NodeIndexSeek   |              1 | p                | :Post(Id)                           |
+------------------+----------------+------------------+-------------------------------------+

0 rows available after 205 ms, consumed after another 0 ms
neo4j>
```

Thus, considering the real-time nature of OLTP, we chose the above query(shown as the “Query design” section).Although this query scan many user nodes, the entry of the query is set on the relationship instead of the node, neo4j has faster retrieval speed for the relationship, so the total query performance is also very good.

4.1.2 Query for [SQ2]

- Query Requirement
  Assuming each tag represents a topic, find the most viewed question in a given topic.
- Query Design

```
query = "match (question:Question)-[:TAGGED]->(tag:Tag{TagName:'"+\
        topic\
        +"'}) with max(question.ViewCount) as m match (question:Question{ViewCount:m}) return question"
```

Query Design	
Input Parameter	topic (data type: String)
Output Result	Id, ViewCount, OwnerUserId, Title

- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query sq2 -sq2 neu
ral-networks
{'question': (_392:Post:Question {Id: 1294, OwnerUserId: 144, Title: 'How does H
inton\'s "capsules theory" work?', ViewCount: 19807})}
vlan-2631-10-19-6-0:src mistletoe$
```

- Query Performance Analysis
  The space performance:
  Before using index:

```
Runtime version 3.4

+--------------------+----------------+--------------------------+---------------------------------+
| Operator           | Estimated Rows | Variables                | Other                           |
+--------------------+----------------+--------------------------+---------------------------------+
| +ProduceResults    |           5026 | question, m, question    |                                 |
| |                  +----------------+--------------------------+---------------------------------+
| +Projection        |           5026 | question -- question, m  | {question : question}           |
| |                  +----------------+--------------------------+---------------------------------+
| +Filter            |           5026 | question, m              | question.ViewCount = m          |
| |                  +----------------+--------------------------+---------------------------------+
| +Apply             |           5026 | question, m              |                                 |
| |\                 +----------------+--------------------------+---------------------------------+
| | +NodeByLabelScan |          50257 | question -- m            | :Question                       |
| |                  +----------------+--------------------------+---------------------------------+
| +EagerAggregation  |             23 | m                        |                                 |
| |                  +----------------+--------------------------+---------------------------------+
| +Filter            |            546 | anon[26], question, tag  | question:Question               |
| |                  +----------------+--------------------------+---------------------------------+
| +Expand(All)       |           1336 | anon[26], question -- tag | (tag)<-[:TAGGED]-(  question@7) |
| |                  +----------------+--------------------------+---------------------------------+
| +Filter            |             29 | tag                      | tag.TagName = $`  AUTOSTRING0`  |
| |                  +----------------+--------------------------+---------------------------------+
| +NodeByLabelScan   |            287 | tag                      | :Tag                            |
+--------------------+----------------+--------------------------+---------------------------------+
```

After using index:

```
+-------------------+----------------+--------------------------+---------------------------------+
| Operator          | Estimated Rows | Variables                | Other                           |
+-------------------+----------------+--------------------------+---------------------------------+
| +ProduceResults   |             15 | question, m, question    |                                 |
| |                 +----------------+--------------------------+---------------------------------+
| +Projection       |             15 | question -- question, m  | {question : question}           |
| |                 +----------------+--------------------------+---------------------------------+
| +Apply            |             15 | question, m              |                                 |
| |\                +----------------+--------------------------+---------------------------------+
| | +NodeIndexSeek  |             15 | question -- m            | :Question(ViewCount)            |
| |                 +----------------+--------------------------+---------------------------------+
| +EagerAggregation |              4 | m                        |                                 |
| |                 +----------------+--------------------------+---------------------------------+
| +Filter           |             19 | anon[26], question, tag  | question:Question               |
| |                 +----------------+--------------------------+---------------------------------+
| +Expand(All)      |             47 | anon[26], question -- tag | (tag)<-[:TAGGED]-(  question@7) |
| |                 +----------------+--------------------------+---------------------------------+
| +NodeIndexSeek    |              1 | tag                      | :Tag(TagName)                   |
+-------------------+----------------+--------------------------+---------------------------------+
```

The time performance:

```
neo4j-sh (?)$ match (question:Question)-[:TAGGED]->(tag:Tag{TagName:'neural-networks'}) with max(question.
+-------------------------------------------------------------------------------------------------------+
| question                                                                                              |
+-------------------------------------------------------------------------------------------------------+
| Node[392]{ViewCount:19807,Title:"How does Hinton's "capsules theory" work?",Id:1294,OwnerUserId:144} |
+-------------------------------------------------------------------------------------------------------+
1 row
7 ms
neo4j-sh (?)$
```

For this query, we use two match clauses and relevant conditions so in order to improve the query performance, we create two indexes on :Question and :Tag. As the above pictures show after using the index of :Tag(TagName) and :Question(ViewCount) the estimated rows dramatically decrease. Also, the Neo4j operator add the “NodeIndexSeek” stage which means Neo4j select the index to optimize the query performance. So the space performance is enhanced obviously by using indexes.

4.1.3 Query for [AQ3]

- Query Requirement
  Given a topic, find the champion user and all questions the user has answers accepted in that topic.
- Query Design

```
query = '''
match (a:Answer)<-[r:ACCEPT]-(q:Question)-[:TAGGED]->(t:Tag)
where t.TagName= '%(topic)s'
with a.OwnerUserId as user,count(r) as num with user,max(num) as max
order by max desc limit 1
with user
match (u:Answer)<-[r:ACCEPT{OwnerUserId:user}]-(q:Question)-[tag_r:TAGGED]->(t:Tag{TagName: '%(topic)s'})
return q.Id as Id,q.Title as Title
order by q.Id asc
'''
```

Query Design	
Input Parameter	topic (data type: String)
Output Result	Id, Title

- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query aq3 -aq3 dee
p-learning
{'Id': 3402, 'Title': 'Is there ever a need to combine deep learning frameworks?
 (Eg. Tensorflow & Torch)?'}
{'Id': 3453, 'Title': 'What are the pros and cons of using a spatial transformat
ion network to predict the next video frame?'}
{'Id': 4080, 'Title': 'Is it necessary to clear the replay memory regularly in a
 DQN when an agent plays against itself?'}
{'Id': 4085, 'Title': 'Policy gradients for multiple continuous actions'}
{'Id': 4167, 'Title': 'Reinforcement learning for robotic motion planning - Prob
lem statement ideas'}
{'Id': 4346, 'Title': 'Is the new Alpha Go implementation using Generative Adver
sarial Networks?'}
{'Id': 4425, 'Title': 'Deep Learning – Classification or Regression Approach?'}
{'Id': 4740, 'Title': 'I don\'t understand the "Target Network" in Deep Q-Learni
ng'}
{'Id': 5185, 'Title': 'What is the purpose of the GAN'}
```

- Query Performance Analysis

  The space performance:

  Before using index:

```
+--------------------+----------------+-------------------------------------------------------------------+-----------------------------------+
| Operator           | Estimated Rows | Variables                                                         | Other                             |
+--------------------+----------------+-------------------------------------------------------------------+-----------------------------------+
| +ProduceResults    |             21 | anon[345], anon[356], q, r, t, Id, Title, max, tag_r, u, user     |                                   |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Projection        |             21 | Id, Title -- anon[345], anon[356], q, r, t, max, tag_r, u, user   | {Id : anon[345], Title : anon[356]} |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Sort              |             21 | anon[345], anon[356], q, r, t, max, tag_r, u, user                | anon[345]                         |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Projection        |             21 | anon[345], anon[356] -- q, r, t, max, tag_r, u, user              | { : q.Id,  : q.Title}             |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Filter            |             21 | q, r, t, max, tag_r, u, user                                      | u:Answer; u.OwnerUserId = user    |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Expand(All)       |            206 | r, u -- q, t, max, tag_r, user                                    | (  q@269)-[  r@258:ACCEPT]->(u)   |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Filter            |            546 | q, t, max, tag_r, user                                            | q:Question                        |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Expand(All)       |           1336 | q, tag_r -- t, max, user                                          | (  t@298)<-[tag_r:TAGGED]-(  q@269) |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Filter            |             29 | t, max, user                                                      | t.TagName = $`  AUTOSTRING1`      |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Apply             |             21 | t, max, user                                                      |                                   |
| |\                 +----------------+-------------------------------------------------------------------+-----------------------------------+
| | +NodeByLabelScan |            287 | t -- max, user                                                    | :Tag                              |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Top               |              1 | max, user                                                         | max; 1                            |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +EagerAggregation  |              4 | max -- user                                                       | user                              |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +EagerAggregation  |             14 | num, user                                                         | user                              |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Filter            |            206 | anon[42], q, r, t, a                                              | a:Answer                          |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Expand(All)       |            206 | r, a -- anon[42], q, t                                            | (  q@30)-[  r@19:ACCEPT]->(a)     |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Filter            |            546 | anon[42], q, t                                                    | q:Question                        |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Expand(All)       |           1336 | anon[42], q -- t                                                  | (  t@54)<-[:TAGGED]-(  q@30)      |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +Filter            |             29 | t                                                                 | t.TagName = $`  AUTOSTRING0`      |
| |                  +----------------+-------------------------------------------------------------------+-----------------------------------+
| +NodeByLabelScan   |            287 | t                                                                 | :Tag                              |
+--------------------+----------------+-------------------------------------------------------------------+-----------------------------------+
```

After using index:

```
+-------------------+----------------+-----------------------------------------------------------------+-----------------------------------+
| Operator          | Estimated Rows | Variables                                                       | Other                             |
+-------------------+----------------+-----------------------------------------------------------------+-----------------------------------+
| +ProduceResults   |              1 | anon[345], anon[356], q, r, t, Id, Title, max, tag_r, u, user   |                                   |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Projection       |              1 | Id, Title -- anon[345], anon[356], q, r, t, max, tag_r, u, user | {Id : anon[345], Title : anon[356]} |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Sort             |              1 | anon[345], anon[356], q, r, t, max, tag_r, u, user              | anon[345]                         |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Projection       |              0 | anon[345], anon[356] -- q, r, t, max, tag_r, u, user            | { : q.Id,  : q.Title}             |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Filter           |              0 | q, r, t, max, tag_r, u, user                                    | t:Tag; t.TagName = $`  AUTOSTRING1` |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Expand(All)      |              3 | t, tag_r -- q, r, max, u, user                                  | (  q@269)-[tag_r:TAGGED]->(  t@298) |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Filter           |              1 | q, r, max, u, user                                              | q:Question                        |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Expand(All)      |              1 | q, r -- max, u, user                                            | (u)<-[  r@258:ACCEPT]-(  q@269)   |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Apply            |              0 | max, u, user                                                    |                                   |
| |\                +----------------+-----------------------------------------------------------------+-----------------------------------+
| | +NodeIndexSeek  |              4 | u -- max, user                                                  | :Answer(OwnerUserId)              |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Top              |              1 | max, user                                                       | max; 1                            |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +EagerAggregation |              2 | max -- user                                                     | user                              |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +EagerAggregation |              3 | num, user                                                       | user                              |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Filter           |              7 | anon[42], q, r, t, a                                            | a:Answer                          |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Expand(All)      |              7 | r, a -- anon[42], q, t                                          | (  q@30)-[  r@19:ACCEPT]->(a)     |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Filter           |             19 | anon[42], q, t                                                  | q:Question                        |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +Expand(All)      |             47 | anon[42], q -- t                                                | (  t@54)<-[:TAGGED]-(  q@30)      |
| |                 +----------------+-----------------------------------------------------------------+-----------------------------------+
| +NodeIndexSeek    |              1 | t                                                               | :Tag(TagName)                     |
+-------------------+----------------+-----------------------------------------------------------------+-----------------------------------+
```

The time performance:

```
$ match (a:Answer)<-[r:ACCEPT]-(q:Question)-[:TAGGED]->(t:Tag)  where t.TagName= 'deep-learning' with a.Owne…
```

Table | Text | Code

Id	Title
3402	"Is there ever a need to combine deep learning frameworks? (Eg. Tensorflow & Torch)?"
3453	"What are the pros and cons of using a spatial transformation network to predict the next video frame?"
4080	"Is it necessary to clear the replay memory regularly in a DQN when an agent plays against itself?"
4085	"Policy gradients for multiple continuous actions"
4167	"Reinforcement learning for robotic motion planning - Problem statement ideas"
4346	"Is the new Alpha Go implementation using Generative Adversarial Networks?"
4425	"Deep Learning – Classification or Regression Approach?"
4740	"I don't understand the "Target Network" in Deep Q-Learning"
5185	"What is the purpose of the GAN"

Started streaming 9 records after 8 ms and completed after 8 ms.

For this query, we use two match clauses and relevant conditions, so in order to improve the query performance, we create two indexes on :Answer and :Tag.
As the above pictures show after using the index of :Answer(OwnerUserId) and :Tag(TagName) the estimated rows dramatically decrease. The number of the rows estimated of the first "Expand(All)" reduces from 1336 to only 47 by using indexes. This is the most obvious stage that shows the consequence of performance optimization. Also, the space performance in other stages are enhanced to a large extent.

4.1.4 Query for [AQ5]

- Query Requirement
  Discover questions with arguable accepted answer.
- Query Design

```
query = '''
match (vq:Vote{VoteTypeId:2})-[rq:VOTE]->(q:Question)-[:ACCEPT]->()
with q.Id as q_id,count(rq) as q_num
where q_num > %(alpha)s
with q_id
match (vac:Vote{VoteTypeId:2})-[rac:VOTE]-(aac:Answer)<-[:ACCEPT]-(:Question{Id:q_id})
with q_id,aac.Id as aac_id,count(rac) as rac_num
match (a:Answer{ParentId:q_id})-[ac:VOTE]-(:Vote{VoteTypeId:2})
with q_id,a.Id as a_id,count(ac) as ac_num,rac_num
with q_id,max(ac_num) as ac_num,rac_num
where ac_num > rac_num
return q_id,ac_num as numOfMaxAnswerVote,rac_num as numOfAcceptedAnswerVote
'''
```

Query Design	
Input Parameter	alpha (data type: int)
Output result	questionId, numOfMaxAnswerVote, numOfAcceptedAnswerVote

- Query Execution

```
MACdeMacBook-Pro-3:src mistletoe$ python3 main.py -type neo -query aq5 -aq5 15
{'q_id': 248, 'numOfMaxAnswerVote': 4, 'numOfAcceptedAnswerVote': 3}
MACdeMacBook-Pro-3:src mistletoe$
```

- Query Performance Analysis

  The space performance:

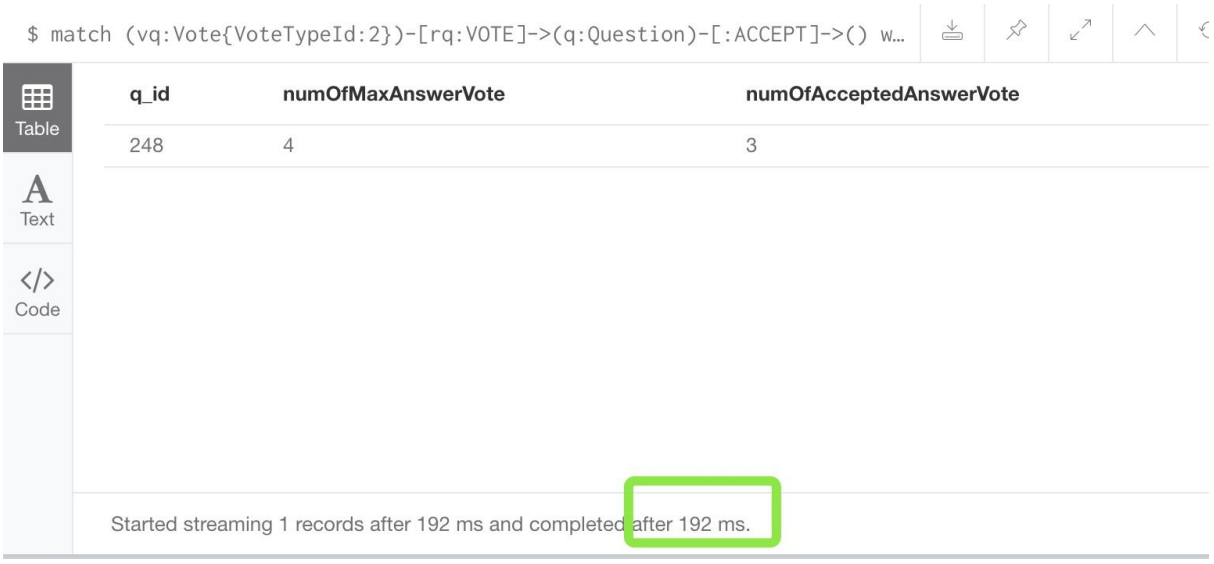
  Before using index:

```
+------------------+----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| Operator         | Estimated Rows | Identifiers                                                                     | Other                                                                        |
+------------------+----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +ProduceResults  |              5 | rac_num, q_id, anon[535], numOfMaxAnswerVote, ac_num, numOfAcceptedAnswerVote   | 3.4; 3.4                                                                     |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Projection      |              5 | rac_num, q_id, anon[535], numOfMaxAnswerVote, ac_num, numOfAcceptedAnswerVote   | {q_id : q_id, numOfMaxAnswerVote : ac_num, numOfAcceptedAnswerVote : rac_num} |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |              5 | anon[535], ac_num, q_id, rac_num                                                | anon[535]                                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Projection      |              6 | anon[535], ac_num, q_id, rac_num                                                | {q_id : q_id,   ac_num@496 : ac_num, rac_num : rac_num,  : ac_num > rac_num}  |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |             6 | ac_num, q_id, rac_num                                                           | q_id, rac_num                                                                |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |            40 | ac_num, a_id, q_id, rac_num                                                     | q_id, a_id, rac_num                                                          |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |           1620 | rac_num, q_id, a, ac, aac_id, anon[377]                                         | anon[377].VoteTypeId = $`  AUTOINT3`                                         |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |          16196 | rac_num, q_id, a, ac, aac_id, anon[377]                                         | (a)-[ac:VOTE]-(anon[377])                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |           5821 | a, aac_id, q_id, rac_num                                                        | a.ParentId = q_id                                                            |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Apply           |           1620 | a, aac_id, q_id, rac_num                                                        |                                                                              |
| |\               +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| | +NodeByLabelScan |        58210 | a, aac_id, q_id, rac_num                                                        | :Answer                                                                      |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |            19 | aac_id, q_id, rac_num                                                           | q_id, aac_id                                                                 |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |            346 | rac, q_id, anon[139], aac, q_num, vac, anon[231], anon[243]                     | vac.VoteTypeId = $`  AUTOINT2`                                               |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |           3461 | rac, q_id, anon[139], aac, q_num, vac, anon[231], anon[243]                     | (aac)-[rac:VOTE]-(vac)                                                       |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |           1244 | q_id, anon[139], aac, q_num, anon[231], anon[243]                               | aac:Answer                                                                   |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |           1244 | q_id, anon[139], aac, q_num, anon[231], anon[243]                               | (anon[243])-[anon[231]:ACCEPT]->(aac)                                        |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |           3295 | anon[139], anon[243], q_id, q_num                                               | anon[139]                                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Apply           |            346 | anon[139], anon[243], q_id, q_num                                               |                                                                              |
| |\               +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| | +NodeByLabelScan |        43934 | anon[139], anon[243], q_id, q_num                                               | :Question                                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Projection      |             20 | anon[139], q_id, q_num                                                          | {q_id : q_id, q_num : q_num,  : q_num > $`  AUTOINT1`}                       |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |            20 | q_id, q_num                                                                     | q_id                                                                         |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |            417 | vq, rq, q, anon[54], anon[66]                                                   | vq.VoteTypeId = $`  AUTOINT0`                                                |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |           4172 | vq, rq, q, anon[54], anon[66]                                                   | (q)<-[rq:VOTE]-(vq)                                                          |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |            812 | anon[54], anon[66], q                                                           | (q)-[anon[54]:ACCEPT]->(anon[66])                                            |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +NodeByLabelScan |           2151 | q                                                                               | :Question                                                                    |
+------------------+----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
```

  After using index:

```
+------------------+----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| Operator         | Estimated Rows | Identifiers                                                                     | Other                                                                        |
+------------------+----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +ProduceResults  |              1 | ac_num, rac_num, q_id, anon[472], numOfMaxAnswerVote, numOfAcceptedAnswerVote   | 3.4; 3.4                                                                     |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Projection      |              1 | ac_num, rac_num, q_id, anon[472], numOfMaxAnswerVote, numOfAcceptedAnswerVote   | {q_id : q_id, numOfMaxAnswerVote : ac_num, numOfAcceptedAnswerVote : rac_num} |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |              1 | anon[472], ac_num, q_id, rac_num                                                | anon[472]                                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Projection      |              1 | anon[472], ac_num, q_id, rac_num                                                | {q_id : q_id,   ac_num@440 : ac_num, rac_num : rac_num,  : ac_num > rac_num}  |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |             1 | ac_num, q_id, rac_num                                                           | q_id, rac_num                                                                |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |             0 | ac_num, a_id, q_id, rac_num                                                     | q_id, a_id, rac_num                                                          |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |              0 | rac_num, q_id, anon[335], a, ac, aac_id                                         | anon[335].VoteTypeId = $`  AUTOINT3`                                         |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |              5 | rac_num, q_id, anon[335], a, ac, aac_id                                         | (a)-[ac:VOTE]-(anon[335])                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Apply           |              0 | a, aac_id, q_id, rac_num                                                        |                                                                              |
| |\               +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| | +NodeIndexSeek |              2 | a, aac_id, q_id, rac_num                                                        | :Answer(ParentId)                                                            |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |             1 | aac_id, q_id, rac_num                                                           | q_id, aac_id                                                                 |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |              1 | rac, q_id, anon[125], aac, anon[215], q_num, anon[203], vac                     | vac.VoteTypeId = $`  AUTOINT2`                                               |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |             15 | rac, q_id, anon[125], aac, anon[215], q_num, anon[203], vac                     | (aac)-[rac:VOTE]-(vac)                                                       |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |              5 | q_id, anon[125], aac, anon[215], q_num, anon[203]                               | aac:Answer                                                                   |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |              5 | q_id, anon[125], aac, anon[215], q_num, anon[203]                               | (anon[215])-[anon[203]:ACCEPT]->(aac)                                        |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |             14 | anon[125], anon[215], q_id, q_num                                               | anon[125]                                                                    |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Apply           |              1 | anon[125], anon[215], q_id, q_num                                               |                                                                              |
| |\               +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| | +NodeIndexSeek |             19 | anon[125], anon[215], q_id, q_num                                               | :Question(Id)                                                                |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Projection      |             19 | anon[125], q_id, q_num                                                          | {q_id : q_id, q_num : q_num,  : q_num > $`  AUTOINT1`}                       |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +EagerAggregation |            19 | q_id, q_num                                                                     | q_id                                                                         |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |            348 | vq, rq, q, anon[54], anon[66]                                                   | (q)-[anon[54]:ACCEPT]->(anon[66])                                            |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Filter          |            921 | q, rq, vq                                                                       | q:Question                                                                   |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +Expand(All)     |           1646 | q, rq, vq                                                                       | (vq)-[rq:VOTE]->(q)                                                          |
| |                +----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
| +NodeIndexSeek   |           1862 | vq                                                                              | :Vote(VoteTypeId)                                                            |
+------------------+----------------+---------------------------------------------------------------------------------+------------------------------------------------------------------------------+
```

The time performance:



For this query, we use the indexes of :Vote(VoteTypeId), :Answer(ParentId) and :Question(Id) to optimize the performance. Before using the indexes the query will scan 2151 nodes when the first match and where clauses is executed, this operation scan all question nodes which is less efficient. After using index of :Vote(VoteTypeId), the first match and where clauses only scan 1862 vote nodes. The result of the query optimization is enhance more obviously in the second match and where clauses, the number of estimated rows decrease from 43934 to only 19, and the third one from 58210 to only 2. Thus, the performance is enhanced totally.

4.1.5 Query for [AQ6]

- Query Requirement
  Discover the top five coauthors of a given user.
- Query Design

```
query = "match (a:Post)-[b:COOPERATE|ANSWER]->(c:Post) where a.OwnerUserId="+str(user_id)+\
        " return c.OwnerUserId as userId,count(c) as num order by num desc limit 5"
```

Query Design	
Input Parameter	user_id (data type: int)
Output result	userId, num

- Query Execution

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query aq6 -aq6 439
8
{'userId': 1671, 'num': 5}
{'userId': 9161, 'num': 4}
{'userId': 11571, 'num': 4}
{'userId': 4302, 'num': 3}
{'userId': 6019, 'num': 3}
```

- Query Performance Analysis

  The space performance:

  Before using index:

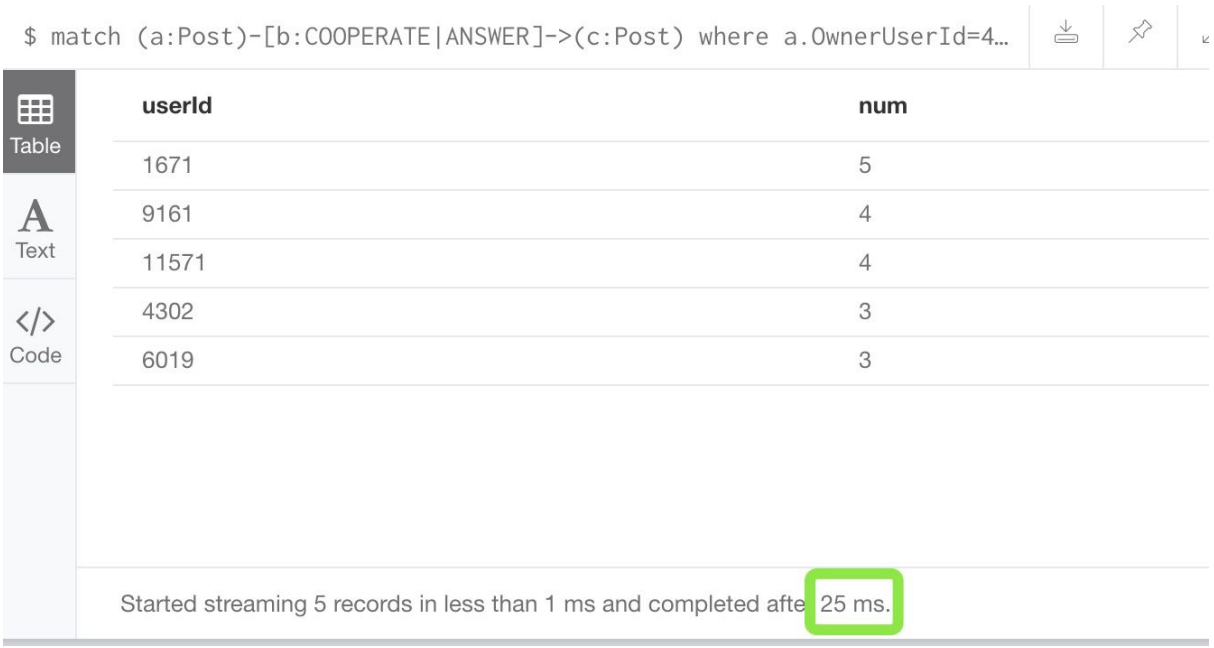
```
+------------------+----------------+----------------------------------+----------------------------------------+
| Operator         | Estimated Rows | Variables                        | Other                                  |
+------------------+----------------+----------------------------------+----------------------------------------+
| +ProduceResults  |              5 | anon[102], anon[80], num, userId |                                        |
| |                +----------------+----------------------------------+----------------------------------------+
| +Projection      |              5 | num, userId -- anon[102], anon[80] | {userId : `anon[80]`, num : `anon[102]`} |
| |                +----------------+----------------------------------+----------------------------------------+
| +Top             |              5 | anon[102], anon[80]              | anon[102]; 5                           |
| |                +----------------+----------------------------------+----------------------------------------+
| +EagerAggregation |             29 | anon[102], anon[80]              | anon[80]                               |
| |                +----------------+----------------------------------+----------------------------------------+
| +Filter          |            828 | a, b, c                          | c:Post                                 |
| |                +----------------+----------------------------------+----------------------------------------+
| +Expand(All)     |            828 | b, c -- a                        | (a)-[b:COOPERATE|:ANSWER]->(c)         |
| |                +----------------+----------------------------------+----------------------------------------+
| +Filter          |            528 | a                                | a.OwnerUserId = $`  AUTOINT0`          |
| |                +----------------+----------------------------------+----------------------------------------+
| +NodeByLabelScan |           5280 | a                                | :Post                                  |
+------------------+----------------+----------------------------------+----------------------------------------+
```

  After using index:

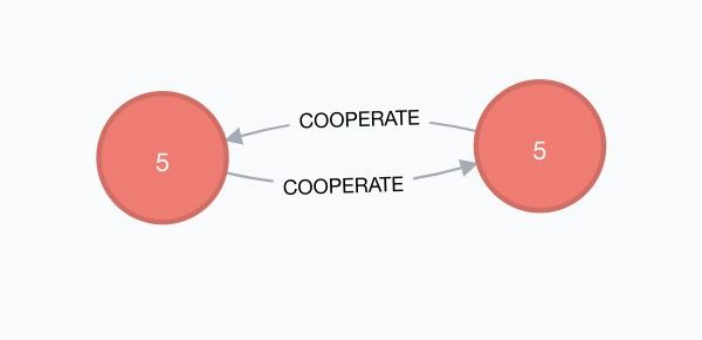
```
Runtime version 3.4

+------------------+----------------+----------------------------------+----------------------------------------+
| Operator         | Estimated Rows | Variables                        | Other                                  |
+------------------+----------------+----------------------------------+----------------------------------------+
| +ProduceResults  |              2 | anon[102], anon[80], num, userId |                                        |
| |                +----------------+----------------------------------+----------------------------------------+
| +Projection      |              2 | num, userId -- anon[102], anon[80] | {userId : `anon[80]`, num : `anon[102]`} |
| |                +----------------+----------------------------------+----------------------------------------+
| +Top             |              2 | anon[102], anon[80]              | anon[102]; 5                           |
| |                +----------------+----------------------------------+----------------------------------------+
| +EagerAggregation |             2 | anon[102], anon[80]              | anon[80]                               |
| |                +----------------+----------------------------------+----------------------------------------+
| +Filter          |              4 | a, b, c                          | c:Post                                 |
| |                +----------------+----------------------------------+----------------------------------------+
| +Expand(All)     |              4 | b, c -- a                        | (a)-[b:COOPERATE|:ANSWER]->(c)         |
| |                +----------------+----------------------------------+----------------------------------------+
| +NodeIndexSeek   |              3 | a                                | :Post(OwnerUserId)                     |
+------------------+----------------+----------------------------------+----------------------------------------+
```

The time performance:



For this query, the query is quite easy and efficient, since we created the relationship that is :COOPERATE between among answer nodes. This relationship can connect the answers that answer the same question, picture below shows this relationship.



Also, the:ANSWER relationship can find the coauthors between question and answer nodes. Moreover, we define that both answer and question nodes are post nodes.
To further optimize the performance, we create the index of :Post(OwnerUserId), as can be seen from the above space performance pictures, the estimated rows remarkably decrease from 5280 to only 3 by using index.

5.3 Comparison

5.3.1 Schema Difference

Both MongoDB and Neo4j are NoSQL databases. The former is schema-less while the latter is categorized under the categorized under the graph databases. The following is the details about the difference of the MongoDB schema and Neo4j schema.

MongoDB is a document store to store BSON documents with regard to fast write and read, and there is no relationships stored in MongoDB. Thus, MongoDB allows us to store and retrieve blocks of data very quickly, but does not store relationships. However, Neo4j allows us to store the data having the complex relationships, and those data are stored as graphs. If the user considers the graph having highly relationship data or needs navigation, Neo4j could be a better choice. For instance, for this assignment, the query AQ5 has the complicated relationships between vote, question and answer. Thus, if use MongoDB to realize this query, we need more than one join stages ($lookup stage), it could lead to the huge Cartesian product produced which is not efficient. But Neo4j are more suitable for deal with complex relationships among data, so we choose Neo4j to realize AQ5. The same reason that chooses AQ3 for Neo4j. As for AQ6, this query aims to capture the interpersonal network between users, so this interpersonal network is also more suitable for Neo4j to store and for the further query.

5.3.2 Performance Comparison

MongoDB is suitable for more complex aggregate queries, and MongoDB has more aggregation stages for aggregate queries and MongoDB are more suitable for retrieving dates. It supports high scalability of various data collections, which can store data easily. Thus, for this assignment, we choose AQ1, AQ2, AQ4 these three queries to use MongoDB due to the high complexity of the aggregation stages. Moreover, Neo4j more supports to query the data with the complicated relationships, due to using the Graph store which is more suitable for query based on traversing graph paths.(like AQ3,AQ5,AQ6).

Moreover, we compared the performance of SQ1 on MongoDB and Neo4j. We found that SQ1 is more suitable for the MongoDB schema we designed. The time performance and space performance of SQ1 are better presented in MongoDB. However, SQ2 is more suitable for Neo4j's schema design, and has higher space efficiency though it has the almost the same time performance as MongoDB.

5. User Manual

The below query results are based on Python 3 and the version 3.4 of both MongoDB and Neo4j.

You can use the command line to conduct a query. The syntax as the below:

Command Line Instruction		
-type	indicate **database**	eg. mongo, neo
-query	indicate **specific query**	eg. sq1, aq1
-sq1	query for sq1	question id
-sq2	query for sq1	topic (tag)
-aq1	query for aq1	topic list
-aq2	query for aq2	start_date end_date
-aq3	query for aq3	topic (tag)
-aq4	query for aq4	user_id, alpha, date
-aq5	query for aq5	alpha
-aq6	query for aq6	user_id

For instance:
The below command is to execute query of aq4

```
>>python3 main.py -type mongo -query aq4 -aq4 4398 4 2018-08-30T00:00:00
```

5.1.1 Query for [SQ1]

- MongoDB

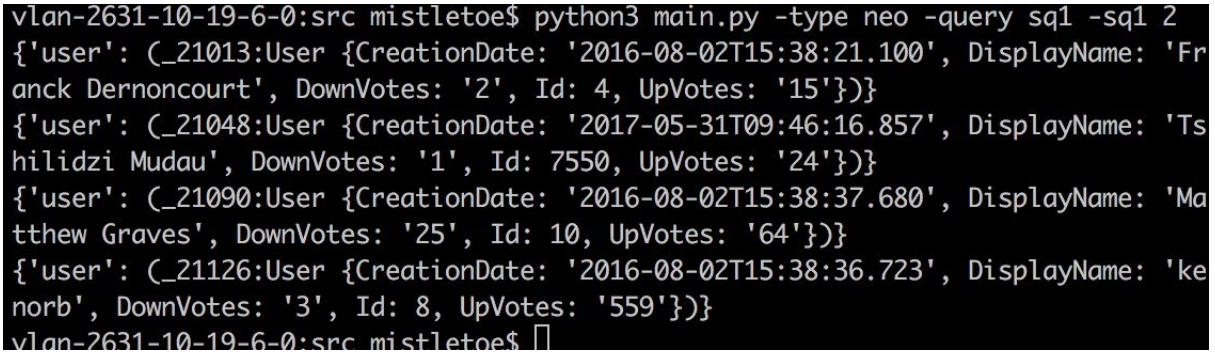
Query Instruction	
Input Parameter	question_id (data type: int)
Output result	CreationDate, DisplayName, UpVotes, DownVotes

Sample Query Result:

- Neo4j

Query Instruction	
Input Parameter	question_id (data type: int)
Output result	CreationDate, DisplayName, UpVotes, DownVotes

Sample Query Result:



5.1.2 Query for [SQ2]

- MongoDB

Query Instruction	
Input Parameter	topic (data type: String)
Output result	_id, Id,PostTypeId, AcceptedAnswerId, CreationDate, ViewCount, OwnerUserId, Title, Tags, AcceptedAuthorId

Sample Query Result:

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type mongo -query sq2 -sq2 n
eural-networks
{'_id': ObjectId('5baf3bce11ae17bbac4d60df'), 'Id': 1294, 'PostTypeId': 1, 'Acce
ptedAnswerId': 1313, 'CreationDate': datetime.datetime(2016, 8, 4, 8, 28, 6, 277
000), 'ViewCount': 19807, 'OwnerUserId': 144, 'Title': 'How does Hinton\'s "caps
ules theory" work?', 'Tags': ['neural-networks'], 'AcceptedAuthorId': 10}
vlan-2631-10-19-6-0:src mistletoe$
```

- Neo4j

Query Instruction	
Input Parameter	topic (data type: String)
Output Result	Id, ViewCount, OwnerUserId, Title

Sample Query Result:

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query sq2 -sq2 neu
ral-networks
{'question': (_392:Post:Question {Id: 1294, OwnerUserId: 144, Title: 'How does H
inton\'s "capsules theory" work?', ViewCount: 19807})}
vlan-2631-10-19-6-0:src mistletoe$
```

5.1.4 Query for [AQ1]

Query Instruction	
Input Parameter	*topics (data type: String)
Output Result	Id, Title, Interval

Sample Query Result:

```
vlan-2631-10-19-0-130:src mistletoe$  python3 main.py -type mongo -query aq1 -aq1 neural-networks
[{'Id': 1, 'Title': 'What is "backprop"?', 'Interval': 69873}]
vlan-2631-10-19-0-130:src mistletoe$
```

5.1.4 Query for [AQ2]

Query Instruction	
Input Parameter	start_date (data type: String), end_date (date type: String)
Output result	Topic, Num

Sample Query Result:

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type mongo -query aq2 -aq2 2
018-08-01T00:00:00 2018-08-31T00:00:00
{'Topic': 'neural-networks', 'Num': 65}
{'Topic': 'machine-learning', 'Num': 44}
{'Topic': 'deep-learning', 'Num': 39}
{'Topic': 'reinforcement-learning', 'Num': 28}
{'Topic': 'convolutional-neural-networks', 'Num': 24}
vlan-2631-10-19-6-0:src mistletoe$
```

5.1.4 Query for [AQ3]

Query Instruction	
Input Parameter	topic (data type: String)
Output Result	Id, Title

Sample Query Result:

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query aq3 -aq3 dee
p-learning
{'Id': 3402, 'Title': 'Is there ever a need to combine deep learning frameworks?
 (Eg. Tensorflow & Torch)?'}
{'Id': 3453, 'Title': 'What are the pros and cons of using a spatial transformat
ion network to predict the next video frame?'}
{'Id': 4080, 'Title': 'Is it necessary to clear the replay memory regularly in a
 DQN when an agent plays against itself?'}
{'Id': 4085, 'Title': 'Policy gradients for multiple continuous actions'}
{'Id': 4167, 'Title': 'Reinforcement learning for robotic motion planning - Prob
lem statement ideas'}
{'Id': 4346, 'Title': 'Is the new Alpha Go implementation using Generative Adver
sarial Networks?'}
{'Id': 4425, 'Title': 'Deep Learning – Classification or Regression Approach?'}
{'Id': 4740, 'Title': 'I don\'t understand the "Target Network" in Deep Q-Learni
ng'}
{'Id': 5185, 'Title': 'What is the purpose of the GAN'}
```

5.1.3 Query for [AQ4]

Query Instruction	
Input Parameter	user_id (data type: int), alpha (data type: int), date (data type: String)
Output result	Id, Title, CreationDate

Sample Query Result:

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type mongo -query aq4 -aq4 4
398 4 2018-08-30T00:00:00
{'Id': 7755, 'CreationDate': datetime.datetime(2018, 8, 29, 16, 4, 16, 113000),
'Title': 'How to implement a constrained action space in reinforcement learning?
'}
{'Id': 7737, 'CreationDate': datetime.datetime(2018, 8, 28, 0, 17, 55, 907000),
'Title': 'creating application to transform human computer experience into physi
cal activity'}
{'Id': 7736, 'CreationDate': datetime.datetime(2018, 8, 27, 18, 41, 56, 223000),
 'Title': 'In imitation learning, do you simply inject optimal (state, action, r
eward, s(t+1)) experiences into your experience replay buffer?'}
{'Id': 7734, 'CreationDate': datetime.datetime(2018, 8, 27, 16, 13, 16, 433000),
 'Title': 'AI composing music'}
{'Id': 7727, 'CreationDate': datetime.datetime(2018, 8, 27, 10, 18, 17, 893000),
 'Title': 'How is it possible to teach a neural network to perform addition?'}
vlan-2631-10-19-6-0:src mistletoe$
```

5.1.4 Query for [AQ5]

Query Instruction	
Input Parameter	alpha (data type: int)
Output result	questionId, numOfMaxAnswerVote, numOfAcceptedAnswerVote

Sample Query Result:

```
MACdeMacBook-Pro-3:src mistletoe$ python3 main.py -type neo -query aq5 -aq5 15
{'q_id': 248, 'numOfMaxAnswerVote': 4, 'numOfAcceptedAnswerVote': 3}
MACdeMacBook-Pro-3:src mistletoe$
```

5.1.5 Query for [AQ6]

Query Instruction	
Input Parameter	user_id (data type: int)
Output result	userId, num

Sample Query Result:

```
vlan-2631-10-19-6-0:src mistletoe$ python3 main.py -type neo -query aq6 -aq6 439
8
{'userId': 1671, 'num': 5}
{'userId': 9161, 'num': 4}
{'userId': 11571, 'num': 4}
{'userId': 4302, 'num': 3}
{'userId': 6019, 'num': 3}
```

6.Contribution Specification

		weight	Camille	Leal	TOTAL
MongoDB	schema design	10%	35%	65%	100%
	query design	10%	15%	85%	100%
	performance optimization	15%	60%	40%	100%
Neo4j	schema design	10%	20%	80%	100%
	query design	10%	15%	85%	100%
	performance optimization	15%	50%	50%	100%
Execution Script (Python)		5%	0%	100%	100%
Report		25%	60%	40%	100%
TOTAL		100%	40.0%	60.0%	100.0%

7.References

Aggregation Pipeline Optimization. (2018). Retrieved from https://docs.mongodb.com/v3.4/core/aggregation-pipeline-optimization/

Mior, M., Salem, K., Aboulnaga, A., & Liu, R. (2017). NoSE: Schema Design for NoSQL Applications. *IEEE Transactions On Knowledge And Data Engineering*, *29*(10), 2275-2289. doi:10.1109/tkde.2017.2722412