

# *FIT5137 Assignment 2: Neo4j*



*Student Name:* PEIYU LIU

*Student ID:* [REDACTED]

*Email:* [REDACTED]tudent.monash.edu

## ASSESSMENT COVER SHEET

<b>Student ID number</b>	Unit Name and Code:		FIT5137 Advanced database technology	
	Campus:		Clayton	
	Assignment Title:		FIT5137 Assignment 2: Neo4j	
	Name of Lecturer:		Dr. Agnes Haryanto	
	Name of Tutor:		Hongli Song/Shuyi Sun	
	Tutorial Day and Time:		Friday 10 am - 12 pm (Lab 02 )	
	Phone Number:		0452288954	
	Email Address:		pliu0030@student.monash.edu	
<b>Given Name</b>	Has any part of this assignment been previously submitted as part of another unit/course? <input type="checkbox"/> Yes <input checked="" type="checkbox"/> No			
	Due Date:		Week 12, Wednesday, 20-October-2021, 11:55 pm	Date Submitted: 19-October-2021
	<p>All work must be submitted by the due date. If an extension of work is granted this must be specified with the signature of the lecturer/tutor.</p> <p>Extension granted until (date) _____ Signature of lecturer/tutor _____</p> <p>Please note that it is your responsibility to retain copies of your assessments.</p>			
<b>Family name</b>	<p><b><i>Intentional plagiarism or collusion amounts to cheating under Part 7 of the Monash University (Council) Regulations</i></b></p> <p><b>Plagiarism:</b> Plagiarism means taking and using another person's ideas or manner of expressing them and passing them off as one's own. For example, by failing to give appropriate acknowledgement. The material used can be from any source (staff, students or the internet, published and unpublished works).</p> <p><b>Collusion:</b> Collusion means unauthorised collaboration with another person on assessable written, oral or practical work and includes paying another person to complete all or part of the work.</p> <p>Where there are reasonable grounds for believing that intentional plagiarism or collusion has occurred, this will be reported to the Associate Dean (Education) or delegate, who may disallow the work concerned by prohibiting assessment or refer the matter to the Faculty Discipline Panel for a hearing.</p>			
	<p><b>Student Statement:</b></p> <ul style="list-style-type: none"> <li>• I have read the university's Student Academic Integrity <a href="#">Policy</a> and <a href="#">Procedures</a>.</li> <li>• I understand the consequences of engaging in plagiarism and collusion as described in Part 7 of the Monash University (Council) Regulations <a href="http://adm.monash.edu/legal/legislation/statutes">http://adm.monash.edu/legal/legislation/statutes</a></li> <li>• have taken proper care to safeguard this work and made all reasonable efforts to ensure it could not be copied.</li> <li>• No part of this assignment has been previously submitted as part of another unit/course.</li> <li>• I acknowledge and agree that the assessor of this assignment may for the purposes of assessment, reproduce the assignment and:             <ul style="list-style-type: none"> <li>i. provide to another member of faculty and any external marker; and/or</li> <li>ii. submit it to a text matching software; and/or</li> <li>iii. submit it to a text matching software which may then retain a copy of the assignment on its database for the purpose of future plagiarism checking.</li> </ul> </li> <li>• I certify that I have not plagiarised the work of others or participated in unauthorised collaboration when preparing this assignment.</li> </ul> <p>Signature <u>PEIYU LIU</u> Date <u>2021/10/18</u></p> <p>* delete (iii) if not applicable</p>			

The information on this form is collected for the primary purpose of assessing your assignment and ensuring the academic integrity requirements of the University are met. Other purposes of collection include recording your plagiarism and collusion declaration, attending to course and administrative matters and statistical analyses. If you choose not to complete all the questions on this form it may not be possible for Monash University to assess your assignment. You have a right to access personal information that Monash University holds about you, subject to any exceptions in relevant legislation. If you wish to seek access to your personal information or inquire about the handling of your personal information, please contact the University Privacy Officer: [privacyofficer@adm.monash.edu.au](mailto:privacyofficer@adm.monash.edu.au)

# Table of content

<b>Table of content</b>	<b>3</b>
<b>Tasks</b>	<b>4</b>
C.1. Database Design	4
• Create a project MASL.	4
• Create a MASLgraph.	4
• Potential nodes and edges.	4
• Import data from the CSV files.	5
• Design explanations on the graph.	6
C.2. Queries	7
0. Pre. create Indexes	7
1. Query 1	7
2. Query 2	8
3. Query 3	8
4. Query 4	9
5. Query 5	9
6. Query 6	10
7. Query 7	10
8. Query 8	11
9. Query 9	11
10. Query 10	12
C.3. Database Modifications	13
1. Modification 1	13
2. Modification 2	14
3. Modification 3	15
C.4. Connecting to Drivers	17
*connect to graph database	17
*queries	18
*Database modification	21

# Tasks

## C.1. Database Design

- Create a project MASL.

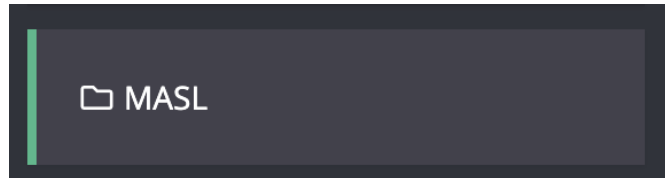


Figure 1: Project

- Create a MASLgraph.

## MASL



Figure 2: Graph

- Potential nodes and edges.

★ Potential nodes:

- UFO
- YEAR
- DAY
- MONTH
- HOUR
- SHAPE
- WEATHER\_CONDITION
- WIND\_DIRECTION
- LOCATION(city, state, countyName, longitude, latitude)

★ Potential relationships:

- (UFO)-[:IS\_HAPPENED\_IN]->(LOCATION)
- (UFO)-[:IS\_SHAPE\_OF]->(SHAPE)
- (UFO)-[:IS\_WEATHER\_CONDITION\_OF]->(WEATHER\_CONDITION)
- (UFO)-[:IS\_WEATHER\_DIRECTION\_OF]->(WEATHER\_DIRECTION)
- (UFO)-[:IN\_MONTH\_OF]->(MONTH)
- (UFO)-[:IN\_YEAR\_OF]->(YEAR)
- (UFO)-[:IN\_DAY\_OF]->(DAY)

➤ (UFO) -[:IN\_HOUR\_OF]-> (HOUR)

- Import data from the CSV files.

★ Completed code please check in TaskC1.cypher

```
//Import state
LOAD CSV WITH HEADERS FROM 'file:///states_a2.csv' AS data
WITH data
// check null value
WHERE data.city IS NOT NULL
// set node with properties[city,county,state,lat,lng]
// ensure data format is right using toUpper and toFloat
MERGE(location:Location{
    city:toUpper(data.city),
    lat:toFloat(data.lat),
    lng:toFloat(data.lng),
    countyName:toUpper(data.countyName),
    state:toUpper(data.state)});
```

Figure 3: Import state\_a2.csv

```
// Import ufo
LOAD CSV WITH HEADERS FROM 'file:///ufo_a2.csv' AS data
WITH data
// Create nodes for each row in ufo.csv
CREATE(u:UFO{
})
// give conditions to check null value: CASE X WHEN NULL THEN NULL ELSE X
// match data value with csv file, `` to avoid space between variables' name
SET u.duration = (CASE data.`duration` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE trim(data.`duration`)END)
// trim to remove empty space in front of string value or in the end of string value
SET u.text = (CASE data.`text` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE trim(data.`text`)END)
SET u.summary = (CASE data.`summary` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE trim(data.`summary`)END)
// ensure float and integer format when I set value to my properties
SET u.pressure = (CASE toFloat(data.`pressure`) WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`pressure`)END)
SET u.temp = (CASE toFloat(data.`temp`) WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`temp`)END)
SET u.hail = (CASE toInteger(data.`hail`) WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toInteger(data.`hail`)END)
SET u.heatindex = (CASE toFloat(data.`heatindex`) WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`heatindex`)END)
SET u.windchill = (CASE data.`windchill` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`windchill`)END)
SET u.rain = (CASE data.`rain` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toInteger(data.`rain`)END)
SET u.vis = (CASE data.`vis` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`vis`)END)
SET u.dewpt = ((CASE data.`dewpt` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`dewpt`)END))
SET u.thunder = (CASE data.`thunder` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toInteger(data.`thunder`)END)
SET u.fog = (CASE data.`fog` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toInteger(data.`fog`)END)
SET u.precip= (CASE data.`precip` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`precip`)END)
SET u.wspd= (CASE data.`wspd` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`wspd`)END)
SET u.tornado= (CASE data.`tornado` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toInteger(data.`tornado`)END)
SET u.hum= (CASE data.`hum` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`hum`)END)
SET u.snow= (CASE data.`snow` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toInteger(data.`snow`)END)
SET u.wgust= (CASE data.`wgust` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toFloat(data.`wgust`)END)
SET u.city= (CASE data.`city` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toUpper(data.`city`)END)
SET u.state= (CASE data.`state` WHEN "" THEN NULL WHEN "NA" THEN NULL ELSE toUpper(data.`state`)END)
// traverse each row and unwind shape as individual node for query's convenience.
// CASE WHEN ELSE END to deal with null value.
// Create relationships with UFO
FOREACH(IGNOREME IN CASE data.`shape` WHEN NULL THEN NULL WHEN "NA" THEN NULL ELSE toUpper(trim(data.`shape`)) END)
    MERGE (s:Shape {shape:toUpper(trim(data.`shape`))})
    MERGE (u) -[:IS_SHAPE_OF]-> (s)
)
// traverse each row and unwind weahter condition as individual node for query's convenience.
// CASE WHEN ELSE END to deal with null value.
// Create relationships with UFO
FOREACH(IGNOREME IN CASE data.`conds` WHEN NULL THEN NULL WHEN "NA" THEN NULL ELSE toUpper(trim(data.`conds`)) END)
    MERGE (c:Conds {condition:toUpper(trim(data.`conds`))})
    MERGE (u) -[:IS_WEAHTER_CONDITION_OF]-> (c)
)
```

Figure 4: Part of importing ufo\_a2.csv

- Design explanations on the graph.

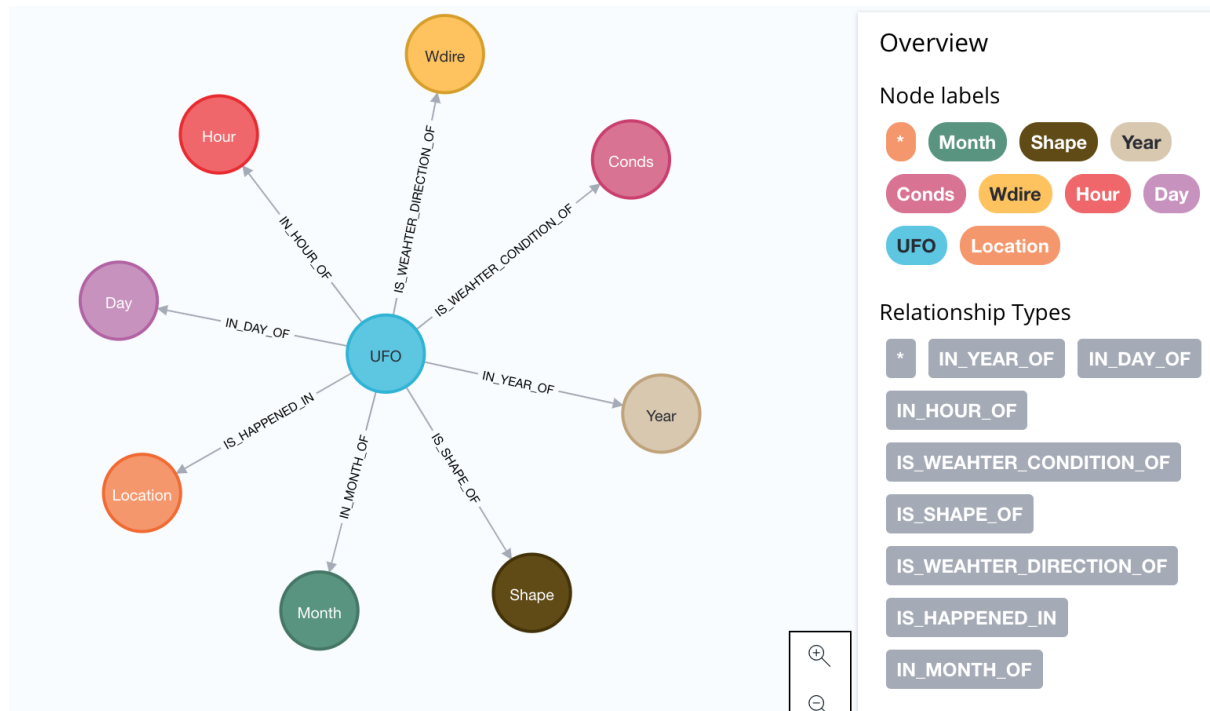


Figure 5: Graph schema of my database design

Figure 5 is the screenshot of my graph database schema(CALL db.schema.visualization). I have 9 nodes and 8 relationships.

Observing the "ufo\_a2.csv" dataset, I extract the year, month, day, hour, shape, weather condition, and wind direction from the "ufo\_a2.csv" dataset and set them as individual nodes.

Years can be grouped by different years. There are not too many different years, so it is better to regard different years as separate nodes. Use relationships to connect year and UFO so that I do not need to create more than 3000 pieces of year information. It is the same reason for the month(only having a unique value from 1 to 12), day(1-31), hour(0-24). The shape column also does not have too many unique values. Values of shape descriptions are duplicated, so I create separate shape nodes for each unique shape value. It is the same reason for weather conditions and wind direction.

Data of other columns as properties of node UFO. Each row in the "ufo\_a2.csv" dataset is regarded as one UFO node.

Observing the "state\_a2.csv" dataset, Each row has different location records. There are many duplicated county names and city names on the dataset. So I think it is better to regard each location as an entity. I create one location node and then set city, state, county name, longitude and latitude as properties of the location node.

In conclusion, my design assumption is according to the data distribution of each column. If the value of this column is low distribution, I tend to set this column as an individual node. And use a relationship to connect additional nodes with UFO nodes.

## C.2. Queries

★ (Completed code in TaskC2.cypher)

### 0. Pre. create Indexes

```
CREATE INDEX ON:Location(state);  
  
CREATE INDEX ON:Location(city);  
  
CREATE INDEX ON:Location(countyName);
```

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
index_12fed2f	BTREE	NONUNIQUE	NODE	[ "Location" ]	[ "countyName" ]	ONLINE
index_343aff4e	LOOKUP	NONUNIQUE	NODE	[]	[]	ONLINE
index_accb94fe	BTREE	NONUNIQUE	NODE	[ "Location" ]	[ "state" ]	ONLINE
index_cac4c6f1	BTREE	NONUNIQUE	NODE	[ "Location" ]	[ "city" ]	ONLINE
index_f7700477	LOOKUP	NONUNIQUE	RELATIONSHIP	[]	[]	ONLINE

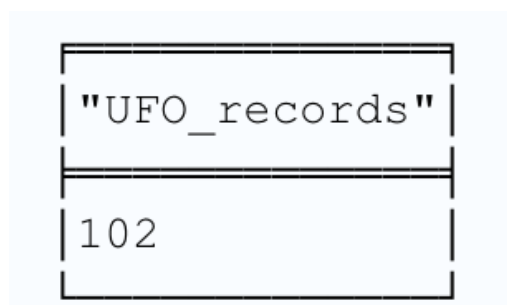
Figure 6: Create indexes on city, state, county

Explanation: Location contains multiple properties(city, state and county). Observing query questions, state, city and county are used frequently. So it is better to set an index on each of them to do efficient queries.

### 1. Query 1

How many UFO sightings were recorded in April?

```
MATCH (u:UFO) -[:IN_MONTH_OF]->(m:Month)  
  
WHERE m.month =4  
  
RETURN COUNT(u) AS UFO_records;
```



"UFO_records"
102

Figure 7: query 1 result



## 2. Query 2

Show all unique weather conditions, for UFOs in 'CIRCLE' shape appeared in 'AZ' state before 2014(exclusive). Display all weather conditions in lowercase letters.

```
MATCH (u:UFO)-[:IN_YEAR_OF]->(y:Year),
(u:UFO)-[:IS_WEATHER_CONDITION_OF]->(c:Conds),
(u:UFO)-[:IS_HAPPENED_IN]->(l:Location{state:'AZ'}),
(u:UFO)-[:IS_SHAPE_OF]->(s:Shape{shape:'CIRCLE'})
WHERE y.year<2014
RETURN DISTINCT toLower(c.condition) AS `unique weather conditions`;
```

"unique weather conditions"
"scattered clouds"
"partly cloudy"
"clear"
"overcast"
"mostly cloudy"

Figure 7: query 2 result

## 3. Query 3

Show all unique UFO shapes that appeared in 2015 but not in 2000.

```
MATCH (u:UFO)-[:IN_YEAR_OF]->(y:Year{year:2015}), (u:UFO)-[:IS_SHAPE_OF]->(s:Shape)
WHERE NOT (u:UFO)-[:IN_YEAR_OF]->(y:Year{year:2000})
RETURN DISTINCT s.shape AS `unique shapes`;
```

"unique shapes"
"FIREBALL"
"FORMATION"
"CHANGING"
"LIGHT"
"CIRCLE"
"OVAL"
"TRIANGLE"
"SPHERE"
"DISK"
"CONE"
"OTHER"

Figure 8: query 3 result



#### 4. Query 4

List all unique years in ascending order if it has 'at high speeds' in the text in each UFO sighting recording.

```
MATCH (u:UFO)-[:IN_YEAR_OF]->(y:Year)
WHERE u.text CONTAINS 'at high speeds'
RETURN DISTINCT y.year AS `year`
ORDER BY y.year ASC;
```

"year"
2008
2011
2013

Figure 9: query 4 result

#### 5. Query 5

Count how many times each wind direction appeared across all years, sort the number of times of each direction in descending order.

```
MATCH (u:UFO)-[:IS_WEATHER_DIRECTION_OF]->(wd:Wdire),
(u:UFO)-[:IN_YEAR_OF]->(y:Year)
RETURN wd.direction AS `wdire`,COUNT(wd.direction) AS `times of each direction`
ORDER BY COUNT(wd.direction) DESC;
```

"wdire"	"times of each direction"
"NORTH"	745
"SOUTH"	299
"WEST"	217
"SSE"	184
"SSW"	183
"SW"	179
"WSW"	177
"SE"	172
"WNW"	138
"ESE"	132
"NW"	129
"VARIABLE"	129
"EAST"	121
"NNW"	98
"ENE"	75
"NE"	71
"NNE"	70

Figure 10: query 5 result

## 6. Query 6

Display the nearest city information around 'CORAL SPRINGS' city in 'BROWARD' county of 'FL'. The output should also display the distance calculated between 'CORAL SPRINGS' city and the nearest city you found.

```
MATCH (l:Location{city:'CORAL SPRINGS',countyName:'BROWARD',state:'FL'})
WITH l, point({longitude:l.lng,latitude:l.lat}) AS location1
MATCH (l2:Location)
WITH location1,l,l2,point({longitude:l2.lng,latitude:l2.lat}) AS location2
WHERE location1 <> location2
WITH l.city AS `source city`,l2.city AS `target city`,distance(location1,location2)
AS distance
RETURN `source city`,`target city`,distance
ORDER BY distance ASC
LIMIT 1;
```

"source city"	"target city"	"distance"
"CORAL SPRINGS"	"MARGATE"	5394.95935153865

Figure 11: query 6 result

## 7. Query 7

Find the year with the least number of different kinds of UFO shapes. Display the year and the number of counting in the output.

```
MATCH (u:UFO)-[:IN_YEAR_OF]->(y:Year),(u:UFO)-[:IS_SHAPE_OF]->(s:Shape)
RETURN y.year AS Year, COUNT(DISTINCT(s.shape)) AS `number of counting`
ORDER BY `number of counting` ASC
LIMIT 1;
```

"Year"	"number of counting"
1997	3

Figure 12: query 7 result

## 8. Query 8

What is the average temperature, pressure, and humidity of each UFO shape? (The output should also display the average values rounded to 3 decimal places)

```
MATCH (u:UFO)-[:IS_SHAPE_OF]->(s:Shape),(u:UFO)

WITH DISTINCT(s.shape) AS `shape category`,round(AVG(u.temp),3) AS `average
temperature`,round(AVG(u.hum),3) AS `average humidity`,round(AVG(u.pressure),3) AS
`average pressure`

RETURN `shape category`,`average temperature`,`average humidity`,`average pressure`

ORDER BY `average humidity` DESC;
```

"shape category"	"average temperature"	"average humidity"	"average pressure"
"CONE"	null	null	-0.202
"FIREBALL"	1.046	-0.608	-0.313
"TRIANGLE"	0.847	-0.655	-0.332
"LIGHT"	0.899	-0.729	-0.382
"FORMATION"	0.968	-0.779	-0.373
"OVAL"	1.101	-0.816	-0.371
"CIRCLE"	1.096	-0.816	-0.349
"SPHERE"	1.096	-0.862	-0.418
"DISK"	1.216	-0.995	-0.445
"CHANGING"	1.073	-1.02	-0.445
"CHEVRON"	0.016	-1.184	-0.255
"UNKNOWN"	0.716	-1.263	0.066
"OTHER"	1.249	-1.504	-0.386
"TEARDROP"	2.075	-1.659	-0.806
"RECTANGLE"	0.982	-1.659	-0.399
"CIGAR"	1.549	-2.634	-0.255
"FLASH"	1.835	-2.682	-1.973

Figure 13: query 8 result

## 9. Query 9

Display the top 3 counties with the most number of different cities.

```
MATCH (l:Location)
RETURN l.countyName AS countyName, COUNT(l.city) AS `city numbers`
ORDER BY `city numbers` DESC
LIMIT 3;
```

"countyName"	"city numbers"
"WASHINGTON"	87
"JEFFERSON"	79
"LOS ANGELES"	79

Figure 14: query 9 result

## 10. Query 10

Rank the total number of UFO sighting recordings according to each state, display the state in descending order in the output.

```
MATCH(u:UFO)-[:IS_HAPPENED_IN]->(l:Location)
WITH l.state AS State
RETURN State, COUNT(*) AS `number of UFO sighting recordings`
ORDER BY State DESC;
```

"State"	"number of UFO sighting recordings"
"WY"	5
"WV"	8
"WI"	22
"WA"	80
"VT"	1
"VA"	42
"UT"	29
"TX"	280
"TN"	44
"SD"	9
"SC"	47
"RI"	2
"PA"	58
"OR"	43
"OK"	65
"OH"	58
"NY"	39
"NV"	104
"NM"	21
"NJ"	50
"NH"	5
"NE"	15
"ND"	5
"NC"	60
"MT"	11
"MS"	16
"MO"	39
"MN"	19
"MI"	41
"ME"	1
"MD"	109
"MA"	43
"LA"	38
"KY"	37
"KS"	33
"IN"	34
"IL"	59
"ID"	22
"IA"	19
"HI"	12
"GA"	66
"FL"	303
"DE"	11
"DC"	1
"CT"	6
"CO"	161
"CA"	339
"AZ"	455
"AR"	17
"AL"	23
"AK"	9

Figure 15: query 10 result

## C.3. Database Modifications

★ (Completed code in TaskC3.cypher)

### 1. Modification 1

MASL has gained new information about a new UFO sighting. Therefore, insert all of the information provided in Table 1.

```
// Check database before change it.  
// show no such node in my database.
```

```
1 MATCH (u)-[:IN_YEAR_OF]→(y2:Year{year:2021}),  
2 (u)-[:IN_MONTH_OF]→(m2:Month{month:1}),  
3 (u)-[:IN_DAY_OF]→(d2:Day{day:14}),  
4 (u)-[:IN_HOUR_OF]→(h2:Hour{hour:23}),  
5 (u)-[:IS_HAPPENED_IN]→  
  (l2:Location{state:'IN',city:'HIGHLAND',countyName:'LAKE'})  
6 RETURN u;
```

(no changes, no records)

Figure 16: screenshot of database before setting new node

```
MATCH (u1:UFO)-[:IN_MONTH_OF]→(m:Month{month:8}),  
(u1:UFO)-[:IN_DAY_OF]→(d:Day{day:14}),  
(u1:UFO)-[:IN_HOUR_OF]→(h:Hour{hour:16}),  
(u1:UFO)-[:IN_YEAR_OF]→(y:Year{year:1998}),  
(u1:UFO)-[:IS_WEATHER_CONDITION_OF]→(c1:Conds),  
(u1:UFO)-[:IS_WEATHER_DIRECTION_OF]→(w1:Wdire)  
CREATE (u:UFO{})  
SET u.duration='25 minutes'  
SET u.text='Awesome lights were seen in the sky'  
SET u.summary='Awesome lights'  
SET u.pressure=u1.pressure  
SET u.temp=u1.temp  
SET u.windchill=u1.windchill  
SET u.rain=u1.rain  
SET u.vis=u1.vis  
SET u.dewpt=u1.dewpt  
SET u.thunder=u1.thunder  
SET u.fog=u1.fog  
SET u.precip=u1.precip  
SET u.wspd=u1.wspd  
SET u.tornado=u1.tornado  
SET u.hum=u1.hum  
SET u.snow=u1.snow  
SET u.wgust=u1.wgust  
SET u.heatindex=u1.heatindex  
SET u.hail=u1.hail
```

```

MERGE (u)-[:IN_YEAR_OF]->(y2:Year{year:2021})
MERGE (u)-[:IS_WEATHER_CONDITION_OF]->(c1)
MERGE (u)-[:IS_WEATHER_DIRECTION_OF]->(w1)
MERGE (u)-[:IN_MONTH_OF]->(m2:Month{month:1})
MERGE (u)-[:IN_DAY_OF]->(d2:Day{day:14})
MERGE (u)-[:IN_HOUR_OF]->(h2:Hour{hour:23})
MERGE (u)-[:IS_HAPPENED_IN]->(l2:Location{state:'IN',city:'HIGHLAND',countyName:'LAK
E'})
RETURN u,l2,c1,w1,d2,m2,h2,y2;

```

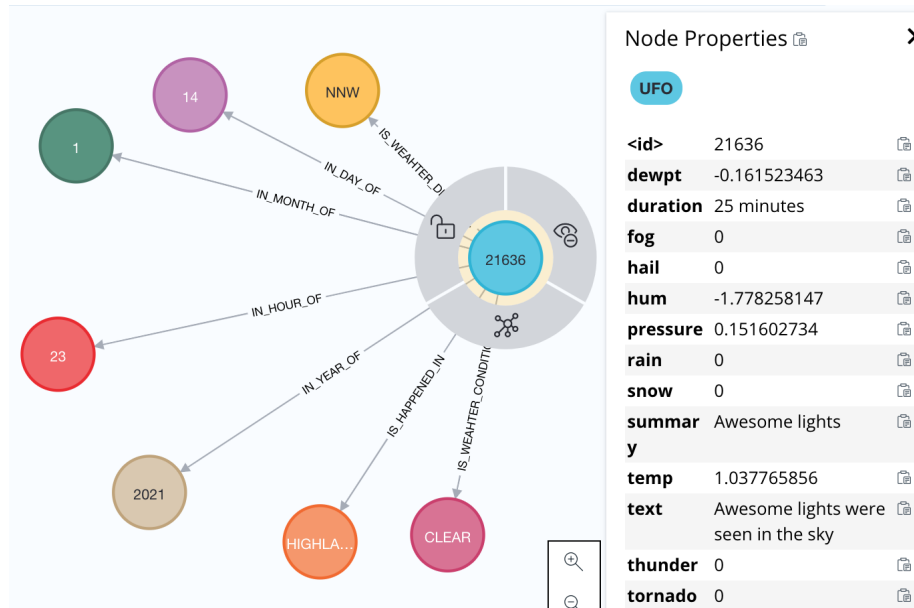


Figure 17: screenshot of after setting new node

## 2. Modification 2

They have also realised that for all UFO sightings recorded in 2011 and 2008 with 'Unknown' shape should be 'flying saucer' and also the 'Clear' weather condition should be 'Sunny/Clear'. Update these records with the correct UFO shape and weather conditions.

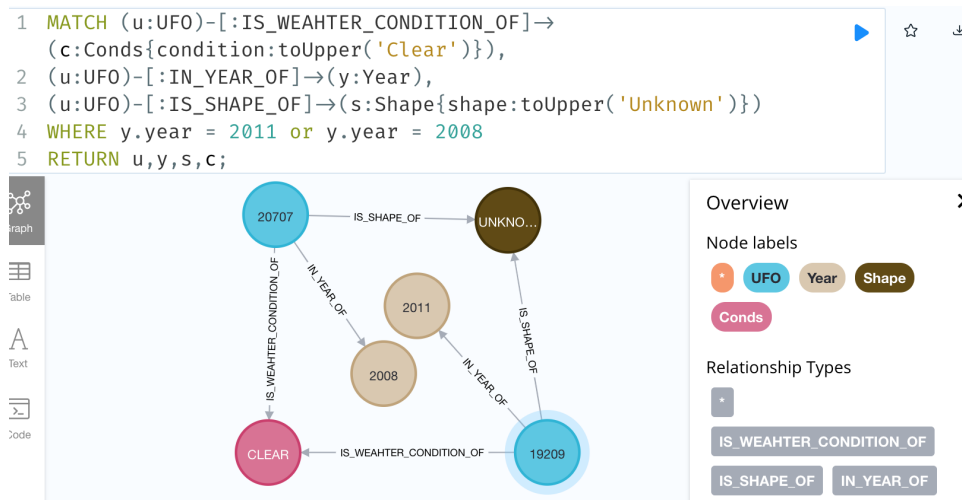


Figure 18: Screenshot of before changing properties' values

```

MATCH (u:UFO)-[:IS_WEATHER_CONDITION_OF]->(c:Conds{condition:toUpper('Clear')}),
(u:UFO)-[:IN_YEAR_OF]->(y:Year),

(u:UFO)-[:IS_SHAPE_OF]->(s:Shape{shape:toUpper('Unknown')})

WHERE y.year = 2011 or y.year = 2008

SET s.shape = 'flying saucer'

SET c.condition = 'Sunny/Clear'

RETURN u,y,s,c;

```

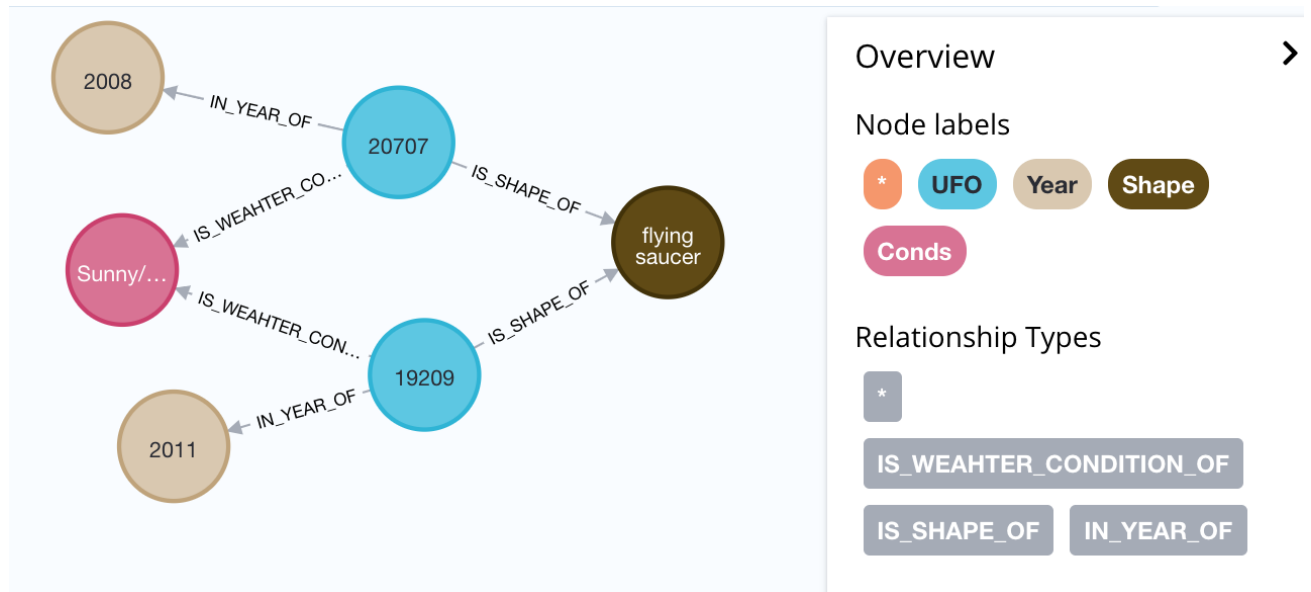


Figure 19: Screenshot of after changing data

### 3. Modification 3

The management has realised that 'ARCADIA' city in FL state was an error. Therefore delete the 'ARCADIA' city from the database.

```

MATCH(l:Location{city:'ARCADIA',state:'FL'})
RETURN l;

```

The screenshot shows a graph visualization of the query results. The graph contains a single node labeled 'ARCADIA' (Location). The right sidebar shows the **Node Properties** panel with the following information:

Location	
<id>	11167
city	ARCADIA
countyName	DESOTO
lat	27.2213
lng	-81.8587
state	FL

Figure 20: Screenshot of before deleting this node



```
MATCH(l:Location{city:'ARCADIA',state:'FL'})
```

```
DETACH DELETE l;
```

```
MATCH(l:Location{city:'ARCADIA',state:'FL'})  
DETACH DELETE l;
```

Deleted 1 node, completed after 2 ms.

```
MATCH(l:Location{city:'ARCADIA',state:'FL'})  
RETURN l;
```

(no changes, no records)

Figure 21: Screenshot of after deleting the node

## C.4. Connecting to Drivers

```
# download python neo4j driver
# python -m pip install py2neo

# upgrade neo4j to latest version
pip install py2neo --upgrade

# import library
import py2neo

# import functions from library
from py2neo import Graph, Node, Relationship
```

### \*connect to graph database

```
g= Graph(host = 'localhost',auth = ('neo4j','5137'))

# clear graph database
# g.run("MATCH (n) RETURN n")
# g.run("MATCH (n) DETACH DELETE n")

# import data-state
g.run("LOAD CSV WITH HEADERS FROM 'file:///states_a2.csv' AS data WITH data \
      WHERE data.city IS NOT NULL\
      MERGE (location:Location{city:toUpper(data.city), \
      lat:toFloat(data.lat),lng:toFloat(data.lng), \
      countyName:toUpper(data.countyName), \
      state:toUpper(data.state)})");

# import data-ufo
Completed code in TaskC4.py
g.run("LOAD CSV WITH HEADERS FROM 'file:///ufo_a2.csv' AS data WITH data\
      CREATE (u:UFO{ })\
      SET u.duration = (CASE data.`duration` WHEN '' THEN NULL WHEN 'NA' THEN NULL\
      ELSE trim(data.`duration`)END)\
      FOREACH(IGNOREME IN CASE data.shape WHEN '' THEN NULL WHEN 'NA' THEN NULL\
      ELSE toUpper(trim(data.shape)) END|\
      MERGE (s:Shape {shape:toUpper(trim(data.shape))})\
      MERGE (u) -[:IS_SHAPE_OF]-> (s)");

#create relationships
g.run("LOAD CSV WITH HEADERS FROM 'file:///ufo_a2.csv' AS data\
      WITH data\
      MATCH (u:UFO{\
```

```

    city:toUpper(data.city),\
    state:toUpper(data.state),\
    text:data.text,\
    summary:data.summary,\
    pressure:toFloat(data.pressure)})\
MATCH(location:Location{city:toUpper(data.city),state:toUpper(data.state)})\
CREATE(u)-[:IS_HAPPENED_IN]->(location);\
MATCH (y:Year) SET y.year = toInteger(y.year);\
MATCH (y:Month) SET y.month = toInteger(y.month);\
MATCH (y:Day) SET y.day = toInteger(y.day);\
MATCH (y:Hour) SET y.hour = toInteger(y.hour);";

```

## \*queries

Completed code in TaskC4.py

```

g.run("CREATE INDEX ON:Location(state);\
CREATE INDEX ON:Location(city);\
CREATE INDEX ON:Location(countyName);");

```

# query 1

```

g.run("MATCH(u:UFO)-[:IN_MONTH_OF]->(m:Month)\
WHERE m.month =4\
RETURN COUNT(u) AS UFO_records;")

```

[6] ✓ 0.3s

... UFO\_records

102

Figure 22: neo4j python driver query 1

#query 2

```

g.run("MATCH(u:UFO)-[:IN_YEAR_OF]->(y:Year),\
(u:UFO)-[:IS_WEATHER_CONDITION_OF]->(c:Conds),\
(u:UFO)-[:IS_HAPPENED_IN]->(l:Location{state:'AZ'}),\
(u:UFO)-[:IS_SHAPE_OF]->(s:Shape{shape:'CIRCLE'})\
WHERE y.year<2014\
RETURN DISTINCT toLower(c.condition) AS `unique weather conditions`;")

```

[8] ✓ 0.5s

... unique weather conditions

scattered clouds

partly cloudy

clear

Figure 23:Figure 22: neo4j python driver query 2

#query 3

```
g.run("MATCH(u:UFO)-[:IN_YEAR_OF]->(y:Year{year:2015}),(u:UFO)-[:IS_SHAPE_OF]->(s:Shape)\nWHERE NOT (u:UFO)-[:IN_YEAR_OF]->(y:Year{year:2000})\nRETURN DISTINCT s.shape AS `unique shapes`")
```

✓ 0.8s

unique shapes

-----

OVAL  
LIGHT  
DISK

Figure 23: Figure 22: neo4j python driver query 3

#query 4

```
g.run("MATCH (u:UFO)-[:IN_YEAR_OF]->(y:Year)\nWHERE u.text CONTAINS 'at high speeds'\nRETURN DISTINCT y.year AS `year`\nORDER BY y.year ASC;")
```

✓ 0.4s

year

2008
2011
2013

Figure 24: query 4

#query 5

```
g.run("MATCH(u:UFO)-[:IS_WEATHER_DIRECTION_OF]->(wd:Wdire),\n(u:UFO)-[:IN_YEAR_OF]->(y:Year)\nRETURN wd.direction AS `wdire`,COUNT(wd.direction) AS `times of each direction`\nORDER BY COUNT(wd.direction) DESC;")
```

✓ 0.3s

wdire	times of each direction
NORTH	745
SOUTH	299
WEST	217

Figure 25: query 5

#query 6

```
g.run("MATCH (l:Location{city:'CORAL SPRINGS',countyName:'BROWARD',state:'FL'}) \nWITH l, point({longitude:l.lng,latitude:l.lat}) AS location1 \nMATCH(l2:Location) \nWITH location1,l,l2,point({longitude:l2.lng,latitude:l2.lat}) AS location2 \nWHERE location1 <> location2 \nWITH l.city AS `source city`,l2.city AS `target city`,distance(location1,location2) AS distance \nRETURN `source city`,`target city`,distance \nORDER BY distance ASC \nLIMIT 1;")
```

✓ 0.4s

source city	target city	distance
CORAL SPRINGS	MARGATE	5394.95935153865

Figure 26: query 6

#query 7

```
g.run("MATCH (u:UFO)-[:IN_YEAR_OF]->(y:Year),(u:UFO)-[:IS_SHAPE_OF]->(s:Shape) \
RETURN y.year AS Year, COUNT(DISTINCT(s.shape)) AS `number of counting` \
ORDER BY `number of counting` ASC \
LIMIT 1;")
```

[23] ✓ 0.5s

...

Year	number of counting
1997	3

Figure 27: query 7

#query 8

```
g.run("MATCH (u:UFO)-[:IS_SHAPE_OF]->(s:Shape),(u:UFO)\
WITH DISTINCT(s.shape) AS `shape category`,round(AVG(u.temp),3) AS `average temperature`,\
round(AVG(u.hum),3) AS `average humidity`,round(AVG(u.pressure),3) AS `average pressure`\
RETURN `shape category`,`average temperature`,`average humidity`,`average pressure`\
ORDER BY `average humidity` DESC;")
```

✓ 0.3s

shape category	average temperature	average humidity	average pressure
CONE	null	null	-0.202
FIREBALL	1.046	-0.608	-0.313
TRIANGLE	0.847	-0.655	-0.332

Figure 28: query 8

#query 9

```
g.run("MATCH (l:Location)\
RETURN l.countyName AS countyName, COUNT(l.city) AS `city numbers`\
ORDER BY `city numbers` DESC \
LIMIT 3;")
```

6] ✓ 0.2s

..

countyName	city numbers
WASHINGTON	87
JEFFERSON	79
LOS ANGELES	79

Figure 29: query 9

#query 10

```
g.run("MATCH(u:UFO)-[:IS_HAPPENED_IN]->(l:Location)\
WITH l.state AS State \
RETURN State, COUNT(*) AS `number of UFO sighting recordings`\
ORDER BY State DESC;")
```

8] ✓ 0.1s

..

State	number of UFO sighting recordings
WY	5
WV	8
WI	22

Figure 30:query 10

## \*Database modification

# modification 1

```

g.run("MATCH (u1:UFO)-[:IN_MONTH_OF]->(m:Month{month:8}),\
(u1:UFO)-[:IN_DAY_OF]->(d:Day{day:14}),\
(u1:UFO)-[:IN_HOUR_OF]->(h:Hour{hour:16}),\
(u1:UFO)-[:IN_YEAR_OF]->(y:Year{year:1998}),\
(u1:UFO)-[:IS_WEAHTER_CONDITION_OF]->(c1:Conds),\
(u1:UFO)-[:IS_WEAHTER_DIRECTION_OF]->(w1:Wdire) \
CREATE (u:UFO{ }) \
SET u.duration='25 minutes' \
SET u.text='Awesome lights were seen in the sky' \
SET u.summary='Awesome lights' \
SET u.pressure=u1.pressure \
SET u.temp=u1.temp \
SET u.windchill=u1.windchill \
SET u.rain=u1.rain \
SET u.vis=u1.vis \
SET u.dewpt=u1.dewpt \
SET u.thunder=u1.thunder \
SET u.fog=u1.fog \
SET u.precip=u1.precip \
SET u.wspd=u1.wspd \
SET u.tornado=u1.tornado \
SET u.hum=u1.hum \
SET u.snow=u1.snow \
SET u.wgust=u1.wgust \
SET u.heatindex=u1.heatindex \
SET u.hail=u1.hail \
MERGE(u)-[:IN_YEAR_OF]->(y2:Year{year:2021}) \
MERGE (u)-[:IS_WEAHTER_CONDITION_OF]->(c1) \
MERGE(u)-[:IS_WEAHTER_DIRECTION_OF]->(w1) \
MERGE(u)-[:IN_MONTH_OF]->(m2:Month{month:1}) \
MERGE(u)-[:IN_DAY_OF]->(d2:Day{day:14}) \
MERGE(u)-[:IN_HOUR_OF]->(h2:Hour{hour:23}) \
MERGE(u)-[:IS_HAPPENED_IN]->(l2:Location{state:'IN',city:'HIGHLAND',countyName:'LAKE'}) \
RETURN u,l2,c1,w1,d2,m2,h2,y2;"

```

✓ 1.8s

u	l2	c1	w1	d2	m2	h2	y2
(_:UFO {dewpt: -0.161523463, duration: '25 minutes', fog: 0, hail: 0, hum: -1.778258147, pressure: 0.151602734, rain: 0, snow: 0, summary: 'Awesome lights', temp: 1.037765856, text: 'Awesome lights were seen in the sky', thunder: 0, tornado: 0, vis: 1.457893595, wspd: 0.551841457})	(_:Location {city: 'HIGHLAND', countyName: 'LAKE', state: 'IN'})	(_:Conds {condition: 'CLEAR'})	(_:Wdire {direction: 'NNW'})	(_:Day {day: 14})	(_:Month {month: 1})	(_:Hour {hour: 23})	(_:Year {year: 2021})

Figure 31: modification 1

# modification 2

```
g.run("MATCH (u:UFO)-[:IS_WEATHER_CONDITION_OF]->(c:Conds{condition:toUpper('Clear')}),\n(u:UFO)-[:IN_YEAR_OF]->(y:Year),\n(u:UFO)-[:IS_SHAPE_OF]->(s:Shape{shape:toUpper('Unknown')}) \\\nWHERE y.year = 2011 or y.year = 2008 \\\nSET s.shape = 'flying saucer' \\\nSET c.condition = 'Sunny/Clear' \\\nRETURN u,y,s,c;")
```

0.4s

uysc

(.42796:UFO {city: 'RIDGECREST', dewpt: -0.827131849, duration: '15 seconds', fog: 0, hail: 0, hum: -2.539301637, pressure: -1.343469169, rain: 0, snow: 0, state: 'CA', summary: 'at three ten this morning and it was in the sky approxamitly in the sky about south east then it moved a little ((anonymous report))', temp: 1.660973936, text: 'From Dim To Bright To Dim On 9-6-08 I spent the evening watching for unexplained phenomena. Sunset was at 7:11 and I was situated at Latitude: 35\u00b032'4.77"N and Longitude: 117\u00b0032'43.80"W I got the sunset time from the internet. I got my camera gear out and got it all set up so that I could be ready for anything. I sat down in my easy chair until it was time for the stars to start showing themselves to the world. One by one they appeared. If you are not fast while you are watching for the stars, they will sneak up on you. Looking at the First Quarter Moon that was to the south of my vantage point, there was a bright star/planet that showed up to the east of the moon and just a little higher in the night sky compared to the moon. It was very bright and steady. I turned around only to find a brighter star/planet on the horizon where the sun had just set. When I turned back around, there was a second star even farther east of the star that was near the moon! It too was bright. \u201csorry, I only know that a planet does not twinkle and stars do\u201d I don\u2019t know the names. Anyhow, I started my scan around the heavens to see what was appearing and by the time I got back to the bright star/planet east of the moon, I noticed something very odd. The star/planet to the left of the one that was near the moon was now GONE. One minute it was there and the next minute it was gone. I know it was not a plane as it just sat there in one place when I first noticed it. As the evening progressed, I noticed two objects passing on by. One of them looked like it was headed towards LA and about a half hour later I saw one that was headed north away from LA. Ordinarily I would suspect that they were planes, but there were no strobes on these two objects that looked like stars passing in the night sky. Even though there was no strobe to them, I just noted them as planes. Same speed, no deviation whatsoever. It was about 8:30pm when I looked to the north and off of the big dipper to the right, I noticed this light. This light started to get very bright and all of a sudden, I had surmised that this thing was coming right at me. It just kept on getting bigger and bigger. I turned around and grabbed my camcorder and stand as a complete unit, and when I turned back around, I had to say to myself, what\u2019s the use. It, by the time I turned around started to get smaller again. So I just stood there watching it as it eventually disappeared. This object did not go up, go down, go from side to side or anything. It was as if there was a very high candle power light that was being turned up and then back off slowly. This lasted for about 15 seconds. I have no idea what it was, but I can finally say: I have seen something in the nights sky that makes no sense to me. Twice\u201d In one night\u201d A disappearing planet and a light that made no sense. I would go back out again, but the bugs are so bad as the sun goes down that I did not have fun due to these bothersome critters. I was so busy swatting them that I lost sight of the fact that Rattlesnakes were out and about. A video that is called "Contact has begun" with James Gilliland. I had never seen it before and I did not know it was James in this video when I downloaded it. I only downloaded it because of the title. The very first UFO on this video is exactly what it was that I saw on the above trip I took on Saturday. Instead of two, there was only one in my case and NOW I can finally say,,, " I HAVE SEEN A UFO " Now I am a minority. LOL ((NUFORC Note: We have calculated the nearest city, based on the GPS co-ordinates provided above by the witness. PD))', thunder: 0, tornado: 0, vis: 0.0642725, wspd: -0.357641004))

(.41298:UFO {city: 'BINGHAMTON', dewpt: -0.060673707, duration: '15:00', fog: 0, hail: 0, hum: -1.255040748, pressure: 1.92208262, rain: 0, snow: 0, state: 'NY', summary: 'Faintly visible shape-shifting UFO with no illumination at a height of 15000 ft, moving erratically over Chennai, India', temp: 0.629104821, text: "Oct.08, 2011 Red Orb's sightings in Binghamton,NY There were 15-20 bright orbs in the south east horizon in the city of binghamton. I was walking home and stopped at a gas station And noticed the bright red orbs. I alerted other patrons and everyone was Concerned and people including myself began Filming and photographing them. Some have been added to YouTube, These red orbs rose in to the sky making no Noise and moving very slow and then vanishing As more red orbs rose. I made 2 calls to 911 to report this incident.", thunder: 0, tornado: 0, vis: 0.0642725, wspd: -0.357641004))

(.43659:Year {year: 2008})

(.43593:Shape {shape: 'flying saucer'})

(.43601:Conds {condition: 'Sunny/Clear'})

Figure 32: modification 2

# modification 3

```
g.run("MATCH(l:Location{city: 'ARCADIA', state: 'FL'}) \\\nDETACH DELETE l;")
```

0.2s

(No data)

Figure 33: modification 3