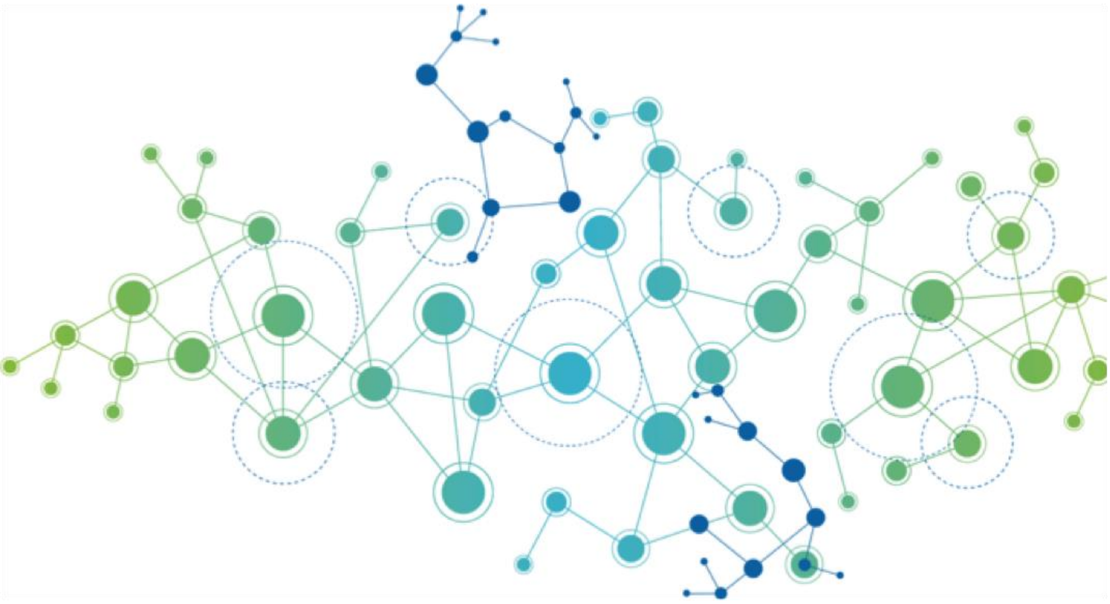
ΟΙΚΟΝΟΜΙΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΑΘΗΝΩΝ

ATHENS UNIVERSITY
OF ECONOMICS
AND BUSINESS

MINING BIG DATASETS PROJECT No2



DEPARTMENT: DEPARTMENT OF MANAGEMENT SCIENCE AND TECHNOLOGY

COURSE: MINING BIG DATASETS

PROFESSOR: Y.KOTIDIS, I.FILIPPIDOU

STUDENTS: CHARILAOS PETRAKOGIANNIS (F2822112), I.DIMOS (F2822102)

ACADEMIC PERIOD: 2021-2022

Table of Context

INTRODUCTION

In this specific project we were given a part of open flights airports network, which contained airports, airlines and flights between airports. In particular, the data contained 7698 airports, 6161 airlines, 6956 cities, 237 countries and 65.935 flights between airports.

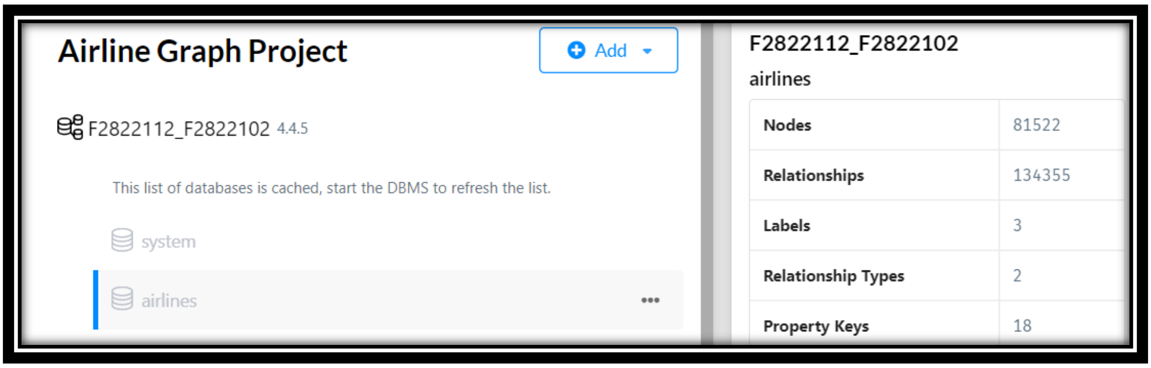
The aforementioned data was in three different ".csv" files, which we had to load in the neo4j program with which we would complete this project.

Below we are going to present you:

- A detailed description of our graph model using a chart and a verbal description of the elements.
- The commands we used in order to import the files to the database.
- The Cypher code for the required queries with their respective results.

PART A: THE GRAPH MODEL

For us to complete this project, we needed to create a new database within the neo4j desktop (Figure 1):



F2822112_F2822102 4.4.5

This list of databases is cached, start the DBMS to refresh the list.

system

airlines

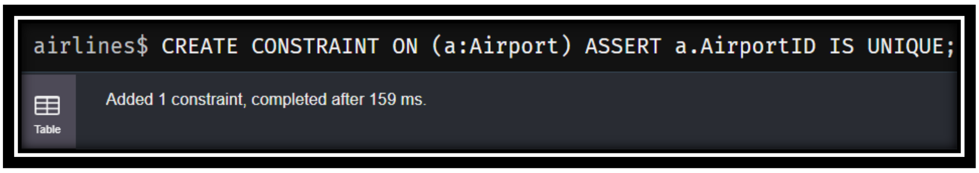
F2822112_F2822102

airlines

Nodes	81522
Relationships	134355
Labels	3
Relationship Types	2
Property Keys	18

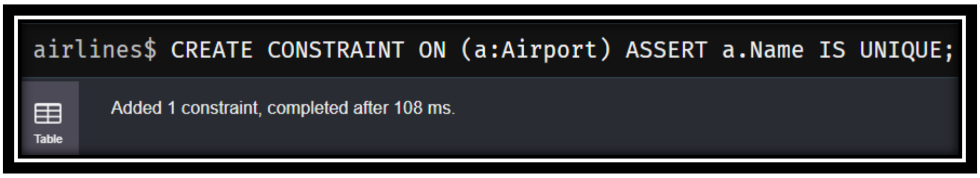
Figure 1 : The neo4j graph database

Our database consisted of three different nodes (Airports, Routes, and Airlines) and two types of directed relationships (From/To) between the vertices of Routes (flights) and Airports (Figures 2-6). In order for us to load the data in neo4j, we had to declare some constraints that would assure us that we would keep only the unique IDs and names of the existing airports:



```
airlines$ CREATE CONSTRAINT ON (a:Airport) ASSERT a.AirportID IS UNIQUE;
```

Added 1 constraint, completed after 159 ms.



```
airlines$ CREATE CONSTRAINT ON (a:Airport) ASSERT a.Name IS UNIQUE;
```

Added 1 constraint, completed after 108 ms.

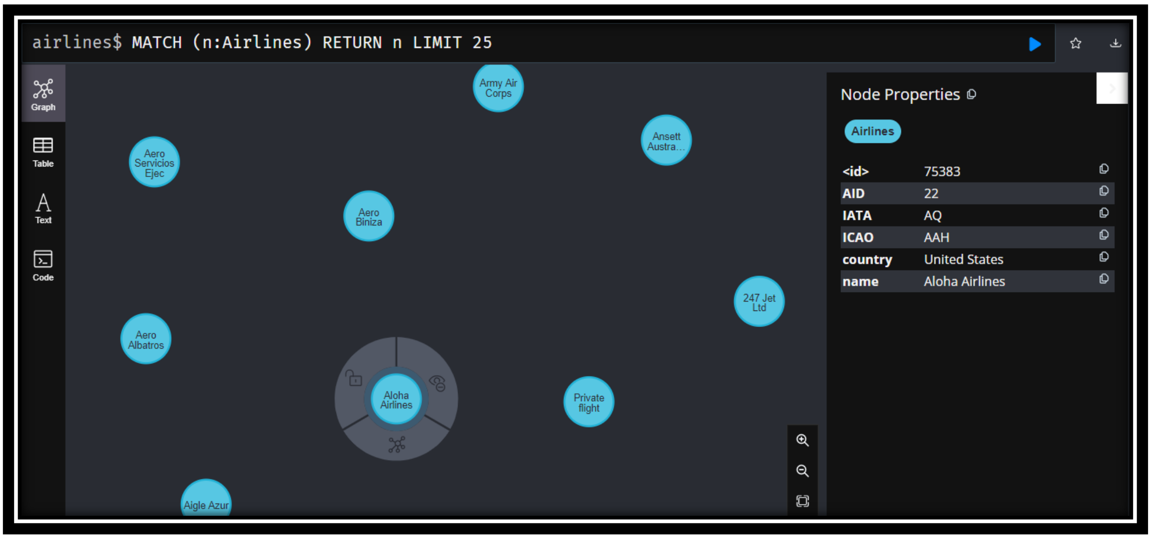
Moreover, in order for us to conduct the project's queries, we had to equip our nodes and edges with the appropriate attributes:

Nodes

- Airlines (Node's ID, IATA, ICAO, Country of origin, Name of the company, Airline ID of each company)
- Routes (Node's ID, Airline's code, Airline ID of each company, Destination, Destination's ID, Equipment's code, Source's code, Source's ID, Number of stops)
- Airports (Node's ID, IATA, ICAO, City of origin, Country of origin, Airport ID, Latitude, Longitude, Name of the airport)

Relationships

- From/To (Relationship's ID, Airline's code, Airline's ID)



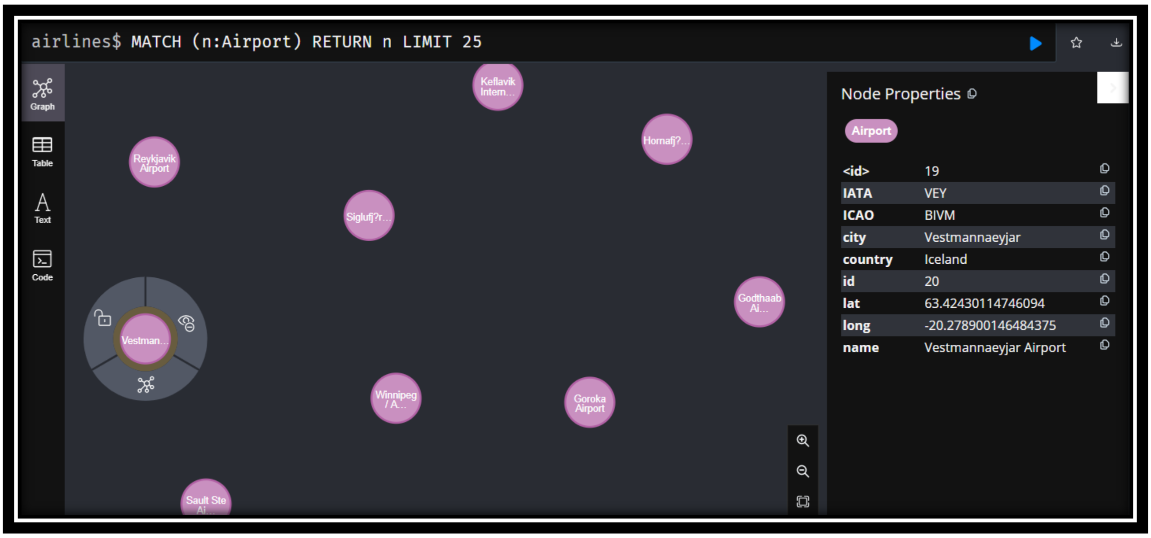
```
airlines$ MATCH (n:Airlines) RETURN n LIMIT 25
```

Node Properties

Airlines

<id>	75383
AID	22
IATA	AQ
ICAO	AAH
country	United States
name	Aloha Airlines

Figure 2 : Airlines' nodes



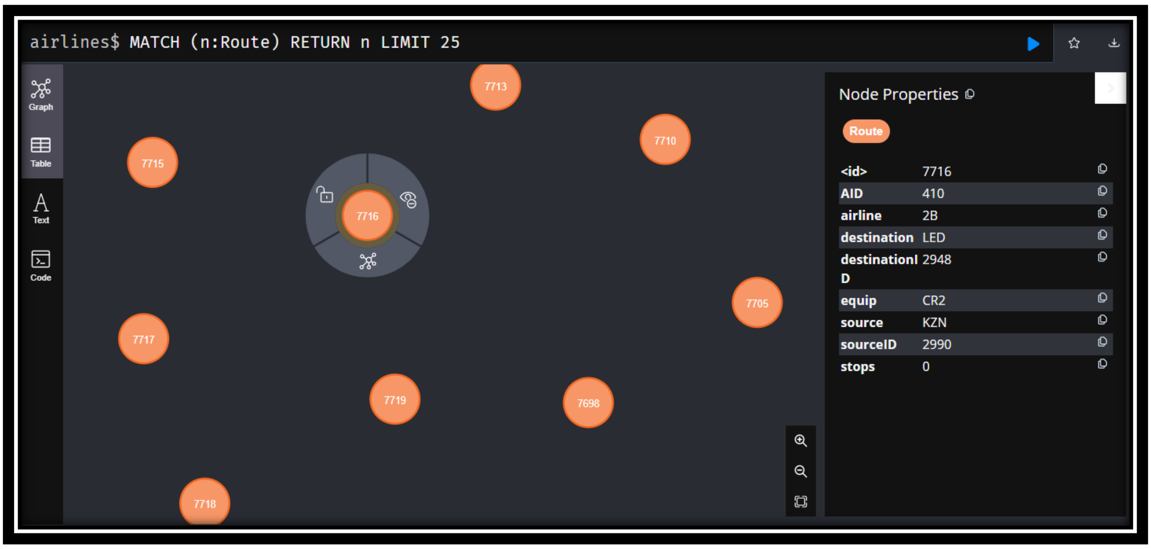
```
airlines$ MATCH (n:Airport) RETURN n LIMIT 25
```

Node Properties

Airport

<id>	19
IATA	VEY
ICAO	BIVM
city	Vestmannaeyjar
country	Iceland
id	20
lat	63.42430114746094
long	-20.278900146484375
name	Vestmannaeyjar Airport

Figure 3 : Airports' nodes



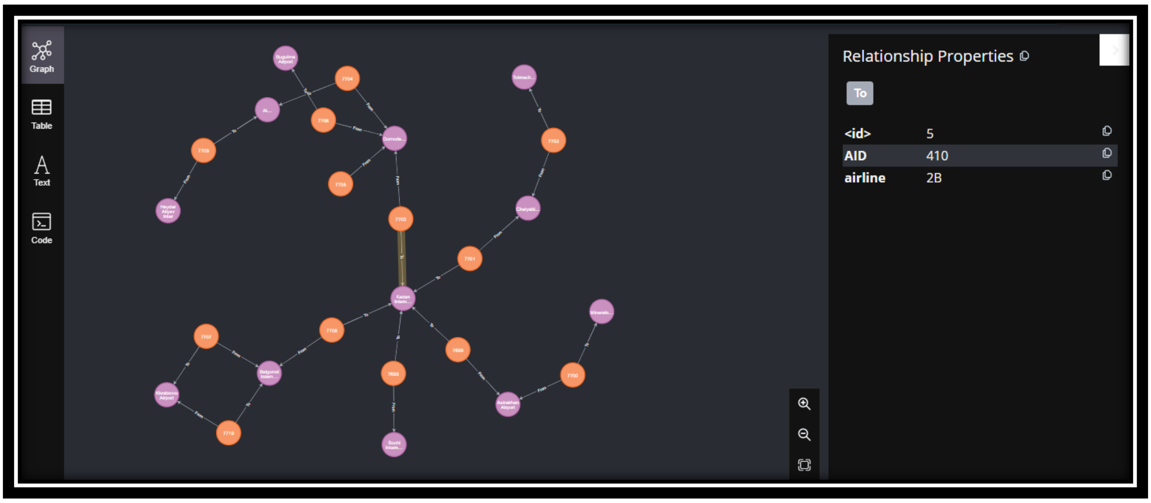
```
airlines$ MATCH (n:Route) RETURN n LIMIT 25
```

Node Properties

Route

<id>	7716
AID	410
airline	2B
destination	LED
destinationID	2948
equip	CR2
source	KZN
sourceID	2990
stops	0

Figure 4 : Routes' nodes

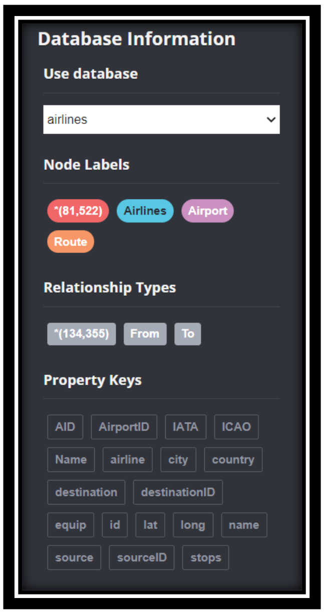


Relationship Properties

To

<id>	5
AID	410
airline	2B

Figure 5 : Relationships between airports and routes (flights)



Database Information

Use database

airlines

Node Labels

*(81,522) Airlines Airport Route

Relationship Types

*(134,355) From To

Property Keys

AID AirportID IATA ICAO Name airline city country destination destinationID equip id lat long name source sourceID stops

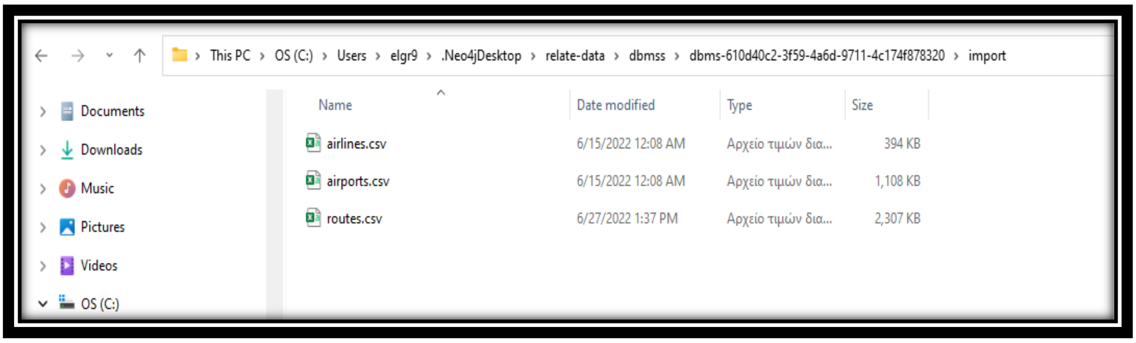
Figure 6 : Database's accumulated information

It has to be mentioned that even though we did not use directly the “Airlines” nodes, their existence was crucial for our queries, due to the fact that they were necessary in order for us to find the name of each airline through the two-digit codes or the unique IDs we had in our possession.

PART B: IMPORT OF THE FILES INTO THE DATABASE

In figure 7 below we can see the directory we placed the files so the neo4j be able to load them into our database.

It is very important to mention that we decided not to drop the null values from our csv files as the majority of the columns were containing information about the flights and only few of them were nulls.

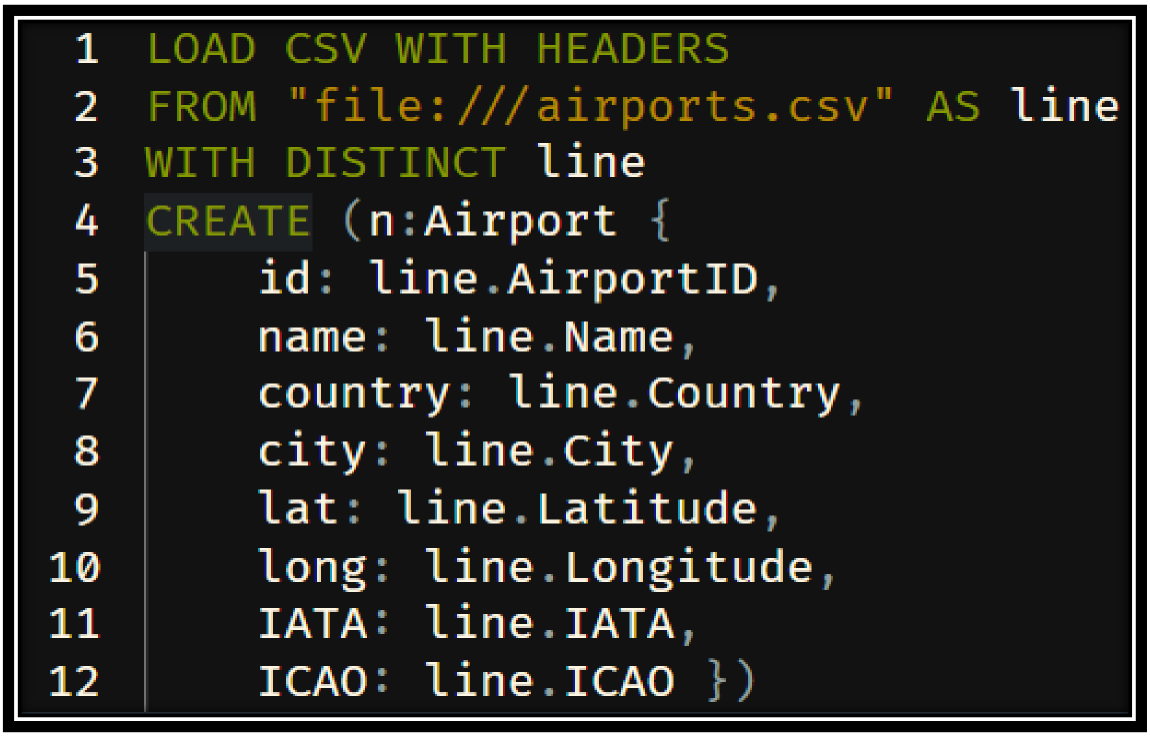


This PC › OS (C:) › Users › elgr9 › .Neo4jDesktop › relate-data › dbmss › dbms-610d40c2-3f59-4a6d-9711-4c174f878320 › import

Name	Date modified	Type	Size
airlines.csv	6/15/2022 12:08 AM	Αρχείο τιμών δια...	394 KB
airports.csv	6/15/2022 12:08 AM	Αρχείο τιμών δια...	1,108 KB
routes.csv	6/27/2022 1:37 PM	Αρχείο τιμών δια...	2,307 KB

Figure 7 : Directory that contains the files for the database

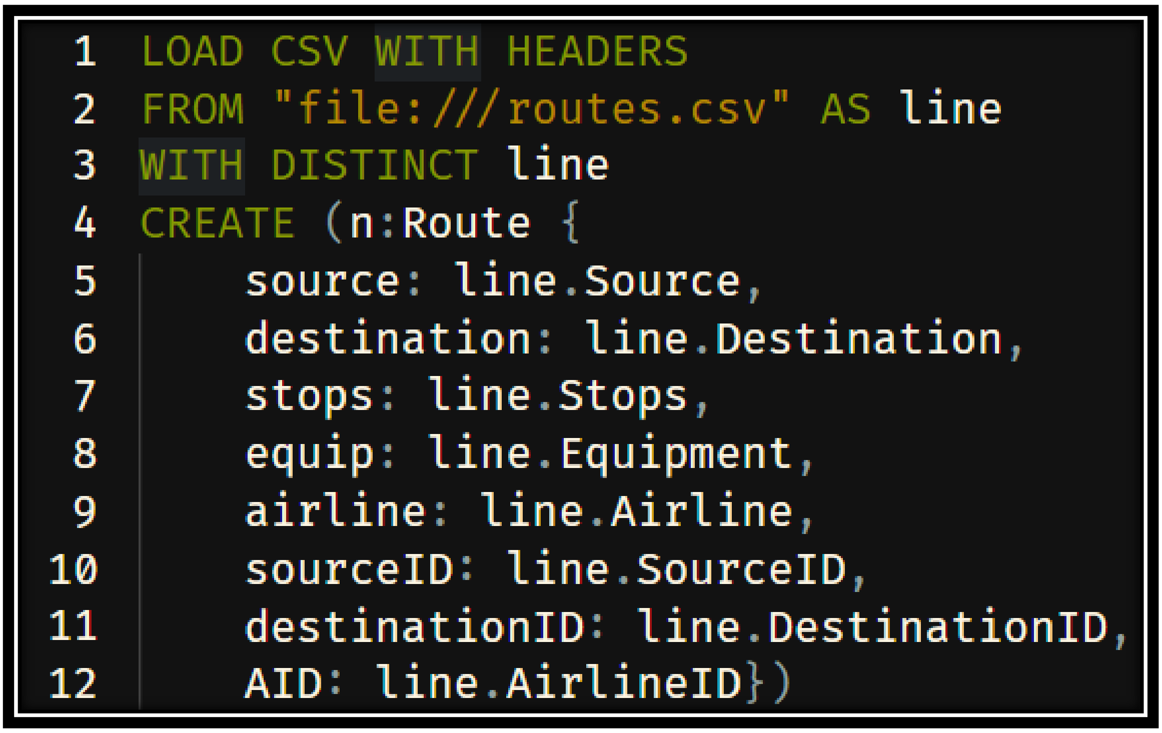
Import of the file that contains information about the Airports



```
LOAD CSV WITH HEADERS
FROM "file:///airports.csv" AS line
WITH DISTINCT line
CREATE (n:Airport {
    id: line.AirportID,
    name: line.Name,
    country: line.Country,
    city: line.City,
    lat: line.Latitude,
    long: line.Longitude,
    IATA: line.IATA,
    ICAO: line.ICAO })
```

Figure 8 : Importation of the Airports.csv file

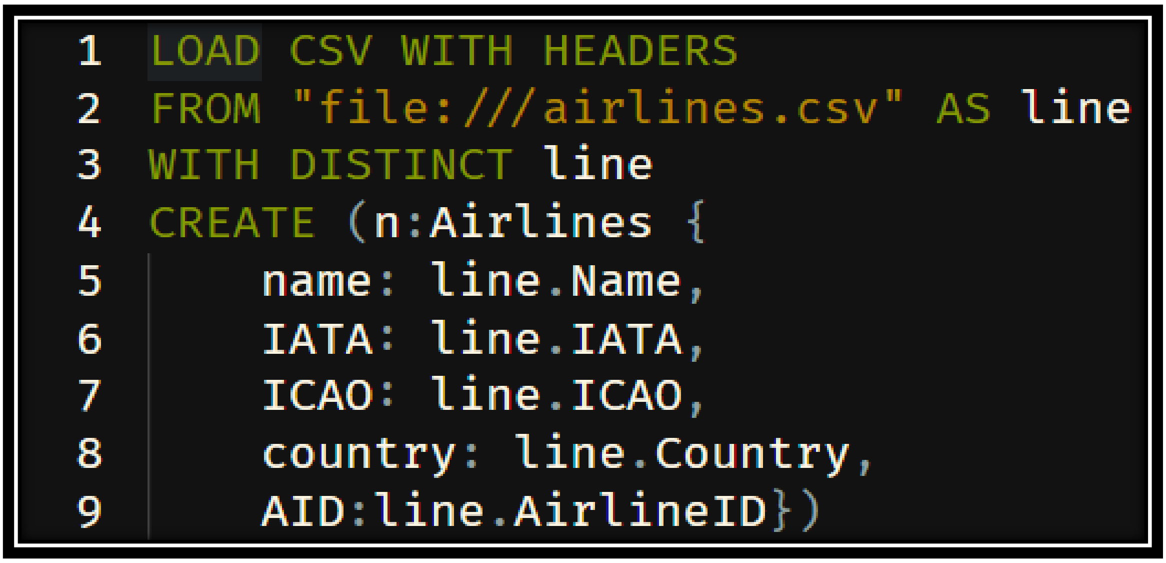
Import of the file that contain information about the Routes



```
LOAD CSV WITH HEADERS
FROM "file:///routes.csv" AS line
WITH DISTINCT line
CREATE (n:Route {
        source: line.Source,
        destination: line.Destination,
        stops: line.Stops,
        equip: line.Equipment,
        airline: line.Airline,
        sourceID: line.SourceID,
        destinationID: line.DestinationID,
        AID: line.AirlineID})
```

Figure 9 : Importation of the routes.csv file

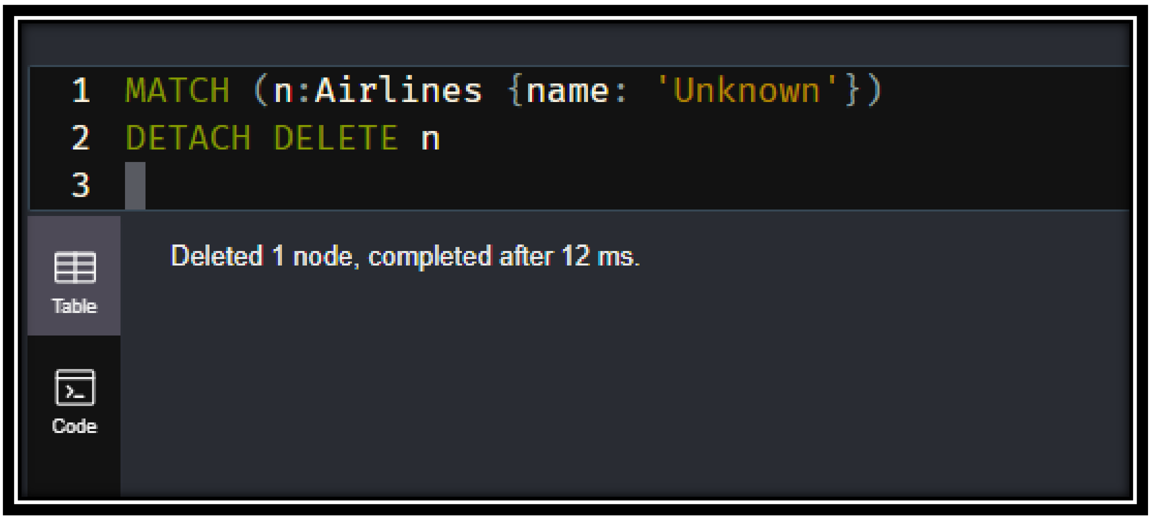
Import of the file that contain information about the Airlines



```
LOAD CSV WITH HEADERS
FROM "file:///airlines.csv" AS line
WITH DISTINCT line
CREATE (n:Airlines {
        name: line.Name,
        IATA: line.IATA,
        ICAO: line.ICAO,
        country: line.Country,
        AID:line.AirlineID})
```

Figure 10 : Importation of the airlines.csv file

In this step we have also created a query to delete one observation that has the name “Unknown”.



```
MATCH (n:Airlines {name: 'Unknown'})
DETACH DELETE n
```

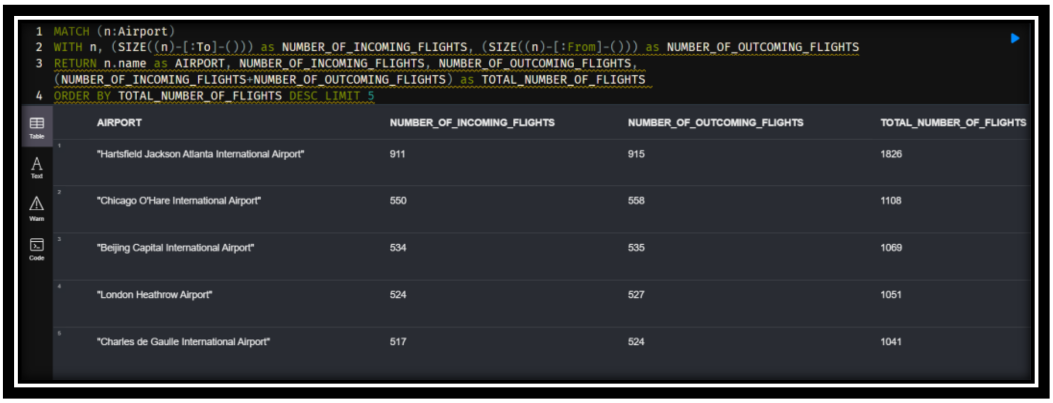
Deleted 1 node, completed after 12 ms.

Figure 11 : Deletion of the Unknown observation

PART C: THE CYPHER CODE

1) Which are the top 5 airports with the most flights. Return airport name and number of flights.

Solution



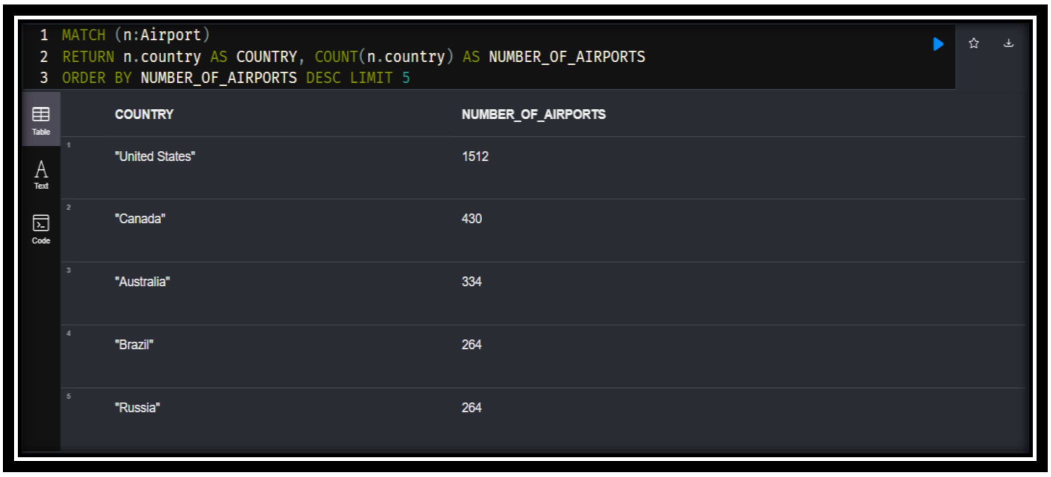
```
MATCH (n:Airport)
WITH n, (SIZE((n)-[:To]-())) as NUMBER_OF_INCOMING_FLIGHTS, (SIZE((n)-[:From]-())) as NUMBER_OF_OUTCOMING_FLIGHTS
RETURN n.name as AIRPORT, NUMBER_OF_INCOMING_FLIGHTS, NUMBER_OF_OUTCOMING_FLIGHTS,
(NUMBER_OF_INCOMING_FLIGHTS+NUMBER_OF_OUTCOMING_FLIGHTS) as TOTAL_NUMBER_OF_FLIGHTS
ORDER BY TOTAL_NUMBER_OF_FLIGHTS DESC LIMIT 5
```

	AIRPORT	NUMBER_OF_INCOMING_FLIGHTS	NUMBER_OF_OUTCOMING_FLIGHTS	TOTAL_NUMBER_OF_FLIGHTS
1	"Hartsfield Jackson Atlanta International Airport"	911	915	1826
2	"Chicago O'Hare International Airport"	550	558	1108
3	"Beijing Capital International Airport"	534	535	1069
4	"London Heathrow Airport"	524	527	1051
5	"Charles de Gaulle International Airport"	517	524	1041

Figure 12 : Query of question No1

2) Which are the top 5 countries with the most airports. Return country name and number of airports.

Solution



```
MATCH (n:Airport)
RETURN n.country AS COUNTRY, COUNT(n.country) AS NUMBER_OF_AIRPORTS
ORDER BY NUMBER_OF_AIRPORTS DESC LIMIT 5
```

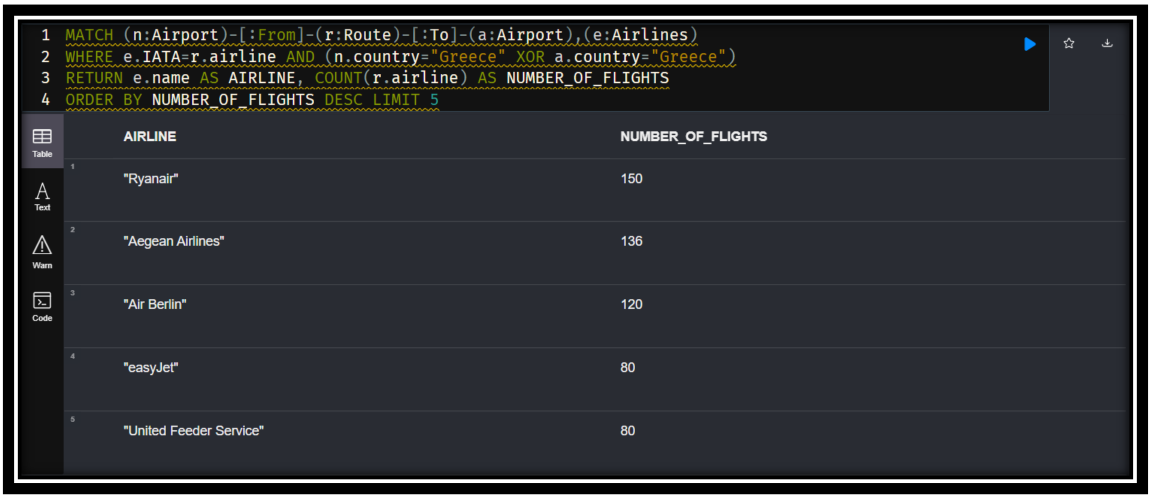
COUNTRY	NUMBER_OF_AIRPORTS
"United States"	1512
"Canada"	430
"Australia"	334
"Brazil"	264
"Russia"	264

Figure 13 : Query of question No2

3) Which are the top 5 airlines with international flights from/to 'Greece'. Return airline name and number of flights.

Note: In this task we've worked with two different approaches. In the first case we worked using IATA codes, which are a unique identifier for each airline. Their difference from Airline ID is that they take into account as a separate entity each subsidiary company that belongs to a larger group. This results in some minor differences in the results produced one way (IATA) and the other (Airline ID). We present both ways because, being non-experts in air data, we want to cover every possible possibility that can produce correct results.

Solution 1

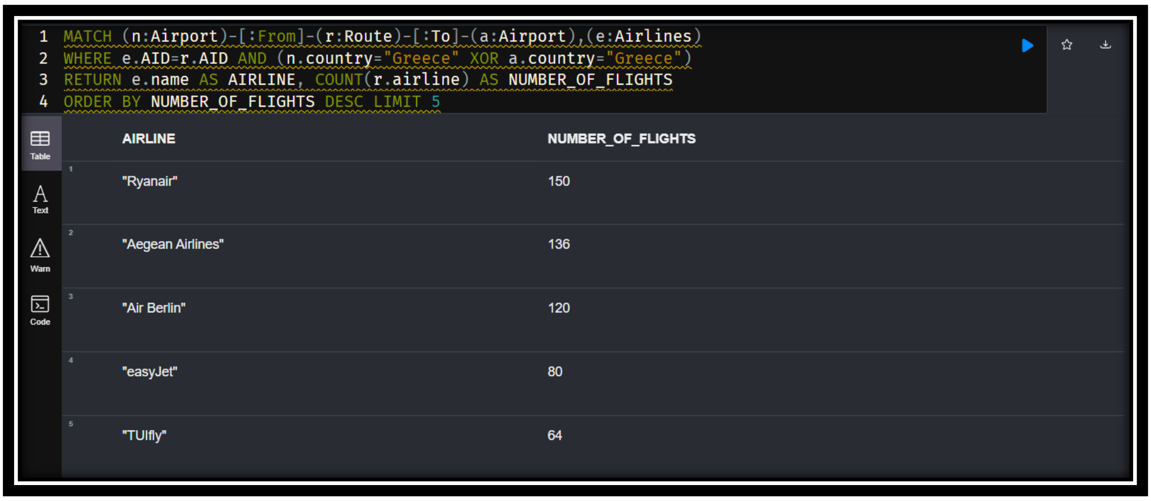


```
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport),(e:Airlines)
WHERE e.IATA=r.airline AND (n.country="Greece" XOR a.country="Greece")
RETURN e.name AS AIRLINE, COUNT(r.airline) AS NUMBER_OF_FLIGHTS
ORDER BY NUMBER_OF_FLIGHTS DESC LIMIT 5
```

AIRLINE	NUMBER_OF_FLIGHTS
"Ryanair"	150
"Aegean Airlines"	136
"Air Berlin"	120
"easyJet"	80
"United Feeder Service"	80

Figure 14 : Query of question No3 (first way)

Solution 2



```
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport),(e:Airlines)
WHERE e.AID=r.AID AND (n.country="Greece" XOR a.country="Greece")
RETURN e.name AS AIRLINE, COUNT(r.airline) AS NUMBER_OF_FLIGHTS
ORDER BY NUMBER_OF_FLIGHTS DESC LIMIT 5
```

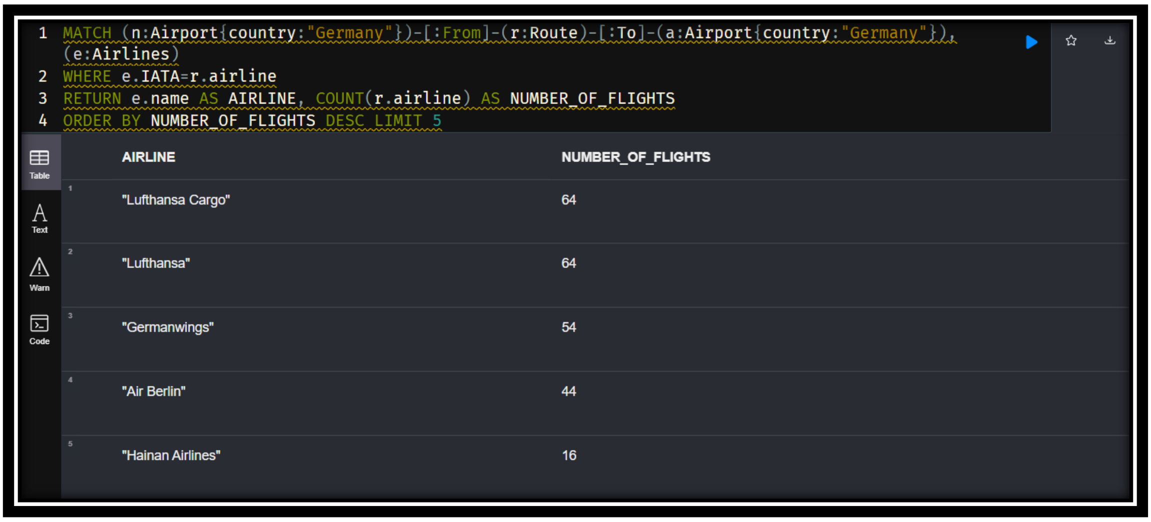
AIRLINE	NUMBER_OF_FLIGHTS
"Ryanair"	150
"Aegean Airlines"	136
"Air Berlin"	120
"easyJet"	80
"TUIfly"	64

Figure 15 : Query of question No3 (second way)

4) Which are the top 5 airlines with local flights inside 'Germany'. Return airline name and number of flights.

Note: As in the previous question, here too we present two different ways of solving that lead to similar, but not identical, results for the simple reason that the IATA codes consider subsidiaries as different entities, while the IDs do not.

Solution 1

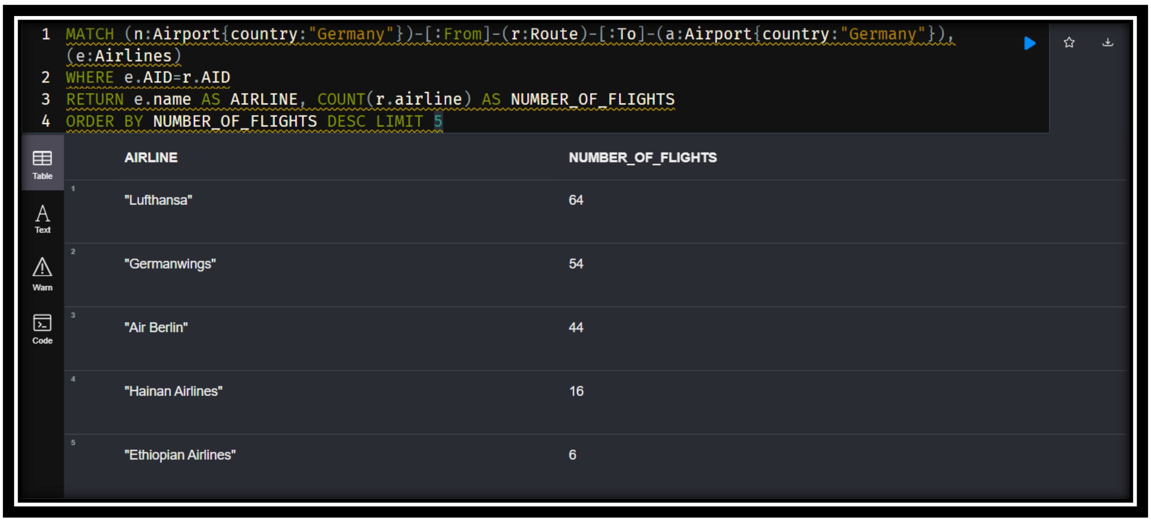


```
MATCH (n:Airport{country:"Germany"})-[:From]-(r:Route)-[:To]-(a:Airport{country:"Germany"}),
(e:Airlines)
WHERE e.IATA=r.airline
RETURN e.name AS AIRLINE, COUNT(r.airline) AS NUMBER_OF_FLIGHTS
ORDER BY NUMBER_OF_FLIGHTS DESC LIMIT 5
```

AIRLINE	NUMBER_OF_FLIGHTS
"Lufthansa Cargo"	64
"Lufthansa"	64
"Germanwings"	54
"Air Berlin"	44
"Hainan Airlines"	16

Figure 16 : Query of question No4 (first way)

Solution 2



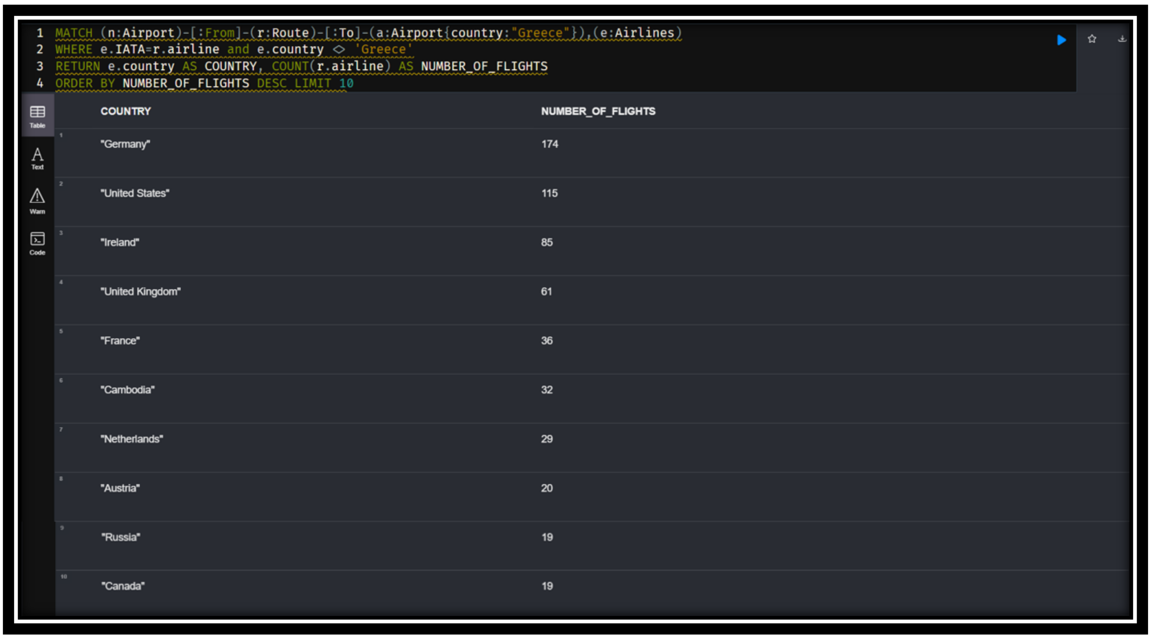
```
MATCH (n:Airport{country:"Germany"})-[:From]-(r:Route)-[:To]-(a:Airport{country:"Germany"}),
(e:Airlines)
WHERE e.AID=r.AID
RETURN e.name AS AIRLINE, COUNT(r.airline) AS NUMBER_OF_FLIGHTS
ORDER BY NUMBER_OF_FLIGHTS DESC LIMIT 5
```

	AIRLINE	NUMBER_OF_FLIGHTS
1	"Lufthansa"	64
2	"Germanwings"	54
3	"Air Berlin"	44
4	"Hainan Airlines"	16
5	"Ethiopian Airlines"	6

Figure 17 : Query of question No4 (second way)

5) Which are the top 10 countries with flights to Greece. Return country name and number of flights.

Solution



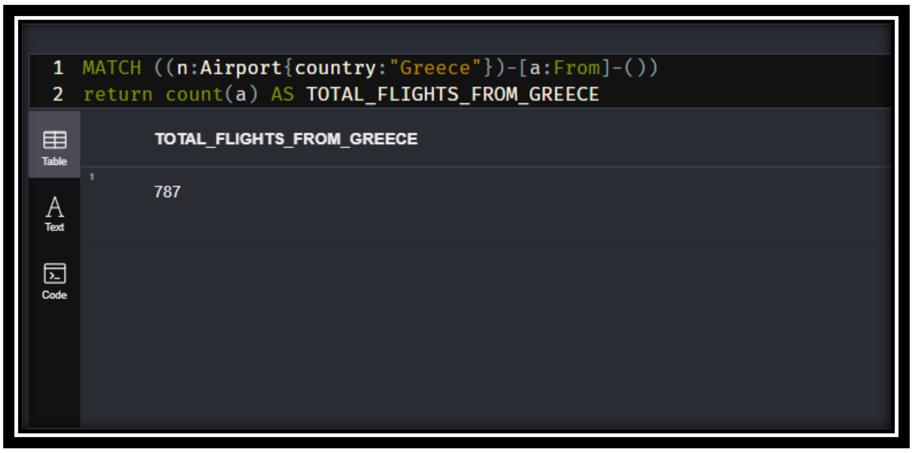
```
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport{country:"Greece"}),(e:Airlines)
WHERE e.IATA=r.airline and e.country <> 'Greece'
RETURN e.country AS COUNTRY, COUNT(r.airline) AS NUMBER_OF_FLIGHTS
ORDER BY NUMBER_OF_FLIGHTS DESC LIMIT 10
```

	COUNTRY	NUMBER_OF_FLIGHTS
1	"Germany"	174
2	"United States"	115
3	"Ireland"	85
4	"United Kingdom"	61
5	"France"	36
6	"Cambodia"	32
7	"Netherlands"	29
8	"Austria"	20
9	"Russia"	19
10	"Canada"	19

Figure 18 : Query of question No5

6)Find the percentage of air traffic (inbound and outbound) for every city in Greece. Return city name and the corresponding traffic percentage in descending order.

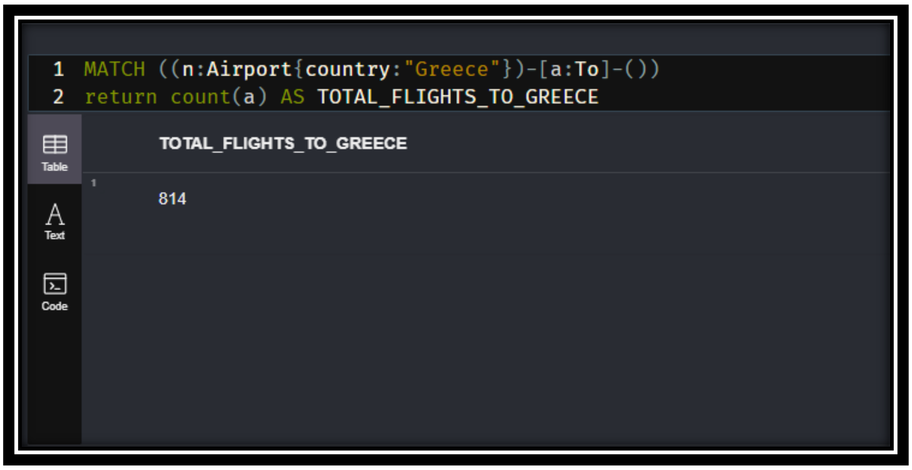
Note: In order to implement the query No 6, we created two extra queries to compute the total flights in Greece. The total flights in Greece will help us to compute the air traffic percentage , as their sum is the denominator of our fraction. The total is **1601** flights in Greece. The queries can be seen below in the figures 19 ,20 and the main query in figure 21 **:**



```
MATCH ((n:Airport{country:"Greece"})-[a:From]-())
return count(a) AS TOTAL_FLIGHTS_FROM_GREECE
```

	TOTAL_FLIGHTS_FROM_GREECE
1	787

Figure 19 : Query to compute the total flights from Greece

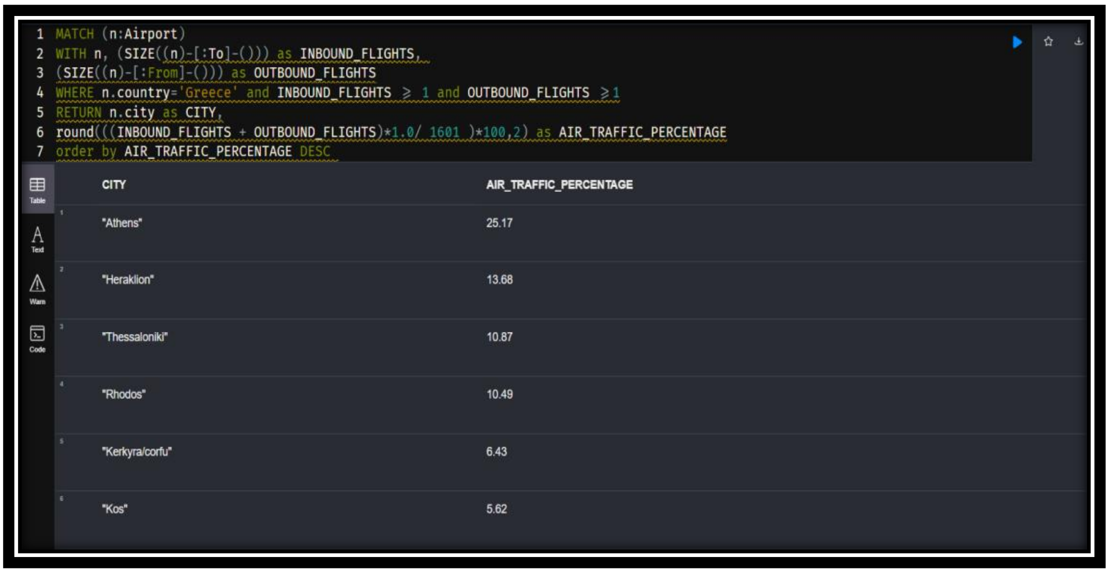


```
MATCH ((n:Airport{country:"Greece"})-[a:To]-())
return count(a) AS TOTAL_FLIGHTS_TO_GREECE
```

	TOTAL_FLIGHTS_TO_GREECE
1	814

Figure 20 : Query to compute the total flights to Greece

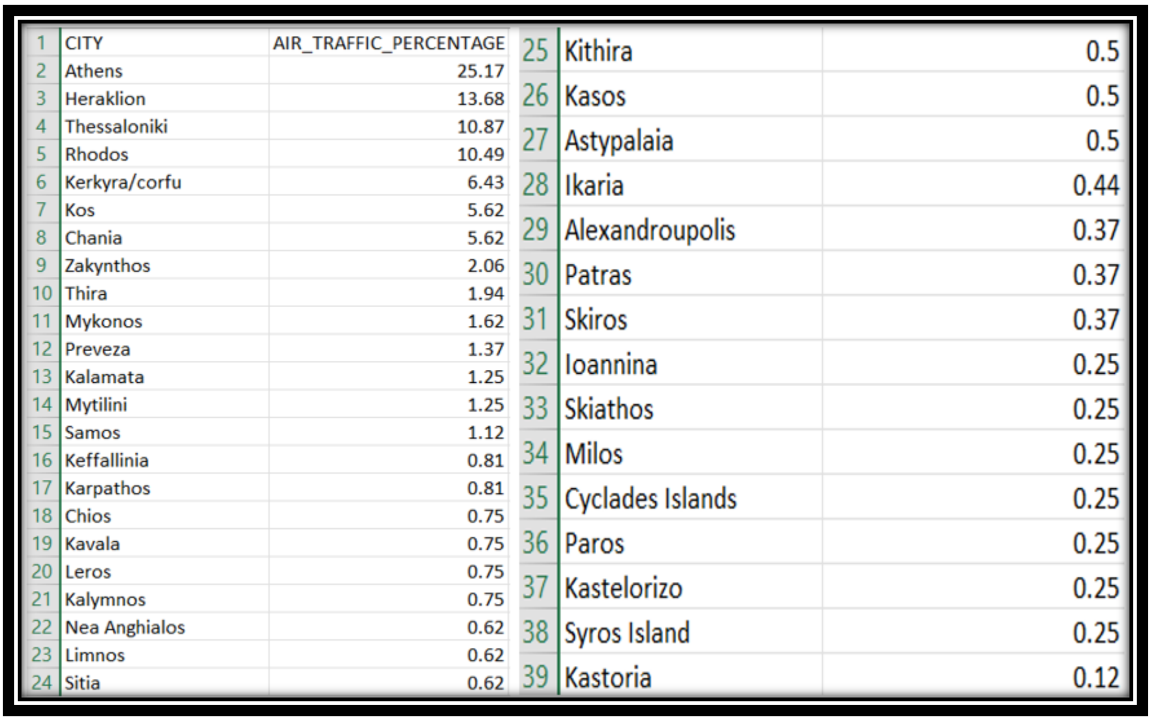
Main Query



```
MATCH (n:Airport)
WITH n, (SIZE((n)-[:To]-())) as INBOUND_FLIGHTS,
(SIZE((n)-[:From]-())) as OUTBOUND_FLIGHTS
WHERE n.country='Greece' and INBOUND_FLIGHTS ≥ 1 and OUTBOUND_FLIGHTS ≥1
RETURN n.city as CITY,
round(((INBOUND_FLIGHTS + OUTBOUND_FLIGHTS)*1.0/ 1601 )*100,2) as AIR_TRAFFIC_PERCENTAGE
order by AIR_TRAFFIC_PERCENTAGE DESC
```

	CITY	AIR_TRAFFIC_PERCENTAGE
1	"Athens"	25.17
2	"Heraklion"	13.68
3	"Thessaloniki"	10.87
4	"Rhodos"	10.49
5	"Kerkyra/corfu"	6.43
6	"Kos"	5.62

Figure 21 : Query of question No6

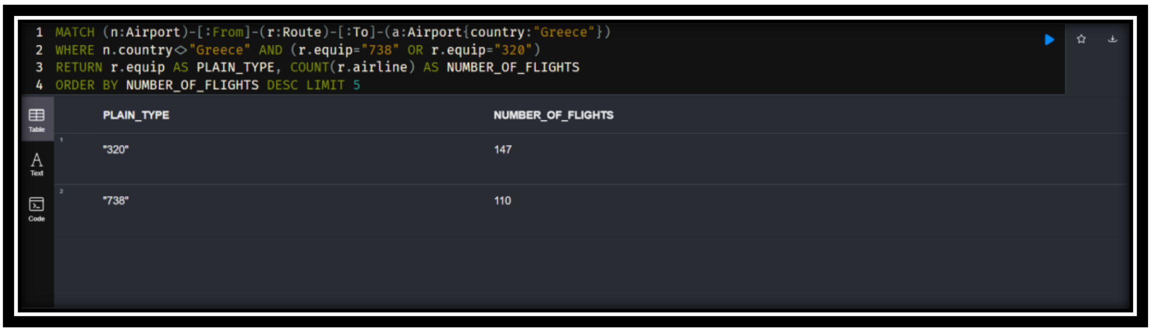


1	CITY	AIR_TRAFFIC_PERCENTAGE
2	Athens	25.17
3	Heraklion	13.68
4	Thessaloniki	10.87
5	Rhodos	10.49
6	Kerkyra/corfu	6.43
7	Kos	5.62
8	Chania	5.62
9	Zakynthos	2.06
10	Thira	1.94
11	Mykonos	1.62
12	Preveza	1.37
13	Kalamata	1.25
14	Mytilini	1.25
15	Samos	1.12
16	Keffallinia	0.81
17	Karpathos	0.81
18	Chios	0.75
19	Kavala	0.75
20	Leros	0.75
21	Kalymnos	0.75
22	Nea Anghialos	0.62
23	Limnos	0.62
24	Sitia	0.62
25	Kithira	0.5
26	Kasos	0.5
27	Astypalaia	0.5
28	Ikaria	0.44
29	Alexandroupolis	0.37
30	Patras	0.37
31	Skiros	0.37
32	Ioannina	0.25
33	Skiathos	0.25
34	Milos	0.25
35	Cyclades Islands	0.25
36	Paros	0.25
37	Kastelorizo	0.25
38	Syros Island	0.25
39	Kastoria	0.12

Figure 22 : Complete table that contains every Greek city and its corresponding traffic percentage

7) Find the number of international flights to Greece with plane types '738' and '320'. Return for each plane type the number of flights.

<u>Solution</u>



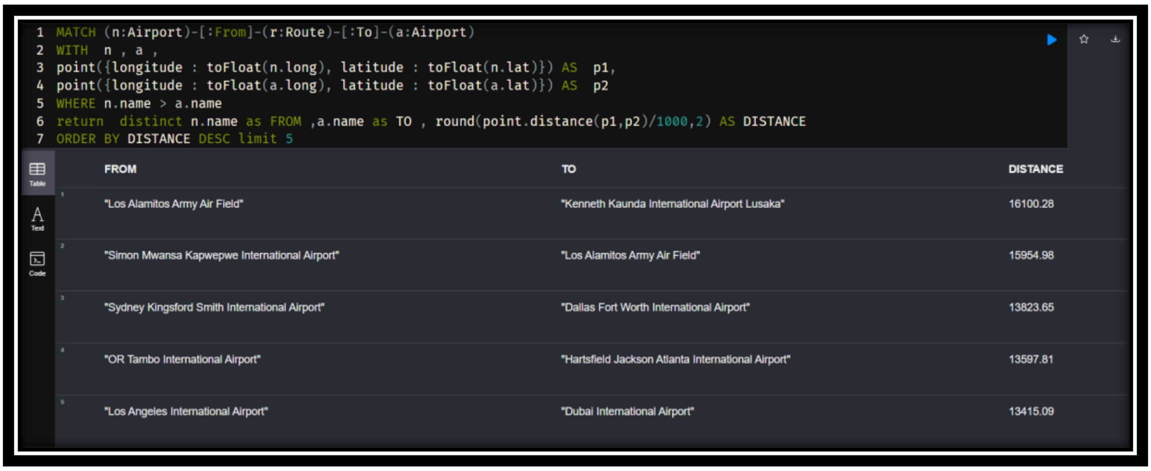
```
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport{country:"Greece"})
WHERE n.country<>"Greece" AND (r.equip="738" OR r.equip="320")
RETURN r.equip AS PLAIN_TYPE, COUNT(r.airline) AS NUMBER_OF_FLIGHTS
ORDER BY NUMBER_OF_FLIGHTS DESC LIMIT 5
```

	PLAIN_TYPE	NUMBER_OF_FLIGHTS
1	"320"	147
2	"738"	110

Figure 23 : Query of question No7

8) Which are the top 5 flights that cover the biggest distance between two airports (use function point({ longitude: s1.longitude, latitude: s1.latitude }) and function distance(point1, point2)). Return From (airport) To (airport) and distance in km.

Solution



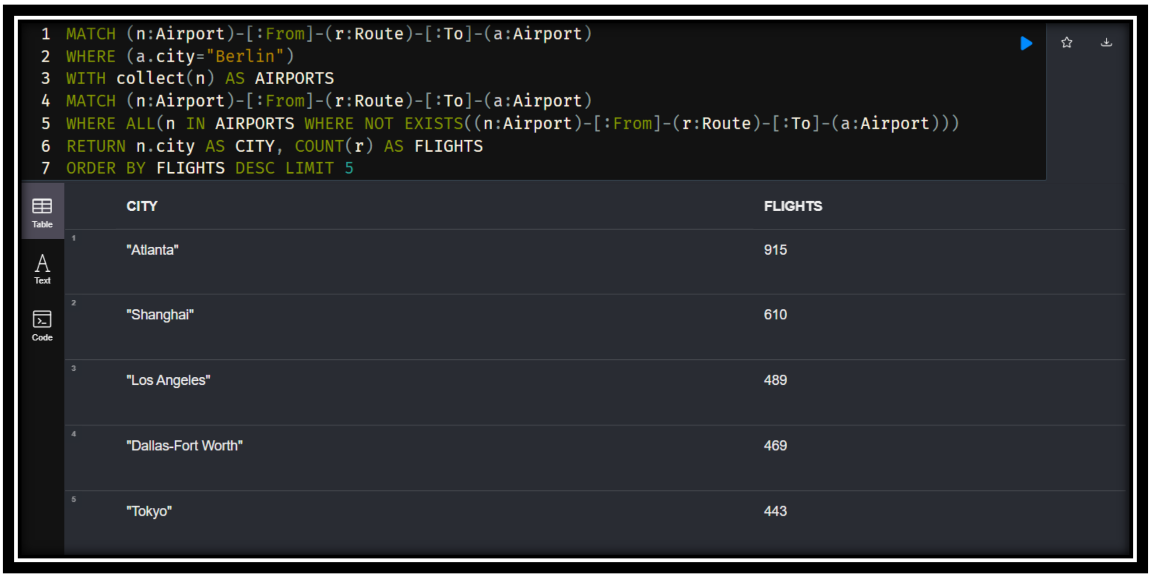
```
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport)
WITH  n , a ,
point({longitude : toFloat(n.long), latitude : toFloat(n.lat)}) AS  p1,
point({longitude : toFloat(a.long), latitude : toFloat(a.lat)}) AS  p2
WHERE n.name > a.name
return  distinct n.name as FROM ,a.name as TO , round(point.distance(p1,p2)/1000,2) AS DISTANCE
ORDER BY DISTANCE DESC limit 5
```

	FROM	TO	DISTANCE
1	"Los Alamitos Army Air Field"	"Kenneth Kaunda International Airport Lusaka"	16100.28
2	"Simon Mwansa Kapwepwe International Airport"	"Los Alamitos Army Air Field"	15954.98
3	"Sydney Kingsford Smith International Airport"	"Dallas Fort Worth International Airport"	13823.65
4	"OR Tambo International Airport"	"Hartsfield Jackson Atlanta International Airport"	13597.81
5	"Los Angeles International Airport"	"Dubai International Airport"	13415.09

Figure 24 : Query of question No8

9) Find 5 cities that are not connected with direct flights to 'Berlin'. Score the cities in descending order with the total number of flights to other destinations. Return city name and score.

Solution



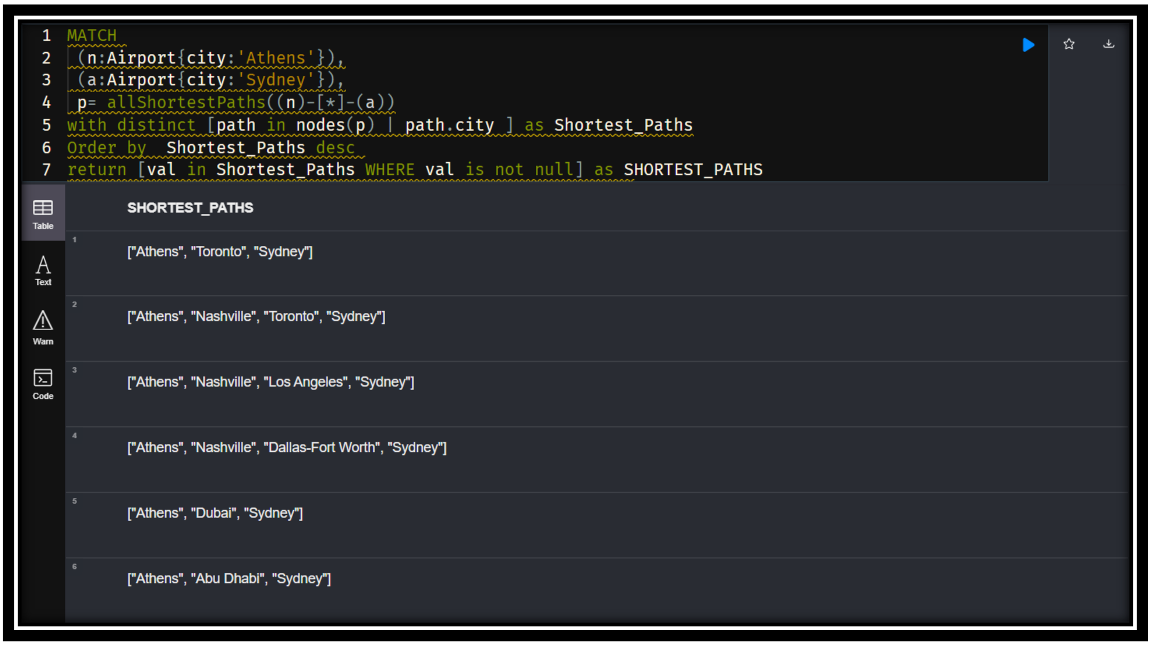
```
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport)
WHERE (a.city="Berlin")
WITH collect(n) AS AIRPORTS
MATCH (n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport)
WHERE ALL(n IN AIRPORTS WHERE NOT EXISTS((n:Airport)-[:From]-(r:Route)-[:To]-(a:Airport)))
RETURN n.city AS CITY, COUNT(r) AS FLIGHTS
ORDER BY FLIGHTS DESC LIMIT 5
```

	CITY	FLIGHTS
1	"Atlanta"	915
2	"Shanghai"	610
3	"Los Angeles"	489
4	"Dallas-Fort Worth"	469
5	"Tokyo"	443

Figure 25 : Query of question No9

10) Find all shortest paths from ‘Athens’ to ‘Sydney’. Use only relations between flights and city airports.

Solution



```
MATCH
 (n:Airport{city:'Athens'}),
 (a:Airport{city:'Sydney'}),
 p= allShortestPaths((n)-[*]-(a))
with distinct [path in nodes(p) | path.city ] as Shortest_Paths
Order by  Shortest_Paths desc
return [val in Shortest_Paths WHERE val is not null] as SHORTEST_PATHS
```

	SHORTEST_PATHS
1	["Athens", "Toronto", "Sydney"]
2	["Athens", "Nashville", "Toronto", "Sydney"]
3	["Athens", "Nashville", "Los Angeles", "Sydney"]
4	["Athens", "Nashville", "Dallas-Fort Worth", "Sydney"]
5	["Athens", "Dubai", "Sydney"]
6	["Athens", "Abu Dhabi", "Sydney"]

Figure 26 : Query of question No10