Assignment II:
Neo4j Graph database

Names - Student IDs: Panagiota Gkourioti - P2822109

Marinos Kandylakis - P2822117

Thaleia Koletsi - P2822120

Course: Mining Big Datasets – Spring 2022

Instructors: Y. Kotidis, I. Filippidou

Athens University of Economics and Business

M.Sc. Program in Business Analytics

Contents

1. Dataset

The dataset used for the current project is a subset of OpenFlights Airports network, which contains information about the world's airline network, including airports, airlines and flights between airports. In particular, the dataset contains 7698 Airports, 6161 Airlines, 6956 Cities, 237 Countries and 65935 Flights between Airports.

The data is split into 3 csv files:

- **Airports.csv:** contains information regarding the airports, specifically, the features used for the analysis are the Airport ID, the name of the airport, the main city served by airport, the country where the airport is located, IATA/ICAO codes, latitude and longitude.

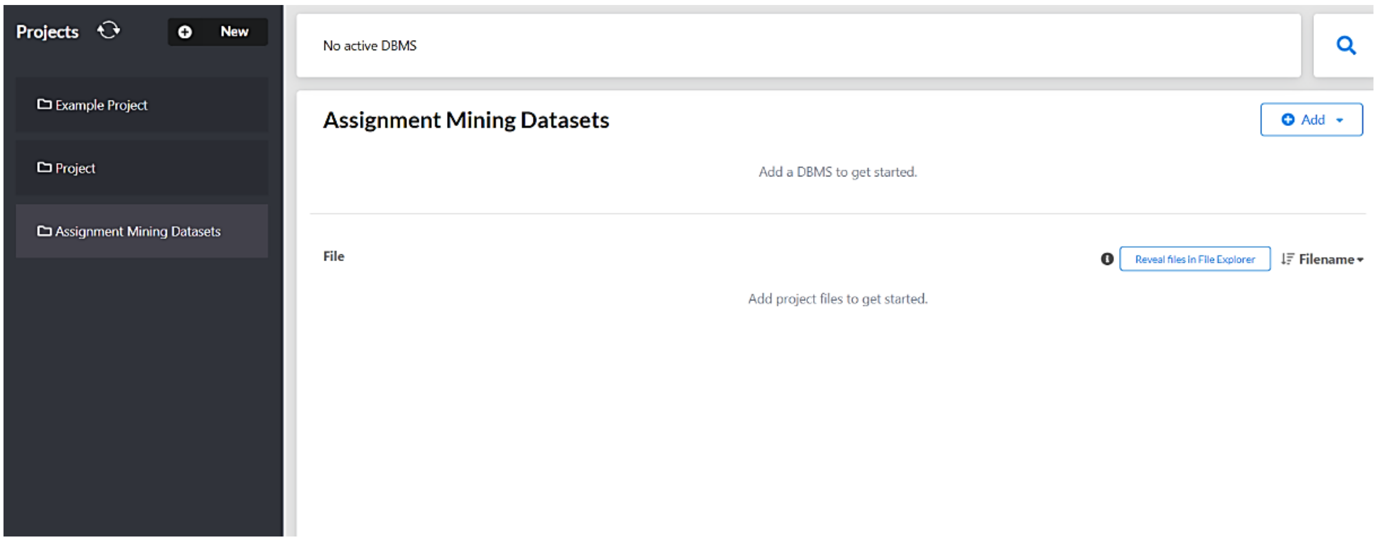
- **Airlines.csv:** contains information regarding the airlines, specifically, the features used for the analysis are the Airline ID, the name of the airline, IATA/ICAO codes, the country where the airport is located and whether the airline is or has until recently been operational (active).

- **Routes.csv:** contains information regarding the routes performed by the airlines and from one airport to another. Routes are directional: if an airline operates services from A to B and from B to A, both A-B and B-A are listed separately. The features included in the analysis are the Airline ID, the Source and Destination Airports (IATA/ICAO code), the Source and Destination Airport IDs, the number of stops and the plane types generally used on this flight.

2. Importing the dataset into Neo4j

The steps followed to create the graph database in Neo4j and import the dataset are presented below:

✓ We initially create a new project named “Assignment Mining Datasets”.



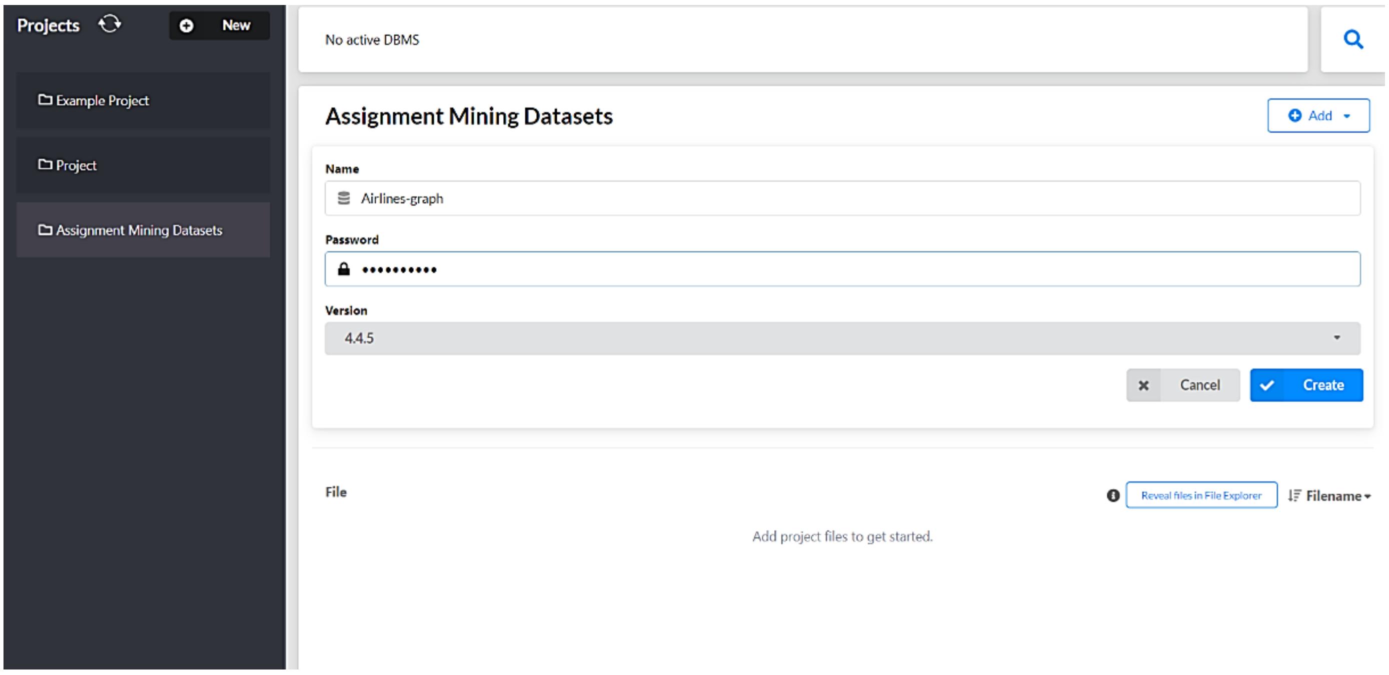
Assignment Mining Datasets

Add a DBMS to get started.

Add project files to get started.

Figure 2.1.: Neo4j - Create a new project

✓ We then add a Local Database Management System, called “Airlines_graph”.



Assignment Mining Datasets

Name: Airlines-graph

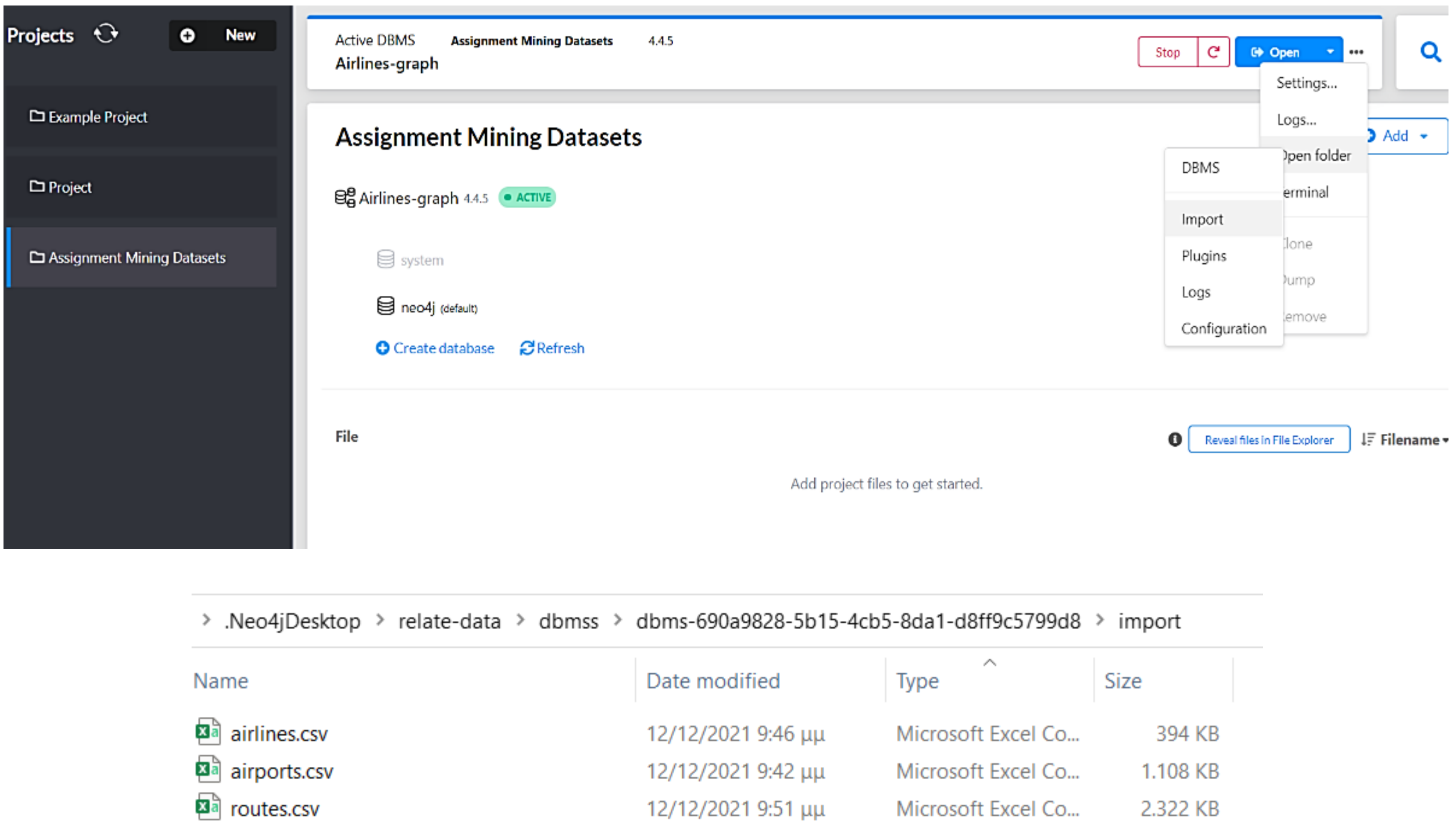
Password: ••••••••••

Version: 4.4.5

Add project files to get started.

Figure 2.2.: Neo4j – Add a local DBMS

✓ Having created the DBMS “Airlines_graph”, we start running it and proceed to import the csv files.



› .Neo4jDesktop › relate-data › dbmss › dbms-690a9828-5b15-4cb5-8da1-d8ff9c5799d8 › import

Name	Date modified	Type	Size
airlines.csv	12/12/2021 9:46 μμ	Microsoft Excel Co...	394 KB
airports.csv	12/12/2021 9:42 μμ	Microsoft Excel Co...	1.108 KB
routes.csv	12/12/2021 9:51 μμ	Microsoft Excel Co...	2.322 KB

Figure 2.3.: Neo4j - Start DBMS and import csv files

✓ We open the Neo4j Browser to load the CSVs and continue with the creation of the graph.

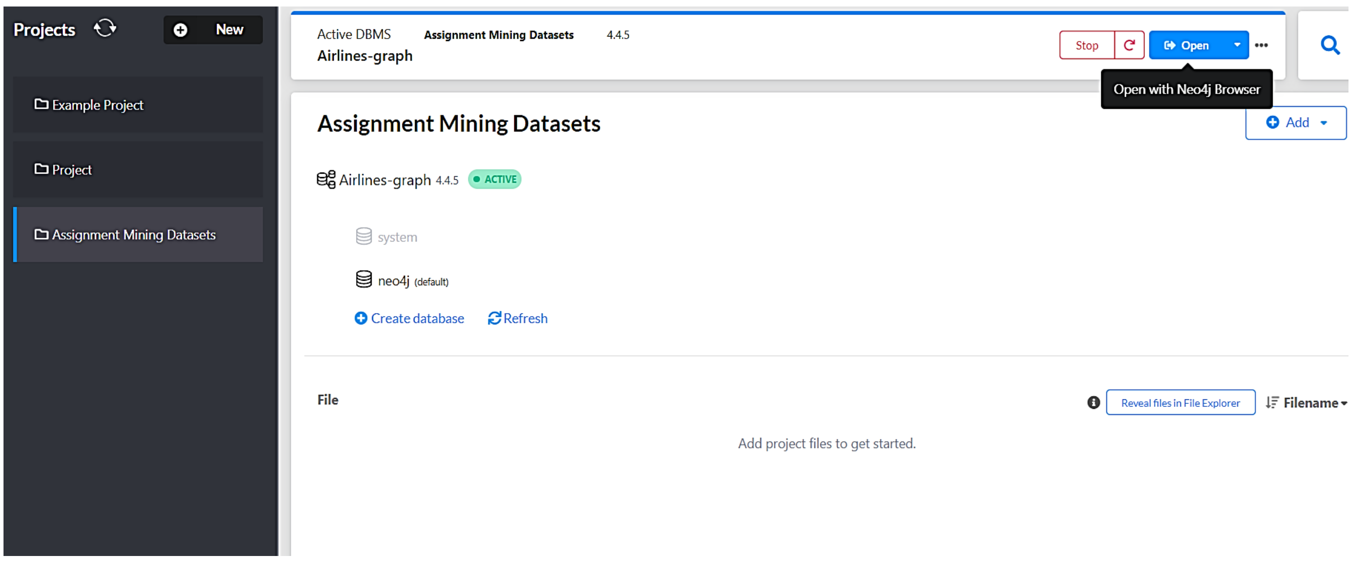


Figure 2.4.: Neo4j – Open the Neo4j Browser

3. Property graph model

The following figure indicates the graph representation of the designed model, that includes the **nodes** and the **relationships** between them. The idea is to create four main nodes. Specifically, in the graph there are two different nodes for the "Airports", which are connected via a "Route" node as well as the node "Airlines", which identifies by which airline a route was performed. We also create three relationships to connect them, namely, "Departs_from" one airport, "Arrives_to" another airport and route "Performed_by" airline.

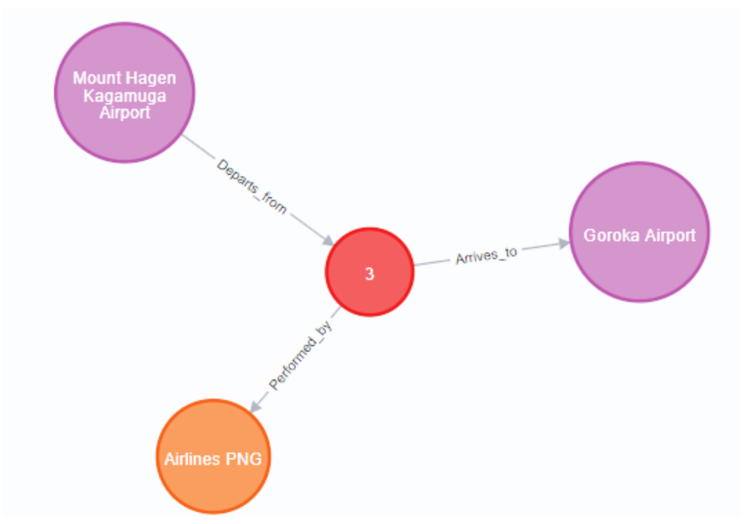
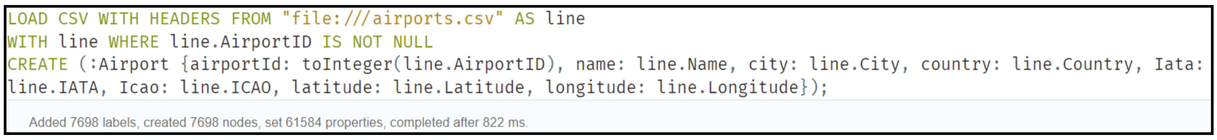


Figure 3.1.: Graph model example

3.1. Nodes creation

For the creation of the first node "Airports", we initially load the airports.csv and from the available attributes we only keep the *name* of the airport, the *city*, the *country*, the *IATA*, which is a three-letter code for the airports, the *ICAO* which is a four-letter code for the airports, and the *location coordinates*. The rest of the attributes are not necessary for the analysis. Thus, excluding the airport ids which are null, we execute the below query and 7,698 airport nodes are created.

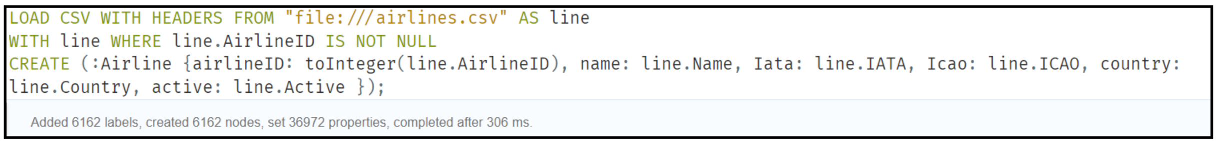


```
LOAD CSV WITH HEADERS FROM "file:///airports.csv" AS line
WITH line WHERE line.AirportID IS NOT NULL
CREATE (:Airport {airportId: toInteger(line.AirportID), name: line.Name, city: line.City, country: line.Country, Iata:
line.IATA, Icao: line.ICAO, latitude: line.Latitude, longitude: line.Longitude});
```

```
Added 7698 labels, created 7698 nodes, set 61584 properties, completed after 822 ms.
```

The working process for the creation of the other nodes, “Airlines” and “Routes”, is identical.

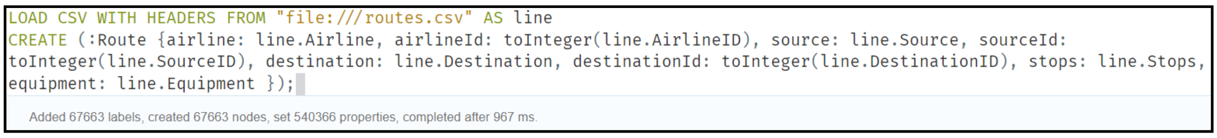
With regards to “Airlines”, the required attributes are the *name* of the airline, the *IATA* and *ICAO* codes, the *country* and the description “*active*”, which indicates if the airline is or has until recently been operational or not. So, after the execution of the following query 6,162 airline nodes are created.



```
LOAD CSV WITH HEADERS FROM "file:///airlines.csv" AS line
WITH line WHERE line.AirlineID IS NOT NULL
CREATE (:Airline {airlineID: toInteger(line.AirlineID), name: line.Name, Iata: line.IATA, Icao: line.ICAO, country:
line.Country, active: line.Active });
```

```
Added 6162 labels, created 6162 nodes, set 36972 properties, completed after 306 ms.
```

Concerning “Routes”, from the corresponding csv file we keep all provided attributes except from *codeshare* that is unnecessary for the graph modeling. So, after the execution of the following query, 67,663 route nodes are created.

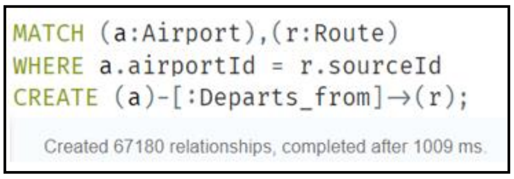


```
LOAD CSV WITH HEADERS FROM "file:///routes.csv" AS line
CREATE (:Route {airline: line.Airline, airlineId: toInteger(line.AirlineID), source: line.Source, sourceId:
toInteger(line.SourceID), destination: line.Destination, destinationId: toInteger(line.DestinationID), stops: line.Stops,
equipment: line.Equipment });
```

```
Added 67663 labels, created 67663 nodes, set 540366 properties, completed after 967 ms.
```

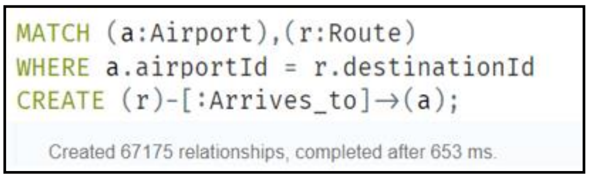
3.2. Relationships creation

Next step of the graph modeling is the design of the relationships between the nodes. As already referred in previous section, the relationships are three. Starting with the airports and routes nodes, we set up the connections *DEPARTS_FROM* and *ARRIVES_TO* as follows, and we observe that 67,180 and 67,175 relationships are created respectively:



```
MATCH (a:Airport),(r:Route)
WHERE a.airportId = r.sourceId
CREATE (a)-[:Departs_from]→(r);
```

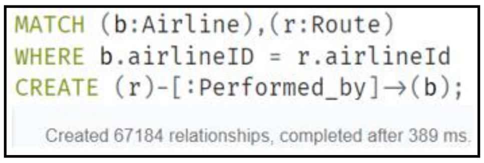
```
Created 67180 relationships, completed after 1009 ms.
```



```
MATCH (a:Airport),(r:Route)
WHERE a.airportId = r.destinationId
CREATE (r)-[:Arrives_to]→(a);
```

```
Created 67175 relationships, completed after 653 ms.
```

Finally, through the *PERFORMED_BY* relationship we link each route node with the relative airline by which it was performed.



```
MATCH (b:Airline),(r:Route)
WHERE b.airlineID = r.airlineId
CREATE (r)-[:Performed_by]→(b);
```

```
Created 67184 relationships, completed after 389 ms.
```

The below figure is a subgraph representation of all the above executions.

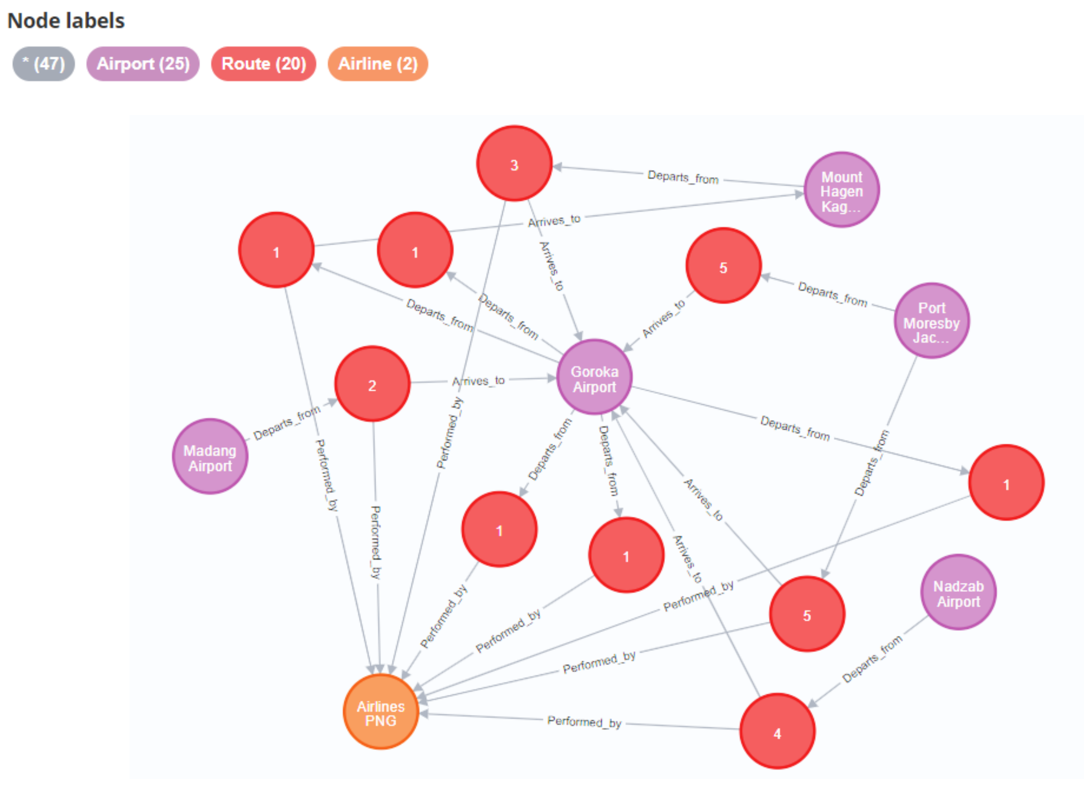


Figure 3.2.: Subgraph of the model example

4. Queries

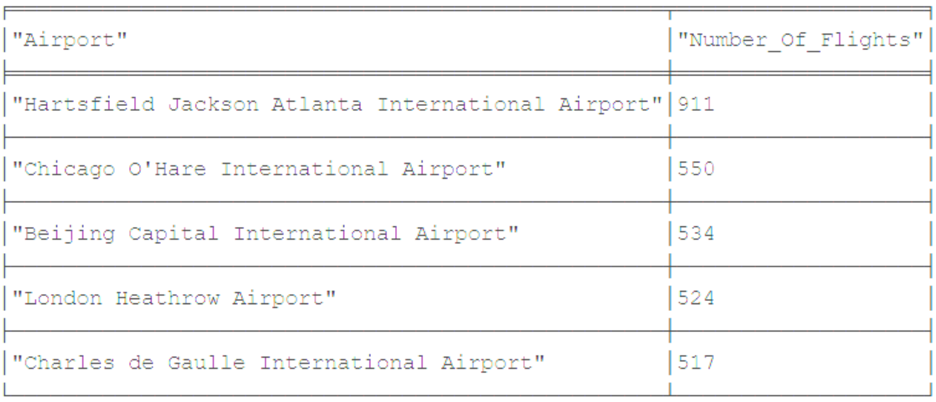
After the creation of our database, we attempt to execute the following queries using the Cypher language.

4.1. Top 5 airports with the most flights

To find the top airports with the most flights, we take all the Airport nodes, count all their inbound/outbound flights and sort them in descending order.

```
MATCH (a:Airport)
WITH a, SIZE((a)←[:Departs_from|Arrives_to]-()) as Number_Of_Flights
RETURN a.name AS Airport, Number_Of_Flights
ORDER BY Number_Of_Flights
DESC LIMIT 5;
```

The Top 5 airports with the most flights are:



```
│"Airport"                                   │"Number_Of_Flights"│
├────────────────────────────────────────────┼───────────────────┤
│"Hartsfield Jackson Atlanta International Airport"│911          │
├────────────────────────────────────────────┼───────────────────┤
│"Chicago O'Hare International Airport"      │550                │
├────────────────────────────────────────────┼───────────────────┤
│"Beijing Capital International Airport"     │534                │
├────────────────────────────────────────────┼───────────────────┤
│"London Heathrow Airport"                   │524                │
├────────────────────────────────────────────┼───────────────────┤
│"Charles de Gaulle International Airport"   │517                │
```

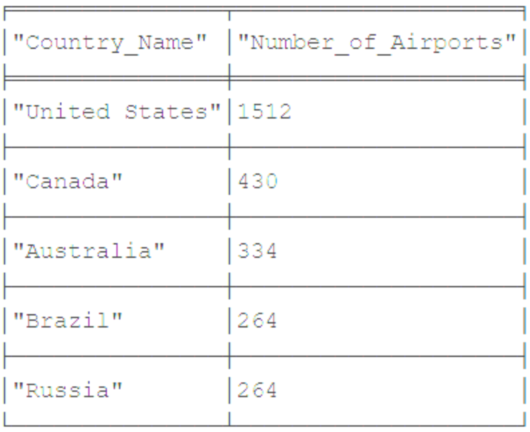
The ‘Hartsfield Jackson Atlanta International Airport’ has the most flights (incoming/outgoing), up to 911.

4.2. Top 5 countries with the most airports

For this query, we count all airports grouping by the country they are located at and extract the top 5 countries according to the highest number of airports.

```
MATCH (a:Airport)
RETURN a.country AS Country_Name, COUNT(a) AS Number_of_Airports
ORDER BY Number_of_Airports
DESC LIMIT 5;
```

The Top 5 countries with the most airports are:

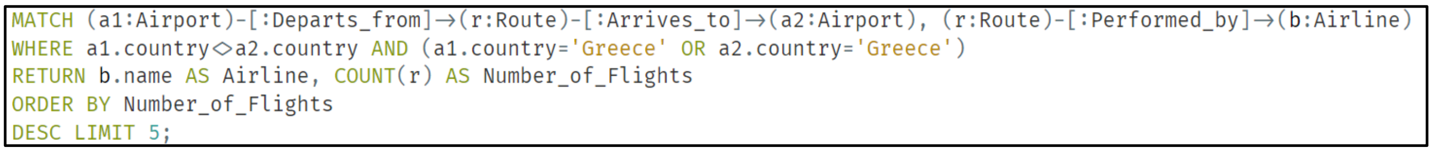


```
│"Country_Name" │"Number_of_Airports"│
├───────────────┼────────────────────┤
│"United States"│1512                │
├───────────────┼────────────────────┤
│"Canada"       │430                 │
├───────────────┼────────────────────┤
│"Australia"    │334                 │
├───────────────┼────────────────────┤
│"Brazil"       │264                 │
├───────────────┼────────────────────┤
│"Russia"       │264                 │
```

We observe that United States has almost 4 times the Airports of Canada and 7 times the Airports of Russia found in this dataset.

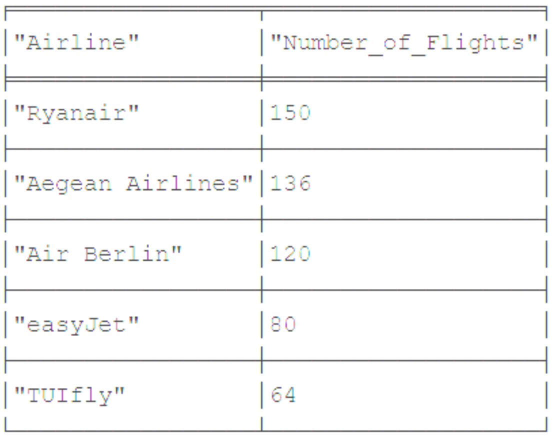
4.3. Top 5 airlines with international flights from/to ‘Greece’

On the below query we match all the inbound/outbound flights of Greece excluding the internal flights of Greece, as well as the airlines performing them, in order to find the top 5 airlines conducting flights from/to Greece.



```
MATCH (a1:Airport)-[:Departs_from]->(r:Route)-[:Arrives_to]->(a2:Airport), (r:Route)-[:Performed_by]->(b:Airline)
WHERE a1.country<>a2.country AND (a1.country='Greece' OR a2.country='Greece')
RETURN b.name AS Airline, COUNT(r) AS Number_of_Flights
ORDER BY Number_of_Flights
DESC LIMIT 5;
```

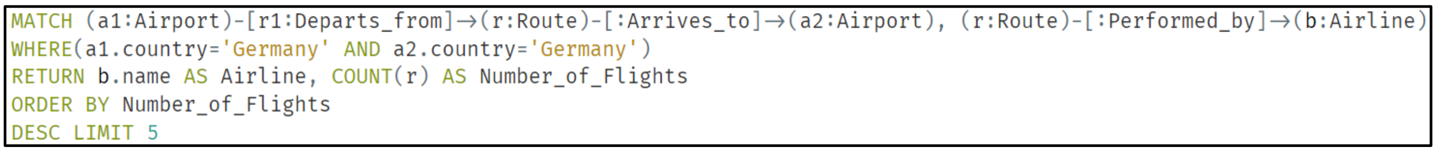
The top 5 airlines with international flights from/to ‘Greece’ are:



```
╒══════════════════╤═══════════════════╕
│"Airline"         │"Number_of_Flights"│
╞══════════════════╪═══════════════════╡
│"Ryanair"         │150                │
├──────────────────┼───────────────────┤
│"Aegean Airlines" │136                │
├──────────────────┼───────────────────┤
│"Air Berlin"      │120                │
├──────────────────┼───────────────────┤
│"easyJet"         │80                 │
├──────────────────┼───────────────────┤
│"TUIfly"          │64                 │
└──────────────────┴───────────────────┘
```

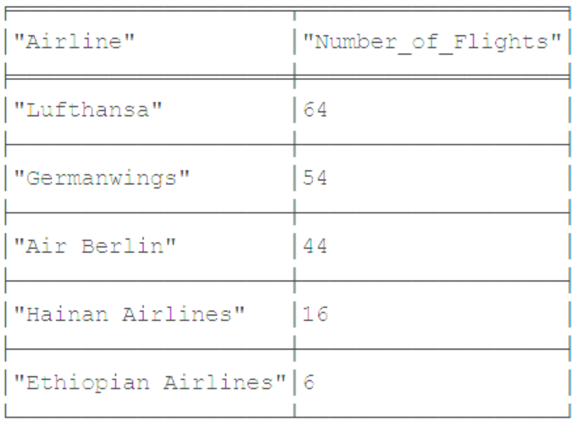
4.4. Top 5 airlines with local flights inside ‘Germany’

Similarly to the previous query, we now attempt to find the airlines conducting the most flights inside Germany.



```
MATCH (a1:Airport)-[r1:Departs_from]->(r:Route)-[:Arrives_to]->(a2:Airport), (r:Route)-[:Performed_by]->(b:Airline)
WHERE(a1.country='Germany' AND a2.country='Germany')
RETURN b.name AS Airline, COUNT(r) AS Number_of_Flights
ORDER BY Number_of_Flights
DESC LIMIT 5
```

The top 5 airlines with local flights inside Germany are the following:

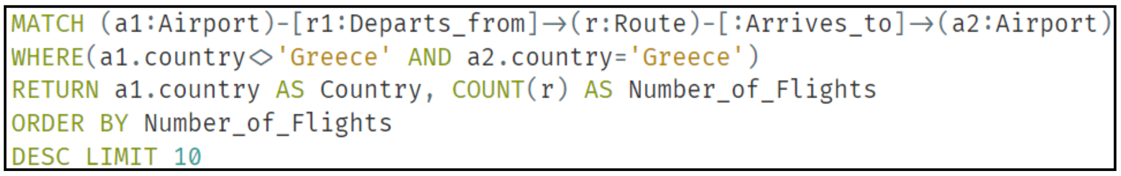


"Airline"	"Number_of_Flights"
"Lufthansa"	64
"Germanwings"	54
"Air Berlin"	44
"Hainan Airlines"	16
"Ethiopian Airlines"	6

We can observe that Lufthansa has the most flights inside Germany as it was expected since its one of the biggest airlines in the world.

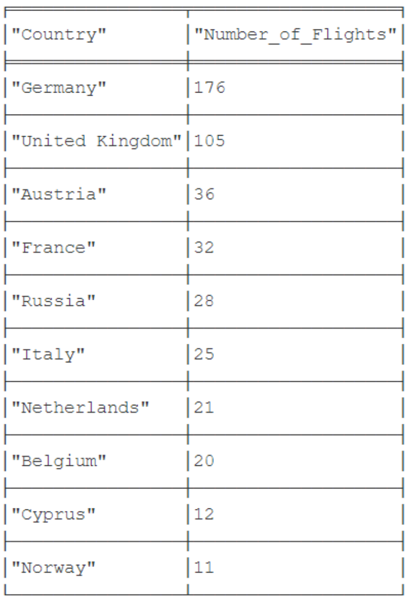
4.5. Top 10 countries with flights to Greece

To find the top 10 countries with flights to Greece, we work as per below. Using the *match* command, we search for the pattern inside of the command and by setting specific criteria for the two countries of the airports, we retrieve the necessary information. Finally, we sort the results of the query in descending order.



```
MATCH (a1:Airport)-[r1:Departs_from]->(r:Route)-[:Arrives_to]->(a2:Airport)
WHERE(a1.country<>'Greece' AND a2.country='Greece')
RETURN a1.country AS Country, COUNT(r) AS Number_of_Flights
ORDER BY Number_of_Flights
DESC LIMIT 10
```

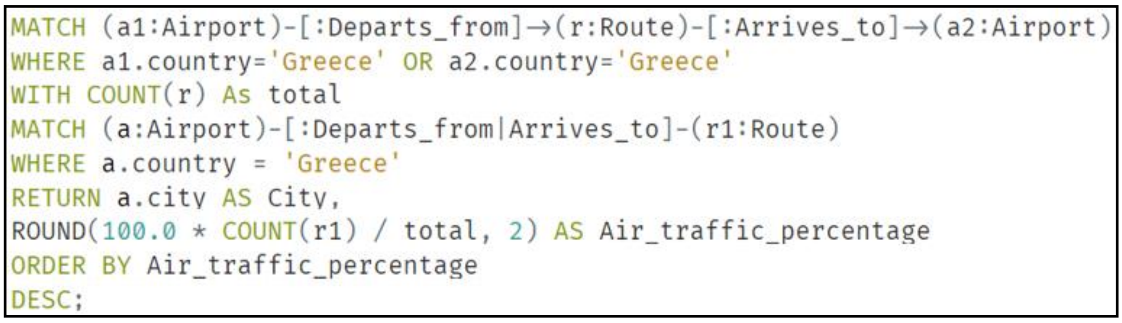
The Top 10 countries with flights to Greece are:



"Country"	"Number_of_Flights"
"Germany"	176
"United Kingdom"	105
"Austria"	36
"France"	32
"Russia"	28
"Italy"	25
"Netherlands"	21
"Belgium"	20
"Cyprus"	12
"Norway"	11

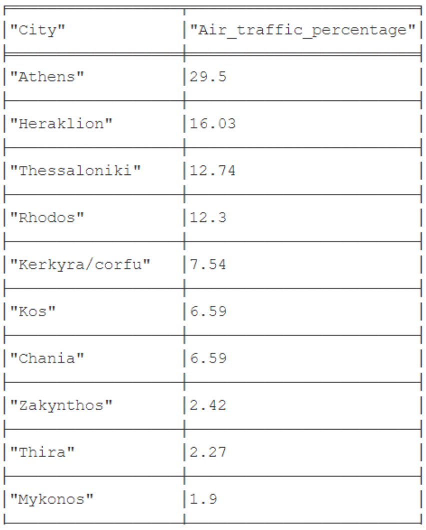
4.6. Air traffic of cities

In order to find the percentage of air traffic (inbound/outbound) for every city in Greece, we should create two sequential queries. In the first one, the total number of flights (inbound/outbound) of Greece is calculated and then by using *with* command, which allows us to pipe the results from one query to another, we save this number to *total* variable. In the second query, we count the flights for every city of Greece and then we calculate the percentage of air traffic for each of those by dividing the flights with the *total* variable.

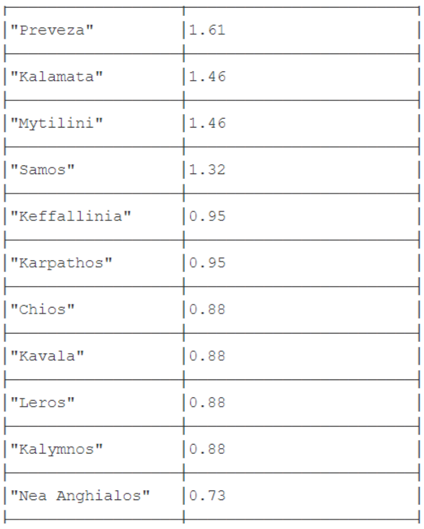
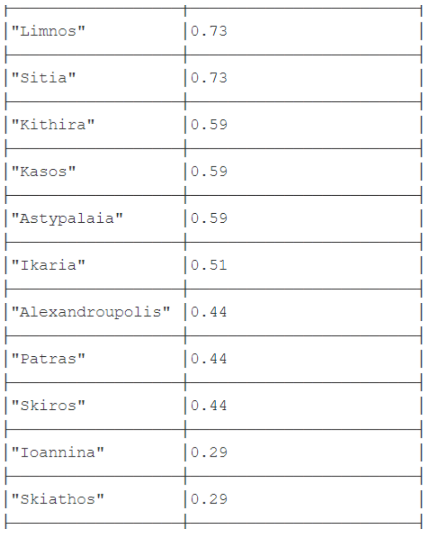
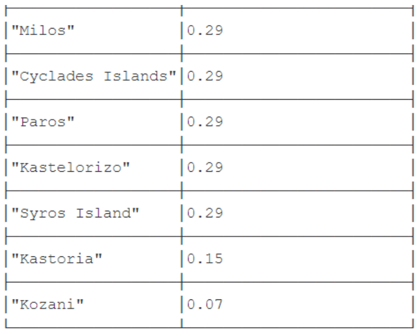


```
MATCH (a1:Airport)-[:Departs_from]→(r:Route)-[:Arrives_to]→(a2:Airport)
WHERE a1.country='Greece' OR a2.country='Greece'
WITH COUNT(r) As total
MATCH (a:Airport)-[:Departs_from|Arrives_to]-(r1:Route)
WHERE a.country = 'Greece'
RETURN a.city AS City,
ROUND(100.0 * COUNT(r1) / total, 2) AS Air_traffic_percentage
ORDER BY Air_traffic_percentage
DESC;
```

The percentage of air traffic for every Greek city is:



```
╒════════════════╤════════════════════════╕
│"City"          │"Air_traffic_percentage"│
╞════════════════╪════════════════════════╡
│"Athens"        │29.5                    │
├────────────────┼────────────────────────┤
│"Heraklion"     │16.03                   │
├────────────────┼────────────────────────┤
│"Thessaloniki"  │12.74                   │
├────────────────┼────────────────────────┤
│"Rhodos"        │12.3                    │
├────────────────┼────────────────────────┤
│"Kerkyra/corfu" │7.54                    │
├────────────────┼────────────────────────┤
│"Kos"           │6.59                    │
├────────────────┼────────────────────────┤
│"Chania"        │6.59                    │
├────────────────┼────────────────────────┤
│"Zakynthos"     │2.42                    │
├────────────────┼────────────────────────┤
│"Thira"         │2.27                    │
├────────────────┼────────────────────────┤
│"Mykonos"       │1.9                     │
└────────────────┴────────────────────────┘
```

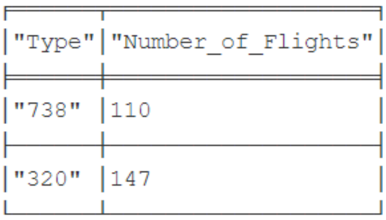
"Preveza"	1.61
"Kalamata"	1.46
"Mytilini"	1.46
"Samos"	1.32
"Keffallinia"	0.95
"Karpathos"	0.95
"Chios"	0.88
"Kavala"	0.88
"Leros"	0.88
"Kalymnos"	0.88
"Nea Anghialos"	0.73
"Limnos"	0.73
"Sitia"	0.73
"Kithira"	0.59
"Kasos"	0.59
"Astypalaia"	0.59
"Ikaria"	0.51
"Alexandroupolis"	0.44
"Patras"	0.44
"Skiros"	0.44
"Ioannina"	0.29
"Skiathos"	0.29
"Milos"	0.29
"Cyclades Islands"	0.29
"Paros"	0.29
"Kastelorizo"	0.29
"Syros Island"	0.29
"Kastoria"	0.15
"Kozani"	0.07

4.7. Plane types

By setting the following criteria, we calculate the number of international flights to Greece with plane types ‘738’ and ‘320’.

```
MATCH (a1:Airport)-[:Departs_from]→(r:Route)-[:Arrives_to]→(a2:Airport)
WHERE a1.country <> 'Greece' AND a2.country = 'Greece' AND r.equipment in ['738','320']
RETURN r.equipment AS Type, COUNT(r) AS Number_of_Flights
```

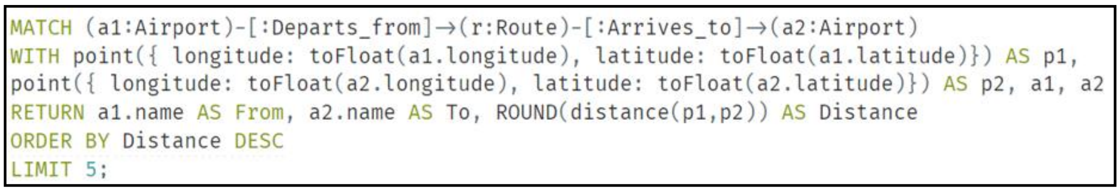
The number of flights per plane type are:



```
╒══════╤═══════════════════╕
│"Type"│"Number_of_Flights"│
╞══════╪═══════════════════╡
│"738" │110                │
├──────┼───────────────────┤
│"320" │147                │
└──────┴───────────────────┘
```

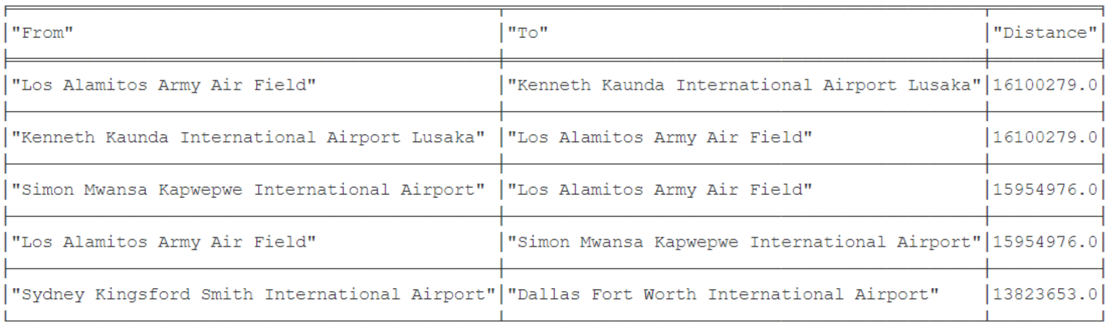
4.8. Top 5 flights between airports with big distance

By using the functions *point* and *distance* (and the relevant location coordinates) we can calculate the geographical distance between the available airports and then, by sorting in descending order, we conclude to the top five flights which cover the biggest distance between two airports.



```
MATCH (a1:Airport)-[:Departs_from]→(r:Route)-[:Arrives_to]→(a2:Airport)
WITH point({ longitude: toFloat(a1.longitude), latitude: toFloat(a1.latitude)}) AS p1,
point({ longitude: toFloat(a2.longitude), latitude: toFloat(a2.latitude)}) AS p2, a1, a2
RETURN a1.name AS From, a2.name AS To, ROUND(distance(p1,p2)) AS Distance
ORDER BY Distance DESC
LIMIT 5;
```

The top 5 flights that connect two airports with the biggest distance are:



```
╒═══════════════════════════════════════════╤═══════════════════════════════════════════╤══════════╕
│"From"                                     │"To"                                       │"Distance"│
╞═══════════════════════════════════════════╪═══════════════════════════════════════════╪══════════╡
│"Los Alamitos Army Air Field"              │"Kenneth Kaunda International Airport Lusaka"│16100279.0│
├───────────────────────────────────────────┼───────────────────────────────────────────┼──────────┤
│"Kenneth Kaunda International Airport Lusaka"│"Los Alamitos Army Air Field"            │16100279.0│
├───────────────────────────────────────────┼───────────────────────────────────────────┼──────────┤
│"Simon Mwansa Kapwepwe International Airport"│"Los Alamitos Army Air Field"            │15954976.0│
├───────────────────────────────────────────┼───────────────────────────────────────────┼──────────┤
│"Los Alamitos Army Air Field"              │"Simon Mwansa Kapwepwe International Airport"│15954976.0│
├───────────────────────────────────────────┼───────────────────────────────────────────┼──────────┤
│"Sydney Kingsford Smith International Airport"│"Dallas Fort Worth International Airport"│13823653.0│
└───────────────────────────────────────────┴───────────────────────────────────────────┴──────────┘
```

4.9. 5 cities that are not connected with direct flights to 'Berlin

By using the function *collect* and the *with* command, we can keep the cities with direct flights to Berlin in a separate variable "Cities". Then, we can calculate the total number of flights by excluding the cities in variable "Cities", to find those that do not have direct flights to Berlin. Finally, by sorting by the number of flights to other destinations in descending order, we conclude to the top five cities not connected with direct flights to Berlin.

```
MATCH (a1:Airport)-[:Departs_from]→(r:Route)-[:Arrives_to]→(a2:Airport)
WHERE a2.city = 'Berlin' AND r.stops = '0'
WITH collect(a1.city) AS Cities
MATCH (a1:Airport)-[:Departs_from]→(r:Route)-[:Arrives_to]→(a2:Airport)
WHERE NOT(a1.city IN Cities)
RETURN a1.city AS City, COUNT(r) AS Number_Of_Flights
ORDER BY COUNT(r) DESC
LIMIT 5;
```

The top five cities not connected with direct flights to Berlin are:

```
╒══════════════════╤═══════════════════╕
│"City"            │"Number_Of_Flights"│
╞══════════════════╪═══════════════════╡
│"Atlanta"         │915                │
├──────────────────┼───────────────────┤
│"Shanghai"        │610                │
├──────────────────┼───────────────────┤
│"Los Angeles"     │489                │
├──────────────────┼───────────────────┤
│"Dallas-Fort Worth"│469               │
├──────────────────┼───────────────────┤
│"Tokyo"           │443                │
└──────────────────┴───────────────────┘
```

4.10. Shortest paths from ‘Athens’ to ‘Sydney’

The final query attempts to find all shortest paths from Athens, Greece to Sydney, Australia, using only relations between flights and city airports. We set a condition to specify the countries and another one to exclude airline nodes.

```
MATCH p = shortestPath((a1:Airport{city:'Athens'})-[*]-(a2:Airport{city:'Sydney'}))
WHERE NONE(n in nodes(p) WHERE n:Airline) AND a1.country = 'Greece'
RETURN a1.name AS Athens_Airport, a2.name AS Sydney_Airport, length(p) AS Path_Length
```

The length of the shortest paths from Athens to Sydney along with the corresponding airports are:

```
╒═════════════════════════╤═════════════════════════╤═════════════╕
│"Athens_Airport"         │"Sydney_Airport"         │"Path_Length"│
╞═════════════════════════╪═════════════════════════╪═════════════╡
│"Eleftherios Venizelos   │"Sydney / J.A. Douglas   │4            │
│International Airport"   │McCurdy Airport"         │             │
├─────────────────────────┼─────────────────────────┼─────────────┤
│"Eleftherios Venizelos   │"Sydney Kingsford Smith  │4            │
│International Airport"   │ International Airport"  │             │
└─────────────────────────┴─────────────────────────┴─────────────┘
```

These paths are also displayed in Figure 4.1., which shows that the shortest path from Sydney to Athens is via Abu Dhabi and from Athens to Sydney is via Toronto.

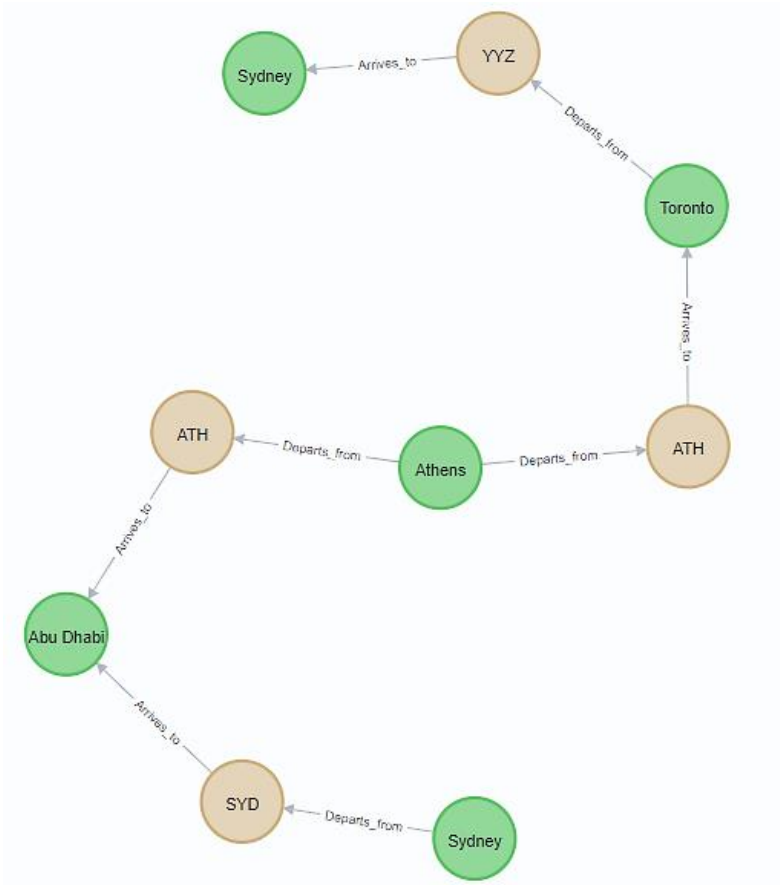


Figure 4.1.: Shortest paths from Athens to Sydney