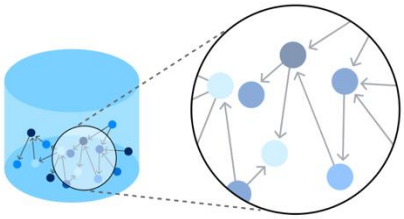
Athens University of Economics and Business
Department of Management Science and Technology

Mining Big Datasets

M.Sc. Business Analytics
Part Time 2021-2023

Neo4j Graph database



Maria Zafeiropoulou
A.M: P2822113
&
Marianna Konstantopoulou
A.M: P2822122

Athens, 03/07/2022

Part I: Importing the datasets and creating the graph

In this assignment we were instructed to represent the data as a property graph by creating the necessary entities and giving the required names, types, and properties. The graph we created for its implementation is fully examined below. We will utilize a diagram to better convey our thoughts:

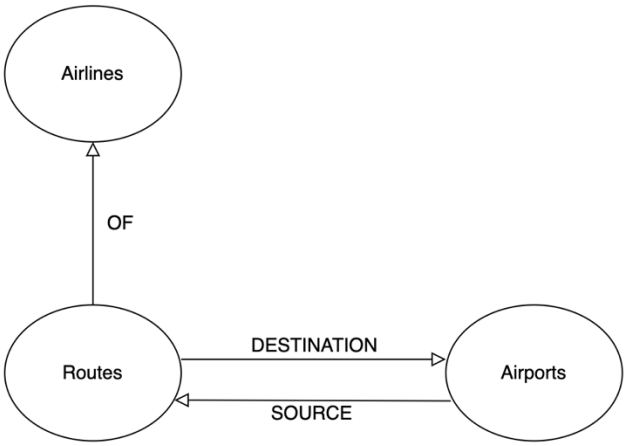


Figure 1: Theoretical graph model chart

After reviewing all the files that define the airline network and represent the necessary properties on nodes and edges of a graph, we decided that it is best to create 3 nodes (Airlines, Airports and Routes) that are connected with the following relationships:

- **Source**: from airport to route
- **Destination**: from route to airport
- **Of**: from route to airline

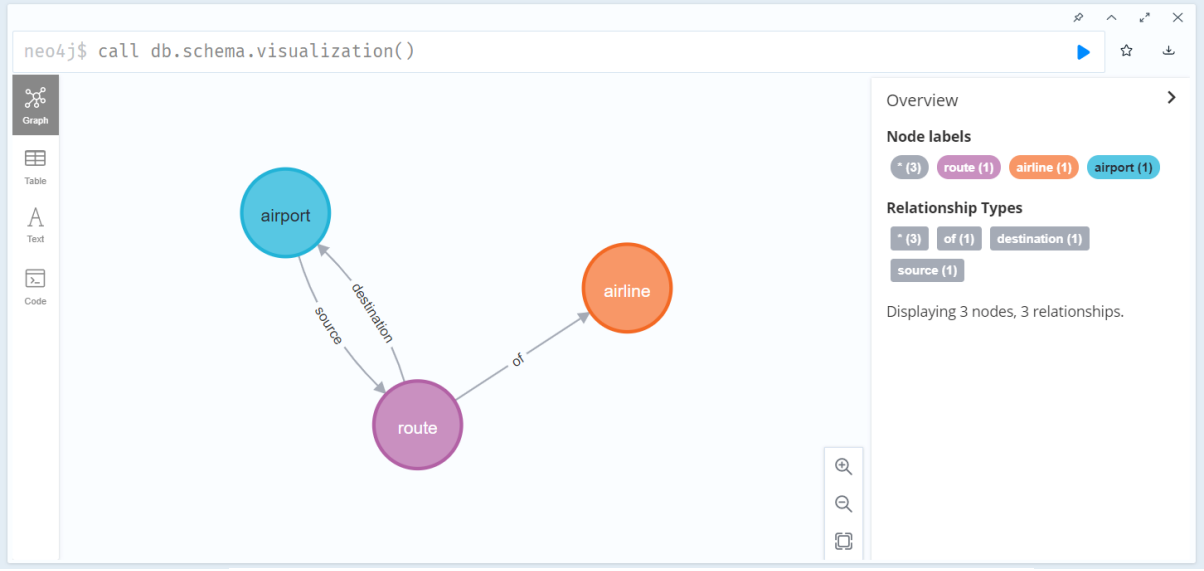


Figure 2: Graph model chart extracted from Neo4j

Firstly, we loaded the CSV files to the Neo4j database.

Figure 3: Loading CSV file for Airlines

Figure 4: Loading CSV file for Airports

Figure 5: Loading CSV file for Routes

Then, we created all nodes indexes (unique constraint).

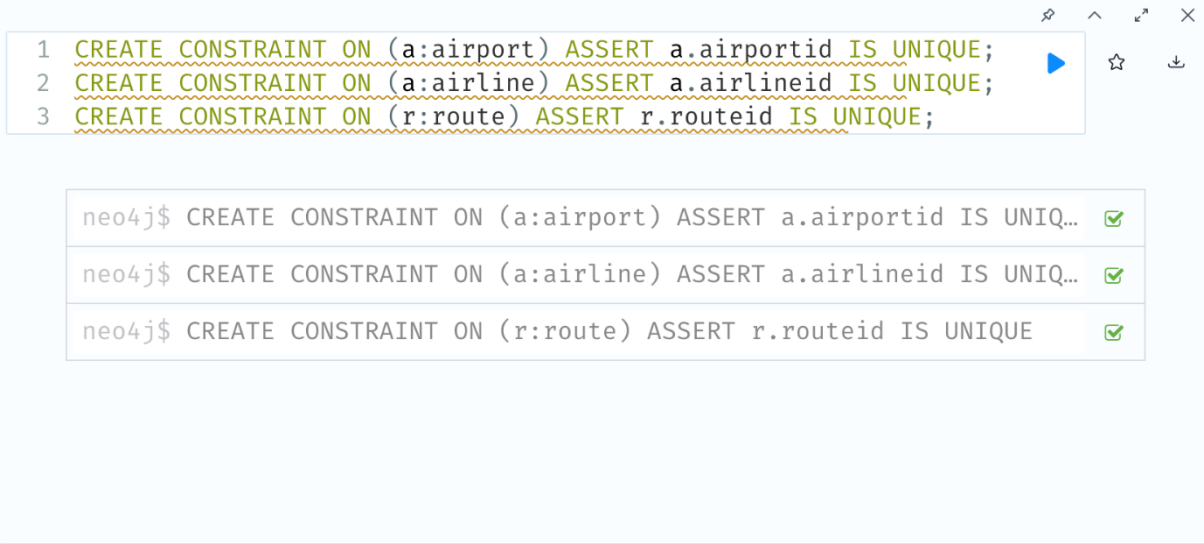


Figure 6: Creating all nodes indexes

And finally, we had to create the nodes with the selected attributes.
For Airports we chose:

Attributes
AirportID
Name
Country
City
Latitude
Longitude
IATA

Figure 7: Creating Airports nodes

For Airlines we chose:

Attributes
AirlineID
Name
Country

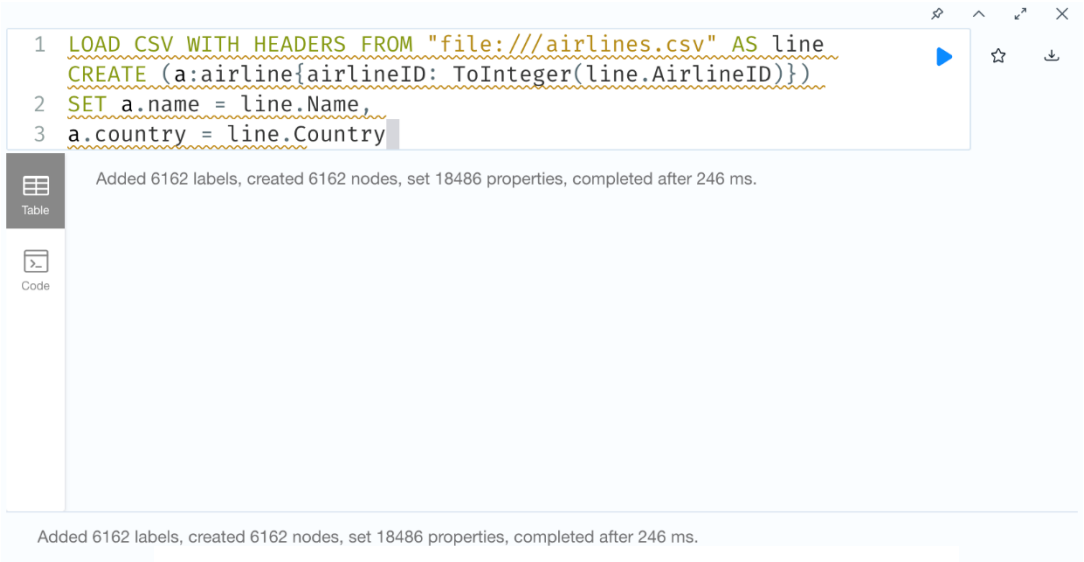


Figure 8: Creating Airlines nodes

For Routes we chose:

Attributes
Routeid
SourceID
DestinationID
Equipment
Destination
Source
AirlineID
Stops

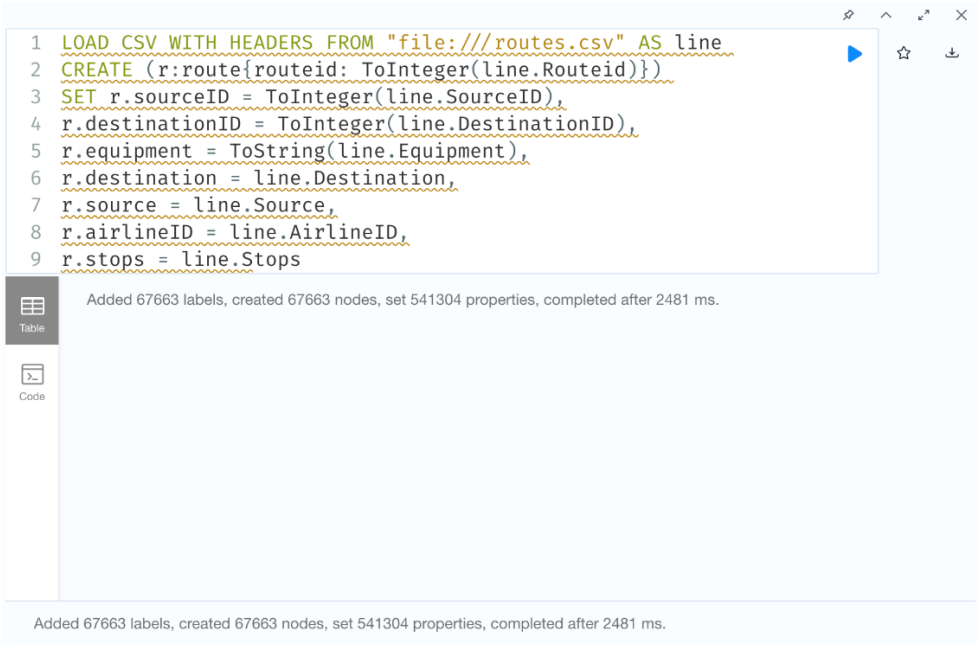


Figure 9: Creating Routes nodes

Figure 10: Creating “source” relationship from airport to route

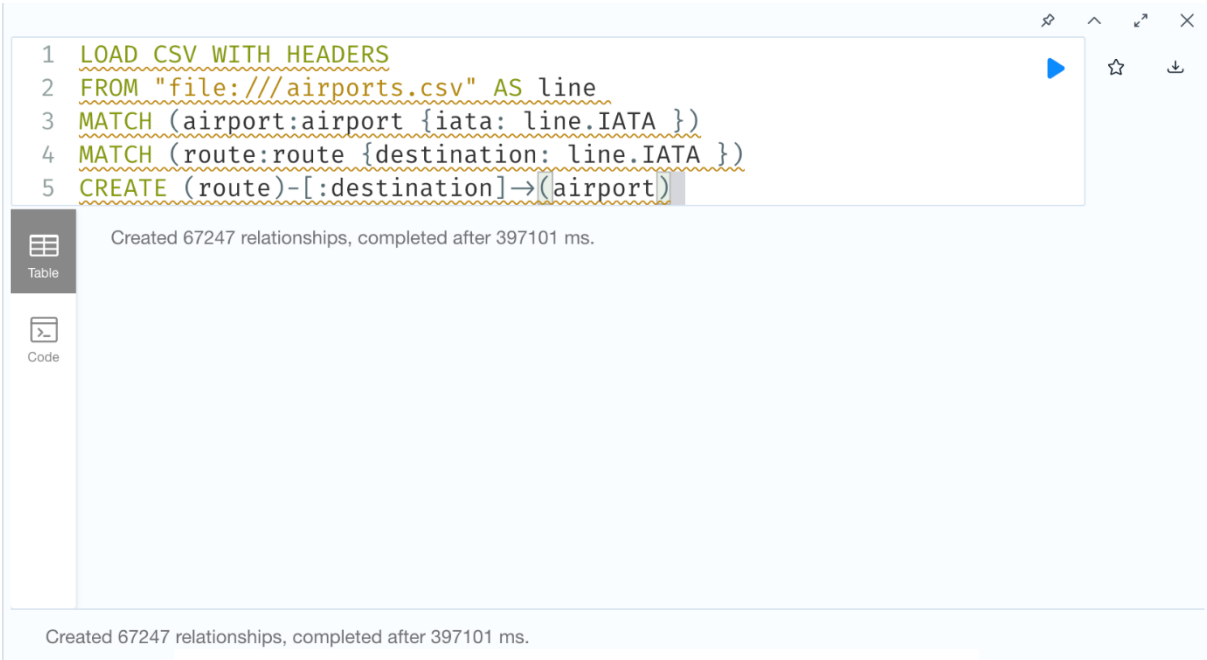


Figure 11: Creating “destination” relationship from route to airport

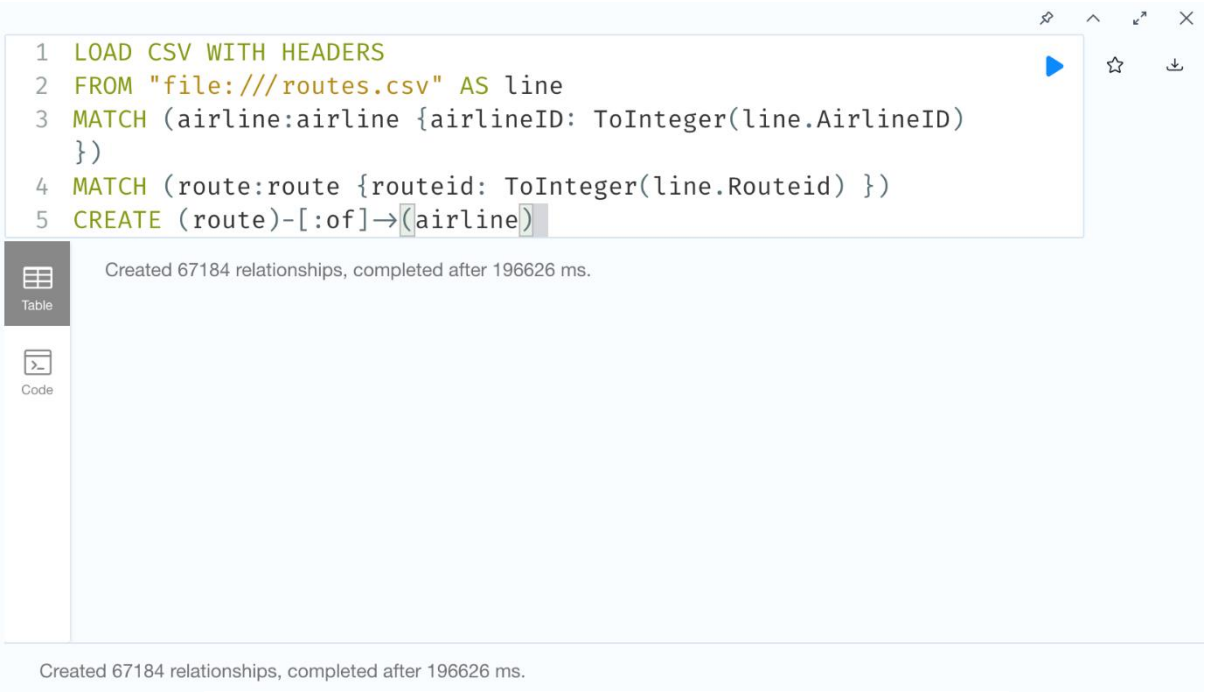


Figure 12: Creating “of” relationship from route to airline

Part II: Querying the database

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

```
MATCH (airport1: airport)-[:source]→(route1: route)
MATCH (route2: route)-[:destination]→(airport1: airport)
WITH count(distinct(route1.routeid)) AS In, count(distinct(route2.routeid)) AS Out, airport1.name
AS Airport_Name
RETURN Airport_Name, (In+Out) AS Number_of_routes
ORDER BY Number_of_routes DESC
LIMIT 5;
```

Table | Text | Code

	Airport_Name	Number_of_routes
1	"Hartsfield Jackson Atlanta International Airport"	1826
2	"Chicago O'Hare International Airport"	1108
3	"Beijing Capital International Airport"	1069
4	"London Heathrow Airport"	1051
5	"Charles de Gaulle International Airport"	1041

Started streaming 5 records after 21 ms and completed after 3052 ms.

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

```
MATCH (airport)
RETURN airport.country AS Country, COUNT(distinct airport.airportID) AS Number_Of_Airports
ORDER BY Number_Of_Airports DESC
LIMIT 5;
```

Table | Text | Code

	Country	Number_Of_Airports
1	"United States"	1512
2	"Canada"	430
3	"Australia"	334
4	"Brazil"	264
5	"Russia"	264

Started streaming 5 records after 12 ms and completed after 154 ms.

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

```
MATCH (airport1: airport)
WHERE NOT airport1.country = 'Greece'
MATCH (airport1:airport)-[:source]→(route1: route)
MATCH (route1: route)-[:destination]→(airport2: airport {country: 'Greece'})
MATCH (route1: route)-[:of]→(airline)
MATCH (airport3: airport)
WHERE NOT airport3.country = 'Greece'
MATCH (a4:airport {country: 'Greece'})-[:source]→(route2: route)
MATCH (route2: route)-[:destination]→(airport3: airport)
MATCH (route2: route)-[:of]→(airline)
WITH count(distinct(route1.routeid)) AS to_Greece, count(distinct(route2.routeid)) AS from_Greece,
airline.name AS airline_Name
```

Table | Text | Code

	airline_Name	number_of_routes
1	"Ryanair"	150
2	"Aegean Airlines"	136
3	"Air Berlin"	120
4	"easyJet"	80
5	"TUIfly"	64

Started streaming 5 records after 63 ms and completed after 2032 ms.

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

```
MATCH (airport1:airport {country:'Germany'})-[:source]→(route1: route)
MATCH (route1: route)-[:destination]→(airport2:airport {country:'Germany'})
MATCH (route1: route)-[:of]→(airline)
RETURN airline.name AS Airline, COUNT(*) AS Flights
ORDER BY Flights DESC
LIMIT 5;
```

Table | Text | Code

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

Started streaming 5 records after 19 ms and completed after 46 ms.

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

```
MATCH (airport1: airport)
WHERE NOT airport1.country = 'Greece'
MATCH (airport1:airport)-[:source]→(route1: route)
MATCH (route1: route)-[:destination]→(a2: airport {country: 'Greece'})
MATCH (route1: route)-[:of]→(airline)
RETURN airport1.country as Country, COUNT(*) AS Flights
ORDER BY Flights DESC
LIMIT 10;
```

Table | Text | Code

	Country	Flights
1	"Germany"	176
2	"United Kingdom"	105
3	"Austria"	36
4	"France"	32
5	"Russia"	28
6	"Italy"	25

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.

```
CALL{MATCH (airport1: airport{country: 'Greece'})-[:source]→(route1: route)
MATCH (route2: route)-[:destination]→(airport1)
WITH count(distinct(route1.routeid)) AS Inb, count(distinct(route2.routeid)) AS Out
RETURN (Inb+Out) AS Total}
MATCH (airport1: airport{country: 'Greece'})-[:source]→(route1: route)
MATCH (route2: route)-[:destination]→(airport1)
WITH count(distinct(route1.routeid)) AS Inb1, count(distinct(route2.routeid)) AS Out1,
Total,airport1.city AS City
RETURN City, ROUND(((Inb1+Out1)*100.00)/Total,2) AS Traffic
ORDER BY Traffic DESC;
```

Table | Text | Code

	City	Traffic
1	"Athens"	25.19
2	"Heraklion"	13.69
3	"Thessaloniki"	10.88
4	"Rhodos"	10.5
5	"Kerkyra/corfu"	6.44
6	"Kos"	5.63

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

```
MATCH (airport1: airport)
WHERE NOT airport1.country = 'Greece'
MATCH (route1 :route)
WHERE route1.equipment = '738' OR route1.equipment = '320'
MATCH (airport1: airport)-[:source]→(route1:route)
MATCH (route1:route)-[:destination]→(airport2: airport {country: 'Greece'})
MATCH (route1:route)-[:of]→(airline1: airline)
RETURN route1.equipment AS Airplane_Type, COUNT(*) AS Flights
ORDER BY Flights DESC;
```

	Airplane_Type	Flights
1	"320"	147
2	"738"	110

Started streaming 2 records after 20 ms and completed after 40 ms.

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.

```
MATCH (airport1: airport)-[:source]→(route1: route)
MATCH (route1)-[:destination]→(airport2: airport)
WITH airport1.name AS From, airport2.name AS To, ROUND(distance(point({ longitude:
airport1.longitude, latitude: airport1.latitude }),point({ longitude: airport2.longitude, latitude:
airport2.latitude })),4) AS Dist
WHERE From < To
RETURN  distinct From, To, Dist
ORDER BY Dist DESC
LIMIT 5;
```

	From	To	Dist
1	"Hartsfield Jackson Atlanta International Airport"	"OR Tambo International Airport"	13597809.9273
2	"Dubai International Airport"	"Los Angeles International Airport"	13415094.5371
3	"King Abdulaziz International Airport"	"Los Angeles International Airport"	13404826.0739
4	"Dubai International Airport"	"George Bush Intercontinental Houston Airport"	13139523.5681
5	"Dubai International Airport"	"San Francisco International Airport"	13034662.8697

Started streaming 5 records after 26 ms and completed after 836 ms.

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.

```
CALL{
MATCH (airport1: airport)-[:source]→(route1: route)
MATCH (route1: route)-[:destination]→(airport2: airport{city:'Berlin'})
WITH airport1.name AS air_name
RETURN COLLECT(air_name) AS non_wanted_airports}
MATCH (airport3: airport)
WHERE NOT airport3.city = 'Berlin'
MATCH (airport4:airport)
WHERE NOT airport4.city = 'Berlin'
MATCH (airport3)-[:source]→(route2:route)
MATCH (route2)-[:destination]→(airport4)
WHERE NOT airport3.name in non_wanted_airports
```

	City	Flights
1	"Atlanta"	915
2	"Shanghai"	606
3	"Los Angeles"	489
4	"Dallas-Fort Worth"	469
5	"Tokyo"	443

Started streaming 5 records after 266 ms and completed after 636 ms.

```
neo4j$
MATCH (airport3: airport)
WHERE NOT airport3.city = 'Berlin'
MATCH (airport4:airport)
WHERE NOT airport4.city = 'Berlin'
MATCH (airport3)-[:source]→(route2:route)
MATCH (route2)-[:destination]→(airport4)
WHERE NOT airport3.name in non_wanted_airports
WITH airport3.city as City, count(distinct(route2.routeid)) as Flights
RETURN distinct City, Flights
ORDER BY Flights DESC
LIMIT 5;
```

	City	Flights
1	"Atlanta"	915
2	"Shanghai"	606
3	"Los Angeles"	489
4	"Dallas-Fort Worth"	469
5	"Tokyo"	443

Started streaming 5 records in less than 1 ms and completed after 193 ms.

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