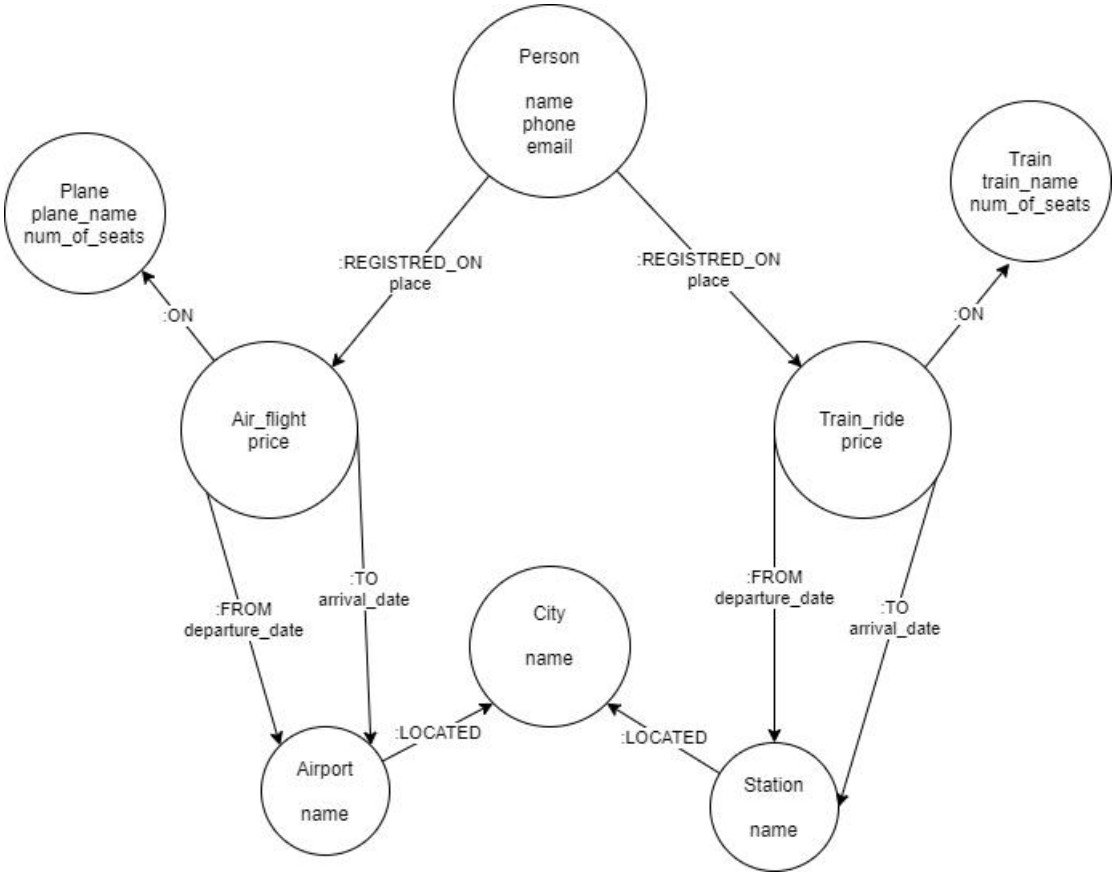
Модель данных



Вершины:

1. Person.

Характеризует пользователя сервиса.

Свойства:

name - имя пользователя;
phone - телефон пользователя;
email - телефон пользователя;

2. Air_flight.

Характеризует авиа рейс.

Свойства:

Price - стоимость рейса;

3. Train_ride.

Характеризует поездку на поезде.

Свойства:

Price - стоимость поездки;

4. Plain.

Характеризует самолет.

Свойства:

Plane_name - название самолета;
Num_of_seats - количество мест;

5. Train.

Характеризует поезд.

Свойства:

train_name - название поезда;
Num_of_seats - количество мест;

6. Airport.

Характеризует аэропорт.

Свойства:

Name - название аэропорта;

7. Station.

Характеризует станцию.

Свойства:

Name - название станции;

8. City.

Характеризует город.

Свойства:

Name - название города;

Отношения:

1. :Located.

Характеризует принадлежность станций и аэропортов к определенному городу.

Нет свойств.

2. :From.

Характеризует принадлежность авиа или жд рейса к аэропорту или станции отправления.

Свойства:

Departure_date - дата и время отправления.

3. :To.

Характеризует принадлежность авиа или жд рейса к аэропорту или станции прибытия.

Свойства:

arrival_date - дата и время прибытия.

4. :On.

Характеризует принадлежность авиа или жд рейса к самолету или поезду.

Нет свойств.

5. :Registred_on.

Характеризует купленные пользователем авиа рейсы или поездки.

Свойства:

Place - место посадки пассажира.

Вычисление примерного объема данных.

Вершины:

1. Person.

Id - int = 4b
Name - string - 2b * 50 = 100b
Phone - string - 2b * 20 = 40b
Email - string - 2b * 50 - 100b

Sum = 244b

2. Plane.

Id - int = 4b
Plane_name - string - 2b*20=40b
Num_of_seats - int =4b

Sum = 48b

3. Train = Plane = 48b

4. Air_flight.
Id - int = 4b
Price - int = 4b

Sum = 8b

5. Train_ride = Air_flight = 8b

6. City.
Id - int = 4b
Name - string - 2b*50 = 100b

Sum = 104b

7. Airport.

Id - int = 4b
Name - string = 2b*50 = 100b

Sum = 104b

8. Station = Airport = 104b

Отношения:

1. :On.
Id - int = 4b

Sum = 4b

2. :Registed_on.
Id - int = 4b
Place - string = 2b*10 = 20b

3. :From.
Id - int = 4b
Departue_date - datetime = 15*4b = 60b

Sum = 64b

4. :To = :From = 64b

5. :Located.
Id - int = 4b

Каждый авиа рейс и поездка на поезде обязательно имеет :From, :To и [:On]->(Plane | Train)

Получается
V_air_flight(N)=V_train_ride(N)=(8b+64b*2+4b+48b)*N=188b * N

Каждый аэропорт и станция обязательно имеет :Located
V_airport(N)=V_station(N)=(104b+4b)*N=108b*N

Пусть имеется A городов, B пользователей, C авирейсов, D поездок на поезде, E аэропортов, F станций и G покупок билетов, получается чистый объем:

(A*104b + B*244b + (C + D)*188b + (E + F)*108b + G * 20b)

Фактический объем:

Запросы.

Добавить авиа рейс из заданных аэропортов.

```
match (st1:Airport{name:'St_1'})
match (st2:Airport{name:'St_4'})
create (r:Air_flight{price:200})
create (r)-[:_FROM{diparture_time:datetime("2019-06-01")}]->(st1)
create (r)-[:_TO{arrival_time:datetime("2019-06-01")}]->(st2)
create (r)-[:_ON]->(:Plane{name:'aerobus_1', num_of_seats:300})
```

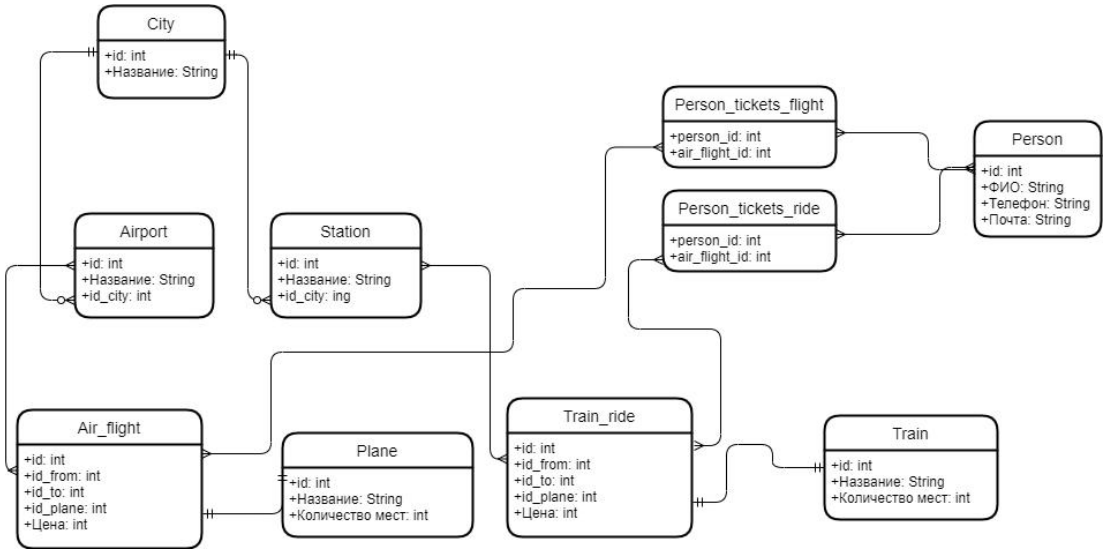
Зарегестрировать пользователя на рейс из города А в Б

```
match (a1:Airport)-[:LOCATED]->(:City{name:'A'})
match (a2:Airport)-[:LOCATED]->(:City{name:'B'})
match (a1)<-[:`_FROM`]-(af:Air_flight)-[:`_TO`]->(a2)
create (p:Person{phone:'43252'})-[:REGISTED_ON]->(af)
```

Вывести количество оставшихся свободных мест на авиа рейсе

```
match (p:Person)-[:REGISTED_ON]->(a:Air_flight) where ID(a)=8
match (a)-[:`_ON`]->(plane:Plane)
return plane.num_of_seats-count(p)
```

Аналогичная реляционная модель



City

Содержит информацию о городах.

Id - уникальный идентификатор города. Тип - int. 4b
Name - название города. Тип - String. 2b*50=100b

V = 104b

Airport

Содержит информацию об аэропортах.

Id - уникальный идентификатор аэропора. Тип - int. 4b
Name - название аэропорта. Тип - String. 2b*50=100b
Id_city - уникальный идентификатор города, к которому принадлежит аэропорт. Тип - int. 4b

V = 108b

Station

Содержит информацию о жд станциях.

Id - уникальный идентификатор станции. Тип - int. 4b
Name - название станции. Тип - String. 2b*50 = 100b
Id_city - уникальный идентификатор города, к которому принадлежит станция. Тип - int. 4b

V = 108b

Plane

Id - уникальный идентификатор аэропора. Тип - int. 4b
Name - название самолета. Тип - String. 2b*20 = 40b
Num_of_seats - количество мест в самолете. Тип - int. 4b

V = 48b

Train

Id - уникальный идентификатор аэропора. Тип - int. 4b
Name - название поезда. 2b*20 = 40b
Num_of_seats - количество мест в поезде Тип - int. 4b

V = 48b

Air_flight

Id - уникальный идентификатор авиа рейса. Тип - int. 4b
Id_to - уникальный идентификатор аэропора прибытия. Тип - int. 4b
Id_from - уникальный идентификатор аэропора отправления. Тип - int. 4b
Id_plane - уникальный идентификатор самолета. Тип - int. 4b
Price - цена за рейс. Тип - int. 4b

V = 20b

Train_ride

Id - уникальный идентификатор поездки на поезде. Тип - int. 4b
Id_to - уникальный идентификатор аэропорта прибытия. Тип - int. 4b
Id_from - уникальный идентификатор аэропорта отправления. Тип - int. 4b
Id_train - уникальный идентификатор поезда. Тип - int. 4b
Price - цена за поездку. Тип - int. 4b

V = 20b

Person

Содержит информацию о пользователях.

Id - уникальный идентификатор аэропора. Тип - int. 4b
Name - имя пользователя. Тип - String. 2b*50 = 100b
Phone - телефон пользователя. Тип - String. 2b*20 = 40b
Email - почта пользователя. Тип - String. 2b*50 = 100b

V = 244b

Person_tickets_flight

Содержит информацию о купленных пользователями билетах на авиа рейсы.

Person_id - уникальный идентификатор пользователя. Тип - int. 4b
Air_flight_id - уникальный идентификатор оплаченного рейса. Тип - int. 4b

V = 8b

Person_tickets_ride
Содержит информацию о купленных пользователями билетах на поезда.

Person_id - уникальный идентификатор пользователя. Тип - int. 4b
Train_ride_id - уникальный идентификатор поездки на поезде. Тип - int. 4b

V = 8b

Пусть имеется A городов, B пользователей, C авирейсов, D поездок на поезде, E аэропортов, F станций, G покупок билетов, H самолетов, I поездов, получается чистый объем:

(A*104b + B*244b + (C + D)*20b + (E + F)*108b + G * 8b + (H+I)*48b)

Запросы.

Получить id рейсов из города А в Б.

```
Select id from Air_flight
Join Airport
On Airport.id = id_from
Join Airport
On Airport.id = id_to
Join City
On airport.id_city = City.id
Where City_from.name = ‘Saint-Petersburg’ and City_to.name = ‘Moscow’;
```

Зарегистрировать пользователя на рейс.

```
Insert into Person_tickets_flight
    Values(flight_id,person_id)
```

Сравнение Neo4j и SQL.

Запросы на Cipher являются более компактными, чем на SQL. Также для реализации отношения, требуется создания дополнительных сущностей.

Графовая модель данных занимает больше места в памяти, чем реляционная, но работает быстрее.

Из результатов, делаем вывод, что neo4j подходит лучше, для решения данной задачи.