Q14

Model the following system as a graph model, and answer the queries using Cypher.

Government provides various scholarships for students. A students apply for scholarship. Student can get benefits of more than one scholarship if satisfies the criteria. Student can recommend it for his friends or other family members

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.

2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.

3.Answer the following Queries in Cypher:

a.List the names of scholarship for OBC category.

b.Count no. of students who are benefited by scholarship in year 2020-2021

c.Update the income limit for scholarship.

d.List the most popular scholarship.

// Create Students, Scholarships, and Categories with Relationships

```
CREATE
  (:Student {name: 'Student1', income: 30000, year: '2020-2021'})-[:APPLIED_FOR]-
>(:Scholarship {name: 'Scholarship1', criteria: 'Criteria1', income_limit: 40000, popularity: 10})-
[:BELONGS_TO]->(:Category {name: 'OBC'}),
  (:Student {name: 'Student2', income: 25000, year: '2020-2021'})-[:APPLIED_FOR]-
>(:Scholarship {name: 'Scholarship2', criteria: 'Criteria2', income_limit: 30000, popularity: 15})-
[:BELONGS_TO]->(:Category {name: 'General'}),
  (:Student {name: 'Student3', income: 35000, year: '2020-2021'})-[:APPLIED_FOR]-
>(:Scholarship {name: 'Scholarship3', criteria: 'Criteria3', income_limit: 35000, popularity: 20})-
[:BELONGS_TO]->(:Category {name: 'SC'});
```

a.List the names of scholarship for OBC category.

```
MATCH (scholarship:Scholarship)-[:BELONGS_TO]->(category:Category {name: 'OBC'})
RETURN scholarship.name;
```

b.Count no. of students who are benefited by scholarship in year 2020-2021

```
MATCH (student:Student)-[:APPLIED_FOR]->(scholarship:Scholarship)
WHERE student.year = '2020-2021'
RETURN COUNT(DISTINCT student) AS numberOfStudents;
```

c.Update the income limit for scholarship.

```
MATCH (scholarship:Scholarship {name: 'Scholarship1'})
SET scholarship.income_limit = 48000;
```

d.List the most popular scholarship.

```
MATCH (scholarship:Scholarship)
RETURN scholarship.name, scholarship.popularity
ORDER BY scholarship.popularity DESC
LIMIT 1;
```

Q15)
Model the following movie system as a Graph database.
Consider a information about of movie and actors. One movie can have more than one actor

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.
2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.
3.Answer the Queries:

a. Find movie which made highest business.
b. Display details of movie along with actors.
c. List all the movie of “Shahrukh Khan”.
d. Display all movie having more than 2 awards received

```
// Create Movies
CREATE (movie1:Movie {Title: 'Movie1', Business: 1000000})
CREATE (movie2:Movie {Title: 'Movie2', Business: 1500000})

// Create Actors
CREATE (actor1:Actor {Name: 'Amitabh Bacchan'})
CREATE (actor2:Actor {Name: 'Nana Patekar'})
CREATE (actor3:Actor {Name: 'Shahrukh Khan'})

// Create Awards
CREATE (award1:Award {Category: 'Best Actor'})
CREATE (award2:Award {Category: 'Best Movie'})
CREATE (award3:Award {Category: 'Best Director'})

// Create Relationships
CREATE (actor1)-[:ACTED_IN {Role: 'Lead'}]->(movie1)
CREATE (actor2)-[:ACTED_IN {Role: 'Supporting'}]->(movie1)
CREATE (actor3)-[:ACTED_IN {Role: 'Lead'}]->(movie2)

CREATE (movie1)-[:WON_AWARD {Category: 'Best Actor'}]->(award1)
CREATE (movie1)-[:WON_AWARD {Category: 'Best Movie'}]->(award2)
CREATE (movie1)-[:WON_AWARD {Category: 'Best Director'}]->(award3)
CREATE (movie2)-[:WON_AWARD {Category: 'Best Director'}]->(award3)
```

a. Find movie which made highest business.

```
MATCH (m:Movie)
RETURN m
ORDER BY m.Business DESC
LIMIT 1;
```

b. Display details of movie along with actors.

```
MATCH (a:Actor)-[:ACTED_IN]->(m:Movie)
RETURN m, a;
```

c. List all the movie of “Shahrukh Khan”.

```
MATCH (a:Actor {Name: "Shahrukh Khan"})-[:ACTED_IN]->(m:Movie)
RETURN m;
```

d. Display all movie having more than 2 awards received

```
MATCH (m:Movie)-[:WON_AWARD]->(a:Award)
WITH m, COUNT(a) AS awardCount
WHERE awardCount > 2
RETURN m;
```

Q16)
Model the following food service industry information as a graph model, and answer the following queries using Cypher.
Consider food service industries like ZOMATO, Swiggy around us. Popular restaurants are connected to these industries to increase sell. A person order food through this industry and get offers. A person give rate(1-5 stars) to company its facility/facilities. and can recommend this to his/her friends.

4.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.
5.Create nodes and relationships, along with their properties, and visualize your actual Graph model.
6.Answer the Queries.

a.Count no. of customers who place order on "1/1/2023"
b.List the names of customers whose name starts with S and place order using Swiggy
c.List the names of hotels with high rating (>=4).
d.List the most recommended hotels in……. area.

```
//Creating Persons
CREATE (amit:Person {name: "Amit", order_date: "01/01/2023"})
CREATE (suraj:Person {name: "Suraj",order_date: "04/01/2023"})
CREATE (sanjay:Person {name: "Sanjay",order_date: "08/01/2023"})
CREATE (gaurav:Person {name: "Gaurav", order_date: "01/01/2023"})

//Creating Hotels
CREATE (saffron:Hotel {name: "Hotel Saffron", rating:3.2})
CREATE (mariote:Hotel {name: "Hotel Mariote", rating:4.6})
CREATE (indigo:Hotel {name: "Hotel Indigo", rating:2.4})

//Creating Food Delievery Agents
CREATE (zomato:FDA {name:"Zomato"})
CREATE (swiggy:FDA {name:"Swiggy"})

//Creating Relationships
CREATE (amit)-[:OrdersFrom]->(zomato)-[:FOR]->(saffron)
CREATE (suraj)-[:OrdersFrom]->(swiggy)-[:FOR]->(indigo)
CREATE (sanjay)-[:OrdersFrom]->(swiggy)-[:FOR]->(mariote)
CREATE (gaurav)-[:OrdersFrom]->(zomato)-[:FOR]->(saffron)
```

a.Count no. of customers who place order on "1/1/2023"

```
Match(n:Person{order_date:1/1/2023}) return count(n)
```

b.List the names of customers whose name starts with S and place order using Swiggy

```
Match(p:Person)-[:OrdersFrom]->(swiggy)WHERE p.name STARTS WITH 'S'
RETURN p.name;
```

c.List the names of hotels with high rating (>=4).

```
Match(h:Hotel)    WHERE h.rating>4     RETURN h.name;
```

d.List the most recommended hotels in……. area.

```
MATCH (hotel:Hotel)          RETURN hotel.name, hotel.rating
ORDER BY hotel.rating DESC        LIMIT 2;
```

Q17)
Model the following Books and Publisher information as a graph model, and answer the following queries using Cypher.
Author wrote various types of books which is published by publishers. A reader reads a books according to his linking and can recommend/provide review for it.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.
2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.
3.Answer the Queries

a. List the names of authors who wrote “Comics”.
b. Count no. of readers of book published by “Sage”.
c. List all the publisher whose name starts with “N” [4]
d. List the names of people who have given a rating of (>=3) for book

```
CREATE (taa:Book {title: "The Amazing Adventures", genre: "Comics"})
CREATE (mu:Book {title: "Mystery Unveiled", genre: "Mystery"})

// Creating Publishers
CREATE (sage:Publisher {name: "Sage"})
CREATE (NH:Publisher {name: "New Horizon"})

// Creating Readers
CREATE (alice:Reader {name: "Alice"})
CREATE (bob:Reader {name: "Bob"})

//Creating Writers
CREATE(js:Person{name:"John Smith"})
CREATE(jd:Person{name:"John Doe"})

//Creating relationships
CREATE(mu)-[:PublishedBy]->(sage)
CREATE(taa)-[:PublishedBy]->(NH)
CREATE(jd)-[:WROTE]->(taa)
CREATE(js)-[:WROTE]->(mu)
CREATE(alice)-[:READS{rating:3.4}]->(mu)
CREATE(bob)-[:READS{rating:4.5}]->(taa)
```

a. List the names of authors who wrote “Comics”.

```
Match(p:Person)-[:WROTE]->(b:Book)
Where b.genre="Comics"
Return p.name
```

b. Count no. of readers of book published by “Sage”.

```
Match(r:Reader)-[:READS]->(b:Book)-[:PublishedBy]->(ps:Publisher)
Where ps.name="Sage"
Return Count(r)
```

c. List all the publisher whose name starts with “N” [4]

```
Match(ps:Publisher)
Where ps.name STARTS WITH 'N'
RETURN ps.name
```

d. List the names of people who have given a rating of (>=3) for book

```
Match(r:Reader)-[rs:READS]->(b:Book)
WHERE rs.rating>3
Return r.name
```

Q18)

Model the following Doctor's information system as a graph model, and answer the following queries using Cypher.

Consider the doctors in and around Pune. Each Doctor is specialized in some stream like Pediatric, Gynaec, Heart Specialist, Cancer Specialist, ENT, etc. A doctor may be a visiting doctor across many hospitals or he may own a clinic. A person can provide a review/can recommend a doctor.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.

2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.

3.Answer the Queries

a. List the Orthopedic doctors in ……. Area.

b .List the doctors who has specialization in

c. List the most recommended Pediatrics in Seren Medows.

d. List all the who visits more than 2 hospitals

```
// Creating Doctors
CREATE (smith:Doctor {name: "Dr. Smith", specialization: "Orthopedic"})
CREATE (john:Doctor {name: "Dr. Johnson", specialization: "Pediatric"})
CREATE (william:Doctor {name: "Dr. Williams", specialization: "ENT"})

// Creating Hospitals
CREATE (cityh:Hospital {name: "City Hospital", area: "Kothrud"})
CREATE (seren:Hospital {name: "Seren Medows", area: "Baner"})
CREATE (ycm:Hospital {name: "YCM Hospital", area: "Pimpri"})

// Creating Clinics
CREATE (hbc:Clinic {name: "Healthy Bones Clinic", area: "Aundh"})

//Creating Patients
CREATE(alice:Person{name:"Alice"})

// Creating Relationships
CREATE (smith)-[:WORKS_AT]->(cityh)
CREATE (smith)-[:OWNS]->(hbc)
CREATE (john)-[:WORKS_AT]->(cityh)
CREATE (john)-[:WORKS_AT]->(seren)
CREATE (john)-[:WORKS_AT]->(ycm)
CREATE (william)-[:WORKS_AT]->(seren)
CREATE (alice)-[:RECOMMENDED]->(john)
CREATE (alice)-[:REVIEWED {rating: 4}]->(john)
```

a. List the Orthopedic doctors in kothrud Area.

```
Match (d:Doctor)-[:WORKS_AT]->(h:Hospital{area:"Kothrud"})
Return d.name
```

b .List the doctors who has specialization in ENT

```
Match (d:Doctor{specialization:"ENT"})
Return d.name
```

b. List the most recommended Pediatrics in Seren Medows.

```
MATCH (doctor:Doctor {specialization: "Pediatric"})-[:WORKS_AT]-
>(hospital:Hospital {name: "Seren Medows"})
MATCH (person:Person)-[r:RECOMMENDED]->(doctor)
WITH doctor, COUNT(r) AS recommendations
ORDER BY recommendations DESC
LIMIT 1
RETURN doctor.name, recommendations;
```

d. List all the doctors who visits more than 2 hospitals

```
MATCH (doctor:Doctor)-[:WORKS_AT]->(hospital:Hospital)
WITH doctor, COUNT(DISTINCT hospital) AS hospitalCount
WHERE hospitalCount > 2
RETURN doctor.name, hospitalCount;
```

Q19)

Model the following Laptop manufacturing information system as a graph model, and answer the following queries using Cypher.

Consider an Laptop manufacturing industries which produces different types of laptops. A customer can buy a laptop, recommend or rate a the product.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.

2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.

3.Answer the Queries

a. List the characteristics of laptop.

b. List the name of customers who bought a “DELL” company laptop

c. List the customers who purchase a device on “26/01/2023”

d. List the most recommended device.

```
// Creating Laptops
CREATE (dell:Laptop {brand: "DELL", model: "Inspiron", characteristics: "15-
inch, 8GB RAM, 512GB SSD"})
CREATE (hp:Laptop {brand: "HP", model: "Pavilion", characteristics: "14-inch, 16GB RAM, 1TB HDD"})
CREATE (lenovo:Laptop {brand: "Lenovo", model: "ThinkPad", characteristics: "13-
inch, 12GB RAM, 256GB SSD"})

// Creating Customers
CREATE (alice:Customer {name: "Alice"})
CREATE (bob:Customer {name: "Bob"})
CREATE (charlie:Customer {name: "Charlie"})

// Creating Relationships
CREATE (alice)-[:BOUGHT{purchase_date:"26-1-23"}]->(dell)
CREATE (bob)-[:BOUGHT{purchase_date:"23-7-23"}]->(lenovo)
CREATE (charlie)-[:BOUGHT{purchase_date:"21-1-23"}]->(hp)
CREATE (alice)-[:REVIEWED {rating: 2}]->(dell)
CREATE (bob)-[:REVIEWED {rating: 3}]->(hp)
CREATE (charlie)-[:REVIEWED {rating: 4}]->(lenovo)
CREATE (charlie)-[:RECOMMENDED]->(lenovo)
CREATE (alice)-[:RECOMMENDED]->(lenovo)
CREATE (bob)-[:RECOMMENDED]->(hp)
```

a. List the characteristics of hp laptop.

```
MATCH(hp:Laptop{brand:"HP"})
Return hp.brand, hp.model, hp.characteristics
```

b. List the name of customers who bought a “DELL” company laptop

```
Match(c:Customer)-[:BOUGHT]->(dell:Laptop{brand:"DELL"})
Return c.name
```

c. List the customers who purchase a device on “26/01/2023”

```
Match(c:Customer)-[:BOUGHT]->(dell:Laptop{brand:"DELL"})
Return c.name
```

d. List the most recommended device.

```
Match(c:Customer)-[r:RECOMMENDED]->(l:Laptop)
With l, Count(r) as recommendations
Order By recommendations Desc
Limit 1
return l.brand, recommendations
```

Q20)

Model the following nursery management information as a graph model, and answer the following queries using Cypher.

Nursery content various types of plants, fertilizers and required products. Customer visit the nursery or use an app , purchase the plants and necessary products also rate and recommend the app

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.

2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.

3.Answer the following queries using Cypher:

a.List the types of plants from your graph model

b.List the popular flowering plants.

c.List the names of plants sold plant where qty>500 in last 2 days

d.List the names of suppliers in decreasing order who supplies “Creepers”.

```
// Creating Plants
CREATE (rose:Plant {name: "Rose", type: "Flowering"})
CREATE (tulip:Plant {name: "Tulip", type: "Flowering"})
CREATE (bamboo:Plant {name: "Bamboo", type: "Non-Flowering"})
CREATE (creeper:Plant {name: "Creeper", type: "Creepers"})

// Creating Fertilizers
CREATE (nb:Fertilizer {name: "Nitrogen Boost", type: "Liquid"})
CREATE (om:Fertilizer {name: "Organic Mix", type: "Granular"})

// Creating Products
CREATE (gs:Product {name: "Garden Shears", type: "Tool"})
CREATE (pf:Product {name: "Plant Food", type: "Liquid"})

// Creating Customers
CREATE (alice:Customer {name: "Alice"})
CREATE (bob:Customer {name: "Bob"})

// Creating Suppliers
CREATE (gru:Supplier {name: "Gardens R Us"})
CREATE (gtc:Supplier {name: "Green Thumb Co."})

// Creating App
CREATE (app:App)

// Creating Relationships
CREATE (grs)-[:SUPPLIES {quantity: 600}]->(rose)
CREATE (grs)-[:SUPPLIES {quantity: 300}]->(creeper)
CREATE (gtc)-[:SUPPLIES {quantity: 700}]->(tulip)
CREATE (gtc)-[:SUPPLIES {quantity: 800}]->(creeper)
CREATE (alice)-[:PURCHASE{pdate:"1-7-23",quantity:600}]->(rose)
CREATE (bob)-[:PURCHASE{pdate:"2-7-23",quantity:500}]->(tulip)
CREATE (alice)-[:REVIEWED {rating: 5}]->(rose)
CREATE (bob)-[:RECOMMENDED]->(app)
```

a.List the types of plants from your graph model

```
MATCH (plant:Plant)
RETURN DISTINCT plant.type;
```

b.List the popular flowering plants.

```
MATCH (plant:Plant)
RETURN DISTINCT plant.type;
```

c.List the names of plants sold where qty>500 in last 2 days

```
Match(:Customer)-[pr:PURCHASE]->(p:Plant)
Where pr.pdate="1-7-23" OR pr.pdate="2-7-23" AND pr.quantity>500
Return p.name
```

d.List the names of suppliers in decreasing order who supplies “Creepers”.

```
MATCH (s:Supplier)-[ss:SUPPLIES]->(p:Plant{name:"Creeper"})
WITH s, ss
ORDER BY ss.quantity DESC
RETURN s.name, ss.quantity;
```

Q21)
Model the following Medical information as a graph model, and answer the following queries using Cypher.
There are various brands of medicine like Dr. Reddy, Cipla, SunPharma etc. Their uses vary across different states in India. The uses of medicine is measured as %, with a high use defined as >=90%, Medium Use between 50 to 90%, and Low Use<50%. Each medicine manufactures various types of medicine products like Tablet, Syrup, and Powder etc.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.
2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.
3.Answer the following queries using Cypher:

a. List the names of different medicines considered in your graph
b .List the medicine that are highly Used in Rajasthan.
c. List the highly used tablet in Gujarat.
d. List the medicine names manufacturing “Powder”

```
// Creating nodes
CREATE (medicine1:Medicine {name: 'Medicine1'})
CREATE (medicine2:Medicine {name: 'Medicine2'})
CREATE (brand1:Brand {name: 'Dr. Reddy'})
CREATE (brand2:Brand {name: 'Cipla'})
CREATE (state1:State {name: 'Rajasthan'})
CREATE (state2:State {name: 'Gujarat'})
CREATE (product1:Product {name: 'Tablet'})
CREATE (product2:Product {name: 'Syrup'})
CREATE (product3:Product {name: 'Powder'})

// Creating relationships and setting properties
CREATE (brand1)-[:MANUFACTURED_BY]->(medicine1)
CREATE (brand2)-[:MANUFACTURED_BY]->(medicine2)
CREATE (medicine1)-[:HAS_USE {percentage: 80}]->(state1)
CREATE (medicine2)-[:HAS_USE {percentage: 95}]->(state1)
CREATE (medicine1)-[:HAS_USE {percentage: 93}]->(state2)
CREATE (product1)-[:BELONGS_TO]->(medicine1)
CREATE (product2)-[:BELONGS_TO]->(medicine1)
CREATE (product3)-[:BELONGS_TO]->(medicine2)
```

a. List the names of different medicines considered in your graph

```
MATCH (m:Medicine) RETURN m.name;
```

b .List the medicine that are highly Used in Rajasthan.

```
MATCH (:State {name: 'Rajasthan'})<-[h:HAS_USE]-
(m:Medicine) Where h.percentage>=90 RETURN m.name;
```

c List the highly used tablet in Gujarat.

```
MATCH (:State {name: 'Gujarat'})<-[h:HAS_USE]-(m:Medicine)<-[:BELONGS_TO]-
(p:Product {name: 'Tablet'}) where h.percentage>=90 RETURN m.name;
```

d List the medicine names manufacturing “Powder”

```
MATCH (p:Product {name:'Powder'})-[:BELONGS_TO]->(m:Medicine)
RETURN m.name;
```

Q22)
Model the following Car Showroom information as a graph model,and answer the queries using Cypher. Consider a car showroom with different models of cars like sofas Honda city, Skoda, Creta, Swift, Ertiga etc. Showroom is divided into different sections, onesection for each car model; each section is handled by a sales staff. A sales staff can handle one or more sections.
Customer may enquire about car. An enquiry may result in a purchase by the customer.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.
2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.
3.Answer the following queries:

a. List the types of cars available in the showroom.
b. List the sections handled by Mr. Narayan.
c. List the names of customers who have done only enquiry but not made any purchase.
d. List the highly sale car model.

```
// Nodes and Labels
CREATE (showroom:Showroom {name: 'Car Showroom'})
CREATE (hondaCity:Car {name: 'Honda City'})
CREATE (skoda:Car {name: 'Skoda'})
CREATE (creta:Car {name: 'Creta'})
CREATE (swift:Car {name: 'Swift'})
CREATE (ertiga:Car {name: 'Ertiga'})

CREATE (hondaCitySection:Section {name: 'Honda City Section', type:"Sedan"})
CREATE (skodaSection:Section {name: 'Skoda Section',type:"Sedan"})
CREATE (cretaSection:Section {name: 'Creta Section',type:"SUV"})
CREATE (swiftSection:Section {name: 'Swift Section',type:"Hatchback"})
CREATE (ertigaSection:Section {name: 'Ertiga Section',type:"MPV"})

CREATE (narayan:Staff {name: 'Mr. Narayan'})
CREATE (ashish:Staff {name: 'Mr. Ashish'})
CREATE (navin:Staff {name: 'Mr. Navin'})
CREATE (john:Customer {name: 'John Doe'})
CREATE (jane:Customer {name: 'Jane Smith'})

CREATE (showroom)-[:HAS_SECTION]->(hondaCitySection)
CREATE (showroom)-[:HAS_SECTION]->(skodaSection)
CREATE (showroom)-[:HAS_SECTION]->(cretaSection)
CREATE (showroom)-[:HAS_SECTION]->(swiftSection)
CREATE (showroom)-[:HAS_SECTION]->(ertigaSection)

CREATE (hondaCitySection)-[:HAS_CAR_MODEL]->(hondaCity)
CREATE (skodaSection)-[:HAS_CAR_MODEL]->(skoda)
CREATE (cretaSection)-[:HAS_CAR_MODEL]->(creta)
CREATE (swiftSection)-[:HAS_CAR_MODEL]->(swift)
CREATE (ertigaSection)-[:HAS_CAR_MODEL]->(ertiga)

CREATE (narayan)-[:HANDLES_SECTION]->(hondaCitySection)
CREATE (navin)-[:HANDLES_SECTION]->(skodaSection)
CREATE (narayan)-[:HANDLES_SECTION]->(cretaSection)
CREATE (ashish)-[:HANDLES_SECTION]->(swiftSection)
CREATE (narayan)-[:HANDLES_SECTION]->(ertigaSection)

CREATE (john)-[:ENQUIRY]->(hondaCity)
CREATE (jane)-[:ENQUIRY]->(creta)
CREATE (jane)-[:PURCHASE]->(creta)
```

a. **List the types of cars available in the showroom.**

```
Match(c:Car) return c.name
```

b. **List the sections handled by Mr. Narayan.**

```
MATCH (:Staff{name:"Mr. Narayan"})-[:HANDLES_SECTION]->(s:Section) Return s.name
```

c. **List the names of customers who have done only enquiry but not made any purchase.**

```
MATCH (customer:Customer)-[:ENQUIRY]->(car)
WHERE NOT (:Customer)-[:PURCHASE]->(car)
RETURN DISTINCT customer.name;
```

d. **List the highly sale car model.**

```
MATCH (customer:Customer)-[p:PURCHASE]->(c:Car)
with c,Count(p) as purchases
Order By purchases Desc
Limit 1
RETURN c.name;
```

Q23)

Model the following Automobile information system as a graph model,and answer the following queries using Cypher.

Consider an Automobile industry manufacturing different types of vehicles like Two- Wheeler, Four-Wheeler, etc. A customer can buy one or more types of vehicle. A person can recommend or rate a vehicle type.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.

2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.

3.Answer the following Queries:

a. List the characteristics of four wheeler types.

b. List the name of customers who bought a two wheeler vehicle.

c. List the customers who bought more than one type of vehicle.

d. List the most recommended vehicle type.

```
CREATE (industry:Industry {name: 'Automobile Industry'})
CREATE (twoWheeler:VehicleType {name: 'Two-Wheeler', etype:"Single Cylinder"})
CREATE (threeWheeler:VehicleType {name: 'Three-Wheeler',etype:"Twin Cylinder"})
CREATE (fourWheeler:VehicleType {name: 'Four-Wheeler',etype:"Multi Cylinder"})
CREATE (customerJohn:Customer {name: 'John Doe'})
CREATE (customerJane:Customer {name: 'Jane Smith'})
CREATE (customerJosh:Customer {name: 'Josh Carter'})
// Relationships and Properties
CREATE (customerJohn)-[:BOUGHT]->(twoWheeler)
CREATE (customerJohn)-[:Recommended]->(twoWheeler)
CREATE (customerJohn)-[:Recommended]->(threeWheeler)
CREATE (customerJohn)-[:BOUGHT]->(threeWheeler)
CREATE (customerJosh)-[:Recommended]->(threeWheeler)
CREATE (customerJosh)-[:BOUGHT]->(threeWheeler)
CREATE (customerJane)-[:BOUGHT]->(fourWheeler)
CREATE (customerJane)-[:Recommended]->(fourWheeler)
CREATE (customerJane)-[:BOUGHT]->(twoWheeler)


CREATE (industry)-[:HAS]->(twoWheeler)
CREATE (industry)-[:HAS]->(threeWheeler)
CREATE (industry)-[:HAS]->(fourWheeler)
```

a. List the characteristics of four wheeler types.

```
Match(v:VehicleType{name:"Four-Wheeler"})
Return v.name, v.etype
```

b. List the name of customers who bought a two wheeler vehicle.

```
Match(v:VehicleType{name:"Two-Wheeler"})<-[:BOUGHT]-(c:Customer)
Return c.name
```

b. List the customers who bought more than one type of vehicle.

```
Match(v:VehicleType)<-[b:BOUGHT]-(c:Customer)
With c,count(b) as buys
Where buys>1
Return c.name, buys
```

d. List the most recommended vehicle type.

```
Match(c:Customer)-[r:Recommended]->(v:VehicleType)
with v, count(r) as recommends
Order BY recommends DESC
Limit 1
Return v.name
```

Q24)

Model the following Library information system as a graph model,and answer the following queries using Cypher.

Consider a library information system having different types of books like text, reference, bibliography etc. A student can buy one or more types of book. A student can recommend or rate a book according to its type.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.

2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.

3.Answer the following Queries :

a. List the books of type “text”

b. List the name of student who bought a text and reference types books.

c. List the most recommended book type.

d. List the student who buy the more than one type of book.

```
// Nodes and Labels
CREATE (library:Library {name: 'Library Information System'})
CREATE (textBook:BookType {name: 'Text'})
CREATE (referenceBook:BookType {name: 'Reference'})
CREATE (bibliographyBook:BookType {name: 'Bibliography'})
CREATE (studentJohn:Student {name: 'John Doe'})
CREATE (studentJane:Student {name: 'Jane Smith'})
CREATE (studentJosh:Student {name: 'Josh Smith'})
// Relationships and Properties
CREATE (library)-[:HAS]->(textBook)
CREATE (library)-[:HAS]->(referenceBook)
CREATE (library)-[:HAS]->(bibliographyBook)
CREATE (studentJohn)-[:BOUGHT]->(textBook)
CREATE (studentJohn)-[:BOUGHT]->(referenceBook)
CREATE (studentJosh)-[:BOUGHT]->(referenceBook)
CREATE (studentJane)-[:BOUGHT]->(textBook)
CREATE (studentJosh)-[:BOUGHT]->(bibliographyBook)
CREATE (studentJane)-[:RECOMMENDS]->(textBook)
CREATE (studentJohn)-[:RECOMMENDS]->(textBook)
CREATE (studentJohn)-[:RECOMMENDS]->(referenceBook)
CREATE (studentJane)-[:RATES {rating: 5}]->(textBook)
CREATE (studentJohn)-[:RATES {rating: 4}]->(referenceBook)
```

a. List the books of type “text”

```
Match (s:Student)-[:BOUGHT]->(bt:BookType{name:"Text"})
Return s.name
```

b. List the name of student who bought a text and reference types books.

```
Match (br:BookType{name:"Reference"})<-[:BOUGHT]-(s:Student)-[:BOUGHT]-
>(bt:BookType{name:"Text"})
Return s.name
```

c. List the most recommended book type.

```
MATCH (st:Student)-[r:RECOMMENDS]->(b:BookType)
WITH st, COUNT(r) AS recommend
ORDER BY recommend DESC
LIMIT 1
RETURN st.name
```

d. List the student who buy the more than one type of book.

```
MATCH (st:Student)-[r:BOUGHT]->(b:BookType)
WITH st, COUNT(r) AS buys
where buys>1
RETURN st.name
```

Q25)
Model the following University information system as a graph model, and answer the following queries using Cypher.
University has various departments like Physics, Geography, Computer etc. Each department conducts various courses and a course may be conducted by multiple departments. Every course may have recommendations provided by people.

1.Identify the labels and relationships, along with their properties, and draw a high-level Graph model.
2.Create nodes and relationships, along with their properties, and visualize your actual Graph model.
3.Answer the following Queries :

a. List the details of all the departments in the university.
b. List the names of the courses provided by Physics department.
c.List the most recommended course in Geography department.
d.List the names of common courses across Mathematics and computer department.

```
//Creating Departments
CREATE(phy:Department{name:"Physics"})
CREATE(geo:Department{name:"Geography"})
CREATE(comp:Department{name:"Computer"})
CREATE(math:Department{name:"Mathematics"})

// Creating Courses
CREATE (stat:Courses{name:"Statistics"})
CREATE (arc:Courses{name:"Archiology"})
CREATE (opt:Courses{name:"Optics"})
CREATE (thermo:Courses{name:"Thermodynamics"})
CREATE (ds:Courses{name:"Data Science"})

// Creating Persons
CREATE (shub:Person{name:"Shubham"})
CREATE (suraj:Person{name:"Suraj"})
CREATE (yash:Person{name:"Yash"})

// Create Relationships
CREATE (math)-[:Provides]->(stat)
CREATE (comp)-[:Provides]->(stat)
CREATE (phy)-[:Provides]->(opt)
CREATE (phy)-[:Provides]->(thermo)
CREATE (comp)-[:Provides]->(ds)
CREATE (geo)-[:Provides]->(arc)
CREATE (geo)-[:Provides]->(opt)
CREATE (shub)-[:Recommends]->(stat)
CREATE (suraj)-[:Recommends]->(opt)
CREATE (yash)-[:Recommends]->(ds)
```

a. List the details of all the departments in the university.

```
MATCH(d:Department)        Return d.name
```

b. List the names of the courses provided by Physics department.

```
MATCH(d:Department{name:"Physics"})-[:Provides]->(c:Courses)
Return c.name
```

c.List the most recommended course in Geography department.

```
MATCH(p:Person)-[r:Recommends]->(c:Courses)<-[:Provides]-(geo:Department{name:"Geography"})
with c, count(r) as recommendations
Order BY recommendations Desc
Limit 1
Return c.name, recommendations
```

d.List the names of common courses across Mathematics and computer department.

```
MATCH(:Department{name:"Mathematics"})-[:Provides]->(c:Courses)<-[:Provides]-
(:Department{name:"Computer"})
Return c.name
```