# SLIP 12

Q2.

CREATE (Physics:Department {name: "Physics"}), (Geography:Department {name: "Geography"}), (Computer:Department {name: "Computer"}), (Mathematics:Department {name: "Mathematics"}), (course1:Course {name: "Physics 101"}), (course2:Course {name: "Physics 201"}), (course3:Course {name: "Geography 101"}), (course4:Course {name: "Computer Science 101"}), (course5:Course {name: "Mathematics 101"}), (person1:Person {name: "John"}), (person2:Person {name: "Alice"}), (Physics)-[:DEPARTMENT\_CONDUCTS\_COURSE]->(course1), (Physics)-[:DEPARTMENT\_CONDUCTS\_COURSE]->(course2), (Geography)-[:DEPARTMENT\_CONDUCTS\_COURSE]->(course3), (Computer)-[:DEPARTMENT\_CONDUCTS\_COURSE]->(course4), (Mathematics)-[:DEPARTMENT\_CONDUCTS\_COURSE]->(course5), (course1)-[:COURSE\_HAS\_RECOMMENDATION]->(person1), (course1)-[:COURSE\_HAS\_RECOMMENDATION]->(person2), (course2)-[:COURSE\_HAS\_RECOMMENDATION]->(person1), (course3)-[:COURSE\_HAS\_RECOMMENDATION]->(person2), (course4)-[:COURSE\_HAS\_RECOMMENDATION]->(person1), (course5)-[:COURSE\_HAS\_RECOMMENDATION]->(person2);

Q1)MATCH (d:Department) RETURN d;

Q2)MATCH (d:Department {name: "Physics"})-[:DEPARTMENT\_CONDUCTS\_COURSE]->(c:Course) RETURN c.name;

3)MATCH (d:Department {name: "Geography"})-[:DEPARTMENT\_CONDUCTS\_COURSE]->(c:Course)-[:COURSE\_HAS\_RECOMM ENDATION]->(p:Person) RETURN c.name, COUNT(p) AS recommendations ORDER BY recommendations DESC LIMIT 1;

4)MATCH (math:Department {name: "Mathematics"})-[:DEPARTMENT\_CONDUCTS\_COURSE]->(math\_course:Course), (comp:Department {name: "Computer"})-[:DEPARTMENT\_CONDUCTS\_COURSE]->(comp\_course:Course) WHERE math\_course.name = comp\_course.name RETURN math\_course.name;

# SLIP 13

Q2.

CREATE (student1:Student {name: "Alice"}), (student2:Student {name: "Bob"}), (book1:Book {name: "Introduction to Algorithms"}), (book2:Book {name: "Database Systems"}), (book3:Book {name: "The Art of Computer Programming"}), (type1:Type {name: "Text"}), (type2:Type {name: "Reference"}), (type3:Type {name: "Bibliography"}), (student1)-[:STUDENT\_BOUGHT\_BOOK]->(book1)-[:BOOK\_OF\_TYPE]->(type1), (student1)-[:STUDENT\_BOUGHT\_BOOK]->(book2)-[:BOOK\_OF\_TYPE]->(type2), (student2)-[:STUDENT\_BOUGHT\_BOOK]->(book3)-[:BOOK\_OF\_TYPE]->(type1), (student1)-[:STUDENT\_RECOMMENDED\_BOOK]->(book1), (student1)-[:STUDENT\_RECOMMENDED\_BOOK]->(book2), (student2)-[:STUDENT\_RECOMMENDED\_BOOK]->(book3);

A)MATCH (b:Book)-[:BOOK\_OF\_TYPE]->(t:Type {name: "Text"}) RETURN b.name;

b) MATCH (s:Student)-[:STUDENT\_BOUGHT\_BOOK]->(:Book)-[:BOOK\_OF\_TYPE]->(t:Type) WHERE t.name IN ["Text", "Reference"] RETURN DISTINCT s.name;

c) MATCH (:Student)-[:STUDENT\_RECOMMENDED\_BOOK]->(b:Book)-[:BOOK\_OF\_TYPE]->(t:Type) RETURN t.name, COUNT(b) AS recommendations ORDER BY recommendations DESC LIMIT 1;

d) MATCH (s:Student)-[:STUDENT\_BOUGHT\_BOOK]->(b:Book)-[:BOOK\_OF\_TYPE]->(t:Type) WITH s, COUNT(DISTINCT t) AS numTypes WHERE numTypes > 1 RETURN s.name;

# SLIP 14

Q2.

// Create vehicle types

CREATE (two\_wheeler:VehicleType {name: 'Two-Wheeler', characteristics: 'Small, Agile'}),

(four\_wheeler:VehicleType {name: 'Four-Wheeler', characteristics: 'Spacious, Stable'}),

(truck:VehicleType {name: 'Truck', characteristics: 'Heavy-duty, Large'});

// Create customers

CREATE (customer1:Customer {name: 'John'}),

(customer2:Customer {name: 'Alice'}),

(customer3:Customer {name: 'Bob'});

// Create relationships for customers buying vehicles

MATCH (customer:Customer)

CREATE (customer)-[:CUSTOMER\_BOUGHT\_VEHICLE\_TYPE]->(two\_wheeler);

MATCH (customer:Customer {name: 'Alice'})

CREATE (customer)-[:CUSTOMER\_BOUGHT\_VEHICLE\_TYPE]->(four\_wheeler),

(customer)-[:CUSTOMER\_BOUGHT\_VEHICLE\_TYPE]->(truck);

// Create recommendations and ratings

CREATE (person1:Person {name: 'David'}),

(person2:Person {name: 'Eve'}),

(person3:Person {name: 'Frank'});

MATCH (two\_wheeler:VehicleType {name: 'Two-Wheeler'}),

(four\_wheeler:VehicleType {name: 'Four-Wheeler'})

CREATE (two\_wheeler)-[:PERSON\_RECOMMENDS\_VEHICLE\_TYPE]->(person1),

(four\_wheeler)-[:PERSON\_RECOMMENDS\_VEHICLE\_TYPE]->(person2),

(four\_wheeler)-[:PERSON\_RECOMMENDS\_VEHICLE\_TYPE]->(person3);

1)List the characteristics of four wheeler types.

MATCH (v:VehicleType {name: 'Four-Wheeler'})

RETURN v.characteristics;

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

MATCH (c:Customer)-[:CUSTOMER\_BOUGHT\_VEHICLE\_TYPE]->(:VehicleType {name: 'Two-Wheeler'})

RETURN c.name;

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

MATCH (c:Customer)-[:CUSTOMER\_BOUGHT\_VEHICLE\_TYPE]->(v:VehicleType)

WITH c, COUNT(v) AS numVehicles

WHERE numVehicles > 1

RETURN c.name;

4)List the most recommended vehicle type.

MATCH (v:VehicleType)-[:PERSON\_RECOMMENDS\_VEHICLE\_TYPE]->(p:Person)

RETURN v.name, COUNT(p) AS recommendations

ORDER BY recommendations DESC

LIMIT 1;

# SLIP15

Q2)

// Create car models

CREATE (honda\_city:CarModel {name: 'Honda City'}),

(skoda:CarModel {name: 'Skoda'}),

(creta:CarModel {name: 'Creta'}),

(swift:CarModel {name: 'Swift'}),

(ertiga:CarModel {name: 'Ertiga'});

// Create sections

CREATE (section\_honda\_city:Section {name: 'Honda City Section'}),

(section\_skoda:Section {name: 'Skoda Section'}),

(section\_creta:Section {name: 'Creta Section'}),

(section\_swift:Section {name: 'Swift Section'}),

(section\_ertiga:Section {name: 'Ertiga Section'});

// Create sales staff

CREATE (narayan:SalesStaff {name: 'Mr. Narayan'}),

(john:SalesStaff {name: 'Mr. John'}),

(alice:SalesStaff {name: 'Ms. Alice'});

// Create relationships for sections handled by sales staff

MATCH (section:Section)

CREATE (narayan)-[:SECTION\_HANDLED\_BY]->(section);

// Create customers

CREATE (customer1:Customer {name: 'Bob'}),

(customer2:Customer {name: 'Alice'}),

(customer3:Customer {name: 'Frank'});

// Create enquiries and purchases

CREATE (enquiry1:Enquiry),

(enquiry2:Enquiry),

(purchase1:Purchase),

(purchase2:Purchase),

(purchase3:Purchase);

// Create relationships for customer enquiries and purchases

MATCH (customer:Customer)

CREATE (customer)-[:CUSTOMER\_ENQUIRED\_ABOUT]->(enquiry1),

(customer)-[:CUSTOMER\_ENQUIRED\_ABOUT]->(enquiry2),

(customer)-[:CUSTOMER\_PURCHASED]->(purchase1),

(customer)-[:CUSTOMER\_PURCHASED]->(purchase2),

(customer)-[:CUSTOMER\_PURCHASED]->(purchase3);

1)List the types of cars available in the showroom.

MATCH (car:CarModel)

RETURN car.name;

2)List the sections handled by Mr. Narayan.

MATCH (sales\_staff:SalesStaff {name: 'Mr. Narayan'})-[:SECTION\_HANDLED\_BY]->(section:Section)

RETURN section.name;

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

MATCH (customer:Customer)-[:CUSTOMER\_ENQUIRED\_ABOUT]->(enquiry:Enquiry)

WHERE NOT ((customer)-[:CUSTOMER\_PURCHASED]->())

RETURN customer.name;

4)List the highly sale car model

MATCH (car:CarModel)-[purchase: CUSTOMER\_PURCHASED]->()

RETURN car.name, COUNT(purchase) AS sales

ORDER BY sales DESC

LIMIT 1;

# SLIP 16

Q2)

// Create medicine brands

CREATE (dr\_reddy:MedicineBrand {name: 'Dr. Reddy'}),

(cipla:MedicineBrand {name: 'Cipla'}),

(sun\_pharma:MedicineBrand {name: 'Sun Pharma'});

// Create states

CREATE (rajasthan:State {name: 'Rajasthan'}),

(gujarat:State {name: 'Gujarat'}),

(maharashtra:State {name: 'Maharashtra'});

// Create relationships for medicine brands and states

MATCH (brand:MedicineBrand), (state:State)

CREATE (brand)-[:MEDICINE\_TYPE\_USED\_IN\_STATE {usage\_percentage: 95}]->(state);

// Create medicine types

CREATE (tablet:MedicineType {name: 'Tablet', usage\_percentage: 85}),

(syrup:MedicineType {name: 'Syrup', usage\_percentage: 60}),

(powder:MedicineType {name: 'Powder', usage\_percentage: 40});

// Create relationships for medicine brands and types

MATCH (brand:MedicineBrand)

CREATE (brand)-[:BRAND\_MANUFACTURES\_MEDICINE\_TYPE]->(tablet),

(brand)-[:BRAND\_MANUFACTURES\_MEDICINE\_TYPE]->(syrup),

(brand)-[:BRAND\_MANUFACTURES\_MEDICINE\_TYPE]->(powder);

// Create medicine products

CREATE (tablet\_product:MedicineProduct {name: 'Tablet'}),

(syrup\_product:MedicineProduct {name: 'Syrup'}),

(powder\_product:MedicineProduct {name: 'Powder'});

// Create relationships for medicine types and products

MATCH (type:MedicineType)

CREATE (type)-[:MEDICINE\_PRODUCT\_OF\_TYPE]->(tablet\_product),

(type)-[:MEDICINE\_PRODUCT\_OF\_TYPE]->(syrup\_product),

(type)-[:MEDICINE\_PRODUCT\_OF\_TYPE]->(powder\_product);

1)List the names of different medicines considered in your graph

MATCH (type:MedicineType)

RETURN type.name;

2)List the medicine that are highly Used in Rajasthan.

MATCH (:State {name: 'Rajasthan'})<-[:MEDICINE\_TYPE\_USED\_IN\_STATE {usage\_percentage: 90}]-(:MedicineBrand)-[:BRAND\_MANUFACTURES\_MEDICINE\_TYPE]->(type:MedicineType)

RETURN type.name;

3.List the highly used tablet in Gujarat.

MATCH (:State {name: 'Gujarat'})<-[:MEDICINE\_TYPE\_USED\_IN\_STATE {usage\_percentage: 90}]-(:MedicineBrand)-[:BRAND\_MANUFACTURES\_MEDICINE\_TYPE]->(:MedicineType {name: 'Tablet'})

RETURN 'Tablet' AS Medicine\_Type;

4.List the medicine names manufacturing “Powder”

MATCH (:MedicineType {name: 'Powder'})<-[:MEDICINE\_PRODUCT\_OF\_TYPE]-(:MedicineProduct)

RETURN 'Powder' AS Medicine\_Type;

# SLIP 17

Q2)

// Create plants

CREATE (rose:Plant {name: 'Rose', type: 'Flowering', quantity: 1000}),

(lily:Plant {name: 'Lily', type: 'Flowering', quantity: 800}),

(tulip:Plant {name: 'Tulip', type: 'Flowering', quantity: 600}),

(cactus:Plant {name: 'Cactus', type: 'Succulent', quantity: 1200}),

(fern:Plant {name: 'Fern', type: 'Foliage', quantity: 700});

// Create fertilizers

CREATE (nitrogen: Fertilizer {name: 'Nitrogen'}),

(phosphorus: Fertilizer {name: 'Phosphorus'}),

(potassium: Fertilizer {name: 'Potassium'});

// Create products

CREATE (pots: Product {name: 'Pots'}),

(shovel: Product {name: 'Shovel'}),

(watering\_can: Product {name: 'Watering Can'});

// Create customers

CREATE (customer1: Customer {name: 'Alice'}),

(customer2: Customer {name: 'Bob'}),

(customer3: Customer {name: 'Charlie'});

// Create app

CREATE (app: App {name: 'NurseryApp'});

// Create suppliers

CREATE (supplier1: Supplier {name: 'Supplier1'}),

(supplier2: Supplier {name: 'Supplier2'}),

(supplier3: Supplier {name: 'Supplier3'});

// Create relationships

MATCH (plant: Plant), (supplier: Supplier)

WHERE plant.name = 'Creepers'

CREATE (supplier)-[:SUPPLIES]->(plant);

MATCH (plant: Plant {name: 'Rose'})<-[:PURCHASED\_PLANT {quantity: 1000}]-(:Customer)

RETURN plant;

MATCH (plant: Plant {name: 'Flowering'}) RETURN plant;

MATCH (supplier: Supplier)-[:SUPPLIES]->(plant: Plant {name: 'Creepers'})

RETURN supplier.name

ORDER BY plant.quantity DESC;

1.List the types of plants from your graph model

MATCH (plant:Plant)

RETURN DISTINCT plant.type;

2.List the popular flowering plants.

MATCH (plant:Plant {type: 'Flowering'})

WHERE plant.quantity > 500

RETURN plant.name;

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

MATCH (plant:Plant)

WHERE plant.quantity > 500 AND date(plant.date) >= date()-2

RETURN plant.name;

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

MATCH (supplier:Supplier)-[:SUPPLIES]->(:Plant {name: 'Creepers'})

RETURN supplier.name

ORDER BY supplier.name DESC;

# SLIP 18

Q2)

// Create laptops

CREATE (dell:Laptop {name: 'DELL', characteristics: 'High performance, Sleek design'}),

(hp:Laptop {name: 'HP', characteristics: 'Reliable, Durable'}),

(lenovo:Laptop {name: 'Lenovo', characteristics: 'Affordable, Lightweight'}),

(apple:Laptop {name: 'Apple', characteristics: 'Premium build, MacOS'});

// Create customers

CREATE (customer1:Customer {name: 'Alice'}),

(customer2:Customer {name: 'Bob'}),

(customer3:Customer {name: 'Charlie'});

// Create relationships for purchases

MATCH (customer:Customer)

CREATE (customer)-[:PURCHASED\_BY]->(dell),

(customer)-[:PURCHASED\_BY]->(hp),

(customer)-[:PURCHASED\_BY]->(apple);

// Create recommendations

CREATE (customer1)-[:RECOMMENDED\_BY]->(lenovo),

(customer2)-[:RECOMMENDED\_BY]->(dell),

(customer3)-[:RECOMMENDED\_BY]->(hp);

1.List the characteristics of…….. laptop.

MATCH (laptop:Laptop {name: 'DELL'})

RETURN laptop.characteristics;

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

MATCH (customer:Customer)-[:PURCHASED\_BY]->(:Laptop {name: 'DELL'})

RETURN customer.name;

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

MATCH (customer:Customer)-[:PURCHASED\_BY]->(:Laptop)

WHERE date(customer.purchase\_date) = date('2023-01-26')

RETURN customer.name;

4.List the most recommended device.

MATCH (:Customer)-[:RECOMMENDED\_BY]->(laptop:Laptop)

RETURN laptop.name, COUNT(\*) AS recommendations

ORDER BY recommendations DESC

LIMIT 1;

# SLIP 19

Q2)

// Create doctors

CREATE (doc1:Doctor {name: 'Dr. Rajesh'}),

(doc2:Doctor {name: 'Dr. Priya'}),

(doc3:Doctor {name: 'Dr. Anand'}),

(doc4:Doctor {name: 'Dr. Sneha'});

// Create specializations

CREATE (ortho:Specialization {name: 'Orthopedic'}),

(gyn:Specialization {name: 'Gynaec'}),

(heart:Specialization {name: 'Heart Specialist'}),

(cancer:Specialization {name: 'Cancer Specialist'}),

(ent:Specialization {name: 'ENT'}),

(pediatrics:Specialization {name: 'Pediatrics'});

// Create hospitals

CREATE (hospital1:Hospital {name: 'Seren Medows'}),

(hospital2:Hospital {name: 'Apollo Hospital'}),

(hospital3:Hospital {name: 'Ruby Hall'});

// Create clinics

CREATE (clinic1:Clinic {name: 'Pune Clinic'}),

(clinic2:Clinic {name: 'Lotus Clinic'});

// Create areas

CREATE (area1:Area {name: 'Koregaon Park'}),

(area2:Area {name: 'Baner'});

// Create reviewers

CREATE (reviewer1:Reviewer {name: 'Alice'}),

(reviewer2:Reviewer {name: 'Bob'}),

(reviewer3:Reviewer {name: 'Charlie'});

// Create relationships

MATCH (doc:Doctor)

WHERE doc.name = 'Dr. Rajesh' OR doc.name = 'Dr. Priya'

CREATE (doc)-[:SPECIALIZES\_IN]->(ortho),

(doc)-[:WORKS\_AT]->(clinic1);

MATCH (doc:Doctor)

WHERE doc.name = 'Dr. Anand' OR doc.name = 'Dr. Sneha'

CREATE (doc)-[:SPECIALIZES\_IN]->(pediatrics),

(doc)-[:WORKS\_AT]->(hospital1);

MATCH (doc:Doctor {name: 'Dr. Rajesh'})-[:WORKS\_AT]->(clinic1),

(doc:Doctor {name: 'Dr. Priya'})-[:WORKS\_AT]->(clinic2),

(doc:Doctor {name: 'Dr. Anand'})-[:WORKS\_AT]->(hospital1),

(doc:Doctor {name: 'Dr. Sneha'})-[:WORKS\_AT]->(hospital2);

// Create recommendations

MATCH (doctor:Doctor)

CREATE (reviewer1)-[:RECOMMENDED\_BY]->(doctor),

(reviewer2)-[:RECOMMENDED\_BY]->(doctor),

(reviewer3)-[:RECOMMENDED\_BY]->(doctor);

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

MATCH (doctor:Doctor)-[:SPECIALIZES\_IN]->(:Specialization {name: 'Orthopedic'})-[:WORKS\_AT]->(:Clinic)-[:LOCATED\_IN]->(:Area {name: 'Koregaon Park'})

RETURN doctor.name;

2. List the doctors who has specialization in \_\_\_\_

MATCH (doctor:Doctor)-[:SPECIALIZES\_IN]->(:Specialization {name: 'Pediatrics'})

RETURN doctor.name;

3. List the most recommended Pediatrics in Seren Medows.

MATCH (doctor:Doctor)-[:SPECIALIZES\_IN]->(:Specialization {name: 'Pediatrics'})-[:WORKS\_AT]->(:Hospital {name: 'Seren Medows'})

RETURN doctor.name, COUNT(\*) AS recommendations

ORDER BY recommendations DESC

LIMIT 1;

4.List all the who visits more than 2 hospitals

MATCH (doctor:Doctor)-[:WORKS\_AT]->(hospital:Hospital)

WITH doctor, COUNT(DISTINCT hospital) AS numHospitals

WHERE numHospitals > 2

RETURN doctor.name;

# SLIP 20

Q2.

// Create authors

CREATE (author1:Author {name: 'J.K. Rowling'}),

(author2:Author {name: 'Stan Lee'}),

(author3:Author {name: 'Stephen King'});

// Create books

CREATE (book1:Book {title: 'Harry Potter', type: 'Fantasy'}),

(book2:Book {title: 'Spiderman', type: 'Comics'}),

(book3:Book {title: 'It', type: 'Horror'});

// Create publishers

CREATE (publisher1:Publisher {name: 'Scholastic'}),

(publisher2:Publisher {name: 'Marvel'}),

(publisher3:Publisher {name: 'Penguin Random House'});

// Create readers

CREATE (reader1:Reader {name: 'Alice'}),

(reader2:Reader {name: 'Bob'}),

(reader3:Reader {name: 'Charlie'});

// Create relationships

MATCH (author:Author {name: 'J.K. Rowling'})-[:WROTE]->(book:Book {title: 'Harry Potter'}),

(book)-[:PUBLISHED\_BY]->(publisher1),

(author2:Author {name: 'Stan Lee'})-[:WROTE]->(book2:Book {title: 'Spiderman'}),

(book2)-[:PUBLISHED\_BY]->(publisher2),

(author3:Author {name: 'Stephen King'})-[:WROTE]->(book3:Book {title: 'It'}),

(book3)-[:PUBLISHED\_BY]->(publisher3);

// Create recommendations

MATCH (reader:Reader)

CREATE (reader)-[:READ\_BY]->(book1),

(reader)-[:READ\_BY]->(book2),

(reader)-[:READ\_BY]->(book3),

(reader)-[:RECOMMENDED\_BY]->(author1),

(reader)-[:RECOMMENDED\_BY]->(author2),

(reader)-[:RECOMMENDED\_BY]->(author3);

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

MATCH (author:Author)-[:WROTE]->(book:Book {type: 'Comics'})

RETURN author.name;

2.Count no. of readers of \_\_\_\_\_book published by “Sage”.

MATCH (book:Book {title: 'Harry Potter'})-[:PUBLISHED\_BY]->(:Publisher {name: 'Scholastic'})<-[:READ\_BY]-(reader:Reader)

RETURN COUNT(DISTINCT reader) AS numReaders;

3.List all the publisher whose name starts with “N”

MATCH (publisher:Publisher)

WHERE toLower(substring(publisher.name, 0, 1)) = 'n'

RETURN publisher.name;

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

MATCH (reader:Reader)-[:RECOMMENDED\_BY]->(author:Author),

(author)-[:WROTE]->(book:Book {title: 'Harry Potter'})

WHERE reader.rating >= 3

RETURN reader.name;