ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Συστήματα Ανάλυσης & Διαχείρισης Μεγάλων Δεδομένων

Διδάσκων Καθηγητής
Ι. Κωτίδης

Φροντιστήριο 3

Neo4j-Cypher

Καπέτης Χρυσόστομος
mkap@aueb.gr

Food Delivery Demo

Θεωρείστε μια εφαρμογή διανομής παραγγελιών εστιατορίου (π.χ. efood) μέσω της οποίας:

- Οι πελάτες έχουν την δυνατότητα να παραγγείλουν φαγητό, αναψυκτικά και ποτά από διάφορα εστιατόρια.
- Τα εστιατόρια ειδοποιούνται αυτόματα για τις παραγγελίες των πελατών τις οποίες ετοιμάζουν.
- Οι παραγγελίες παραδίδονται στους πελάτες από τους διανομείς.
- Η εφαρμογή προτείνει εστιατόρια κοντά στην τοποθεσία του πελάτη, εστιατόρια από τα οποία παρήγγειλαν άλλοι πελάτες καθώς και τρόφιμα και ποτά που οι πελάτες προτιμούν να παραγγέλνουν.

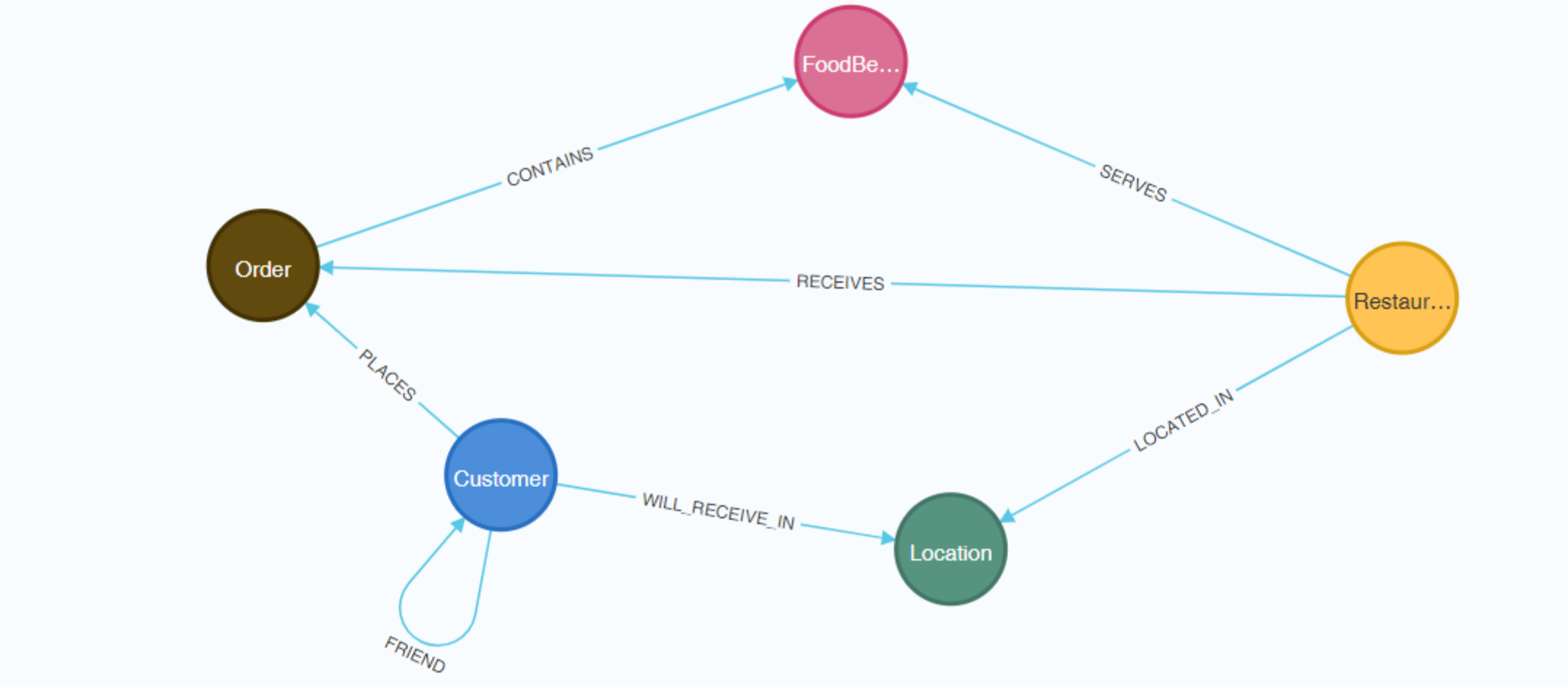
FoodDelivery-GraphDB: Εννοιολογικό Μοντέλο

1. **Προσδιορισμός Κόμβων (Nodes)**
   - Restaurant: Συνεργαζόμενα εστιατόρια
   - Customer: Πελάτες
   - Location: Τοποθεσίες εστιατορίων και πελατών
   - Order: Οι παραγγελίες των πελατών
   - FoodBeverage: Κατάλογος φαγητών και ποτών

2. Προσδιορισμός ακμών (Edges)

- SERVES: Ένα εστιατόριο σερβίρει συγκεκριμένα φαγητά.
- LOCATED_IN: Ένα εστιατόρια βρίσκεται σε μία τοποθεσία.
- WILL_RECEIVE_IN: Ένας πελάτης θα παραλάβει την παραγγελία σε μία τοποθεσία.
- PLACES: Ένας πελάτης κάνει παραγγελίες.
- CONTAINS: Περιεχόμενο παραγγελίας (είδη, ποσότητα, τιμή)
- RECEIVES: Ένα εστιατόριο λαμβάνει παραγγελίες.
- FRIEND: Ένας πελάτης είναι φίλος με κάποιον άλλον πελάτη.

FoodDelivery - GraphDB (Εννοιολογικό Μοντέλο)



Create FoodDelivery Graph

Create Unique Constraints

// Create Unique Constraints for all Node Properties.

```
CREATE CONSTRAINT FOR (r:Restaurant) REQUIRE r.name IS UNIQUE;
CREATE CONSTRAINT FOR (l:Location) REQUIRE l.locationID IS UNIQUE;
CREATE CONSTRAINT FOR (fb:FoodBeverage) REQUIRE fb.name IS UNIQUE;
CREATE CONSTRAINT FOR (c:Customer) REQUIRE c.email IS UNIQUE;
CREATE CONSTRAINT FOR (o:Order) REQUIRE o.orderID IS UNIQUE;
```

Create Nodes (1)

//Restaurants

```
CREATE (:Restaurant {name: "Subway", description: "American * Sandwiches * Fast Food * Salad", opening: "10:00 AM", closing: "16:00 PM", deliveryCharge: 1.5});
CREATE (:Restaurant {name: "Pancake House", description: "American * Desserts * Pancakes * Pasta",opening: "9:00 AM", closing: "16:00 PM", deliveryCharge: 0.0});
CREATE (:Restaurant {name: "Pizza Hut", description: "Pizza * Pasta",opening: "11:00 AM", closing: "17:00 PM", deliveryCharge: 5.0});
CREATE (:Restaurant {name: "Jamba Juice", description: "Smoothies * Superfoods",opening: "10:00 AM", closing: "19:00 PM", deliveryCharge: 1.5});
```

//FoodBeverages

```
//FoodBeverages -- subway
CREATE (:FoodBeverage {name: "Sliced Chicken and Ham", amount:149.50});
CREATE (:FoodBeverage {name: "Subway Melt", amount:184.00});
CREATE (:FoodBeverage {name: "Meatball Marinara Regular Meal", amount:281.75});
CREATE (:FoodBeverage {name: "Coke in Can", amount:51.75});
```

Create Nodes (2)

//FoodBeverages -- pancake house

```
CREATE (:FoodBeverage {name: "3 Pcs. Chocolate Marble Pancake", amount:260.00});
CREATE (:FoodBeverage {name: "3 Pcs. Walnut Pancake", amount:277.00});
CREATE (:FoodBeverage {name: "Bacon Waffle", amount:260.00});
CREATE (:FoodBeverage {name: "Spaghetti with Meat Sauce Large", amount:383.00});
CREATE (:FoodBeverage {name: "Minute Maid 250ml Orange Juice", amount:58.00});
```

//FoodBeverages -- pizza hut

```
CREATE (:FoodBeverage {name: "Super Supreme Pan Pizza Regular", amount:375.00});
CREATE (:FoodBeverage {name: "Meat Lovers Regular", amount:375.00});
CREATE (:FoodBeverage {name: "Spaghetti Bolognese With Meatsauce", amount:129.00});
CREATE (:FoodBeverage {name: "Fried Chicken", amount:395.00});
CREATE (:FoodBeverage {name: "KitKat Pops", amount:109.00});
```

//FoodBeverages -- Jamba juice

```
CREATE (:FoodBeverage {name: "Caribbean Passion", amount:200.00});
CREATE (:FoodBeverage {name: "Berry Pomegranate Power", amount:220.00});
CREATE (:FoodBeverage {name: "Strawberry Whirl", amount:220.00});
CREATE (:FoodBeverage {name: "Acai Supercharger", amount:220.00});
```

Create Nodes (3)

// Customers
CREATE (:Customer {name: "Kylie", email:"kylie@gmail.com", password:"123456"});
CREATE (:Customer {name: "Ivana", email:"ivana@gmail.com", password:"6789"});
CREATE (:Customer {name: "Fletcher", email:"fletcher@yahoo.com", password:"191919"});
CREATE (:Customer {name: "Hugh", email:"hugh@outlook.com", password:"12345"});
CREATE (:Customer {name: "John", email:"John@gmail.com", password:"1321"});

//Restaurant Locations
//Location -- Jamba juice
CREATE (:Location {locationID: 1, description: "Ayala Malls - The 30th", geoLocation: point({longitude:121.0647222, latitude:14.5802778})});

//Location -- Subway
CREATE (:Location {locationID: 2, description: "Rockwell Business Center Tower 3", geoLocation: point({longitude:121.0677778, latitude:14.5891667})});

//Location -- Pancake house
CREATE (:Location {locationID: 3, description: "The Medical City - Ortigas", geoLocation: point({longitude:121.0691667, latitude:14.5891667})});

Create Nodes (4)

//Location -- Pizza hut
CREATE (:Location {locationID: 4, description: "Robinsons Galleria", geoLocation: point({longitude:121.0600000, latitude:14.5908333})});

//Customer Locations
//Location -- Kylie
CREATE (:Location {locationID: 5, description: "The Linden Suites", geoLocation: point({longitude:121.0602778, latitude:14.5813889})});

//Location -- Ivana
CREATE (:Location {locationID: 6, description: "Tektite Tower", geoLocation: point({longitude:121.0616667, latitude:14.5822222})});

//Location -- Fletcher
CREATE (:Location {locationID: 7, description: "Ateneo de Manila University School of Medicine", geoLocation: point({longitude:121.0702778, latitude:14.5905556})});

//Location -- Hugh
CREATE (:Location {locationID: 8, description: "MMDA Traffic Engineering Department", geoLocation: point({longitude:121.0672222 , latitude:14.5836111})});

Create Nodes (5)

//Orders

```
CREATE (:Order {orderID:"order1", orderDate:datetime({year:2020, month:7, day:22, hour:11, minute:15}),totalCost:299.00 });
CREATE (:Order {orderID:"order2", orderDate:datetime({year:2020, month:7, day:22, hour:11, minute:20}),totalCost:440.00 });
CREATE (:Order {orderID:"order3", orderDate:datetime({year:2020, month:7, day:22, hour:7, minute:15}),totalCost:660.00 });
CREATE (:Order {orderID:"order4", orderDate:datetime({year:2020, month:7, day:22, hour:7, minute:20}),totalCost:220.00 });
CREATE (:Order {orderID:"order5", orderDate:datetime({year:2020, month:7, day:22, hour:11, minute:35}),totalCost:375.00 });
CREATE (:Order {orderID:"order6", orderDate:datetime({year:2020, month:7, day:22, hour:11, minute:50}),totalCost:420.00 });
CREATE (:Order {orderID:"order7", orderDate:datetime({year:2020, month:7, day:22, hour:11, minute:25}),totalCost:258.00 });
CREATE (:Order {orderID:"order8", orderDate:datetime({year:2020, month:7, day:22, hour:11, minute:30}),totalCost:440.00 });
```

Create Relationships (1)

// Restaurant--SERVES-->FoodBeverage

```
MATCH (r:Restaurant {name: "Subway"}), (fb:FoodBeverage {name: "Sliced Chicken and Ham"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Subway"}), (fb:FoodBeverage {name: "Subway Melt"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Subway"}), (fb:FoodBeverage {name: "Meatball Marinara Regular Meal"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Subway"}), (fb:FoodBeverage {name: "Coke in Can"}) MERGE (r)-[:SERVES]->(fb);

MATCH (r:Restaurant {name: "Pancake House"}), (fb:FoodBeverage {name: "3 Pcs. Chocolate Marble Pancake"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pancake House"}), (fb:FoodBeverage {name: "3 Pcs. Walnut Pancake"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pancake House"}), (fb:FoodBeverage {name: "Bacon Waffle"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pancake House"}), (fb:FoodBeverage {name: "Spaghetti with Meat Sauce Large"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pancake House"}), (fb:FoodBeverage {name: "Minute Maid 250ml Orange Juice"}) MERGE (r)-[:SERVES]->(fb);
```

Create Relationships (2)

// Restaurant--SERVES-->FoodBeverage

```
MATCH (r:Restaurant {name: "Pizza Hut"}), (fb:FoodBeverage {name: "Super Supreme Pan Pizza Regular"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pizza Hut"}), (fb:FoodBeverage {name: "Meat Lovers Regular"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pizza Hut"}), (fb:FoodBeverage {name: "Spaghetti Bolognese With Meatsauce"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pizza Hut"}), (fb:FoodBeverage {name: "Fried Chicken"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Pizza Hut"}), (fb:FoodBeverage {name: "KitKat Pops"}) MERGE (r)-[:SERVES]->(fb);

MATCH (r:Restaurant {name: "Jamba Juice"}), (fb:FoodBeverage {name: "Caribbean Passion"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Jamba Juice"}), (fb:FoodBeverage {name: "Berry Pomegranate Power"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Jamba Juice"}), (fb:FoodBeverage {name: "Strawberry Whirl"}) MERGE (r)-[:SERVES]->(fb);
MATCH (r:Restaurant {name: "Jamba Juice"}), (fb:FoodBeverage {name: "Acai Supercharger"}) MERGE (r)-[:SERVES]->(fb);
```

Create Relationships (3)

// Restaurant --LOCATED_IN --> Location

```
MATCH (r:Restaurant {name: "Jamba Juice"}), (l:Location {locationID:1}) MERGE (r)-[:LOCATED_IN]->(l);
MATCH (r:Restaurant {name: "Subway"}), (l:Location {locationID:2}) MERGE (r)-[:LOCATED_IN]->(l);
MATCH (r:Restaurant {name: "Pancake House"}), (l:Location {locationID:3}) MERGE (r)-[:LOCATED_IN]->(l);
MATCH (r:Restaurant {name: "Pizza Hut"}), (l:Location {locationID:4}) MERGE (r)-[:LOCATED_IN]->(l);
```

// Customer --WILL_RECEIVE_IN --> Location

```
MATCH (c:Customer {email: "kylie@gmail.com"}), (l:Location {locationID:5}) MERGE (c)-[:WILL_RECEIVE_IN]->(l);
MATCH (c:Customer {email: "ivana@gmail.com"}), (l:Location {locationID:6}) MERGE (c)-[:WILL_RECEIVE_IN]->(l);
MATCH (c:Customer {email: "fletcher@yahoo.com"}), (l:Location {locationID:7}) MERGE (c)-[:WILL_RECEIVE_IN]->(l);
MATCH (c:Customer {email: "hugh@outlook.com"}), (l:Location {locationID:8}) MERGE (c)-[:WILL_RECEIVE_IN]->(l);
```

Create Relationships (4)

//Customer --PLACES --> Order

```
MATCH (c:Customer {email: "fletcher@yahoo.com"}), (o:Order {orderID: "order1"}) MERGE (c)-[:PLACES]->(o);
MATCH (c:Customer {email: "fletcher@yahoo.com"}), (o:Order {orderID: "order2"}) MERGE (c)-[:PLACES]->(o);

MATCH (c:Customer {email: "kylie@gmail.com"}), (o:Order {orderID: "order3"}) MERGE (c)-[:PLACES]->(o);
MATCH (c:Customer {email: "kylie@gmail.com"}), (o:Order {orderID: "order4"}) MERGE (c)-[:PLACES]->(o);

MATCH (c:Customer {email: "ivana@gmail.com"}), (o:Order {orderID: "order5"}) MERGE (c)-[:PLACES]->(o);
MATCH (c:Customer {email: "ivana@gmail.com"}), (o:Order {orderID: "order6"}) MERGE (c)-[:PLACES]->(o);

MATCH (c:Customer {email: "hugh@outlook.com"}), (o:Order {orderID: "order7"}) MERGE (c)-[:PLACES]->(o);
MATCH (c:Customer {email: "hugh@outlook.com"}), (o:Order {orderID: "order8"}) MERGE (c)-[:PLACES]->(o);
```

Create Relationships (5)

//Order- CONTAINS ->FoodBevrage

```
MATCH (o:Order {orderID: "order1"}), (fb:FoodBeverage {name: "Sliced Chicken and Ham"}) MERGE (o)-[:CONTAINS {quantity: 2, unitCost: 149.50, computedCost: 2 * 149.50}]->(fb);
MATCH (o:Order {orderID: "order2"}), (fb:FoodBeverage {name: "Berry Pomegranate Power"}) MERGE (o)-[:CONTAINS {quantity: 2, unitCost: 220.00, computedCost: 2 * 220.00}]->(fb);

MATCH (o:Order {orderID: "order3"}), (fb:FoodBeverage {name: "3 Pcs. Walnut Pancake"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 277.00, computedCost: 1 * 277.00}]->(fb);
MATCH (o:Order {orderID: "order3"}), (fb:FoodBeverage {name: "Spaghetti with Meat Sauce Large"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 383.00, computedCost: 1 * 383.00}]->(fb);

MATCH (o:Order {orderID: "order4"}), (fb:FoodBeverage {name: "Berry Pomegranate Power"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 220.00, computedCost: 1 * 220.00}]->(fb);
MATCH (o:Order {orderID: "order5"}), (fb:FoodBeverage {name: "Super Supreme Pan Pizza Regular"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 375.00, computedCost: 1 * 375.00}]->(fb);
```

Create Relationships (6)

//Order- CONTAINS ->FoodBevrage

```
MATCH (o:Order {orderID: "order6"}), (fb:FoodBeverage {name: "Caribbean Passion"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 200.00, computedCost: 1 * 200.00}]->(fb);
MATCH (o:Order {orderID: "order6"}), (fb:FoodBeverage {name: "Strawberry Whirl"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 220.00, computedCost: 1 * 220.00}]->(fb);

MATCH (o:Order {orderID: "order7"}), (fb:FoodBeverage {name: "Spaghetti Bolognese With Meatsauce"}) MERGE (o)-[:CONTAINS {quantity: 2, unitCost: 129.00, computedCost: 2 * 129.00}]->(fb);

MATCH (o:Order {orderID: "order8"}), (fb:FoodBeverage {name: "Berry Pomegranate Power"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 220.00, computedCost: 2 * 220.00}]->(fb);
MATCH (o:Order {orderID: "order8"}), (fb:FoodBeverage {name: "Strawberry Whirl"}) MERGE (o)-[:CONTAINS {quantity: 1, unitCost: 220.00, computedCost: 1 * 220.00}]->(fb);
```

Create Relationships (7)

//Restaurant --RECEIVES-->Order

```
MATCH (r:Restaurant {name: "Subway"}),(o:Order {orderID: "order1"}) MERGE (r)-[:RECEIVES]->(o);
MATCH (r:Restaurant {name: "Jamba Juice"}),(o:Order {orderID: "order2"}) MERGE (r)-[:RECEIVES]->(o);

MATCH (r:Restaurant {name: "Pancake House"}),(o:Order {orderID: "order3"}) MERGE (r)-[:RECEIVES]->(o);
MATCH (r:Restaurant {name: "Jamba Juice"}),(o:Order {orderID: "order4"}) MERGE (r)-[:RECEIVES]->(o);

MATCH (r:Restaurant {name: "Pizza Hut"}),(o:Order {orderID: "order5"}) MERGE (r)-[:RECEIVES]->(o);
MATCH (r:Restaurant {name: "Jamba Juice"}),(o:Order {orderID: "order6"}) MERGE (r)-[:RECEIVES]->(o);

MATCH (r:Restaurant {name: "Pizza Hut"}),(o:Order {orderID: "order7"}) MERGE (r)-[:RECEIVES]->(o);
MATCH (r:Restaurant {name: "Jamba Juice"}),(o:Order {orderID: "order8"}) MERGE (r)-[:RECEIVES]->(o);
```

Create Relationships (8)

// Create FRIEND relationships

```
MATCH (c1:Customer {name: "Kylie"}), (c2:Customer {name: "Ivana"})  MERGE (c1)-[:FRIEND]->(c2);

MATCH (c2:Customer {name: "Ivana"}), (c3:Customer {name: "Fletcher"}) MERGE (c2)-[:FRIEND]->(c3);

MATCH (c3:Customer {name: "Fletcher"}), (c1:Customer {name: "Kylie"}) MERGE (c3)-[:FRIEND]->(c1);

MATCH (c4:Customer {name: "Hugh"}), (c2:Customer {name: "Ivana"}) MERGE (c4)-[:FRIEND]->(c2);

MATCH (c5:Customer {name: "John"}), (c4:Customer {name: "Hugh"}) MERGE (c5)-[:FRIEND]->(c4);

MATCH (c5:Customer {name: "John"}), (c1:Customer {name: "Kylie"}) MERGE (c5)-[:FRIEND]->(c1);

MATCH (c5:Customer {name: "John"}), (c3:Customer {name: "Fletcher"}) MERGE (c5)-[:FRIEND]->(c3);
```

Create FoodDelivery Graph

Querying Food Delivery Graph (1)

1. Εμφάνισε όλα τα εστιατόρια

```
MATCH (r:Restaurant)RETURN r
```

2. Εμφάνισε τον τιμοκατάλογο του εστιατορίου ‘Subway’

```
MATCH (r:Restaurant {name: "Subway"})-[:SERVES]->(fb:FoodBeverage)
RETURN fb.name AS Item, fb.amount AS Price

//Εναλλακτικά

MATCH (r:Restaurant)-[:SERVES]->(fb:FoodBeverage)
WHERE r.name = "Subway"
RETURN fb.name AS Item, fb.amount AS Price
```

Querying Food Delivery Graph (2)

3. Εμφάνισε τα εστιατόρια σε απόσταση ακτίνας 1000 μέτρων από την τοποθεσία ενός πελάτη (έστω 'Fletcher').

```
MATCH (c:Customer {name: "Fletcher"})-[:WILL_RECEIVE_IN]->(cl:Location),
          (r:Restaurant)-[:LOCATED_IN]->(rl:Location)
WITH r, rl, point.distance(cl.geoLocation, rl.geoLocation) AS Distance
WHERE Distance <= 1000
RETURN r.name AS Restaurant, rl.description AS RestaurantLocation,Distance
ORDER BY Distance

// εναλλακτικά

MATCH (c:Customer {name: "Fletcher"})-[:WILL_RECEIVE_IN]->(cl:Location),
    (r:Restaurant)-[:LOCATED_IN]->(rl:Location)
WHERE point.distance(cl.geoLocation, rl.geoLocation) <= 1000
RETURN r.name AS Restaurant, rl.description AS RestaurantLocation,
          point.distance(cl.geoLocation, rl.geoLocation) as Distance
ORDER BY Distance
```

Querying Food Delivery Graph (4)

4. Τι έχει παραγγείλει ένας συγκεκριμένος πελάτης και από πού (π.χ. ‘Kylie’).

```
MATCH (c:Customer {name: "Kylie"})-[:PLACES]->(o:Order)-[:CONTAINS]->(fb:FoodBeverage),
    (r:Restaurant)-[:RECEIVES]->(o)
RETURN fb.name AS FoodBeverage, r.name AS Restaurant, o.orderID AS OrderID
ORDER BY OrderID
```

Querying Food Delivery Graph (5)

5. Εμφάνισε το περιεχόμενο μιας συγκεκριμένης παραγγελίας (π.χ. Order3)

```
MATCH (o:Order {orderID: "order3"})-[c:CONTAINS]->(fb:FoodBeverage)
RETURN fb.name AS FoodBeverage,  c.quantity AS Quantity
```

Querying Food Delivery Graph (6)

6. Από ποια άλλα εστιατόρια παραγγέλνουν οι πελάτες ενός συγκεκριμένου εστιατορίου (έστω Pizza Hut).

```
MATCH (r1:Restaurant {name: "Pizza Hut"})-[:RECEIVES]->(o1:Order)<-
[:PLACES]-(c:Customer)-[:PLACES]->(o2:Order)<-[:RECEIVES]-
(r2:Restaurant)
WHERE r2 <> r1
RETURN DISTINCT r2.name AS OtherRestaurants
```

Querying Food Delivery Graph (7)

7. Τι άλλο παρήγγειλαν οι πελάτες που παρήγγειλαν ένα συγκεκριμένο φαγητό ή ποτό (π.χ. 'Berry Pomegranate Power‘);

```
MATCH (fb1:FoodBeverage {name: "Berry Pomegranate Power"})<-[:CONTAINS]-
      (o1:Order)<-[:PLACES]-(c:Customer)-[:PLACES]->(o2:Order)-[:CONTAINS]->
      (fb2:FoodBeverage)
WHERE fb2 <> fb1
RETURN DISTINCT fb2.name AS FoodBeverages
```

Querying Food Delivery Graph (8)

8. **Πόσες παραγγελίες έλαβε κάθε εστιατόριο σε μια συγκεκριμένη ημέρα (π.χ. 22-07-2022)**

```
MATCH (r:Restaurant)-[:RECEIVES]->(o:Order)
WHERE date(o.orderDate) = date("2020-07-22")
RETURN r.name AS Restaurant, COUNT(o) AS TotalOrders
ORDER BY TotalOrders DESC
```

Querying Food Delivery Graph (8)

8. **Πόσες παραγγελίες έλαβε κάθε εστιατόριο σε μια συγκεκριμένη ημέρα (π.χ. 22-07-2022)**

```
MATCH (r:Restaurant)-[:RECEIVES]->(o:Order)
WHERE date(o.orderDate) = date("2020-07-22")
RETURN r.name AS Restaurant, COUNT(o) AS TotalOrders
ORDER BY TotalOrders DESC
```

Querying Food Delivery Graph (9)

9. Εμφανίστε έναν κατάλογο με τα είδη (φαγητά/ποτά) και τη συνολική ποσότητα των παραγγελιών τους. Στον κατάλογο να περιλαμβάνονται μόνο τα είδη με συνολική ποσότητα παραγγελιών μεγαλύτερη από ένα όριο (π.χ. 1 τεμάχιο).

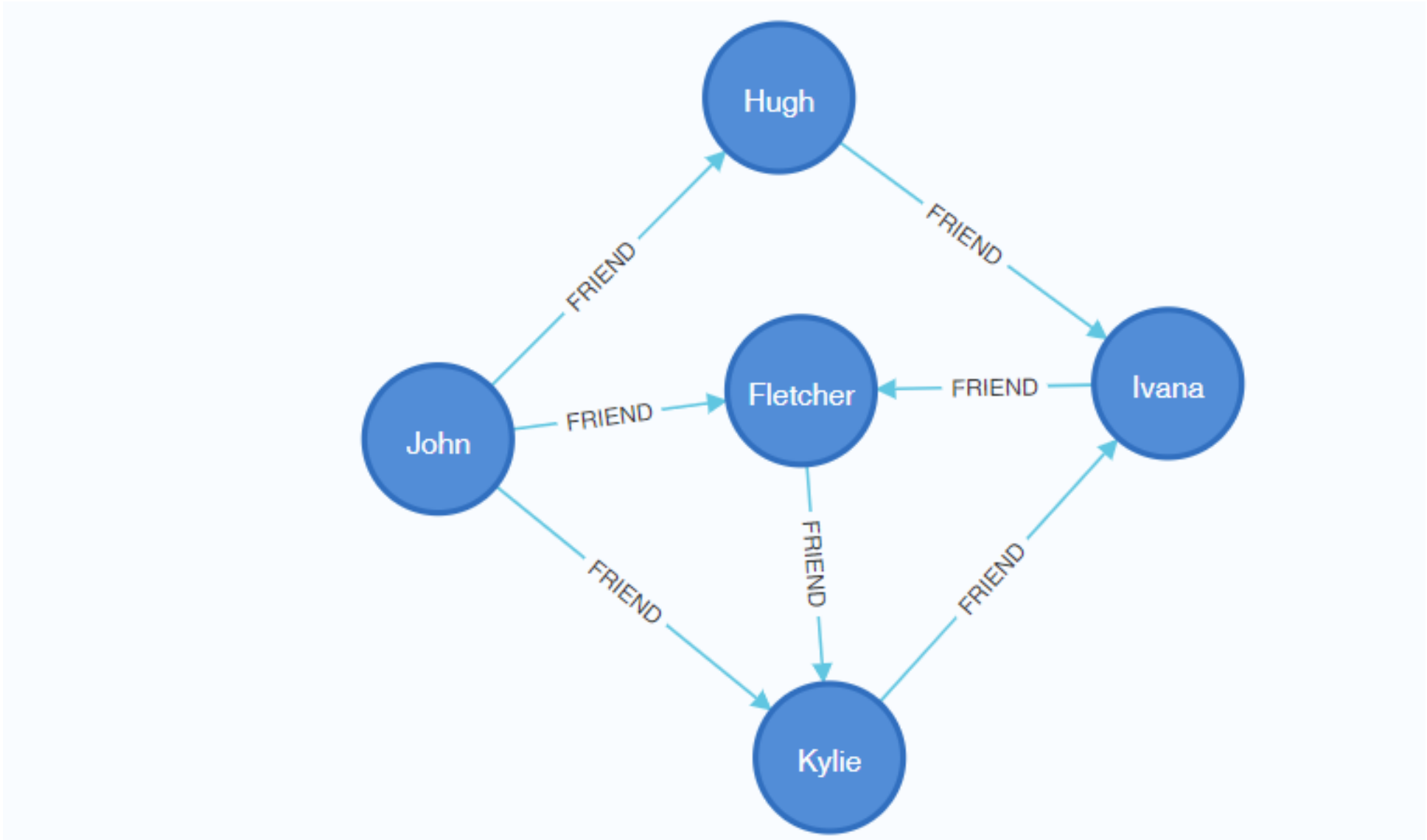
```
MATCH (o:Order)-[c:CONTAINS]->(fb:FoodBeverage)
WITH fb.name AS FoodBeverage, SUM(c.quantity) AS TotalQuantity
WHERE TotalQuantity > 1
RETURN FoodBeverage, TotalQuantity
ORDER BY TotalQuantity DESC
```

Querying Food Delivery Graph (10)

10. Εμφανίστε έναν κατάλογο με την συνολική αξία των παραγγελιών ανα πελάτη.

```
MATCH (c:Customer)-[:PLACES]->(o:Order)
RETURN c.name AS CustomerName, SUM(o.totalCost) AS Total
ORDER BY Total DESC
```

FRIENDS


Hugh
FRIEND
FRIEND
John
FRIEND
Fletcher
FRIEND
Ivana
FRIEND
FRIEND
FRIEND
Kylie

Querying Food Delivery Graph (11)

11. Ποιοι πελάτες είναι μεταξύ τους φίλοι;

```
MATCH (c1:Customer)-[:FRIEND]->(c2:Customer)
RETURN  c1.name AS Customer1, c2.name AS Customer2
```

Querying Food Delivery Graph (12)

12. Ποιοι είναι οι φίλοι των φίλων ενός πελάτη (π.χ. John);

```
MATCH (john:Customer {name: "John"})-[:FRIEND]->(:Customer)-[:FRIEND]->(fof)
WHERE NOT (john)-[:FRIEND]->(fof)
RETURN  DISTINCT fof.name AS FriendOfFriend
```

Querying Food Delivery Graph (12)

13. Από ποια εστιατόρια παραγγέλνουν οι φίλοι ενός πελάτη (έστω Fletcher) ;

```
MATCH (fletcher:Customer {name: "Fletcher"})-[:FRIEND]->(friend:Customer)-[:PLACES]->(o:Order)<-[:RECEIVES]-
(r:Restaurant)
RETURN DISTINCT friend.name AS FriendName, r.name AS RestaurantName
```