

#### Ödev 4 – İleri Cypher Sorguları ile Use Case Geliştirme

Tarif ve malzeme domaini, çok sayıda varlık (malzeme, tarif, tat, teknik vs.) ve bu varlıklar arasında zengin ilişki türleri içermektedir. Bu nedenle bu domain, gerçek hayatla bağlantılı olduğundan ve anlaşılması kolay olduğundan birçok karmaşık senaryoyu modellemek için ideal bir örnektir.

Tarif ve malzeme domain'i üzerinde çalışarak, yemek tarifleri, malzemeler, lezzetler ve pişirme teknikleri arasındaki çok boyutlu ilişkileri inceleyeceğiz.

Bu ödevde, Neo4j graph database'de ileri seviye Cypher sorgu tekniklerini kullanarak use case'ler geliştirdim. Tarif(Recipe) ve malzeme(Ingredients) domain'i üzerinde çalışarak, UNION ALL ile veri birleştirme, WITH, UNWIND, dinamik property erişimi, aggregation işlemleri, graph algoritmaları ve APOC gibi gelişmiş teknikleri uyguladım. EXPLAIN/PROFILE komutları ile performans analizi yaptım.

Ödevde, daha önce oluşturduğum tarif (recipe) graph database'ini kullanacağım. Bu database aşağıdaki node türlerini ve ilişkileri içermektedir:

##### Node Türleri:

- **Recipe:** {id, name, difficulty, time\_minutes}
- **Ingredient:** {id, name, category}
- **Flavor:** {id, name}
- **Technique:** {id, name}

##### Relationships Türleri:

- **CONTAINS:** (Recipe) → (Ingredient)
- **HAS\_FLAVOR:** (Recipe) → (Flavor)
- **USES\_TECHNIQUE:** (Recipe) → (Technique)
- **CAN\_REPLACE:** (Ingredient) → (Ingredient)

### Hem tatlı hem de kremamsı tarifleri tek sorguda getirebilir miyiz?

```
MATCH (r:Recipe)-[:HAS_FLAVOR]->(f:Flavor {name: "Tatlı"}) // Tatlı tarifleri bulur.
```

```
RETURN r.name AS recipe_name, "Sweet" AS type
```

UNION ALL // sonuclari birleştirir.

```
MATCH (r:Recipe)-[:HAS_FLAVOR]->(f:Flavor {name: "Kremamsı"}) // Kremamsı tarifleri bulur.
```

```
RETURN r.name AS recipe_name, "Creamy" AS type
```

UNION ALL: Farklı Sorgu Sonuçlarını Birleştirir.

User: Aura (busengokler@gmail.com)

neo4j\$

```
1 MATCH (r:Recipe)-[:HAS_FLAVOR]->(f:Flavor {name: "Tatlı"})
2 RETURN r.name AS recipe_name, "Sweet" AS type
3 UNION ALL
4 MATCH (r:Recipe)-[:HAS_FLAVOR]->(f:Flavor {name: "Kremamsı"})
5 RETURN r.name AS recipe_name, "Creamy" AS type
```

Table RAW

	recipe_name	type
1	"Sütlaç"	"Sweet"
2	"Kek"	"Sweet"
3	"Cheesecake"	"Creamy"
4	"Magnolia"	"Creamy"

### En çok malzeme kullanan 3 tarifi bulup, bunların ortalama süresini hesaplayabilir miyiz?

```
MATCH (r:Recipe)-[:CONTAINS]->(i:Ingredient)
// Malzeme sayısına göre sıralanması ve ilk 3 tarifi seçilmesi
```

```
WITH r, count(i) AS ingredient_count
```

```
ORDER BY ingredient_count DESC
```

```
LIMIT 3
```

// Seçilen 3 tarifi ortalama süresini hesaplama ve tarifleri toplama

```
WITH collect(r) AS top_recipes,
avg(r.time_minutes) AS avg_time
```

// Toplanan tariflerin tek tek sonuçlarının döndürülmesi

```
UNWIND top_recipes AS recipe
```

```
RETURN recipe.name, recipe.time_minutes, avg_time
```

User: Aura (busengokler@gmail.com)

neo4j\$

```
1 MATCH (r:Recipe)-[:CONTAINS]->(i:Ingredient)
2 WITH r, count(i) AS ingredient_count
3 ORDER BY ingredient_count DESC
4 LIMIT 3
5 WITH collect(r) AS top_recipes, avg(r.time_minutes) AS avg_time
6 UNWIND top_recipes AS recipe
7 RETURN recipe.name, recipe.time_minutes, avg_time
```

Table RAW

	recipe.name	recipe.time_minut	avg_time
1	"Kek"	60.0	65.0
2	"Cheesecake"	90.0	65.0
3	"Sütlaç"	45.0	65.0

## Property isimleri dinamik olarak kullanılabilir miyiz?

MATCH (r:Recipe) //Tarifleri bulur

// Her tarif için erişilecek property name listesini tanımladım.

WITH r, ['name', 'difficulty', 'time\_minutes'] AS properties

// Bu listedeki her bir elemanı ayrı bir satıra açtım

UNWIND properties AS prop

// Her tarif için dinamik olarak erişilen properties'i döndürdüm.

RETURN r.name AS recipe, prop AS property\_name, r[prop] AS property\_value

Burada bir tarifin property\_name'leri ve onların value'ları çıktı olarak gelmektedir.

User: Aura (busengokler@gmail.com)

neo4j\$

```
1 MATCH (r:Recipe)
2 WITH r, ['name', 'difficulty', 'time_minutes'] AS properties
3 UNWIND properties AS prop
4 RETURN r.name AS recipe, prop AS property_name, r[prop] AS property_value
```

	recipe	property_name	property_value
1	"Sütlaç"	"name"	"Sütlaç"
2	"Sütlaç"	"difficulty"	"Kolay"
3	"Sütlaç"	"time_minutes"	45.0
4	"Kek"	"name"	"Kek"
5	"Kek"	"difficulty"	"Orta"
6	"Kek"	"time_minutes"	60.0
7	"Cheesecake"	"name"	"Cheesecake"
8	"Cheesecake"	"difficulty"	"Zor"
9	"Cheesecake"	"time_minutes"	90.0

## Her kategorideki malzemeleri gruplayabilir ve her grubun tarif sayısını hesaplayabilir miyiz?

MATCH (i:Ingredient)-[:CONTAINS]-(r:Recipe)

// Kategoriye göre gruplama ve her kategori için unique tarifleri liste içinde toplama

WITH i.category AS category, collect(DISTINCT r) AS recipes

RETURN category,

size(recipes) AS recipe\_count,

[recipe IN recipes | recipe.name][0..3] AS sample\_recipes

ORDER BY recipe\_count DESC

User: Aura (busengokler@gmail.com)

neo4j\$

```
1 MATCH (i:Ingredient)-[:CONTAINS]-(r:Recipe)
2 WITH i.category AS category, collect(DISTINCT r) AS recipes
3 RETURN category,
4     size(recipes) AS recipe_count,
5     [recipe IN recipes | recipe.name][0..3] AS sample_recipes
6 ORDER BY recipe_count DESC
```

category	recipe_count	sample_recipes
"Tatlandırıcı"	5	["Sütlaç", "Kek", "Cheesecake"]
"Süt Ürünü"	5	["Sütlaç", "Kek", "Cheesecake"]
"Tahıl"	3	["Sütlaç", "Kek", "Magnolia"]
"Aroma"	2	["Sütlaç", "Kek"]
"Hayvansal"	1	["Kek"]
"Hazır Gıda"	1	["Cheesecake"]
"Baharat"	1	["Magnolia"]

## En çok bağlantıya sahip malzemeleri bulabilir miyiz?

MATCH (i:Ingredient)

// Her malzeme kaç tarifte kullanıldı

OPTIONAL MATCH (i)-[:CONTAINS]-(r:Recipe)

WITH i, count(r) AS recipe\_connections

// Her malzemenin kaç diğer malzeme ile yer değiştirebileceği

OPTIONAL MATCH (i)-[:CAN\_REPLACE]-(other:Ingredient)

WITH i, recipe\_connections, count(other) AS replacement\_connections

RETURN i.name, recipe\_connections, replacement\_connections,

(recipe\_connections + replacement\_connections) AS total\_centrality

ORDER BY total\_centrality DESC

LIMIT 5

User: Aura (busengokler@gmail.com)

neo4j\$

```
1 MATCH (i:Ingredient)
2 OPTIONAL MATCH (i)-[:CONTAINS]-(r:Recipe)
3 WITH i, count(r) AS recipe_connections
4 OPTIONAL MATCH (i)-[:CAN_REPLACE]-(other:Ingredient)
5 WITH i, recipe_connections, count(other) AS replacement_connections
6 RETURN i.name,
7       recipe_connections,
8       replacement_connections,
9       (recipe_connections + replacement_connections) AS total_centrality
10 ORDER BY total_centrality DESC
11 LIMIT 5
```

Table RAW

	i.name	recipe_connection	replacement_conr	total_centrality
1	"Şeker"	4	2	6
2	"Tereyağı"	3	0	3
3	"Un"	2	0	2
4	"Süt"	2	0	2
5	"Vanilin"	2	0	2

**APOC:** Neo4j için gelişmiş fonksiyonlar sağlar.

Koşullu veri ekleyebilmek için APOC gerekli.

MATCH (r:Recipe)

CALL apoc.do.when(

r.time\_minutes > 60, // Koşul: Tarifin hazırlık süresi 60 dk'dan fazla mı?

"SET r.time\_min = 'High' RETURN r", //doğruysa çalışacak sorgu

"SET r.time\_min = 'Low' RETURN r", //yanlışsa çalışacak

{r: r} // params

) YIELD value // apoc.do.when'den dönen sonucu almak için

RETURN r.name, r.time\_min

Tarifin adını ve yeni eklenen özelliği döndürür.

User: Aura (busengokler@gmail.com)

neo4j\$

```
1 MATCH (r:Recipe)
2 CALL apoc.do.when(
3   r.time_minutes > 60,
4   "SET r.time_min = 'High' RETURN r",
5   "SET r.time_min = 'Low' RETURN r",
6   {r: r}
7 ) YIELD value
8 RETURN r.name, r.time_min
```

Table RAW

	r.name	r.time_min
1	"Sütlac"	"Low"
2	"Kek"	"Low"
3	"Cheesecake"	"High"
4	"Çikolata Truffle"	"Low"
5	"Magnolia"	"Low"

## EXPLAIN ve PROFILE

Cypher sorgularının nasıl yürütüleceğini anlamak ve performans sorunlarını tespit etmek için kullanılırlar.

- **EXPLAIN:** Sorgunun *planını* gösterir. Verilerin nasıl alınacağını, hangi indekslerin kullanılacağını veya hangi işlemleri yapılacağını açıklar.
- **PROFILE:** Sorguyu *çalıştırır* ve her bir işlem için istatistikler (harcanan zaman, okunan satır sayısı vs.) gösterir.

### EXPLAIN

```
MATCH (r:Recipe)-[:CONTAINS]->(i:Ingredient)
```

```
WHERE i.category = "Süt Ürünleri" AND r.time_minutes < 30
```

```
RETURN r.name, i.name
```

### PROFILE

```
MATCH (r:Recipe)-[:CONTAINS]->(i:Ingredient)
```

```
WHERE i.category = "Süt Ürünleri" AND r.time_minutes < 30
```

```
RETURN r.name, i.name
```

Bu sorgu, malzemeleri (ingredients) tarayıp filtreleyerek 27 db hit (27 veritabanı erişimi) ve 28 ms ile hızlıca tamamlandı. İkinci filtrelemenin 0 db hits ile çalışması, bu adımların oldukça verimli olduğunu ve verilerin etkin bir şekilde kullanıldığını gösteriyor.

```
1 EXPLAIN
2 MATCH (r:Recipe)-[:CONTAINS]->(i:Ingredient)
3 WHERE i.category = "Süt Ürünleri" AND r.time_minutes < 30
4 RETURN r.name, i.name
```

Plan RAW

Cypher version: 5  
Planner: COST  
Runtime: SLOTTED

```
1 PROFILE
2 MATCH (r:Recipe)-[:CONTAINS]->(i:Ingredient)
3 WHERE i.category = "Süt Ürünleri" AND r.time_minutes < 30
4 RETURN r.name, i.name
```

Plan RAW

Cypher version: 5  
Planner: COST  
Runtime: SLOTTED  
27 total db hits in 28 ms.