1**-)Her düğüm oluşturma işlemi için başına bir CREATE anahtar kelimesi eklenmelidir.**

**2-) Düğümler arası ilişkiler kurulurken başta bir kere CREATE yazdıktan sonra, her node işlemi sonrasında virgül konulmalıdır.**

**3- ) (Düğüm)–[:İLİŞKİ]->(DÜĞÜM)**

**4- ) düğümlerde ve ilişkilerde Objesi belirtildikten sonra süslü parantezler açılarak “{}” içlerine özellikler belirtilerek arama veya düğüm oluşturma işlemi yapılabilir.**

* Data Creation&Fix Data Queries

1. **Create Nodes**Açıklama :Verilen düğümler ve özellikleri doğrultusunda düğümler oluşturulur.

* **CREATE (u0:User{name:"Matthew", lastname:"Carol", username:"user0", password:"pass", totalFollowed:65, totalFollowers:36})  
  CREATE (u1:User{name:"Sandra", lastname:"Emily", username:"user1", password:"pass", totalFollowed:21, totalFollowers:28}) …**
* **CREATE (u0)-[:FOLLOW]->(u18) ,  
  (u0)-[:FOLLOW]->(u28), …**
* CREATE (b0:Book{name:"Don Quixote Miguel de Cervantes", totalRead:7, point:1.9})
* CREATE **(u0)-[:READ{rate:4}]->(b137),**  
  (u0)-[:READ**{rate:9}**]->(b8), …
* CREATE (b0)<-[:WRITE]-(a8)

1. **Fix User Data Query / UPDATE QUERY**

* AÇIKLAMA :   
  Her kullanıcı için ayrı ayrı,   
  kullanıcıyı’ı takip eden kullanıcıları bul,  
  bu kullanıcıların takip edenlerin sayısını hesapla sonra  
  kullanıcının totalFollowers değişkenine ata.  
  MATCH (u:User)<-[:FOLLOW]-(f:User)  
  WITH u, COUNT(f) AS totalFollowers  
  SET u.totalFollowers = totalFollowers  
  WITH u  
  MATCH (u)-[:FOLLOW]->(f:User)  
  WITH u, COUNT(f) AS totalFollowed  
  SET u.totalFollowed = totalFollowed

1. **Fix Book Data Query / UPDATE QUERY**

* AÇIKLAMA :   
  Her kitap için, kitabı okuyan kullnaıcıları bul  
  kitabın puan değişkenine, userların oyladığı puanların ortalamasını bul ve ata.  
  MATCH (b:Book)<-[r:READ]-(u:User)   
  WITH b, avg(r.rate) as point, count(u) as totalReaders   
  SET b.totalRead = totalReaders, b.point = round(point, 2)

1. **Fix Author Data Query / UPDATE QUERY**

* AÇIKLAMA :   
  Her yazar için, yazarın yazdığı kitapları bul  
  bu yazarların yazdığı kitapların sayısını hesapla  
  yazarın totalBook değişkenine yazmış olduğu toplam kitap sayısını ata  
  Sonra bu yazarın kitaplarındaki puanlarını topla, sonra kitapların sayısını bul  
  ve yazarın puanı’na bu değeri ata.  
  MATCH (a:Author)-[:WRITE]->(b:Book)  
  WITH a, COUNT(b) AS totalBook  
  SET a.totalBook = totalBook  
  WITH a  
  MATCH (a:Author)-[:WRITE]->(b:Book)  
  WITH a, sum(b.point) as totalPoints, count(b) as totalBooks  
  SET a.point = round(totalPoints / totalBooks,2)
* Read Data Queries
* AÇIKLAMA : Bütün kullanıcı objelerini döndürür.  
  Match (u:User) return u;
* AÇIKLAMA : Kullanıcıların arasında belirtilen $userId değerine sahip userlar’ları döndürür. Match (u:User) Where ID(u)= $userId return u;
* AÇIKLAMA : Kullanıcıların arasında isimleri “Emin” olan kullanıcılardan ilk 10 tanesini döndürür. Match (u:User{name:”Emin”}) return u Limit 10
* AÇIKLAMA : Her kullanıcıyı ve kullanıcının okuduğu kitapları ve okuma ilişkisini döndürür.  
  MATCH (u:User)-[r:READ]->(b:Book) RETURN u,r,b
* AÇIKLAMA : Verilen id’ye sahip userın takip ettiği userları döndürür.  
  MATCH (u:User)-[:FOLLOW]->(f:User)  
  WHERE ID(u) = $userId   
  RETURN f
* **AÇIKLAMA : !!! 🡪 Arkadaş Önerisi**  
  Her kullanıcının takip ettiği kullanıcının (2. şahısların) takip ettiği ancak asıl kullanıcının (1.şahısların) takip etmediği kullanıcıları (yani 3. Şahısları) bulur.  
   Kısacası arkaşlarının arkadaşlarının sana önerilmesi için kullanılıyor.  
  MATCH (u:User)-[:FOLLOW]->(fu:User)-[:FOLLOW]->(f:User)  
  WHERE ID(u) = $userId   
  AND NOT (u)-[:FOLLOW]->(f)  
  WITH f, COUNT(DISTINCT fu) AS num\_followers   
  ORDER BY num\_followers DESC   
  RETURN f LIMIT 5
* MATCH (u:User) WHERE ID(u) = $userId   
  MATCH (u)-[:READ]->(b:Book)   
  RETURN u,b
* MATCH (b:Book) RETURN b ORDER BY b.point DESC LIMIT 5
* **AÇIKLAMA : !!! 🡪 Kitap Önerisi**  
  Kullanıcının takip ettiği kullanıcıların en çok okuduğu  
  Ancak kullanıcının okumadığı kitaplardan ilk 5’ini döndürür.  
  MATCH (u:User) WHERE ID(u) = $userId   
  MATCH (u)-[:FOLLOW]->(u2:User)-[:READ]->(b:Book)  
  WHERE NOT (b)<-[:READ]-(u)  
  WITH b, COUNT(DISTINCT u2) AS num\_readers   
  ORDER BY num\_readers DESC LIMIT 5   
  MATCH (u:User)-[:READ]->(b)  
  RETURN DISTINCT b
* **Delete Data Queries**
* AÇIKLAMA : Bütün objeleri siler.  
  MATCH (n) DETACH DELETE n
* AÇIKLAMA : Her kullanıcının okuduğu kitap ilişkisinin   
  önce bağlantısını koparır sonra siler.  
  MATCH (u:User)-[r:READ]->(b:Book)   
  WHERE ID(u)=$userId  
  AND ID(b) = $bookId DETACH DELETE r
* **Update Data Queries**
* AÇIKLAMA : Kullanıcının başka bir kullanıcıyı 1 kere takip eder.   
  (Create olursa aynı kişi aynı kişiyi defalarca takip edebilir.)   
  MATCH (u:User) WHERE ID(u) = $userId   
  MATCH (u2:User) WHERE ID(u2) = $friendUserId   
  MERGE (u)-[f:FOLLOW]->(u2)
* **Read Data Queries**

AÇIKLAMA : Kitabı okuyan bütün kullanıcıları ve bu kullanıcıların verdiği oyları döndürür.

* MATCH (u:User)-[r:READ]->(b:Book)   
  WHERE ID(b)= $bookId  
  RETURN u, b