SQL Server İle Graf İşleme

SQL Server 2017 versiyonu ile gelen güzel özelliklerden biri de graf işleme özelliğidir. Facebook, Instagram vb. sitelerde yer alan arkadaşlık kayıtları, beğeni kayıtları sistemin arka planında graf yapısı üzerinde tutmaktadır.

SQL Server üzerinde graf yapısı node ve edge olmak üzere iki tablodan oluşmaktadır. Node tablosu, asıl kayıtların tutulduğu tablodur. EDGE tablosu ise bağlantıların tutulduğu tablodur.

Node tablosunda graf yapısında kullanılacak olan ve otomatik atanıp, düzenlenemeyen $node_id isimli bir kolon tutulmaktadır. EDGE tablosuna kayıt girilirken bu kolon değeri kullanılır. EDGE tablosunda da otomatik oluşturulan ve düzenlenemeyen $from_id ve $to_id kolonları bulunmaktadır. Bu kolon değerleri Node tablosundan gelen $node_id değerleridir. Bu bilgilerden sonra örneğimizle devam edelim.

Node Tablolarının Oluşturulması:

```
CREATE TABLE People (
Id int primary key identity,
[Name] nvarchar(100) not null,
) as Node
```

```
CREATE TABLE Books (
Id int primary key identity,
[Name] nvarchar(100) not null,
) as Node
```

EDGE Tablolarının Oluşturulması

```
create TABLE Friends (
ContactDate datetime not null,
) as edge

create TABLE Likes (
LikeDate datetime not null,
) as edge
```

Kitap ve Kişi kayıtlarının girilmesi:

```
insert into People values ('Mustafa Orhan'), ('Muhammed
Yıldız'), ('Bilal Orhan'), ('Said Nur'), ('Hüseyin Varol')
insert into Books values ('SQL Server'),('C#
Yazılım'),('JAVA'),('JavaScript'),('HTML'),('C++')
```

Yeni arkadaş kaydının girilmesi

```
insert into Friends values
        ((select $node_id from People where Id = 7), (select
$node_id from People where Id = 8), '20181121')
```

Bu komutun ilk parametresi EDGE tablosunu anlatırken söz ettiğimiz $from_id ve ikinci parametresi $to_id değerleridir. Son parametre de Friends tablosunu oluştururken girilen kayıt tarihidir.

Yeni Kitap Beğeni kaydının girilmesi

```
insert into Likes values
        ((select $node_id from People where Id = 8), (select
$node_id from Books where Id = 4), '20181121')
```

Sorgu Örnekleri

“Mustafa Orhan” kişisinin beğendiği kitapların listesini alan kod:

```
select Books.Name
from People, Likes, Books
where match (People-(Likes) -> Books)
and People.[Name] = 'Mustafa Orhan'

------------
```

```
--SQL Server
--JavaScript
--C++
```

Bu sorguda graf sorgusunu yapan fonksiyonumuz MATCH fonksiyonudur. Yazdığımız bu sorguda graf yapısını şu şekilde çalıştırmasını istedik. People ve Books tablolarını Likes tablosu üzerinde eşleştir.

“Mustafa Orhan” kişisinin arkadaş listesini gösteren kod:

```
select p2.Name
from People p1, Friends f, People p2
where match (p1-(f) -> p2)
and p1.Name = 'Mustafa Orhan'

-------------------
--Muhammed Yıldız
--Bilal Orhan
--Said Nur
--Hüseyin Varol
```

Not:

Bu sorgularda dikkat edilmesi gereken nokta, grafın tek yönlü çalışıyor olmasıdır. Eğer iki taraflı graf çalıştırmak istiyorsanız iki seçeneğiniz bulunmaktadır.

- Girilen her yeni kayıt için, tersi kayıt girmek
- Select sorgusunda MATCH fonksiyonu iki taraflı çalıştırıp iki sorguyu UNION komutu ile birleştirmek. MATCH fonksiyonu OR, AND, NOT gibi operatorlar ile çalışmadığından bu şekilde bir sorgu çalıştırmak gerekmektedir. Örnek sorgu:

```
select p2.Name
from People p1, Friends f1, Friends f2, People p2
where match (p1-(f1) -> p2)
and p1.Name = 'Hüseyin Varol'
union
select p2.Name
from People p1, Friends f, People p2
where match (p2-(f) -> p1)
and p1.Name = 'Hüseyin Varol'
```

Bu sorgunun ilk parçasında “Hüseyin Varol” kişisinin eklediği arkadaşlar seçiliyorken, ikinci parçasında da “Hüseyin Varol” kişisini ekleyenlerin listesi alınmaktadır. UNION komutu ile bu sorgular tek sonuç seti olarak birleştirilmektedir.