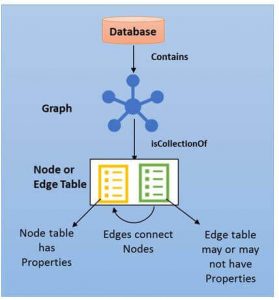
مقدمه Graph Tables

یکی از ویژگی های جدیدی که از نسخه ی SQL Server 2017 پشتیبانی می شود، Graph Table های می باشند. جدول های گراف برای پیاده سازی روابط پیچیده بین جداول کاربرد دارند. این نوع جداول برای پیاده سازی روابط پیچیده چند به چند، روابط سلسله مراتبی و… استفاده خواهند شد و یکی از کاربردهای عمومی که اکثراً با آن آشنایی دارند شبکه های اجتماعی می باشد و البته ساختار این نوع جداول با جداول معمولی در SQL Server تفاوت چندانی ندارد.

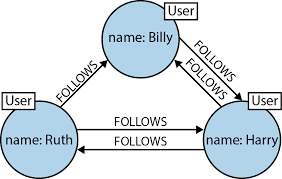
یک گراف دیتابیس (Graph Database) از چه بخش هایی تشکیل می شود:

* Node: هر نود یک موجودیت است.
* Edge: هر یال ارتباط بین دو موجودیت را برقرار می کند.

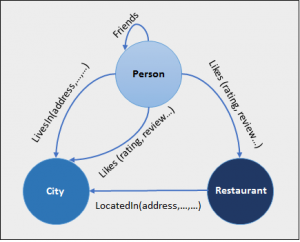
یک گراف دیتابیس مجموعه ای از یال ها و نودها می باشد که جداول نود مانند جداول معمولی در SQL Server می توانند دارای خصوصیت باشند و جداول یال می توانند بدون خصوصیت هم تعریف شوند. تصویر زیر شمای کلی یک گراف دیتابیس را نشان می دهد:



نمونه ی ساده ای از یک گراف دیتابیس در شکل زیر قابل مشاهده است که در آن هر کاربر می تواند کاربر دیگری را Follow کند از توسط کاربر دیگری Follow شود. شاید فکر کنید چنین مدلی در دیتابیس های رابطه ای هم قابل پیاده سازی است ولی این یک مثال ساده است و البته در مدل رابطه ای محدودیت هایی برای این نوع مسائل خواهیم داشت و باید صبر کنید تا مثالی پیچیده تر را باهم مرور کنیم.



نمونه ی دیگری از گراف دیتابیس در شکل زیر نمایش داده شده است که در آن:



یک **فرد** می تواند با فردی دیگر ارتباط **دوستی** داشته باشد

یک **فرد** می تواند در یک **شهر زندگی** کند

یک **رستوران** در یک **شهر** قرار دارد.

یک **فرد** ممکن است یک **رستوران** را **لایک** کند یا به آن **امتیاز** دهد.

جدول Node :

همانطور که عنوان شد جداول نود مانند جداول معمولی دیتابیس دارای خصوصیات می باشند که شما در زمان ایجاد جدول این خصوصیات را نیز تعریف می کنید، اما **نکته ی مهم این است که SQL Server در زمان ایجاد جدول یک ستون دیگر به نام $node\_id به این موجودیت اضافه می کند**. که این فیلد به عنوان کلید یکتای این موجودیت(نه هر رکورد) در دیتابیس محسوب می شود. مقدار این فیلد بصورت JSON می باشد که بصورت خودکار (Computed Field) تولید می شود و اطلاعاتی راجع به رکورد را در خود نگهداری می کند.

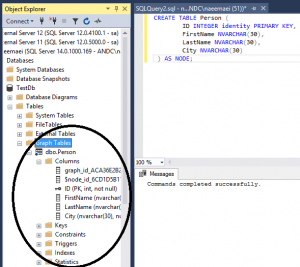
نحوه تعریف یک جدول نود بدین شکل می باشد.

|  |  |
| --- | --- |
| 1  2  3  4  5 | CREATE TABLE      [ database\_name].[ schema\_name ].[schema\_name ].[ table\_name]      ({} [ ,...n ] )      AS [NODE]  [ ; ] |

مثال:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | CREATE TABLE Person (          ID INT identity PRIMARY KEY,          FirstName NVARCHAR (30),  LastName NVARCHAR (30),          City NVARCHAR (30)  ) AS NODE; |

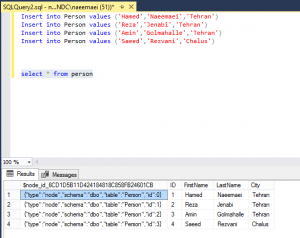
پس از ایجاد این جدول در پنجره Object Explorer می توانید در زیرشاخه ی Graph Tables جدولی که ساخته اید را ببینید:



برای بررسی بیشتر چند رکورد به جدول اضافه می کنیم :

|  |  |
| --- | --- |
| 1  2  3  4 | Insert into Person values ('Hamed','Naeemaei','Tehran')  Insert into Person values ('Reza','Jenabi','Tehran')  Insert into Person values ('Amin','Golmahalle','Tehran')  Insert into Person values ('Saeed','Rezvani','Chalus') |

و در پایان یک Select از جدول می گیریم و نتیجه را می بینیم



همانطور که می بینید در کنار فیلدهایی که تعریف کردیم فیلد $node\_id نیز به جدول اضافه گردیده است

مقدمه Edge در  Graph Tables

برای ارتباط بین دو node در یک گراف از Edge استفاده می کنیم. هر Edge که ارتباط بین دو جدول node را برقرار می کند می تواند دارای خصوصیت (Property) باشد و یا نباشد. این نوع طراحی برای پیاده سازی روابط چند به چند پیچیده بسیار خوب خواهد بود.

برای اینکه بهتر مسئله را درک کنید یک مثال را بررسی خواهیم کرد:

در بخش قبل یک node به نام Person تعریف کرده بودیم که اطلاعات افراد را نگهداری می کرد. حال می خواهیم با ایجاد یک Edge بین افراد ارتباط برقرار کنیم، فرض کنید قصد داریم در یک شبکه ی اجتماعی اطلاعات دوستی افراد را در این مدلسازی پیاده کنیم. برای اینکه از ابتدا مثال را بررسی کنیم، جدول Person را ایجاد کرده و اطلاعات تستی را در آن ثبت می کنیم:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | --------------- Create Person Table ---------------  CREATE TABLE Person (          ID INT identity PRIMARY KEY,          FirstName NVARCHAR(30),  LastName NVARCHAR(30),          City NVARCHAR(30)  ) AS NODE;  --------------- Fill Data in Person Table ---------------  Insert into Person values ('Hamed','Naeemaei','Tehran')  Insert into Person values ('Reza','Jenabi','Tehran')  Insert into Person values ('Amin','Golmahalle','Tehran')  Insert into Person values ('Saeed','Rezvani','Chalus') |

نحوه ساخت Edge Table

حال باید جدولی از نوع Edge یا یال برای ارتباط بین افراد بسازیم، به همین علت ابتدا نحوه ایجاد جدول Edge را توضیح خواهیم داد:

|  |  |
| --- | --- |
| 1  2  3  4  5 | CREATE TABLE      [ database\_name . [ schema\_name ] . | schema\_name . ] table\_name      ( {  } [ ,...n ] )      AS [Edge]  [ ; ] |

ساختار کلی برای ایجاد جدول Edge مانند جداول متداول است با این تفاوت که در انتهای کوئری از عبارت As Edge استفاده می کنیم

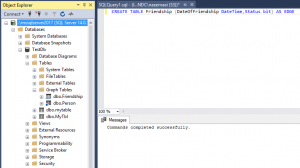
جدول Edge می تواند **دارای خصوصیت** **نباشد**:

|  |  |
| --- | --- |
| 1 | CREATE TABLE Friendship AS EDGE; |

و جدول Edge می تواند **دارای خصوصیت** **باشد**:

|  |  |
| --- | --- |
| 1 | CREATE TABLE Friendship (DateOfEvent DateTime,Status bit) AS EDGE; |

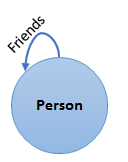
پس از اینکه با نحوه ایجاد این نوع جداول آشنا شدیم، جدول ارتباط دوستی بین افراد را ایجاد خواهیم کرد، که همانند کد خط بالا می باشد. برای این جدول Edge دو فیلد تاریخ رویداد و وضعیت رویداد ( به معنای دوستی یا قطع دوستی ) ایجاد کرده ایم.



همانطور که مشاهده می کنید جدول Edge ایجاد شد و در لیست جداول دیتابیس نمایش داده شده است.

نحوه درج ا

برای درج اطلاعات در Edge Table ها باید مشخص کنیم این یال چه نود هایی را به یکدیگر متصل می کند. مانند تصویر زیر

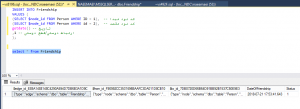


یال Friends بین دو موجودیت Person ارتباط برقرار می کند بنابراین در یک جدول Edge اطلاعات نود مبدا و نود مقصد علاوه بر اطلاعات دیگر ذخیره خواهد شد.

برای شروع کار من سعی می کنم Person شماره یک و Person شماره دو، ارتباط دوستی برقرار کنم که کد آن اینگونه خواهد بود:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | INSERT INTO Friendship  VALUES (  (SELECT $node\_id FROM Person WHERE ID = 1),  -- کد نود مبدا  (SELECT $node\_id FROM Person WHERE id = 2),  -- کد نود مقصد  getdate() -- تاریخ  ,1 -- ارتباط دوستی/قطع دوستی  ); |

بعد از آن یک کوئری روی جدول می زنیم تا ببینیم چه دیتایی در جدول ثبت شده است:



دیتایی که در نتیجه میبینیم آن چیزی نیست که ما می خواهیم. در جداول معمولی Sql Server با استفاده از عملگر Join می توانستیم بین دو جدول ارتباط برقرار کنیم و نتیجه را ببینیم اما برای نمایش اطلاعات دو جدول که با یکدیگر ارتباط دارند در گراف دیتابیس اینگونه نیست. برای ارتباط دو موجودیت در Graph Database باید از یالی که بین نود ها ارتباط برقرار می کند استفاده نماییم. در ادامه Syntax این کوئری را باهم خواهیم دید:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT  p1.id,p1.FirstName, p1.LastName  ,p2.id,p2.FirstName, p2.LastName,f.DateOfFriendShip,Status  FROM      Person p1,Person p2, Friendship f  WHERE      MATCH(p1-(f)->p2) |

نتیجه ی کوئری بالا اینگونه خواهد بود:

