SQL Server Büyük Tabloları Bölmek

Bir güzel SQL Server özelliği ile daha beraberiz. Veri tabanları işlevleri gereği milyonlarca hatta milyarlarca kayıt tutabiliyorlar. Bu kayıtlardan bazıları üzerinden zaman geçtikçe sorgulanmayan veriler olmaya başlıyor. Örneğin online alışveriş sitelerinde üç dört yıl önceki siparişler zorunlu bir işlem olmadığı sürece sorgulanmamaya başlar. Ama sorgularda bu kayıtlar da sorgulandığından sorgular yavaş çalışmaya başlar. Bu durumlar da SQL Server Partitioned Table özelliği ile büyük tabloları bir özelliğine göre farklı dosyalara ayırabiliriz. Böylece gelen sorguda bütün verileri değil de sadece ilgili verinin olduğu dosya üzerinde çalışılır. Saatlerce sürebilecek sorgular bu sayede saniyeler içerisinde bite bilmektedir.

Bölünmüş Tablo Oluşturma

SQL Server Management Studio veya T-SQL ile tablo bölünebilir.

Tablo bölme işlemi genel olarak dört adımda olmaktadır.

- Bölünecek tablonun verilerini tutacak FILEGROUPS ve dosyaların oluşturulması
- Bölünme kurallarını oluşturacak fonksiyonun oluşturulması

- Bölünmüş dosyaların tutulacağı schemanın oluşturulması
- Tablonun oluşturulan fonksiyon ile schema üzerinde bölünmesi

SQL Server Management Studio İle Bölme İşlemi

İlk işlem olarak ilgili veri tabanını sağ tıklayıp özellikler (Properties) ekranını açarak aşağıdaki ekran görüntülerine göre FILEGROUP ve File ekliyoruz.


FILEGROUP Ekleme

“Rows” kısmında “Add Filegroup” butonuna tıklayarak bir filegroup ekliyoruz. Filegroup ekledikten sonra bu filegroup içerisinde yer alacak file ekliyoruz.


FILE Ekleme

“Add” butonu ile bir önceki adımda oluşturduğumuz filegroup içerisinde yer alacak bir dosya ekliyoruz. Burada dikkat etmemiz gereken nokta dosyanın ndf uzantılı olması gerektiğidir.


Tablo Bölme İşlemi Başlatma

Filegroup ve file oluşturma işlemlerinden sonra tabloyu bölme işlemine başlıyoruz. Bölünecek tablo üzerine sağ tıklayarak Storage > Create Partition… yolunu izleyerek işleme başlıyoruz.


Üzerinde Bölme Yapılacak Kolon Seçimi

Açılan “Select a Partitioning Column” ekranında üzerinde bölme şartının çalışacağı kolonu seçiyoruz. Mantıksal olarak gruplanabilen her hangi bir kolon seçilebilir.


Partition Function

“Select a Partition Function” ekranında bölme kuralını içeren partition fonksiyonunu seçiyoruz. Daha önceden oluşturulan bir fonksiyon var ise “Existing partition function” seçeneği ile seçiyoruz. Oluşturulan fonksiyon yok ise “New partition function” seçeneği ile fonksiyon adını giriyoruz. Sistem girilen isimde fonksiyonu otomatik oluşturacaktır.


Şema Seçimi

“Select a Partition Scheme” ekranında da bölünme işlemini tutacak schema seçimi yapıyoruz.


Bölme Kurallarının Belirlenmesi

“Map Partition” ekranında verilerin hangi kurala göre hangi dosyalara bölüneceği kurallarını belirliyoruz. Tarih verisi içeren kolona göre bölme işlemi yapıyor isek “Set Boundaries…” butonuna tıklayarak aralıkları otomatik hesaplatabileceğimiz bir diyalog penceresi açabiliriz.


Set Boundaries…

“Set Boundaries…” ekranında başlangıç ve bitiş tarihlerini girdikten soran bölünmenin aralığını seçiyoruz. Örneğimizde verileri yılına göre böleceğimizi seçmişiz mesela.


Set Boundaries… işlemi sonrası

“Estimate storage” butonuna tıklayarak verilerin bölünme sonrası durumlarını ön izleyebilirz.


İşlem Başlatma

Son adım olarak işlemin ne zaman yapılacağı ile ilgili seçimi de yaptıktan sonra tablo bölme işlemi tamamlanmış olacaktır.

T-SQL İle Tablo Bölme

```
USE [AdventureWorks2017]
GO
BEGIN TRANSACTION
```

```
CREATE PARTITION FUNCTION [ByOrderDate](datetime) AS RANGE LEFT FOR VALUES (N'2011-05-31T00:00:00', N'2012-05-31T00:00:00', N'2013-05-31T00:00:00', N'2014-05-31T00:00:00', N'2015-05-31T00:00:00')

CREATE PARTITION SCHEME [Part2] AS PARTITION [ByOrderDate] TO ([SECONDARY], [SECONDARY], [SECONDARY], [SECONDARY], [SECONDARY], [SECONDARY])

ALTER TABLE [Sales].[SalesOrderDetail] DROP CONSTRAINT [PK_SalesOrderDetail_SalesOrderID_SalesOrderDetailID] WITH ( ONLINE = OFF )


ALTER TABLE [Sales].[SalesOrderDetail] ADD CONSTRAINT [PK_SalesOrderDetail_SalesOrderID_SalesOrderDetailID] PRIMARY KEY NONCLUSTERED
(
	[SalesOrderID] ASC,
	[SalesOrderDetailID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]


CREATE CLUSTERED INDEX [ClusteredIndex_on_Part2_6368150937690256l1] ON [Sales].[SalesOrderDetail]
(
	[ModifiedDate]
)WITH (SORT_IN_TEMPDB = OFF, DROP_EXISTING = OFF, ONLINE = OFF) ON [Part2]([ModifiedDate])


DROP INDEX [ClusteredIndex_on_Part2_6368150937690256l1] ON [Sales].[SalesOrderDetail]

COMMIT TRANSACTION
```