Veri Tabanı Yönetimi ve Modellemesi

HAFTA 9

Haftalık Ders Akışı

1. Veritabanı Kavramlarına Giriş
2. Veri Tabanı Türleri, İlişkisel Veri Tabanı Tasarımı
3. ER Diyagramları ve Normalizasyon
4. SQL Server Arayüzü, Veri Tabanı Nesneleri
5. T-SQL ve SQL Sorguları
6. İndeks ve View
7. Stored Procedure ve Fonksiyonlar
8. Ara Sınav
9. **Tetikleyiciler(Triggers) ve Yedekleme (Back-up)**
10. Kullanıcı Türleri ve Kullanıcı Yönetimi
11. No-SQL Veri Tabanları
12. Proje Sunumları (05.12.2019)
13. Proje Sunumları (12.12.2019)
14. Proje Sunumları (19.12.2019)

Triggers

- Bir olay meydana geldiğinde veritabanının gerçekleştirmesi gereken bir eylem/eylemler
- Trigger'lar
  - Bütünlük kısıtlamaları
  - Veri değişikliklerini denetlemek
  - Karmaşık kısıtlamaları yönetmek
- Trigger işlemi:
  - BEFORE | AFTER | INSTEAD OF
  - INSERT | DELETE | UPDATE

Genel Yapısı

CREATE TRIGGER TriggerName

BEFORE | AFTER | INSTEAD OF

INSERT | DELETE | UPDATE [OF TriggerColumnList**]**

ON TableName

[**REFERENCING {OLD | NEW} AS {OldName | NewName}**

[**FOR EACH {ROW | STATEMENT}]**

[**WHEN Condition]**

<trigger action>

Trigger Event-Condition-Action (ECA)

- E*vent (or events):*
  - *INSERT, UPDATE yada DELETE işlemlerinin yapılabildiği tablolar üzerinde kuralların tetiklenmesidir.*
  - Event before event yada after event şeklinde özelleşebilir.
- *Condition:*
  - *Yürütülmesi gerek action kısmının tanımlanmasıdır.*
  - *Condition opsiyonel olabilir, ancak koşullar doğru ise action çalıştırılabilir*
- *Action:*
  - *Trigger ifadesi verildiğinde ve doğru olarak çalıştırıldığında verilen T-SQL ifadesi çalıştırılabilir.*

Trigger

- Trigger iki seviyede olabilir
- Satır bazında (row level): Her bir satır etkilendiğinde işletilir
- İfade bazında (statement level): Birden fazla satır etkilendiğinde işletilir
- Ayrıca doğrudan (INSERT, UPDATE ve DELETE) ile değiştirilemeyen görünümleri değiştirmeyi sağlayan INSTEAD OF ifadesini triggerlar destekler
- INSTEAD OF triggerlar:
- Orjinal SQL ifadesi yerine başka bir trigger'ı tetikleyebilir

Örnek

İnsert ifadesinden sonra çalışan bir trigger yazılması

Trigger

```
CREATE TRIGGER addStafTrigger

ON tbl_Staff

AFTER  INSERT

NOT FOR REPLICATION

AS

SELECT 'trigger executed'
```

Örnek

Delete işleminden önce kontrol işlemi yapan bir trigger yazılması

```
CREATE Trigger deleteStaffTrigger
 ON  tbl_Staff instead of delete
As
Begin
   If Exists
   (select id from tbl_staff_animal where staff_id  in (Select id from deleted) )
   BEGIN
      RAISERROR('Once diger tablolar silinmeli',16,1)
 ROLLBACK
   END
END
```

Örnek

Tablo üzerinde değişiklik yapılmasını engelleyen bir trigger yazılması

```
CREATE TRIGGER tblSecurity
ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS BEGIN
PRINT 'Bu tablonun degistirilmesi admin tarafindan engellenmistir'
ROLLBACK
END
```

Trigger Aktif-Pasif Durumu

- DISABLE
  - Alter Table tbl_name DISABLE TRIGGER trigger_name
  - DISABLE Trigger ALL ON ALL SERVER;
  - ALTER TABLE dbo.tbl_Staff DISABLE trigger deleteStaffTrigger
- ENABLE
  - Alter Table tbl_name ENABLE TRIGGER trigger_name
  - ALTER TABLE dbo.tbl_Staff ENABLEtrigger deleteStaffTrigger
- DROP (Silme)
  - Drop trigger trigger_name

Avantaj ve Dezavantaj

- Kod tekrarının kaldırılması
  - Log tablolarına verilerin otomatik olarak eklenmesi
- Değişikliklerin tek seferde uygulanması
  - İki ayrı veritabanı arasındaki bağlantı
- Güvenlik
- Bütünlük: Veri bütünlüğünü sağlamak için kontrollerin yapılması triggerlar aracılığı ile sağlanabilir.
- İşlem Gücü
- Client-Server Mimarisine uygun

Avantaj ve Dezavantaj

- Server'ın iş yükünün artması
- Triggerların bir birini tetiklemesi ve bunun öngörülemez olması
- Zamanlama
- Taşınabilirliği az: Birçok server triggerlar için kendi standardını kullanır
  - Örneğin: MsSql – before işlemi yerine instead of kullanılıyor

Yedekleme

- Back-up
  - BACKUP DATABASE Zoo TO DISK='D:\Zoo.bak‘
  - RESTORE DATABASE Zoo FROM DISK='D:\Zoo.bak'
- Attach – Detach
  - Ldf-Mdf

Replication

- Farklı yerlerde üretilen verilerin bir biri ile bağlantılı şekilde çalışabilmesi
- MSSQL tarafından desteklenen türler:
  - Snapshot Replication: Belirli peryotlarla yedek server'a veri akışının sağlanması
  - Transactional Replication: Başlangıçta database'in bir snapshot'ı alınır. Sonrasında her transaction işleminden sonra yedek DB güncellenir
  - Merge Replication: Aynı anda iki DB üzerinde de değişiklik yapılmasını sağlar
  - Peer to Peer: Aynı anda birden fazla server'a eş zamanlı olarak değişiklik izni verir
  - Bidirectional: Her iki yönlü veri akışına izin verir.
  - Updatable Subscriptions: Bir server'ın yayınladığı veriye diğer serverlar abone olur. Yayın yapan server değişiklik yaptığı zaman, diğer server bu değişimi kendi sistemlerine uygular.

Mirroring

- DB server'ın her hangi bir problem karşısında çökmesi durumunda kullanılabilecek bir yöntem
- Bir veritabanının kullanılabilirliğini arttırır.
- Veri korumasını arttırır.
- Sistem güncellemeleri sırasında veritabanının kullanılabilirliğini iyileştirir