Veritabanı Programlama
Bölüm 4
CURSOR

Dr. Öğr. Üyesi Murat TAŞYÜREK (kayubmprogramlama1@gmail.com)

26 Ekim 2023

Kayseri Üniversitesi, Bilgisayar Mühendisliği Bölümü

Cursor

- Cursor, veri kümesinde bulunan verileri okuyarak her seferinde bir kayıt üzerinde işlem yapabilmeyi sağlayan yöntemdir.
- Cursor, varsayılan olarak ileri doğru işlem yapar.
- Cursor, geri giderek satırlar üzerinde işlem yapmayı sağlar.
- En hızlı Cursor, sadece ileri doğru okuma işlemi yapandır.
- Bu nedenle ileri doğru yönnde çalışan cursor'u kullanmanızı tavsiye ediyorum.

Cursor

- Veritabanında **SELECT** sorgusu ile elde edilen kayıtlar üzerinde, döngü yapısı oluşturarak tüm satırları tek tek inceleyebilmek, üzerinde işlem yapabilmek için **Cursor** kullanılır.
- **Cursor** veritabanında saklanmaz.
- **Cursor**'un kullamını bittinde hafızaya idaede edilmelidir.

Neden Cursor kullanılır

- Sorgu sonucu dönen kayıtları inceleyerek güncelleme yapılmak istenen kayıtlarıda güncelleme işlemi yapmak için
- Diğer kullanıcılar tarafından yapılan değişikliklerin görmek için
- Trigger ya da Stored Procedure içerisinde bir sorgu sonucuna satır satır erişmek için

T-SQL Cursor Tanımlama

- Cursor tanımmlanın genel yazım şekli aşağıdaki gibidir.

```
DECLARE cursor_name CURSOR [ LOCAL | GLOBAL ]
     [ FORWARD_ONLY | SCROLL ]
     [ STATIC | KEYSET | DYNAMIC | FAST_FORWARD ]
     [ READ_ONLY | SCROLL_LOCKS | OPTIMISTIC ]
     [ TYPE_WARNING ]
     FOR select_statement
     [ FOR UPDATE [ OF column_name [ ,...n ] ] ]
[;]
```

Cursor

- Cursor ve veri kümesi bildirim sırasında isimlendirilir, daha sonra bu isimler kullanılır.
- Cursor, kendi içerisinde açık ve kapalı olma durumuna sahiptir. Cursor kullanmak için önce açmanız gerekir, kendi kendine kapanmaz, işlem bitincede kapatmanız gerekir.
- Cursor, kendi kendine hafızadan silinmez.
- Kapatılan Cursor'un hafızadan silinmesi gerekir.

Cursor'un Ömrü

Cursor kullanmaya başlamadan önce yaşam döngünüsü (ömrünü) bilmek gerekir. Cursor'un yaşam döngüsü aşağıdaki gibidir.

- Bildirim (tanımlama)
- Açılış
- Kullanım
- Kapanış
- Hafızada ayrılan belleği boşaltmak

Örnek

- Geçen hafta oluşturduğumuz Ogrenciler tablosunda
- Ögrenci Numarası ve Öğrenci adlarını CURSOR vasıtasıyla alan,
- CURSOR üzerinde gezinirken öğrencinin soyadını sorgu yoluyla elde eden
- Öğrenci numarası çift ise Öğrenci adı ve soyadı arasında '–' ekleyerek ekrana yazdıran
- Öğrenci numarası tek ise Öğrenci adı ve soyadı arasında '__' ekleyerek ekrana yazdıran
- uygulamayı T-SQL'de kodlayınız.

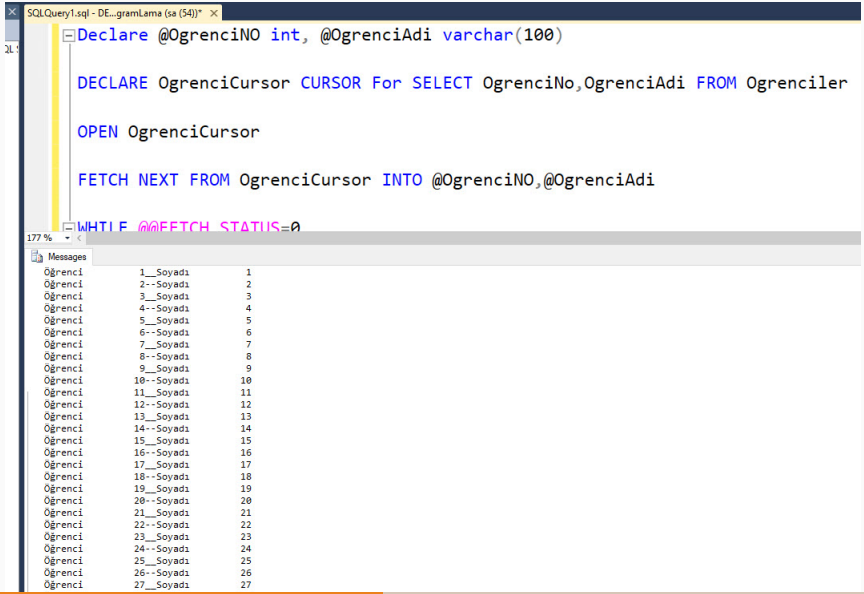
Cursor Kullanımı

- Cursor üstünde dolasmak için FETCH komutundan kullanılır.
- **FETCH** işlemi ile sıradaki satır ilgili sorgudan okunan değerler sırası ile into ifadesinden sonraki değişkenlere atanır.
- **fetch next from** {**cursor adı**} **into** {**değişkenler**}
- **FETCH** işleminin başarılı olup olmadı **@@FETCH_STATUS** ile kontrol edilir.
- **@@FETCH_STATUS=0**, FETCH komutu basari ile gerçeklestirlidi
  @@FETCH_STATUS=-1, FETCH komutunda bir hata ile karsilasildi
  @@FETCH_STATUS=-2, FETCH komutunda tüm kayitlar bittigi için en sona gelindi, daha fazla kayit yer almiyor

T-SQL Code

```
Declare @OgrenciNO int, @OgrenciAdi varchar(100)

DECLARE OgrenciCursor CURSOR For SELECT OgrenciNo,OgrenciAdi FROM Ogrenciler

OPEN OgrenciCursor

FETCH NEXT FROM OgrenciCursor INTO @OgrenciNO,@OgrenciAdi

WHILE @@FETCH_STATUS=0
begin
    DECLARE @OgrenciSoyadi varchar(50)
    SELECT @OgrenciSoyadi=OgrenciSoyadi FROM Ogrenciler WHERE OgrenciNo=@OgrenciNO
    IF @OgrenciNO%2=0
       set @OgrenciAdi=@OgrenciAdi+'--'+@OgrenciSoyadi
    ELSE
        set @OgrenciAdi=@OgrenciAdi+'__'+@OgrenciSoyadi
    print @OgrenciAdi

    FETCH NEXT FROM OgrenciCursor INTO @OgrenciNO,@OgrenciAdi
end

Close OgrenciCursor
DEALLOCATE OgrenciCursor
```

T-SQL Çıktı


SQLQuery1.sql - DE...gramLama (sa (54))*
Declare @OgrenciNO int, @OgrenciAdi varchar(100)
DECLARE OgrenciCursor CURSOR For SELECT OgrenciNo,OgrenciAdi FROM Ogrenciler
OPEN OgrenciCursor
FETCH NEXT FROM OgrenciCursor INTO @OgrenciNO,@OgrenciAdi
WHILE @@FETCH_STATUS=0
177 %
Messages
Ogrenci 1__Soyadı 1
Ogrenci 2--Soyadı 2
Ogrenci 3__Soyadı 3
Ogrenci 4--Soyadı 4
Ogrenci 5__Soyadı 5
Ogrenci 6--Soyadı 6
Ogrenci 7__Soyadı 7
Ogrenci 8--Soyadı 8
Ogrenci 9__Soyadı 9
Ogrenci 10--Soyadı 10
Ogrenci 11__Soyadı 11
Ogrenci 12--Soyadı 12
Ogrenci 13__Soyadı 13
Ogrenci 14--Soyadı 14
Ogrenci 15__Soyadı 15
Ogrenci 16--Soyadı 16
Ogrenci 17__Soyadı 17
Ogrenci 18--Soyadı 18
Ogrenci 19__Soyadı 19
Ogrenci 20--Soyadı 20
Ogrenci 21__Soyadı 21
Ogrenci 22--Soyadı 22
Ogrenci 23__Soyadı 23
Ogrenci 24--Soyadı 24
Ogrenci 25__Soyadı 25
Ogrenci 26--Soyadı 26
Ogrenci 27__Soyadı 27

Cursor Özellikleri - Scope

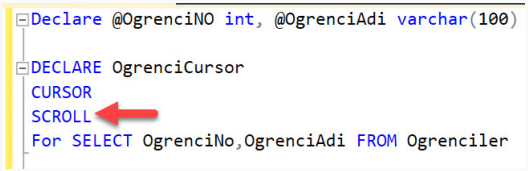
- **Scope**, Cursor'lerin görünebilirlik ayarını belirtir.
- **LOCAL** ve **GLOBAL** olarak ikiye ayrılır.
- Bir stored procedüre içerisinde oluşturulan Cursor **GLOBAL** scope'u içerisinde ise, bir başka stored procedür içerisinden bu Cursor'e **erişilebilir**.
- **GLOBAL** olarak tanımlanan bir Cursor ismi ile başka bir başka stored procedür içerisinde dahi olsa yeni bir Cursor **tanımlanamaz**.
- Cursor **LOCAL** scope içerisinde ise bir başka stored procedür içerisinden erişilemez.

Cursor Özellikleri - Scrollability

- **Kaydırılabilirlik (scrollability)** özelliği, SQL Server'da Cursor'lerin hareket yönünü ifade eder.
- T-SQL'de tanımlanan cursor varsayılan olarak sadece ileriye doğru hareket eder.
- Cursor'ler kaydırılabilirlik seçenekleri ile ileri ve geri hareket kabiliyeti kazandırılabilir.
- Ancak, ileri doğru hareket eden cursor'ler diğerlerine göre daha hızlı çalışır.
- Cursor default olarak **FORWARD_ONLY** ile tanımlanır. Bu tanımlamada sadece **FETCH NEXT** komutu çalışır.

Cursor Özellikleri - Scrollability

- **SCROLL** kelimesi ile Cursor'a Scrollability yeteneği kazandırlır.
- Bu sayede Cursor ileri geri hareket ettirmek edebilir.
- **SCROLL** ile aşağıdaki şekilde tanımlanabilir.



Cursor Özellikleri - Scrollability

- Kaydırma işleminin temel özelliği **FETCH** anahtar sözcüğüdür.
- **FETCH NEXT**: Bir sonraki kaydı getirir.
- **FETCH PRIOR**: Bir önceki kaydı getirir.
- **FETCH FIRST**: İlk kaydı getirir.
- **FETCH LAST**: Son kaydı getirir.
- **FETCH RELATIVE +-X**: Cursor'un bulunduğu satırdan X sayısı kadar ile ya da geri giderek ilgili kaydı getirir.

Static Cursor

- Kaydırma işleminin temel özelliği **FETCH** anahtar sözcüğüdür.
- **FETCH NEXT**: Bir sonraki kaydı getirir.
- **FETCH PRIOR**: Bir önceki kaydı getirir.
- **FETCH FIRST**: İlk kaydı getirir.
- **FETCH LAST**: Son kaydı getirir.
- **FETCH RELATIVE +-X**: Cursor'un bulunduğu satırdan X sayısı kadar ile ya da geri giderek ilgili kaydı getirir.

Cursor Türleri

- T-SQL'de Forward-Only, Static, Dynamic ve Keyset olmak üzere 4 adet cursor türü vardır.
- Forward-Only Cursors, verileri yalnızca bir kez geçerler ve genellikle bir sonraki satırın okunabilmesi için kilitlenirler.
- Forward-Only Cursors, verilerin bir kez okunmasına ihtiyaç duyulan bir senaryoda idealdir.
- Forward-Only Cursors, sonuç kümesindeki her satırı sırayla işlemek için tasarlanmıştır.

Forward-Only Cursor

- Forward-Only Cursorların ileri doğru hareket etmeleri, işleme hızlarını artırır ve bellek kullanımını minimize eder.
- Bu nedenle, Forward-Only Cursors, büyük veri kümeleriyle çalışırken performans sorunlarından kaçınmak için tercih edilir.
- Forward-Only Cursors, bir sonraki satırın okunabilmesi için kilitlenirler.
- Ancak, bir sonraki satırı okuyamazken bir kilit varsa, diğer işlemlerin beklemesi gerektiği için performans sorunlarına neden olabilir.
- Henüz okumadığı satırda yapılan değişiklikleri yakalar.

Static Cursor

- Static Cursors, verilerin kopyalarının bellekte tutulduğu ve verilerin güncelleştirilemediği bir cursor türüdür.
- Sorgu sonuç kümesindeki her satır, Cursor'un oluşturulduğu anda alınan bir anlık görüntüsünü temsil eder.
- Veriler değiştirildiğinde, kopyalanan veriler güncellenmez.
- Static Cursors, bir veri kopyasını bellekte sakladığından, sonuç kümesindeki verilere bir kez erişebilirsiniz. Bu nedenle, Static Cursors, sorgu sonucu küçük bir veri kümesini işlemek için idealdir ve büyük veri kümeleri ile çalışırken performans sorunlarına neden olabilir.
- Static Cursors, her satırın işlenmesi için veritabanına kilitlenmez ve sonuç kümesindeki satırların sırası ve değerleri her zaman aynı kalır (ilk çekilen değerler).

Dynamic Cursor

- Dynamic Cursors, tablodaki verilerin anlık görüntüsünü kullanarak çalışan ve sorgunun sonucuna bağlı olarak veri kümesindeki verileri dinamik olarak değiştirebilen bir cursor türüdür.
- Bu, veriler güncellendiğinde, bu değişikliklerin dinamik Cursorlar tarafından da otomatik olarak yakalanır.
- Dynamic Cursors, verilerin güncellenmesine ve veritabanındaki değişiklikleri işlemeye izin verir.
- Dynamic Cursors, büyük veri kümeleriyle çalışırken performans açısından dezavantajlıdır.
- Dynamic Cursors, sonuç kümesindeki verileri tekrar kullanmak istediğinizde kullanışlıdır.

Keyset Cursor

- Keyset Cursors, bir anahtar kümesi belirleyerek verileri sıralayan ve hareket eden bir cursor türüdür.
- Cursor'un konumu, anahtar kümesiyle belirlenir. Veriler güncellendiğinde, bu cursor türü değişikliklerden etkilenir.
- Keyset Cursors, büyük veri kümeleriyle çalışırken performans açısından avantaj sağlar. Keyset sayesinde, Cursor'un belirli bir konuma gitmesi daha hızlıdır ve sonuç kümesindeki verilerin bir kez okunmasıyla verileri işlemek mümkündür.
- Kullanıldığı tablo üzerinde unique index olması gerekmektedir.
- Keyset cursorlar oluşturuldukları anda bulunan satırlardaki değişikliklere duyarlıdırlar. Lakin oluşturulduktan sonra yeni eklenen satırlara karşı duyarlı değildirler.

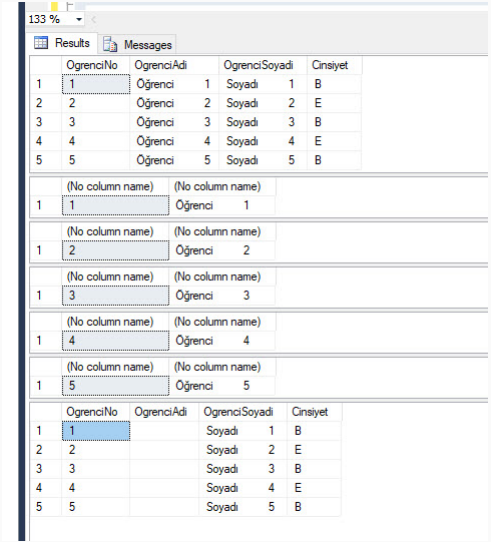
Statik Cursor Örnek

- Öğrenciler tablosunda bulunan ve öğrenci numarası 6'dan küçük olan kayıtların öğrenci numarası ve adını statik cursor kullanarak ele ediniz
- Cursor'u açtıktan sonra öğrenciler tablosunda öğrenci numarası 6'dan küçük olan kayıtların öğrenci adlarını null'a çekiniz.
- CURSOR üzerinde gezinerek her bir satırda öğrenci numarası ve öğrenci adını SELECT ile değerleri ekranda gösteriniz.
- CURSOR kapandıktan sonra öğrenci numarası 6'dan küçük olan kayıtları SELECT ile kontrol ediniz.
- uygulamayı T-SQL'de kodlayınız.

T-SQL Code

```
SELECT * FROM Ogrenciler where OgrenciNo <6

Declare @OgrenciNO int, @OgrenciAdi varchar(100)
DECLARE OgrenciCursor
CURSOR
STATIC
For SELECT OgrenciNo,OgrenciAdi FROM Ogrenciler  where OgrenciNo <6

OPEN OgrenciCursor

update Ogrenciler set OgrenciAdi='' where OgrenciNo <6

FETCH NEXT FROM OgrenciCursor INTO @OgrenciNO,@OgrenciAdi
WHILE @@FETCH_STATUS=0
begin

    SELECT @OgrenciNO, @OgrenciAdi

    FETCH NEXT FROM OgrenciCursor INTO @OgrenciNO,@OgrenciAdi
end

Close OgrenciCursor
DEALLOCATE OgrenciCursor

SELECT * FROM Ogrenciler where OgrenciNo <6
```

T-SQL Çıktı



133 %

Results | Messages

	OgrenciNo	OgrenciAdi	OgrenciSoyadi	Cinsiyet
1	1	Öğrenci 1	Soyadı 1	B
2	2	Öğrenci 2	Soyadı 2	E
3	3	Öğrenci 3	Soyadı 3	B
4	4	Öğrenci 4	Soyadı 4	E
5	5	Öğrenci 5	Soyadı 5	B

	(No column name)	(No column name)
1	1	Öğrenci 1

	(No column name)	(No column name)
1	2	Öğrenci 2

	(No column name)	(No column name)
1	3	Öğrenci 3

	(No column name)	(No column name)
1	4	Öğrenci 4

	(No column name)	(No column name)
1	5	Öğrenci 5

	OgrenciNo	OgrenciAdi	OgrenciSoyadi	Cinsiyet
1	1		Soyadı 1	B
2	2		Soyadı 2	E
3	3		Soyadı 3	B
4	4		Soyadı 4	E
5	5		Soyadı 5	B