شروع کار با SQL Server

درس ششم: تریگرها

سید کاوه احمدی

تریگرها (Triggers)

- Trigger یک روال ذخیره شده است که در پاسخ به بروز یکی از دستورات DML رخ می‌دهد.

ساختار Trigger

```
CREATE TRIGGER trigger_name
ON table_name
{FOR | AFTER | INSTEAD OF}
{[INSERT][,][DELETE][,][UPDATE]}
AS
  BEGIN
     …
  END
```

یک بلوک از دستورات در T-SQL

مثال

- فرض کنید می‌خواهیم به جای حذف فیریکی رکوردها در جدول `Categories`، آنها را به شکل منطقی حذف کنیم.
- برای این منظور یک فیلد با نام `IsOmit` به جدول اضافه می‌کنیم.
- با توجه به اینکه دیگر در این جدول حذف فیزیکی رکوردها صورت نمی‌پذیرد، لذا فرآیند آبشاری تعریف شده هنگام حذف رکوردها (`null` شدن دسته‌ی اخبار مرتبط با رکورد حذف شده) نیز دیگر اجرا نمی‌شود.
  - در این شرایط اخباری خواهیم داشت که دسته‌ی مربوط به آنها حذف (منطقی) شده‌اند!

مثال

- Trigger زیر پس از ویرایش شدن رکوردهای جدول Categories اجرا می‌شود و تمامی رکوردهای جدول News که کلید اصلی CategoryId آنها در جدول Categories حذف منطقی شده است را null می‌کند.

مثال

```
CREATE TRIGGER
        Update_News
ON
        Categories
AFTER
        UPDATE
AS
        BEGIN
                UPDATE
                        News
                SET
                        CategoryId = NULL
                WHERE
                        CategoryId IS NOT NULL AND
                        CategoryId IN (
                                SELECT
                                        Id
                                FROM
                                        Categories
                                WHERE
                                        IsOmit = 1
                        )
        END
```

مثال

ویرایش یک تریگر براساس نام

در صورتی که در رکوردی از جدول، فیلد `IsOmit` ویرایش شده باشد، مقدار `true` باز می‌گرداند

```
ALTER TRIGGER
        Update_News
ON
        Categories
AFTER
        UPDATE
AS
        BEGIN
                IF UPDATE(IsOmit)
                        UPDATE
                                News
                        SET
                                CategoryId = NULL
                        WHERE
                                CategoryId IS NOT NULL AND
                                CategoryId IN (
                                        SELECT
                                                Id
                                        FROM
                                                Categories
                                        WHERE
                                                IsOmit = 1
                                )
        END
```

جدول‌های inserted و deleted

- آخرین تغییرات انجام شده روی جداول دارای `trigger` در دو جدول `inserted` و `deleted` نگهداری می‌شود.
- ساختار این جدول‌ها دقیقا همسان با ساختار جداول اصلی است.

جدول‌های inserted و deleted

- در triggerهای دستور delete فقط از جدول deleted می‌توان استفاده کرد.
- در triggerهای دستور insert فقط از جدول inserted می‌توان استفاده کرد.
- در triggerهای دستور update از هردو جدول inserted و deleted می‌توان استفاده کرد.
  - رکوردها با مقادیر قبلی در جدول deleted و رکوردها با مقادیر جدید در جدول inserted قرار می‌گیرد.

نکته

- عنوان، متن و تعداد نظرات مربوط به یک خبر را می‌خواهیم:

```
SELECT
        N.Title, N.Body, COUNT(C.Id)
FROM
        News N
INNER JOIN
        Comments C
ON
        N.Id = C.NewsId
GROUP BY
        N.Title, N.Body
```

نکته

- بنابراین مجبوریم از پرسش فرعی استفاده کنیم:

```
SELECT
        N.Title,
        N.Body,
        (
                SELECT
                        COUNT(*)
                FROM
                        Comments C
                WHERE
                        N.Id = C.NewsId
        )
FROM
        News N
```

نکته

- می‌توان تعداد نظرات هر خبر را در جدول `News` ذخیره کرد تا هنگام واکشی داده‌ها نیازی به شمارش تعداد نظرات نباشد.
  - برای این منظور فیلد `CommentCount` را برای نگهداری تعداد نظرات مربوط به یک خبر به جدول `News` اضافه می‌کنیم.

نکته

- برای حفظ سازگاری پایگاه داده، هنگام درج یا حذف نظرات، فیلد `CommentCount` خبر باید به بروز رسانی شود:

```
INSERT INTO Comments(Body, NewsId)
VALUES('……', 25);

UPDATE news
SET CommentCount = CommentCount + 1
WHERE ID = 25;
```

- استفاده از `Trigger`ها مطمئن‌ترین راه برای رسیدن به این هدف است!

مثال

- اصلاح تعداد نظرات یک خبر (فیلد `CommentCount`) هنگام درج یا حذف یک نظر با تریگر.

مثال

- اجرا زمان درج

```
CREATE TRIGGER update_commentsnum_afterinsert
ON Comments
AFTER INSERT
AS
        BEGIN
                UPDATE
                        News
                SET
                        CommentCount = CommentCount + 1
                WHERE
                        ID = (SELECT NewsID FROM inserted)
        END
```

در صورتی که به طور همزمان بیش از یک کامنت در جدول درج شود (جدول inserted بیش از یک رکورد داشته باشد)، پرسجو با خطا مواجه می‌شود.

مثال

- اجرا زمان درج

```
CREATE TRIGGER update_commentsnum_afterinsert
ON Comments
AFTER INSERT
AS
        BEGIN
                UPDATE
                        News
                SET
                        CommentCount = CommentCount + 1
                WHERE
                        ID IN (SELECT NewsID FROM inserted)
END
```

مشکل پرسجوی قبل حل می‌شود اما در صورتی که به طور همزمان بیش از یک نظر برای یک خبر درج شود، فیلد CommentCount خبر مربوطه فقط یک واحد اضافه می‌شود.

مثال

- اجرا زمان درج

```
CREATE TRIGGER update_commentsnum_afterinsert
ON Comments
AFTER INSERT
AS
  BEGIN
    UPDATE
      News
    SET
      CommentCount = CommentCount + (
        SELECT COUNT(*)
        FROM inserted I
        WHERE I.NewsId = News.Id
      )
    WHERE
      ID IN (SELECT NewsID FROM inserted)
  END
```

مثال

- اجرا زمان حذف

```
CREATE TRIGGER update_commentsnum_afterdelete
ON Comments
AFTER DELETE
AS
  BEGIN
    UPDATE
      News
    SET
      CommentCount = CommentCount - (
        SELECT COUNT(*)
        FROM deleted D
        WHERE D.NewsId = News.Id
      )
    WHERE
      ID IN (SELECT NewsID FROM deleted)
  END
```

مثال

- اجرا در یک تریگر

```
CREATE TRIGGER update_commentsnum
ON Comments AFTER DELETE, INSERT
AS
BEGIN
      UPDATE
        News
      SET
        CommentCount = CommentCount + (
          SELECT COUNT(*)
          FROM inserted I
          WHERE I.NewsId = News.Id
        )
      WHERE
        ID IN (SELECT NewsID FROM inserted)

      UPDATE
        News
      SET
        CommentCount = CommentCount - (
          SELECT COUNT(*)
          FROM deleted D
          WHERE D.NewsId = News.Id
        )
      WHERE
        ID IN (SELECT NewsID FROM deleted)
END
```

نکته

- کلمه کلیدی AFTER یا FOR باعث می‌شود trigger پس از اعمال تمام دستورات SQL اصلی و تغییرات آبشاری مرتبط با آن اجرا شود.
- T-SQL روشی برای اجرای یک trigger پیش از اجرای دستور اصلی ندارد.
- اما:
  - جدول دارای trigger به صورت ضمنی تشکیل یک تراکنش را می‌دهد.
    - ابتدا پرسش اصلی اجرا می‌شود – سپس تغییرات آبشاری مرتبط – سپس دستورات trigger – در نهایت تراکنش commit می‌شود.

مثال

- تریگر زیر باعث می‌شود از سال ۲۰۱۵ و بعد از آن هیچ خبری به جدول News اضافه نشود!

```
CREATE TRIGGER check_NewsInsertDate
On News AFTER INSERT
AS
	BEGIN
		IF (YEAR(GETDATE()) >= 2015)
			BEGIN
				PRINT('Illegal Insert Date')
				ROLLBACK TRANSACTION
			END
	END
```

مثال

- جدول `authors` را در نظر بگیرید. می‌خواهیم تعداد دفعاتی که کاربر، نام کاربری خود را تغییر می‌دهد ذخیره کنیم (در فیلدی به نام `username_change`).
  - ممکن است بروزرسانی روی سایر فیلدها باشد

مثال

```
CREATE TRIGGER last_update
ON authors AFTER UPDATE
AS
  BEGIN
    DECLARE @old varchar(20), @new varchar(20)

    SELECT
      @old = d.str_user, @new = i.str_user
    FROM
      inserted i, deleted d

    if @new <> @old
      UPDATE
        authors
      SET
        username_change = username_change + 1
      WHERE
        int_id IN (
          SELECT int_id from inserted
        )
  END
```

فقط روی یک رکورد درست کار می‌کند!

مثال

- اصلاح تریگر قبل

```
CREATE TRIGGER
  update_count
ON
  Authors
AFTER
  UPDATE
AS
  BEGIN
    UPDATE
      a
    SET
      UsernameChange = a.UsernameChange + 1
    FROM
      Authors a
    INNER JOIN
      inserted i
    ON
      a.Id = i.Id
    INNER JOIN
      deleted d
    ON
      a.Id = d.Id
    WHERE
      i.UserName != d.UserName
  END
```

INSTEAD OF

- به جای پرسش اصلی اجرا می‌شود.
- مقادیر جدید و قبلی در جداول inserted و deleted قابل دسترس هستند.

INSTEAD OF

- تریگر زیر موجب می‌شود یک نام کاربری بیش از پنج بار تغییر داده نشود.

```
CREATE TRIGGER
  dont_update
ON
  authors
INSTEAD OF
  UPDATE
AS
  BEGIN
    UPDATE
      authors
    SET
      users.str_user = i.str_user
    FROM
      authors
    INNER JOIN
      inserted
    ON
      authors.int_id = inserted.int_id
    WHERE
      authors.user_change < 5
  END
```

مثال

- با استفاده از پرسجوی زیر، به جای حذف رکورد از پایگاه داده (هنگام اجرای دستور DELETE)، فیلد `IsOmit` رکورد را `true` می‌کنیم.
  - در این حالت در صورت حذف اشتباه، بازیابی رکورد ساده‌تر است و سوابق اطلاعات درج شده در پایگاه داده از بین نمی‌رود.
  - مشکل زمانی است که بخواهیم رکوردها را از پایگاه داده بخوانیم: ممکن است برنامه‌نویس فراموش کند شرط `IsOmit = 0` را اعمال کند و رکوردهای حذف شده نیز بازگردانده شود.
    - با استفاده از یک `View` می‌توان این مسئله را حل کرد.

مثال

```
CREATE TRIGGER OmitNews
ON  Categories
INSTEAD OF DELETE
AS
        BEGIN
                UPDATE
                        Categories
                SET
                        IsOmit = 1
                WHERE
                        Id IN (
                                SELECT
                                        Id
                                FROM
                                        DELETED
                )
END
```

[WITH ENCRYPTION]

- باعث می‌شود که محتوای تریگر دیگر قابل نمایش و تغییر نباشد.

```
ALTER TRIGGER OmitNews
ON  Categories
WITH ENCRYPTION
INSTEAD OF DELETE
AS
        BEGIN
                UPDATE
                        Categories
                SET
                        IsOmit = 1
                WHERE
                        Id IN (
                                SELECT
                                        Id
                                FROM
                                        DELETED
                )
END
```