Putra Pandu Adikara, S.Kom

Trigger

Basis Data 2

Konsep Trigger

Trigger

- Trigger SQL adalah pernyataan SQL atau satu set pernyataan SQL yang disimpan dalam database dan harus diaktifkan atau dijalankan ketika suatu event terjadi pada suatu tabel database.
- Event ini berupa DML (**INSERT, UPDATE** dan **DELETE**), **DDL (CREATE, DROP, ALTER)** serta operasi database.
- Sebagai contoh perlu beberapa *business logic* yang harus dilakukan sebelum atau setelah memasukkan record baru dalam tabel database.
- Misal:
  - ketika ada record yang ditambahkan atau dihapus, maka serta merta akan dikirimkan email ke suatu alamat

Trigger vs Stored Procedure

- Terkadang Trigger disebut sebagai bentuk khusus dari stored procedure.

- Perbedaan antara Trigger dan Stored Procedure:
  - Trigger diaktifkan atau dipanggil secara implisit saat sebuah event yang terjadi di tabel database
  - Stored Procedure harus dipanggil secara eksplisit.

Kelebihan Trigger

- Trigger menyediakan cara alternatif untuk memeriksa integritas.
- Trigger bisa menangkap kesalahan dalam *business logic* pada tingkat database.
- Trigger menyediakan cara alternatif untuk menjalankan tugas-tugas yang dijadwalkan.
  - tidak harus menunggu untuk menjalankan tugas-tugas yang dijadwalkan.
  - dapat menangani tugas-tugas sebelum atau setelah perubahan yang dibuat untuk tabel database.
- Trigger sangat berguna untuk mengaudit perubahan data dalam tabel database.

Kelemahan Trigger

- Trigger hanya bisa menyediakan validasi tambahan tapi tidak dapat menggantikan semua validasi.
- Beberapa validasi sederhana dapat dilakukan di level aplikasi.
  - Sebagai contoh, kita dapat memvalidasi inputan di sisi klien menggunakan Javascript atau di sisi server dengan menggunakan script PHP atau ASP.NET.
- Trigger mengeksekusi secara tak terlihat dari klien-aplikasi yang terhubung ke database server sehingga sulit untuk mencari tahu apa yang terjadi di level database.
- Trigger berjalan setiap update yang dibuat ke table sehingga menambah beban kerja ke database dan menyebabkan sistem berjalan lebih lambat.

Manfaat Penggunaan Trigger

- Meng-*generate* nilai kolom turunan (*derived column value)*
- Mencegah transaksi yang tidak valid
- Mengerjakan otorisasi keamanan yg kompleks
- Mengerjakan aturan bisnis (*business rule*) yang kompleks
- Menyediakan pencatatan event (event logging) scr transparan
- Menyediakan audit

Manfaat Penggunaan Trigger

- Mengerjakan *referential integrity* ke seluruh node dalam sebuah basis data terdistribusi
- Menjaga replikasi tabel scr synchronous
- Mengumpulkan statistik dari pengaksesan tabel
- Modifikasi data tabel ketika DML dijalankan pada View
- Mempublikasikan informasi ketika ada database event, user event, dan pernyataan SQL untuk suatu aplikasi (yang berlangganan/subcsribe)

Macam-Macam Trigger

- Row Trigger dan Statement Trigger
- BEFORE dan FOR/AFTER Trigger
- INSTEAD-OF Trigger
- Triggers on System Events and User Events

- Catatan: ada perbedaan antara macam Trigger, seperti BEFORE di Oracle bukan untuk SQL Server, row level, column level, for each row, for each statement, dll. Baca lebih lanjut mengenai ini!

Row Trigger dan Statement Trigger

Row Trigger

- Row Trigger dipanggil melalui statement Trigger tiap kali ada perubahan pada tabel.
  - Sebagai contoh, jika sebuah pernyataan UPDATE memutakhirkan beberapa baris tabel, sebuah Row Trigger dipanggil sekali untuk tiap baris yg dipengaruhi oleh pernyataan UPDATE. Jika Trigger tidak mempengaruhi suatu baris, Row Trigger tidak berjalan.
- Row trigger berguna bila kode aksi dalam Trigger bergantung pada data dari baris-baris yang diubah oleh Trigger

Row Trigger dan Statement Trigger

Statement Trigger

- Sebuah Statement Trigger dipanggil sekali tanpa memperhatikan jumlah jumlah baris dalam tabel yang diubah walaupun tidak ada baris yang terpengaruhi.
  - Misalnya, jika pernyataan DELETE menghapus beberapa baris dari tabel, Triger dipanggil hanya sekali.
- Statement Trigger berguna jika kode di Trigger tidak tergantung pada data yang disediakan oleh pernyataan trigger atau baris yg dipengaruhi.
- Sebagai contoh, penggunaan Statement Trigger:
  - Untuk pengecekan keamanan yang kompleks pada waktu atau user saat ini
  - Menghasilkan satu record data audit

BEFORE dan FOR/AFTER Triggers

- Waktu pemanggilan trigger ada dua, **BEFORE** dan **FOR/AFTER**
- **BEFORE** dan **FOR/AFTER** hanya bisa didefinisikan **untuk tabel bukan ada view**

- Catatan:
  - SQL Server tidak mengenal BEFORE berbeda dengan Oracle
  - SQL Server hanya mengenal FOR/AFTER, INSTEAD OF

BEFORE dan FOR/AFTER Triggers

- **BEFORE Trigger** menjalankan aksi sebelum pernyataan hasil trigger dijalankan. Tipe ini umumnyadigunakan dalam situasi berikut:
  - Aksi dari trigger menentukan apakah pernyataan triger harus diteruskan untuk diselesaikan.
  - Untuk memperoleh nilai kolom tertentu sebelum menyelesaikan pernyataan INSERT atau UPDATE.
- **FOR/AFTER Trigger** menjalankan aksi trigger setelah pernyataannya dijalankan

BEFORE dan FOR/AFTER Triggers

- Ketika DML Trigger dijalankan untuk **INSERT** maka akan terbentuk tabel **INSERTED** yang berisi baris dari record baru yang ditambahkan
- Ketika DML Trigger dijalankan untuk **DELETE** maka akan terbentuk tabel **DELETED** yang berisi baris dari record baru yang dihapus

INSTEAD OF Trigger

- **INSTEAD OF** Trigger hanya dipanggil tapi tidak dieksekusi, didefinisikan khususnya untuk view
- **INSTEAD OF** Trigger mempopulasi dua tabel **INSERTED dan DELETED,** sehingga layaknya simulasi INSERT, UPDATE, atau DELETE
- **INSTEAD OF** Trigger menyediakan cara transparan untuk memodifikasi view yang tidak dapat dimodifikasi secara langsung melalui DML.
  - Misal ketika INSERT maka hanya akan menambahkan baris pada INSERTED, tapi tidak benar-benar menambah baris baru di tabel bersangkutan
  - Penambahan baris dilakukan secara eksplisit dari dalam trigger

Triggers on System Events and User Events

- Trigger dapat digunakan untuk mempublikasikan informasi tentang database event ke subscriber. Aplikasi dapat berlangganan (*subscribe*) seperti halnya *message* dari aplikasi lain.
- Database event antara lain (ada beda dukungan antara Oracle & SQL Server):
  - System Event:
    - Database startup dan shutdown
    - Server error message event
  - User Event
    - Logon and Logoff
    - DDL statement
    - DML Statement

Event pada Trigger

- Trigger dapat dipanggil ketika ada event sebagai berikut:
  - DML statements (DELETE, INSERT, UPDATE)
  - DDL statements (CREATE, ALTER, DROP, GRANT, REVOKE, dll)
  - Operasi database (SERVERERROR, LOGON, LOGOFF, STARTUP, SHUTDOWN, dll)
    - Beda dukungan untuk SQL Server atau Oracle

Deklarasi Trigger
DML Trigger

Membuat Trigger

- Syntax:

```
CREATE TRIGGER [ schema_name . ]trigger_name
ON { table | view }
[ WITH <dml_trigger_option> [ ,...n ] ]
{ FOR | AFTER | INSTEAD OF }
{ [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE
] }
[ WITH APPEND ]
[ NOT FOR REPLICATION ]
AS { sql_statement }
```

Contoh: Membuat Trigger 1

- Contoh Trigger untuk Insert:

```
USE AdventureWorks2008R2;
GO
CREATE TRIGGER NewPODetail
ON Purchasing.PurchaseOrderDetail
AFTER INSERT
AS
   UPDATE PurchaseOrderHeader
   SET SubTotal = SubTotal + LineTotal
   FROM inserted
   WHERE PurchaseOrderHeader.PurchaseOrderID =
inserted.PurchaseOrderID ;
```

Contoh: Membuat Trigger 2

- Contoh Trigger untuk delete:

```
USE AdventureWorks2008R2;
GO
CREATE TRIGGER ShowPODDelete
ON Purchasing.PurchaseOrderDetail
AFTER DELETE
AS
   SELECT * FROM deleted
```

Contoh: Membuat Trigger 3

```
USE AdventureWorks
GO

CREATE TRIGGER INSTEADOF_TR_I_EmpQualification
ON vw_EmpQualification
INSTEAD OF INSERT AS
BEGIN

DECLARE @Code TINYINT
SELECT @Code = qualificationCode FROM lib_Qualification L
INNER JOIN INSERTED I ON L.qualification = I.qualification

IF (@code is NULL )
BEGIN
RAISERROR (N'The provided qualification does not exist in qualification library',
16, 1)
RETURN
END

INSERT INTO employees (empcode, name, designation,qualificationCode,deleted)
SELECT empcode, name, designation, @code, 0
FROM inserted
END
```

Hapus Trigger

- Untuk menghapus trigger gunakan syntax

```
DROP TRIGGER [nama trigger]
```

Perubahan Trigger

- Trigger tidak dapat diubah melalui ALTER.
- Untuk mengubah harus menggunakan DROP TRIGGER kemudian CREATE TRIGGER ulang.

Deklarasi Trigger
DDL Trigger

- Syntax

```
CREATE TRIGGER trigger_name
ON { ALL SERVER | DATABASE }
[ WITH <ddl_trigger_option> [ ,...n ] ]
{ FOR | AFTER } { event_type | event_group } [
,...n ]
AS { sql_statement }
```

Contoh: Database Scope DDL Trigger

```
USE AdventureWorks2008R2;
GO

CREATE TRIGGER safety
ON DATABASE
FOR DROP_SYNONYM
AS
   RAISERROR ('You must disable Trigger "safety"
to drop   synonyms!',10, 1)
   ROLLBACK
GO
DROP TRIGGER safety
ON DATABASE;
GO
```

Contoh: Server scope DDL Trigger

```
CREATE TRIGGER ddl_trig_database
ON ALL SERVER
FOR CREATE_DATABASE
AS
    PRINT 'Database Created.'
    SELECT
EVENTDATA().value('(/EVENT_INSTANCE/TSQLComman
d/CommandText)[1]','nvarchar(max)')
GO

--DROP TRIGGER ddl_trig_database
--ON ALL SERVER;
--GO
```


UNIVERSITAS BRAWIJAYA