БАЗЫ ДАННЫХ

Лекция 15 Триггеры

Триггер

- Триггер - специальный вид хранимых процедур, выполняющихся при событиях базы данных

Триггер

- Имя
- Действие
- Исполнение

Имя триггера - максимум 128 символов

Триггер

- DML-триггеры
- DDL-триггеры

DML-триггеры

- Создаются для таблицы или представления
- Реагируют на события INSERT, DELETE, UPDATE

Триггер

- CREATE TRIGGER
- [schema_name.] trigger_name
- ON { table_name | view_name }
- [WITH dml_trigger_option [,...]]
- { FOR | AFTER | INSTEAD OF }
- { [NSERT] [,] [UPDATE] [,] [DELETE] }
- [WITH APPEND]
- { AS sql_statement |
- EXTERNAL NAME method_name }

DML-триггеры

```
create trigger AUD_AFTER_INSERT
on AUDITORIUM after INSERT
as
 print 'AUD_AFTER_INSERT';
 return;
go

create trigger AUD_AFTER_DELETE
on AUDITORIUM after DELETE
as
 print 'AUD_AFTER_DELETE';
 return;
go

create trigger AUD_AFTER_UPDATE
on AUDITORIUM after UPDATE
as
 print 'AUD_AFTER_UDATE';
 return;
go
```

Таблицы deleted и inserted

Две специальные виртуальные таблицы

- deleted — содержит копии строк, удаленных из таблицы
- inserted — содержит копии строк, вставленных в таблицу

- Структура этих таблиц эквивалентна структуре таблицы, для которой определен триггер

DML-триггеры

```
use BSTU
go
delete AUDITORIUM  where  AUDITORIUM in ('301-4','401-4');

go
create trigger AUD_AFTER
on AUDITORIUM after INSERT, DELETE, UPDATE
as
 declare @ins int = (select count(*) from INSERTED),
         @del int = (select count(*) from DELETED);

 if        @ins > 0 and  @del = 0  print 'Событие: INSERT';
 else  if @ins = 0 and  @del > 0  print 'Событие: DELETE';
 else  if @ins > 0 and  @del > 0  print 'Событие: UPDATE';
 return;
go
```

DML-триггеры

```
alter trigger AUD_AFTER_INSERT
on AUDITORIUM after INSERT
as
 select 'insert:INSERTED', *  from INSERTED
 union
 select 'insert:DELETED',  *  from DELETED;
 return;
go
```

DML-триггеры

```
alter trigger AUD_AFTER_UPDATE
on AUDITORIUM after UPDATE
as
 select 'update:INSERTED', *   from INSERTED
 union
 select 'update:DELETED',  *   from DELETED;
 return;
go
```

DML-триггеры

```
alter trigger AUD_AFTER_DELETE
on AUDITORIUM after DELETE
as
 select 'delete:INSERTED', *   from INSERTED
 union
 select 'delete:DELETED',  *   from DELETED;
 return;
go
```

```
insert  AUDITORIUM  values ('301-4', 'ЛК', 75, '301-4'),
                           ('401-4', 'ЛК', 80, '301-4');
update AUDITORIUM  set AUDITORIUM_CAPACITY += 5
where  AUDITORIUM in ('301-4','401-4');
delete AUDITORIUM  where  AUDITORIUM in ('301-4','401-4');
```

(No column name)	AUDITORIUM	AUDITORIUM_TYPE	AUDITORIUM_CAPACITY	AUDITORIUM_NAME
insert:INSERTED	401-4	ЛК	80	301-4
insert:INSERTED	301-4	ЛК	75	301-4

(No column name)	AUDITORIUM	AUDITORIUM_TYPE	AUDITORIUM_CAPACITY	AUDITORIUM_NAME
update:INSERTED	401-4	ЛК	85	301-4
update:INSERTED	301-4	ЛК	80	301-4
update:DELETED	401-4	ЛК	80	301-4
update:DELETED	301-4	ЛК	75	301-4

(No column name)	AUDITORIUM	AUDITORIUM_TYPE	AUDITORIUM_CAPACITY	AUDITORIUM_NAME
delete:DELETED	401-4	ЛК	85	301-4
delete:DELETED	301-4	ЛК	80	301-4

DML-триггеры

- AFTER-триггеры
- INSTEAD OF-триггеры

AFTER-триггеры

- Триггеры AFTER можно создавать только для базовых таблиц
- Можно использовать для:
  - создания журнала аудита действий в таблицах базы данных
  - реализации бизнес-логики
  - принудительного обеспечения ссылочной целостности

```
CREATE TRIGGER On_View_STUDENT
    ON dbo.all_
    AS   DECLAR
     IF (@C >100)
     BEGIN
```

Не удалось создать триггер "On_View_STUDENT" для dbo.all_stdents". Для представлений допустимы только триггеры INSTEAD OF.

AFTER-триггеры

- AFTER-триггеры - триггеры уровня оператора
- Выполняются по одному разу для каждого оператора
- Выполняются после наступления события

AFTER-триггеры

- AFTER-триггер вызывается после выполнения активизирующего его оператора
- Если оператор нарушает ограничение целостности, то возникшая ошибка не допускает выполнения этого оператора и соответствующих триггеров

AFTER-триггеры

```
alter table AUDITORIUM
add constraint CH_AUDITORIUM  check(AUDITORIUM_CAPACITY >=15)
go
update AUDITORIUM set AUDITORIUM_CAPACITY = 10 where AUDITORIUM = '206-1';
```

```
Msg 547, Level 16, State 0, Line 1
The UPDATE statement conflicted with the CHECK constraint "CH_AUDITORIUM".
The conflict occurred in database "BSTU",
table "dbo.AUDITORIUM", column 'AUDITORIUM_CAPACITY'.
The statement has been terminated.
```

AFTER-триггеры - транзакция

```
alter trigger AUD_AFTER
on AUDITORIUM after  DELETE, UPDATE
as
 declare @c int = (select sum(AUDITORIUM_CAPACITY) from AUDITORIUM);
 if (ISNULL(@c,0) < 1900)
 begin
  raiserror('Общая вместимость аудиторий не может быть < 1900',10,1);
  rollback;
 end;
 return;          ;
```

AFTER-триггеры - транзакция

```
use BSTU
go
delete AUDITORIUM where AUDITORIUM_TYPE = 'ЛБ-К';
```

```
Общая вместимость аудиторий не может быть < 1900
Msg 3609, Level 16, State 1, Line 1
The transaction ended in the trigger. The batch has been aborted.
```

INSTEAD OF-триггеры

- Триггеры уровня оператора
- Выполняются по одному разу для каждого оператора
- Выполняются вместо операции - сама операция не выполняется

INSTEAD OF-триггеры

- Всегда использует таблицы inserted и deleted
- Выполняется после создания таблиц inserted и deleted
- Выполняется перед выполнением проверки ограничений целостности или каких-либо других действий
- INSTEAD OF можно создавать для таблиц и для представлений - выполняется вместо выполнения любых действий с любой таблицей

INSTEAD OF-триггеры

```
use BSTU
go
create trigger AUDTYPE_INSTED
on AUDITORIUM_TYPE instead of INSERT, DELETE, UPDATE
as
  raiserror (N'изменение данных запрещено!!!', 10, 1);
  return;
```

INSTEAD OF-триггеры

```
use BSTU
go
insert AUDITORIUM_TYPE values('ЛК', 'XXX');
update AUDITORIUM_TYPE  set AUDITORIUM_TYPENAME = 'YYY'
       where AUDITORIUM_TYPE = 'ЛК';
delete  AUDITORIUM_TYPE  where AUDITORIUM_TYPE = 'ЛК';
```

```
изменение данных запрещено!!!

изменение данных запрещено!!!

изменение данных запрещено!!!
```

INSTEAD OF-триггеры

- Не могут вызываться рекурсивно (если в триггере сработает операция, снова вызвавшая работу триггера)
- Если образуется рекурсия вызовов триггеров, то будет сделана попытка выполнить оператор

Право на создание триггера

- Владелец базы данных
- Администратор DDL
- Владелец таблицы, для которой определяется триггер

- Это разрешение не может передаваться

Порядок DML-триггеров

- Можно указать порядок выполнения для нескольких триггеров
- sp_settriggerorder () имеет параметр @order:
  - first — указывает, что триггер является первым триггером AFTER, выполняющимся для действия
  - last — указывает, что триггер является последним триггером AFTER, выполняющимся для действия
  - none — отсутствует порядок выполнения (чтобы выполнить сброс)
- Остальные триггеры выполняются в неопределенном порядке

Порядок DML-триггеров

```
use BSTU
go

exec SP_SETTRIGGERORDER @triggername = 'AUD_AFTER_UPDATEB',
                        @order='First', @stmttype = 'UPDATE';
exec SP_SETTRIGGERORDER @triggername = 'AUD_AFTER_UPDATEA',
                        @order='Last', @stmttype = 'UPDATE';
```

Представления каталога

```
use BSTU
go
select t.name, e.type_desc
from sys.triggers  t join  sys.trigger_events e
                        on t.object_id = e.object_id
where OBJECT_NAME(t.parent_id)='AUDITORIUM' and
      e.type_desc = 'UPDATE' ;
```

name	type_desc
AUD_AFTER_UPDATE	UPDATE
AUD_AFTER	UPDATE
AUD_AFTER_UPDATEA	UPDATE
AUD_AFTER_UPDATEB	UPDATE
AUD_AFTER_UPDATEC	UPDATE

sys.triggers

```
select * from sys.triggers;
```

Results | Messages

name	object_id	parent_class	parent_class_desc	parent_id	type	type_desc	is_instead_of_trigger	is_disabled
TR_TEACHER_INS	370100359	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0
TR_TEACHER_DEL	386100416	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0
TR_TEACHER_UPD	402100473	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0
TR_TEACHER	418100530	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0
TR_TEACHER_DEL1	434100587	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0
TR_TEACHER_DEL2	450100644	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0
TR_TEACHER_DEL3	466100701	1	OBJECT_OR_COLUMN	501576825	TR	SQL_TRIGGER	0	0

sys.trigger_events

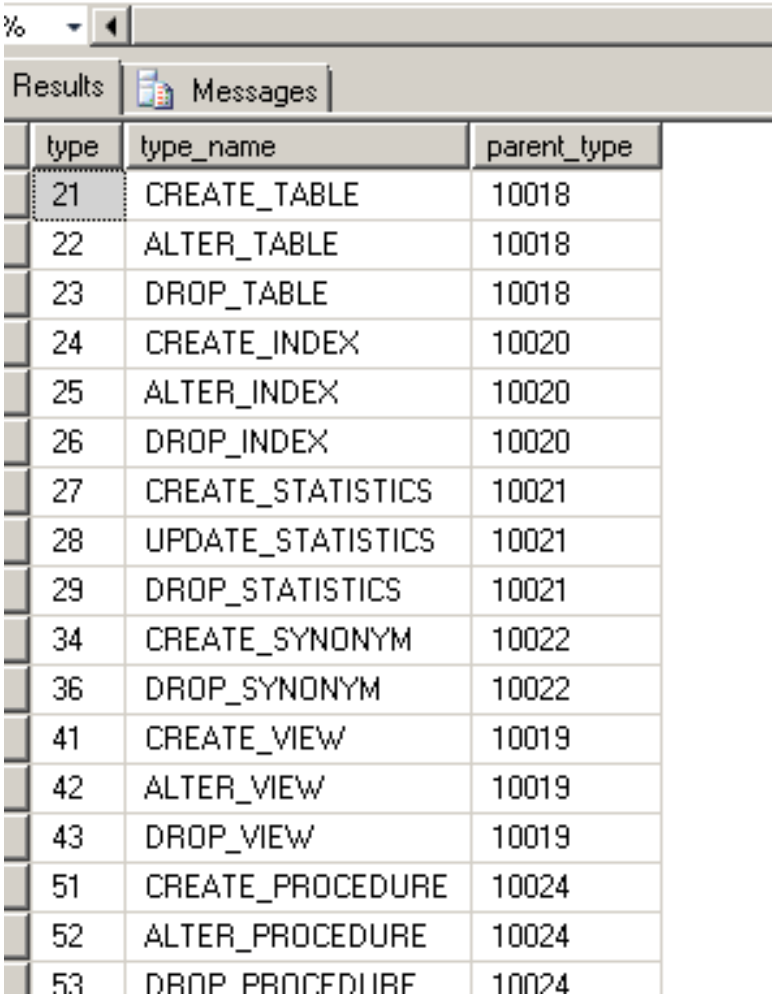
```
select * from sys.trigger_events;
```

Results | Messages

object_id	type	type_desc	is_first	is_last	event_group_type	event_group_type_desc	is_trigger_event
370100359	1	INSERT	0	0	NULL	NULL	1
386100416	3	DELETE	0	0	NULL	NULL	1
402100473	2	UPDATE	0	0	NULL	NULL	1
418100530	1	INSERT	0	0	NULL	NULL	1
418100530	2	UPDATE	0	0	NULL	NULL	1
418100530	3	DELETE	0	0	NULL	NULL	1
434100587	3	DELETE	0	0	NULL	NULL	1
450100644	3	DELETE	0	1	NULL	NULL	1
466100701	3	DELETE	1	0	NULL	NULL	1

sys.trigger_event_types

```
select * from sys.trigger_event_types;
```



Results | Messages

type	type_name	parent_type
21	CREATE_TABLE	10018
22	ALTER_TABLE	10018
23	DROP_TABLE	10018
24	CREATE_INDEX	10020
25	ALTER_INDEX	10020
26	DROP_INDEX	10020
27	CREATE_STATISTICS	10021
28	UPDATE_STATISTICS	10021
29	DROP_STATISTICS	10021
34	CREATE_SYNONYM	10022
36	DROP_SYNONYM	10022
41	CREATE_VIEW	10019
42	ALTER_VIEW	10019
43	DROP_VIEW	10019
51	CREATE_PROCEDURE	10024
52	ALTER_PROCEDURE	10024
53	DROP_PROCEDURE	10024

Порядок DML-триггеров

- Если для таблицы или представления созданы INSTEAD OF и AFTER-триггеры, реагирующие на одно и то же событие, то выполняется только INSTEAD OF- триггер

Порядок DML-триггеров

- sp_helptrigger

```
exec sp_helptrigger @tabname = 'TEACHER';
```

Results | Messages

trigger_name	trigger_owner	isupdate	isdelete	isinsert	isafter	isinsteadof	trigger_schema
TR_TEACHER_INS	dbo	0	0	1	1	0	dbo
TR_TEACHER_DEL	dbo	0	1	0	1	0	dbo
TR_TEACHER_UPD	dbo	1	0	0	1	0	dbo
TR_TEACHER	dbo	1	1	1	1	0	dbo
TR_TEACHER_DEL1	dbo	0	1	0	1	0	dbo
TR_TEACHER_DEL2	dbo	0	1	0	1	0	dbo
TR_TEACHER_DEL3	dbo	0	1	0	1	0	dbo

DDL-триггеры

- CREATE TRIGGER
- [schema_name.] trigger_name
- ON { ALL SERVER | DATABASE }
- [WITH { ENCRYPTION | EXECUTE AS clause_name]
- { FOR | AFTER }
- { event_group | event_type | LOGON }
- AS { batch | EXTERNAL NAME method_name }

DDL-триггеры

- триггеры уровня сервера (ALLSERVER)
- триггеры уровня базы данных (DATABASE)

Триггеры уровня сервера

- Триггеры **уровня сервера** обрабатывают события сервера СУБД:
  - Создание объектов сервера
  - Изменение объектов сервера
  - Удаление объектов сервера
  - Подключение к серверу

DDL-триггеры

```
use MASTER
go
create trigger DROP_DB
on all server
for DROP_DATABASE
as
    print  'удаление базы данных';
    insert SERVER_EVENT_LOG values(EVENTDATA());
go
```

Eventdata

```
<EVENT_INSTANCE>
   <EventType>DROP_DATABASE</EventType>
   <PostTime>2014-06-09T15:05:54.027</PostTime>
   <SPID>53</SPID>
   <ServerName>WIN-FPSVQQ7K4B3</ServerName>
   <LoginName>sa</LoginName>
   <DatabaseName>DB1</DatabaseName>
   <TSQLCommand>
     <SetOptions ANSI_NULLS="ON" ANSI_NULL_DEFAULT="ON"
                 ANSI_PADDING="ON" QUOTED_IDENTIFIER="ON"
                 ENCRYPTED="FALSE" />
     <CommandText>drop database [DB1]; </CommandText>
   </TSQLCommand>
 </EVENT_INSTANCE>
```

Logon

```
use MASTER
go
create trigger LOGON_SERVER
on all server
with execute as 'sa'
for LOGON
as
    insert SERVER_EVENT_LOG values(EVENTDATA());
go
```

Logon

```
<EVENT_INSTANCE>
  <EventType>LOGON</EventType>
  <PostTime>2014-06-09T20:53:19.510</PostTime>
  <SPID>56</SPID>
  <ServerName>WIN-FPSVQQ7K4B3</ServerName>
  <LoginName>sss</LoginName>
  <LoginType>SQL Login</LoginType>
  <SID>XXBAqoGtbUOpWPCPifiLXQ==</SID>
  <ClientHost>192.168.0.200</ClientHost>
  <IsPooled>0</IsPooled>
</EVENT_INSTANCE>
```

Триггеры уровня базы данных

- Триггеры **уровня базы данных** – обработка событий, происходящих в рамках базы данных

Триггеры уровня базы данных

```
use BSTU
go
create trigger DDL_DB
on database
for DDL_DATABASE_LEVEL_EVENTS
as
    insert MASTER.DBO.SERVER_EVENT_LOG values(EVENTDATA());
go
```

Триггеры уровня базы данных

```
<EVENT_INSTANCE>
   <EventType>CREATE_TABLE</EventType>
   <PostTime>2014-06-09T22:01:30.650</PostTime>
   <SPID>51</SPID>
   <ServerName>WIN-FPSVQQ7K4B3</ServerName>
   <LoginName>sa</LoginName>
   <UserName>dbo</UserName>
   <DatabaseName>BSTU</DatabaseName>
   <SchemaName>dbo</SchemaName>
   <ObjectName>t1</ObjectName>
   <ObjectType>TABLE</ObjectType>
   <TSQLCommand>
     <SetOptions ANSI_NULLS="ON" ANSI_NULL_DEFAULT="ON"
                 ANSI_PADDING="ON"  QUOTED_IDENTIFIER="ON"
                 ENCRYPTED="FALSE" />
     <CommandText>create table t1 ( x int); </CommandText>
   </TSQLCommand>
 </EVENT_INSTANCE>
```

Триггеры уровня базы данных

```
create trigger DDL_DB_DROP_TABLE
on database
for DROP_TABLE
as
 declare @t varchar(50)=
 EVENTDATA().value('(/EVENT_INSTANCE/ObjectName)[1]', 'varchar(50)');
 if @t = 'TEACHER'
 begin
   raiserror( N'удаление таблицы TEACHER запрещено', 16, 1);
   rollback;
 end;
```

Уровень 1	Уровень 2	Уровень 3
DDL_ EVENTS	DDL_ DATABASE_ LEVEL_ EVENTS	ALTER_INSTANCE
		DDL_DATABASE_SECURITY_EVENTS
		DDL_DEFAULT_EVENTS
		DDL_FUNCTION _EVENTS
		DDL_PROCEDURE_EVENTS
		DDL_SYNONYM_EVENTS
		DDL_TABLE_VIEW_EVENTS
		DDL_TYPE_EVENTS
		RENAME
	DDL_ SERVER_ LEVEL_ EVENTS	DLL_DATABASE_EVENTS
		DLL_ENDPOINTS_LEVEL
		DLL_EVENT_SESSION_EVENTS
		DLL_MESSAGE_EVENTS

Вопросы?