**Контейнеризация приложений**

Триггер – это хранимая процедура, выполняющаяся автоматически в ответ на действие пользователя.

DML - триггер уровня БД срабатывает при модификации данных (INSRERT, UPDATE, DELETE)

DDL триггер – триггер уровня БД и сервера

(CREATE, ALTER, DROP)

**DML триггеры**

Область применения:

1. Обеспечение целостности данных
2. Вывод предупреждений и модификаций
3. Логирование действия пользователя
4. Поддержка синхронизации копий БД

Особенности:

1. Выполняют медленнее, чем проверка CHECK
2. Могут выполняться при буксировании
3. Вызываются один раз для всей таблицы, а не для отдельных строк
4. Всегда выполняются внутри внешних транзакций

Системные таблицы, использующиеся DML триггером:

|  |  |
| --- | --- |
| MSSQL | MySQL |
| inserted | New (заполняются новыми версиями строк) (INSERT,UPDATE) |
| deleted | Old (заполняется старыми версиями строк) (DELETE, UPDATE) |
|  |  |
|  |  |

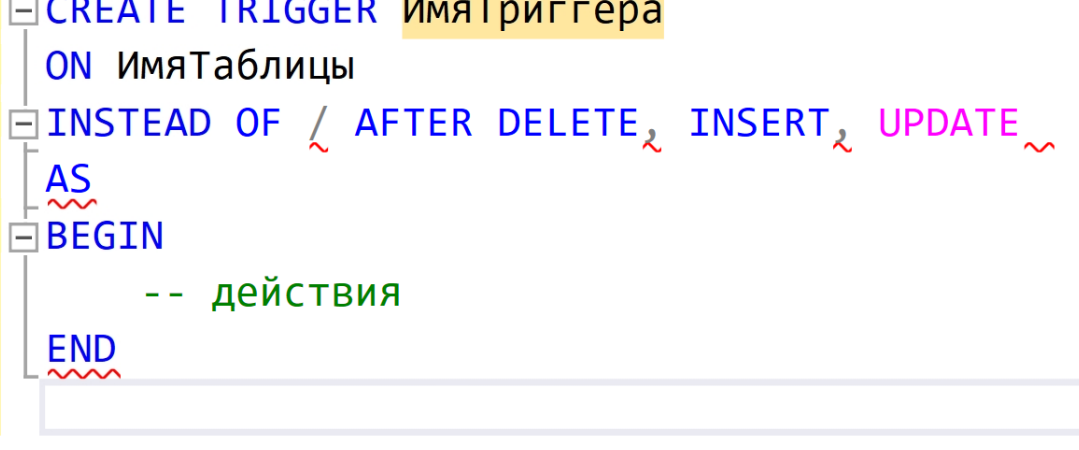
Виды DML триггеров:

INSTEAD OF – заменяет собой команду, которая вызвала срабатывания триггера.

AFTER – выполняется после команды, этих штук может быть несколько.

Порядок операций при выпонении DML команды:

1. Inseted / deleted
2. INSTEAD OF – если он есть, позволяет проверить данные на корректность, сохранить предыдущие версии строк.
3. CHECK – проверка
4. Выполнение DML команды
5. Срабатывание AFTER если он есть, для введения лога и изменения данных для изменения таблицы.



ПРИМЕР

CREATE TRIGGER trChangedGamesCount

ON Game

AFTER DELETE, INSERT, UPDATE

AS

BEGIN

PRINT CONCAT\_WS(' ', '~Изменено строк:', @@ROWCOUNT)

END;

CREATE TABLE [GamePrice](

[GameId] [int] NOT NULL,

[Price] [decimal](16, 2) NOT NULL,

[ChangingDate] [datetime2](7) NOT NULL DEFAULT GetDate(),

CONSTRAINT [PK\_GamePrice] PRIMARY KEY CLUSTERED

(

[GameId] ASC,

[ChangingDate] ASC

)

)

CREATE TRIGGER trSavePrice

ON Game

AFTER UPDATE

AS

BEGIN

IF UPDATE(Price)

INSERT INTO GamePrice(GameId, Price)

SELECT GameId, Price

FROM deleted;

END;

ЗАПИСЬ УДАЛЕННЫХ СТРОК В ТАБЛИЦУ

1. Создать таблицу, удалить категорию

CREATE TABLE [dbo].[DeletedCategory](

[CategoryId] [int] NOT NULL,

[Name] [nvarchar](100) NOT NULL,

[DeletedDate] [datetime2](7) NOT NULL,

[Login] [nvarchar](100) NOT NULL,

PRIMARY KEY CLUSTERED

(

[CategoryId] ASC

)

)

2. Создать триггер

CREATE TRIGGER trSaveCategory

ON Category

AFTER DELETE

AS

BEGIN

INSERT INTO DeletedCategory(CategoryId, Name, DeletedDate, Login)

SELECT CategoryId, Name, GetDate(), ORIGINAL\_LOGIN()

FROM deleted;

END;

3. Создание триггера заменяющего удаление триггера на пометку строки как удаленная:

1. Изменяем таблицу Game (добавляем столбец IsDeleted тип bit DEFAULT=0) + сохранение

2. Создание триггера

CREATE TRIGGER trDeleteGame

ON Game

INSTEAD OF DELETE

AS

UPDATE Game

SET IsDeleted = 1

WHERE GameId IN (SELECT GameId

FROM deleted);

3. Изменение записей связанных таблиц

В таблице Sale будет указываться SaledKeys

В таблице Game будет общее кол-во TotalKeys

Требуется при оформлении покупки уменьшать кол-во ключей у игры

1 ВАРИАНТ: (сначала создаём таблицу Sale)

CREATE TABLE [dbo].[Sale](

[SaleId] [int] IDENTITY(1,1) NOT NULL,

[GameId] [int] NOT NULL,

[SaledKeys] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[SaleId] ASC

)) ON [PRIMARY]

В таблицу Game добавить столбец TotalKeys(DEFAULT = 100, not allows null)

Оформление продажи без учёта имеющегося кол-ва ключей:

Sale -> CREATE TRIGGER

CREATE TRIGGER TrChangeKeysCount

ON Sale

AFTER INSERT

AS

UPDATE Game

SET TotalKeys -= SaledKeys

FROM Game JOIN inserted ON Game.GameId = inserted.GameId

2 ВАРИАНТ: (добавление проверки имеющегося кол-ва ключей)

CREATE TRIGGER [dbo].[TrChangeKeysCountWithCheck]

ON [dbo].[Sale]

INSTEAD OF INSERT

AS

BEGIN

DECLARE @saledKeys INT;

DECLARE @totalKeys INT;

SELECT @saledKeys = SaledKeys, @totalKeys = TotalKeys

FROM Game JOIN inserted ON Game.GameId = inserted.GameId;

IF @totalKeys >= @saledKeys

BEGIN

INSERT INTO Sale(GameId, SaledKeys)

SELECT GameId, SaledKeys

FROM Sale;

UPDATE Game

SET TotalKeys -= SaledKeys

FROM Game JOIN inserted ON Game.GameId = inserted.GameId;

END;

ELSE

RAISERROR('%s, недостаточно ключей для покупки %d ', 18, 1, 'user3515', @totalKeys);

END;