



آزمایشگاه پایگاه داده

جلسه ششم
قسمت دوم آغازگرها

محمد جواد آکوچکیان و محمود فرجی

سال تحصیلی ۱۴۰۱-۱۴۰۲



آغازگرها (Triggers)

- مجموعه ای از کد است که با رخ دادن رویدادی خاص فعال و اجرا می شود.

- دو نوع Trigger در Server SQL وجود دارد :

۱- آغازگرهای (Data Manipulation Language):DML

با تغییراتی که در داده ها با اعمال درج، حذف و تغییر اعمال می شوند.

۲- آغازگرهای (Data Definition Language):DDL

که با تغییرات دیگر در پایگاه داده مانند تعریف و حذف جدول فعال می شوند.

آغازگرها ابزارهای مناسبی برای واکنش اتوماتیک به برخی از رویدادها هستند.



آغازگرها (Triggers)

۱- آغازگرهای (DML(Data Manipulation Language):

- SELECT واکنشی اطلاعات از دیتابیس
- UPDATE ویرایش اطلاعات دیتابیس
- DELETE پاک کردن اطلاعات از دیتابیس
- INSERT INTO اضافه کردن اطلاعات جدید به دیتابیس

۲- آغازگرهای (DDL (Data Definition Language):

- CREATE DATABASE ایجاد یک دیتابیس جدید
- ALTER DATABASE ایجاد تغییرات در دیتابیس
- CREATE TABLE ایجاد یک table جدید
- ALTER TABLE اعمال تغییرات در table
- DROP TABLE پاک کردن یک table
- CREATE INDEX ایجاد یک شاخص
- DROP INDEX حذف یک شاخص



آغازگرهای DDL

- کاربردهای آنها:

- جلوگیری از بروز تغییرات خاص در دیتابیس
- با تغییر در ساختار دیتابیس اتفاقاتی که مورد نظر ما است هم اتفاق بیفتد
- این تریگرها میتوانند بر روی یک دیتابیس خاص یا کل سرور ایجاد شوند
- میتوان از آنها برای ثبت لاگ های اختصاصی مورد نظر ما یا هشدارها استفاده کرد



آغازگرهای DDL

```
CREATE TRIGGER trigger_name  
ON { DATABASE | ALL SERVER }  
[WITH ddl_trigger_option]  
FOR { event_type | event_group }  
AS { sql_statement }
```

- تعریف این آغازگر به صورت زیر است:

- برای تغییر یک آغازگر از دستور مقابل استفاده می گردد:

```
ALTER TRIGGER trigger_name ...
```

- برای حذف یک آغازگر :

```
drop trigger trigger_name on database
```

- اگر بخواهیم یک آغازگر خاص فعال و یا غیرفعال شود از دستورات زیر استفاده می گردد:

```
DISABLE TRIGGER { [ schema_name . ] trigger_name [ , ... n ] | ALL }  
ON { object_name | DATABASE | ALL SERVER } [ ; ]
```



آغازگرهای DDL



```
create trigger t2 on database  
for drop_table  
as  
print 'drop trigger is executing'
```

• مثال تعریف آغازگر



آغازگرهای DDL



• مثال تعریف آغازگر

```
create trigger safety
on database
for drop_table , alter_table
as
print 'you must disable trigger safety to drop or alter tables'
rollback;
drop table color
```



آغازگرهای DDL

- مثال تعریف آغازگر

```
create table logging(id int identity(1,1) primary key ,logdate datetime)
```

```
create trigger t3 on all server  
for create_table  
as  
insert into uni.dbo.logging values(GETDATE())
```




آغازگرهای DDL



- مثال تغییر در یک آغازگر

```
alter trigger t2 on database  
for drop_table  
as  
print 'drop trigger is executing after alter'  
rollback
```



آغازگرهای DDL



- مثال غیر فعال و فعال کردن یک آغازگر

```
disable trigger t2 on database  
enable trigger t2 on database
```



آغازگرهای DDL



- مثال حذف کردن یک آغازگر

```
drop trigger t2 on database
```

```
drop trigger t3 on all server
```



دستور کار

ابتدا یک جدولی با نام **AuditTable** بسازید که شامل فیلدهای **ID**، **EventType**، **DateTime**، **UserName**، **DatabaseName**، **SchemaName**، **TSQLCommand** باشند. (نوع و طول مقادیر این ستون‌ها را متناسب با سناریوی لاگ‌گیری آغازگر در نظر بگیرید).

سپس آغازگری تعریف کنید که با انجام عمل **Create Table**، **Alter Table** و **Drop Table** در جدول فوق لاگ مربوطه را **insert** کند.

راهنمایی: برای مقادیر در نظر گرفته شده در ستون‌های جدول **AuditTable**، می‌توانید از تابع **EVENTDATA()** استفاده کرده و مقادیر مربوط به ستون‌های جدول **AuditTable** را از خروجی این تابع که به صورت **XML** است، بگیرید. سپس برای سه عمل گفته شده کوئری بنویسید و اسکرین شات مربوطه را در گزارش خود نمایش دهید.