T-SQL KODLARİ İÇERİSİNE AÇIKLAMA EKLEME

Bir veya daha fazla satırın çalıştırılmasını Önlemek için veya /*... */" ifadeleri

kullanılır.

“--“ işareti tek satırlık açıklamalarda kullanılır. Açıklama olarak yazılan satırın önüne konulması yeterlidir.

"/*--*/" İşaretleri bir veya birden fazla satır içeren açıklamalarda kullanılır. Açıklama bilgisinin başına /* ve açıklamanın bittiği yere */ yazılması yeterlidir.

Örnek:

```
-- Personel tablosundaki kayıtları listeleyelim

SELECT * FROM personel

/* Aşağıdaki kodlar yardımıyla iki değişken
Tanımlanmış ve değişkenlere değer atanmıştır.*/

DECLARE @d1 INT,@d2 INT
SET @d1=10
SET @d2=15
```

SİSTEM FONKSİYONLARI

Aşağıdaki tabloda sistem değişkenlerinin bir kısmı verilmiştir.

Değişken İsmi	Açıklama
@@ERROR	Aktif bağlantılar için meydana gelen son hatanın hata numarasını içerir. Herhangi bir hata oluşmamışsa 0 değerini içerir.
@@ŞERVICENAME	SQL Server'ın kullandığı ve register 'da kayıtlı olan windows servis ismini içerir.
@@SERVERNAME	SQL Server'ın çalıştığı yerel sunucucunun ismini içerir,
@@VERSION	SQL Server'ın versiyonu hakkında bilgiler içerir.
@@LANGUAGE	SQL Server İçin geçerli olan dili içerir. (us_english, Deutsch, Français, Dansk, Espanol, italiano , v.b,).
@@LANGID	SQL Server için geçerli olan dilin sayısal karşılığını içerir. 0 = İngilizce, 1 = Almanca, 2 = Fransızca . . . v.b.
@@MAX_CONNECTIONS	İzin verilen eşzamanlı maksimum bağlantı sayısını İçerir.
@@SPID	Aktif kullanıcı için verilen oturum ID numarasını içerir.
@@DATEFIRST	SQL Server İçin ayarlı olan haftanın ilk gününün sayısal karşılığını içerir. Default değeri 7'dir. 1 = Pazartesi, 2 =Salı, 3 = Çarşamba, 4 = Perşembe, 5 = Cuma, 6 =Cumartesi, 7 = Pazar
@@ROWCOUNT	Son kullanılan SQL İfadesi sonucunda etkilenen satır sayısını içerir.
@@CONNECTIONS	SQL Server'ın son çalışmasından itibaren kullanılan bağlantı sayısını içerir.
@@CPU_BUSY	SQL Server'ın son çalışmasından itibaren boş kalmadan çalıştığı süreyi milisaniye cinsinden içerir.
@@IDLE	SQL Server'ın son çalışmasından itibaren boş kaldığı süreyi milisaniye cinsinden içerir.
@@PACK_RECEIVED	SQL Server'ın son çalışmasından itibaren ağ üzerinden aldığı paketlerin sayısını içerir,
@@PACK_SENT	SQL Server'ın son çalışmasından itibaren ağ üzerinden gönderdiği paketlerin sayısını içerir.
@@PACKET_ERRORS	SQL Server'ın son çalışmasından itibaren ağ üzerinden yapılan transferlerde meydana gelen hata sayısını içerir.
@@TOTAL_ERRORS	SQL Server'ın son çalışmasından itibaren disk üzerinden okuma yazma işlemi yapılırken meydana gelen hata sayısını içerir.
@@TOTAL_READ	SQl- Server'ın son çalışmasından itibaren meydana gelen disk okuma işleminin sayısını içerir.
@@TOTAL_WRITE	SQL Server'ın son çalışmasından İtibaren meydana gelen diske yazma işleminin sayısın içerir.

SELECT * **FROM sys.messages**

Örnek:

SELECT 5/0

SELECT @@ERROR

Meydana gelen hatayı açıklamasıyla birlikte görüntülemek için aşağıdaki SQL İfadesi kullanılır.

SELECT 5/0

SELECT * FROM master.dbo.sysmessages

WHERE error = @@ERROR

Bu sorgu sonucunda oluşan hatanın kodu ve açıklaması Results penceresinde görüntülenecektir.
Sorgu sonucunda oluşan son sütun(msglangid) SQL Server'ın kullanmış olduğu dili göstermektedir.

	error	severity	dlevel	description	msglangid
1	8134	16	0	Divide by zero error encountered	1033

Örnek: aşağıda verilen ifade SQL Server'ın sistem üzerinde kullanmış olduğu ismi verecektir.
SELECT @@SERVICENAME Sorgu Sonucu → SQLEXPRESS

Örnek: Aşağıda verilen SQL ifadesi SQL Server'ın çalıştığı yerel sunucunun İsmi verecektir.
SELECT @@SERVERNAME Sorgu Sonucu → MY022\SQLEXPRESS

Örnek: Aşağıdaki 5QL ifadesi SQL Server'ın kullanmış olduğu dili verecektir,

SELECT @@LANGUAGE AS 'Kullanılan Dil' Sorgu Sonucu us_english

Örnek: Aşağıdaki SQL ifadesi SQL Server'ın kullanmış olduğu dilin sayısal karşılığını verecektir,

SELECT @@LANGID Sorgu Sonucu O

SET LANGUAGE ifadesi kullanılarak dil ayarı değiştirilebilir,

SET LANGUAGE 'turkish'

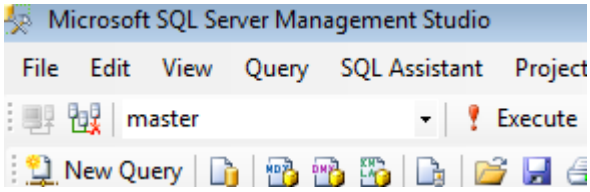
SELECT @@LANGID

Sorgu Sonucu → 22

USE İLE VERİTABANI SEÇİMİ

Yazılan SQL İfadelerinin hangi veritabanı üzerinde çalıştırılacağı kesinlikle seçilmelidir. Aksi durumda yazılan SQL ifadesi ile o anda kullanılan veritabanı alakasız olacağı için SQL ifadesi hata verecektir.

SQL İfadeleri Management Studlo ile hazırlanıyorsa veritabanı görsel ara yüz üzerinden de seçilebilir. Aşağıdaki resimde veritabanı seçimi yapılacak alan gösterilmektedir.



Kullanılacak ventabanını kodlar yardımıyla seçmek için USE ifadesi kullanılır.

Kullanım Sekli
USE veritabanı_adı

Değişkenler

Değişken tanımlama örneği; @deger1 ve @deger2 şeklindeki iki değişkenin değerlerini ve toplamını yazdırmak için aşağıdaki T-SQL yapısı kullanılır.

```
DECLARE @deger1 INT, @deger2 INT, @toplam INT

SET @deger1=125
SET @deger2=56
SET @toplam=@deger1+@deger2
PRINT 'Toplam Sonucu'
PRINT '----------------------------'
PRINT @toplam
```

```
Toplam Sonucu
-------------------------------------
181
```

TABLO TİPİ DEĞİŞKENLER

Tablonun değişken olarak tanımlanması

Kullanım Sekli

```
DECLARE @degişken_adi TABLE(
                                degişken_adi veritipi,
                                degisken_adi veritip,
                                )
```

Tablo tipi değişkenler tanımlanırken veritabanı nesnesi olan tablolarda kullanılan CONSTARINT ifadeler kullanılabilir.

Örneğin, ogr isminde tablo tipi değişken no ve ad isimlerinde iki değer tutuyor ve no değeri otomatik artıyorsa aşağıdaki tanımlanabilir.

```
DECLARE @ogr TABLE(
        no INT IDENTITY(1,1),
        isim NVARCHAR(20)
        )
```

Yukarıda gösterildiği şekilde elde edilen tablo tipi değişkenler diğer tablolarla birleştirilerek sorgulamalar yapılabilir.

Örnek: no, adsoyad ve telefon bilgilerini içerecek, no değeri otomatik artacak ve adsoyad bilgisi NULL değer içermeyecek şekilde personel tablo değişkenini oluşturarak içerisine değer girişi yapalım.

```
DECLARE @personel TABLE(
                    no INT IDENTITY (1, 1),
                    adsoyad VARCHAR(50) NOT NULL,
                    telefon VARCHAR(15)
                    )
INSERT into @personel VALUES ( ‘Kemal Kok' , '35612 ')

INSERT INTO @personel VALUES ('Burak Türk','35613')

SELECT * FROM @personel ORDER BY adsoyad
```

no	adsoyad	telefon
2	Burak Türk	35613
1	Kemal Kok	35612

OUTPUT İŞLEMİ

OUTPUT cümlesi INSERT, UPDATE, DELETE işlemleri sonrasında, işlemlerden etkilenen tüm kayıtlar hakkında bilgilere ulaşmak için kullanılır.

Kullanım şekli

```
UPDATE,INSERT,DELETE Cümlesi
OUTPUT Yapılacak_işlemler
WHERE Şartlar
```

Yapılacak işlemler kısmında INSERTED, DELETED gibi tablolar kullanılarak kayıtların eski ve yeni değerleri hakkındaki bilgilere ulaşılabilir, bu bilgiler tablolara insert edilebilir veya ekrana bastırılabilir. Kayıtlar üzerinde güncelleme geçmişi tutmak gibi işlemlerde işimize yarayabilecek bir yöntemdir.

	INSERT	UPDATE	DELETE
INSERTED	Yeni eklenen kayıtlar	Günceilenen kayıtlar	-
DELETED	-	Kayıtların eski durumu	Silinen kayıtlar

OUTPUT ifadesinin kullanım şekli aşağıda verilmiştir.

İnsert Örneği

```
DECLARE @eklenenler TABLE (
            ad VARCHAR(20) ,
            soyad VARCHAR(20) ,
            maas MONEY)

INSERT INTO tbl_personel(adi, soyadi,maas)
 OUTPUT INSERTED.adi,INSERTED.soyadi,INSERTED.maas INTO
@eklenenler(ad,soyad,maas)

VALUES  ('Ali' , 'Coşkun' , 1500)

SELECT * FROM @eklenenler
```

Delete Örneği

```
DECLARE @silinenler TABLE (
            ad VARCHAR(20),
            soyad VARCHAR(20),
            maas MONEY)
DELETE FROM tbl_personel
OUTPUT DELETED.adi,DELETED.soyadi,DELETED.maas INTO @silinenler
WHERE per_id>24
SELECT * FROM @silinenler
```

Update Örneği;

```
DECLARE @degisiklik TABLE (
            e_ad VARCHAR(20) ,
            e_soyad VARCHAR(20) ,
            e_maas MONEY,
            y_ad VARCHAR(20) ,
            y_soyad VARCHAR(20) ,
            y_maas MONEY)

UPDATE tbl_personel
SET maas = 1500
      OUTPUT INSERTED.adi,INSERTED.soyadi,INSERTED.maas, DELETED.adi,
      DELETED.soyadi, DELETED.maas
      INTO @degisiklik (y_ad,y_soyad,y_maas,e_ad,e_soyad,e_maas)
WHERE maas=1500

SELECT * FROM @degisiklik
```