Begin Try

-- HAta olabilecek durumlar

End try

begin catch

--hata durumunda ne yapılacak

end catch

Error\_LINE() --> HAta satırı

ERROR\_MESSAGE() -->HAta Bilgisi

ERROR\_NUMBER() --> Hata Numarası

ERROR\_SEVERITY()--> HAta derecesi

--> ORN

Begin try

select 1/0

end try

begin catch

select ERROR\_LINE(),ERROR\_MESSAGE(),ERROR\_NUMBER(),ERROR\_SEVERITY()

end catch

Use Yarisma

-->orn

select \* from Il

Select \* from Uye

Begin Try

declare @pno int

set @pno=1

delete from Il where Plaka\_No=@pno

end try

begin catch

if ERROR\_NUMBER() = 547

print('arcelikle ilişkili tablodaki kaydi silin')

else

print(Error\_Number())

end catch

--> Stored Procedure

--SQl sürücüsünde belli bir sorunu yerine getirmek için yaoılan alt programlardır.

--Bir kere derlenmesi yeterlidir.

1.SYSTEM STORED PROCEDURE

HAzır procedureler

SP\_ ön eki ile kullanılır

adını kullanarak çağırabilirsinizç

SP\_Helptext

2.EXTENDED STORED PROCEDURE

-- Hazır ve tanımlı procedureler. Master databse adı altnda tutulur.

HArici programlama dili oluşturulmuştur. Bir ddl içerisinde.

3. Local Stored PROCEDURE

--Kullanıcı tarafından oluşturulan stored procedure yapısında TSQL yapısında oluşturulabilir.

Create Procedure Prc\_adi(@pr1 tip)

AS

SQL ifadeleri...

--> Stored Procedure

--ÖRN Hiç soru göndermeyen üyelerden hangilerinin yarışmaya katıldığını listeleyecek bir stored procedure tanımla

Create Procedure uye\_liste1

AS

select \* from Uye where Uye\_No NOT IN (select Uye\_No from Soru)

and Uye\_No IN (select Uye\_No from istatistik)

exec uye\_liste1

alter procedure pre\_adi(@pr1..)

with recompile

AS

select ..

drop procedure prc\_Adi --> silme

--output ile

Create Procedure ort1(@a int=0,@b int=0,@ort real OUTPUT)

AS

set @ort=(@a+@b)/2

declare @snc real

exec ort1 10,20,@snc OUTPUT

print('sonuc=' + CAST(@snc as varchar(10)))

--return ile

Create Procedure ort2(@a int=0,@b int=0)

AS

return (@a+@b)/2

declare @snc real

exec @snc=ort2 10,20

print('Sonuc='+CAST(@snc as varchar(10)))

-- Fonksiyonlar

Yapısı itibariyle view ve procedure benzeyen bir alt programalama yapısıdır.

dışarıdan değer alabilir ve geriye tek bir değer tablo yada birleştirilmiş değerler döndürebilir.

Fonksiyonlar sql ile birlikte kullanabilirken procedurde böyle bir özellilk yoktur.

3 tip fonksiyon yapısı vardır.

1.SCALAR\_VALUE fonksiyon ==> geriye tek değer döndürür.

Create function Foksiyon\_Adi(@p1 tip)

returns veritipi

AS

Begin

-- SQL ifade

end

-->örn

Kendisine gelen uye numarasına göre bu üyenin kaçtane soru gönderdiğini geriye döndüren bir fonksiyon tanımlayınız ve bu fonksiyonu kullanınız.

Create Function uye\_soru\_say(@uno int)

Returns int

as

Begin

declare @ss int

select @ss = COUNT(Soru.Uye\_No) from Soru where Uye\_No=(select Uye\_No from Uye where Uye\_No=@uno)

return(@ss)

End

Go

select dbo.uye\_soru\_say(1) AS 'Soru Sayısı'

---------------------------------------------------------

2.TAble\_VAlued fonksiyon

-- Bir tablonun kendisini geriye döndürür

Create function f\_adi(@p1 tip)

REtURNs Table

as

REturn(select...)

go

-- her bir üyenin kaç tane soru gönderdiğini geriye döndüren

create Function Soru\_Uye\_Say()

Returns Table

AS

Return(select Uye.Uye\_Adi,Uye.Uye\_Soyadi,COUNT(Soru.Uye\_No) as 'Soru sayisi' from

Uye,Soru where Soru.Uye\_No=uye.Uye\_No group by

Uye.Uye\_Adi,Uye.Uye\_Soyadi)

select \* from dbo.Soru\_Uye\_Say()

-----------------------------------------------------

3.Aggregate fonksiyon(birleştirilmiş)

Bu tip fonksiyonlar farklı tablolardan farklı alanları alarak bir tablo tipi değişken şeklinde geriye döndüren fonksiyonlardır.

Create Function F\_adi(@paramerteler varsa...)

Returns @değişken table(.....)

AS

Begin

...... SQL ifade

END

Go

--> ÖRN ==> kendisine gelen uye numarasına göre kendisi ile en fazla puana sahip uyenin puan farki bu üyenin hangi soru türünden kaç soru gönderdiğini il adinı ve toplamda kaç kere yarıştığını geriye döndürecek bir fonksiyon tanımlayınız.

Create Function F\_Adi(@)

Returns @ table()