

In the name of god

SQL Server

View(Join) & Stored Procedure

**Amirkabir University of
Technology**

Behnaz Motavali

bs.motavali@yahoo.com

دستور Case

مقدار خاصی را بر اساس شرایط تعریف شده ، در دستور Select ، باز می گرداند.

قالب کلی این دستور به اشکال زیر است :

حالت اول :

Case	عبارت			
	When	مقدار اولیه	Then	مقدار بازگشتی
	...			
	[Else]			
End	[As	نام]		

حالت دوم :

Case				
	When	عبارت شرطی	Then	مقدار بازگشتی
	...			
	[Else]			
End	[As	نام]		

دستور Case

مثال :

```
Use [Lab-EasyShop]
Select      FirstName ,
            LastName ,
            Case      Category
                When 1 Then 'Teacher'
                Else      'Student'
            End As Position
From
Customer
```

مثال :

```
Use [Lab-EasyShop]
Select      FirstName ,
            LastName ,
            Case
                When      Category = 1          Then      'Teacher'
                When      Category Between 2 And 10      Then      'Student'
                Else      'Other'
            End As      Position
From
Customer
```

کار با متغیر ها در T-SQL

تعریف متغیر : یکی از اشیا در SQL Server است که برای حمل و نگهداری مقادیر مفرد (یکتا Single) به کار می روند .
متغیر ها در T-SQL به منظور های زیر به کار می روند:

- ۱- به عنوان یک شمارنده که معمولاً تعداد دفعات اجرای یک حلقه را مشخص می کنند .
 - ۲- برای حمل داده های استفاده شده در یک “کنترل جریان” (Control-of-Flow).
 - ۳- ذخیره داده ها به منظور بازگرداندن توسط یک Function و یا Stored Procedure
- از علامت @ به منظور معرفی یک متغیر استفاده می شود .
برای تعریف یک متغیر از قالب زیر استفاده می شود :

Declare @ VariableName [data type]

برای مقدار دهی به متغیر ها از یکی از قالب های زیر استفاده می شود :

Select @ VariableName = Value

یا

Set @ VariableName = Value

Control-of-Flow Language

کار با متغیر ها در T-SQL

مثال :

```
Declare @Number      Int
Set      @Number      =      10
Select   @Number      =      10
```

مثال :

```
Use [Lab-EasyShop]
Declare @Title      nVarChar(100),
        @Titles     nVarChar(Max)

Set      @Titles     =      N"
Select   @Title      =      Title
        From      Category
        Where ID = 1
Select   @Titles     =      Title + N',' + @Titles
        From      Category

Select   @Title      As      Title ,
        @Titles     As      Titles
```

Control-of-Flow Language

کار با متغیر ها در T-SQL

مثال :

```
DECLARE @DBNAME VARCHAR(50)  
SET @DBNAME = 'TEST'
```

```
CREATE DATABASE @DBNAME
```

زبان "کنترل جریان"

زبان کنترل جریان بخشی از دستورات T-SQL را در بر می گیرد که جریان اجرای دستورات را در یک بلوک دستورات ، یک User Defined Function و یا یک Stored Procedure بر عهده می گیرد .
بدون استفاده از دستورات این زبان، دستورات زبان T-SQL به صورت ترتیبی و پشت سرهم اجرا می شوند. با استفاده از این دستورات ، امکان تعریف شرط ، حلقه ، پرش از دستورات و نظایر آن فراهم می شود . کنترل جریان از دستورات زیر تشکیل شده است :

BEGIN...END

BREAK

GOTO

CONTINUE

IF...ELSE

WHILE

RETURN

WAITFOR

Control-of-Flow Language

دستور IF ... Else

برای تعریف شرط بکار می رود .

قالب کلی دستور به شکل زیر است :

If عبارت شرطی

کد دستورات مورد نظر

[ELSE]

[کد دستورات مورد نظر]

مثال :

```
Declare    @Number    Int
Set        @Number    = 15
```

```
if @Number % 2 = 0
```

```
    Print 'Even'
```

```
Else
```

```
    Print 'Odd'
```

Control-of-Flow Language

دستور IF ... Else

مثال :

مطلوبست جابجایی دو مقدار در متغیر ، چنانچه متغیر اولی از دومی
بزرگتر بود بدون استفاده از هیچ متغیر واسطه .

دستور IF ... Else

مثال :

مطلوبست جابجایی دو مقدار در متغیر ، چنانچه متغیر اولی از دومی بزرگتر بود بدون استفاده از هیچ متغیر واسط .

```
Declare    @FirstNumber  Int ,
           @SecondNumber Int
Select     @FirstNumber  =    100 ,
           @SecondNumber =    1

If          @FirstNumber >    @SecondNumber
Begin
    Set     @FirstNumber      = @FirstNumber + @SecondNumber
    Set     @SecondNumber    = @FirstNumber - @SecondNumber
    Set     @FirstNumber      = @FirstNumber - @SecondNumber
End

Else
Begin
    Declare @TempNumber      Int
    Set     @TempNumber      = @FirstNumber
    Set     @FirstNumber     = @SecondNumber
    Set     @SecondNumber    = @TempNumber
End

Select     @FirstNumber      As FirstNumber,
           @SecondNumber     As SecondNumber
```

دستور While

برای تعریف حلقه بکار می رود .

قالب کلی دستور به شکل زیر است :

While عبارت شرطی حضور در حلقه

[Begin]

کد دستورات مورد نظر

[End]

مثال :

```
Declare    @Number    Int
Set        @Number    = 100
```

```
While      @Number > 0
Begin
  Print    @Number
  Set      @Number    = @Number    - 1
End
```

دستور While

مثال :

مطلوبست اعداد زوج بين دو عدد

```
Declare  @FirstNumber  Int ,
          @SecondNumber Int
Select
    @FirstNumber  =    1 ,
    @SecondNumber =   100
Set @FirstNumber = @FirstNumber  + @FirstNumber % 2

While  @FirstNumber <= @SecondNumber
Begin
    Print  @FirstNumber
    Set @FirstNumber = @FirstNumber  + 2
End
```

View

جدولی است **مجازی** که اطلاعات ذخیره شده در بانک اطلاعاتی را به گونه ای متفاوت ارائه می کند .

از View به دلایل مختلفی استفاده می شود که ذیلأً به برخی از مهمترین علل آن اشاره می شود :

- ۱- فیلتر کردن فیلدهای یک جدول
 - ۲- فیلتر کردن رکوردهای یک جدول
 - ۳- ترکیب یک, دو یا چند جدول
 - ۴- تغییر ساختمان جداول بدون اثر گذاشتن بر روی لایه های بالاتر
-

پنج نوع View وجود دارد:

۱- System Views

۲- Standard Views

۳- Indexed Views

۴- Partitioned Views

۵- Temporary Views

دسته اول View ها ، اطلاعات سیستمی را نگهداری می کنند و قابل ویرایش یا حذف نیستند

دسته دوم View ها ، که موضوع درس ماست.

دسته سوم View ها ، پس از بحث Index ها مورد بحث قرار می گیرد .

دسته چهارم View ها ، برای ارتباط بین جداول بر روی سرورهای مجزا ، استفاده می شود .

دسته پنجم View ها ، موقتی هستند و پس از استفاده از بین می روند (در بانک اطلاعاتی tempdb)

ساخت View ها

برای ساخت View ها از دستور Create View استفاده می شود .

قالب کلی این دستور به شکل زیر است :

نام Create View

AS

Select ...

نکات مهم :

از دستور **Order By** جهت مرتب سازی اطلاعات در View ها **نمی توان** استفاده کرد .

بازیابی اطلاعات یک View ، **مانند بازیابی اطلاعات در جداول** می باشد و کلیه قوانین حاکم بر آن در اینجا نیز مصداق دارد .

دستور Create View بایستی **دستور اول در Query** قرار بگیرد ، چنانچه قبل از آن دستوری

بایستی اجرا شود ، بایستی با جداکننده Go جدا گردد .

Example

Create View InventoryView

As

Select

Item.Title As Item,
Color.Title As Color,
Inventory.Quantity

From

Item ,
Color ,
Inventory

Where

Item.ID = Inventory.Item_ID And
Color.ID = Inventory.Color_ID

ایجاد تغییرات در View ها

قالب کلی این دستور به شکل زیر است :

Alter View نام
AS
Select ...

مثال :

Alter View InventoryView
As

Select	Item.Title	As Item,
	Color.Title	As Color,
	Inventory.Quantity ,	
	Inventory.Quantity % 2	As Reminder

From

Item ,
Color ,
Inventory

Where

Item.ID	=	Inventory.Item_ID	And
Color.ID	=	Inventory.Color_ID	

حذف View ها

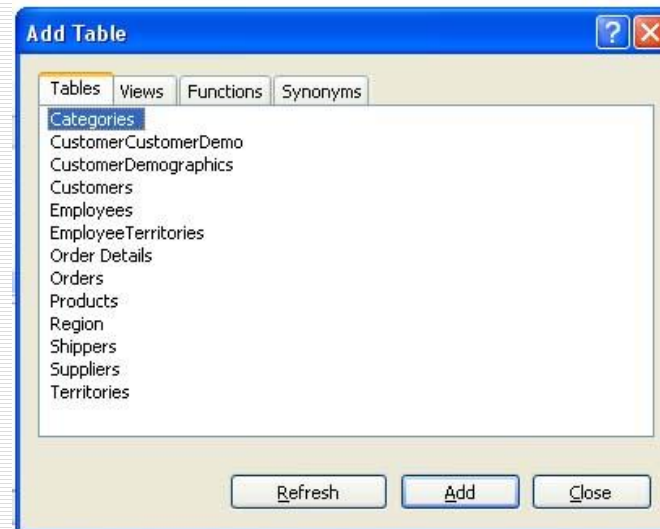
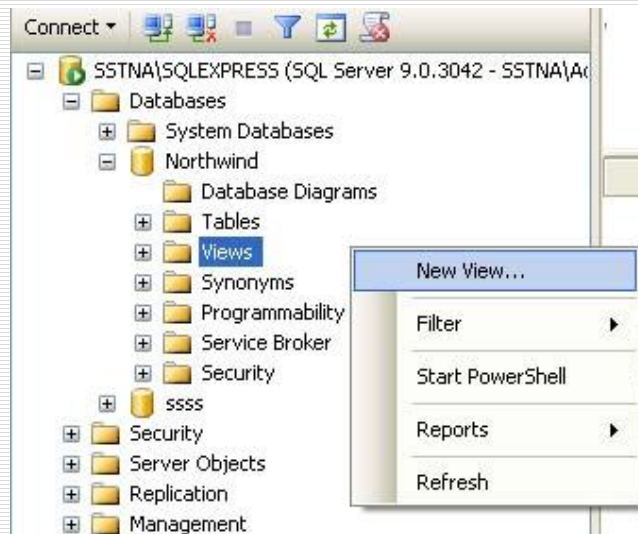
برای حذف View ها از دستور Drop View استفاده می شود .
قالب کلی این دستور به شکل زیر است :

نام Drop View

مثال :

Drop View InventoryView

ساخت View ها بصورت گرافیکی



ساخت View ها بصورت گرافیکی

SSTNA\SQLXP...- dbo.View_1*

Categories


- ☐ * (All Columns)
- ☐ CategoryID
- ☐ CategoryName
- ☐ Description
- ☐ Picture

CustomerCust...

- ☐ * (All Columns)
- ☐ CustomerID
- ☐ CustomerTypeID

CustomerDem...

- ☐ * (All Columns)
- ☐ CustomerTypeID
- ☐ CustomerDesc



	Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...	Or...
▶				<input checked="" type="checkbox"/>					
				<input checked="" type="checkbox"/>					
				<input checked="" type="checkbox"/>					
				<input checked="" type="checkbox"/>					

```
SELECT
FROM    dbo.CustomerDemographics INNER JOIN
        dbo.CustomerCustomerDemo ON dbo.CustomerDemographics.CustomerTypeID = dbo.CustomerCustomerDemo.CustomerTypeID CROSS JOIN
        dbo.Categories
```

Stored Procedure

قطعه برنامه هایی که عمدتاً از دستورات زبان T-SQL تشکیل می شوند و می توانند هرزمان توسط کاربر فراخوانی شوند، عملیات مورد نظر وی را بر روی اشیاء مختلف بانک اطلاعاتی انجام دهند .

دقت کنید که Stored Procedure ها توانایی بسیار در حل انواع مسائل مرتبط با بانک اطلاعاتی دارند و در معماری Fat Server کمک شایانی به کاهش بار ترافیک شبکه می نمایند .

از مهمترین ویژگیهای Stored Procedure ها می توان به موارد زیر اشاره کرد :

- ۱- قابلیت پذیرفتن پارامترهای متعددی را به عنوان ورودی و خروجی دارند .
 - ۲- امکان فراخوانی سایر Procedure ها را دارند و از یک مجموعه دستورات پشتیبانی می کنند .
 - ۳- هنگام فراخوانی ، موفقیت آمیز بودن و یا برخورد با اشکالات را می توانند گزارش نمایند .
-

Performance: In a client/server or internet environment, stored procedures can reduce network traffic because multiple SQL statements can be invoked with a single stored procedure. Only the request and the final results need to be sent across the network.

Reusability: Stored procedures allow code to reside in one place, the database server. Multiple client programs can call the procedures as needed, without duplicating code.

Maintenance: Code changes are only required in one place.

Data Integrity: Stored procedures can perform column validations, so if all applications use the same procedure, the data is always validated.

Security: If a given group of users requires access to specific data items, you can provide a stored procedure that returns just those items. You can then grant access to call the stored procedure, without giving those users any additional authorization to the underlying database objects.

Database protection: Stored procedures run in a separate address space from the database engine, eliminating the possibility of users corrupting the DBMS.

Stored Procedure

انواع مختلفی از Stored Procedure ها وجود دارند :

۱- System Stored Procedure

۲- Standard Stored Procedure

۳- Temporary Stored Procedure

۴- Extended Stored Procedure

Stored Procedure

دسته اول Stored Procedure هایی هستند که توسط خود SQL Server در اختیار کاربر قرار می گیرند و معمولاً دارای پیشوند "Sp_" می باشند .

دسته دوم Stored Procedure توسط کاربر و برای محقق سازی اهداف گوناگون، مورد استفاده قرار می گیرند.

دسته سوم Stored Procedure ها نیز توسط کاربران و به منظور استفاده موقت به کار گرفته می شوند .

دسته چهارم Stored Procedure آنهایی هستند که با زبانی به غیر از T-SQL ایجاد شده اند .

در نگارش SQL Server 2005 علاوه بر قابلیت های توسعه ای که در نسخ قبلی فقط بر روی بانک اطلاعاتی Master و زبان C++ وجود داشت ، امکان بوجود آوردن Stored Procedure هایی به زبان های مختلف

تحت NET Framework 2.0. نیز نظیر C# و یا VB.Net وجود دارد.

این دسته از Stored Procedure ها معمولاً دارای پیشوند "Xp_" می باشند .

اساس درس ما بر ایجاد و توسعه دسته دوم Stored Procedure انطباق دارد.

ایجاد Stored Procedure

برای ایجاد یک Stored Procedure از قالب زیر استفاده می شود

Create Procedure نام مورد نظر

(فهرست پارامترها)

As

Begin

لیست دستورات T-SQL

End

چند نکته :

۱- Stored Procedure ها می توانند یک مقدار Int را به عنوان نتیجه عملیات اجرایی خود ، با استفاده از دستور Return باز گردانند .

۲- پارامتر ها در Stored Procedure ها می توانند دارای مقادیر پیش فرض باشند .

۳- Stored Procedure می تواند مقادیر متغیر های ورودی را تغییر داده و در صورت نیاز مقادیر تغییر یافته فوق را با

استفاده از عبارت Output در هنگام تعریف متغیر ، باز گردانند .

مثال :

مطلوبست Stored Procedure برای جابجایی اعداد

```
Use [Lab-Inventory]
Go
Create Procedure  Swap
    (
        @FirstNumber      Int      Output ,
        @SecondNumber      Int      Output
    )
As
Begin
    Select
        @FirstNumber      = @FirstNumber  + @SecondNumber ,
        @SecondNumber = @FirstNumber  - @SecondNumber ,
        @FirstNumber      = @FirstNumber  - @SecondNumber
Select  @FirstNumber As Col1, @SecondNumber As Col2
End
```

مثال :

مطلوبست Stored Procedure برای بازگرداندن اعداد اول بین دو عدد

```
Use [Lab-Inventory]
```

```
Go
```

```
Create Procedure PrimeNumber(  
    @FirstNumber    Int    =1 ,  
    @SecondNumber    Int    =100  
)
```

```
As
```

```
    Begin
```

```
        ...
```

```
    End
```

اجرای Stored Procedure

برای اجرای یک Stored Procedure با استفاده از دستور Execute از قالب زیر استفاده می شود :

Execute *[فهرست پارامترها]* نام *SP* مورد نظر [=نام متغیر]

مثال :

Execute PrimeNumber 3,1000

چند نکته :

۱- چنانچه هنگام اجرای Stored Procedure ها ، تمایل داریم از مقادیر پیش فرض استفاده کنیم ، می توانیم از

عبارت Default به جای مقدار ورودی استفاده کنیم

مثال :

Execute PrimeNumber Default , 500

اجرای Stored Procedure

۲- در صورتیکه پارامتری در Stored Procedure بصورت Output تعریف شده ، هنگام فراخوانی آن نیز می بایست از عبارت **Output** در کنار مقدار ورودی استفاده نمود و **ضمناً مقدار ورودی بایستی حتماً یک متغیر** باشد .

مثال :

```
Declare    @X      Int ,  
           @Y      Int  
  
Select    @X= 100 ,   @Y = 20  
  
Execute    Swap  @X   Output ,   @Y Output
```

اجرای Stored Procedure

۳- برای دریافت مقدار بازگشتی یک Stored Procedure ، ابتدا بایستی یک متغیر از نوع Int تعریف کرده و در زمان اجرای Stored Procedure مقدار خروجی را در آن قرار داد .
مثال :

```
Declare @Result Int
Execute @Result = PrimeNumber 6,250
Select @Result As [Count]
```

۴- امکان صدا زدن یک Stored Procedure در عملیات و دستورالعمل های یک Stored Procedure دیگر وجود دارد ، تنها بایستی دقت کرد که تو در تو بودن عملیات ، تا ۳۲ مرحله قابل انجام است . این محدودیت شامل حال Stored Procedure هایی که با زبان CLR در .Net نوشته می شوند ، نمی شود .
برای پی بردن به مرحله تو رفتگی جاری می توان از تابع @@NestLevel استفاده نمود .

Like Operator

اپراتور مقایسه ای Like برای مقایسه مقدار یک رشته با الگویی خاص به کار می رود :

«الگوی مورد نظر» Like «رشته مورد نظر»

Character	Description	Example
%	همه چیز	Title LIKE '%com%' acomputers , computers : True FirstName LIKE 'aha%' ahang , ahani hangar : True sahab , jahan : False
_ (underscore)	همه چیز با طول یک کاراکتر	FirstName LIKE '_ean' Dean, Sean, Bean : True Alean , Roean : False
[]	یکی از اعضای مجموعه	LastName LIKE '[CL]arsen' Carsen & Larsen : True Parsen : False CLarsen: ?
[-]	یکی از اعضای کران مشخص	LastName LIKE '[C-P]arsen' Carsen, Larsen, Karsen, Parsen : True Yarsen : False

Like Operator

[^]	غیر از یکی از اعضای مجموعه یا کران مشخص	<p>LastName LIKE 'de[^]%'</p> <p>همه فامیلهایی که با de شروع می شوند اما حرف بعدی آنها l نیست .</p> <p>debtor , dekka , demote : True</p> <p>delta , delvar : False</p>
[%]	%	<p>Operation LIKE '5[%]'</p> <p>5% : True</p> <p>25% : False</p>
[[]]	[<p>Operation LIKE 'N[[]'</p> <p>N[: True</p> <p>N[m] : False</p>
[_]	_	<p>Operation LIKE '[_]n'</p> <p>_n : True</p> <p>m_n : False</p>
]]	<p>Operation LIKE '5]'</p> <p>5] : True</p> <p>25] : False</p>

Like Operator

برای درک بهتر به چند مثال زیر توجه کنید :

۱- سال شمسی بین ۱۳۳۰ تا ۱۳۹۹

۲- ۵ رقمی بودن شماره دانشجویی

۳- ستون Scode اگر با حرف A شروع شد با ارقام ۱ الی ۵ خاتمه یابد ، اگر با حرف B شروع شد با ارقام ۶ الی ۹ خاتمه یابد .

Like Operator

برای درک بهتر به چند مثال زیر توجه کنید :

۱- سال شمسی بین ۱۳۳۰ تا ۱۳۹۹

Year Like '[1][3][3-9][0-9]'

۲- ۵ رقمی بودن شماره دانشجویی

StudentNumber Like '[1-9][0-9][0-9][0-9][0-9]'

۳- رشته Scode اگر با حرف A شروع شد با ارقام ۱ الی ۵ خاتمه یابد ، اگر با حرف B شروع شد با ارقام ۶ الی ۹ خاتمه یابد .

Scode Like 'A %[1-5]' Or Scode Like 'B %[6-9]'

توابع رشته ای در SQL Server

Ascii	مقدار کد Ascii مربوط به کاراکتر مورد نظر را برمی گرداند
Char	یک کد Ascii مربوط را به یک کاراکتر معادل تبدیل می کند
Unicode	مقدار کد Unicode مربوط به کاراکتر مورد نظر را برمی گرداند
NChar	یک کد Unicode مربوط را به یک کاراکتر معادل تبدیل می کند
CharIndex	در یک رشته محل شروع یک عبارت را اعلام می کند
Left	تعداد مشخص کاراکتر از سمت چپ رشته را بر می گرداند
Right	تعداد مشخص کاراکتر از سمت راست رشته را بر می گرداند
Substring	تعداد مشخص کاراکتر از محل مشخص شده در رشته را بر می گرداند
Lower	رشته را به حروف کوچک تبدیل می کند
Upper	رشته را به حروف بزرگ تبدیل می کند
Replace	در کل رشته مورد نظر عبارت خاصی را با عبارت دیگری جایگزین می کند
LTRim	کل فضا های خالی سمت چپ یک رشته را حذف می کند
RTrim	کل فضا های خالی سمت راست یک رشته را حذف می کند
Space	به تعداد مورد نظر ، فضای خالی بر می گرداند
Reverse	یک رشته را بصورت برعکس بر می گرداند
Len	طول یک رشته را بر می گرداند.
Str	یک مقدار عددی را به یک رشته تبدیل می کند.
Replicate	یک رشته را به تعداد مشخص تکرار می کند .

کنترل خطاها با SEH

در SQL Server امکان کنترل خطاها به روش SEH ، (Structured Exception Handling) با استفاده از بلوک Try-Catch وجود دارد . هنگام بروز خطا در بلوک Try ، اجرای دستورات به بلوک Catch منتقل خواهد شد در بلوک Catch می توان از دو تابع Error_Message() و Error_Number() برای بررسی جزئیات خطا استفاده نمود .

قالب کلی استفاده از آن به شکل زیر است :

Begin Try

...

End Try

Begin Catch

...

End Catch

ایجاد خطا در SQL Server

از دستور RaisError برای تولید خطا در SQL Server استفاده می شود .
قالب کلی استفاده از آن به شکل زیر است :

RaisError (محل بروز خطا, شدت خطا, پیام خطا)

نکته : اعداد مربوط به شدت خطا و محل بروز خطا اعدادی دلخواه هستند که توسط کاربر وارد می شوند
اما محدوده خاصی از آنها در SQL Server تفسیر خاصی دارند .

مثال :

Example

```
BEGIN TRY
    RAISERROR ('Error raised in TRY block.', 16, 1 );
END TRY
BEGIN CATCH
    DECLARE @ErrorMessage NVARCHAR(4000);
    DECLARE @ErrorSeverity INT;
    DECLARE @ErrorState INT

    SELECT
        @ErrorMessage = ERROR_MESSAGE(),
        @ErrorSeverity = ERROR_SEVERITY(),
        @ErrorState = ERROR_STATE();

    RAISERROR (@ErrorMessage, @ErrorSeverity, @ErrorState );
END CATCH;
```
