SQL Server

Part: View(Join) & Stored Procedure

آزمایشگاه پایگاه داده

حمیدرضا رمضانی

h.ramezany72@gmail.com

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 ذیلاً 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

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

ساخت View ها

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

قالب کلی این دستور به شکل زیر است :

```
Create View نام
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 برآن 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 .

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

```
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 Viewها

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
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

حذف Viewها

برای حذف View ها از دستور Drop View استفاده می شود .

قالب کلی این دستور به شکل زیر است :

Drop View نام

مثال :

Drop View InventoryView

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

انواع Join ها

یکی دیگر از روشهای ترکیب و تلفیق اطلاعات در جداول مختلف ، استفاده از Join هاست .

سه نوع Join وجود دارد :

۱- Inner Join

۲- Outer Join

۳- Cross Join

تلفیق اطلاعات با استفاده از Join ها علاوه بر پوشش دادن طیف وسیع تری از نحوه تلفیق اطلاعات ، از حجیم شدن کد در دستور Where و همچنین کند شدن بر اثر فیلتر کردن اطلاعات جلوگیری می کند.

Join ها در دستور From بکار می روند.

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

Inner Join

برای تلفیق اطلاعات دو یا چند جدول که در رابطه منطقی دارای مقادیر متناظر هستند ، بکار می رود .

قالب کلی آن به شکل زیر است :

جدول ۱

Inner Join

جدول ۲

On **عبارت شرطی برای ایجاد یک رابطه منطقی**

مثال :

```
Select
        Item.Title ,
        Inventory.Quantity
  From
        Item
        Inner Join
        Inventory
          on      Item.ID      =   Inventory.Item_ID
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

Inner Join

مثال :

```
Select
        Item.Title              As Item,
        Color.Title             As Color,
        Inventory.Quantity
   From
        Item
          Inner Join
        Inventory
          on    Item.ID     =   Inventory.Item_ID
          Inner Join
        Color
          on    Color.ID    =   Inventory.Color_ID
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

Outer Join

برای تلفیق اطلاعات دو یا چند جدول که در رابطه منطقی حتی دارای مقادیر متناظر نیستند ، بکار می رود

به سه صورت می تواند وجود داشته باشد :

۱- Left Outer Join

۲- Right Outer Join

۳-Full Outer Join

قالب کلی آن به شکل زیر است :

جدول ۱

{Left | Right | Full} Outer Join

جدول ۲

On

عبارت شرطی برای ایجاد یک رابطه منطقی

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

Cross Join

خروجی Cross Join ، حاصل ضرب دکارتی مجموعه رکوردهای دو جدول می باشد .

دقت کنید : در Cross Join قسمت On و عبارت شرطی مرتبط کننده جداول حذف می‌شود

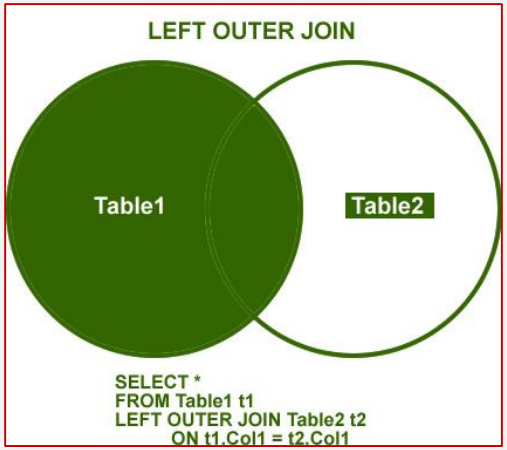
مثال :

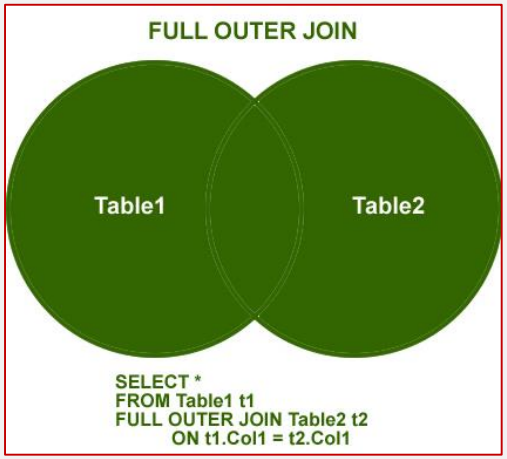
```
Select
        Item.Title          As Item,
        Color.Title         As Color
  From
        Item
           Cross Join
        Color
```

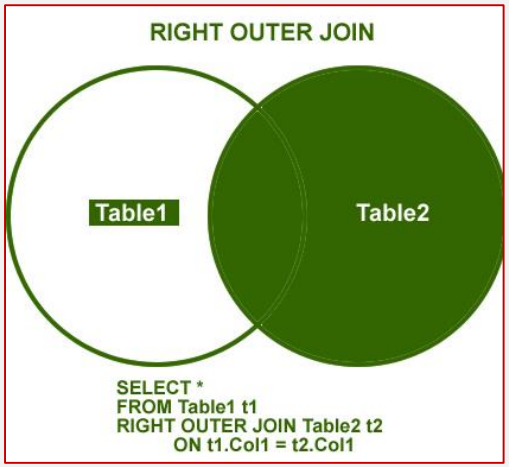
آزمایشگاه پایگاه داده
حمیدرضا رمضانی

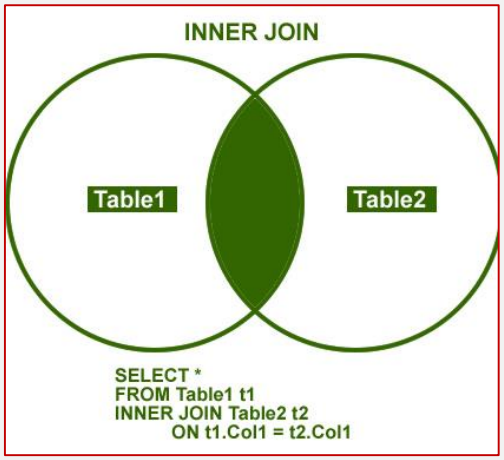
JOIN

آزمایشگاه پایگاه داده
حمیدرضا رمضانی









دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

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 توانند مقادیر متغییر 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 .

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

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
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

اجرای Stored Procedure

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

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

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
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

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
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

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.

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

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
```

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

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 دلبخواه هستند که توسط کاربر وارد می شوند اما محدوده خاصی از آنها در SQL Server تفسیر خاصی دارند .

مثال :

آزمایشگاه پایگاه داده
حمیدرضا رمضانی

```
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;
```