آموزش دستورات مختصر و مفید SQL (بخش سوم)

بهزاد عبدالخالقی

دستورات SQL مختصر و مفید : (قسمت سوم)

۱. Index View:

زمانیکه نیاز به خواندن اطلاعات از چندین Table با اطلاعات زیاد باشد استفاده از Index View ها می تواند مفید باشد. فرض کنید گزارشی داریم که اطلاعات آن از چندین جدول خوانده می شود. در حالت عادی هربار که اقدام به تهیه گزارش می کنیم می بایستی رکوردها از جداول مختلف Fetch شده و نمایش داده شوند که این عمل برروی Performance تاثیر منفی دارد , جهت بهبود این روند می توان از Materialize View ها و اعمال Index برروی آنها استفاده نمود . استفاده از Index ها دسترسی به اطلاعات را بسیار سریعتر می کند . (بخصوص زمانیکه تعداد رکوردها بسیار زیاد باشد)
Materialize View ها همانند یک Table عمل می کنند که مقادیر رکوردها را در خود ذخیره می کنند بنابراین با Update شدن جداول ، مقادیر View نیز Update می شود و در نتیجه هنگام تغییرات جداول بدلیل اعمال همزمان تغییرات در View یک Over Head خواهیم داشت که در برخی موارد این Over Head می تواند قابل چشم پوشی باشد . با توجه به این موضوع بهتر است در مواردی که به تغییرات در جداول کم است از این نوع View ها استفاده کنیم .

نکته : در هر View فقط می توان یک Unique Clustered Index تعریف کرد ولی بتعداد نامحدودی می توان Index های Non-Clustered تعریف کرد .

نکته : Index View در نسخه های SQL Server Enterprise و SQL Server Development Edition قابل اجرا می باشد و در SQL Server Personal Edition قابل اجرا نمی باشد.

Index View ها در SQL Server به دو طریق قابل استفاده می باشند .

روش اول :

View مستقیما توسط یک Query عادی صدا زده شود , ولی بجای اینکه از دستور SELECT بطور مستقیم استفاده کنیم از دستور Unique Clustered استفاده کنیم .

روش دوم :

هر Query که در SQL Server 2000 اجرا می شود بطور اتوماتیک چک کند که Index Query وجود دارد یا خیر .

برای اینکه بهترین استفاده را از Index View ها SQL Server داشته باشیم بهتر است از SQL Server Enterprise استفاده کنیم . در صورتیکه از نسخه های دیگر SQL Server 2000 استفاده کنید آنها بصورت اتوماتیک از Query Optimizer استفاده نمی کنند .

مواردی که پیشنهاد می شود از Index View ها استفاده نمود :

- اطلاعات تجارتی (Data Marts) , Data warehouse , سیستمهای تصمیم گیرنده (Decision Support) , Data mining و OLAP Application .
- View هایی که حاصل Join دو یا چند جدول بزرگ می باشند .
- View هایی که حاصل مجموعه ای از اطلاعات هستند (Aggregate Data)
- Repeated Pattern of Queries

مواردی که پیشنهاد نمی شود از Index View ها استفاده نمود :

- OLTP Application هایی که تغییرات Database ی آنها زیاد است .
- Query هایی که حاصل از Join چند Table نمی باشند .

انتخاب بهترین Unique Clustered Index ها و non- Clustered Index ها بسیار مهم و اساسی است .

توجه به این نکته مهم است که برروی یک جدول و View مربوط به آن نباید Index یک Index تعریف کرد چرا که باعث افزونگی اطلاعات می شود .

شرایط استفاده از Index View ها :

- در صورتیکه تغییرات داده های جداول زیاد است نباید از Index View ها استفاده نمود
- در صورتیکه تعداد رکوردهای نتیجه View با تعداد رکوردهای جدول پایه یکسان باشد استفاده از Index View برروی Performance ممکن است اثری نداشته باشد .
- بجای ایجاد یک Index View بزرگ بهتر است از دو یا چند Index View کوچکتر استفاده شود .البته در صورتیکه Join بین جداول از Database های مختلف باشد یا View نتیجه UNOIN کردن چند جدول باشد از Index View ها استفاده نکنید .
- گاهی اوقات نمی توان یک Index View ایجاد کرد که نتیجه مورد نظر ما را حاصل کند و می بایستی دو یا چند Index View ایجاد کرد تا بتوان به نتیجه مورد نظر دسترسی پیدا کرد .

راه حل دوم جهت استفاده از Index View ها بکار بردن "Clustered index" ها می باشد :

همانطور که می دانید Index View ها فضای حافظه را بصورت فیزیکی اشغال می کنند و به دلیل Update شدن همزمان با تغییرات جداول پایه دارای Over Head می باشند . ولی نباید فراموش کنیم که Index View تنها یک View است و View ها را می توان با دستورات WHERE Clause محدود کرد .

مثال : فرض کنید یک View داریم که حاصل Join دو جدول می باشد که جدول اول ۵ میلیون رکورد دارد و جدول دوم ۲ میلیون رکورد . می خواهیم یک Index View با کمترین Over head ایجاد کنیم . در این مثال نتیجه پرس و جو (Query) حداکثر ۲۰۰ رکورد وجود خواهد داشت بنابراین پشتیبانی SQL Server از آن بسیار ساده و سریع خواهد بود چرا که تعداد رکوردها در مقایسه با حجم کل رکوردهای دو جدول اصلی بسیار ناچیز است و SQL Server فقط کافیست که تغییرات در Index View را کنترل کند نه در جداول بزرگ اصلی . بنابراین یکی از راههایی که می توان به یک View با سرعت بالا در Fetch کردن اطلاعات رسید محدود کردن حجم رکوردها می باشد .

اکثر مواقع در طراحی Database به Performance به عنوان یک موضوع جانبی نگاه می کنیم و بنابراین بسیاری از اصول را از ابتدا در نظر نمی گیریم :

گاهی اوقات رعایت نکردن این اصل باعث می شود تا ما به View ها و Query های بسیار پیچیده روی آوریم تا بتوانیم مشکل Performance را رفع کنیم چرا که طراحی سیستم و کد برنامه را براحتی نمی توانیم تغییر دهیم .اگر شما SQL Server 2000 Enterprise را اجرا کنید ابزار جدیدی را مشاهده می کنید که به شما برای رفع این مشکل کمک می کنند . این ابزار در واقع همان Index View ها می باشد . زمانیکه مستقیما نمی توان برای یک Query یک Index View ایجاد کرد با استفاده از ابزار Query Optimizer می

توان این کار را انجام داد . بطور مثال اگر یک Query با Performance ضعیف داشته باشیم که توسط برنامه (Application) اجرا می شود می توان با استفاده از Index View سرعت آنرا به مراتب بهبود ببخشید .

ایده آل ترین زمان برای ایجاد Index View ها زمانی است که Clustered و non-Clustered Index ها برای جدول ایجاد می شود :

در صورتیکه دید مناسبی از نظر فنی به Table ها نداشته باشیم و جدول را طراحی کرده و Data Entry هم داشته باشیم درصورتیکه بخواهیم Index View تعریف کنیم افزونگی و سربار زیادی خواهیم داشت . افزونگی اطلاعات سربار زیادی به سیستم وارد می کند و I/O Performance زیادی خواهیم داشت بنابراین می بایستی حتی الامکان سعی کنیم تا این حالت اتفاق نیافتد . در صورتیکه ایجاد Index View بصورت همزمان میسر نباشد حتما می بایستی آنرا در نظر داشته باشیم تا در اولین زمان ممکن نسبت به ایجاد آن اقدام نماییم.

استفاده از SQL Server Index Wizard :

یکی از راههای مناسب برای Index View ها استفاده از SQL Server Index Wizard است . زمانیکه Index Wizard را اجرا می کنید SQL Server بطور اتوماتیک Index های بالقوه (Potential View) را جستجو می کند و آنها را پیشنهاد می کند ولی توجه داشته باشید هرگز به این ابزار بعنوان بهترین راه حل اعتماد نکنید چرا که این ابزار تمام شرایط مختلفی که شما در جداول خودتان در نظر دارید را نمی شناسد و ممکن است شرایطی را ایجاد کند که در طراحی شما دیده نشده باشد .

Index گذاری روی View :

فرمت کلی دستور به شکل زیر است :

```
CREATE [ UNIQUE ] [ CLUSTERED | NONCLUSTERED ] INDEX index_name
   ON { table | view } ( column [ ASC | DESC ] [ ,...n ] )
[ WITH < index_option > [ ,...n] ]
[ ON filegroup ]

< index_option > :: =
   { PAD_INDEX |
      FILLFACTOR = fillfactor |
      IGNORE_DUP_KEY |
      DROP_EXISTING |
   STATISTICS_NORECOMPUTE |
   SORT_IN_TEMPDB
}
```

آرگومانها :

UNIQUE:
یک Index یکتا (Unique) برروی Table یا View ایجاد می کند . (در هر Table یا View) فقط یک Unique Index می توان ایجاد کرد .
یک Clustered Index در یک View باید Unique باشد .
زمانیکه یک Index از نوع Unique ایجاد می شود SQL Server در همه شرایط (Insert , Update) تست می کند که اطلاعات تکراری ایجاد نشود (Duplicate Data) و در صورتیکه این شرایط اتفاق افتد خطا رخ می دهد .

CLUSTERED:
یک Unique Clustered Index باید قبل از هر Index دیگری روی یک View اعمال شود .

در صورتیکه از یک Table استفاده کنیم و روی آن Index را اعمال کنیم سرعت View آن با View معمولی تفاوتی نمی کند چرا که در اینصورت هنگامیکه View معمولی هم گرفته می شود از Index ی که برروی Index View گذاشته شده است استفاده می کند .

۱,۱. س: مقایسه View معمولی و Index View

View معمولی :

```
CREATE VIEW SimpleView AS
SELECT MVPersId, ElmntRef, PersRef, IssueYear, IssueMonth, EffectYear, EffectMonth, Val FROM
PayMVPers
```

Index View :

```
CREATE VIEW IndexView WITH SCHEMABINDING AS
SELECT MVPersId, ElmntRef, PersRef, IssueYear, IssueMonth, EffectYear, EffectMonth, Val
FROMdbo. PayMVPers
```

نکته : در Index View ها حتما باید نام جدول را با Owner آن مشخص کنید (dbo.PayMVPers)
نکته : در صورتیکه Owner جداول با یکدیگر فرق کنند Index View آنرا پشتیبانی نمی کند.

اعمال Index برروی Index View :

```
CREATE UNIQUE CLUSTERED INDEX Idx1 ON IndexView (MVPersId)
WITH FILLFACTOR = 80
```

دسته بندی در View معمولی :

```
SELECT * FROM SimpleView
WHERE IssueYear = 1382
  AND IssueMonth = 1
  AND EffectYear = 1382
  AND EffectMonth = 2
```

دسته بندی در Index View :

```
SELECT * FROM IndexView
WHERE IssueYear = 1382
  AND IssueMonth = 1
  AND EffectYear = 1382
  AND EffectMonth = 2
```

مثال)
دسته بندی در View معمولی :
دستور ایجاد View :

```
CREATE VIEW IndexView  AS
SELECT PMV.*, E.ElmntName, P.FName, P.LName
FROM PayMVPers PMV, Element E, Personnel P
WHERE PMV.ElmntRef = E.Serial
  AND PMV.PersRef = P.Id
```

دستور SELECT بکار رفته :

```
SELECT PMV.*, E.ElmntName, P.FName, P.LName
FROM PayMVPers PMV, Element E, Personnel P
WHERE PMV.ElmntRef = E.Serial
```

```
  AND PMV.PersRef = P.Id
```

دسته بندی در Index View :

فرمت دستور :

```
CREATE VIEW IndexView2 WITH SCHEMABINDING AS
SELECT PMV.MVPersId, PMV.ElmntRef, PMV.PersRef, PMV.IssueYear, PMV.IssueMonth,
PMV.EffectYear,
  PMV.EffectMonth, PMV.Val, E.ElmntName, P.FName, P.LName
FROM dbo.PayMVPers PMV, dbo.Element E, dbo.Personnel P
WHERE PMV.ElmntRef = E.Serial
  AND PMV.PersRef = P.Id
```

دستور SELECT بکار رفته :

```
SELECT PMV.MVPersId, PMV.ElmntRef, PMV.PersRef, PMV.IssueYear, PMV.IssueMonth,
PMV.EffectYear,
  PMV.EffectMonth, PMV.Val, E.ElmntName, P.FName, P.LName
FROM dbo.PayMVPers PMV, dbo.Element E, dbo.Personnel P
WHERE PMV.ElmntRef = E.Serial
  AND PMV.PersRef = P.Id
```