آموزش دستورات مختصر و مفید SQL (بخش پایانی)

بهزاد عبدالخالقی

۱. تنظیمات SQL Server جهت کارآیی (Performance) بهتر:

۱,۱. بهینه سازی tempdb Database :

اگر tempdb Database توسط برنامه های مختلف دائما در حال استفاده است بهتر است آنرا به گونه ای تنظیم نمایید تا بهترین کارآیی را داشته باشد. همانطور که می دانید استفاده از tempdb ها فضای زیادی را نیاز دارد ، بنابراین بهتر است تا با استفاده از تکنیکهای موجود این فضا را سازماندهی شده در اختیار آن قرار دهیم.

۱,۱,۱. س: چطور مي توان مسير tempdb Database را پس از نصب SQL Server تغییر داد.

ج : ممکن است پس از مدتی کار با SQL Server به لحاظ زیاد شدن حجم tempdb Database فضای هارد دیسک اشغال شده و امکان ادامه کار با بانک اطلاعاتی میسر نباشد لذا می بایستی مسیر tempdb Database را تغییر دهیم . این کار توسط دستورات زیر امکان پذیر می باشد.

```
USE master
go

ALTER DATABASE tempdb MODIFY FILE (NAME = tempdev, FILENAME =
'E:\tempdb.mdf')
go

ALTER DATABASE tempdb MODIFY FILE (NAME = templog, FILENAME =
'E:\templog.ldf')
Go
```

نکته : مسیر E:\tempdb.mdf مسیری است که می خواهیم tempdb Database را به آنجا منتقل کنیم.

پس از اجرای این دستور می بایستی SQL Server را یکبار Restart کنید تا تغییرات اعمال شوند.

۱,۲. بهینه سازی JOIN :

۱,۲,۱. درصورتیکه جداول مورد استفاده در Join دارای قاعده منظمی هستند یکی از راههای بالا بردن Performance استفاده از Index برروی فیلدهایی است که در Join استفاده شده است. (Joined Columns)

۱,۲,۲. در صورتیکه Join بین جداول دائما در حال تغییر باشد (یعنی Join بین دو یا چند جدول بسته به شرایط دائما تغییر کند و دارای نظم خاصی نباشد) بهتر است متناسب با این تغییرات از Index ها استفاده گردد. درصورتیکه فیلدهای استفاده شده بطور طبیعی Compact شده نباشند می توان از فیلدهای معادل بعنوان کلید برای هر فیلد استفاده نمود.

۱,۲,۳. بخاطر داشته باشید زمانیکه Foreign key برروی جدول اطلاعاتی ایجاد می گردد ، بطور اتوماتیک Index در همان زمان ایجاد نمی شود. لذا هنگامیکه در Join بین این جدول با جدول دیگر از فیلد Foreign key استفاده می کنید

می بایستی Index هم ایجاد کنید. افزودن Index به Foreign Key سرعت بسیار بالایی را در خواندن اطلاعات ایجاد می کند.

۱,۲,٤. هر گز از Join بین جداولی که دارای فیلد یکتا (Unique) نمی باشند استفاده نکنید. به لحاظ اینکه این فیلدها نمی توانند Index شوند (فیلد Index برروی فیلدهایی که دارای مقدار Unique هستند قابل اعمال است) لذا سرعت اجرای Join بسیار پایین است. بهترین شکل برای Performance بالا در این حالت استفاده از Unique Index ها می باشد.

۱,۲,٥. بهترین Performance استفاده از Index ها برروی فیلدهای عددی (Numeric) است. بهتر است از فیلدهای VarChar, Char و یا دیگر فیلدهای غیر عددی در Join استفاده نشود.

۱,۲,٦. برای بالاتر بردن Performance می بایستی فیلدهای Join شده دارای یک نوع Type حتی الامکان یک اندازه (Width) باشند. بعبارتی نمی بایستی فیلدها non-Unicdoe را با فیلدهای Unicdoe ترکیب کنید(مثلا VarChar و NVarChar) حتی در صورتیکه SQL Server تایپ این فیلدها را بتواند به یکدیگر تبدیل کند ، Join بین این فیلدها نه تنها دارای سرعت پایینی است بلکه امکان Index گذاری برروی آنها توسط SQL Server نیز وجود نخواهد داشت.

۱,۲,۷. یکی دیگر از روشهای بهبود Performance ، محدود کردن شرطها با استفاده از دستور WHERE است بعبارتی با استفاده از WHERE تعداد رکوردهای قابل بازیابی را محدود می کنیم. بنابراین بهتر است هنگام Join کردن چند جدول اطلاعاتی تمام رکوردها را بازیابی نکنیم و رکوردهای مورد نیاز را با استفاده از شرطهای مورد نظر فراخوانی کنیم.

۱,۲,۸. حتی الامکان از * در هنگام SELECT استفاده نکنید و فیلدهای مورد نیاز را به تفکیک استفاده کنید . حتی درصورتیکه نیاز به استفاده از تمام فیلدها در Join است بهتر است نام تمام فیلدها را در دستور قید کنید. استفاده از * دارای چندین اشکال است :

۱,۲,۸,۱. خوانایی دستور را پایین می آورد.

۱,۲,۸,۲. هنگام فراخوانی رکوردها می بایستی اطلاعات بیشتری را بازیابی کند .(بعنوان مثال در جدول Personnel در صورتیکه فقط نام افراد را بخواهیم درصورتیکه از * استفاده کنیم می بایستی اطلاعات فیلدهای دیگر نیز بازیابی شوند که بلا استفاده می باشند) همانگونه که اشاره شد هر چه تعداد رکوردهای قابل بازیابی کمتر باشند سرعت بالاتر خواهد رفت.

۱,۲,۸,۳. فیلدهای Join تکراری خواهند بود (مثلا در Join بین جداول Personnel و PersAddress در صورتیکه از * استفاده کنیم فلید Id بعنوان فیلد ارتباط دهنده بین دو جدول یکبار از جدول Personnel و باردیگر از جدول PersAddress بازیابی می شود)

۱,۲,۹. حتی الامکان از CROSS JOIN استفاده نکنید مگر اینکه تنها راه باشد. بسیاری از افراد کم تجربه از CROSS JOIN ها استفاده می کنند و سپس برای از بین بردن اطلاعات تکراری از GROUP BY و یا DISTICNT استفاده می کنند.

۱,۲,۱۰. در برخی موارد نیاز به استفاده از دستورات محاسباتی در SELECT ها می باشد (دستوراتی از قبیل SUM, Average) یکی از راههای محاسبه این فیلدها استفاده از Subquery می باشد. در اینحالت در یک Subquery عملیات محاسباتی انجام می شود و Query اصلی نیز الباقی اطلاعات مورد نیاز بازیابی می شوند. در این روش هنگام بازیابی اطلاعات به ازای هر رکورد می بایستی عملیات محاسباتی آن صورت گرفته و در بعنوان یک فیلد در دستور قرار گیرد . این روش در صورتیکه تعداد رکوردها زیاد باشد یا تعداد Subquery های محاسباتی زیاد باشند دارای Performance مناسبی نمی باشد. راه حل دیگری پیشنهاد می گردد که دارای Performance بهتری نسبت به روش اول می باشد . در این روش می توان هر Subquery محاسباتی را بطور جداگانه اجرا کرد و Query اصلی را نیز بدون Subquery ها بطور جداگانه اجرا کرد بنابراین تعدادی Query وجود خواهد داشت که در فیلد Join با یکدیگر مشترک می باشند در این حالت کافیست با استفاده از دستورات Left Join و یا Right Join این نتایج را با یکدیگر Join نمود . در روش دوم به لحاظ اینکه عملیات مربوط در Subquery ها بصورت یکجا انجام می شوند لذا دارای سرعت بالایی بوده و در نتیجه Performance بهتری را خواهیم داشت.

۱,۳. موارد استفاده از Temp Table ها:

حتی الامکان می بایستی از Temp Table ها استفاده نشود. استفاده از Temp Table ها در Tempdb ایجاد می شوند و علاوه براینکه سربار (Overhead) برروی SQL Server می گذارند باعث پایین آمدن Performance نیز می شوند. تنها در برخی استفاده از از Temp Table ها مانعی ندارد. برخی از این استثناها عبارتند از :

۱,۳,۱. هنگامی که بخواهیم نتیجه یک Query را از یک Nested Stored Procedure به Stored Procedure دیگری منتقل کنیم که در اینصورت تنها راه حل ممکن استفاده از Temp Table ها می باشد.

۱,۳,۲. در صورتیکه بخواهیم در Stored Procedure از Cursor استفاده کنیم . با توجه به سرعت پایین Cursor ها می توان از Temp Table ها بعنوان راه حل استفاده نمود. البته در صورتیکه راه حل بهتری بتوانید پیدا کنید که بجای جایگزینی Temp Table با Cursor از آن استفاده کنید، پیشنهاد می شود تا از آن روش استفاده نمایید.

۱,۳,۳. درصورتیکه راه حلی جز استفاده از Temp Table ها نداشته باشید، می توانید با رعایت موارد زیر از آن استفاده نمایید:

۱,۳,۳,۱. سعی کنید فقط فیلدهایی را که نیاز دارید در Temp Table درج نمایید.

۱,۳,۳,۲. از دستور SELECT INTO برای ایجاد Temp Table استفاده نکنید بلکه ابتدا با استفاده از دستور CREATE TABLE جدول مجازی (Temp Table) را ایجاد کنید و سپس با استفاده از دستور INSERT INTO رکوردهای مورد نظر را در آن درج نمایید.

۱,۳,۳,۳. حتما سعی کنید از clustered و non- clustered Index ها در Temp Table ها استفاده نمایید. (مخصوصا هنگامیکه تعداد رکوردهای موجود در Temp Table ها بسیار زیاد باشد)

۱,۳,۳,۴. حتما پس از استفاده از Temp Tale ها آنرا حذف کنید (در اینصورت فضای Tempdb آزاد می شود) و هرگز منتظر نباشید تا با قطع شدن Connection بطور اتوماتیک Tempdb حذف شود.

۱,۳,۳,۵. در صورتیکه استفاده از Temp Table ها در برنامه هایتان زیاد است بهتر است با تنظیمات SQL Server مسیر آنرا به یک دیسک مجزا منتقل کنید تا فضای اصلی کمتر اشغال گردد.

۱,۳,۳,۶. هرگز Table Temp ها را در Transaction ها ایجاد نکنید چراکه در اینصورت برخی از جداول سیستمی (Sysindex ,Syscolumns ,Syscomments) در حالت Lock باقی می مانند و عملا تا پایان Transaction آزاد نمی شوند و بنابراین بسیاری از دستورات که توسط کاربران دیگر صادر می شود تا پایان Transaction غیر قابل اجرا خواهد بود . برای جلوگیری از این مشکل بهتر است قبل از Transaction آنرا ایجاد کنید تا جداول سیستمی Lock نشوند