عنوان: مفهوم READ_COMMITTED_SNAPSHOT در EF 6
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۴ ۸:۰
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, SQL Server

مدتی است که حالت READ_COMMITTED_SNAPSHOT بسیار مورد توجه واقع شده:
- در سایت Stack overflow از آن استفاده می‌شود (^).
- در SQL Server Azure حالت پیش فرض ایجاد دیتابیس‌ها و تراکنش‌های جدید است (^).
- در Entity framework 6 حالت پیش فرض تراکنش‌های ایجاد شده، قرار گرفته است (^).

و ... در Oracle، تنها حالت مدیریت مسایل همزمانی است! (البته به نام MVCC ، اما با همین عملکرد)

اما READ_COMMITTED_SNAPSHOT در SQL Server چیست و کاربرد آن کجا است؟

اگر استفاده گسترده و سنگینی از SQL Server داشته باشید، حتما به پیغام‌های خطای deadlock آن برخورده‌اید:

```
Transaction (Process ID 54) was deadlocked on lock resources with another process and has been chosen
as the deadlock victim.
Rerun the transaction.
```

روش پیش فرض مدیریت مسایل همزمانی در SQL Server، حالت READ COMMITTED است. به این معنا که اگر در طی یک تراکنش مشغول به تغییر اطلاعاتی باشیم، سایر کاربران از خواندن نتیجه آن (اصطلاحا به آن Dirty read گفته می‌شود) منع خواهند شد؛ تا زمانیکه این تراکنش با موفقیت به پایان برسد. هرچند در این حالت سایر تراکنش‌ها امکان ویرایش یا حذف اطلاعات را خواهند داشت. به علاوه اگر در طی این تراکنش، اطلاعاتی خوانده شوند، سایر تراکنش‌ها تا پایان تراکنش جاری، قادر به تغییر این اطلاعات خوانده شده نخواهند بود (منشاء بروز خطاهای deadlock یاد شده در سیستم‌های پرترافیک).
در SQL Server 2005 برای بهبود مقیاس پذیری SQL Server و کاهش خطاهای deadlock، مکانیزم READ_COMMITTED_SNAPSHOT معرفی گشت.
به صورت خلاصه زمانیکه که تراکنش مورد نظر تحت حالت READ COMMITTED SNAPSHOT انجام می‌شود، optimistic reads and pessimistic writes خواهیم داشت (خواندن‌های خوشبینانه و نوشتن‌های بدبینانه). در این حالت تضمین می‌شود که خواندن اطلاعات داخل یک تراکنش، شامل اطلاعات تغییر یافته توسط سایر تراکنش‌های همزمان نخواهد بود. همچنین زمانیکه در این بین، اطلاعاتی خوانده می‌شود، بر روی این اطلاعات برخلاف حالت READ COMMITTED قفل قرار داده نمی‌شود. بنابراین تراکنش‌هایی که درحال خواندن اطلاعات هستند، تراکنش‌های همزمانی را که در حال نوشتن اطلاعات می‌باشند، قفل نخواهد کرد و برعکس.

نحوه فعال سازی READ_COMMITTED_SNAPSHOT

فعال سازی READ_COMMITTED_SNAPSHOT باید ابتدا در سطح یک بانک اطلاعاتی SQL Server انجام شود:

```
ALTER DATABASE testDatabase SET ALLOW_SNAPSHOT_ISOLATION ON;
ALTER DATABASE testDatabase SET READ_COMMITTED_SNAPSHOT ON;
```

کاری که در اینجا انجام خواهد شد، ایجاد یک snapshot یا یک کپی فقط خواندنی، از بانک اطلاعاتی کاری شما می‌باشد. بنابراین در این حالت، زمانیکه یک عبارت Select را فراخوانی می‌کنید، این خواندن، از بانک اطلاعاتی فقط خواندنی تشکیل شده، صورت خواهد گرفت. اما تغییرات بر روی دیتابیس اصلی کاری درج شده و سپس این snapshot به روز می‌شود.
حالت READ_COMMITTED_SNAPSHOT خصوصا برای برنامه‌های وبی که تعداد بالایی Read در مقابل تعداد کمی Write دارند، به شدت بر روی کارآیی و بالا رفتن سرعت و مقیاس پذیری آن‌ها تاثیر خواهد داشت؛ به همراه حداقل تعداد deadlockهای حاصل شده.

در Entity framework وضعیت به چه صورتی است؟

EF از حالت پیش فرض مدیریت مسایل همزمانی در SQL Server یا همان حالت READ COMMITTED در زمان فراخوانی متد SaveChanges استفاده می‌کند.

در EF 6 این حالت پیش فرض به READ_COMMITTED_SNAPSHOT تغییر کرده است. البته همانطور که عنوان شد، پیشتر باید بانک اطلاعاتی را نیز جهت پذیرش این نوع تراکنش‌ها آماده ساخت.

اگر از نگارش‌های پایین‌تر از EF 6 استفاده می‌کنید، برای استفاده از حالت READ_COMMITTED_SNAPSHOT باید صراحتا IsolationLevel را مشخص ساخت:

```
using (var transactionScope =
  new TransactionScope(TransactionScopeOption.Required,
  new TransactionOptions { IsolationLevel= IsolationLevel.Snapshot }))
{
   // update some tables using entity framework
   context.SaveChanges();
   transactionScope.Complete();
}
```

نظرات خوانندگان

نویسنده: ali.rezayee
تاریخ: ۱۳۹۲/۰۴/۱۴ ۱۵:۴۱

با سلام و تشکر بخاطر این مطلب عالی.
امکان دارد در خصوص بخش « READ_COMMITTED_SNAPSHOT در SQL Server چیست و کاربرد آن کجا است؟ » یک مثال عملی بزنید، مطلب کمی گنگ است.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۴ ۱۸:۱۳

فرض کنید یک جدول نظرات دارید با این تعریف

```
CREATE TABLE [BlogComments](
[Id] [int] IDENTITY(1,1) NOT NULL,
[Body] [nvarchar](max) NULL,
[Date1] [datetime] NOT NULL,
 CONSTRAINT [PK_BlogComments] PRIMARY KEY CLUSTERED
(
[Id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,
 ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

بعد در management studio دو پنجره اجرای کوئری جدید را ایجاد کنید. در پنجره اول بنویسید:

```
-- در پنجره اول
BEGIN TRAN
UPDATE [BlogComments] SET Body='Test' WHERE id=1
```

در پنجره دوم بنویسید

```
-- در پنجره دوم
SELECT TOP 1000 [Id]  ,[Body] ,[Date1] FROM [BlogComments]
```

- ابتدا عبارت پنجره اول را اجرا کنید. این پنجره حاوی یک تراکنش نا تمام است. شروع دارد اما به عمد پایان آن‌را ذکر نکردیم.
- الان پنجره دوم را اجرا کنید.
مشاهده خواهید کرد که ... به جواب نمی‌رسید. کوئری اجرا نمی‌شود و سیستم قفل شده چون تراکنش اول commit نشده (مثلا یک تراکنش طولانی را اینجا شبیه سازی کردیم؛ یا حتی یک اشتباه در تعاریف T-SQL انجام شده).

در ادامه، عملیات این پنجره‌ها را دستی متوقف کنید. بعد مطابق دستوراتی که پیشتر ذکر شد، READ_COMMITTED_SNAPSHOT را روی دیتابیس فعال کنید.
مجددا دو مرحله قبل را اجرا کنید. در این حالت کوئری دوم اجرا خواهد شد، چون اطلاعات را از کپی فقط خواندنی بانک اطلاعاتی شما دریافت می‌کند؛ بر اساس آخرین اطلاعات commit شده در سیستم.

نویسنده: مرادی
تاریخ: ۱۳۹۲/۰۴/۱۵ ۱۰:۵۳

با سلام، در قسمتی از مطلبتان، آورده اید " یک snapshot یا یک کپی فقط خواندنی، "

که این پیش فرض اشتباه را در ذهن خواننده ایجاد می‌کند که از دیتابیس یک کپی گرفته می‌شود، در حالی که از دیتابیس کپی

گرفته نمی‌شود، بلکه Snapshot حاوی تغییرات دیتابیس از لحظه ایجاد Snapshot تا به حال است، مثلا می‌گوید در جدول People، سه رکورد درج شده است، از تفاضل دیتابیس و این Snapshot می‌توان به وضعیت دیتابیس قبل از ایجاد Snapshot پی برد، به همین دلیل است که ایجاد یک Snapshot ولو روی یک دیتابیس چند گیگابایتی نیز در کسری از ثانیه انجام می‌پذیرد.
علاوه بر این امکان استفاده از این امکان در SQL 2000 نیست، ولو با ADO.NET، که البته چیزی رو از ارزش‌های این روش کم نمی‌کند، با سپاس از مطلب خوبتان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۵ ۱۱:۴۸

بحث اصلی هم همین نحوه و محل ذخیره سازی snapshot است.
- Sanpshot مطابق واژه نامه مایکروسافت معنای «نگارش» را می‌دهد. در این حالت کلیه کوئری‌های داخل یک تراکنش، یک نگارش یا snapshot از دیتابیس را مشاهده خواهند کرد. این نگارش یا Row version، در *tempdb* نگه داری می‌شود. با فعال سازی SNAPSHOT isolation، هر زمانیکه یک ردیف به روز رسانی می‌شود، موتور SQL Server یک نسخه از اطلاعات اولیه این ردیف را در tempdb ذخیره می‌کند (اینجا بود که عنوان شد با یک کپی فقط خواندنی از اطلاعات در حین واکشی اطلاعات سر و کار خواهید داشت).
خلاصه الگوریتم کاری آن :
الف) با آغاز یک تراکنش، یک عدد متوالی منحصربفرد تراکنش (شماره نگارش) ایجاد شده و به آن نسبت داده می‌شود.
ب) در حین این تراکنش، موتور SQL Server، به tempdb مراجعه کرده و شماره نگارشی نزدیک و کمتر از شماره نگارش تراکنش جاری را پیدا می‌کند. همچنین SQL Server بررسی می‌کند که این شماره یافت شده حتما جزو تراکنش‌های پایان یافته سیستم باشد.
ج) بر اساس این شماره یافت شده، نگارش معتبری از اطلاعات از tempdb استخراج می‌شود.
به این ترتیب یک تراکنش، کلیه اطلاعات موجود در ابتدای کار خود را بدون قرار دادن قفلی بر روی جداول مرتبط، دریافت خواهد کرد.
اطلاعات بیشتر

- در متن ذکر گردید که از SQL Server 2005 به بعد قابلیت فوق اضافه شده.
- همچنین SQL Server 2000 دیگر پشتیبانی رسمی ندارد و استفاده از آن حداقل از لحاظ امنیتی معقول نیست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۲۳ ۱۲:۵

خبری در اینباره از تیم SQL Server بعدی

```
The Hekaton team also found that multi-version concurrency control (MVCC) proved robust
in scenarios with higher workloads and higher contention
```

اطلاعات بیشتر

نویسنده: amir ranjbarian
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۳۷

با سلام؛ در صورتی که بخواهیم این مورد را در دیتابیسی که از filestream استفاده میکنه فعال کنیم با این خطا ALTER DATABASE failed because the READ_COMMITTED_SNAPSHOT and the ALLOW_SNAPSHOT_ISOLATION options cannot be set to ON when a database has FILESTREAM filegroups. To set READ_COMMITTED_SNAPSHOT or ALLOW_SNAPSHOT_ISOLATION to ON, you must remove the FILESTREAM filegroups from the database.
مواجه می‌شویم من در دیتابیس از filestreame استفاده کردم برای ذخیره فایلهای مورد نیاز در نرم افزار خودم، می‌خواستم بدونم آیا استفاده از این روش (filestreame) اصولا خوب هست یا نه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۴۵

این محدودیت از نگارش **R2** اس کیوال سرور 2008 به بعد برطرف شده .