بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

جداسازی منابع مورد نیاز تراکنش توسط Transaction Isolation Level ها

مدرس : مهندس افشین رفوآ

دوره آموزش SQL Server Administration

جداسازی منابع مورد نیاز تراکنش توسط Transaction Isolation Level ها

Isolation level هایی که در زیر نام برده شده برای تراکنش هایی که به جداول memory-optimized دسترسی دارند، قابل استفاده می باشند.

SNAPSHOT

REPEATABLE READ

SERIALIZABLE

READ COMMITTED

Isolation level یک تراکنش می تواند به عنوان بخشی از بلاک تجزیه ناپذیر stored procedure کامپایل شده به dll تعریف شود. به هنگام دسترسی به جداول بهینه سازی شده بر اساس حافظه از طریق Transact-SQL تفسیر شده، می توان با استفاده از table-level hint ها، isolation level را تعیین کرد.

شما باید isolation level تراکنش را حین تعریف stored procedure کامپایل شده به dll، مشخص کنید. شما همچنین باید isolation level را به هنگام دسترسی به جداول memory-optimized از طریق تراکنش های کاربری در Transact-SQL تفسیر شده، در table hint ها تعریف کنید.

READ COMMITTED برای جداول بهینه سازی شده بر اساس حافظه با تراکنش های autocommit نیز قابل استفاده می باشد. READ COMMITTED در تراکنش های کاربری و یا در بلاک های تجزیه ناپذیر معتبر نمی باشد. READ COMMITTED همراه با تراکنش های کاربری صریح یا ضمنی پشتیبانی نمی شود. READ_COMMITTED_SNAPSHOT برای جداول memory-optimized که دارای تراکنش های autocommit می باشند و همچنین تنها در صورتی که query به هیچ جدول ذخیره شده بر روی دیسک دسترسی نداشته باشد، پشتیبانی شده و قابل استفاده می باشد. علاوه بر این تراکنش هایی که به واسطه ی Transact-SQL تفسیر شده راه اندازی می شود و isolation level آن بر روی SNAPSHOT تنظیم شده، قادر به دسترسی به جداول memory-optimized نیستند. تراکنش هایی که از طریق Transact-SQL تفسیر شده و با سطوح REPEATABLE READ و یا SERIALIZABLE مورد دسترسی قرار می گیرند، بایستی برای دسترسی به جداول بهینه سازی شده بر اساس حافظه از SNAPSHOT بهره بگیرند.

همان طور که پیش از این گفته شد، READ COMMITTED سطح پیش فرض در SQL Server محسوب می شود. هنگامی که isolation level در session مربوطه بر روی READ COMMITED یا حتی پایین تر تنظیم شده باشد، شما می توانید یکی از دو راه زیر را پیش بگیرید:

1. یا به صورت صریح یک isolation level hint بالاتر را (برای مثال WITH (SNAPSHOT)) را به منظور دسترسی به جداول بهینه سازی شده بر اساس حافظه مورد استفاده قرار دهید.
2. و یا گزینه ی MEMORY_OPTIMIZED_ELEVATE_TO_SNAPSHOT را مشخص کنید که سطح isolation را برای دسترسی به جداول memory-optimized بر روی SNAPSHOT تنظیم می کند (گویی شما hint های WITH(SNAPSHOT) را برای تمامی دیگر جداول memory-optimized لحاظ (include) می کنید). به عنوان جایگزین برای روش قبلی، اگر isolation level در session بر روی READ COMMITTED تنظیم شده باشد، می توان از تراکنش های autocommit بهره گرفت.

تراکنش های SNAPSHOT که در Transact-SQL تفسیر شده راه اندازی شده باشد، توانایی دسترسی به جداول memory-optimized را ندارند.

Isolation level تراکنش پشتیبانی شده و قابل استفاده برای جداول بهینه سازی شده بر اساس حافظه همان ضمانت منطقی (logical guarantee) را فراهم می نماید که جداول ذخیره شده بر روی دیسک ارائه می دهند. مکانیزم بکار گرفته شده برای ارائه ی ضمانت های isolation level متفاوت می باشد.

برای جداول ذخیره شده بر روی دیسک، بیشتر isolation level guarantee ها (تضمین جداسازی تراکنش ها) با بهره گیری از اعمال قفل (blocking) پیاده سازی می شوند که به کمک مسدود سازی (blocking) از ایجاد تداخل جلوگیری می کند. این در حالی است که برای جداول memory-optimized، ضمانت ها با بهره گیری از یک مکانیزم تشخیص تداخل اعمال می شوند که نیاز به اخذ یا اعمال قفل را به کلی از میان بر می دارد. در این میان یک استثنا وجود دارد و آن هم SNAPSHOT برای دسترسی به جدول ذخیره شده بر روی دیسک (disk-based) می باشد که درست به همان شیوه ای که برای جداول memory-optimized انجام می شود صورت گرفته و از طریق مکانیزم تشخیص و شناسایی تداخل پیاده می شود.

SNAPSHOT

این سطح تعیین می کند که اطلاعات خوانده شده توسط هر دستور معینی در یک تراکنش، باید بر مبنا (و از نظر) تراکنش همان نسخه ای باشد که در آغاز transaction موجود بود. تراکنش تنها می تواند اصلاحات و تغییرات وارد شده به اطلاعاتی را که پیش از شروع تراکنش تایید ثبت شده اند، بشناسد. تغییراتی که توسط دیگر تراکنش ها پس از شروع تراکنش جاری اعمال می شوند برای دستوراتی که در تراکنش مزبور (جاری) در حال اجرا هستند قابل رویت نمی باشد. دستورات موجود در تراکنش یک نسخه ی فوری (snapshot) از اطلاعات تایید ثبت شده (committed) به همان حالتی که در آغاز تراکنش بوده اند تهیه می کند.

عملیات write (همچون update، insert و delete) همیشه کاملا از دیگر تراکنش ها ایزوله یا جداسازی می شوند. از این رو، عملیات write در یک تراکنش SNAPSHOT ممکن است با دیگر عملیات write که توسط دیگر تراکنش ها اجرا می شوند، تداخل پیدا کند. هنگامی که تراکنش جاری سعی بر بروز رسانی یا حذف سطری می کند که توسط تراکنش دیگری حذف یا بروز رسانی شده و آن تراکنش پس از اینکه تراکنش جاری آغاز شد، تایید ثبت گردد، transaction مورد نظر با پیغام خطای زیر خاتمه می یابد:

Error 41302. تراکنش جاری سعی بر بروز رسانی سطری در جدول X داشته که از پیش و از ابتدای شروع تراکنش بروز رسانی شده بوده است، به این خاطر تراکنش به طور ناگهانی متوقف شد.

هنگامی که تراکنش جاری سعی بر درج یک سطر با مقدار کلید اصلی (primary) یکسان با سطری که توسط تراکنش دیگری که پیش از تراکنش جاری تایید ثبت شده می کند، commit به طور موفقیت آمیز انجام داده نخواهد شد و پیغام خطای 41325 صادر می گردد.

The current transaction failed to commit due to a serializable validation ".Error 41325 " failure.

"تراکنش جاری موفق به دلیل یک serializable validation failure موفق به تایید ثبت نشد."

چنانچه تراکنشی عملیات write بر روی یک جدول انجام دهد که پیش از تایید ثبت تراکنش حذف گردد، تراکنش گفته شده با پیغام خطای زیر خاتمه می یابد:

The current transaction failed to commit due to a repeatable read " .Error 41305 " validation failure.

"تراکنش جاری به دلیل رخداد repeatable read validation failure موفق به تایید ثبت نشد."

REPEATABLE READ

این سطح شامل ضمانت هایی (guarantee) که توسط سطح SNAPSHOT ارائه شده است نیز می شود. بعلاوه، REPEATABLE READ اطمینان کسب می کند که هر سطری که توسط تراکنش خوانده می شود، در زمانی که تراکنش سطر را تایید ثبت می کند، آن سطر توسط هیچ تراکنش دیگری تغییر نمی کند. تمامی عملیات read که طی تراکنش اجرا می شود همواره تا پایان تراکنش تکرار پذیر می باشد.

اگر تراکنش جاری سطری را خوانده که آن سطر توسط تراکنش دیگری که پیش از تراکنش جاری تایید ثبت شده، بروز رسانی شده باشد، در آن صورت commit با صدور پیغام خطای زیر با شکست مواجه می شود.

The current transaction failed to commit due to a repeatable read " .Error 41305 "validation failure.

"تراکنش جاری به دلیل رخ دادن repeatable read validation failure موفق به تایید ثبت نشد. "

SERIALIZEABLE

این سطح دربردارنده ی ضمانت هایی که توسط REPEATABLE READ ارائه می گردد نیز می باشد. هیچ سطر فرضی (phantom row) مابین snapshot و انتهای تراکنش ظاهر نشده است. سطرهای فرضی با وضعیت فیلتر (filter condition) یک select، update یا delete مطابقت دارد.

پتراکنش مورد نظر طبق روال خود اجرا می شود گویی هیچ تراکنش همروندی وجود ندارد. تمامی عملیات تقریبا به طور همزمان و درست در زمان تایید ثبت (serialization point واحد) رخ می دهند.

در صورت نقض هر یک از این ضمانت ها (guarantee)، تراکنش با صدور پیغام خطای زیر موفق به تایید ثبت نمی شود:

Error 41325. " The current transaction failed to commit due to a serializable validation failure".

"تراکنش فعلی به دلیل serializable validation failure موفق به تایید ثبت نشده است."