Administracja i programowanie pod Microsoft SQL Server 2000

Paweł Rajba

pawel@ii.uni.wroc.pl
http://www.kursy24.eu/

Zawartość modułu 8

- Transakcje i blokady
  - Wprowadzenie do transakcji, rodzaje transakcji
  - Punkty zapisu, odzyskiwanie i punkty kontrolne
  - Zagnieżdżanie i niejawne zaczynanie transakcji
  - Zabronione instrukcje, rozważanie o transakcjach
  - Błędy w transakcjach
  - Zasoby do blokowania, rodzaje i zgodność blokad
  - Poziomy odseparowania, czas oczekiwania
  - Zakleszczenia, informacje o blokadach

Wprowadzenie

- Kilka uwag
  - Jedna transakcja może zawierać wiele wsadów, jeden wsad może zawierać wiele transakcji
  - Transakcje spełniają własność ACID:
    - niepodzielność
    - spójność
    - odseparowanie
    - wytrzymałość
  - W ramach transakcji zwykle umieszcza się instrukcje modyfikujące dane

Rodzaje transakcji

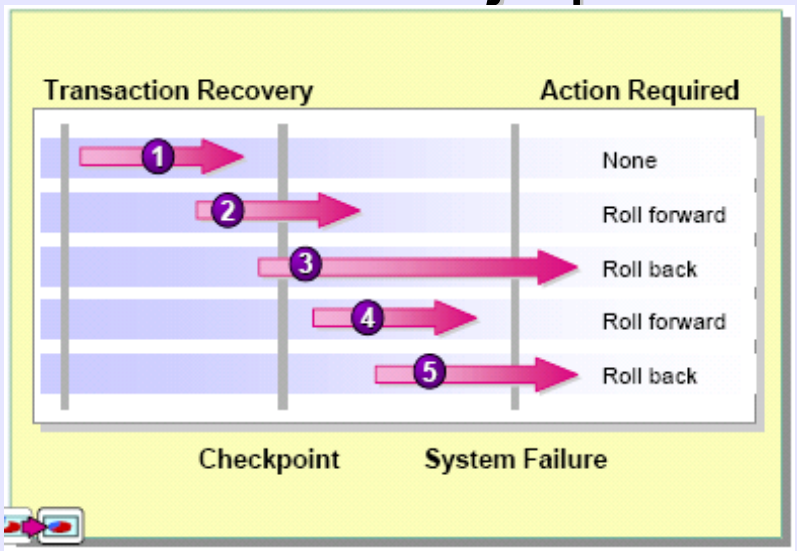
- Są dwa rodzaje transakcji w SQL Server
  - domniemane (implicit) dla poleceń INSERT, ...
  - na żądanie (explicit)
    - transakcję rozpoczynamy poleceniem BEGIN TRANSACTION [WITH MARK ...]
    - transakcję kończymy poleceniami COMMIT lub ROLLBACK

Punkty zapisu

- Są po to, żeby można było wycofać fragment transakcji, a nie wszystko
- Tworzenie punktu zapisu odbywa się poprzez polecenie
  - SAVE TRANSACTION nazwa
- Wycofanie do podane punktu odbywa się poprzez polecenie
  - ROLLBACK TRANSACTION nazwa_punktu_zapisu

Odzyskiwanie i punkty kontrolne

- Punkt kontrolny
  - zmiany danych wykonywane w ramach transakcji są wprowadzane do bazy danych w momencie wykonania tzw. punktu kontrolnego
  - znaczenie i skutki ilustruje poniższy rysunek



Zagnieżdżanie transakcji

- SQLServer pozwala na zagnieżdżanie instrukcji BEGIN TRANSACTION/COMMIT(ROLLBACK)
- Stopień zagnieżdżenia jest określony przez zmienną @@trancount (jeśli brak aktywnej transakcji – zmienna ma wartość 0)
- Uwagi:
  - Jeśli stopień zagnieżdżenia jest równy 1, wtedy zatwierdzane są transakcji ze wszystkich poziomów
  - Operację wycofania możemy wykonać na dowolnym poziomie zagnieżdżenia, jednak wycofywane są wszystkie transakcje

Zagnieżdżanie transakcji

- Inaczej mówiąc:
  - BEGIN TRAN – zwiększa @@trancount o 1
  - COMMIT – zmniejsza @@trancount o 1
  - ROLLBACK – ustawia @@trancount na 0
- Jeszcze jedna uwaga
  - Wywołanie COMMIT lub ROLLBACK w przypadku, gdy nie ma aktywnej transakcji, skończy sie błędem

Zagnieżdżanie transakcji

- Nazewnictwo transakcji
  - Warto nazywać transakcję, ponieważ możemy lepiej panować nad kodem, jednak należy uważać:
    - jeśli wykonujemy ROLLBACK, to podana nazwa musi być nazwą transakcji z poziomu 1
    - wykonując COMMIT, podana nazwa nie ma znaczenia, ponieważ wykonywany tak naprawdę jest i tak COMMIT na poziomie 1
- UWAGA:
  - zagnieżdżonych transakcji należy w miarę możliwości unikać (mogą się pojawić kłopoty z blokadami)

Niejawne zaczynanie transakcji

- Opcja Implicit Transactions
  - jej włączenie sprawia, że określone instrukcje automatycznie rozpoczynają transakcję
  - w takiej konfiguracji nie są dozwolone transakcje zagnieżdżone
  - transakcja musi być jawnie zakończona poleceniem COMMIT lub ROLLBACK
  - domyślnie opcja ta jest wyłączona
- Ustawienie opcji
  - SET IMPLICIT_TRANSACTIONS {ON | OFF}

Niejawne zaczynanie transakcji

- Polecenia, które przy włączeniu opcji implicit transactions rozpoczną transakcję:
  - ALTER TABLE, INSERT
  - CREATE, OPEN
  - DELETE, REVOKE
  - DROP, SELECT
  - FETCH, TRUNCATE TABLE
  - GRANT, UPDATE

Zabronione instrukcje

- Poniższych instrukcji nie można umieszczać w transakcjach:
  - ALTER DATABASE
  - BACKUP LOG
  - CREATE DATABASE
  - DROP DATABASE
  - RECONFIGURE
  - RESTORE DATABASE
  - RESTORE LOG
  - UPDATE STATISTICS

Rozważania o transakcjach

- Czas wykonania powinien być jak najkrótszy
  - należy przemyśleć użycie np. pętli while
  - czas blokad jest krótszy
- Nie należy w ramach transakcji czekać na dane od użytkownika
- Wszystkie niezbędne analizy danych należy wykonać przed rozpoczęciem transakcji
- Transakcja powinna uzyskiwać dostęp do minimalnego zbioru wierszy

Przykład

- sqlserver-p08-01.sql

Błędy w transakcjach

- Kilka uwag
  - pojawienie się błędu nie zawsze powoduje automatyczne wycofanie transakcji
  - błędy krytyczne powodują przerwanie wykonywania wsadu i wycofanie transakcji
  - warto sprawdzać zmienną @@ERROR po wykonaniu każdej instrukcji
  - błędy składniowe spowodują wstrzymanie wykonywania wsadu, odwołania do nieistniejących obiektów spowodują przerwanie wsadu w miejscu odwołania

Błędy w transakcjach

- Błędy, które najczęściej się pojawiają są zwykle związane z zasobami
- Należy zwrócić szczególną uwagę na
  - brak uprawnień do obiektu
  - naruszenie ograniczeń
  - powtórzenie wartości przy aktualizacji i wstawianiu
  - zakleszczenie z innym użytkownikiem
  - niedozwolone wartości dla bieżącego typu danych

Błędy w transakcjach

- Przydatna konstrukcja
  - declare @retcode int
    exec @retcode=procedura
  - dzięki tej konstrukcji mamy lepszą kontrolę nad błędami
- Opcja XACT_ABORT
  - powoduje przerwanie wsadu przy każdym błędzie
  - ustawienie przez polecenie
    - set xact_abort { on | off }
  - skutki jej włączenia są czasami problematyczne

Błędy w transakcjach

- Obsługa błędów w T-SQLu jest dosyć uciążliwa, nieelegancka i niespójna
  - nie można utworzyć podprogramu odpowiedzialnego za obsługę błędów
  - zwykle błędy obsługuje się w sposób podobny do przedstawionego w przykładach (bazuje na goto)

Błędy w transakcjach

- Obsługa błędów w T-SQLu jest dosyć uciążliwa, nieelegancka i niespójna
  - nie ma prostego kryterium pozwalającego ustalić, które błędy są krytyczne, a które nie
    - można pobrać kod błędu i ustalić wagę błędu na podstawie tabeli sysmessages
    - kłopot w tym, że niektóre błędy o wadze 16 nie są krytyczne (np. 515, 544, 547, 550), a niektóre o wadze 15 są krytyczne (np. błąd składniowy polegający na odwołaniu się do nieistniejącej funkcji)
      :)

Przykład

- sqlserver-p08-02.sql

Blokady

- Blokady zapobiegają następującym zjawiskom
  - utracona zmiana
    - dwóch edytuje wiersz, zapisują zmiany; przepadają zmiany tego, który zapisał jako pierwszy
  - brudne dane (dirty read)
    - edycja danych, które nie zostały zatwierdzone, po czym zostały wycofane
  - powtarzalność odczytu (nonrepeatable read)
    - czytamy dane, ktoś je modyfikuje, czytamy dane powtórnie; wyniki obliczeń wykonane za pierwszym i drugim razem mogą być inne co prowadzi do niespójności

Blokady

- Blokady zapobiegają następującym zjawiskom
  - odczyty fantomów
    - pojawiają się, gdy transakcje nie są od siebie odizolowane
    - w jednej transakcji modyfikowane są wszystkie pozycje dla danego regionu, w drugiej w tym samym momencie dodawana jest jeszcze jedna pozycja dla tego regionu; przy ponownym czytaniu wierszy przez pierwszą transakcję ten jeden wiersz nie jest zmodyfikowany

Zasoby do blokowania

- Blokowane mogą być
  - RID – identyfikator wiersza, czyli cały wiersz w tabeli
  - Key – wiersz w indeksie, używane do blokady zakresu wierszy
  - Page – 8KB strona
  - Extent – obszar stron tabeli lub indeksu
  - Table – cała tabela razem z indeksami
  - Database – cała baza danych, używane przy odtwarzaniu bazy danych

Rodzaje blokad

- Blokady podstawowe
  - Dzielone (shared)
    - są nakładane przy wykonywaniu operacji tylko-do-odczytu
    - SQL Server zwalnia taką blokadę na wiersz jak tylko przejdzie do odczytu kolejnego wiersza, chyba że jest ustawiony wyższy poziom izolacji transakcji wtedy np.
    - blokada będzie istniała dopóki wszystkie wiersze wyniku nie zostaną przekazane do klienta

Rodzaje blokad

- Blokady podstawowe
  - Wyłączne (exclusive)
    - są nakładana przy operacjach INSERT, UPDATE, DELETE
    - nałożenie tej blokady uniemożliwia nałożenie blokad dzielonych
    - blokada może zostać nałożona dopiero, gdy na zasobie nie jest nałożona żadna inna blokada
      - w szczególności tylko jedna transakcja może nałożyć taką blokadę

Rodzaje blokad

- Blokady specjalne
  - Intent locks
    - są to blokady intencyjne, które określają co SQL Server zamierza zrobić; służą do minimalizacji konfliktów blokad
    - np. transakcja ma wyłączną blokadę na wiersz, intent lock uniemożliwia innej transakcji blokadę całej tabeli
    - rodzaje takich blokad
      - IS – transakcja zamierza czytać część zasobów przez nałożenie na nie blokady S
      - IX – transakcja zamierza modyfikować część zasobów (nie wszystkie) poprzez nałożenie na nie blokady X
      - SIX – transakcja zamierza czytać wszystkie wiersze i niektóre z nich modyfikować (poprzez nakładanie odpowiednich blokad)

Rodzaje blokad

- Blokady specjalne
  - Update locks
    - nakładana na dane, które mogą być aktualizowane
    - rozważmy sytuację, gdy dwie transakcje nałożyły blokadę S na wiersz a następnie obie chcą go zmodyfikować; będą wtedy czekać na założenie blokady X w nieskończoność
    - blokada typu U zapobiega takiej sytuacji, ponieważ wtedy tylko jedna z transakcji może nałożyć taką blokadę
      - jeśli dane są modyfikowane, blokada jest zamieniana na X
    - tego typu blokady zapobiegają zakleszczeniom, gdy wiele transakcji czyta dane z zamiarem ich aktualizacji

Rodzaje blokad

- Blokady specjalne
  - Schema locks
    - blokują możliwość zmiany lub usunięcia schematu tabeli lub indeksu
    - są dwa rodzaje takiej blokady
      - Schema stability (Sch-S) – blokuje możliwość usunięcia zasobu
      - Schema modification (Sch-M) – blokuje innym sesjom możliwość dostępu do zasobu, który jest aktualnie modyfikowany
  - Bulk update locks
    - blokada umożliwia procesom wykonanie operacji bulk copy, blokując jednocześnie dostęp do tabeli procesom, które chcą na niej wykonać coś innego

Kompatybilność blokad

- Na zasób jest nałożona blokada. Kombatybilność określa, która inna blokada może być założona dodatkowo
- Zasady określa poniższa tabelka

	Existing granted lock					
Requested lock	**IS**	**S**	**U**	**IX**	**SIX**	**X**
Intent shared (IS)	Yes	Yes	Yes	Yes	Yes	No
Shared (S)	Yes	Yes	Yes	No	No	No
Update (U)	Yes	Yes	No	No	No	No
Intent exclusive (IX)	Yes	No	No	Yes	No	No
Shared with intent exclusive (SIX)	Yes	No	No	No	No	No
Exclusive (X)	No	No	No	No	No	No

Poziomy odseparowania

- READ UNCOMMITED
  - nie są nakładane żadne blokady
  - mogą się pojawić ,,brudne" odczyty
- READ COMMITED
  - nakładane są dzielone blokady przy czytaniu
  - ,,brudne" odczyty już nie pojawią

Poziomy odseparowania

- REPEATABLE READ
  - gwarantowany jest brak ,,brudnych" odczytów i niepowtarzalnych odczytów
  - blokady odczytu są utrzymywane do końca transakcji
- SERIALIZABLE
  - zapobiega modyfikacji i wstawianiu wierszy, które spełniają warunek WHERE wierszy biorących udział w transakcji
  - nie pojawią fantomy

Poziomy odseparowania

- Do ustawienia poziomu odseparowania służy polecenie
  - SET TRANSACTION ISOLATION LEVEL
    { READ COMMITTED | READ UNCOMMITTED |
    REPEATABLE READ | SERIALIZABLE }
- Poziom odseparowania można sprawdzić przez polecenie DBCC USEROPTIONS

Czas oczekiwania na zasób

- Opcja lock timeout
  - określa ile milisekund transakcja będzie czekała na dostęp do zasobu, zanim zostanie zwrócony błąd locking error
    - powyższy błąd nie powoduje przerwania transakcji
  - domyślna wartość to -1, czyli transakcja będzie czekać w nieskończoność
  - ustawienie opcji
    - SET LOCK_TIMEOUT milisekundy
  - ustalenie aktualnej wartości
    - SELECT @@lock_timeout

Zakleszczenia

- Pojawiają się, gdy dwie transakcje wzajemnie czekają na blokowane przez siebie zasoby
- SQL Server kończy zakleszczenie poprzez wyznaczenie ,,ofiary zakleszczenia", czyli proces, który po prostu ubija; zwykle jest to ten młodszy proces
  - pojawia się wtedy komunikat o numerze 1205

Zakleszczenia

- Porady na zmniejszenie ryzyka powstania zakleszczenia
  - używać zasobów w transakcjach w tej samej kolejności
  - transakcje powinny być jak najkrótsze
    - czasowo
    - pod względem ilości kroków

Informacje o blokadach

- Listę blokad możemy zobaczyć
  - Enterprise Manager | Management | Current Activity
  - jako wynik procedury sp_lock
- Znaczenie niektórych pól wyniku
  - Type – określa rodzaj blokowanego zasobu
    - DB – baza danych, EXT – obszar (extent)
    - TAB – tabela (table),
    - KEY – klucz w indeksie (key),
    - PAG – stronę (page),
    - RID – wiersz (row identifier).

Informacje o blokadach

- Znaczenie niektórych pól wyniku
  - Resource – adres blokowanego zasobu
    - Przykładowo: 1:528:0 – wiersz 0, na stronie 528 i pliku 1
  - Mode – tryb blokowania, który składa się z
    - shared (S), exclusive (X), intent (I),
    - update (U), or schema (Sch).
  - Status
    - GRANT – blokada uzyskana
    - WAIT – oczekiwana
    - CNVRT – w trakcie zmiany

Przykłady

- sqlserver-p08-03.sql
- sqlserver-p08-04.sql
- sqlserver-p08-05.sql

