Rozproszona baza danych

elementy bazy danych są zintegrowane za pomocą systemu zarządzania rozproszoną bazą danych, który jest oprogramowaniem umożliwiającym połączenie autonomicznych baz danych w jedną całość, utrzymanie spójności danych, oraz udostępnianie ich użytkownikom przy założeniu przeźroczystości rozproszenia.

Transakcje rozproszone

Transakcja rozproszona to transakcja, której polecenia DML (insert, update, delete) i select odwołują się do tabel znajdujących się co najmniej w dwóch węzłach rozproszonej bazy danych. Transakcja rozproszona jest często nazywana transakcją globalną.

• Transakcja rozproszona jest reprezentowana przez zbiór transakcji lokalnych

• W każdej z baz danych, do której odwołuje się transakcja rozproszona tworzona jest jedna transakcja lokalna

Graf sesji składa się z 6 węzłów baz danych BD1 do BD6, które są adresowane przez transakcję rozproszoną. Łuki grafu stanowią żądania wykonania poleceń z jednej bazy w innej.

W systemach wielodostępnych wielu użytkowników może wykonywać współbieżnie różne operacje, które mogą równocześnie korzystać z tych samych danych (zasobów).

Współbieżne wykonywanie programów użytkowników jest istotne dla wydajności działania aplikacji wykorzystujących bazę danych. Dostęp do danych na dysku jest częsty i dość wolny, dlatego procesor może jednocześnie wykonywać wiele różnych programów realizujących operacje na bazie danych – współdzielonych tabelach i rekordach.

Przetwarzanie współbieżne w systemach baz danych zwiększa wydajność, ale wymaga odpowiednich mechanizmów synchronizacji dostępu procesów do wspólnych danych.

Jeśli jednak użytkownicy próbowaliby jednocześnie (współbieżnie) modyfikować (odczytywać i zapisywać) te same dane, wynik działania różnych operacji na bazie danych mógłby doprowadzić do błędnych wyników, w zależności od kolejności wykonywania (uszeregowania) operacji wykonywanych na bazie danych.

W celu zapewnienia spójności (poprawności) danych w czasie próby równoczesnego ich modyfikowania wprowadzono mechanizm transakcji.

Cechy transakcji

Atomowość-Jedną z własności transakcji jest atomowość, gwarantująca wykonanie wszystkich operacji składowych transakcji w sposób niepodzielny. Atomowa realizacja zbioru operacji modyfikujących na wielu różnych serwerach, które dodatkowo mogą ulegać awarii, jest zadaniem bardzo trudnym, a często wręcz niemożliwym do zrealizowania. Pojawiają się również problemy ze skalowalnością transakcji. zbiór operacji składowych transakcji musi być wykonany w całości albo wcale; każda transakcja powinna być z punktu widzenia użytkownika niepodzielną operacją.

• Spójność (ang. consistency) – transakcja powinna przekształcać system z jednego stanu spójnego w inny spójny stan.

• Izolacja (ang. isolation) – transakcje nie powinny sobie wzajemnie przeszkadzać w działaniu; użytkownicy powinni mieć wrażenie, że każda transakcja jest wykonywana niezależnie od innych działających współbieżnie transakcji, przetwarzanych w tym samym środowisku; skutki współbieżnego wykonania transakcji powinny być takie same, jak wówczas gdyby wykonywano je w sposób szeregowy.

• Trwałość (ang. durability) – wyniki działania zatwierdzonych transakcji powinny być zachowane w pamięci trwałej.

2pc protocol

Głównym problemem jest zagwarantowanie atomowości transakcji rozproszonej, gdyż standardowy mechanizm zatwierdzania transakcji nie gwarantuje jej atomowości (ze względu na możliwy długi czas trwania, wynikający z czasu transmisji oraz zawodności łączy).

Wysokie prawdopodobieństwo awarii w fazie zatwierdzania w systemach rozproszonych baz danych wymusiło powstanie nowego protokołu synchronizacji zmian w transakcjach rozproszonych.

W tym przypadku zatwierdzanie lub wycofywanie transakcji rozproszonej, gwarantujące atomowość jest realizowane za pomocą specjalizowanego mechanizmu, tzw. protokołu zatwierdzania dwu-fazowego - 2PC (ang. twophase commit), który rozpoczyna się z chwilą rozpoczęcia zatwierdzenia transakcji (wykonanie instrukcji commit przez klienta).

Każda z trzech baz danych BD1, BD2, BD3 posiada swój własny moduł lokalnego menadżera transakcji (lokalny MT), identycznie jak w standardowej scentralizowanej bazie danych. Ponadto, do zarządzania transakcjami rozproszonymi jest niezbędny moduł globalnego menadżera transakcji (globalny MT). Jego zadaniem jest koordynowanie wykonania zarówno lokalnych jak i rozproszonych transakcji zainicjowanych w jego węźle.

Poszczególne węzły realizujące transakcję rozproszoną komunikują się za pośrednictwem modułu komunikacji, istniejącego w każdym węźle.

Protokół zatwierdzania 2-fazowego 2PC w systemie Oracle składa się z dwóch głównych faz przetwarzania:

- przygotowania, zwanej również głosowaniem (ang. voting phase),

- decyzji (ang. decision phase), obejmującej: zatwierdzanie(commit) lub wycofanie (abort/rollback), oraz zakończenie (forget).

W fazie przygotowania następuje przygotowanie transakcji lokalnych do zatwierdzania lub wycofywania.

W fazie decyzji (zatwierdzania) następuje zatwierdzanie lub wycofywanie transakcji lokalnych.

W fazie zakończenia następuje usuwanie z systemu informacji o transakcji rozproszonej i zwalnianie niezwolnionych w fazie zatwierdzania zasobów.

Zatwierdzanie lub wycofywanie transakcji rozproszonej, gwarantujące atomowość jest realizowane za pomocą tzw. protokołu zatwierdzania dwufazowego — 2PC (ang. two-phase commit).

Protokół 2PC może być implementowany w jednym z trzech wariantów:

• scentralizowanego 2PC,

• zdecentralizowanego 2PC,

• liniowego 2PC.

Uczestnikami protokołu są:

• Koordynator globalny (KG) – węzeł sieci, w którym zainicjowano tranaskcję rozproszoną

• Koordynator lokalny (KL) – węzeł sieci, któremu podlegają inne węzły

• Uczestnik (U) – węzeł sieci z transakcją lokalną

Zadaniem koordynatora globalnego jest zarządzanie całą transakcją, tj. doprowadzenie jej w całości do zatwierdzenia lub wycofania

Koordynator lokalny (ang. local coordinator) jest węzłem który otrzymuje żądanie, przetwarza je i wysyła kolejne żądania do innych podległych mu węzłów. KL nie koordynuje całej transakcji globalnej a jedynie te żądania, które sam wysłał. Uczestnik (ang. node) jest węzłem, do którego jest kierowane żądanie transakcji rozproszonej, a sam węzeł nie wysyła żadnych żądań.

SCENTRALIZOWANY 2PC:

W fazie przygotowania, koordynator globalny wykonuje następujące czynności:

1. Koordynator globalny wybiera węzeł zatwierdzania spośród spośród wszystkich węzłów biorących udział w transakcji rozproszonej.

2. KG wysyła do wszystkich węzłów z wyjątkiem węzła zatwierdzania komunikat PREPARE, wymuszający przygotowanie się transakcji lokalnych do zatwierdzenia.

3. Po zakończeniu operacji przygotowania do zatwierdzenia transakcji, każdy węzeł przesyła do koordynatora globalnego komunikat PREPARED jeżeli się przygotował do zatwierdzenia swojej transakcji lokalnej.

4. W ostatnim kroku tej fazy koordynator globalny odbiera komunikaty od wszystkich węzłów, z wyjątkiem węzła zatwierdzania. Po ich odebraniu następuje przejście do drugiej fazy, tj. zatwierdzania.

W fazie **przygotowania** w węźle uczestnika są wykonywane następujące operacje:

1. Odbiór komunikatu PREPARE od koordynatora globalnego.

2. Dokonanie zapisów w bieżących plikach dziennika powtórzeń (ang. online redo logs). W plikach tych znajdują się wszystkie zmiany wykonane w bazie danych (np. wstawienie, zmodyfikowanie, usunięcie rekordu) wprowadzone zarówno przez zatwierdzone, jak i niezatwierdzone transakcje. Informacje z plików dziennika powtórzeń są wykorzystywane w czasie odtwarzania bazy danych po awarii.

3. Wysłanie komunikatu PREPARE do węzłów podległych.

4. Odbiór komunikatów od węzłów podległych.

5. Wysłanie komunikatu do koordynatora globalnego.

Jeżeli w węźle uczestnika i w żadnym z węzłów mu podległych nie dokonano modyfikacji danych, wówczas uczestnik wysyła do koordynatora globalnego komunikat READ-ONLY. Jeśli sam uczestnik i wszystkie węzły mu podległe przygotowały się do zatwierdzania, uczestnik wysyła do koordynatora globalnego komunikat PREPARED. Jeśli natomiast albo uczestnik albo przynajmniej jeden z węzłów mu podległych nie mogły się przygotować, wówczas uczestnik wycofuje swoją transakcję lokalną i wysyła do koordynatora globalnego komunikat ABORT.

W fazie zatwierdzania są realizowane następujące kroki prowadzące do zatwierdzenia transakcji.

1. Koordynator globalny odbiera potwierdzenia od uczestników.
   2. Jeśli wszyscy odpowiedzieli komunikatem PREPARED, wówczas KG wysyła do węzła zatwierdzania komunikat COMMIT, czyli żądanie zatwierdzenia transakcji rozproszonej.
   3. Węzeł zatwierdzania zatwierdza transakcję i wysyła do KG komunikat potwierdzający zatwierdzenie (COMMITTED).
   4. Po otrzymaniu od WZ komunikatu COMMITTED, KG wysyła komunikat COMMIT do pozostałych węzłów (uczestników).

Zdecentralizowany:

• Uczestnicy przygotowują się w sposób identyczny do omówionego wcześniej, ale komunikat PREPARED (READY_COMMIT) bądź ABORT wysyłają do wszystkich węzłów.

• W ten sposób, każdy węzeł ma pełen obraz stanu transakcji rozproszonej i sam może podjąć decyzję o zatwierdzeniu lub wycofaniu transakcji lokalnej.

Cechy zdecentralizowanego protokołu 2PC:

- większy ruch sieciowy ze względu na większą liczbę przesyłanych komunikatów,

- brak fazy decyzji, gdyż każdy uczestnik zna odpowiedź pozostałych uczestników i na jej podstawie może podjąć decyzję o zatwierdzeniu lub wycofaniu swojej transakcji lokalnej.

W architekturze liniowego 2PC węzły są uporządkowane liniowo i każdy z nich otrzymuje numer.

Węzeł KG otrzymuje numer 1.

Uczestnicy otrzymują kolejne numery, 2, 3, do n.

W czasie wymiany komunikatów, każdy węzeł o numerze (i) komunikuje się tylko z węzłami sąsiednimi, tj. węzłem bezpośrednio go poprzedzającym (numer i -1) i węzłem bezpośrednio po nim następującym (numer i+1). ostatni węzeł w łańcuchu podejmuje decyzję o zatwierdzeniu/wycofaniu na podstawie zawartości komunikatu od węzła n-1 • komunikat ten zawiera decyzje wszystkich wcześniejszych węzłów

ROZPOCZĘCIE TRANSAKCJI:

Domyślnie w Oracle ustawiony jest tryb autocommit (on), w którym każde polecenie DML (insert, update, delete, create) i polecenia z grupy DCL (grant, revoke) kończą się poprzez niejawne zatwierdzenie poleceniem commit.

Transakcja może być jawnie wycofana poleceniem rollback.

Zakończenie jednej transakcji jest początkiem transakcji następnej.

Zakończenie transakcji

Transakcja może zakończyć swoje działanie na cztery sposoby:

• zatwierdzeniem (COMMIT) po zakończeniu wszystkich swoich akcji,

• samo-wycofaniem (ROLLBACK) - anulowaniem wszystkich wykonanych akcji, może zostać przerwana przez SZBD („aportowana”, ang. abort), a następnie wycofana i restartowana (np. z powodu zakleszczenia - ang. deadlock lub na skutek błędnego wykonania pojedynczej akcji),

• zanim dojdzie do końca może zostać przerwana z powodu awarii serwera, lub dysku - po podniesieniu systemu po awarii dotychczasowe zmiany wprowadzone przez transakcję zostają na chwilę przywrócone a następnie wycofane przez system (ponieważ transakcja nie zakończyła się zatwierdzeniem).

W każdej z tych sytuacji albo cała transakcja zostaje wykonana albo żadna jej część nie zostaje wykonana. Transakcję można traktować jak pojedynczą, atomową operację na bazie danych.

Poniższy przykład (rollback) pokazuje efekt wycofania transakcji. W tym przykładzie instrukcja ROLLBACK spowoduje wycofanie instrukcji INSERT, ale utworzona tabela będzie nadal istnieć.

PRZERWANIA

W czasie fazy COMMIT (ROLLBACK) następuje awaria sieci, węzła lub zdalnej bazy danych

– nie wszystkie węzły zatwierdziły (wycofały)

– nie wszystkie węzły potwierdziły zakończenie operacji – transakcja rozproszona w stanie “in-doubt”

• Automatyczne odtwarzanie transakcji rozproszonej (proces RECO) w stanie “in-doubt” po usunięciu awarii – wynik: wszystkie węzły zatwierdzą lub wszystkie wycofają

Jeżeli awaria któregoś z węzłów nastąpi w czasie realizowania transakcji rozproszonej, wówczas taka transakcja będzie w tzw. stanie zawieszenia (ang. in-doubt) do momentu usunięcia awarii, nawiązania połączenia ze zdalnym węzłem i doprowadzenia transakcji do końca. Doprowadzenie do końca transakcji rozproszonej będącej w stanie zawieszenie będziemy nazywać odtworzeniem transakcji. Transakcję w stanie zawieszenia można odtworzyć na dwa sposoby: automatycznie - uruchamiając dedykowany do tego celu proces systemowy o nazwie RECO, lub "ręcznie" - przeszukując węzły i zatwierdzając lub wycofując ich lokalne transakcje.

Transakcję w stanie "in-doubt" należy odtworzyć (zatwierdzić lub wycofać) "ręcznie" w przypadku gdy:

- dane zablokowane przez taką tranaskcję muszą być natychmiast zwolnione,

-czas usunięcia awarii sprzętowej i możliwość uruchomienia procesu automatycznego odtwarzania transakcji są bardzo długie, -nie działa proces automatycznego odtwarzania.

Rozproszone zarządzanie transakcjami

Do zarządzania współbieżnymi transakcjami użytkowników SZBD stosuje następujące mechanizmy służące do zapewnienia aksjomatów ACID:

• blokady (zamki) (ang. lock) zakładane na obiekty – ograniczające albo wręcz uniemożliwiające działanie innych transakcji na zablokowanym obiekcie,

• dziennik (ang. log) - zapisywanie wszystkich zmian w bazie danych do specjalnego dziennika (logu), aby w razie potrzeby móc: dla nie zatwierdzonej transakcji wycofać wprowadzone przez nią zmiany, w przypadku awarii odtworzyć spójny stan bazy danych.

• kopia zabezpieczająca bazy danych (backup) - wykonywana w regularnych odstępach czasu, na przykład raz na dzień; w przypadku awarii danych na dysku pozwala przywrócić spójny stan bazy danych.

• wielowersyjność - możliwość odczytywania danych zmienianych równocześnie przez inne transakcje w takiej postaci, w jakiej istniały w pewnych chwilach w przeszłości (np. w chwili rozpoczynania się danego zapytania lub danej transakcji).

Plan wykonania transakcji

SZBD stosując pewne reguły blokowania danych i protokoły obsługi transakcji na bieżąco podejmuje decyzje o tym akcja, której transakcji ma być wykonana, jako następna - dynamicznie tworząc pewien plan wykonywania akcji zbioru współbieżnych transakcji

Plan szeregowy jest to plan, ustawiający wykonywanie transakcji w ciąg: najpierw akcje jednej transakcji, następnie akcje drugiej transakcji itd.

Dwa plany są równoważne, jeśli efekt realizacji obu planów jest taki sam dla każdego stanu bazy danych tzn. po realizacji każdego z planów, zaczynając z tego samego stanu bazy, otrzymujemy ten sam stan bazy danych.

Plan, który umożliwia wycofanie każdej transakcji nazywa się planem odtwarzalnym. Plan odtwarzalny jest istotny do zapewnienia własności atomowości i trwałości.

Można zauważyć, że jesteśmy w stanie wycofać transakcję T, jeśli w danym planie nie ma zjawisk nie zatwierdzonego odczytu ani nadpisania niezatwierdzonych danych, gdyż wówczas żadna inna transakcja nie korzysta z niezatwierdzonych zmian transakcji T i dlatego transakcję T można wycofać.

Jeśli jakaś transakcja skorzystała z wyników transakcji T, to tę transakcję też trzeba wycofać, co może być już niemożliwe, jeśli ta transakcja została wcześniej zatwierdzona.

W celu rozwiązania problemu generowania i realizacji planów (przeplotów akcji transakcji) wyłącznie szeregowalnych i odtwarzalnych wprowadzono w systemach scentralizowanych baz danych mechanizm blokowania obiektów (danych) oraz odpowiednie protokoły blokowania dwufazowego 2PL (ang. two-phase locking) (Strict 2PL i 2PL).

Instrukcje te stanowią pewną niepodzielną jednostkę i powinny być wykonane w całości. Z punktu widzenia użytkownika każda transakcja jest operacją niepodzielną.

W przypadku niepowodzenia, choć jednej instrukcji wchodzącej w skład transakcji, musi ona zostać wycofana (tj. w momencie wystąpienia błędu w ramach danej instrukcji system zarządzania bazą danych wycofuje transakcję, czyli następuje wycofanie wszystkich modyfikacji dokonanych wcześniej przez operacje wykonane w ramach transakcji).

Transakcje mogą być wykonywane współbieżnie i mogą korzystać ze wspólnych danych (zasobów).

Ponieważ operacje zapisu i odczytu różnych transakcji mogą być ze sobą przeplatane w sytuacjach konfliktowych (równoczesnego dostępu do wspólnych zasobów) system zarządzania bazą danych steruje kolejnością wykonywania operacji (szereguje operacje) różnych transakcji w taki sposób, aby nie powstawały niespójności danych.

Z punktu widzenia użytkownika transakcje powinny być widziane, jako pojedyncze operacje, które są wykonywane w całości.

Współbieżna realizacja transakcji, dopuszcza przeplot ich operacji (akcji). Przyjmuje się, że współbieżne wykonanie transakcji jest poprawne, jeśli w jego wyniku otrzymamy wynik równoważny pewnemu szeregowemu wykonaniu transakcji w tym przypadku T1, T2

Współbieżną realizację transakcji można ograniczyć, wprowadzając blokady zasobów wykorzystywanych przez transakcję. W ten sposób można zapewnić ich izolację oraz równoważność współbieżnego wykonania transakcji z wykonaniem szeregowym, a więc wymusić realizację tylko planów szeregowalnych.

Problem dostepnu do wspolbieznych danych:

(reguła wszystko albo nic – każda operacja wykonywana w ramach transakcji blokuje wszystkie potrzebne jej zasoby (np. tabelę) na czas trwania transakcji;

Można ustawić taki tryb obsługi (izolacji) transakcji, aby wszystkie operacje objęte transakcją wykonały się po kolei i nie były przeplatane z operacjami innych transakcji (np. tryb izolacji serializable).

W trakcie wykonywania transakcja może być wycofana w dowolnym momencie (jawnie przez instrukcję rollback lub niejawnie na skutek wystąpienia błędu operacji), wówczas wszelkie wprowadzone dotąd przez nią zmiany zostaną anulowane

Tryby izolacji:

• read-committed – jest to tryb domyślny; transakcja T1 „widzi" zmiany dokonane przez transakcję T2 dopiero po ich zatwierdzeniu przez T2;

DIRTY READS – występuje, kiedy jedna transakcja zmienia dane, ale nie commituje ich (nie zatwierdza), a druga transakcja ma pozwolenie do czytania tych zmienionych danych. Występuje w tym przypadku duża możliwość, że zostaną odczytane niespójne dane.

NON-REPEATABLE READS – transakcja odczytuje zmienione dane odczytując ten sam zbiór danych w dwóch różnych momentach czasowych podczas trwania tej samej transakcji.

Dwa różne rezultaty polecenia SELECT podczas działania tej samej transakcji.

PHANTOM READS – występuje w transakcjach z predykatem WHERE x <, gdzie odczyt danych w dwóch różnych poleceniach SELECT w tej samej transakcji zwraca różną liczbę wierszy

• read-only – transakcja działa na wersji danych z momentu jej rozpoczęcia; „nie widzi" zmian w bazie dokonanych przez inne, zatwierdzone transakcje; nie może też modyfikować żadnych danych;

• serializable – transakcja, podobnie jak w przypadku read-only nie widzi zmian dokonanych przez inne, zatwierdzone transakcje; może jednak zmieniać dane, ale tylko te, które nie zostały zmienione przez inne zatwierdzone transakcje (tryb ten odpowiada szeregowemu wykonywaniu transakcji). SERIALIZABLE – blokuje zakres wierszy (LOCK) aż do momentu zakończenia transakcji

Ms SQL Server 2008+ dostarcza 5 poziomów izolacji transakcji, instrukcja SET TRANSACTION ISOLATION LEVEL LevelName (Read uncommitted, Read committed, Repeatable read, Snapshot, Serializable) ט Baza danych H2 ma 3 poziomy, instrukcja: SET LOCK_MODE (0 - Read Uncommitted, 1 - Serializable, 3 - Read Committed) ט MySQL 5.7

ma 4 poziomy izolacji: Read uncommitted, Read committed, Repeatable read, Serializable υ Oracle ma 3 poziomy izolacji: Read Committed, Serializable, Read Only

Rodzaje blokad:

Współdzielona, typu S (ang. shared lock) - daje transakcji współdzielony dostęp do zasobu, na przykład, kilka transakcji może jednocześnie odczytywać wiersze tej samej tabeli. Jeśli transakcja zakłada współdzieloną blokadę, inne transakcje też mogą założyć współdzieloną blokadę, ale nie mogą założyć blokady drugiego rodzaju, to jest wyłącznej blokady.

Wyłączna, typu X (ang. exclusive lock) - daje transakcji wyłączne prawo do wprowadzania zmian obiektu. Tylko jedna transakcja może mieć założoną wyłączną blokadę na obiekcie i w tym czasie nie może być na nim założonej żadnej innej blokady nawet współdzielonej.

Zakleszczenie (deadlock) – to kombinacja blokad, która uniemożliwia na wykonanie jednej lub wielu transakcji oraz nie może być rozwiązana bez zakończenia z zewnątrz jednej z transakcji

Zawsze blokuj zasoby w tej samej kolejności, dokładniej rób zapytania CRUD w takiej samej kolejności υ Przykład: υ Zawsze blokuj numery konta począwszy od najniższego numeru υ Zawsze blokuj tabelę Accounts przed datelą Customers