Bazy Danych

Pytania egzaminacyjne

2018/2019

Spis treści

1 Architektury systemów baz danych

2 Relacyjny model danych, normalizacja relacji

Proszę podać przykład tabeli (relacji), która jest w trzeciej postaci normalnej, ale nie jest w postaci normalnej Boyce'a Codd'a.

Dla tabeli (relacji) z podanymi zależnościami funkcyjnymi proszę powiedzieć w której postaci normalnej jest ta tabela.

Podaną tabelę należy doprowadzić do postaci normalnej Boyce'a Codd'a.

Proszę podać przykład uzasadniający denormalizację.

Proszę podać przykład tabeli, która jest w postaci normalnej Boyce'a Codd'a, a w której jest redundancja.

Proszę podać przykład tabeli, która jest w 5NF, ale jest w niej redundancja.

Proszę przedstawić algorytm doprowadzenia relacji do **postaci normalnej Boyce'a Codd'a**.

- Szukamy wszystkich nietrywialnych, pełnych zależności funkcyjnych, które naruszają warunek BCNF, tzn. lewa strona zalezności funkcyjnej nie jest nadkluczem.
- Bierzemy jedną z takich zależności funkcyjnych $A \rightarrow B$ (obojetnie którą, algorym jest niederministyczny).
- Dzielimy schemat relacji na dwa nierozłączne podzbiory: jeden zawierający wszystkie atrybuty wystepujące w zależności (*) naruszającej

BCNF, drugi zawierający atrybuty z lewej storny rozważanej zależności (*) oraz atrubuty nie wystąpujące ani z lewej ani z prawej strony tej zależności.

- Strategie stosujemy do relacji powstałych w wyniku dekompozycji do chwili, gdy wszystkie relacje są w BCNF.

3 Model ER

Proszę przedstawić przykład diagramu ER w notacji Barkera, zawierającego dwie encje i związek między nimi (związek bez własnych atrybutów). Diagram powinien być taki, że po jego transformacji do modelu relacyjnego otrzymamy trzy relacje (trzy tabele).

Proszę omówić trzy schematy transformacji hierarchii encji do modelu relacyjnego.

---

1. Utworzenie jednej tabeli ze wszystkimi atrybutami i kluczami obcymi, tj. wspólnymi i specyficznymi dla podencji.
2. Utworzenie osobej tabeli dla każdej podencji.
3. Utworzenie osobnej tabeli na atrybuty wspólne i osobnej tabeli dla każdej podencji.
   Tabele powstałe z podencji zawierają klucz podstawiowy i atrybuty specyficzne, tabela wspólna i tabele powstałe z podencji są połączone ograniczeniami referencyjnymi.

4 Transakcje

4.1 Właściwiości (ACID)

Proszę omówić własności **ACID** transakcji. W jaki sposób implementowane są własności **A i D**? Proszę podać jak wykorzystywany jest **dziennik transakcji** oraz co to jest strategia **No-Fix/No-Flush** i jak wpływa ona na sposoby odtwarzania systemu po awarii.

WŁASNOŚCI ACID

- **Atomicity** (atomowość, niepodzielność)
  - Transakcja jest niepodzielną jednostką przetwarzania, musi być wykonywana w całości lub wcale.
- **Consistency** (spójność)
  - Po wykonaniu transakcji baza danych musi być w stanie spójnym, tj. muszą zostać zachowane wszystkie więzy narzucone na dane.
- **Isolation** (separacja, izolacja)
  - Transakcja powinna wyglądać tak, jakby była wykonywana w izolacji od innych transakcji.
- **Durability** (trwałość)
  - Po zatwierdzeniu transakcji jej efekty muszą być trwałe w systemie, nawet jeśli nastąpi uszkodzenie systemu natychmiast po zatwierdzeniu.

IMPLEMENTACJA A I D

- Za odtwarzanie i w pewnym sensie za **atomowość, trwałość** oraz spójność, jest odpowiedzialny **moduł zarządzania odtwarzaniem** *(recovery-management component).*
- Modyfikacja danych następuje w buforze w pamięci RAM. Buforem zarządza specjalny menadżer (zarządca). W pewnym momencie zarządca bufora musi skopiować nową zawartość bloku z powrotem na dysk.

STRATEGIA NO-FIX/NO-FLUSH

- Stosowana w większości relacyjnych systemów baz zdanych wykorzystujących dzienniki transakcji.
- **NO-FIX**
  - Blok skopiowany do RAM **może** być skopiowany lub przeniesiony z powrotem na dysk zanim transakcja, która ten blok zmodyfikowała się skończy.
- **NO-FLUSH**

  - Na końcu transakcji **nie ma obowiązku** zsynchronizowania zmienionych przez tę transakcję bloków z dyskiem.
  - Synchronizacja może być wykonana później.
  - Zwiększa wydajność.
  - Na tradycyjnych dyskach HDD operacje kopiowania bloków mogą być grupowane.

WPŁYW NA SPOSOBY ODTWARZANIA SYSTEMU PO AWARII

- **No-Flush** oznacza, że nie mamy gwarancji, że natychmiast po zakończeniu transakcji na dysku znajdą się zmienione dane.
- Trwałość transakcji w większości systemów zapewniają **dzienniki transakcji**.
- **No-Fix** oznacza, że może się zdarzyć, że blok zmieniony przez transakcje zostanie skopiowany na dysk, zanim transakcja zakończy się.
- Synchronizacja buforów z RAM z dyskiem może być realizowane cyklicznie w ramach **punktów kontrolnych** *(control point, check point)*.
- **Punkty kontrolne** ułatwiają **odtwarzanie systemu po awarii**, kiedy dyski z danymi i z dziennikiem transakcji nie zostały uszkodzone.
- **Odtwarzanie po awarii systemu (RECOVERY)**
  - redo
  - undo
- **Odtwarzanie po awarii dysków z danymi**
  - RESTORE (przywracanie plików z kopii zapasowych)
  - RECOVERY

STOSOWANIE DZIENNIKA TRANSAKCJI

- **Dziennik transakcji** jest plikiem na dysku, zawierającym **informację o wszystkich wprowadzonych** przez transakcję **zmianach**.
- Transakcja **nie jest uznana** za zakończoną, jeśli fizycznie na dysku w pliku dziennika nie znajdą się wpisy opisujące wszystkie zmiany oraz informacja o zatwierdzeniu transakcji.

- Wpis (rekord) w dzienniku może zawierać znacznik transakcji, starą i nową wartość zmienianego elementu, informację o rodzaju operacji, może zawierać informację o tzw. operacji kompensującej. Są też wpisy dotyczące rozpoczęcia transakcji i jej zatwierdzenia, w pewnych systemach także wpisy dotyczące operacji odczytu.

- Dzięki dziennikowi można powtórzyć te operacje, których efekty nie zostały jeszcze zapisane na dysku, mimo że operacja została zatwierdzona (no-flush).

- **W przypadku odtwarzania po awarii** może być wymagane **powtórzenie** *(redo)* niektórych operacji (zatwierdzone) i **wycofanie** *(undo)* innych (niezatwierdzone).

4.2 Harmonogramy, szeregowalność konfliktowa i perspektywiczna

---

Proszę podać definicje **harmonogramu szeregowalnego, szeregowalnego konfliktowo i szeregowalnego perspektywicznie**. Jakie znaczenie w praktyce ma pojęcie **szeregowalności konfliktowej?**

HARMONOGRAM SZEREGOWALNY

- **Jeśli jego wpływ na stan bazy danych jest taki sam jak pewnego harmonogramu szeregowego, niezależnie od stanu początkowego tej bazy danych**.

- Harmonogramy są **równoważne co do wyniku** *(result equivalent)*, jeżeli dają ten sam stan bazy danych bez względu na początkowy stan bazy.

HARMONOGRAM SZEREGOWALNY KONFLIKTOWO *(conflict serializable)*

- Harmonogramy są **równoważne konfliktowo** *(conflict equivalent)* jeżeli kolejność wszystkich operacji konfliktowych jest w nich taka sama.

- Harmonogram **S** jest szeregowalny konfliktowo, jeżeli jest on równoważny konfliktowo z pewnym szeregowym harmonogramem **S'**.

- W takim przypadku możemy zamieniać kolejność niekonfliktowych operacji w **S** do momentu, aż utworzony zostanie równoważny harmonogram szeregowy **S'**.

HARMONOGRAM SZEREGOWALNY PERSPEKTYWICZNIE *(view serializability)*

- Jest on **równoważny perspektywicznie** pewnemu harmonogramowi szeregowemu.
- **Równoważność perspektywiczna**:
  - Harmonogram **S i S'** zwierają te same instrukcje i dla każdego elementu danych **Q**:
    1. T_k jest transakcją, która czyta **Q** jako pierwsza => T_k musi być transakcją, która czyta **Q** jako pierwsza.
    2. T_i czyta **Q** zapisany przez T_j => T_i czyta **Q** zapisany przez T_j
    3. T_m jest ostatnią transakcją, która zapisuje **Q** => T_m jest ostatnią transakcją, która zapisuje **Q**
- Oznacza to samo co **szeregowalność konfliktowa, jeśli** założymy ograniczenie co do operacji **zapisów** we wszystkich transakcjach harmonogramu.
  - Każda operacja `WRITE[x]` jest poprzedzona operacją `READ[x]`
  - Wartość zapisana przez `WRITE[x]` zależy tylko od wartości elementu `x` odczytanej przez operacje `READ[x]` (jest pewną nie stałą funkcją tylko elementu `x`, nie zależy od wartości innych elementów.)
- Szeregowalność perspektywiczna zapenia **spójność** bazy danych, ponieważ powoduje, że wyniki harmonogramu są takie same jak wyniki pewnego harmonogamu szeregowego.

ZNACZENIE W PRAKTYCE SZEREGOWALNOŚCI KONFLIKTOWEJ

- Zapewnia **spójność** bazy danych.

Proszę podać przykłady harmonogramów szeregowalnych nie szeregowych.

Proszę podać przykłady harmonogramów szeregowalnych konfliktowo i takich, które nie są szeregowalne konfliktowo.

4.3 Poziomy izolacji transakcji

Proszę omówić poziom izolacji transakcji wybrany przez egzaminatora.

4.4 Sterowanie współbieżnymi transakcjami w oparciu o blokady

4.5 Sterowanie współbieżnymi transakcjami z wykorzystaniem wielowersyjności i blokad

Dla przedstawionego harmonogramu proszę podać jak będzie wyglądać sterowanie współbieżnością w wybranym przez egzaminatora poziomie izolacji transakcji.

Proszę omówić wybrane przez egzaminatora problemy związane ze współbieżnym wykonaniem transakcji.

4.6 Zakleszczenia

Proszę napisać przykładowy harmonogram, który doprowadzi do zakleszczenia. Jak mogą być wykrywane zakleszczenia?

4.7 Kursory, sterowanie współbieżnością w kursorach

Proszę omówić, jak wygląda sterowanie współbieżnością w kursorach w systemie Microsoft SQL Server.

5 Język SQL

Proszę napisać zdanie w języku SQL, które zrealizuje cel podany przez egzaminatora.

---

6 Procedury, funkcje i wyzwalacze

Proszę napisać procedurę w języku Transact SQL, funkcjonalność procedury poda egzaminator.

Proszę napisać funkcję skalarną w języku Transact SQL, opisaną przez egzaminatora.

Proszę napisać w języku Transact SQL opisaną przez egzaminatora funkcję zwracającą zestaw rekordów.

Proszę napisać w języku Transact SQL wyzwalacz, który zrealizuje zadaną przez egzaminatora funkcjonalność.

---

7 Indeksy, typy indeksów, statystyki, wykorzystanie przez optymalizatory kwerend

---

Proszę omówić budowę indeksu typu drzewo B+. Proszę podać wersje tego indeksu (w systemie Microsoft SQL Server: clustered i non-clustered, w Oracle IOT).

---

8 Budowa fizyczna baz danych - macierze RAID

Proszę omówić macierze RAID (wybrany przez egzaminatora poziom).

---

9 Charakterystyka baz danych NoSQL

Proszę powiedzieć co oznacza termin skalowanie poziome w systemach baz danych i jak realizowane jest skalowanie poziome w bazach danych NoSQL.

Proszę podać główne typy baz danych NoSQL. Proszę podać przykłady dokumentów w formacie JSON. Jak dokumenty są przechowywane w systemach NoSQL zawierających wiele węzłów (komputerów) połączonych siecią komputerową? Czy łączenie danych z różnych dokumentów odbywa się na ogół w bazie danych (jak operacja JOIN w bazach relacyjnych), czy w aplikacji? Czy w bazach dokumentowych należy unikać redundancji tak, jak w systemach relacyjnych?
