Krok 8: Projekt bazy danych

Tu omówiono następujące tematy:

* Rzeczy do rozważenia przy projektowaniu bazy danych

* Różnice w filozofii projektowania baz danych i najlepszych praktykach projektowania baz danych operacyjnych i baz danych docelowych BI

* Wielowymiarowe założenie projektowe agregacji i podsumowania

* Podstawowe objaśnienia schematu gwiazdy i schematu płatka śniegu

* Aspekty projektowania fizycznej bazy danych, w tym opcje implementacji (takie jak wolne miejsce i miejsce w buforze), umieszczanie fizycznych zestawów danych, partycjonowanie, klastrowanie, indeksowanie, reorganizacje, tworzenie kopii zapasowych i odzyskiwanie oraz równoległe wykonywanie zapytań

* Krótkie opisy działań związanych z projektowaniem bazy danych, rezultatów wynikających z tych działań oraz zaangażowanych ról

* Ryzyko niewykonania kroku 8

Rzeczy do rozważenia

Raporty i zapytania

* Jakie wspólne wzorce raportowania istnieją w różnych działach?

* Jakiego poziomu szczegółowych danych będą wymagali ludzie biznesu w przypadku zapytań szczegółowych?

* Ile zapytań ad hoc o szczegółowe dane, które projektujemy, wystąpi?

* Ile wymiarów raportowania powinniśmy wziąć pod uwagę? Czym oni są?

* Ile nowych wymiarów trzeba będzie dodać w przyszłości?

Rozważania projektowe

* Czy powinniśmy przechowywać dane zagregowane i podsumowane?

* Jak dużego równoczesnego wykorzystania danych powinniśmy się spodziewać?

* Jak duże będą docelowe bazy danych BI? Jakie są przewidywane ilości danych i czynniki wzrostu?

* Ile danych historycznych będziemy przechowywać?

* Jaka będzie częstotliwość obciążeń?

* Czy bazy danych będą musiały być dystrybuowane?

* Jakie są wymagania dotyczące dostępności?

Rozważania dotyczące wydajności

* Jakie są wymagania dotyczące wydajności?

* Jak pogrupujemy tabele? Według kolumny daty lub według innych kolumn?

* Jakie stoły powinny znajdować się w jednym miejscu?

* Jak podzielimy stoły? Czy podzielimy się według daty?

* Jakich typów algorytmów indeksowania powinniśmy używać (B-drzewo, hash, odwrócony plik, sparse, binarny)?

* Czy możemy równolegle uruchamiać wiele operacji (zapytania, obciążenia)?

Wybór systemu zarządzania bazą danych

* Jakiego systemu zarządzania bazami danych (DBMS) używamy w naszych istniejących aplikacjach? Czy użyjemy tego samego DBMS dla docelowych baz danych BI?

* Czy nasz obecny DBMS skaluje się do oczekiwanego rozmiaru?

* Czy jesteśmy zadowoleni z obecnego DBMS? Jeśli nie, to co robimy o tym?

* Czy będziemy musieli licencjonować (kupić) inny DBMS?

Rekrutacja

* Jakie mamy umiejętności do projektowania docelowych baz danych BI?

* Czy mamy wystarczającą liczbę administratorów baz danych?

* Czy jeden administrator bazy danych może być zaangażowany w ten projekt w pełnym wymiarze godzin?

* Czy posiada umiejętności projektowania wielowymiarowego? Jeśli nie, jak szybko może przejść szkolenie?

* Czy będziemy musieli zatrudnić konsultanta, który będzie mentorem administratora bazy danych i zespołu?

Wymagania dotyczące wspomagania decyzji BI dla danych zagregowanych i podsumowanych wprowadziły nowy typ projektu bazy danych i nowy sposób przechowywania danych. Ten nowy schemat projektowania wielowymiarowej bazy danych, w połączeniu z nową technologią BI, wspiera możliwość „wycinania i krojenia" informacji na niezliczone sposoby w celach raportowania i analizy. Aby wdrożyć możliwości slicing i dicing, administratorzy baz danych i programiści muszą nauczyć się nowych technik projektowania i muszą nabyć nowy sposób pracy z bazami danych. Muszą zacząć od zrozumienia sposobów dostępu do danych. Dostęp do danych można uzyskać w sposób konwencjonalny (zwykle szczegółowe rekordy pobierane za pomocą zapytań w języku SQL) lub w sposób wielowymiarowy (zwykle podsumowane rekordy pobierane za pomocą narzędzia do przetwarzania analitycznego online [OLAP]). Wielowymiarowe techniki przechowywania danych i dostępu do danych, które obsługują krojenie i kostkowanie, umożliwiają przeglądanie informacji z różnych perspektyw, takich jak Produkty według fabryki według segmentu rynku i Segmenty rynku według produktu według fabryki.

Różnice w filozofii projektowania baz danych

Za docelowymi bazami danych BI stoi zupełnie inna filozofia projektowania w porównaniu z bazami operacyjnymi. Tabela podsumowuje różnice między tymi dwoma typami baz danych. v

Operacyjne bazy danych

Celem projektu operacyjnej bazy danych jest zapobieganie przechowywaniu tych samych atrybutów danych w wielu miejscach, a tym samym uniknięcie anomalii aktualizacji spowodowanych nadmiarowością. Innymi słowy, z operacyjnego punktu widzenia chcesz uniknąć przechowywania tych samych danych w wielu kolumnach w wielu tabelach, aby nie utraciły one synchronizacji. Projektowanie znormalizowanych struktur baz danych ma kluczowe znaczenie dla tworzenia relacyjnych baz danych, które wspierają tę intencję. Normalizacja zapewnia, że dane są tworzone, przechowywane i modyfikowane w spójny, nienadmiarowy sposób.

Operacyjne bazy danych:

* Ukierunkowany na eliminację nadmiarowości, koordynację aktualizacji i powtarzanie tego samego rodzaju operacji wiele razy dziennie, każdego dnia (na przykład rezerwacje linii lotniczych, wpłaty i wypłaty z kont bankowych, rezerwacje pokoi hotelowych).

* Większość systemów transakcyjnych wymaga czasu reakcji poniżej sekundy.

* Wysoce znormalizowane, aby wspierać spójne aktualizacje i utrzymanie integralności referencyjnej.

* Przechowuj bardzo mało danych pochodnych. W razie potrzeby dane są zwykle wyprowadzane dynamicznie.

* Nie przechowuj danych historycznych. Zapisy historyczne są archiwizowane.

* Lekko podsumowane, głównie do celów sprawozdawczych.

Docelowe bazy danych BI:

* Ukierunkowany na obsługę szerokiej gamy zapytań i raportów. Zapytania i raporty mogą różnić się w zależności od analityka biznesowego lub działu. Wszystkie zapytania i raporty mogą nie być uruchamiane tego samego dnia i mogą nie być uruchamiane codziennie (na przykład kwartalne raporty analizy trendów dotyczące sprzedaży regionalnej, miesięczny raport z realizacji zamówień).

* Chociaż czas odpowiedzi jest ważny, nie można oczekiwać subsekund. Typowe czasy odpowiedzi to sekundy, minuty lub godziny.

* Wysoce zdenormalizowane, aby zapewnić szybkie pobieranie szerokiego zakresu i dużej ilości danych. Dane, które należą do siebie z perspektywy raportowania analitycznego, są zwykle przechowywane razem.

* Przechowuj duże ilości danych pochodnych. Oszczędza to czas na zapytania i raporty.

* Przechowuj duże ilości danych historycznych, często na pewnym poziomie podsumowania, ale równie często na poziomie szczegółowym.

* Wiele poziomów wstępnie obliczonych, podsumowanych danych, od słabo podsumowanych do bardzo podsumowanych.

Większość systemów operacyjnych jest zaprojektowana zgodnie z filozofią wprowadzania danych (wprowadzanie danych), a nie zgodnie z filozofią wyprowadzania danych (raportowanie i zapytania). Celem filozofii wprowadzania danych jest uczynienie wprowadzania danych tak wydajnym, jak to tylko możliwe, przy wykonywaniu setek tysięcy transakcji dziennie, przy jednoczesnym eliminowaniu lub minimalizowaniu nadmiarowości danych. Nadmiarowość danych prowadzi do niespójności, a niespójności są często przyczyną niskiej jakości danych. Dlatego, próbując rozwiązać ogromne problemy z jakością i redundancją danych w systemach operacyjnych, celem jest uniknięcie redundancji (z wyjątkiem redundancji kluczowej, która jest nieunikniona). Cel ten osiąga się poprzez

normalizację. Chociaż normalizacja działa dobrze w przypadku systemów operacyjnych, wymagania dotyczące raportowania różnią się od wymagań dotyczących wprowadzania danych. Raportowanie wykorzystuje dane, które zostały już utworzone, co oznacza, że nie mogą wystąpić anomalie aktualizacji. Chociaż bardzo korzystne jest to, że dane są spójne i nienadmiarowe w wyniku znormalizowanego projektu bazy danych, ten sam projekt utrudnia raportowanie. Na przykład, aby utworzyć raporty z analizy trendów strategicznych, należy uzyskać dostęp do wielu tabel i odczytać każdy wiersz w tych tabelach. Jest to nie tylko złożone, ale także wyjątkowo nieefektywne, gdy jest uruchamiane w znormalizowanym projekcie bazy danych, ponieważ wymaga skanowania tabel i wykonywania dużych wielotabelowych sprzężeń JOIN. Z tego powodu większość docelowych baz danych BI opiera się na wielowymiarowym projekcie, w którym dane do raportów analizy trendów strategicznych są przechowywane w sposób wstępnie obliczony i wstępnie podsumowany. W tym przykładzie projekt operacyjnej bazy danych przedstawia bazę danych zamówień, w której klienci są powiązani z zamówieniami, a każde zamówienie składa się z wielu pozycji. Przy każdym złożonym zamówieniu pozycje należy odjąć od oddzielnej bazy danych zapasów. Projekt docelowej bazy danych BI przedstawia bazę danych z podsumowaniami używanymi do identyfikowania trendów w czasie. W tym projekcie te same dane o zamówieniach, elementach zamówienia i zasobach mogą znajdować się w wielu tabelach (Podsumowanie miesięczne, Podsumowanie regionalne, Podsumowanie produktu), aczkolwiek podsumowane według różnych wymiarów. Podczas gdy operacyjne bazy danych zazwyczaj przechowują dane szczegółowe (atomowe), docelowe bazy danych BI w większości przechowują dane podsumowane.

Bazy danych docelowych BI

W przeciwieństwie do filozofii wprowadzania danych (wprowadzania danych) systemów operacyjnych, filozofia wyprowadzania danych (raportowanie i zapytania) aplikacji BI obejmuje następujące kwestie projektowe.

* Docelowe bazy danych BI są zaprojektowane do uproszczonego, wysokowydajnego pobierania danych, a nie do wydajnego przechowywania i konserwacji danych (które są ważnymi względami projektowymi w przypadku operacyjnych baz danych).

* Wyeliminowanie lub zminimalizowanie nadmiarowości danych nie jest celem projektowania docelowych baz danych BI. Jeśli trzeba dokonać wyboru, przedkłada się nadmiarowość danych nad złożoność, ale nadmiarowość musi być kontrolowana. Zbędne dane muszą być spójne i możliwe do uzgodnienia.

* Poniżej przedstawiono podstawowe założenia do projektowania docelowych baz danych BI.

- Dane są przechowywane w taki sposób, aby były łatwo dostępne w sposób interesujący dla przedsiębiorców.

- Projekt zależy od dostępu i użytkowania.

- Znormalizowany projekt niekoniecznie jest intuicyjny dla przedsiębiorcy i dlatego może stać się dość skomplikowany.

- Nie można wymyślić danych BI! Wszystkie dane w docelowych bazach danych BI muszą istnieć w bieżących wewnętrznych lub zewnętrznych operacyjnych źródłach danych lub być z nich wyprowadzone.

Kluczową decyzją dla wszystkich aplikacji BI jest to, czy i na jakim poziomie przechowywać dane podsumowane w docelowych bazach danych BI. Administrator bazy danych i główny deweloper mogą

zdecydować się na przechowywanie zarówno danych szczegółowych, jak i podsumowanych, razem w tej samej docelowej bazie danych BI lub w różnych docelowych bazach danych BI. Ta decyzja dotycząca projektu bazy danych musi być oparta na wymaganiach dotyczących dostępu i użytkowania.

Logiczny projekt bazy danych

Ze względu na różnice w intencjach i celach między systemami operacyjnymi a aplikacjami BI opracowano różne techniki projektowania baz danych dla docelowych baz danych BI. Te wysoce zdenormalizowane projekty przechowują zagregowane i podsumowane dane w sposób wielowymiarowy. Logiczne projekty baz danych są dokumentowane jako fizyczne modele danych z technicznymi metadanymi. Agregacja i podsumowania są prawdopodobnie najważniejszymi czynnikami przyczyniającymi się do dobrej wydajności aplikacji BI. Jeśli większość analityków biznesowych potrzebuje podsumowania swoich danych, sumy te należy wstępnie obliczyć i przechowywać w celu szybkiego wyszukiwania. Ważne jest, aby omówić poziom szczegółowości z przedstawicielem biznesowym, a także z innymi analitykami biznesowymi, którzy będą korzystać z docelowych baz danych BI, ponieważ będą oni oczekiwać, że projekt bazy danych umożliwi im drążenie do określonego poziomu szczegółowości. Wielowymiarowe projekty baz danych umożliwiają szybkie wyszukiwanie szerokiego zakresu danych. Dwie popularne techniki projektowania wielowymiarowego to schemat gwiaździsty i schemat płatka śniegu, oba opisane poniżej.

Schemat gwiazdy

W schemacie gwiaździstym dane są reprezentowane jako tablica wstępnie obliczonych wartości, zwanych faktami, wokół których przeprowadzana jest analiza. Te wstępnie obliczone fakty reprezentują niepodzielne wartości danych operacyjnych, które zostały wstępnie podsumowane przez pewne wymiary, takie jak klient, produkt i czas. Wymiar w schemacie gwiaździstym jest podobny do encji w logicznym modelu danych: jest to obiekt biznesowy, o którym zbierane są dane do celów biznesowych. Schemat gwiaździsty odzwierciedla widok zapytania biznesowego. Jak sama nazwa wskazuje, schemat gwiaździsty ma pośrodku jeden obiekt, zwany tabelą faktów, który jest połączony promieniście z wieloma obiektami, zwanymi tabelami wymiarów.

Schemat gwiaździsty ma dwa i tylko dwa poziomy: tabelę faktów i serię jednopoziomowych tabel wymiarów. Tabele faktów mają następujące cechy:

* Tabela faktów przedstawia krytyczne wydarzenie biznesowe (działalność biznesową lub transakcję, taką jak sprzedaż lub roszczenie).

* Fakty to wymierne aspekty zdarzenia biznesowego; oznacza to, że są to kolumny w tabeli faktów.

* Tabela faktów łączy się z powiązanymi tabelami wymiarów (obiektami biznesowymi, takimi jak klient lub produkt).

* Tabela faktów ma długi klucz złożony składający się z kluczy podstawowych powiązanych tabel wymiarów (które są kluczami obcymi w tabeli faktów).

* Dla danego obszaru tematycznego może istnieć wiele bardzo redundantnych tabel faktów. Każda tabela faktów może zawierać inny poziom agregacji tych samych danych. Na przykład:

- Fakty dotyczące sprzedaży według sklepów według regionu według daty

- Fakty dotyczące sprzedaży według produktu według sklepu według daty

- Fakty dotyczące sprzedaży według klientów według regionu według daty

* Tabele faktów są długie i wąskie: tabele mają ogromną liczbę wierszy (długie), ale w tabelach jest stosunkowo mało kolumn (wąskie). Tabele wymiarów mają bardzo różne cechy.

* Tabele wymiarów to obiekty biznesowe reprezentujące różne perspektywy, z których można przeglądać i analizować fakty w tabeli faktów.

* Tabele wymiarów zwykle mają jednoatrybutowy klucz podstawowy.

* Tabele wymiarów są zdenormalizowane, co oznacza, że dane należące do siebie z określonej perspektywy biznesowej, takie jak hierarchia zestawień, są zgrupowane w jednej tabeli. Daje to pewne nadmiarowe wartości danych, które są dopuszczalne w tym schemacie projektu.

* Tabele wymiarów są krótkie i szerokie: tabele mają stosunkowo niewiele wierszy (krótkie), ale w tabelach jest wiele kolumn (szerokie).

* Jeśli to możliwe, tabele wymiarów powinny być współdzielone z tabelami faktów (wymiary zgodne).

* Jeden wymiar to zawsze wymiar czasu z atrybutami opisującymi sygnaturę czasową, takimi jak rok kalendarzowy, kwartał, sezon, okres obrachunkowy lub okres rozliczeniowy. Niektóre inne przykłady typowych tabel wymiarów to klient, produkt, zasady, przedstawiciel handlowy, region i sklep.

Większość wielowymiarowych DBMS skutecznie radzi sobie z optymalizacją dużych wielostolikowych JOIN. Jedną z metod określenia, czy DBMS skutecznie rozwiązuje zapytanie, jest przyjrzenie się zoptymalizowanemu planowi zapytania. Na przykład:

* Jeśli tabela faktów jest ostatnią tabelą JOIN, jest to wskaźnik optymalizacji. Jeśli tabela faktów wydaje się być gdzieś pośrodku lub nawet gdzieś na początku, DBMS może nie optymalnie rozwiązać JOIN, chyba że używa bardziej wyrafinowanych algorytmów JOIN.

* Jeśli DBMS nie używa JOIN produktów kartezjańskich, SZBD może wziąć kwalifikujące się klucze wierszy i zastosować je względem indeksu złożonej tabeli faktów lub może zastosować je poprzez przecięcie indeksu względem wielu indeksów pojedynczej kolumny tabeli faktów.

W obu przypadkach sprawdź, czy Twój DBMS wykonuje zapytania wielowymiarowe w najbardziej wydajny sposób, ponieważ od tego zależy Twoja wydajność. Z wielu powodów schemat gwiaździsty jest najpopularniejszym schematem projektowania baz danych dla aplikacji BI.

* Zapewnia najlepszą wydajność dla zapytań i raportów analizy trendów, które zawierają dane historyczne z lat.

* Zapewnia maksymalną elastyczność w wielowymiarowej analizie danych.

* Jest obsługiwany przez większość dostawców relacyjnych DBMS z modyfikacjami ich optymalizatora DBMS.

* Jego prostota sprawia, że złożona analiza danych jest znacznie mniej trudna niż w przypadku standardowego znormalizowanego projektu. O wiele łatwiej jest zadawać pytania takie jak:

- Który broker ubezpieczeniowy daje nam najbardziej lub najmniej intratny biznes?

- Jakie są najczęściej występujące rodzaje roszczeń od tego brokera ubezpieczeniowego?

- Kiedy pojawiają się te roszczenia?

Powyższe pytania są typowymi pytaniami drążącymi (z prośbą o bardziej szczegółowe dane) i typowymi pytaniami podsumowującymi (z prośbą o bardziej podsumowane dane).

Schemat płatka śniegu

Schemat płatka śniegu to odmiana schematu gwiazdy, z wyjątkiem płatka śniegu, w którym punkty gwiazdy promieniują na więcej punktów. W schematach typu płatek śniegu poziomy hierarchii w tabelach wymiarów są znormalizowane, zwiększając w ten sposób liczbę tabel. W tabeli wymieniono zalety i wady schematów płatków śniegu.

Zalety:

* Rozmiar tabel wymiarów jest zmniejszony i unika się nadmiarowości wartości danych, ponieważ hierarchie nadrzędny-podrzędny nie są już zwinięte.

* Zwiększona elastyczność aplikacji.

Wady:

* Zwiększona liczba tabel może niekorzystnie wpłynąć na zapytanie wydajności ze względu na niezbędne dodatkowe JOIN.

* Zwiększa się nakład pracy na konserwację bazy danych, ponieważ jest więcej tabel do utrzymania.

Fizyczny projekt bazy danych

Ponieważ aplikacje BI zwykle wymagają szczegółowych danych operacyjnych, a także danych podsumowanych, i ponieważ często muszą przechowywać część lub całość tych danych nadmiarowo, rozmiar niektórych docelowych baz danych BI może być ogromny. Bazy danych zbliżające się lub przekraczające jeden terabajt danych nazywane są bardzo dużymi bazami danych (VLDB). Projektowanie VLDB to duże wyzwanie, a codzienne obowiązki związane z utrzymaniem tych VLDB są wymagające. Należy podjąć wiele trudnych decyzji dotyczących projektowania fizycznego i zastosować pewne wysoce skuteczne ulepszenia wydajności. Poniższe sekcje przedstawiają kilka sugerowanych wskazówek.

Opcje implementacji

Prawie każdy DBMS pozwala administratorowi bazy danych wybierać spośród wielu opcji implementacji. Przy wdrażaniu docelowej bazy danych BI należy zwrócić szczególną uwagę na wybór właściwych opcji. Potrzeba doświadczenia, aby wiedzieć, która kombinacja opcji zapewni pożądany poziom wydajności. Decyzje wdrożeniowe obejmują:

* Ile wolnego miejsca do wyboru?

* Ile miejsca w buforze zadeklarować?

* Jak duży ustawić rozmiar bloku?

* Czy użyć dowolnej techniki zagęszczania

Umieszczenie fizycznego zbioru danych

Kolejną podstawową kwestią wpływającą na wydajność jest rozmieszczenie zestawów danych. Metody osiągania szybkiej odpowiedzi obejmują kombinacje:

* Przechowywanie często używanych danych na szybkich urządzeniach.

* Przechowywanie różnych poziomów agregacji na różnych platformach. Ze względu na wydajność może być konieczne przechowywanie zagregowanych danych na rozproszonych serwerach klasy średniej, przy jednoczesnym zachowaniu szczegółowych danych na komputerze mainframe.

* Stripowanie dysków w sposób z przeplotem, aby zoptymalizować użycie kontrolera wejścia/wyjścia (I/O). Używanie wielu małych dysków zamiast kilku dużych dysków, rozdzielanie tych dysków na oddzielne kontrolery i zapisywanie danych na urządzeniach zwiększa przepustowość we/wy.

* Umieszczanie zbiorów danych w taki sposób, aby w miarę możliwości unikać długich poszukiwań. Wybieranie adresów i schematów wyszukiwania, które wymagają kilku wyszukiwań, najlepiej tylko jednego na wyszukiwanie.

* Uruchamianie wielu operacji równolegle.

Zastanów się również, czy oddzielić indeksy od danych i umieścić je na osobnych dyskach.

Partycjonowanie

Upewnij się, że tabele są efektywnie podzielone na partycje na wielu dyskach. Jest to szczególnie ważne w przypadku VLDB, gdzie tabele faktów mogą sięgać kilkuset gigabajtów. Partycjonowanie umożliwia rozłożenie danych z jednej „logicznej" tabeli na wiele fizycznych zestawów danych. Fizyczna dystrybucja danych jest oparta na kolumnie partycjonowania, którą najczęściej jest data. Ponieważ kolumna partycjonująca musi być częścią klucza podstawowego tabeli, kolumna partycjonująca nie może być kolumną pochodną i nie może zawierać wartości NULL. Partycjonowanie umożliwia tworzenie kopii zapasowych i przywracanie części tabeli bez wpływu na dostępność innych części tej samej tabeli, które nie są objęte kopią zapasową ani przywracane.

Grupowanie

Zdefiniuj wymagania dotyczące tabeli klastrowej i fizycznie umieść powiązane tabele na dysku. Klastrowanie jest bardzo przydatną techniką sekwencyjnego dostępu do dużych ilości danych. Grupowanie odbywa się za pomocą wskaźników grupowania, które określają, w jakiej kolejności wiersze w tabelach są fizycznie przechowywane w zestawach danych. Najlepiej byłoby, gdyby klucz podstawowy każdej tabeli był klastrowany, aby uniknąć podziałów stron, to znaczy upewnić się, że nowe wiersze wstawiane do tabel będą przechowywane sekwencyjnie na dysku zgodnie z kolumnami w ich indeksie klastrowania. Korzystanie z tej techniki może znacznie poprawić wydajność, ponieważ sekwencyjny dostęp do danych jest normą w aplikacjach BI. Gdy wiersze tabeli nie są już przechowywane w tej samej kolejności co jej indeks klastrowy (fragmentacja danych), ucierpi wydajność i konieczne będzie zreorganizowanie tabeli.

Indeksowanie

Istnieją dwie skrajne strategie indeksowania, z których żadna nie jest wskazana: jedna strategia to indeksowanie wszystkiego, a druga to indeksowanie niczego. Zamiast popadać w te skrajności, indeksuj te kolumny, które są często przeszukiwane i mają wysoki rozkład wartości, takie jak Data otwarcia konta. Nie indeksuj kolumn, które mają niski rozkład wartości, takich jak kod płci. Po podjęciu decyzji, które kolumny indeksować, określ strategię indeksowania, której chcesz użyć. Większość DBMS oferuje kilka metod dostępu do wyboru, dostęp sekwencyjny lub dostęp bezpośredni przy użyciu jednego z następujących dobrze znanych algorytmów indeksowania:

* B-drzewo

* haszysz

* Odwrócony plik

* Rzadki

* Binarny

Skonsultuj się ze swoim dostawcą DBMS, aby wybrać najbardziej optymalną metodę dostępu (algorytm indeksowania) dla używanego produktu DBMS.

Reorganizacje

Czasami będziesz musiał zreorganizować bazy danych, ponieważ ładowanie przyrostowe spowoduje fragmentację zestawów danych w czasie, a wstawione wiersze nie będą już przechowywane w logicznej kolejności. Ta fragmentacja może skutkować długimi łańcuchami pobierania danych, a wydajność może znacznie spaść. Większość DBMS zapewnia procedury reorganizacji w celu przeorganizowania pofragmentowanej bazy danych w celu odzyskania miejsca zajmowanego przez usunięte dane lub przeniesienia rekordów z obszarów przepełnionych do wolnego miejsca w obszarach danych pierwotnych. Podstawowe czynności związane z reorganizacją bazy danych to skopiowanie starej bazy danych na inne urządzenie, ponowne zablokowanie wierszy i przeładowanie ich. Nie jest to trywialny wysiłek w przypadku docelowych baz danych BI. Dobrą wiadomością jest to, że wszystkie DBMS mogą wykonać procedurę częściowej reorganizacji partycji bazy danych, co jest kolejnym powodem, dla którego administrator bazy danych partycjonuje docelowe bazy danych BI.

Kopii zapasowych i odzyskiwania

Ponieważ oprogramowanie i sprzęt mogą ulec awarii, konieczne jest ustanowienie procedur tworzenia kopii zapasowych i odzyskiwania. DBMS zapewniają narzędzia do wykonywania pełnych kopii zapasowych, a także przyrostowych. Wiele organizacji ma błędne wrażenie, że docelowe bazy danych BI można zawsze odtworzyć z oryginalnych danych źródłowych. Nie zdają sobie sprawy, że odtworzenie celu BI może zająć bardzo dużo czasu bazom danych, jeśli muszą ponownie uruchomić wszystkie początkowe i historyczne programy do ekstrakcji/transformacji/ładowania (ETL) — zakładając, że oryginalne pliki źródłowe są nadal dostępne. Odzyskiwanie po awarii jest również problemem dla aplikacji BI. Jeśli taśmy lub kasety z kopiami zapasowymi zostaną zniszczone podczas awarii, odtworzenie docelowych baz danych BI może być trudne i może zająć bardzo dużo czasu (jeśli odzyskanie jest w ogóle możliwe). Z tego powodu wiele firm decyduje się na przechowywanie kopii zapasowych baz danych w lokalizacjach zdalnych.

Równoległe wykonywanie zapytań

Aby poprawić wydajność zapytania, podziel pojedyncze zapytanie na składniki, które mają być uruchamiane jednocześnie. Niektóre produkty DBMS oferują przezroczyste wykonywanie równoległe, co oznacza, że nie musisz wiedzieć, jak podzielić zapytanie na komponenty, ponieważ DBMS robi to za Ciebie. Wydajność jest znacznie zwiększona, gdy wiele części jednego zapytania działa równolegle na wielu procesorach. Inne zastosowania równoległego wykonywania zapytań to ładowanie partycji obszaru tabel, budowanie indeksów oraz skanowanie lub sortowanie tabel. Przetwarzanie równoległe jest bardzo ważną koncepcją dla aplikacji BI i powinno być brane pod uwagę, gdy tylko jest to możliwe.

Działania związane z projektowaniem baz danych

Czynności związane z projektowaniem bazy danych nie muszą być wykonywane liniowo. Poniższa lista zawiera krótki opis czynności związanych z Krokiem 8, Projektowanie Bazy Danych.

1. Przejrzyj wymagania dotyczące dostępu do danych. Baza danych

administrator musi przejrzeć wymagania dotyczące dostępu do danych i analizy (raporty, zapytania), które zostały przeanalizowane i sfinalizowane w kroku 6. Musi również przejrzeć wyniki prototypowania z głównym deweloperem aplikacji, aby pomóc określić najbardziej odpowiedni schemat projektowy dla BI docelowe bazy danych.

2. Określ wymagania dotyczące agregacji i podsumowania. Przed przystąpieniem do ostatecznego schematu projektu dla docelowych baz danych BI administrator bazy danych musi sfinalizować wymagania dotyczące agregacji i podsumowania danych z przedstawicielem biznesowym i głównym deweloperem aplikacji. Zwróć szczególną uwagę na eksplozję agregacji i podsumowań oraz ogólnie na eksplozję danych. Ludzie biznesu często proszą o dane „na wszelki wypadek", kiedy będą ich potrzebować, a potem rzadko z nich korzystają, jeśli w ogóle.

3. Zaprojektuj docelowe bazy danych BI. Powszechne twierdzenia, że wszystkie aplikacje BI dotyczą tylko wielowymiarowej analizy, a wielowymiarowe raportowanie nie są prawdziwe! Na przykład niektórzy analitycy finansowi (statystycy) podlegający dyrektorowi finansowemu lub dyrektorowi generalnemu z naciskiem określają swoje wymagania podobne do tego: „Muszę być w stanie zadać dowolne pytanie dotyczące jakichkolwiek szczegółowych danych w jakikolwiek sposób. Nie próbuj mnie pakować w dowolne z góry określone wzorce raportowania. Nie mam żadnego!" Analitycy ci potrzebują całkowitej elastyczności ad hoc w stosunku do szczegółowych danych historycznych i zawsze są gotowi zrezygnować z wydajności, nawet jeśli oznacza to, że ich zapytania będą działać przez wiele godzin lub przez całą noc. Chociaż tego typu analitycy stanowią zdecydowanie mniejszość, istnieją i należy wziąć pod uwagę ich wymagania dotyczące dostępu do danych. W związku z tym, chociaż projekty większości docelowych baz danych BI będą oparte na schemacie wielowymiarowym, niektóre będą oparte na schemacie relacji encji. Projekty baz danych są dokumentowane jako fizyczne modele danych. Wymogi dotyczące dostępu do danych oraz agregacji i podsumowania danych określą najbardziej odpowiedni projekt bazy danych. Jeśli istnieją oczywiste wzorce raportowania lub jeśli wymagania wymagają możliwości analizy plasterków i kostek, najbardziej odpowiedni projekt bazy danych jest wielowymiarowy. Jeśli nie ma wymagań dotyczących raportowania, a analitycy biznesowi upierają się, że potrzebują dostępu ad hoc do swoich szczegółowych danych, wówczas najbardziej odpowiednim projektem jest projekt relacji podmiot, który jest bardziej znormalizowany z niewielką liczbą lub bez agregacji lub podsumowań. Nie są to jedyne dwa schematy projektowe mające zastosowanie do docelowych baz danych BI. W przypadku niektórych typów wymagań dotyczących dostępu i analizy najbardziej odpowiedni może być projekt hybrydowy.

4. Projektować fizyczne struktury baz danych. Grupowanie, partycjonowanie, indeksowanie i odpowiednie umieszczanie zestawów danych to cztery najważniejsze cechy projektu fizycznej bazy danych. Administrator bazy danych powinien grupować najczęściej używane tabele, aby ograniczyć ruch ramienia dysku. Musi również określić, gdzie umieścić zestawy danych i jak podzielić tabele na wiele dysków. Na koniec musi wybrać strategię indeksowania.

5. Zbuduj docelowe bazy danych BI. Fizyczne bazy danych są budowane, gdy język definicji danych (DDL) jest uruchamiany w systemie DBMS. Administrator bazy danych używa języka DDL do opisywania struktur bazy danych (np. grup pamięci masowej, partycji bazy danych) w systemie DBMS.

Bezpieczeństwo bazy danych jest ustanawiane, gdy język kontroli danych (DCL) jest uruchamiany w systemie DBMS. W standardowych relacyjnych bazach danych bezpieczeństwo jest narzucane na poziomie tabeli lub widoku. Ze względu na wymiarowy charakter docelowych baz danych BI możliwość wchodzenia w szczegółowe dane, czasami między bazami danych, stanowi często pomijane zagrożenie bezpieczeństwa. Przyznaj uprawnienia do bazy danych osobom lub grupom, do których zostały przypisane osoby. Zarządzanie bezpieczeństwem na poziomie indywidualnym może szybko stać się

koszmarem konserwacji, dlatego większość organizacji woli konfigurować identyfikatory grupowe (identyfikatory grupowe). Każdemu identyfikatorowi grupy przyznawana jest pewna forma dostępu do tworzenia, odczytu, aktualizacji i usuwania (CRUD) do tabel. Ścieżka audytu może następnie pokazać, który konkretny „ID użytkownika" pod którym identyfikatorem grupy uzyskał dostęp do bazy danych. W przypadku naruszenia bezpieczeństwa „infiltrator" często może zostać zlokalizowany za pomocą tej ścieżki audytu.

6. Opracuj procedury konserwacji bazy danych. Gdy baza danych trafi do produkcji, ważne będzie zarezerwowanie czasu na wykonanie kopii zapasowej bazy danych lub reorganizację pofragmentowanych tabel. Dlatego należy ustanowić procedury dotyczące funkcji konserwacji bazy danych.

7. Przygotuj się do monitorowania i dostrajania projektów baz danych. Po wdrożeniu aplikacji BI docelowe bazy danych BI muszą być monitorowane i dostrajane. Najlepszy projekt bazy danych nie gwarantuje ciągłej dobrej wydajności, częściowo dlatego, że tabele ulegają fragmentacji, a częściowo dlatego, że rzeczywiste wykorzystanie docelowych baz danych BI zmienia się w czasie. Monitoruj wydajność zapytań w czasie wykonywania za pomocą narzędzia do monitorowania wydajności, które ma możliwości diagnostyczne. Nie pomaga wiedzieć, że wydajność uległa pogorszeniu bez znajomości przyczyn. Diagnozowanie wydajności

problemy są zwykle znacznie trudniejsze niż ich odkrywanie.

8. Przygotuj się do monitorowania i dostrajania projektów zapytań. Ponieważ wydajność jest takim wyzwaniem w aplikacjach BI, musisz zbadać wszystkie sztuczki, aby rozwiązać ten problem. Równoległe wykonywanie zapytań to jedna z tych sztuczek, które mogą zwiększyć wydajność zapytań.

Rezultaty wynikające z tych działań

1. Fizyczny model danych Fizyczny model danych, znany również jako logiczny projekt bazy danych, jest diagramem fizycznych struktur bazy danych, które będą zawierać dane BI. W zależności od wybranego schematu projektu bazy danych ten diagram może być diagramem relacji encji, diagramem schematu gwiazdy lub diagramem płatka śniegu. Pokazuje tabele, kolumny, klucze podstawowe, klucze obce, liczność, referencje zasady i indeksy integralności.

2. Fizyczny projekt docelowych baz danych BI. Fizyczne elementy projektu bazy danych obejmują umieszczenie zestawu danych, indeks rozmieszczania, partycjonowanie, klastrowanie i indeksowanie. Te fizyczne komponenty bazy danych muszą być zdefiniowane w DBMS podczas tworzenia docelowych baz danych BI.

3.Język definicji danych. DDL to zestaw instrukcji SQL, które informują SZBD o typach fizycznych struktur baz danych, takich jak bazy danych, przestrzenie tabel, tabele, kolumny i indeksy.

4. Język kontroli danych. DCL to zestaw instrukcji SQL, które informują DBMS, jakie typy dostępu CRUD należy przyznać ludziom, grupom, programom i narzędziom.

5. Fizyczne docelowe bazy danych BI. Uruchamianie (wykonywanie) instrukcji DDL i DCL tworzy rzeczywiste docelowe bazy danych BI.

6. Procedury utrzymania bazy danych. Procedury te opisują czas i częstotliwość przydzieloną do wykonywania bieżących czynności konserwacyjnych bazy danych, takich jak tworzenie kopii zapasowych bazy danych, odzyskiwanie (w tym odzyskiwanie po awarii) i reorganizacja bazy danych. Procedury powinny również określać proces i częstotliwość działań monitorowania wyników.

Role zaangażowane w te działania

* Główny programista aplikacji

Główny programista aplikacji i administrator bazy danych powinni przejrzeć wszystkie wnioski wyciągnięte podczas czynności prototypowania. Główny programista aplikacji powinien pomóc administratorowi bazy danych określić, które zapytania i raporty mogą być wykonywane równolegle i jaki rodzaj zabezpieczeń jest potrzebny.

* Administrator danych

Administrator danych powinien przekazać administratorowi bazy danych logiczny model danych i metadane. Logiczny model danych i metadane będą pomocne administratorowi bazy danych przy projektowaniu docelowych baz danych BI. Dzieje się tak, nawet jeśli wybrano schemat projektowania wielowymiarowej bazy danych, ponieważ jednostki i relacje w logicznym modelu danych są idealnym punktem wyjścia do projektowania dopasowanych wymiarów i znormalizowanych wymiarów płatka śniegu.

* Administrator bazy danych

Administrator bazy danych ponosi główną odpowiedzialność za projektowanie bazy danych. Musi znać ścieżki dostępu, ważyć przewidywane wolumeny danych i czynniki wzrostu oraz rozumieć ograniczenia platformy. Musi utworzyć i uruchomić DDL i DCL, aby zbudować fizyczne bazy danych. Ponadto odpowiada za wybór najodpowiedniejszych opcji wdrożeniowych.

Administratorzy baz danych, a nie programiści, powinni projektować bazy danych. Projektowanie bazy danych zwykle jest – i powinno być – częścią opisu zadania dla administratorów baz danych, ponieważ wymaga specjalnego szkolenia dotyczącego optymalizatora DBMS.

* Główny programista ETL

Proces ETL jest zależny od projektu bazy danych. Główny programista ETL powinien być zaangażowany w działania związane z projektowaniem bazy danych, aby być na bieżąco z wszelkimi zmianami projektu bazy danych, które będą miały wpływ na proces ETL lub specyfikacje programowania ETL.

Ryzyko niewykonania kroku 8

Tabele nie są zwykłymi plikami w bazie danych i nie są tylko innym sposobem na przypadkowe przechowywanie niektórych danych. Relacyjne silniki DBMS są oparte na skomplikowanych wewnętrznych zestawach reguł. Zasady te muszą być zrozumiane i przestrzegane. Organizacje zatrudniają do tego administratorów baz danych. Jednak zbyt często programiści, którzy nie są dogłębnie zaznajomieni z wewnętrznym działaniem swoich silników DBMS, mogą projektować docelowe bazy danych BI i projektują je słabo. Może to mieć katastrofalny wpływ na wydajność. W rzeczywistości może zabić aplikację BI, jeśli nie całą inicjatywę wspierania decyzji BI.