Pro Dialog
Wydawnictwo NAKOM - Poznan

Magazyny danych – stan obecny i kierunki rozwoju

Robert WREMBEL, Zbyszko KRÓLIKOWSKI, Mikolaj MORZY

Politechnika Poznanska, Instytut Informatyki
ul. Piotrowo 3a, 60-965 Poznan

Otrzymano [...]

Streszczenie. Magazyny danych sa bardzo duzymi bazami danych, w których gromadzi sie dane pochodzace z wielu heterogenicznych zródel, np. innych scentralizowanych lub rozproszonych baz relacyjnych, relacyjno–obiektowych, obiektowych oraz ze zródel innych niz bazy danych, takich, jak arkusze kalkulacyjne, pliki XML, zasoby WWW. Potrzeba gromadzenia takich danych jest szczególnie widoczna w sektorze handlowym. Analiza informacji o rynku, popycie i sprzedazy danych towarów, trendach i anomaliach jest jednym z etapów podejmowania strategicznych decyzji w przedsiebiorstwie. Analize przeprowadza sie zwykle za pomoca aplikacji OLAP wykorzystujacych bardzo zlozone zapytania i operujacych na ogromnych ilosciach danych, rzedu setek, czy tysiecy gigabajtów. Kluczowym problemem w magazynach danych jest wiec efektywnosc przetwarzania informacji. W ostatnim czasie wiele uwagi poswiecono opracowaniu modeli danych (wielowymiarowy model danych), metod fizycznej implementacji modelu wielowymiarowego (relacyjny i wielowymiarowy OLAP), nowych algorytmów optymalizacji zapytan (indeksy bitmapowe).

Artykul ten jest poswiecony omówieniu procesu konstruowania magazynu danych, zagadnien zwiazanych z pielegnacja i odswiezaniem magazynu oraz kwestiom architektury magazynu danych. W artykule przedstawiono nowy model przetwarzania (OLAP), wielowymiarowy model danych, metody jego implementacji oraz opisano podstawy zagadnienia optymalizacji zapytan w srodowisku magazynu danych.

Slowa kluczowe: magazyn danych, OLAP, OLTP, analiza danych, wielowymiarowy model danych, kostka danych, indeks bitmapowy

1 Wprowadzenie

Sposób, w jaki uzytkownik korzysta z bazy danych nazywamy modelem przetwarzania. Tradycyjny model przetwarzania – *przetwarzanie transakcji w trybie on-line* (ang. On-Line Transaction Processing OLTP) doskonale nadaje sie do obslugi biezacej dzialalnosci firmy, obslugi operacji bankowych i ksiegowych, wykonywania powtarzalnych i dobrze zdefiniowanych procesów. Model ten charakteryzuje sie krótkimi i prostymi transakcjami które operuja na niewielkich ilosciach danych. Wiekszosc operacji stanowia operacje modyfikacji i zapisu krotek, równolegle wykonuja sie setki badz tysiace transakcji, stad kluczowe znaczenie maja kwestie integralnosci, odtwarzania czy izolacji. Uzytkownikami systemów OLTP sa najczesciej urzednicy, pracownicy obslugi klienta, itp. Niestety, model OLTP nie znajduje zastosowania w procesach analizy danych, w systemach wspomagania decyzji czy tez w przypadku procesów nie do konca dobrze zdefiniowanych.

Wspomaganie decyzji stanowi drugi obszar zastosowania systemów baz danych. Rozwijajace sie galezie przemyslu, handlu, medycyny, nauki wymagaja skladowania i przetwarzania ogromnych ilosci danych. Dane sa zwykle przechowywane w systemach informatycznych posiadajacych rózne struktury i wykorzystujacych rózne modele danych (np. hierarchiczne, relacyjne, obiektowe), w dokumentach tekstowych, czy arkuszach kalkulacyjnych. Poniewaz przedsiebiorstwa, instytucje, organizacje (producenci danych) sa czesto geograficznie rozproszone, wiec same dane maja równiez charakter rozproszony. Posiadanie danych opisujacych dzialanie przedsiebiorstwa w dluzszym przedziale czasu pozwala na analize trendów, anomalii, poszukiwania wzorców zachowan (wzorców zakupów, podobienstw miedzy klientami, itp.) czy tez odkrywania wiedzy. Dane przechowywane w bazie danych zawieraja olbrzymia ilosc potencjalnie uzytecznej wiedzy, która moze zostac uzyta w procesie podejmowania decyzji strategicznych dotyczacych dzialalnosci przedsiebiorstwa. Dla potrzeb analizy danych opracowano zatem nowy model przetwarzania, nazywany *przetwarzaniem analitycznym w trybie on-line* (ang. On-Line Analytical Processing OLAP) oraz stworzono nowy typ architektury bazy danych nazywany *magazynem danych* (ang. data warehouse). Magazyny danych sa bardzo duzymi bazami danych, w których gromadzi sie dane pochodzace z wielu heterogenicznych zródel, np. innych scentralizowanych lub rozproszonych baz relacyjnych, relacyjno–obiektowych, obiektowych oraz ze zródel innych niz bazy danych, np. arkusze kalkulacyjne, dokumenty XML lub pliki tekstowe. Analiza informacji o rynku, popycie i podazy danych towarów, trendach, modach i anomaliach jest bardzo istotna dla procesu podejmowania strategicznych decyzji w przedsiebiorstwie. Analize przeprowadza sie zwykle poprzez bardzo zlozone zapytania operujace na ogromnych ilosciach danych, rzedu setek, czy tysiecy gigabajtów. Kluczowym problemem w magazynach danych jest wiec efektywnosc przetwarzania informacji.

W niniejszym artykule przedstawiony zostanie szczególowo model przetwarzania OLAP oraz metody modelowania, projektowania i implementacji magazynu danych.

2 Model przetwarzania analitycznego w trybie on-line (OLAP)

Komercyjnie dostepne systemy transakcyjne (systemy zarzadzania bazami danych SZBD) dostarczaja efektywnych rozwiazan dla takich problemów jak: efektywne i bezpieczne przechowywanie danych, transakcyjne odtwarzanie danych, dostepnosc danych, optymalizacja dostepu do danych, zarzadzanie wspólbieznoscia. W znacznie mniejszym stopniu systemy te wspomagaja operacje agregacji danych, wykonywania podsumowan czy tez optymalizacji zlozonych zapytan formulowanych ad hoc. Systemy te w niewielkim stopniu wspomagaja równiez integracje danych z róznych heterogenicznych zródel danych. W ostatnim czasie prace badawcze i rozwojowe prowadzone nad rozszerzeniem funkcjonalnosci systemów baz danych doprowadzily do opracowania nowego modelu przetwarzania danych, którego podstawowym celem jest wspomaganie procesów podejmowania decyzji, oraz opracowania nowego typu relacyjnej bazy danych nazwanego *magazynem danych* (ang. data warehouse).

Nowy model przetwarzania danych, nazwany *przetwarzaniem analitycznym on-line* (ang. On-Line Analytical Processing OLAP), ma za zadanie wspieranie procesów analizy magazynów danych dostarczajac narzedzi umozliwiajacych analize magazynu w wielu „wymiarach" definiowanych przez uzytkowników (czas, miejsce, klasyfikacja produktów, itp.). Analiza magazynu polega na obliczaniu agregatów dla zadanych „wymiarów" magazynu. Nalezy podkreslic, ze proces analizy jest calkowicie sterowany przez uzytkownika. Mówimy czasami o *analizie danych sterowanej zapytaniami* (ang. query-driven exploration). Typowym przykladem takiej analizy jest zapytanie o sprzedaz produktów w supermarkecie w kolejnych kwartalach, miesiacach, tygodniach, itp., zapytanie o sprzedaz produktów z podzialem na rodzaje produktów (AGD, produkty spozywcze, kosmetyki, itp.), czy wreszcie zapytanie o sprzedaz produktów z podzialem na oddzialy supermarketu. Odpowiedzi na powyzsze zapytania umozliwiaja decydentom okreslenie waskich gardel sprzedazy, produktów przynoszacych deficyt, oraz podjecie odpowiednich dzialan poprawiajacych sytuacje. Niestety, obliczenia agregatów sa bardzo czasochlonne. Czesciowym rozwiazaniem problemu efektywnosci analizy magazynu danych jest wstepne przeprowadzenie obliczen i zmaterializowanie otrzymanych wyników w magazynie danych w celu ich pózniejszego wykorzystania.. Jednakze, liczba mozliwych agregatów jest wykladniczo zalezna od liczby „wymiarów", stad rodzi sie pytanie, które z agregatów materializowac, które agregaty pozostawic do obliczen w trybie on-line, oraz w jaki sposób wykorzystac zmaterializowane agregaty do obliczenia innych agregatów. Pojawia sie szereg dodatkowych bardzo interesujacych pytan: w jaki sposób reprezentowac i realizowac dostep do agregatów, w jaki sposób aktualizowac agregaty, czy nie nalezy równiez materializowac posrednich wyników obliczen (nie tylko samych agregatów), np. wyników niektórych operacji polaczen, które sa wspólne dla wielu agregatów.

Model OLAP adresowany jest przede wszystkim do decydentów, menadzerów, analityków systemowych. Model ten charakteryzuje sie m.in. mala liczba zlozonych i kosztownych zapytan.

Znakomita wiekszosc transakcji w modelu OLAP sklada sie z operacji odczytu duzych wolumenów danych. Dane sa historyczne, zagregowane, zintegrowane i wielowymiarowe. Pojedyncze zapytanie operuje na tysiacach badz milionach rekordów. Podstawowa miara wydajnosci systemów nie jest ilosc transakcji na jednostke czasu (jak w przypadku systemów OLTP), lecz czas odpowiedzi na pojedyncze zapytanie. Poniewaz operacja modyfikacji magazynu danych jest bardzo rzadka, kryteria izolacji i integralnosci transakcji nie sa tak istotne, jak w przypadku systemów OLTP.

Analiza danych ma olbrzymie znaczenie dla procesów wspomagania podejmowania decyzji. Jednakze, aby przeprowadzic taka analize, nalezy dysponowac odpowiednimi danymi opisujacymi dzialalnosc przedsiebiorstwa. Bardzo rzadko informacje te sa dostepne w jednej bazie danych. Z reguly sa one rozproszone po wielu oddzialowych, rozproszonych geograficznie i heterogenicznych bazach danych. Stad, opracowujac koncepcje systemu wspomagania podejmowania decyzji nalezy odpowiedziec na dwa zasadnicze pytania odnosnie architektury takiego systemu i modelu przetwarzania. Pierwsze pytanie brzmi: czy analiza powinna miec charakter rozproszony czy scentralizowany, innymi slowy, czy dane nalezy zgromadzic i przetwarzac w jednym miejscu w sposób scentralizowany, czy tez korzystajac z mechanizmu transakcji rozproszonych mozna przetwarzac dane w sposób rozproszony. Drugie pytanie dotyczy koegzystencji dwóch systemów – systemu biezacej obslugi dzialania przedsiebiorstwa oraz systemu wspomagania podejmowania decyzji. Oba systemy operuja na tych samych danych, stad pytanie, czy oba modele OLAP i OLTP moga wspólistniec w tym samym systemie bazy danych, czy tez powinny funkcjonowac niezaleznie.

W chwili obecnej odpowiedz na powyzsze pytania brzmi: analiza powinna miec charakter scentralizowany, a modele OLAP i OLTP powinny funkcjonowac niezaleznie. Oczywiscie, odpowiedz na pytania o architekture i model przetwarzania jest uzalezniona od aktualnego stanu rozwoju technologii informatycznej. Ze wzgledu na charakter i pracochlonnosc obliczen, czesciowo równiez ze wzgledu na problem autoryzacji dostepu do danych, analiza danych jest aktualnie prowadzona w sposób scentralizowany. Wraz z rozwojem sieci komputerowych, wzrostem predkosci transmisji danych, nalezy sie jednak spodziewac przechodzenia od modelu przetwarzania analitycznego scentralizowanego do modelu przetwarzania analitycznego rozproszonego.

3 Projektowanie magazynu danych

Magazyn danych jest „(...) zorientowana tematycznie, zintegrowana, zmienna w czasie i nieulotna kolekcja danych wspierajacych proces wspierania decyzji” (W.H.Inmon). Jest to zbiór technologii, których celem jest wspieranie decydentów (menedzerów, analityków) i umozliwianie im podejmowania lepszych i szybszych decyzji. Zaklada sie, ze w magazynie danych znajduja sie wlasciwe dane we wlasciwym miejscu, wlasciwym czasie i wlasciwym koszcie dla wlasciwych decyzji.

Magazyny danych sa **zorientowane tematycznie** w zaleznosci od podstawowego obszaru dzialalnosci danego przedsiebiorstwa. Dla przykladu, operacyjne bazy danych firmy ubezpieczeniowej moga byc zorientowane na poszczególne segmenty dzialalnosci firmy, np. ubezpieczenia na zycie, ubezpieczenia samochodów, domów, itp. Magazyn danych tej firmy bedzie zorientowany na klientów, typy ubezpieczen, polisy, konta, zadania wyplat, itp. Nalezy raz jeszcze powtórzyc, ze celem magazynu danych nie jest wspieranie biezacej dzialalnosci danego przedsiebiorstwa lecz asystowanie decydentom w analizie rynku, w podejmowaniu decyzji strategicznych dla przedsiebiorstwa, w znajdowaniu waskich gardel w dzialalnosci firmy.

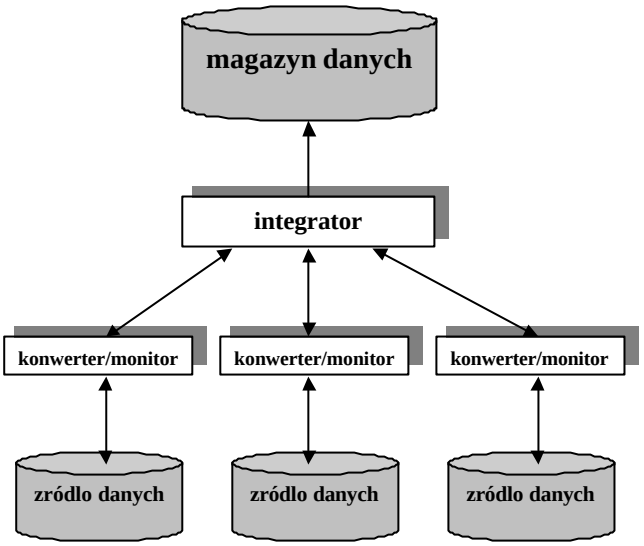
Aby wypelnic postawione przed nimi cele magazyny danych musza dysponowac pelna wiedza dotyczaca calokształtu funkcjonowania firmy. Niezbedna staje sie wiec **integracja danych** z wielu heterogenicznych zródel. Waznym wymaganiem systemów informatycznych jest dostep do informacji pochodzacych zarówno z heterogenicznych baz danych (wykonanych w róznych technologiach), jak i ze zródel innych niz bazy danych, np. arkusze kalkulacyjne, dokumenty tekstowe, pliki semi- i strukturalne HTML, SGML czy XML. Informacje przechowywane w róznych zródlach sa integrowane w magazynie danych. Integracja polega na rozwiazywaniu niekompatybilnosci miedzy róznymi typami danych, uspójnianiu nazewnictwa, konwersji wartosci, itp.

Dane operacyjne sa regularnie aktualizowane i zmieniane. Transakcje w systemach OLTP dokonuja dostepów do pojedynczych rekordów, zmieniaja ich wartosci, usuwaja badz wstawiaja rekordy. Magazyny danych natomiast sa **nieulotne**, po zaladowaniu danych do magazynu dane te podlegaja tylko operacji odczytu. Aktualizacja danych przechowywanych w magazynie danych, czyli *odswiezenie danych* (ang. refresh) odbywa sie w trybie wsadowym w okreslonych odstepach czasu (raz na tydzien, raz na miesiac).

Istotna cecha magazynów danych jest ich **zmiennosc w czasie**. Horyzont czasowy magazynu danych jest znaczaco wiekszy niz horyzont czasowy operacyjnych baz danych. Bazy operacyjne przechowuja aktualne wartosci danych i nie zawsze zawieraja element czasu. W przeciwienstwie do nich, magazyny danych przechowuja cala historie danych (czyli zbiór migawek zrobionych w pewnych odstepach czasowych) i czas stanowi zawsze jeden z podstawowych elementów skladowych magazynu danych.

Technologia magazynów danych stala sie istotna strategia integracji heterogenicznych zródel informacji. Zgodnie z raportem grupy META wydatki na budowe magazynów danych osiagnely w roku 1998 wysokosc $15 mld i z kazdym rokiem wzrastaja. Szacunkowy koszt jednego projektu zwiazanego z tworzeniem badz modernizacja magazynu danych wynosi obecnie $3 mln i równiez wzrasta.

Architektura magazynu danych zostala przedstawiona na Rys. 1 [16]. Obiekty oznaczone jako *zródlo danych* reprezentuja wymienione wyzej heterogeniczne zródla informacji. Z kazdym z takich zródel jest zwiazana warstwa oprogramowania –*konwerter/monitor.*



Rys.1. Architektura magazynu danych

Zadaniem modulu *konwertera* jest transformowanie danych z formatu wykorzystywanego w zródle, do formatu wykorzystywanego w magazynie. Dlatego, dla kazdego modelu danych zródlowych konieczne jest zastosowanie specyficznego modulu *konwertera*. Przykladowo, jesli zródlo przechowuje dane w dokumentach tekstowych, a magazyn zostal zaprojektowany z wykorzystaniem modelu relacyjnego, to *konwerter* musi zapewnic poprawne odwzorowanie danych z plików w struktury modelu relacyjnego.

Konwersja danych ze zródel do magazynu rozpoczyna sie od procesu *ekstrakcji danych* (ang. data extraction). Dane sa czytane ze struktur zródlowych (plików tekstowych, arkuszy kalkulacyjnych, sieci WWW, baz danych) za pomoca *bramek* (ang. gateways) i standardowych interfejsów (Information Builders EDA/SQL, ODBC, JDBC, Oracle Open Connect, Sybase Enterprise Connect, Informix Enterprise Gateway, itp.), choc w wielu przypadkach wystepuje potrzeba zaimplementowania wyspecjalizowanych procedur do ekstrakcji danych ze zródel niestandardowych. Kolejnym krokiem jest *czyszczenie danych* (ang. data cleaning, data cleansing, data scrubbing). Proces ten ma na celu zapewnienie jakosci i poprawnosci danych. Ma on szczególne znaczenie w przypadku, gdy zródla danych sa wysoce heterogeniczne, posluguja sie róznymi schematami danych lub opisuja te same obiekty swiata rzeczywistego w rózny sposób. Dane przed transformacja czesto nazywane sa danymi „brudnymi”. Przykladami zabrudzenia danych sa [5]:

- różne formaty danych dla tego samego pola (np. informacja o województwie moze byc przedstawiona jako pelna nazwa, skrót nazwy lub specjalny identyfikator)
- niespójne wartosci tej samej danej spowodowane bledami przy wprowadzaniu
- niezgodnosci miedzy wartoscia atrybutu i jego nazwa (np. pole *Nazwa* moze zawierac nazwe firmy lub nazwisko indywidualnego klienta)
- brakujace wartosci, które zgodnie ze schematem magazynu danych powinny byc wypelnione
- redundantne informacje na temat jakiegos obiektu swiata rzeczywistego (moga pojawiac sie zarówno w ramach jednego zródla informacji, jak i w dwóch róznych zródlach)

Podstawowymi metodami czyszczenia danych sa:

- konwersja i normalizacja: transformacja i standaryzacja heterogenicznych formatów danych
- czyszczenie specjalne: uspójnianie wartosci pola na podstawie slownika synonimów
- czyszczenie niezalezne od domeny: porównywanie pól pochodzacych z róznych zródel w celu okreslenia stopnia ich podobienstwa
- czyszczenie oparte na regulach: okreslanie podobienstwa miedzy polami na podstawie zespolu predefiniowanych regul

Po wyczyszczeniu danych nastepuje etap *ladowania danych* (ang. data loading) do magazynu danych. Procesem ladowania zarzadza zaznaczony na Rys.1. modul *integratora.* Ladowanie danych pociaga za soba dodatkowe przetwarzanie, np. sprawdzanie ograniczen integralnosciowych, sortowanie, podsumowywanie, budowanie indeksów, itp. Ladowanie danych odbywa sie najczesciej w trybie wsadowym. Aplikacja dokonujaca zaladowania danych musi pozwalac administratorowi na monitorowanie procesu ladowania, zawieszanie i odwieszanie ladowania, restartowanie po awarii (np. po naruszeniu ograniczen integralnosciowych). Proces ladowania danych moze byc bardzo czasochlonny i trwac wiele godzin badz dni. W rzeczywistosci taki proces moze byc traktowany jako jedna transakcja konstruujaca nowa baze danych. Po zaladowaniu danych wszystkie nastepujace zmiany w danych zródlowych sa propagowane do magazynu danych podczas odswiezania. *Odswiezanie danych* (ang. data refresh) to proces propagowania zmian zachodzacych w zródlach danych do magazynu. Z odswiezaniem wiaza sie dwa istotne problemy: w jaki sposób odswiezac oraz kiedy odswiezac. Odswiezanie magazynu danych moze zachodzic okresowo (np. pod koniec dnia, kiedy magazyn danych nie jest uzywany), natychmiastowo (zmiany propagowane sa do magazynu natychmiast po wystapieniu w danych zródlowych) lub w sposób okreslony przez charakter zródla (np. po wystapieniu okreslonej ilosci zmian). Jesli chodzi o metody odswiezania mozna wyróznic zasadniczo dwie metody:

- data shipping: relacja w magazynie danych jest traktowana jak zdalna migawka zródla danych. Wyzwalacze dzialajace w zródle sa odpowiedzialne za aktualizacje logu migawki i propagacje danych do magazynu
- transaction shipping: log transakcji w zródle danych jest okresowo przegladany celem wykrycia zaistnialych zmian. Wszelkie zmiany sa przesylane do serwera replikacji który przesyla nowe transakcje do magazynu danych

Zadaniem modulu *monitora* jest wykrywanie zmian w danych zródlowych i ich przekazywanie do warstwy oprogramowania *integratora* (po uprzedniej konwersji do modelu danych magazynu). Sposób wykrywania zmian w danych zródlowych zalezy od wlasnosci samych zródel. Wyróznia sie cztery nastepujace rodzaje zródel danych [16]:

- aktywne (ang. active sources), tzn. posiadajace zaimplementowane mechanizmy wyzwalaczy, które informuja *monitor* o zmianach zachodzacych w danych zródlowych
- utrzymujace dzienniki operacji wykonywanych na danych zródlowych (ang. logged sources) – zmiany sa wykrywane przez analize zawartosci dziennika przez modul *monitora*
- umozliwiajace wydawanie zapytan (ang. queryable sources) – w celu wykrycia zmian w danych zródlowych *monitor* okresowo wydaje zapytania do zródla
- wspierajace mechanizm migawek (ang. snapshot sources) – w tego rodzaju zródlach zmiany wykrywa sie przez porównanie zawartosci kolejnych migawek

W przypadku gdy problem wykrywania i propagowania zmian w danych zródlowych jest trudny do rozwiazania, stosuje sie okresowe uaktualnianie calego magazynu w trybie wsadowym. Wówczas magazyn jest niedostepny dla uzytkowników. Ta technika jest wykorzystywana równiez w przypadku gdy aktualnosc danych nie jest konieczna dla poprawnego dzialania firmy, instytucji czy organizacji, oraz gdy dopuszczalna jest czasowa niedostepnosc danych.

Istotnym skladnikiem magazynu danych jest *repozytorium metadanych* (ang. metadata repository), w którym przechowywane sa informacje wspomagajace zarzadzanie magazynem. Metadane wspomagaja tez efektywny dostep do zasobów magazynu poprzez przechowywanie informacji o zawartosci magazynu, zaleznosciach miedzy poszczególnymi komponentami systemu lub ich fizycznej lokalizacji w obrebie magazynu. Repozytorium zazwyczaj zawiera nastepujace informacje:

- liste zródlowych baz danych i opis ich zawartosci
- opisy i charakterystyki bramek miedzy bazami zródlowymi a magazynem
- schemat magazynu
- definicje perspektyw i danych wyliczalnych, przechowywanych w magazynie

- opisy wymiarów i hierarchii (patrz dalej)
- predefiniowane zapytania i raporty
- lokalizacje tematycznych hurtowni danych
- zasady czyszczenia, transformacji i korekcji danych zródlowych
- zasady odswiezania danych
- profile uzytkowników i profile grup uzytkowników
- dane dotyczace bezpieczenstwa magazynu (autoryzacja uzytkowników, prawa dostepu do poszczególnych komponentów magazynu)

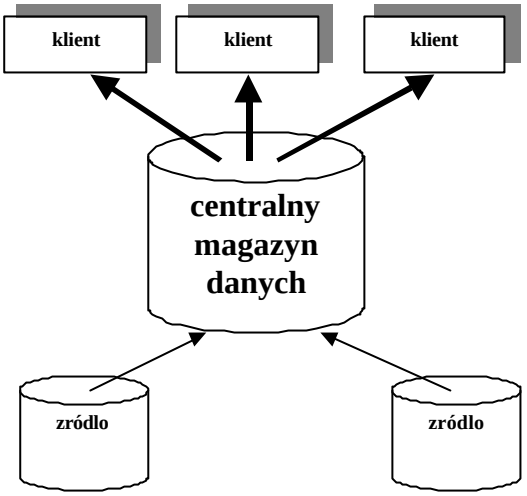
Istniejace komercyjne systemy zarzadzania bazami danych wspierajace technologie hurtowni danych umozliwiaja integracje danych pochodzacych z systemów relacyjnych, dostarczanych przez najwiekszych wytwórców. Firma Oracle dostarcza oprogramowanie o nazwie *Oracle Loader* i *Oracle Transparen Gateways* umozliwiajace dostep oraz eksportowanie i importowanie danych z wiekszosci systemów relacyjnych. IBM umozliwia wczytywanie i transformowanie danych do formatu bazy DB2 za pomoca oprogramowania o nazwie *DataPropagator*, *DataRefresher* i *DataHub. SourcePoint* jest oprogramowaniem firmy Software AG sluzacym do ladowania hurtowni danymi pochodzacymi z innych relacyjnych zródel. Informix realizuje dostep do danych zgromadzonych w bazie *Oracle*, *Sybase* i *DB2* za pomoca oprogramowania *Informix–Enterprise Gateway*. Oprogramowanie firmy Sybase umozliwia bezposrednie wczytywanie danych z wielu systemów relacyjnych. Wszystkie z istniejacych na rynku systemów wspieraja standard ODBC (ang. Open Database Connecitvity) dostepu do bazy danych.

4 Fizyczne struktury magazynów danych

Istnieja trzy podstawowe architektury konstrukcji magazynów danych [9]:

- architektura scentralizowana
- architektura sfederowana
- architektura wielowarstwowa

W architekturze scentralizowanej istnieje tylko jeden magazyn danych przechowujacy wszystkie dane potrzebne do analiz. Architektura scentralizowana zostala przedstawiona na Rys.2.

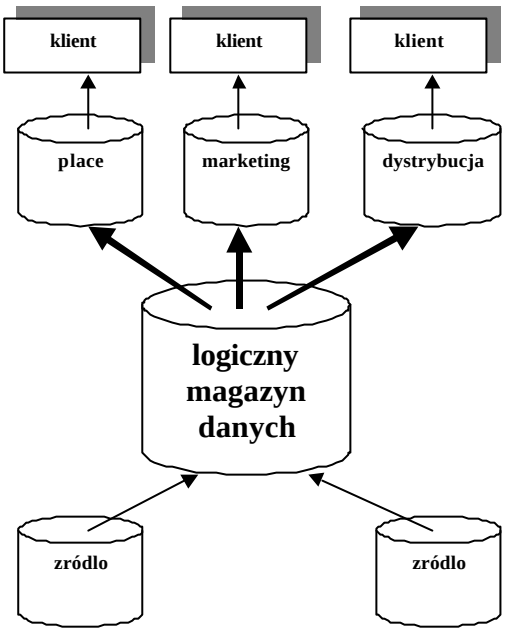


Rys.2. Architektura scentralizowana

Efektywnosc systemu scentralizowanego jest mniejsza niz systemu rozproszonego. Wszystkie zapytania i operacje odswiezenia magazynu danych wykonywane sa w jednej bazie danych. Zaleta architektury scentralizowanej jest latwosc dostepu do danych i latwosc administrowania magazynem. W architekturze scentralizowanej nie zachodzi problem integracji schematów magazynu poniewaz wystepuje tylko jeden spójny model danych.

Podstawowymi powodami, dla których stosuje sie rozproszone architektury magazynów danych, sa: skalowalnosc, zwiekszenie dostepnosci i odpornosci na awarie oraz zmniejszenie czasu odpowiedzi systemu (dane sa zlokalizowane blizej aplikacji klientów, w tematycznych magazynach danych wolumeny przeszukiwanych danych sa mniejsze).

W przypadku decentralizacji magazynu danych kazdy z wezlów rozproszonego magazynu danych posiada wlasne repozytorium metadanych. W architekturze sfederowanej dane sa zintegrowane logicznie, lecz fizycznie sa przechowywane w osobnych bazach danych. Wazna cecha architektury sfederowanej jest wiec fakt, ze magazyn danych jest czysto wirtualny. *Tematyczne magazyny danych* (ang. data marts) przechowuja szczególowe dane, które sa wykorzystywane do wykonywania zapytan. Schemat architektury sfederalizowanej przedstawiono na Rys.3.



Rys.3. Architektura sfederowana

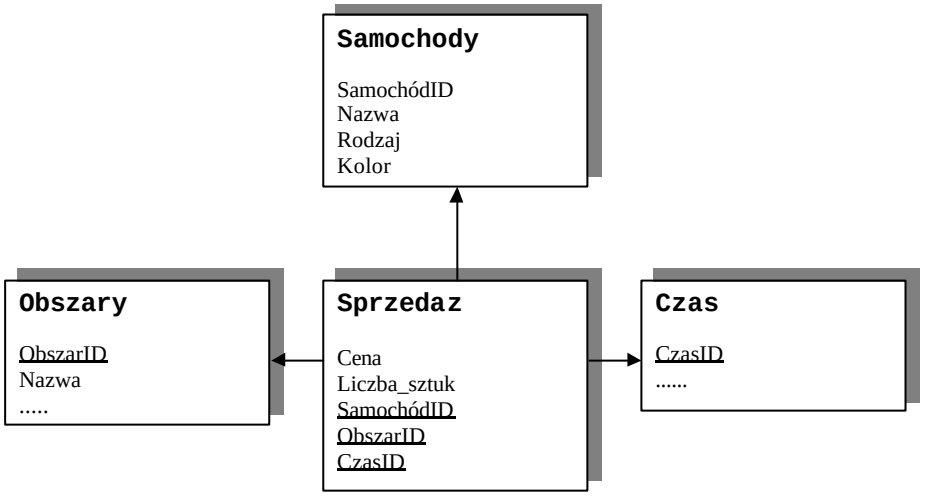
W przypadku architektury wielowarstwowej centralny magazyn danych jest fizyczny. Dodatkowo w lokalnych bazach danych przechowywane sa dane charakterystyczne dla danego oddzialu. Lokalne bazy danych przechowuja kopie tych fragmentów magazynu, które sa istotne dla danego oddzialu. Kopie te sa wstepnie przetworzone i wzbogacone o agregaty, statystyki, itp. Zawartosc lokalnych baz danych nie jest jednak tak szczególowa, jak w przypadku architektury sfederowanej.

5 Modele danych dla magazynów danych

Bazy danych wspierajace technologie magazynów danych mozna podzielic na dwa rodzaje ze wzgledu na wykorzystywane przez nie modele danych. Pierwszy, to magazyny danych wykorzystujace *model relacyjny*, nazywane równiez *ROLAP* (ang. Relational OLAP), drugi to magazyny danych wykorzystujace wielowymiarowy model danych, nazywane równiez *MOLAP* (ang. Multidimensional OLAP).

5.1 ROLAP

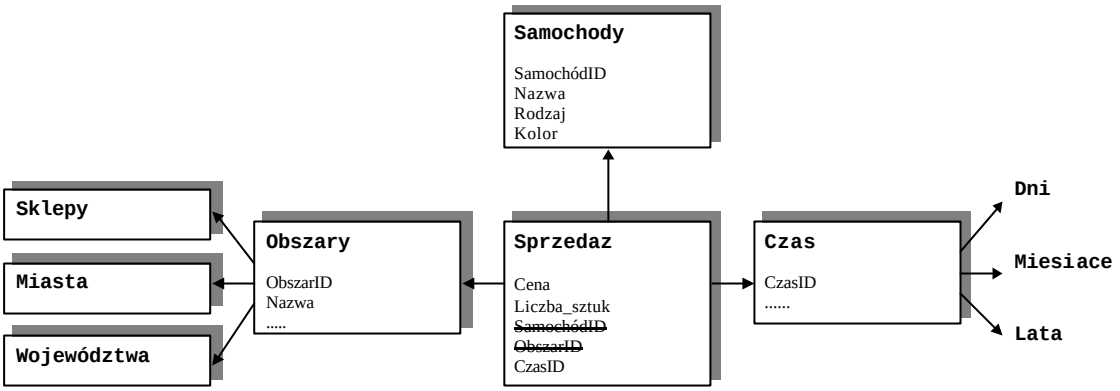
Magazyn danych tego typu zbudowany jest w oparciu o system zarzadzania relacyjna baza danych posiadajacy mechanizmy efektywnego przetwarzania zapytan typu OLAP. Zwykle schemat takiego magazynu posiada strukture *gwiazdy* (ang. star schema) lub strukture bardziej zlozona, przypominajaca *platek sniegu* (ang. snowflake schema). W celu skrócenia czasu potrzebnego na wyznaczenie wyników zapytania relacje bazy danych sa czesto denormalizowane, np. zawieraja wartosci zagregowane, sa wynikiem polaczenia wielu innych relacji [2]. Przykladowy schemat o strukturze gwiazdy przedstawia Rys.4.



Rys.4. Schemat gwiazdy

Centralna relacja *Sprzedaz* zawiera informacje o sprzedazy samochodów, w pewnych obszarach geograficznych, w okreslonym czasie. Relacje *Samochody*, *Obszary* i *Czas* sa nazywane *wymiarami* (ang. dimensions), natomiast relacja centralna jest nazywana *relacja faktów* (ang. fact table). Atrybuty relacji faktów przechowujace informacje o sprzedazy sa nazywane *miarami* (ang. measures), np. *Cena*, *Liczba_sztuk*. Relacja faktów *Sprzedaz* zawiera równiez atrybuty *SamochódID*, *ObszarID*, *CzasID*, których wartosci wskazuja na odpowiednie wymiary i pozwalaja zidentyfikowac dany fakt (sprzedaz pewnej ilosci samochodów za pewna cene) w przestrzeni wymiarów (jaki samochód? gdzie sprzedany? kiedy?).

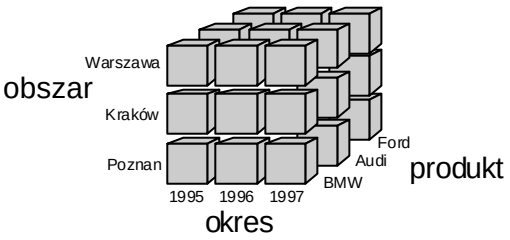
Wymiary moga tworzyc hierarchie, wówczas schemat przybiera postac platka sniegu [8]. Rysunek 5 przedstawia przykladowy schemat o takiej strukturze.



Rys.5. Schemat platka sniegu

5.2 MOLAP

Magazyn danych zaprojektowany w technologii MOLAP do przechowywania danych wykorzystuje *wielowymiarowe tablice* (ang. multidimensional arrays) zwane tez *kostkami danych* (ang. data cubes). Tablice te zawieraja wstepnie przetworzone (m.in. zagregowane) dane pochodzace z wielu zródel. Przykladowa tablica zawierajaca trzy wymiary: *Obszar*, *Okres* i *Produkt* oraz zagregowane informacje o sprzedazy samochodów w poszczególnych latach w wybranych miastach zostala przedstawiona na Rys.6.



Rys.6. Tablica trójwymiarowa

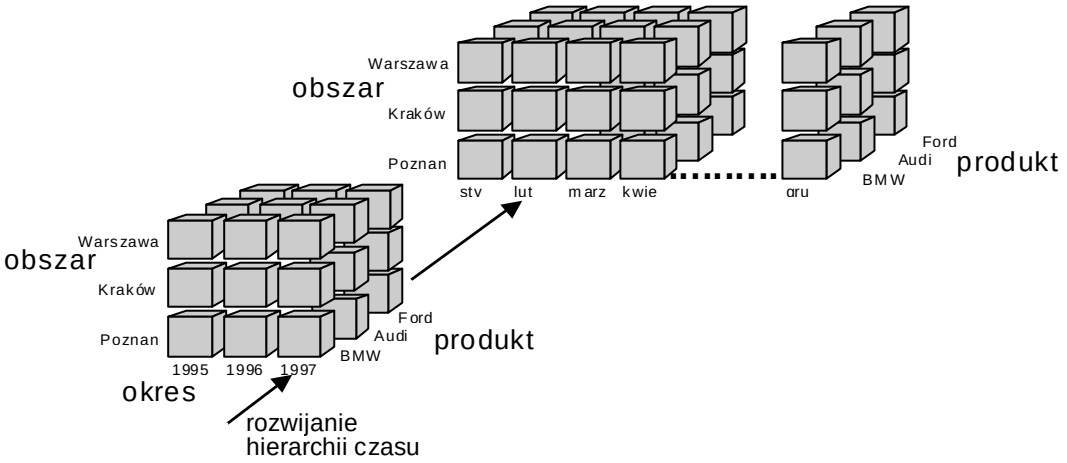
Analize danych wielowymiarowych wspomagaja specjalne operatory, do których naleza:

- *wyznaczanie punktu centralnego* (ang. pivoting)

  Operacja ta polega na wskazaniu miary i okresleniu wymiarów, w których wybrana miara bedzie prezentowana. Przykladowo, w wymiarze produktu reprezentujacego samochód marki „BMW" i wymiarze obszaru reprezentujacego sklepy województwa poznanskiego moze byc prezentowana liczba sprzedanych samochodów.

- *rozwijanie* (ang. drilling down)

  Rozwijanie polega na zaglebianiu sie w hierarchie danego wymiaru w celu przeprowadzenia bardziej szczególowej analizy danych. Jako przyklad rozwazmy informacje o sprzedazy samochodów marek BMW, Audi i Ford, w latach 1995, 1996 i 1997, w poszczególnych miastach (por. Rys. 6). W celu dokonania analizy sprzedazy w poszczególnych mie siacach roku 1997 nalezy rozwinac hierarchie reprezentujaca czas, tj. rok 1997 (zob. Rys. 7). Analiza sprzedazy w poszczególnych dniach wybranego miesiaca bedzie mozliwa po rozwinieciu hierarchii reprezentujacej ten miesiac.



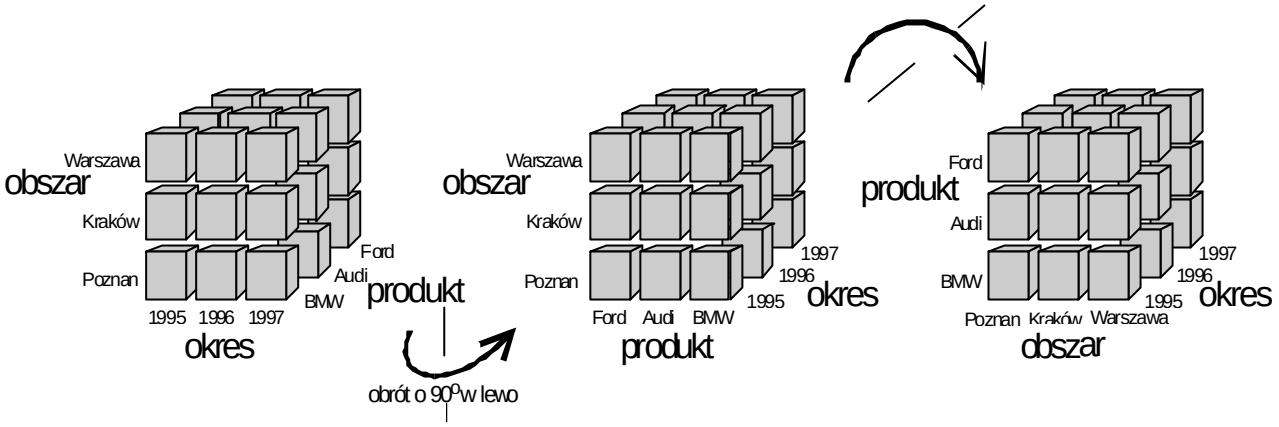
Rys.7. Operacja rozwijania hierarchii wymiaru

- *zwijanie* (ang. rolling up)

  Zwijanie jest operacja odwrotna do rozwijania i polega na nawigowaniu w góre hierarchii danego wymiaru. Dzieki tej operacji mozna przeprowadzac analize danych zagregowanych na wyzszym poziomie hierarchii wymiarów.

- *obracanie* (ang. rotating)

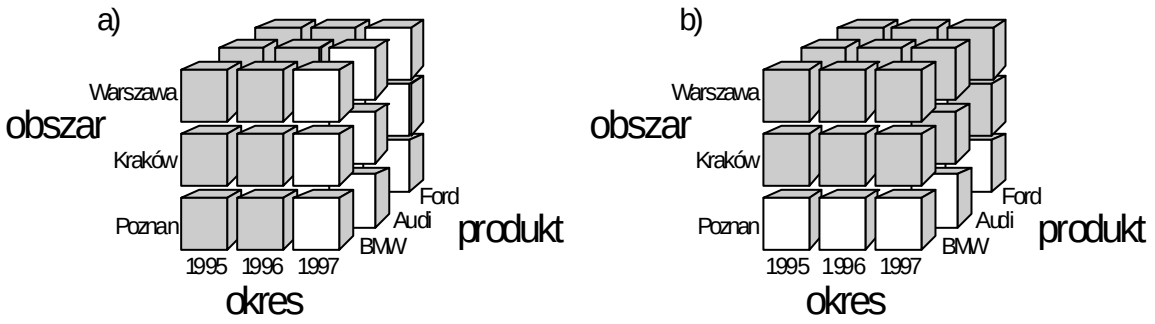
  Operacja obracania (zob. Rys. 8) umozliwia prezentowanie danych w róznych ukladach. Celem jej jest zwiekszenie czytelnosci analizowanych informacji.



Rys.8. Operacja obracania

- *wycinanie* (ang. slicing and dicing)

  Operacja ta umozliwia zawezenie analizowanych danych do wybranych wymiarów, a w ramach kazdego z wymiarów – zawezenie analizy do konkretnych jego wartosci. Przykladowo, dyrektor do spraw marketingu bedzie zainteresowany wielkoscia sprzedazy wszystkich produktów, we wszystkich miastach kraju, w roku biezacym (zob. Rys. 9a). Natomiast kierownika oddzialu firmy w Poznaniu bedzie interesowala wielkosc sprzedazy wszystkich produktów, w ciagu calego okresu dzialalnosci (zob. Rys. 9b).



Rys.9. Wycinanie danych w róznych wymiarach

- *obliczanie rankingu* (ang. ranking)

  Operacja ta umozliwia uporzadkowanie informacji w danym wymiarze, zgodnie z wartosciami wybranych miar (w kolejnosci malejacej lub narastajacej). Przykladowo, w wymiarze roku 1997 mozna uporzadkowac marki samochodów zgodnie z narastajacym porzadkiem liczby sprzedanych egzemplarzy.

6 Efektywnosc systemu hurtowni danych

Aby systemy relacyjnych baz danych mogly efektywnie przetwarzac zapytania typu OLAP i zarzadzac danymi rzedu terabajtów musza posiadac nowe wlasnosci zwiekszajace ich efektywnosc. Nowymi cechami systemów relacyjnych sa m.in.: przetwarzanie równolegle, parcelacja danych i nowe techniki indeksowania.

6.1 Przetwarzanie równolegle

Przetwarzanie równolegle (ang. parallel processing) polega na rozbiciu zlozonych operacji na mniejsze, które nastepnie sa wykonywane równolegle, np. na wielu procesorach lub komputerach. W efekcie, czas wykonania calej operacji jest krótszy. W przypadku hurtowni danych, najczesciej równolegle przetwarza sie zapytania, sortuje dane, wykonuje operacje odczytu i zapisu na dysk, buduje tablice i indeksy oraz wczytuje dane do hurtowni.

Przetwarzanie równolegle wspieraja m.in. systemy zarzadzania bazami danych: *Oracle7* i *Oracle8* (Oracle Corporation) , *DB2* (IBM), *OnLine Extended Parallel Server*, *OnLine Dynamic Server* (Informix), *Red Brick Warehouse* (Red Brick), *Sybase IQ* (Sybase) [3].

6.2 Parcelacja danych

Parcelacja danych (ang. data partitioning) umozliwia automatyczne rozpraszanie danych (pochodzacych z jednej lub wielu relacji) na wiele dysków, znajdujacych sie w tym samym lub wielu wezlach (komputerach) sieci. Dzieki podzialowi duzej relacji na mniejsze:

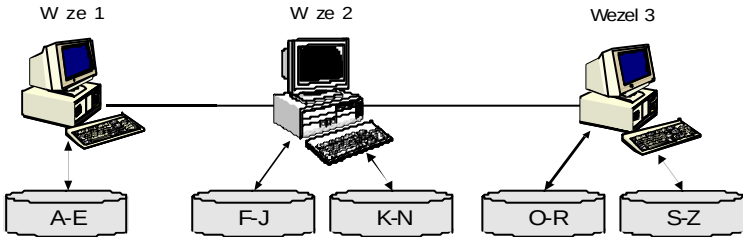
- bardzo kosztowne operacje wejscia/wyjscia, tj. dostepu do dysków moga byc wykonywane równolegle,
- równowazone jest obciazenie dysków,
- polecenia SQL moga byc wykonywane równolegle, np. tworzenie relacji i indeksów, wykonywanie zapytan,
- wzrasta bezpieczenstwo danych w przypadku awarii sprzetu,

- wzrasta szybkosc tworzenia kopii zapasowych bazy i szybkosc odtwarzania danych po awarii.

W komercyjnych SZBD stosuje sie cztery podstawowe techniki parcelacji danych: *round–robin* (ang. round–robin partitioning), *parcelacja bazujaca na wartosci* (ang. range partitioning), *parcelacja haszowa* (ang. hash partitioning), *parcelacja hybrydowa* (ang. hybrid partitioning).

Technika *round–robin* umozliwia równomierne rozproszenie danych w wezlach sieci. Przykladowo, jesli w sieci znajduja sie trzy wezly, to pierwsza krotka relacji zostanie umieszczona w wezle pierwszym, druga – w wezle drugim, trzecia krotka – w wezle trzecim, czwarta – znów w wezle pierwszym itd. Rozwiazanie to ma jednak wade. Poniewaz dane sa rozproszone w sposób przypadkowy, wiec odnalezienie zadanych informacji wymaga przeszukania wszystkich wezlów.

W przypadku *parcelacji bazujacej na wartosci* rozmieszczenie danych w sieci zalezy od wartosci samych danych. Przykladowo, relacja zawierajaca informacje o klientach sieci supermarketów moze byc podzielona zgodnie z wartoscia pierwszej litery nazwiska, jak na Rys.8. Wezel pierwszy posiada jeden dysk, na którym umieszczono klientów o nazwiskach rozpoczynajacych sie od liter A do E. Wezly drugi i trzeci posiadaja po dwa dyski, na których umieszczono klientów o nazwiskach odpowiednio z zakresów F–J, K–N, O–R, S–Z.



Rys. 8. Parcelacja danych bazujaca na wartosci

Ten sposób rozpraszania danych jest efektywny dla zapytan wykorzystujacych zakresy wartosci w predykatach selekcji, poniewaz umozliwia szybki dostep do danych z zadanego zakresu, bez potrzeby przeszukiwania wszystkich wezlów.

W przypadku *parcelacji haszowej* dane sa umieszczane w wezlach zgodnie z wartoscia systemowej funkcji haszowej. Argumentem wejsciowym tej funkcji jest wartosc atrybutu, a jej wynikiem – adres wezla, w którym zostanie umieszczona krotka. W celu odnalezienia zadanych informacji SZBD wykorzystuje te sama funkcje haszowa. Zaleta tej metody jest mozliwosc automatycznego umieszczania w tym samym wezle krotek pochodzacych z róznych, powiazanych z soba relacji. W ten sposób zwieksza sie efektywnosc wykonywania operacji laczenia krotek, gdyz laczone z soba krotki znajduja sie w tym samym wezle.

Parcelacja hybrydowa umozliwia dwustopniowe rozpraszanie danych. W kroku pierwszym dane sa umieszczane w poszczególnych wezlach za pomoca parcelacji haszowej. W kroku drugim dane sa umieszczane na poszczególnych dyskach danego wezla, za pomoca parcelacji bazujacej na wartosci. Dzieki tej technice wzrasta równomiernosc rozproszenia danych i obciazenia wezlów.

6.3 Indeksowanie danych

Indeksowanie polega na laczeniu wartosci atrybutów kluczowych z adresami fizycznych bloków dyskowych w celu przyspieszenia dostepu do danych. Dla bardzo zlozonych zapytan typu OLAP, operujacych na ogromnej liczbie danych, standardowe indeksy w postaci B–drzew okazuja sie nieefektywne poniewaz

- nie zapewniaja wystarczajaco szybkiego dostepu do danych,
- ich rozmiar jest zbyt duzy, przez co wzrastaja koszty ich przetwarzania, przechowywania i utrzymywania.

Dla tej klasy zastosowan stosuje sie wiec nowe struktury indeksów, tj. *indeksy bitmapowe* (ang. bit–mapped indexes) i *indeksy polaczeniowe* (ang. join indexes) [1], [10], [11], [12], [17].

Idea *indeksów bitmapowych* jest wykorzystanie pojedynczych bitów do zapamietania informacji o tym, ze dana wartosc atrybutu wystepuje w okreslonej krotce relacji. Dla kazdej unikalnej wartosci atrybutu jest przechowywana tablica bitów, zwana *mapa bitowa*. Kazdy bit mapy odpowiada jednej krotce relacji R – bit pierwszy odpowiada pierwszej krotce relacji R, bit drugi – drugiej krotce itp. Dla mapy A='w' bit n przyjmuje wartosc jeden, jesli atrybut A krotki o numerze n przyjmuje wartosc 'w'. W przeciwnym przypadku bit n przyjmuje wartosc zero. Liczba bitów mapy bitowej odpowiada liczbie krotek relacji R.

Indeks bitmapowy jest zbiorem map bitowych dla wszystkich unikalnych wartosci danego atrybutu. Indeks tego typu moze równiez posiadac strukture B–drzewa, w którego lisciach zamiast adresów rekordów sa przechowywane mapy bitowe.

Powyzsza koncepcja zostanie zilustrowana przykladem. Tabela 1 przedstawia fragment relacji *Sprzedaz* przechowujacej informacje o sprzedazy samochodów. Dla atrybutu *kolor* zdefiniowano indeks bitmapowy skladajacy sie z dwóch map bitowych. Pierwsza z nich opisuje te krotki relacji, dla których atrybut *kolor* przyjmuje wartosc *'zielony'*, a druga – te krotki, dla których *kolor*='*niebieski*'. Poniewaz atrybut ten przyjmuje dwie rózne wartosci, wiec indeks zawiera dwie mapy bitowe. Bit pierwszy mapy *kolor='zielony'* przyjmuje wartosc 1, co oznacza, ze atrybut *kolor* pierwszej krotki przyjmuje wartosc *'zielony'*. Natomiast bit drugi mapy przyjmuje wartosc 0 poniewaz wartoscia atrybutu *kolor* krotki drugiej nie jest *'zielony'*.

Sprzedaz		
klientID	**marka**	**Kolor**
1010	Fiat	zielony
1020	BMW	niebieski
1030	Fiat	zielony
1040	Audi	zielony
1050	Volvo	zielony
1060	Fiat	niebieski
1070	Ford	niebieski
1080	Opel	zielony
1090	Opel	niebieski
1100	Ford	zielony

Tabela 1. Przykladowa relacja *Sprzedaz*

kolor	
zielony	**niebieski**
1	0
0	1
1	0
1	0
1	0
0	1
0	1
1	0
0	1
1	0

Tabela 2. Indeks bitmapowy dla atrybutu *kolor*

Podstawowa zaleta indeksów bitmapowych jest ich maly rozmiar dla atrybutów o waskiej dziedzinie wartosci. Jako przyklad rozwazmy relacje *R* zawierajaca milion krotek. Przyjmijmy, ze na atrybucie *A* przyjmujacym cztery rózne wartosci utworzono indeks bitmapowy. Indeks ten sklada sie z czterech map bitowych o rozmiarze miliona bitów kazda, co daje rozmiar kazdej z map w przyblizeniu 125kB. Laczny rozmiar indeksu bitmapowego wynosi zatem w przyblizeniu 500kB (4 x 125kB).

Wyznaczmy teraz rozmiar tradycyjnego indeksu w postaci B–drzewa zbudowanego na atrybucie *A*, przyjmujac nastepujace zalozenia: adresy krotek sa czterobajtowe, liscie zawieraja skompresowane listy adresów krotek. Wówczas, indeks przechowuje milion adresów krotek, a wiec jego rozmiar wynosi w przyblizeniu 4MB. Jest wiec osiem razy wiekszy od odpowiadajacego mu indeksu bitmapowego.

Rozwazmy jednak te sama relacje *R*, której atrybut *A* przyjmuje 64 rózne wartosci. W tym przypadku indeks bitmapowy bedzie mial rozmiar 8MB (64 x 125kB). Bedzie wiec wiekszy niz odpowiadajacy mu indeks w postaci B–drzewa. W celu zmniejszenia rozmiarów indeksów bitmapowych stosuje sie ich automatyczna kompresje, m.in. w systemach *Oracle7*, *Oracle8* i *Sybase IQ*.

Indeksy bitmapowe sa bardziej efektywne od indeksów w postaci B–drzewa tylko dla okreslonej klasy zapytan kierowanych do bazy danych. Sa to zapytania wykorzystujace duza liczbe predykatów warunkowych oraz zapytania wykorzystujace funkcje COUNT. Wieksza efektywnosc tych indeksów wynika z:

- duzej szybkosci przetwarzania map bitowych za pomoca operatorów AND, OR i NOT – dla popularnych procesorów 64–bitowych, w jednym cyklu sa przetwarzane 64 bity mapy.
- malego rozmiaru indeksów – indeksy takie zdefiniowane na atrybutach o waskiej dziedzinie sa znacznie mniejsze od indeksów w postaci B–drzewa, dzieki czemu moga byc przechowywane w pamieci operacyjnej.

- Mozliwosci wykonywania operacji logicznych i funkcji COUNT bezposrednio na indeksach bitmapowych (znajdujacych sie w pamieci operacyjnej), a nie na samych krotkach. W rezultacie, system nie wykonuje kosztownych odczytów danych z dysku w czasie przetwarzania polecenia. Dane sa odczytywane dopiero po wykonaniu wszystkich operacji logicznych na indeksach. Ich wynikiem jest mapa bitowa opisujaca tylko te krotki, które spelniaja warunki selekcji.

Indeksy bitmapowe wykazuja jednak mniejsza efektywnosc dla zapytan wyszukujacych dane z zadanych zakresów (operatory >, < itp.), poniewaz dla takiej klasy zapytan nalezy wykonac wiele operacji sum logicznych na mapach bitowych dla wszystkich wartosci z zadanego zakresu.

Rozmiar indeksu bitmapowego zalezy od krotnosci atrybutu, dla którego zostal zdefiniowany. Dla danych o niskiej krotnosci indeks bitmapowy jest maly, w porównaniu z indeksem w postaci B–drzewa. Natomiast dla danych o wysokiej krotnosci indeks bitmapowy zawiera wiele map bitowych o niskiej gestosci, co pociaga za soba spadek efektywnosci ich przetwarzania i zwieksza koszty skladowania.

Utrzymywanie indeksów bitmapowych w czasie pracy systemu moze byc kosztowne, poniewaz:

- wstawienie pojedynczej krotki do relacji wymaga uaktualnienia wszystkich map bitowych zdefiniowanych dla tej relacji,
- w przypadku usuwania krotek nalezy utrzymywac dodatkowa mape bitowa dla kazdej relacji, której krotki sa usuwane.

Innym rodzajem indeksu zwiekszajacym szybkosc przetwarzania danych jest tzw. *indeks polaczeniowy* (ang. join index). Indeks tego typu laczy z soba krotki z róznych relacji posiadajace te sama wartosc atrybutu polaczeniowego, jest wiec struktura zawie rajaca zmaterializowane polaczenie wielu relacji. Indeks taki posiada strukture B–drzewa zbudowanego na atrybucie polaczeniowym relacji (badz na wielu takich atrybutach). Liscie indeksu zawieraja wspólne wartosci atrybutu polaczeniowego relacji wraz z listami adresów krotek w kazdej z laczonych relacji.

Odmiana tego indeksu jest tzw. *bitmapowy indeks polaczeniowy* (ang. bit–mapped join index), który rózni sie od powyzszego tym, ze w lisciach zamiast adresów krotek znajduja sie mapy bitowe opisujace krotki laczonych relacji.

Oceniajac przydatnosc indeksów bitmapowych w magazynach danych nalezy wziac pod uwage trzy nastepujace aspekty: zysk czasowy ze stosowania indeksów, rozmiar indeksów i latwosc utrzymywania struktury indeksów w trakcie pracy systemu.

Sposród istniejacych na rynku serwerów baz danych mechanizmy tworzenia i zarzadzania indeksami bitmapowymi posiadaja: *Oracle7* i *Oracle8* (Oracle), *Sybase IQ* (Sybase), *OnLine Dynamic*

Server, *OnLine Extended Parallel Server* i *OnLine Workgroup Server* (Informix) i *Red Brick Warehouse* (Red Brick Systems) [1], [6], [7], [12], [13], [14], [15]

7 Podsumowanie

W architekturze magazynów danych dane pochodzace z wielu heterogenicznych zródel sa integrowane do globalnego schematu magazynu danych. Dane sa przechowywane w magazynie, który z punktu widzenia uzytkownika jest zwyczajna baza danych. Po zaladowaniu danych do magazynu uzytkownicy moga wydawac zapytania do magazynu (analogicznie do zwyczajnej bazy danych), ale, generalnie, nie wolno im modyfikowac danych znajdujacych sie w magazynie, poniewaz takie modyfikacje nie znajdowalyby odzwierciedlenia w zmianach danych zródlowych i moglyby prowadzic do niespójnosci magazynu. Istnieja trzy glówne podejscia do zagadnienia aktualnosci magazynu danych [4]:

- Magazyn jest okresowo rekonstruowany z danych zródlowych. Rekonstrukcja przebiega najczesciej w nocy (gdy z magazynu nie korzystaja zadne aplikacje) i wymaga wylaczenia magazynu. Wada tej metody jest czasochlonnosc oraz fakt, ze dane w magazynie nie odzwierciedlaja aktualnego stanu danych zródlowych.
- Magazyn danych jest okresowo aktualizowany i zmiany, które zaszly w zródlach, sa propagowane do magazynu. Takie podejscie wymaga uaktualnienia znacznie mniejszej ilosci danych. Z drugiej strony inkrementalna aktualizacja jest znacznie trudniejsza a algorytmy wykrywajace zmiany w danych zródlowych sa bardziej skomplikowane.
- Magazyn danych jest aktualizowany natychmiast po wystapieniu zmiany w zródle. Ze wzgledu na ilosc komunikacji i przetwarzania zwiazanego z takim rozwiazaniem, znajduje ono zastosowanie tylko w malych magazynach danych, w których dane zródlowe zmieniaja sie stosunkowo powoli.

Magazyny danych odgrywaja szczególna role w aplikacjach OLAP. Po pierwsze, magazyn danych jest potrzebny do scentralizowania i zorganizowania danych w taki sposób, by efektywnie wspierac aplikacje OLAP. Dane potrzebne do analizy moga byc rozproszone w wielu bazach danych przedsiebiorstwa. Poza tym zapytania OLAP sa zlozone i wykorzystuja duze wolumeny danych, stad ich wykonanie w systemach zorientowanych na przetwarzanie transakcji (czyli systemach OLTP) jest z reguly nieefektywne.

Aplikacje OLAP wykorzystuja wielowymiarowy model danych. W modelu tym dane prezentowane sa jako komórki w wielowymiarowej przestrzeni i nazywane sa miarami. Istnieja dwie podstawowe metody implementacji takiego modelu danych:

- ROLAP – dane sa przechowywane w relacyjnych tabelach ulozonych w schemat gwiazdy. W centrum takiej gwiazdy znajduje sie relacja faktów, przechowujaca nieprzetworzone informacje o miarach, zas wymiary przechowywane sa w relacjach wymiarów.
- MOLAP – dane wielowymiarowe przechowywane sa w wyspecjalizowanych strukturach nazywanych kostkami danych. Dane przechowywane w kostce danych moga byc wstepnie przetworzone, oprócz tego dla takiej kostki moga byc zdefiniowane specjalne operatory, np. rozwijanie i zwijanie wymiarów, obracanie, wycinanie, itp.

W dziedzinie magazynów danych duzego znaczenia nabiera zagadnienie efektywnosci przetwarzania zlozonych i czasochlonnych zapytan, które przetwarzaja duze ilosci danych. W celu poprawy efektywnosci dzialania magazynów danych stosuje sie wiele technik, których celem jest poprawa owej efektywnosci. Naleza do nich m.in. przetwarzanie równolegle, parcelacja danych oraz nowe metody indeksowania, np. omówione w niniejszym artykule indeksy bitmapowe.

8 Bibliografia

[1] Abbey M., Corey M. J., *Oracle8 – A Beginner's Guide*, Osborne McGraw–Hill, 1997, ISBN 0-07-882393-5

[2] Colliat G., *OLAP, Relational, and Multidimensional Database Systems*, SIGMOD Record, Vol.25, No.3, wrzesien 1996

[3] Data Management Strategies, Cutter Information Corp., http://www.cutter.com/itgroup/

[4] Garcia-Molina, H., Ullman, J.D., Widom, J., *Database System Implementation*, Prentice Hall, 2000

[5] Hurwicz, M., *Take Your Data To The Cleaners*, BYTE, styczen 1997

[6] *INFORMIX–OnLine Extended Parallel Server for Loosely Coupled Cluster and Massively Parallel Processing Architectures*, materialy informacyjne firmy Informix, 1997

[7] *INFORMIX–Data Warehouse Servers. The Evolution of Integrated Relational OLAP*, Informix White Paper, 1996, http://www/informix/com/informix/corpinfo/zines/whitpprs/dwserver/dwserver.htm

[8] *Designing the Data Warehouse on Relational Databases*, http://www/informix/com/informix/corpinfo/zines/whitpprs/stg/metacube.htm

[9] Jarke, M., Lenzerini, M., Vassiliou, Y., Vassiliadis, P., *Fundamentals of Data Warehouses*, Springer-Verlag Berlin Heidelberg, 2000

[10] O'Neil P., Graefe G., *Multi–Table Joins Through Bitmapped Join Indices*, SIGMOD Record, Vol.24, No.3, wrzesien 1995

[11] O'Neil P., Quass D., *Improved Performance with Variant Indexes*, materialy konferencyjne SIGMOD, Tuscon, Arizona, USA, 1997

[12] *Bitmapped Indexes in Oracle7*, an Oracle White Paper, 1995

[13] *A Red Brick Systems White Paper*, http://www.redbrick.com/rbs/whitepapers/datawah_wp.html

[14] Sarawagi S., *Indexing OLAP data*, Data Engineering Bulletin 20(1), 1997

[15] *Optimizing Interactive Performace for the Data Warehouse*, http://www.novasys.com.cy/sysprod06.thm

[16] Widom J., *Research Problems in Data Warehousing*, materialy konferencyjne, 4–th International Conference on Information and Knowledge Management (DIKM), 1995

[17] Wrembel R., *Nowe struktury indeksów dla magazynów danych*, materialy konferencyjne III Jesiennej Szkoly PLOUG, Zakopane, 1997