Wydanie II

Adam Pelikant

HURTOWNIE DANYCH

Od przetwarzania analitycznego do raportowania

Helion

Redaktor prowadzący: Grzegorz Krzystek
Projekt okładki: Studio Gravite
Bazę danych wykorzystaną w zapytaniach analitycznych zamieszczonych w książce można znaleźć pod adresem: *ftp://ftp.helion.pl/przyklady/hurda2.zip*

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/hurda2
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-7411-9

Printed in Poland.

Spis treści

Rozdział 3.

Struktura hurtowni danych

Rola hurtowni danych w procesie przetwarzania

Po rozważaniach przeprowadzonych w poprzednim rozdziale pojawia się wątpliwość, czy potrzebne nam są jeszcze jakieś dodatkowe narzędzia do prowadzenia analizy danych. Przecież zakres funkcji proponowany po stronie transakcyjnej jest tak duży. Niestety, nie wyczerpuje to oczekiwań, jakie przed analizami relacyjnych modeli baz danych stawia współczesność. Przede wszystkim należy odrzucić założenie, przyjmowane dotąd niejawnie, że mamy niewyczerpane zasoby sprzętowe. Duża ilość danych z reguły pociąga za sobą dużą ilość żądań dostępu. Dla dużych wolumenów przetwarzanie złożonych zapytań analitycznych wiąże się z dużym zapotrzebowaniem na pamięć. Powoduje to znaczące obciążenie systemu, co może prowadzić do problemów z wydajnością bieżącego przetwarzania operacji transakcyjnych. Prostym rozwiązaniem wydaje się przeniesienie wykonywania zapytań analitycznych na okres najmniejszego obciążenia bazy danych — zwykle pomiędzy 3 a 5 w nocy (nocne marki już śpią, a ranne ptaszki jeszcze nie wstały). Dodatkowe zamknięcie przetwarzania w ramach transakcji z adekwatnie wysokim poziomem izolacji transakcji (read only) powoduje, że wszystkie analizy prowadzone będą dla tego samego zestawu rekordów, np. z godziny 0:00, kiedy przestawiono poziom izolacji transakcji i rozpoczęto transakcję „analityczną". Silnik serwera odciążony, zestaw danych ustalony, liczba funkcji analitycznych wystarczająca — można powiedzieć, że wszystkie warunki zostały spełnione. Niestety, wraz ze wzrostem liczby danych rośnie zarówno zapotrzebowanie na liczbę wykonywanych analiz, jak i czas realizacji każdej z nich i proste zabiegi odciążające serwer mogą być niedostateczne. Ponadto założono dość wyidealizowany zestaw danych wyjściowych. Przede wszystkim przyjęto, że wszystkie dane zgromadzone w systemie transakcyjnym są przydatne w analizie. W praktyce przetwarzaniu analitycznemu podlega tylko niewielka liczba kolumn wyjściowych tabel. Pomijane są najbardziej szczegółowe informacje zawarte w rekordach. Drugim uproszczeniem jest założenie, że wszystkie informacje są wprowadzone poprawnie. Niestety, jakże często w praktyce nie są stosowane słowniki wartości, a istotne wartości atrybutów są wpisywane ręcznie, np. nazwy miast. W przetwarzaniu transakcyjnym powstaje również sporo „śmieci" będących efektem niefrasobliwych działań operatorów: zapisywanie pustych lub niepełnych rekordów, etc. Te dane powinny zostać odfiltrowane i ujednolicone. Najważniejszym jednak problemem pozostaje to, że w przypadku prowadzenia analiz dla dużych przedsiębiorstw rzadko kiedy mamy do czynienia

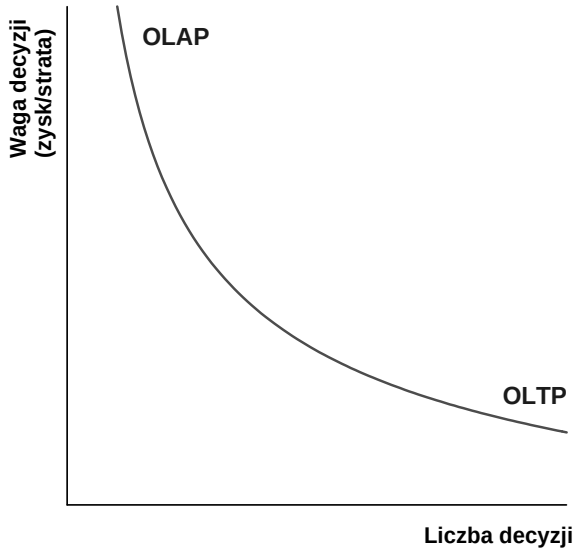
z jednolitym systemem transakcyjnym obowiązującym w całej firmie. Różne działy mają różnie przechowywane dane, inny system obsługuje księgowość, inny sprzedaż, a kolejny jest używany w logistyce. Dane te powinny zostać zintegrowane tak, aby można było dokonywać wspólnych obliczeń dla całej firmy jednocześnie. Pamiętajmy również, że część danych historycznych nie musiała w całości zostać przeniesiona na platformy relacyjne, stąd sporo informacji może pochodzić ze starych baz btrieve'owych lub plików MS Excela. Te wszystkie wyzwania spowodowały konieczność wprowadzenia specjalizowanych narzędzi do analizy danych, jakimi są hurtownie danych. Być może eksperci związani z zarządzaniem lub ekonomią nie są przekonani do tych argumentów, ale „nauka to wiara w ignorancję ekspertów" (Richard Feynman).

Co zatem oferują hurtownie danych? Przede wszystkim:

- fizyczne odseparowanie przetwarzania analitycznego od przetwarzania transakcyjnego (różne serwery);
- postać danych przygotowaną do prowadzenia analiz, zarówno pod względem zawartości, jak i organizacji — perspektywy lub ich materializację do tabel;
- wstępne przetworzenie danych — wyznaczenie podsumowań dla wybranych kolumn;
- model przechowywania danych adekwatny do potrzeb analizy — pozwala to na szybkie wybieranie i filtrowanie danych otrzymanych z analizy;
- tworzenie nowych analiz korzystających z istniejących obliczeń i specjalnego zestawu funkcji i operatorów dedykowanego dla potrzeb hurtowni danych;
- odpytywanie danych za pomocą rozszerzenia języka SQL — MDX SQL (*MultiDimensional eXtension*);
- wizualizację danych uzyskanych po przeprowadzeniu analiz.

Najważniejszym pytaniem, jakie powinniśmy sobie postawić, zanim zaczniemy tworzyć hurtownię danych, jest to, dla kogo ma być ona przeznaczona. W każdym szanującym się przedsiębiorstwie bardzo duża liczba pracowników (zwykle przeważająca) ma dostęp tylko do jakiegoś fragmentu bazy danych. Pracownicy najniższych szczebli będą zapewne mieli dostęp tylko do odczytu wybranych danych. Do pewnego poziomu — im wyżej w hierarchii, tym większe możliwości dostępu do danych i manipulowania nimi. Załóżmy, że opisujemy strukturę dostępu do danych w banku, a podstawowym poziomem jest operator (kasjer). Oczywiście są w tej strukturze również osoby o niższym dostępie do danych, np. ochrona nie ma wglądu w transakcje, ale może mieć wgląd w dane dotyczące informacji, który operator kiedy pracuje. Nasz operator wykonuje w czasie pracy ogromną liczbę transakcji, podejmując przy tym bardzo dużą liczbę decyzji — czy dokonać wypłaty i do jakiej kwoty, czy można przyjąć depozyt, etc. Decyzje te są oczywiście ściśle określone wewnętrznymi uregulowaniami banku i nasz operator ma znikomą szansę na ich interpretację. Ważne jest, że pomimo bardzo dużej liczby decyzji ich oddziaływanie na funkcjonowanie firmy jest niewielkie. Również ewentualne skutki podjęcia niewłaściwej decyzji nie zaważą na ogólnej sytuacji banku. Patrząc na kolejne szczeble w strukturze organizacyjnej, widzimy o wiele szerszy dostęp do danych, np. możliwość sprawdzania całej historii rachunków łącznie z informacją dotyczącą źródeł, z których wpływały przelewy, debetów, etc. Na tym szczeblu mogą być podejmowane decyzje o udzielaniu wysokich kredytów lub zakłada-

niu specjalnych lokat czy kont. W tym miejscu liczba podejmowanych decyzji jest zdecydowanie mniejsza, rośnie samodzielność decydowania, ale skutki popełnienia błędu są również poważniejsze. Pomimo to nie powinny być one niebezpieczne dla firmy jako całości. Podjęcie właściwej decyzji może owocować poważnym zyskiem. Na poziomie zarządczym jednostki lub grupy paradoksalnie maleje zapotrzebowanie na dostęp do szczegółowych danych. Prawdopodobnie pracownicy tego szczebla mają potencjalny do nich dostęp, jednak trudno sobie wyobrazić, że własnoręcznie wykonują transakcje lub przeglądają szczegółowe dane pojedynczego klienta. Bardziej są zainteresowani danymi zbiorczymi, np. tym, jakie są wyniki finansowe z rozbiciem na rodzaj produktu — dla różnych kredytów, lokat; która grupa klientów przynosi największe dochody i jakimi produktami można by ją zainteresować; jak kształtowała się sytuacja finansowa jednostki na przestrzeni czasu. Tutaj liczba podejmowanych decyzji nie jest duża, ale potencjalne błędy mogą być odczuwalne. Tak samo jak podjęcie poprawnej decyzji może przynieść naprawdę duży zysk. Jeśli przeniesiemy się na najwyższy, centralny szczebel w hierarchii, zarząd całej firmy (wszystkich banków jednostkowych), to zauważymy, że nie ma już tam zapotrzebowania na dostęp do danych szczegółowych. Natomiast znacznie rośnie rola przeprowadzanych analiz. To one są podstawą podejmowania najważniejszych w skali przedsiębiorstwa decyzji strategicznych, np. dotyczących oprocentowania poszczególnych produktów, zakresu i adresatów kampanii reklamowej, ogólnych zasad prowadzenia rachunków i udzielania kredytów. Liczba podejmowanych decyzji jest niewielka, natomiast ich waga znacząca. Zarówno skutki błędów, jak i poprawnych działań mogą decydować o losach firmy. Ten schematyczny rozkład liczby decyzji i ich konsekwencji dla firmy przedstawia rysunek 3.1. Zaznaczono na nim miejsce systemów transakcyjnych, *OLTP* (*On Line Transactional Processing*), oraz analitycznych, *OLAP* (*On Line Analytical Processing*). W części środkowej zasięg obu mechanizmów przetwarzania może być zmienny, w zależności od sposobu organizacji firmy. Może również istnieć zakres, w którym istnieje równoległy dostęp do nich obu.



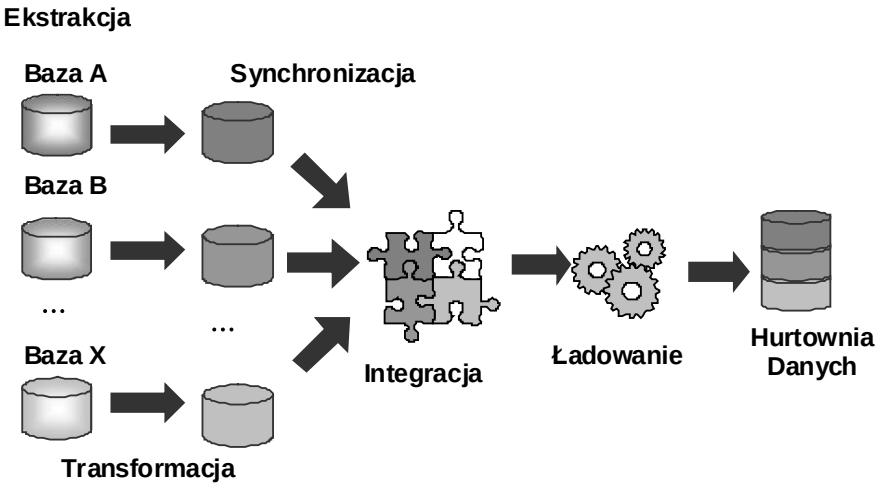
Rysunek 3.1. *Miejsce przetwarzania analitycznego i transakcyjnego w zależności od wagi i liczby decyzji*

Można zatem powiedzieć, że przetwarzanie transakcyjne odpowiada za bieżącą obsługę klienta (firmy). Miernikiem oceny jest łatwość przetwarzania, szybkość dostępu do danych mierzona liczbą transakcji w jednostce czasu. Drugim aspektem jest pewność przechowywania danych i zapewnienie dostępu do nich tylko autoryzowanym użytkownikom (ochrona danych). Oba kryteria są obiektywne i łatwe do sprawdzenia i porównania.

W przypadku przetwarzania analitycznego *OLAP* kryterium jest jakość podejmowanych decyzji, które są skutkiem przeprowadzanych analiz. Ponieważ na różnym szczeblu zakres podejmowanych decyzji jest różny, zestaw prowadzonych analiz musi być do nich dostosowany. W związku z tym hurtownie danych, a przynajmniej dostęp do ich różnych części, muszą być dedykowane do poziomu w hierarchii zarządzania firmą. Narzędzia wykorzystujące *OLAP* są siłą rzeczy rozwiązaniami dedykowanymi. Ponieważ o skutku podjętej decyzji możemy przekonać się dopiero ex post, rozwiązania analityczne muszą być modyfikowane zgodnie z wynikami tych działań, czyli powinny ewoluować wraz z rozwojem firmy. Niestety, postawione kryterium oceny trudno nazwać obiektywnym, gdyż jeśli jakieś działanie przyniosło zysk, to nie da się wykluczyć, że inne spowodowałoby jeszcze lepszy wynik. Podobnie w przypadku straty możliwe jest, że istnieje działanie lepsze, dające wynik pozytywny lub przynajmniej mniej negatywny. Stąd proces udoskonalania rozwiązań analitycznych musi być oparty na doświadczeniu, intuicji i nie da się go w pełni zobiektywizować, a tym samym zoptymalizować pod względem formalnym.

Proces integracji danych

Jak już wspomniano, pierwszym etapem budowy hurtowni, poprzedzającym wykonywanie jakichkolwiek analiz, jest zasilenie jej danymi (rysunek 3.2). Całość tego procesu określana jest skrótem *ETL* (*Extract, Transform, Load*). W większości przypadków problem wywodzi się z prawdziwego założenia, że w dużych organizacjach dane do analiz pochodzą z wielu źródeł, które mają niejednolitą strukturę. Pierwszym etapem jest oczywiście wskazanie tych źródeł. Zazwyczaj należy wskazać odpowiedni sterownik, most (*getway*) dostępu do danych, np. dedykowane rozwiązania: *ODBC*, *OLE DB*, *JDBC* itd. Dla każdego z nich trzeba zdefiniować sposób połączenia — łańcuch połączeniowy (*connection string*) — oraz sposób i dane potrzebne do autoryzacji na wskazanym serwerze. Kolejny krok to wskazanie obiektów (tabel), które zawierają dane potrzebne do przeprowadzenia analizy. Ponieważ rzadko mamy do czynienia z tabelami tak zorganizowanymi po stronie relacyjnej, że zawierają tylko takie dane, a ponadto ich postać nie jest dopasowana do potrzeb analiz (część danych jest obliczana na podstawie kilku pól, np. *wartość=cena*ilość*, opis osoby zawiera połączenie nazwiska z imieniem itd.), zwykle budowane są odpowiednie perspektywy. Pobranie danych z perspektywy wymaga wykonania zapytania, które ją definiuje, a które może być złożone, zatem często stosuje się ich materializację do nowych tabel, bezpośrednio wykorzystywanych w analizie, co prowadzi do poprawy wydajności zasilania hurtowni.



RYSUNEK 3.2. *Proces ETL i jego elementy składowe oraz miejsce hurtowni danych w przetwarzaniu danych*

Ponieważ dane mogą być zapisywane w różnej postaci, należy je ujednolicić. Najprostszym przykładem jest pole daty, które nie ma standardowego formatowania i jest zależne od ustawień narodowych, ale również może być definiowane przez programistę. Najprostszym zabiegiem jest przekształcenie daty na wewnętrzną postać numeryczną, a następnie zwrotnej konwersji na postać zgodną z jednym wybranym formatowaniem. Taki zabieg powinien być przeprowadzony po stronie serwera relacyjnego, ponieważ dla różnych producentów oprogramowania typ daty ma różną datę początkową, od której jest wyznaczana wewnętrzna liczbowa reprezentacja (Oracle — 1 stycznia 4712 p.n.e., MS SQL Server — datatime: 1 stycznia 1753, smalldatatime: 1 stycznia 1900). Jest jeszcze gorzej — jeśli dane tego typu zapisywane są w polach tekstowych, możemy mieć do czynienia z pełną dowolnością zapisu, różnorodną w pojedynczej bazie lub nawet tabeli. Powoduje to konieczność budowania specjalnego oprogramowania do parsowania i formatowania tych danych. Na szczęście coraz rzadziej mamy z tym do czynienia. Podobne kłopoty mogą nas spotkać w przypadku liczb rzeczywistych: różne separatory dziesiętne, stosowanie separacji tysięcy (z reguły *Chr(162)* — unbreakable space, a nie, jak się zwykło sądzić, *Chr(32)* — spacja), a w przypadku waluty — odmienny zapis jej symbolu. Dużym problemem jest ujednolicenie zapisu danych tekstowych, które nie były pobierane ze słowników wartości. Dla każdego zapis *Łódź*, *Lodz* czy *Uć* (najkrótsza nazwa miasta w Polsce) jest na pierwszy rzut oka nazwą tego samego miasta — chociaż zniekształconą. O ile trzeci wariant nie będzie występował, poza dowcipami z brodą, o tyle nie da się wykluczyć błędów w zapisie, spowodowanych nieuwagą, nieumiejętnością wprowadzenia polskich znaków diakrytycznych czy też wynikających z nieznajomości ortografii. Usunięcie błędów zapisu wydaje się łatwe, kiedy mamy tę operację wykonać ręcznie. Natomiast napisanie oprogramowania, które będzie automatyzować ten proces, jest zadaniem naprawdę złożonym, zwłaszcza że istnieje wiele podobnie brzmiących nazw, jak choćby województwa lubelskie i lubuskie, które w przypadku błędów w zapisie mogą się jeszcze bardziej upodobnić. Należy zauważyć, że informacja geograficzna jest często

podstawą wykonywania analiz. Dodajmy do tego jeszcze fakt, że dane po stronie relacyjnej są często zaśmiecone w efekcie niedokończonych działań operatorów — niedokończone wpisy w rekordach, nieusunięte puste transakcje. Błędy mogą pojawiać się również (niewspółmiernie rzadziej) na skutek usterek przetwarzania, niezależnych od operatora, pojawiających się na poziomie aplikacji obsługującej bazę danych czy też po stronie serwera danych. Proces oczyszczania danych pomimo pozorów prostoty jest jednym z najżmudniejszych i najbardziej kłopotliwych zadań przygotowywania danych.

Kolejnym etapem jest synchronizacja danych pochodzących z różnych źródeł. Może to być związane z ujednoliceniem indeksowania kluczy tabel zarówno słownikowych, jak i zawierających informacje o przeprowadzanych operacjach, eliminacją powieleń wartości w słownikach, etc. Po poprawnie wykonanym oczyszczaniu proces ten powinien być względnie mało skomplikowany. Następnie dane te mogą zostać zintegrowane do wspólnej postaci, np. do pośredniego schematu relacyjnego utworzonego na odseparowanej od wyjściowych źródeł instancji serwera BD.

Pozostaje jeszcze proces ładowania danych do schematu, który będzie bezpośrednio wykorzystywany przez hurtownię. Trzeba zaznaczyć, że ze względu na fakt, iż analizy prowadzone są na bardzo dużych wolumenach danych, proces ładowania może być dość długotrwały. Cały proces *ETL* należy do najbardziej czasochłonnych operacji wśród rozważanych przez nas zagadnień. Ponieważ dane po stronie baz transakcyjnych cały czas przyrastają, są modyfikowane, proces zasilania hurtowni też powinien odbywać się cyklicznie. Najgorszym pomysłem jest ponawianie za każdym razem całego cyklu wraz z niezmienionymi po stronie transakcyjnej, wcześniej załadowanymi rekordami. Warto rozważyć proces zasilania różnicowego, co wiąże się ze znakowaniem rekordów wcześniej przesłanych lub replikowaniem danych albo ze stosowaniem kopiowania różnicowego z wykorzystaniem polecenia *MERGE* czy też kursorów.

Elementy hurtowni danych

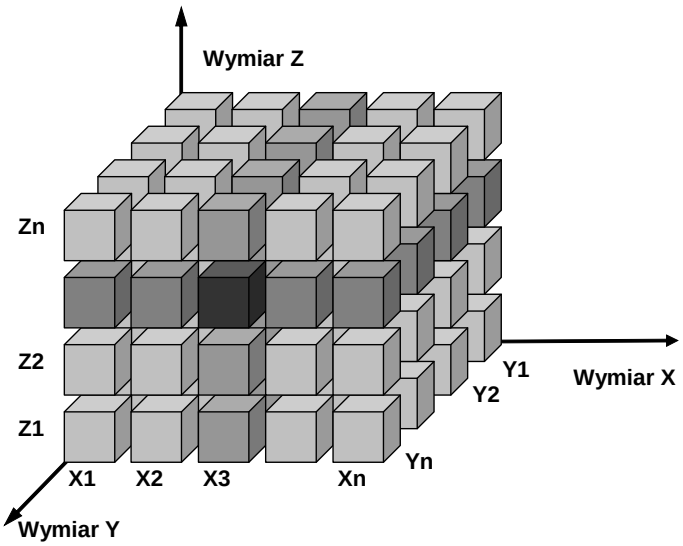
W końcu jesteśmy po stronie hurtowni danych, gdzie przechowywane są wstępnie przetworzone dane. Dane te mogą mieć w dalszym ciągu postać relacyjną *ROLAP* (*Relational OnLine Analytical Processing*) lub najczęściej stosowaną postać wielowymiarową *MOLAP* (*Multidimensional OnLine Analytical Processing*). Dopuszczalne jest przechowywanie i przetwarzanie struktur mieszanych *HOLAP* (*Hybrid OnLine Analytical Processing*). Z reguły jednak myślimy o hurtowni danych jako pewnej strukturze wielowymiarowej, co z reguły przyprawia wiele osób o dreszcze, gdyż jak powiedział Albert Einstein, „gdy niematematyk słyszy o czwartym wymiarze, to dostaje gęsiej skórki". Od wersji 2016 został wprowadzony model tabelaryczny (*Tabular model*), który podczas standardowej instalacji jest wskazywany jako domyślny. Możemy traktować go jako spłaszczenie kostki przez wyznaczenie iloczynu kartezjańskiego na wszystkich wymiarach. Zasadniczą cechą tego rozwiązania jest umożliwienie przetwarzania „*in-memory*", co powinno mieć znaczący wpływ na wydajność w przypadku na tyle niewielkich danych, które mogą zostać w całości załadowane do pamięci. W przypadku większych danych stosuje się mechanizm

DirectQuery odwołujący się bezpośrednio do tabel, który nie wydaje się być aż tak wydajny. Jednak jak sami piszą twórcy *SQL Server Analysis Services*, głównym motywem wprowadzenia tego modelu było zapewnienie większej intuicyjności dedykowanej deweloperom w dostępie do przetworzonych danych. Jednak i w tym modelu posługujemy się pojęciami wymiaru i miary, czyli „klątwa Einsteina" działa również w tym przypadku.

Górne ograniczenie liczby wymiarów wynosi z reguły kilkadziesiąt i jest zależne od konkretnego środowiska. Zgodnie z dokumentacją MS w tym rozwiązaniu liczba wymiarów jest nieograniczona. W tym kontekście można przytoczyć anegdotę dotyczącą Stefana Banacha, wybitnego specjalisty między innymi z zakresu topologii. Rozpoczynając wykłady z nowym rocznikiem studentów, mówił tak:

> *— Wyobraźcie sobie dziesięciowymiarową, ortogonalną przestrzeń metryczną — patrzy na studentów. — No dobrze, wyobraźcie sobie pięciowymiarową, ortogonalną…*
> *Rozumiem, wyobraźcie sobie trójwymiarową przestrzeń euklidesową — znów rozgląda się z wyraźną rezygnacją. — Dobrze, weźmy kartkę papieru.*

Jesteśmy w o wiele trudniejszej sytuacji, gdyż ograniczenie się do dziesięciu wymiarów, ilustrowanych osiami liczbowymi, jak proponował Banach, jest bardzo dużym uproszczeniem tego, z czym mamy do czynienia na co dzień, analizując struktury hurtowni danych. Cóż, możliwości postrzegania przez człowieka ograniczają się do trzech wymiarów i tyle jesteśmy w stanie z wykorzystaniem perspektywy narysować na kartce papieru. Dlatego też kolejne ilustracje ograniczają się do takiej postaci. Podstawową strukturę możemy zatem przedstawić tak jak na rysunku 3.3.

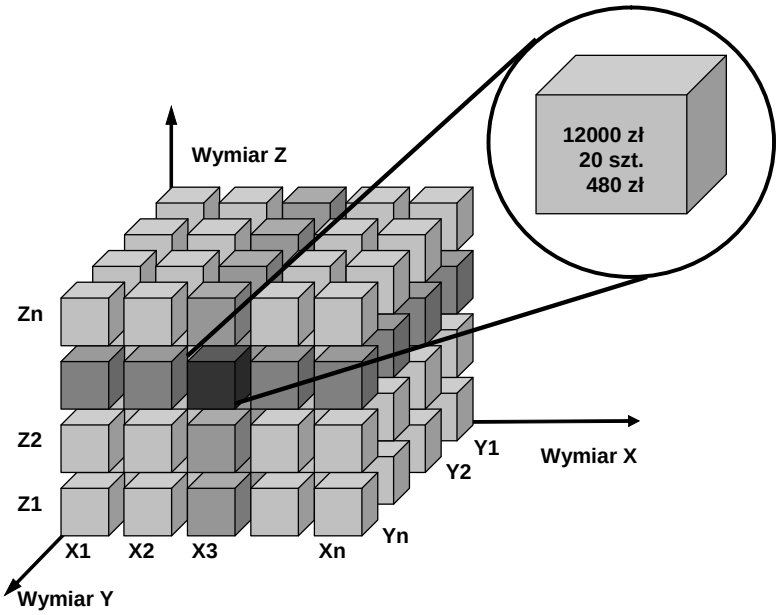


RYSUNEK 3.3. *Podstawowa struktura hurtowni danych — uproszczona do trzech wymiarów*

Osie współrzędnych reprezentują wymiary, z reguły opisane przez wartości dyskretne. Dlatego podobieństwo do sześcianu narysowanego w układzie kartezjańskim jest dość zwodnicze. Co prawda możemy przyjąć, że wartości atrybutów na osiach są w jakiś sposób uporządkowane, ale nie stanowią osi liczbowych. Przykładem takiego wymiaru, często stosowanego w praktyce, może być położenie geograficzne, określające np. lokalizację jednostek naszej firmy przez podanie miasta: Wrocław, Łódź, Warszawa, Olsztyn, Szczecin. Jak widać, zostało wykorzystane uporządkowanie w kierunku narastającej wartości szerokości geograficznej (od południa do północy geograficznej). Takie wartości, atrybuty, względem których dokonujemy sortowania, nazywamy regresorem. Intuicja podpowiada, że wybór sposobu rozmieszczenia atrybutów na osi może mieć wpływ na rezultaty analiz, chociażby przy porównywaniu wyników dwóch sąsiadów. Może jednak w naszym przykładzie zmienić układ atrybutów i posłużyć się ich długością geograficzną ze wschodu na zachód? W realiach Polski ma to uzasadnienie zarówno historyczne, jak i gospodarcze. Nasza oś miałaby wtedy postać: Olsztyn, Warszawa, Łódź, Wrocław, Szczecin. Możemy również zastosować jako regresor liczbę pracowników każdego z oddziałów (lub wybranego działu tego oddziału, np. zbytu, handlowego, etc.). Takie uporządkowanie ma równie silne, a może i silniejsze niż poprzednie umocowanie w praktyce — większe jednostki powinny przynosić większe zyski. Jak widać, wybór sposobu organizacji osi może mieć różne uwarunkowania i prowadzić do innej postaci wymiaru. Innym przykładem często spotykanego wymiaru jest oś produktów. Trzymając się przykładu bankowego, możemy mieć tu atrybuty o wartościach: lokata, kredyt, karta, etc. Tak jak poprzednio, trudno jednoznacznie określić sposób organizacji osi. Wydaje się, że na tym tle dość dobrze prezentuje się oś opisująca kolor, w przypadku banku być może mało praktyczna, ale jeśli wybierzemy wartości: bronze, silver, gold, platinum, to mamy określenie rodzaju karty kredytowej lub płatniczej. Poza stopniowaniem zgodnym z wartością kruszcu możemy również stosować jako regresor długość fali odpowiadającej danej barwie. Ale znając pomysłowość handlowców, czasami trudno określić, z czym mamy do czynienia. Proszę pomyśleć, cóż to za kolor „noc Kairu". Jest to przykład wzięty z życia — miałem kiedyś samochód w tym kolorze. W przypadku kolejnego auta mój kolega, profesor ASP, był łaskaw powiedzieć, że jest w kolorze szampańskim i wtedy naprawdę zwątpiłem w swoją zdolność postrzegania barw. Ale przecież mamy również wymiary reprezentowane przez wartości numeryczne. W tym przypadku sprawa wydaje się zupełnie prosta. Wymiar jest w sposób naturalny uporządkowany i nie powinien nastręczać żadnych problemów. Jednak jeśli założymy, że atrybutem jest cena towaru, to w zasadzie trudno będzie nam znaleźć dwa towary o takiej samej wartości z dokładnością co do grosza. W wyniku będziemy mieli oś, która de facto dubluje informacje o towarach, na dodatek trudną do przeanalizowania, ponieważ odczytanie nazwy towaru na podstawie ceny wiąże się z dodatkowym przeszukiwaniem bazy. Analiza sprzedaży dla dokładnej ceny produktu też wydaje się mało celowa, zwłaszcza że dany towar mógł na przestrzeni czasu zmieniać swoją cenę. Dlatego atrybuty numeryczne rzadko kiedy reprezentowane są przez dokładne wartości; znacznie częściej są reprezentowane przez ich przedziały, które są etykietowane napisami objaśniającymi ich sens (w dziedzinie cen: bardzo tanie, tanie, przeciętne, drogie, bardzo drogie), co powoduje przejście do atrybutów symbolicznych, dyskretnych. Procesy dyskretyzacji i etykietowania (dobór właściwych słowników) są zadaniami dość złożonymi i zostaną precyzyjniej wyjaśnione w omawianych dalej rozwiązaniach praktycznych.

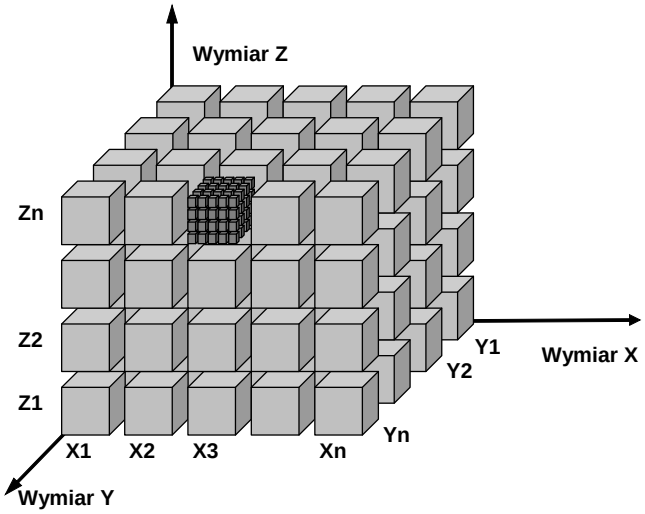
W praktyce najczęściej występuje oś daty (czasu), która w wewnętrznej reprezentacji każdego systemu komputerowego ma postać liczby. Tak samo jak w przypadku innych wymiarów numerycznych, problemem nie jest uporządkowanie, ale dyskretyzacja. Na szczęście tutaj możemy mówić o naturalnych poziomach granulacji: rok, półrocze, kwartał, miesiąc, tydzień, dzień, a później podobnie w odniesieniu do czasu. Dodatkowo poziomy te stanowią naturalną hierarchię — rok składa się z półroczy, rok i półrocze z kwartałów itd. Musimy ustalić, czy wszystkie one są dla nas interesujące oraz na jakim najniższym poziomie zakończyć podział. Niestety, pojawia się również i drugi problem, który wyraźnie widać przy tygodniach. Miesiąc nie składa się z pełnych tygodni (jest to możliwe tylko dla lutego w latach nieprzestępnych), co zaburza mechanizm pełnego zagnieżdżania się poziomu podrzędnego w nadrzędnym. Znów musimy podjąć arbitralną decyzję, co zrobić z początkami i końcami tygodni nie w pełni należącymi do miesięcy. Jak widać z przykładów, proces tworzenia wymaga podejmowania wielu decyzji, które muszą być oparte tylko na doświadczeniu projektanta hurtowni danych oraz na rozpoznaniu potrzeb środowiska, dla którego ta hurtownia jest przeznaczona.

Jeśli zdefiniowaliśmy już wymiary w ilustracji graficznej, trzy (x, y, z) to wybór wartości atrybutu, np. x_3 definiuje przekrój (plaster kostki), do którego będziemy mieli szybki dostęp — rysunki 3.3 i 3.4. Podobnie w przypadku atrybutu z_i definiowany jest przekrój dla ustalonej wartości tego atrybutu. Ponadto możemy w ten sposób odwołać się do części wspólnej dla pary wartości niezależnych atrybutów. Wybór trzech wartości x_i, y_j, z_k prowadzi w przypadku trzech wymiarów do wskazania konkretnej komórki, w której są przechowywane wartości funkcji agregujących wyznaczonych w trakcie przetwarzania. Powoduje to, że odzyskanie wartości analiz dla płatów, ich przecięć czy konkretnych komórek jest bardzo szybkie. Podstawowymi funkcjami agregującymi stosowanymi podczas definiowania kostki są suma i liczebność elementów (*SUM*, *COUNT*). Obie te funkcje są tzw. funkcjami addytywnymi. Oznacza to, że jeśli zostały wyznaczone dla pewnej liczby danych, to po ich powiększeniu (przesłanie kolejnej porcji z systemu transakcyjnego do hurtowni) nie ma konieczności przeliczania od początku całej struktury, tylko do policzonych uprzednio wartości dodajemy wartości uzyskane z przetworzenia nowych danych. Możemy, odwołując się do przykładu bankowego i rysunku 3.4, przyjąć, że w oddziale z_k (Zgierz) w miesiącu x_i (marcu) sprzedano produkty y_j (karty kredytowe) za łączną sumę 12 000 zł, a transakcji było 20. W środowisku Analysis Services nie definiuje się innych funkcji agregujących, stąd wartość średnia musi być obliczana definicyjnie ($\underline{x} = \Sigma x_i/n$). Dokładnie rzecz ujmując, istnieje funkcja *AVG*, ale nie jest wyznaczana na wyjściowym zestawie wartości, lecz na ich podsumowaniach — średnia z sumy. Na rysunku 3.4 trzecią wartością jest właśnie wartość średnia. O funkcjach, które da się wyznaczyć na podstawie wzoru, w którym występują tylko funkcje addytywne, mówimy, że są quasi-addytywne. Również one nie wymagają przeliczenia całej kostki. Zastosowanie wyrażenia pozwalającego na wyznaczenie innych wartości w komórkach wiąże się z utworzeniem tzw. miary kalkulowanej (obliczanej). Niestety część takich wyrażeń nie jest addytywna ani quasi-addytywna. Używanie ich wiąże się z pogorszeniem wydajności i należy stosować je roztropnie.



RYSUNEK 3.4. *Podstawowa struktura hurtowni danych — zawartość pojedynczej komórki*

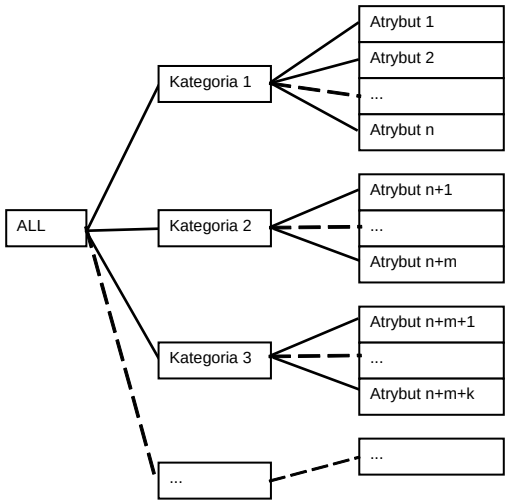
W poprzednich rozważaniach założono, że każdy wymiar ma jeden poziom granulacji. Jednak praktyka mówi, że większość atrybutów może mieć podpoziomy, stąd każdy z nich może zostać rozwinięty i pokazywać poziom niższy. Ilustruje to rysunek 3.5, gdzie jedna z komórek została przedstawiona w postaci podkostki, co ma symbolizować właśnie taki stan. Oznacza to, że komórka ta zawiera nie tylko podsumowania dla właściwego poziomu, ale również wskazanie na wszystkie elementy poziomów pochodnych. Ta sama ilustracja może też symbolizować to, że dane do hurtowni niekoniecznie muszą być pobierane bezpośrednio ze schematu relacyjnego, ale mogą pochodzić także z mniejszych, tematycznych hurtowni danych, dawniej nazywanych składnicami danych (*Data Marts*). Podzielam pogląd, że dla rozwiązań dedykowanych czy też ich połączeń należy stosować nazwę „hurtownia danych", dodatkowo określając tylko jej zakres. Jeśli łączymy wiele rozwiązań tematycznych, to raczej dane z jednego z nich będą zawierały jeden płat, np. wszystkie agregacje dla jednego oddziału lub przynajmniej większy jego fragment, np. dla zestawu produktów.



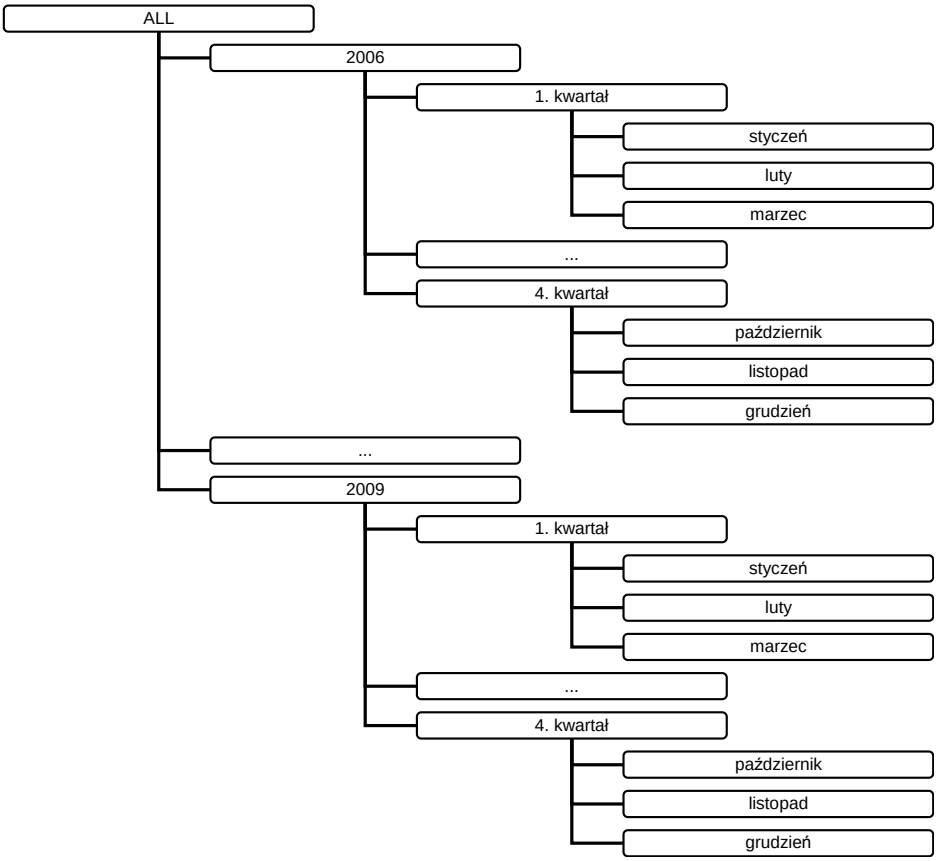
RYSUNEK 3.5. *Idea przechowywania danych w strukturze wielowymiarowej, definiowania przekrojów i drążenia danych*

Powróćmy do problematyki definiowania wymiarów. Wiele atrybutów, nawet symbolicznych, ma bardzo dużą liczbę wartości. Opłaca się wtedy połączyć je w grupy. Zwykle wykonuje się to, tworząc hierarchię, gdzie wyższy poziom wyznacza pogrupowane atrybuty. Możliwe jest tworzenie sztucznych poziomów agregacji, opierając się na równej liczebności grupy lub jawnie podając liczbę grup. Najczęściej jednak jest tak, że możliwe jest wyznaczenie logicznego poziomu nadrzędnego (rysunek 3.6). Towary możemy połączyć w kategorie towarów. Oczywiście takich poziomów możemy wyznaczać wiele, np. kategorie, podkategorie, grupy, towary. Liczba poziomów w hierarchii nie jest ograniczona, ale musi być w tym przypadku dana jawnie. Zwykle wymiary uzupełniane są na każdym poziomie o atrybut reprezentujący nieznane (najczęściej *null*) wartości na każdym poziomie — *unknown*.

Każdy z wymiarów może mieć zdefiniowanych wiele hierarchii. Naturalnym przykładem drzewiastej struktury jest wymiar daty i czasu omawiany poprzednio (rysunek 3.7). W tym przypadku również mamy do czynienia z wieloma poziomami, przy czym ich liczba może być naprawdę duża: rok, półrocze, kwartał, miesiąc, dzień. Co więcej, możemy definiować różne częściowo niezależne hierarchie dla tego samego wymiaru: rok, trymestr, miesiąc; rok, miesiąc, dzień; rok, tydzień, dzień. Jak widać, hierarchie te mogą mieć wspólne poziomy, mogą stanowić podzbiór już istniejącej hierarchii. Zawsze jednak mają zdefiniowaną podczas ich tworzenia liczbę poziomów. Każdy wymiar hierarchiczny ma zdefiniowany obiekt nadrzędny o domyślnej nazwie *ALL*, który stanowi korzeń drzewa (*ROOT*). Wskazuje on na podsumowania dla wszystkich elementów danego wymiaru.

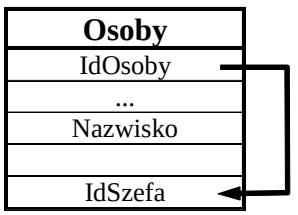


RYSUNEK 3.6. *Hierarchiczna struktura wymiaru o dwóch poziomach*



RYSUNEK 3.7. *Przykład struktury hierarchicznej dla wymiaru daty*

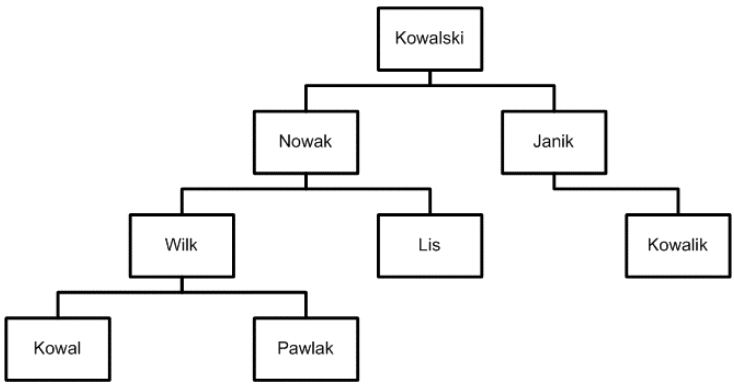
Innym przykładem hierarchii są hierarchie oparte na relacji rodzic — potomek. Rozważmy przykład tabeli *Osoby*, w której kluczem głównym jest pole *IdOsoby* (rysunek 3.8). Każdy pracownik ma swojego przełożonego — szefa, którego wskazuje klucz obcy *IdSzefa*. Każdy szef jest również pracownikiem firmy posiadającym własny identyfikator *IdOsoby*. Dlatego w polu *IdSzefa* zapisywane są wskazania na pole *IdOsoby* właściwego przełożonego. Mówimy tutaj o samozłączeniu, gdyż relacja „jeden do wielu" dotyczy pól tej samej tabeli. Taka sama struktura jest w stanie opisać dowolny graf. Mówimy o notacji krawędziowej, ponieważ para wartości *IdOsoby*, *IdSzefa* występująca w tym samym rekordzie pokazuje krawędź grafu między tymi dwoma węzłami.



IdOsoby	Nazwisko	IdSzefa
1	*Kowalski*	
2	*Nowak*	*1*
3	*Janik*	*1*
4	*Wilk*	*2*
5	*Lis*	*2*
6	*Kowal*	*4*
7	*Pawlak*	*4*
8	*Kowalik*	*3*

RYSUNEK 3.8. *Definicja samozłączenia w tabeli Osoby — krawędziowa reprezentacja grafu*

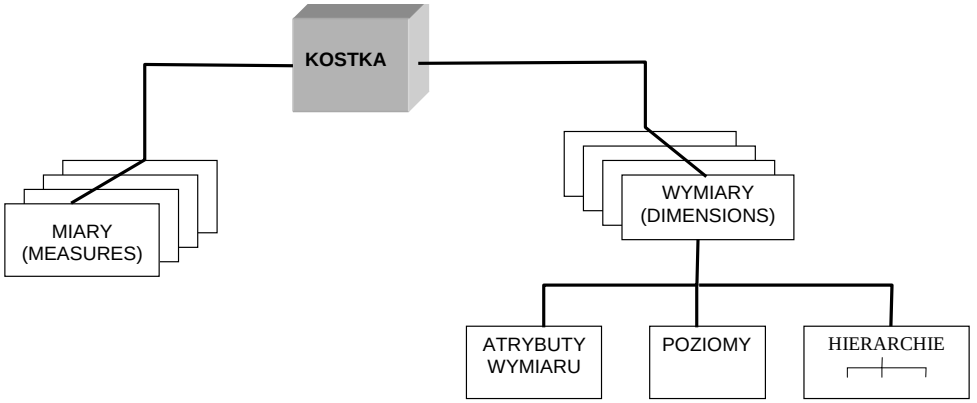
W celu wyznaczenia najbardziej nadrzędnego elementu — szefa wszystkich szefów (capo di tutti capi) — zwykle podajemy w kluczu obcym tego rekordu wartość *NULL*. Również tego typu hierarchie mogą stanowić osie wielowymiarowej struktury hurtowni danych. Ciekawą ich cechą jest to, że nie posiadają ściśle określonej liczby poziomów. Ich liczba wynika z rzeczywistych danych, a każda reorganizacja może spowodować zmianę tej struktury łącznie z liczbą poziomów. Przykład hierarchii rodzic — potomek przedstawia rysunek 3.9.



RYSUNEK 3.9. *Drzewiasta struktura wymiaru rodzic — potomek; odwzorowanie tabeli z rysunku 3.8*

Samozłączenie może występować na wielu poziomach. W sferze organizacji przedsiębiorstwa możemy definiować wzajemną podległość różnych jednostek działów. Nadrzędnym działem będzie np. dyrekcja, której bezpośrednio podlegają działy administracji i techniczny. Natomiast dział handlowy jest podporządkowany administracji. Może dojść do ciekawej sytuacji, kiedy założymy, że szefem pracownika może być tylko osoba z tego samego działu lub działu, który występuje bezpośrednio nad nim w strukturze podległości działów. Z reguły tak jest w praktyce. Mamy tutaj do czynienia ze zjawiskiem synchronizowania hierarchii (drzew). Pomimo że takie wymiary mają swoje uzasadnienie praktyczne, nie mają jeszcze, ze względu na swoją złożoność, realizacji praktycznej w żadnym komercyjnym rozwiązaniu hurtowni danych.

Możemy już dokonać pierwszego podsumowania wiadomości o strukturach, które będziemy za chwilę tworzyć. Podstawą jest wielowymiarowa struktura (kostka) zawierająca podsumowania w każdej z elementarnych komórek wyznaczone podczas przetwarzania (rysunek 3.10). Atrybuty zawierające te podsumowania noszą nazwę miar (*MEASURES*). Dla kostki może być wyznaczonych wiele miar, które są zwykle podsumowaniem. W większości przypadków wyznaczana jest również miara reprezentująca liczbę jednostek, obliczana przez zliczenie wystąpień klucza podstawowego tabeli zawierającej dane do sumowania. Miary mogą być też wyznaczane na podstawie wyrażeń; nazywane są wtedy miarami kalkulowanymi (obliczanymi).



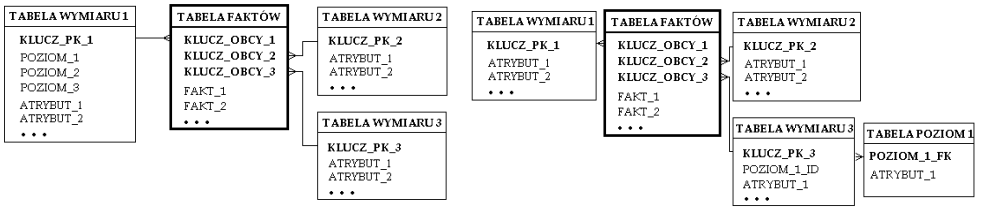
RYSUNEK 3.10. *Poziomy definiowania miar i wymiarów w hurtowni danych*

Krawędzie kostek są wyznaczane przez wymiary (*DIMENSIONS*). Wymiary mogą być opisywane zarówno atrybutami opisowymi (dyskretnymi), które wymagają ustalenia sposobu porządkowania, jak i numerycznymi (ciągłymi), które z reguły muszą być dzielone na dyskretne zakresy, przedziały. Atrybuty są często grupowane w poziomy, a te z kolei mogą wyznaczać dla każdego wymiaru wiele hierarchii. Jak widać, problemy ze zrozumieniem wielowymiarowości reprezentowanej przez układ ortogonalnych osi liczbowych jest drobnostką w porównaniu ze strukturą wielowymiarową, z którą przyszło nam się mierzyć. Na marginesie można dodać, że w hurtowniach wymiary nie muszą być ortogonalne czy niezależne liniowo. Można sobie wyobrazić atrybuty częściowo skorelowane, wyznaczające dwa różne wymiary. Takie podejście wymuszane będzie przez oczekiwanie odbiorców albo, co jest równie częste, na skutek braku możliwości wyeliminowania zależnych wymiarów przed przeprowadzeniem analizy. Możemy

dokonać tylko korekty a posteriori. Praktyczne wymagania analiz bywają bardzo złożone. No cóż, „rzeczywistość nie zawsze jest tym, czym się wydaje" (Terry Pratchett (Mort)). Co dają nam takie komplikacje formalne i pojęciowe? Ponieważ w każdej komórce mamy już przygotowane policzone dane, a wskazanie ich odbywa się przez wybranie odpowiedniego atrybutu, poziomu lub ich zbioru, pozyskiwanie danych jest dużo szybsze niż w przypadku bezpośredniego odpytywania struktur relacyjnych.

Przyjrzyjmy się strukturze relacyjnej odwzorowującej na poziomie warstwy logicznej, a także na poziomie pośredniej warstwy relacyjnej strukturę wielowymiarową przed jej przetworzeniem (rysunek 3.11). Centralną jej część stanowi zawsze tabela faktów, która zawiera wszystkie wielkości stanowiące podstawę do wyznaczania miar, czyli fakty. Wartości tych atrybutów muszą być numeryczne ze względu na wykonywane na nich później obliczenia (w praktyce również w przypadku funkcji *COUNT*, która radzi sobie z atrybutami innych typów). Poza faktami w tabeli tej muszą być zdefiniowane klucze obce, służące do połączenia z faktami tabel reprezentujących wymiary. W większości środowisk tabela ta musi zawierać klucz podstawowy. Jeśli nie jest to klucz fizyczny (mający swój odpowiednik w tabeli relacyjnej), to tworzony jest klucz logiczny — identyfikator wiersza w tabeli faktów. Za pomocą własnych kluczy podstawowych, za pośrednictwem kluczy obcych, z tabelą faktów połączone są tabele reprezentujące dane definiujące wymiar. Jeśli wszystkie tabele wymiarów połączone są bezpośrednio z tabelą faktów, mówimy o kostce w modelu gwiazdy. Jeżeli istnieje chociaż jeden wymiar łączący się z tabelą faktów za pośrednictwem innej tabeli reprezentującej wymiar, to mówimy o modelu płatka śniegu. Taki model występuje najczęściej wtedy, gdy wymiar ma ustalone przynajmniej dwa poziomy, a dane do niego są zgromadzone co najmniej w dwóch tabelach. W modelu płatka pośrednicząca tabela wymiaru musi mieć zdefiniowany dodatkowy klucz obcy, za którego pośrednictwem łączy się tabela poziomu nadrzędnego. Czasami spotyka się również model podwójnego (wielokrotnego) płatka śniegu — bardzo często nazywany konstelacją faktów. Ma to miejsce wtedy, kiedy dane stanowiące fakty zgromadzone są w więcej niż jednej tabeli, a wymiary są łączone do takiej struktury za pomocą każdej z tych tabel. Taki model jest silnie lansowany przez Ralpha Kimballa. Jednakże za dużą elastyczność w projektowaniu hurtowni płacimy dużą złożonością formalną, nad którą trudno zapanować. Ma to kluczowy wpływ w przypadku rozwiązań komercyjnych na problemy z utrzymaniem produktu. Naturalna jest, chociażby ze względu na złożoność formalną, tendencja do upraszczania modeli przynajmniej do modelu płatka. Często dążymy do zredukowania go do gwiazdy. Najprostszym rozwiązaniem jest stosowanie jako źródeł danych w miejsce tabel perspektyw łączących atrybuty pochodzące z różnych miejsc. W takim przypadku nawet dla wymiarów o wielu poziomach wystarczające jest odwołanie do pojedynczego widoku zbierającego dane z wielu tabel. Można powiedzieć, że relacje zamiast na poziomie tabel są realizowane jako złączenia wielu źródeł schematu relacyjnego. Podobny zabieg dotyczy tabeli faktów, która jest w istocie dynamiczną kopią danych pochodzących z wielu tabel. Często dane te są wstępnie przetworzone w celu zebrania w jednym miejscu zarówno kluczy obcych definiujących punkty dołączenia wymiarów, jak i obliczeń wyznaczających wartości wyrażeń definiujących bardziej złożone fakty (miary). Odpytywanie takiej perspektywy wiąże się jednak z wykonaniem zapytania wybierającego, często dość złożonego, które stanowi jej definicję. Wymaga to znacznego nakładu zasobów w postaci pa-

mięci RAM i dość długiego czasu. Poprawa wydajności implikuje zatem materializację źródeł danych. Ponieważ w środowisku SQL Server nie dysponujemy obiektem perspektywy zmaterializowanej, to materializacji dokonujemy na skutek przepisania zestawu rekordów definiowanych przez perspektywę do pośredniej tabeli, która z kolei jest źródłem danych dla hurtowni.



RYSUNEK 3.11. *Struktury gwiazdy i płatka śniegu*

W rezultacie przetwarzania, a następnie odpytywania kostki otrzymujemy postać wyników analogiczną do dobrze znanej struktury tabeli przestawnej. Jednak taka postać, w której wyniki reprezentowane są przez tabele zawierające dużą ilość danych numerycznych, jest z reguły mało czytelna. Jak dobrze wiemy, największą ilość informacji jesteśmy w stanie pozyskać wzrokowo. Ponieważ wyniki powinny być łatwe do przeanalizowania dla osób zarządzających firmą, ważnym elementem ich prezentacji jest wizualizacja. Powtarzając za Epiktetem, „raz zobaczyć jakąś rzecz znaczy więcej, niż sto razy o niej usłyszeć", warto każdy wynik przetworzyć na postać graficzną. Dlatego finalnym produktem naszych analiz będzie bez wątpienia jakiś wykres. W tym miejscu trzeba oddać należny szacunek trochę niedocenianemu programowi MS Excel, który oferuje jedne z najbardziej plastycznych narzędzi wizualizacji danych.

Idea hurtowni danych ściśle wiąże się z ich kolosalnymi ilościami, gromadzonymi podczas tysięcy różnych sytuacji — przy dowolnej transakcji, w urzędzie, na lotnisku, w internecie... Nawet nasze połączenia telefoniczne są przechowywane przez operatora. Te wszystkie dane trzeba gdzieś pomieścić, sensownie posegregować i zapewnić sobie możliwość sięgnięcia do wybranego ich zakresu bez długotrwałych poszukiwań. Taką możliwość dają właśnie hurtownie danych — przemyślane, bardzo pojemne bazy, oferujące zarówno integrację wprowadzanych danych, jak i znakomite mechanizmy ich przeszukiwania. Jeśli chcesz poszerzyć swoją wiedzę na temat tworzenia i przeglądania ich zawartości, trafiłeś pod właściwy adres!

Książka *Hurtownie danych. Od przetwarzania analitycznego do raportowania* zawiera materiał przeznaczony nie tylko dla studentów wydziałów informatycznych, ale także dla pasjonatów tej tematyki oraz specjalistów zainteresowanych poszerzeniem wiedzy. W możliwie najprostszy, praktyczny sposób opisuje składnię i postać zapytań analitycznych, strukturę hurtowni danych oraz kwestię ich integracji i wizualnego tworzenia elementów hurtowni. Znajdziesz tu także omówienie analizy danych z wykorzystaniem rozszerzenia MDX SQL oraz zastosowań raportowania. Zapoznanie się z tymi informacjami oraz prześledzenie zgromadzonych tu przykładów pozwoli Ci zrozumieć problemy powstające przy budowie hurtowni danych i wykorzystać tę wiedzę we własnych projektach.

- **Zapytania analityczne**
- **Struktura hurtowni danych**
- **Integracja danych**
- **Wizualne tworzenie elementów hurtowni danych**
- **Analiza danych z wykorzystaniem rozszerzenia MDX SQL**
- **Raportowanie**

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-7411-9

Cena: 99,00 zł