Bazy danych

13. Hurtownie danych: fakty i wymiary

P. F. Góra

http://th-www.if.uj.edu.pl/zfs/gora/

2021

1. Świat jest wszystkim, co jest faktem.
1.1. Świat jest zbiorem faktów, nie rzeczy.

Ludwig Wittgenstein, *Traktat logiczno-filozoficzny*

Dane gromadzone przez różne instytucje w czasie ich działania zawierają wiele informacji, które mogą być przydatne przy *późniejszym* optymalizowaniu ich procesów biznesowych: Jak wygląda sprzedaż w poszczególnych sklepach, miesiącach i dniach tygodnia, a nawet godzinach, ile przeciętnie sprzedaje się danego artykułu, jak zmieniają się wzorce sprzedaży (jeden artykuł traci popularność, inny zyskuje), jak wyniki sprzedaży są skorelowane z akcjami promocyjnymi itp itd.

Pozyskiwanie takich danych analitycznych należy do obszaru *business intelligence*. Wyspecjalizowane, duże bazy danych służące do tego celu nazywa się *hurtowniami danych* (ang. *data warehouses*).

"Kanoniczne" przykłady hurtowni danych dotyczą dużych organizacji handlowych, ale narzędzia te wykorzystywane są także przez duże firmy usługowe, sieci hoteli, sieci szpitali, linie lotnicze, przedsiębiorstwa telekomunikacyjne, przedsiębiorstwa dostarczające "media" (ang. *utilities*) i wiele innych.

Przykład

Zakład energetyczny, który wprowadził zdalny, automatyczny odczyt liczników prądu co 24 godziny, zauważył, że jeden z odbiorców z grupy, która dotąd zachowywała się "podobnie", nagle zmienił wzorzec swojego zachowania i zaczął pobierać znacznie mniej prądu. Aby to stwierdzić, zakład energetyczny musiał gromadzić odczyty z poszczególnych liczników z przypisanymi im numerami klienta (i wszelkimi powiązanymi informacjami, takimi jak dane osobowe czy fizyczna lokalizacja licznika) oraz datami odczytów, a następnie sklasyfikować klientów pod względem charakterystyki zużycia prądu. To ostatnie wymaga zastosowania AI, ale w tym celu trzeba szybko, wiele razy odczytywać **bardzo duże** zbiory danych. Dopiero po dokonaniu klasyfikacji można zauważyć, że któryś klient raptownie wypada z dotychczasowej grupy.

Takie zdarzenie może mieć wiele powodów — jednym z nich jest "obejście licznika", czyli kradzież prądu. I dlatego zakłady energetyczne rutynowo prowadzą tego typu analizy.

Bazy OLTP i OLAP

Bazy OLTP — *Online Transaction Processing*

- muszą zapewnić bezpieczeństwo transakcji przy **wielu** użytkownikach jednocześnie konkurujących o dostęp do tych samych danych,
- muszą zapewnić spójność danych przy systemach rozproszonych i skalowaniu poziomym,
- normalizacja zapobiega powstawaniu anomalii danych.

Bazy OLAP — *Online Analytical Processing*

- służą do analizy danych, wymagającej wielokrotnych odczytów dużych zbiorów danych,
- muszą być zoptymalizowane na odczyt,
- są scentralizowane (tylko skalowanie pionowe).

Te dwie funkcje — obsługa transakcji z zapewnieniem spójności danych vs. szybki odczyt dużych zbiorów danych — w zasadzie się wykluczają.

Hurtownia danych

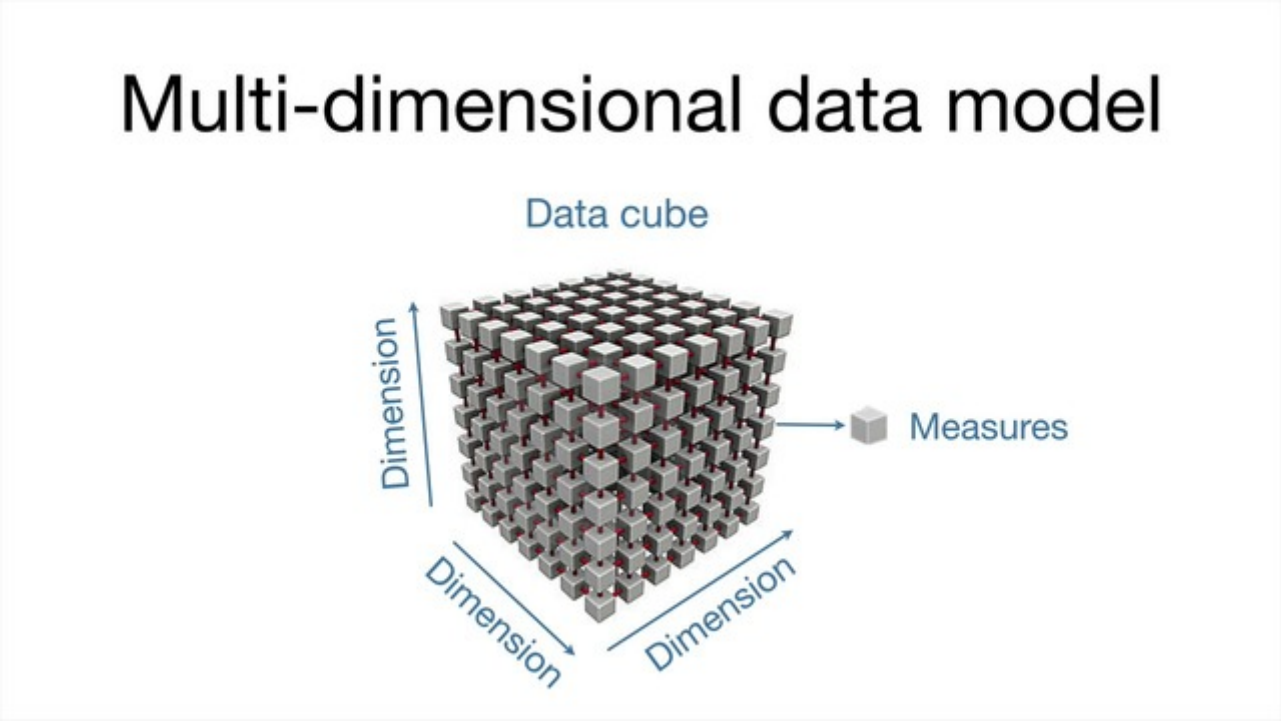
Hurownia danych (ang. *data warehouse*) jest to system, który pozyskuje dane z systemów źródłowych, przekształca je i ładuje do wielowymiarowych struktur, a następnie dostarcza zapytania i analizy wspierające podejmowanie decyzji.

Ralph Kimball

Ta definicja koncentruje się na przeznaczeniu hurtowni, ignorując cechy "techniczne". Istnieje też inna definicja hurtowni danych. Na potrzeby tego wykładu różnice pomiędzy tymi definicjami (i ich konsekwencje!) możemy pominąć.

Kostka danych

W hurtowniach danych dane przechowywane są w wielowymiarowych kostkach.



Komórki takiej kostki zawierają *fakty*, zwane także miarami*, odpowiadające elementranym zdarzeniom (operacjom, transakcjom (w sensie biznesowym), . . .), podlegającym analizie. Współrzędne komórki określają jej położenie względem *wymiarów*. Wymiarów może być 2, 3, 4, aż do kilkunastu (niekiedy — kilkudziesięciu).

W praktyce *wymiary* przechowywane są w osobnych tabelach. Tabela *faktów* oprócz miar przechowuje klucze obce do tabel wymiarów. Model kostki danych jest modelem logicznym, nie fizycznym.

*bardziej precyzyjnie: większości *faktów* przyporządkowane są *miary*.

Przykład

Jan Kowalski (kto?) 11 stycznia 2021 o godz. 8:27 (kiedy?) w sklepie przy ul. Miśnieńskiej 8 w Krakowie (gdzie?)

kupił 3 kg ziemniaków (jaki towar?).

Kto? kiedy? gdzie? jaki towar? określają wymiary; "kg" jest jednostką stowarzyszoną ze współrzędną "ziemniaki" wymiaru "towar" (ze współrzędną "mleko" stowarzyszona byłaby jednostka "l"). 3 jest miarą tego zdarzenia (faktu). Niewyróżnione fragmenty zdania są bez znaczenia ☺

Slice and dice

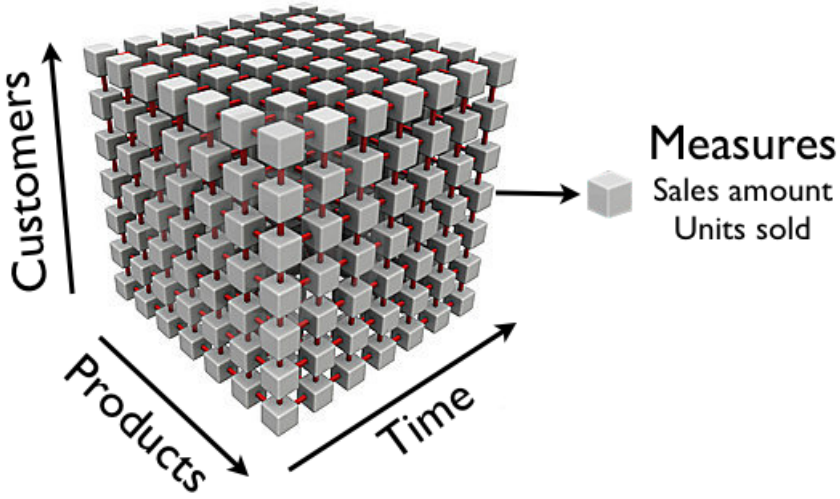
Proces analizy danych zawartych w hurtowni danych nazywany jest niekiedy *slice and dice* (pol. *przecinanie i rzutowanie*). Oznacza on redukowanie rozmiaru danych, który wyjściowo może być bardzo duży i przedstawianie ich w postaci różnych widoków. Bardzo często łączy się to z obliczaniem wielkości statystycznych, takich jak suma, średnia, liczność, wartość największa i najmniejsza. Możemy na przykład pytać o

- średnią sprzedaż produktu X w poszczególnych miesiącach
- największą i najmniejszą liczbę sztuk różnych produktów sprzedanych w poszczególnych sklepach
- średnią liczbę klientów obsłużonych w poszczególnych sklepach w różnych dniach tygodnia
- itd itd.

Nie są to jedyne możliwe typy analiz. Na przykład klasyfikacja, o jakiej mowa była w przykładzie z zakładem energetycznym, wymaga bardziej złożonej analizy danych, niż obliczanie prostych wielkości statystycznych.

Struktury danych w hurtowni danych są uproszczone w celu zapewnienia jak największej wydajności przy sporządzaniu raportów i analiz. Płaci się za to spowolnieniem procesu zasilania hurtowni danymi.

Kostka danych cd



Najbardziej typowym *wymiarem* jest czas (informacja o dacie, godzinie, minucie, sekundzie, dziesiątych częściach sekundy, . . . — w zależności od potrzeb **zamawiającego** hurtownię). Innymi typowymi wymiarami są produkty, klienci (wraz ze wszystkimi dotyczącymi ich danymi), sklepy/placówki świadczące usługi etc. Tabele wymiarów bardzo często są zdenormalizowane (są w pierwszej postaci normalnej).

Przykład

Wymiar `Sklepy` mógłby na przykład zawierać takie dane:

NrSklepu	Ulica	Nr	KodP	Miejsc	Gmina	Powiat	Woj	...
75				Kraków	Kraków	M.Kraków	małopolskie	
76				Kraków	Kraków	M.Kraków	małopolskie	
82				Kraków	Kraków	M.Kraków	małopolskie	
94				Kraków	Kraków	M.Kraków	małopolskie	

Widać, że dane w tabeli powtarzają się. Zgodnie z zasadami normalizacji, informację o tym, że obiekt położony w Krakowie znajduje się jednocześnie w gminie Kraków, w powiecie miejskim krakowskim i w województwie małopolskim, podobnie dla innych miast, należałoby zapisać w osobnej tabeli, która byłaby złączana przy konieczności pobrania odpowiednich danych. Jednak wykonanie złączenia wymaga czasu, co przy wielokrotnych odczytach mogłoby istotnie opóźnić cały proces. Dlatego w hurtowniach danych używa się niskich postaci normalnych. Potencjalnie może to prowadzić do powstania anomalii danych, dlatego należy zapewnić wyjątkową stranność procesu zasilania hurtowni danych danymi źródłowymi.

Inny przykład

W wymiarze `Data` można by przechowywać jedynie datę (w ustalonym formacie, szybkim w obsłudze), ale zazwyczaj przechowuje się ją w wielu formatach, przechowuje się także numer dnia w roku, numer dnia w miesiącu, numer miesiąca, nazwę miesiąca (po polsku i angielsku), nazwę dnia tygodnia (jak wyżej), datę początku tygodnia, datę końca tygodnia, numer kwartału, a w systemie amerykańskich także numery w odniesieniu do początku roku fiskalnego. Wiersz uzupełnia się unikalnym kluczem. Wszystkie wielkości oprócz klucza można obliczyć, a jednak przechowuje się wartości obliczone (co, formalnie, także może prowadzić do anomalii danych), gdyż wielokrotne obliczanie tych samych wielkości może być wolniejsze niż odczytanie ich z bazy, można łatwo założyć indeksy na przykład na numerze miesiąca czy kwartału, co przyspieszy wyszukiwanie odpowiednich danych, a poza tym programowanie każdorazowego obliczania może być źródłem trudnych do zlokalizowania błędów w zapytaniach.

Fakty

Faktem jest każde zdarzenie podlegające analizie. Fakty mają swoje miary (np. wartość sprzedaży, liczba sprzedanych sztuk towaru, stan licznika energii elektrycznej). Miary zazwyczaj wyrażają się w liczbach całkowitych lub dziesiętnych (ang. *decimal* — patrz odpowiedni typ danych w SQL), ale miarami niekiedy mogą być wartości prawda/fałsz, elementy typu wyliczeniowego, a nawet łańcuchy znaków (opisy, teksty). W tabeli faktów, oprócz miar, przechowywane są **klucze obce** do wymiarów. Tabele faktów są zdenormalizowane i, typowo, **bardzo duże**: Pomyślmy o tabeli faktów w hurtowni danych zawierającej informacje o sprzedaży w sieci hipermarketów w dużym okresie.

Zależność pomiędzy *faktami* a *wymiarami* niezbyt precyzyjnie, ale obrazowo można przedstawić tak:

$$\texttt{SELECT}\ \underbrace{\texttt{foobar}}_{\text{fakty}}$$
$$\texttt{FROM}\ \ldots$$
$$\texttt{WHERE}\ \underbrace{\texttt{popo}}_{\text{wymiary}} = \texttt{lupo};$$

Ziarnistość faktów

Na jak szczegółowym poziomie powinniśmy analizować fakty? To zależy od potrzeb biznesowych zamawiającego hurtownię danych! W przypadkach "handlowych" fakt może być stowarzyszony z każdym *beep* wydanym przez czytnik kodów paskowych. Oprócz informacji o kwocie i liczbie sztuk (kilogramów, litrów,...) sprzedanego towaru, z takim faktem stowarzyszymy klucze obce do wymiarów data-i-czas, produkt, sklep, sprzedawca, a być może jeszcze jakichś innych.

Z drugiej strony dla zaoszczędzenia miejsca, naszym faktem możemy uczynić cały paragon, bez specyfikowania, jakie produkty i w jakiej ilości on obejmuje. To niekiedy może być wystarczające, choć z góry wyklucza możliwość przeprowadzanie pewnego typu analiz szczegółowych. Dlatego *zaleca się*, aby fakty

były tak szczegółowe, jak się tylko da, ale ostatecznie zależy to od potrzeb biznesowych klienta, dla którego projektowana jest hurtownia danych. Jeżeli klient pierwotnie zdecyduje się na “grube” ziarno, a później dojdzie do wniosku, że przydałyby mu się bardziej szczegółowe analizy, właściwie trzeba projektować całą hurtownię danych od nowa.

Mieszanie w jednej hurtowni danych faktów o różnej ziarnistości zazwyczaj jest błędem projektowym.

Niekiedy wyróżnia się następujące rodzaje tabel faktów z uwagi na ich ziarnistość:

- Transakcyjne — przechowują informacje o poszczególnych transakcjach (w sensie biznesowym).
- Migawkowe (*snapshot, periodic snapshot*) — każdy wiersz przechowuje dane odpowiadające poszczególnym, z góry określonym okresom czasu: dniom, mieiącom, latom itp.
- Migawki przyrostowe (*accumulating snapshot*) — używane rzadziej, zawierają informacje o procesach, które mają dobrze określony przebieg, wystarczy więc znać bieżący stan procesu. Kłopoty pojawiają się, gdy z jakichś powodów, na ogół niezależnych od projektanta hurtowni danych i przez niego nie przewidzianych, proces zaczyna przebiagać inaczej, niż się tego spodziewamy ☹.

Fakty bez faktów

Często należy zapamiętywać zdarzenia, którym nie sposób przypisać żadnej miary. Są to tak zwane *fakty bez faktów* (ang. *factless facts*), zwane niekiedy faktami bez miar. Przykład 1: Dany pacjent zgłosił się danego dnia, o danej godzinie, do danej placówki medycznej. Nie można temu zdarzeniu przypisać żadnej wartości liczbowej, ale samo to zdarzenie jest faktem, który może podlegać przyszłej analizie. Wiersz tabeli faktów łączy jedynie klucze obce do wymiaru data-i-czas, wymiaru pacjent i wymiaru placówka. Z punktu widzenia teorii baz danych tabela taka jest typową tabelą pomostową, opisującą relację trójargumentową. Wydawać by się mogło, że datę-i-godzinę można uznać za "wartość liczbową" zdarzenia, praktyka pokazuje jednak, że należy ją uznać za osobny wymiar.

Przykład 2: Podobna sytuacja występuje, gdy na autostradzie wyposażonej w kamery rozpoznające numery rejestracyjne pojazdów, samochód o danym numerze został zarejestrowany przez daną kamerę w danym dniu i o danym czasie. Jeżeli kamera sprzężona jest z systemem automatycznego pomiaru prędkości, prędkość tę można (i zapewne należy) zapamietać jako pewną *dodatkową* miarę tego faktu. Odpowiada to sytuacji, gdy relacja trójargumentowa ma swój własny atrybut.

Brakujące fakty

Inna sytuacja występuje, gdy pewnych miar faktów po prostu brakuje — w chwili tworzenia wiersza tabeli faktów wartość któregoś atrybutu nie jest znana. Można tam wstawić wartość `NULL`, a można też wstawić jakąś wartość domyślną, na przykład 0. Zgodnie z zasadami Codda, powinno się wstawiać `NULL`, ale ta specjalna wartość zakłóca szereg operacji arytmetycznych. Zabezpieczamy się przed tym wykonując instrukcję warunkową `IF NOT IsNull(atrybut)`, ale to oczywiście spowalnia operacje. Dlatego w hurtowniach częściej stosuje się jakąś liczbową wartość domyślną oraz dodatkową flagę boole'owską określającą, czy jest to "prawdziwe" zero (true, 1), czy brakujące dane (false, 0); operacje na bitach są szybkie.

Różne rodzaje wymiarów: Wymiary uzgodnione

Wymiar uzgodniony (ang. *conformed dimension*) to taki wymiar, który pozostając w relacji z wieloma rodzajami faktów, ma takie samo znaczenie. Klasycznym przykładem jest data.

Hurtownia danych jest prawie zawsze zasilana z różnych źródeł danych. Aby dwa wymiary można było uznać za uzgodnione, muszą być identyczne lub jeden musi być podzbiorem drugiego. Jeżeli źródła danych mają różną strukturę, należy *bardzo uważać*, czy wymiary są rzeczywiście uzgodnione, czy tylko tak się wydaje.

Wymiary wielokrotnego stosowania

Wymiary wielokrotnego stosowania (ang. *role-playing dimension*) to wymiary, które są przechowywane w *jednej* tabeli wymiarów, ale odnoszą się do innych aspektów danego faktu — wiersz tabeli faktów będzie wówczas zawierał klucze obce do kilku, zapewne różnych, wierszy tabeli wymiaru. Na przykład w systemie rezerwacji hotelowych możemy mieć datę dokonanej rezerwacji, planowaną datę przyjazdu, faktyczną datę przyjazdu, planowaną datę wyjazdu, faktyczną datę wyjazdu — wszystkie odnosić się będą do (potencjalnie) różnych dat, czyli wierszy tabeli wymiaru `Data`.

Wymiary abstrakcyjne

Wymiary abstrakcyjne (ang. *junk dimension* (sic!)) odnoszą się do sytuacji, w której mamy do czynienia z różnymi wymiarami o bardzo małej liczności. Powiedzmy, mamy wymiar płeć (kobieta, mężyczna) i wymiar opisujący grupy wiekowe pacjentów (poniżej 18, 18-25, 25-35, 35-45, 45-55, 55-65, powyżej 65). Zamiast tworzyć złączenia z wieloma małymi tabelami, tworzymy wymiar "abstrakcyjny" jako iloczyn kartezjański dwu małych wymiarów.

klucz	płeć	grupa
1	K	-18
2	K	18-25
.....		
14	M	$65+$

Jeśli odwołamy się do współrzędnej 12 tego wymiaru, oznacza to, że mówimy o mężczyźnie z grupy wiekowej 45-55.

Gdybyśmy jeszcze mieli wymiar “status” (VIP, nie-VIP), moglibyśmy go dołączyć do tego wymiaru, dodając czwartą kolumnę do powyższej tabeli. Cały wymiar miałby wtedy liczność 28.

Wymiary zdegenerowane

Wymiary zdegenerowane (ang. *degenerate dimension*) to wymiary, których liczność jest porównywalna z licznością tabeli faktów. Gdy wszystkie warte przechowania atrybuty zostały zapisane w tabeli faktów, a z wymiaru pozostał nam tylko klucz biznesowy, ten klucz *także* przechowujemy w tabeli faktów. Przykładem może być numer paragonu w sytuacji, gdy cała zawartość paragonu znalazła się w wierszach tabeli faktów w postaci miar lub kluczy obcych do innych wymiarów. Innym przykładem może być numer historii choroby pacjenta.

Dygresja 1: Klucz biznesowy

W żargonie typowym dla hurtowni danych często pojawia się sformułowanie “klucz biznesowy”. Jest to zestaw atrybutów, który jednoznacznie identyfikuje dany obiekt w świecie rzeczywistym. W teorii baz danych odpowiada to *kluczowi kandydującemu* odpowiedniej tabeli. Jeśli klucz biznesowy jest pojedynczym atrybutem (np PESEL dla osoby, NIP dla przedsiębiorstwa), często czyni się go kluczem odpowiedniej tabeli, ale mozna tez wygenerować ”sztuczny” klucz, lepiej odpowiadający przyjetej konwencji nazewniczej lub szybciej obsługiwany. Jeśli klucz obejmuje więcej, niż jeden atrybut (np numer oddziału firmy oraz numer pracownika), w hurtowniach danych, zamiast stosować klucze na dwu (lub więcej) kolumnach, na ogół generuje się w takiej sytuacji jednoatrybutowy klucz “sztuczny”.

Inna klasyfikacja wymiarów — wymiary stałe

Wymiar jest stały, gdy zawartość — liczba wierszy i wartości atrybutów — tabeli tego wymiaru nie zmienia się w czasie całej eksploatacji danej hurtowni danych. Tabela tego wymiaru jest zasilana danymi raz, przy tworzeniu hurtowni.

Na przykład wymiar Data-i-Czas wypełniany jest raz. Zaczyna się od daty daleko w przeszłości (np 1 stycznia 1950), kończy na dacie daleko w przyszłości (np 31 grudnia 2049), z ziarnistością (dni? godziny? setne części sekundy?) wynikającą z potrzeb biznesowych klienta zamawiającego hurtownię. Zasilając tabelę takiego wymiaru, oblicza się wszystkie wielkości wymienione na stronie 15. Jest to rozwiązanie bardziej efektywne od uzupełniania tabeli tego wymiaru gdy dana wartość faktycznie wystąpi (konieczne byłoby nie tylko wyszukiwanie, ale i przebudowa indeksów!).

Wymiary wolnozmienne, typ 0 — zachowaj

W wymiarach wolnozmiennych typu 0 dopuszczalne jest dodawanie unikalnych nowych wierszy do tabeli wymiaru, jeśli dane źródłowe zawierają odniesienie do takiej nowej wartości. Na przykład w sieci przychodni pojawił się nowy pacjent albo sieć sklepów otwarła nową placówkę. Niedopuszczalne jest natomiast modyfikowanie istniejących wierszy — wartości raz zapisane w tabeli takiego wymiaru uważane są za poprawne i nie podlegają dalszej weryfikacji. Oznacza to jednak także, że nie można prostować nawet oczywistych błędów. Jeżeli w danych źródłowych pojawi się “Paweł Gora”, informacja o nim zostanie zapisana w tabeli wymiaru `Pacjent`. Jeśli później pojawi się “Paweł Góra”, spowoduje to wygenerowanie nowego wiersza w tabeli wymiaru (nawet jeśli zgodność pozostałych atrybutów, takich jak PESEL, sugeruje wystąpienie pomyłki przy wprowadzaniu danych w którymś miejscu).

Taki wymiar nie pozwala na śledzenie zmian wartości jakiegoś atrybutu.

Typ 1 — nadpisz

Jeśli w danych źródłowych pojawi się nowa wartość jakiegoś atrybutu z tabeli wymiaru o już istniejącym *kluczu biznesowym*, stara wartość zastępowana jest nową. W tabeli wymiaru przechowywane są tylko najnowsze wartości atrybutów — na przykład jeśli pacjent, który mieszkał w województwie śląskim przeprowadził się do świętokrzyskiego, informacja o poprzednim województwie zamieszkania zostanie utracona; niekiedy, z powodów biznesowych, może to nie stanowić problemu (na przykład nie jest ważne, gdzie świadczono usługi, a jedynie komu). Można w ten sposób poprawiać błędy, jakie powstały przy wprowadzaniu danych, ale można też wprowadzać informacje niepoprawne ☹.

Typ 2 — dodaj wiersz

Jeśli w danych źródłowych pojawi się nowa wartość jakiegoś atrybutu z tabeli wymiaru o już istniejącym *kluczu biznesowym*, do tabeli wymiaru dodawany jest nowy wiersz, któremu nadaje się nowy, unikalny, jednokolumnowy klucz główny. Stary wiersz zostaje zachowany.

Aby móc odróżnić "stare" wiersze od nowego, do tabeli wymiaru dodaje się dwie lub trzy kolumny: znacznik czasu początku ważności i znacznik czasu końca ważności wiersza. Przy dodawaniu nowego wiersza, poprzedni wiersz aktualny ma znacznik końca ważności ustawiany na czas, w którym faktycznie zaszła zmiana (lub w którym zarejestrowano ją w tabeli wymiaru). Nowy wiersz aktualny ma znacznik początku ważności ustawiony tuż po (najmniejsze możliwe ziarno) znaczniku końca ważności poprzedniego wiersza aktualnego. Wiersz aktualny

ma znacznik końca ważności albo ustawiony na `NULL`, albo — częściej, z powodów omówionych przy okazji brakujących miar faktów — na nieskończoność, czyli na największą wartość dopuszczalną w systemie. Można teraz obliczyć, który wiersz jest aktualny. Aby uniknąć obliczania, można wprowadzić trzecią dodatkową kolumnę, boole'owską flagę ważności: wiersz aktualny ma ją ustawioną na true, pozostałe na false.

W ten sposób można śledzić historię zmian poszczególnych atrybutów. Samo to może być przedmiotem analizy, a przy tym usługi, jakie pacjentowi z poprzedniego przykładu wykonano w województwie śląskim, zostaną poprawnie przypisane do województwa śląskiego, nie zaś do świętokrzyskiego, jak byłoby w typie 1.

Typ 3 — dodaj atrybut

W tabeli wymiaru możemy przewidzieć kolumnę (lub kolumny) na poprzednią (poprzednie) wartości jakiegoś atrybutu, oraz kolumny określające, od kiedy są one ważne. Jeśli pacjent przeprowadza się jak w powyższych przykładach, atrybut `woj` zmieni wartość na "świętokrzyskie", atrybut `stare_woj`, dotąd pusty, przybierze wartość "śląskie", a atrybut `od_kiedy_woj` przybierze wartość wskazującą na — na przykład — 1 stycznia 2018.

Pozwala to na ograniczone śledzenie zmian atrybutu. Projektant musi z góry przewidzieć, zmiany których atrybutów będą śledzone i jak głęboko.

Typ 4 — tabele historyczne

Wiersze, które w typie 2 traktowano by jak “stare”, przenoszone są do tabeli historycznej. Ma ona taką samą strukturę, jak podstawowa tabela wymiaru, plus jedną dodatkową kolumnę, określającą do kiedy dany wiersz był ważny. Podstawowa tabela wymiaru zawiera tylko informacje aktualne — wiersz o takiej zawartości zostaje wprowadzony ilekroć w danych źródłowych zmieni się wartość któregoś atrybutu). Przyspiesza to wykonywanie wielu złączeń (wartości aktualne bywają częściej potrzebne, niż historyczne), ale spowalnia wyszukiwanie danych historycznych.

Stosuje się też inne, bardziej złożone mechanizmy obsługi wymiarów wolnozmiennych, takie jak wymiary hybrydowe, ale wykracza to poza ramy wykładu.

Wymiary szybkozmienne

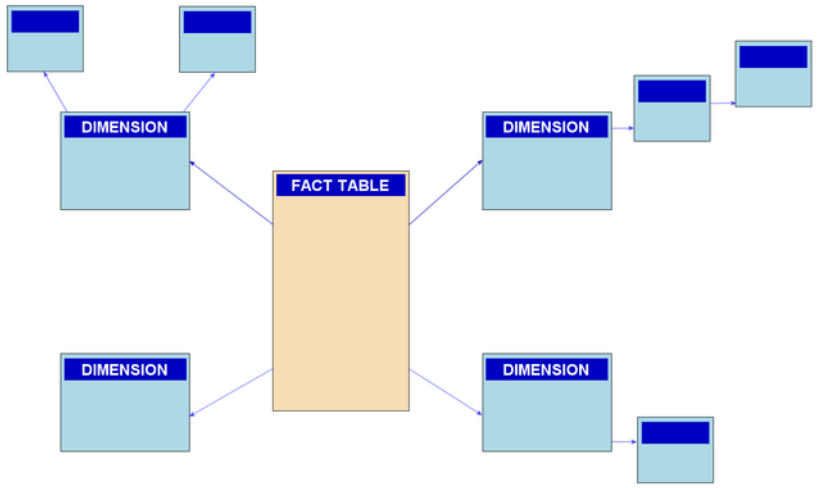
Jeśli jakiś wymiar — pewien atrybut lub pewna niewielka grupa atrybutów tego wymiaru — zmienia się szybko, ale w ograniczonym zakresie, przy czym co to znaczy “szybko” zależy od konkretnej rzeczywistości biznesowej, najczęściej postępuje się następująco:

- Tworzymy tabelę odpowiadającą wszystkim dopuszczalnym wartościom danego atrybutu; nazywa się ją *miniwymiarem*. Jeśli zmieniać się może kilka atrybutów, wszystkie te miniwymiary łączymy w jeden wymiar abstrakcyjny.
- Tworzymy dodatkową tabelę faktów typu “fakty bez faktów”, łączącą wymiar (tę grupę atrybutów wymiaru, która nie zmienia się szybko) i opisany powyżej miniwymiar/wymiar abstrakcyjny, a być może także inne wymiary, na przykład wskazujące na moment zmiany.

Fizyczna organizacja danych — gwiazda i płatek śniegu

Opisana powyżej struktura danych — zdenormalizowana tabela faktów, lub kilka zdenormalizowanych tabel faktów, zawierające, obok miar, klucze obce do zdenormalizowanych tabel wymiarów — nazywana jest strukturą gwiazdy.

Bardziej ogólna jest struktura płatka śniegu, w której tabele faktów są nadal zdenormalizowane, ale tabele wymiarów są znormalizowane. Pozwala to lepiej uwzględnić hierarchiczną informację zawartą w niektórych wymiarach, chroni przed anomaliami danych, pozwala zaoszczędzić miejsce do przechowywania danych, za cenę większego nakładu pracy przy złączeniach. Jednak wiele złączeń nie sięga aż do najniższego poziomu hierarchii. Schemat gwiazdy uznawany jest za przypadek szczególny schematu płatka śniegu.



By SqlPac - Own work, CC BY-SA 3.0

Dygresja 2: Kolumnowe bazy danych

W relacyjnych systemach baz danych tabela zdefiniowana jest jako zbiór wierszy. Wynika z tego, że tabele przechowywane są wierszami (ale kolejność wierszy nie jest ustalona).

Kolumnowa baza danych przechowuje informację w tabelach w kolejności kolumn, nie wierszy.

Przypuśćmy, że w pewne bazie istnieje tabela `Pracownicy`:

rowid	NrPrac	Imię	Nazwisko	Pensja
001	10	Alicja	Zielińska	5200
002	11	Bogdan	Wrona	4900
003	12	Czesław	Kowalski	5600
004	114	Dominik	Wrona	5200
.....				

(1)

W systemie kolumnowym tabela (1) mogłaby być przechowywana w następującej postaci:

10:001;11:002;12:003;114:004;...
Alicja:001;Bogdan:002;Czesław:003;Dominik:004;...;
Zielińska:001;Wrona:002,004;Kowalski:003;...
5200:001,004;4900:002;5600:003;...

Jest to struktura nieco podobna do indeksu: Unikalna wartość atrybutu wskazuje na identyfikator (identyfikatory) wierszy, w których ta wartość występuje. Dlatego kolumnowe bazy danych nazywa się niekiedy “samoindeksującymi”. Uwaga: ten mechanizm nie zadziałałby w przypadku kluczy obejmujących więcej niż jedną kolumnę. Jesto to jeden z powodów, dla których w hurtowniach danych unika się takich kluczy (patrz “klucze biznesowe” wyżej, strona 29).

Zalety baz kolumnowych

Przypuśćmy, że mamy tabelę o znacznej liczbie kolumn. Jeśli jakieś zapytanie potrzebuje informacji tylko z niewielkiej liczby kolumn, RDMS i tak musi wczytywać całe wiersze. Większość wczytanej informacji okazuje się niepotrzebna, a konieczność jej wczytania spowalnia działanie systemu. Może to być mocno zauważalne, jeśli wczytywana tabela jest duża.

W kolumnowej bazie danych wczytywane są tylko te kolumny, których naprawdę potrzebujemy. Z drugiej strony w kolumnowych bazach danych operacje wierszowe (dodawanie, modyfikowanie, usuwanie wiersza) są znacznie wolniejsze.

Ponieważ tabele opisujące wymiary w hurotowniach danych często są jednocześnie duże (wiele wierszy) i “szerokie” (wiele atrybutów), a jednocześnie

można uznać, że ich zawartość nie zmienia się podczas normalnego (analitycznego) trybu pracy hurtowni, często — jeśli zastosowany silnik dopuszcza taką możliwość — są one przechowywane w postaci kolumnowej. (Tabele faktów przechowywane są wierszami.)

Wykorzystanie widoków zmaterializowanych

Tabele faktów w hurtowniach danych mogą być **bardzo duże**, tabele wymiarów są na ogół wyraźnie mniejsze od tabel faktów, ale niekiedy też mogą być duże. Hurtownie danych zoptymalizowane są na odczyt, a złączanie dużych tabel może być kosztowne. Należy używać wszelkich dostępnych narzędzi, aby przyspieszyć odczyt. W szczególności należy unikać wielokrotnego wykonywania operacji, które można by z powodzeniem wykonać tylko raz.

Przypuśćmy, że musimy (spodziewamy się, wiemy, że mamy) wykonać szereg podobnych zapytań:

```
SELECT ... FROM ...
WHERE Q AND P_1;

SELECT ... FROM ...
WHERE Q AND P_2;
.......................
SELECT ... FROM ...
WHERE Q AND P_n;
```

Warunek $\mathcal{Q}$ jest taki sam dla wszystkich tych zapytań. Można o nim myśleć jako o warunku złaczenia (theta-join), jako o filtrze nałożonym na dane, bądź jako o jednym i drugim naraz. Zastosowanie tego warunku redukuje potencjalnie bardzo duży zbiór danych wejściowych do znacznie mniejszego podzbioru.

Operacja ta wkonywana jest we *wszystkich* powyższych zapytaniach — zupełnie niepotrzebnie, bo jej wynik się nie zmienia.

Możemy skorzystać z widoku zmaterializowanego:

```
CREATE MATERIALIZED VIEW MatFoo AS
SELECT ... FROM ... WHERE Q;
```

Następnie wykonujemy

```
SELECT ... FROM MatFoo WHERE P_1;
SELECT ... FROM MatFoo WHERE P_2;
.....................................
SELECT ... FROM MatFoo WHERE P_n;
```

Kluczowe jest wykorzystanie widoku zmaterializowanego — dla zwykłego widoku zapytanie definiujące widok byłoby niejawnie wykonywane za każdym odwołaniem do `MatFoo`, wyniki widoku zmaterializowanego są fizycznie przechowywane w systemie. Jeśli nie mozna użyć widoku zmaterializowanego, można posłużyć się tabelą tymczasową.

Wydaje się, że rodzi to niebezpieczeństwo braku spójności danych: Widok zmaterializowany jest automatycznie odświeżany co jakiś czas (tabela tymczasowa nie jest odświeżana), więc jeśli w międzyczasie dane w tabelach źródłowych uległy zmianom, widok zmaterializowany o zmianach tych dowie się dopiero po pewnym czasie, a wyniki zapytań odnoszących się do takiego widoku mogą opierać się na nieaktualnych danych. Na szczęście, w hurtowniach danych sytuacja taka nie zachodzi.

Tryby pracy hurtowni danych

Hurtownia danych działa w dwu możliwych trybach:

1. Tryb operacyjny, "analityczny". Użytkownicy — możliwe, że wielu użytkowników na raz — mogą czytać dane zawarte w hurtowni, wykonywać predefiniowane zapytania analityczne i przygotowywać predefiniowane raporty. Niekiedy także użytkownicy (lub wybrani użytkownicy) mogą tworzyć własne zapytania i raporty. Użytkownicy pracujący w tym trybie nie mogą w żaden sposób modyfikować (usuwać, modyfikować lub wprowadzać dane) zawartości hurtowni danych ani jej schematu. Dlatego przyjmuje się, że zawartość hurtowni danych nie ulega zmianom, gdy pracuje ona w tym trybie, a wobec tego nie ma konieczności zapewniania obsługi transakcji.
2. Tryb zasilania hurtowni danymi. Może to robić jeden, wybrany użytkownik. Proces zasilania hurtowni danymi na ogół wykonywany jest automatycznie

co określony czas (dzień, tydzień, miesiąc) w zależności od potrzeb biznesowych właściciela hurtowni. Gdy hurtownia jest zasilana danymi, aktualizowane są indeksy itp, wyłącza się dostęp użytkownikom analitycznym. W tym czasie mogą być też wykonywane inne prace konserwatorskie na hurtowni danych. Wszelkie modyfikacje schematu hurtowni są w najwyższym stopniu niewskazane, jako że mogą one zrujnować predefiniowane zapytania i raporty.

Jeśli hurtownia danych zasilana jest danymi raz na dobę, typowy (co nie oznacza, że jest to żelazna zasada) cykl obejmuje 23 godziny pracy analitycznej plus godzinę na zasilanie i konserwację hurtowni.

ETL

Proces zasilania hurtowni danymi nosi nazwę *ETL*: *Extract, Transform, Load*.

Hurtownia zazwyczaj zasilana jest z wielu źródłowych baz danych (przykład: zlokalizowane w różnych miastach sklepy należące do tej samej sieci handlowej). Bazy te mogą być heterogeniczne (przykład: sieć handlowa przejęła mniejszą sieć, której sklepy miały własne bazy danych, być może nie w pełni kompatybilne z "natywną" bazą większej sieci). Ekstrakcja danych polega na odczycie danych z wszystkich potrzebnych systemów źródłowych, niekiedy także z zasobów archiwalnych. Należy zwracać uwagę na okres przechowywania (ang. *retention period*) danych źródłowych w różnych źródłach.

Jest szkodliwym złudzeniem, że system hurtownia, plus proces ETL, plus źródłowe bazy danych można zawsze zaprojektować odgórnie i w sposób spójny, zapewniając pełną kompatybilność danych. Tak się prawie nigdy nie dzieje.

Transformacja danych to przekształcenie ich do postaci pożądanej przez hurtownię (formaty daty, kolejność atrybutów, przeliczanie jednostek). Na tym etapie odbywa się też czyszczenie danych, filtrowanie i weryfikowanie danych, a także (częściowa) agregacja danych i uzupełnianie brakujących danych, jeśli zachodzi taka potrzeba.

Ładowanie — pobrane i przekształcone dane zapisywane są w hurtowni. Obliczane są wielkości zależne od innych danych, jeśli hurtownia ich potrzebuje, przebudowywane są indeksy do wymiarów, jeśli zachodzi taka potrzeba.

Szczegóły procesu ETL zależą od struktury hurtowni danych, w tym wymiarów, ziarnistości faktów, a także od struktury źródłowych baz danych

Ważne uwagi na koniec

Projektując hurtownię danych należy zwracać uwagę **na potrzeby biznesowe jej przyszłego właściciela**. Mowa tu na przykład o przechowywanych miarach faktów, o przechowywanych wymiarach, o tym, czy uwzględniać historyczne zmiany wymiarów, o ziarnistości danych itp. Wszystko to musi być dostosowane do raportów, jakie będzie chciał uzyskiwać właściciel hurtowni. Wymiary muszą być dostosowane do ziarnistości danych, nie na odwrót.

Projektowanie hurtowni danych jest procesem złożonym i wieloetapowym, jeszcze bardziej, niż ma to miejsce w bazach typu OLTP. Jest karygodnym błędem, gdy projektant narzuca użytkownikowi swoje rozwiązania, uniemożliwiając użytkownikowi wyciągnięcie informacji, która jest ukryta w danych źródłowych i która

mogłaby być przydatna w działalności operacyjnej właściciela/użytkownika hurtowni.

Kompletny produkt, jakim jest hurtownia danych, obejmuje, oprócz samej hurtowni, także narzędzia służące do ETL, predefiniowane zapytania analityczne i raporty, niekiedy także możliwość tworzenia własnych zapytań i raportów. Użytkownicy hurtowni mogą to chcieć robić graficznie, bo nie znają i nie muszą znać SQL.

7. O czym nie można mówić, o tym trzeba milczeć.

Ludwig Wittgenstein, *Traktat logiczno-filozoficzny*