

Projektowanie i eksploatacja hurtowni danych i data mart

Piotr Muryjas, Grzegorz Kozieł



Projektowanie i eksploatacja hurtowni danych i data mart

Rada Naukowa Wydawnictwa Politechniki Lubelskiej

Przewodnicząca:

Agnieszka Rzepka

Dyrektor CINT:

Katarzyna Weinper

Przedstawiciele Dyscyplin Naukowych Politechniki Lubelskiej:

Marzenna Dudzińska

Małgorzata Franus

Arkadiusz Gola

Beata Kowalska

Grzegorz Komarzyniec

Jarosław Latański

Tomasz Lipecki

Zbigniew Łagodowski

Lucjan Pawłowski

Natalia Przesmycka

Halina Rarot

Magdalena Rzemieniak

Arkadiusz Syta

Wydawnictwo Politechniki Lubelskiej:

Magdalena Chołojczyk

Karolina Famulska-Ciesielska

Katarzyna Pełka-Smętek

PODRĘCZNIKI POLITECHNIKI LUBELSKIEJ

Projektowanie i eksploatacja hurtowni danych i data mart

Piotr Muryjas, Grzegorz Kozieł



WYDAWNICTWO
POLITECHNIKI
LUBELSKIEJ

Lublin 2026

Recenzenci:

dr hab. **Andrzej Krajka**, prof. UMCS, Uniwersytet Marii Curie-Skłodowskiej w Lublinie

dr hab. **Marek Krótkiewicz**, prof. PWr, Politechnika Wrocławska

Autor:

dr inż. **Piotr Muryjas**, ORCID: 0000-0002-1615-8385

Politechnika Lubelska – ROR: <https://ror.org/024zjzd49>

dr inż. **Grzegorz Kozieł**, ORCID: 0000-0002-9948-7087

Politechnika Lubelska – ROR: <https://ror.org/024zjzd49>

Redaktor prowadzący: **Magdalena Chołojczyk**

Korekta językowa: **Agnieszka Rebelak**

Skład i łamanie: **Katarzyna Pełka-Smętek**

Projekt graficzny serii: **Łukasz Maj**

Publikacja została sfinansowana w ramach programu projakościowego prorektora ds. studenckich:

Konkurs na wydanie podręcznika akademickiego lub skryptu

Ilustracja na okładce została przygotowana za pomocą AI

Publikacja wydana za zgodą **Rrektora Politechniki Lubelskiej**

ISBN: 978-83-7947-675-6 (wersja drukowana)

ISBN: 978-83-7947-676-3 (wersja elektroniczna)

DOI: 10.35784/9788379476763

Wydawca: Wydawnictwo Politechniki Lubelskiej
www.wpl.pollub.pl
ul. Nadbystrzycka 36C, 20-618 Lublin
tel. (81) 538-46-59



WYDAWNICTWO
POLITECHNIKI
LUBELSKIEJ

Książka udostępniona jest na licencji Creative Commons Uznanie autorstwa – na tych samych warunkach 4.0 Międzynarodowe (CC BY-SA 4.0)

Nakład: 50 egz.

Spis treści

Wprowadzenie	9
1. Analityka i jej znaczenie w organizacji	13
2. Analityczne źródła danych w organizacji	17
2.1. Systemy klasy ERP	17
2.2. Hurtownia danych i data mart	19
2.3. Zadania do samodzielnego wykonania	22
3. Architektura platform analitycznych	23
3.1. Rodzaje architektury platform analitycznych	23
3.2. Zadania do samodzielnego wykonania	31
4. Modelowanie wielowymiarowe w projektowaniu hurtowni danych	33
4.1. Istota modelowania wielowymiarowego	34
4.2. Identyfikacja obszaru analizy	37
4.3. Definicja faktów	39
4.4. Definicja szczegółowości faktów	42
4.5. Identyfikacja wymiarów i ich atrybutów	48
4.6. Definicja miar	55
4.7. Modelowanie hurtowni danych	61
4.8. Zadania do samodzielnego rozwiązania	64
5. Modele danych hurtowni danych	65
5.1. Model gwiazdy	66
5.2. Model płatka śniegu	69
5.3. Model konstelacji	76
5.4. Zadania do samodzielnego rozwiązania	79
6. Modelowanie faktów	81
6.1. Podstawowe rodzaje tabel faktów	81
6.2. Dodatkowe rodzaje tabel faktów	99
6.3. Zadania do samodzielnego rozwiązania	104
7. Modelowanie wymiarów	105
7.1. Rodzaje tabel wymiarów	105
7.2. Wymiary wolnozmiennie	122
7.3. Hierarchie wymiaru	140
7.4. Zadania do samodzielnego rozwiązania	146

8. Eksploatacja danych w języku SQL	147
8.1. Tworzenie kostek analitycznych – ROLLUP i CUBE	150
8.2. Partycjonowanie danych – OVER i PARTITION BY	159
8.3. Generowanie rankingów – RANK i DENSE_RANK	173
8.4. Zaawansowane funkcje okna – LAG i LEAD	178
8.5. Tworzenie tabel przestawnych – PIVOT	185
8.6. Zadania do samodzielnego rozwiązania	192
Zakończenie	195
Bibliografia	197

Projektowanie i eksploatacja hurtowni danych i data mart

Streszczenie

W niniejszej publikacji przedstawiono metodykę projektowania hurtowni danych i data mart oraz podstawowe techniki eksploatacji zgromadzonych w nich danych za pomocą języka SQL. Na wstępie wskazano znaczenie analizy danych dla współczesnej organizacji biznesowej do wspomagania procesów decyzyjnych. Następnie scharakteryzowano różne analityczne źródła danych oraz zaprezentowano architektury współczesnych platform analitycznych. W kolejnych rozdziałach szczegółowo omówiono etapy działań, których realizacja prowadzi do zbudowania hurtowni danych lub data mart jako głównego elementu warstwy składowania danych aplikacji analitycznej. W tej części Czytelnik może zapoznać się z podstawowymi pojęciami używanymi w projektowaniu tego typu rozwiązań, modelami danych wykorzystywanymi w hurtowniach danych i data marts, sposobami ich budowy oraz technikami eksploataowania danych w tych zasobach, wykorzystującymi instrukcję *SELECT* języka SQL, które zapewniają wsparcie procesów podejmowania decyzji.

Słowa kluczowe: analiza danych, hurtownia danych, data mart, modelowanie wielowymiarowe, eksploatacja danych

Designing and exploring a data warehouse and data mart

Abstract

This publication presents a methodology for the design of data warehouse and data mart, and basic techniques for exploring the data collected in them using the SQL. The introduction indicates the importance of data analysis for contemporary business organisations in supporting decision making. Then, various analytical data sources have been characterised and the architectures of modern analytical platforms have been presented. The following chapters provide a detailed presentation of the stages of activities leading to the building of a data warehouse or data mart as the main element of the data layer of an analytical application. In this part the reader will become familiar with the basic terms used in designing such solutions, the data models used in data warehouses and data marts, the methods of their building, and with techniques for exploring these data resources, using the *SELECT* statement from SQL, which provide support for decision-making processes.

Keywords: data analysis, data warehouse, data mart, multidimensional modelling, data exploration

Wprowadzenie

Większość współczesnych aplikacji desktopowych czy internetowych wykorzystuje różnego rodzaju bazy danych. Zazwyczaj są to rozwiązania, które wspomagają procesy ewidencjonowania danych. W zależności od przeznaczenia aplikacji zawiera ona w swojej architekturze warstwę danych zbudowaną z użyciem modelu relacyjnego lub nierelacyjnego (tzw. NoSQL). Niezależnie jednak od rodzaju zastosowanego modelu bazy te zawierają obecnie ogromną ilość danych, w których ukryte są różne informacje o występujących w rzeczywistości zjawiskach i procesach. Analiza takich zasobów wymaga wykorzystania odpowiednio skonfigurowanych komputerów, które zapewnią krótki czas uzyskania wyników zapytań kierowanych do tych baz danych. Podejście to jest jednak związane ze zwiększonym kosztem budowy takiej infrastruktury sprzętowej, w której należy zastosować szybkie procesory, dużą liczbę modułów szybkich pamięci operacyjnych czy twarde dyski zapewniające krótki czas realizacji operacji I/O. Istnieje jednak inna możliwość zwiększenia efektywności analizy danych, a jest nią zastosowanie takiego modelu danych, który będzie adekwatny do potrzeb analitycznego przetwarzania zasobów danych zgromadzonych w organizacji.

Celem niniejszego opracowania jest przedstawienie metodyki projektowania hurtowni danych i data mart oraz podstawowych technik eksploatacji zgromadzonych w nich danych za pomocą instrukcji *SELECT* języka SQL,

które zapewnią zaspokojenie informacyjnych potrzeb końcowych użytkowników biznesowych.

Publikacja składa się z ośmiu rozdziałów – ich treść pozwoli Czytelnikowi zrozumieć istotę i znaczenie tworzenia analitycznych rozwiązań informatycznych, umożliwiających gromadzenie istotnego i wartościowego zasobu każdej organizacji, jakim są dane, oraz zapoznać się z technikami budowy tego rodzaju środowisk, które będą wspierać implementację strategii opartej na aktywnym wykorzystywaniu danych w procesach decyzyjnych.

W pierwszym rozdziale omówiono znaczenie analizy danych dla współczesnych menedżerów, którzy będą wykorzystywać jej rezultaty do wspomagania procesów decyzyjnych. Przedstawiono zasady poprawnego doboru różnego rodzaju metryk, których istnienie w analitycznych systemach informatycznych umożliwia poprawną ocenę sytuacji, w jakiej funkcjonuje przedsiębiorstwo.

W drugim rozdziale dokonano oceny użyteczności systemów klasy ERP w realizacji strategii zarządzania przez dane oraz wskazano korzyści wynikające z zastosowania hurtowni danych i data mart jako analitycznych źródeł danych, zapewniających wysoką jakość danych, a także wysoką efektywność ich przetwarzania.

Rozdział trzeci zawiera treści prezentujące różne rodzaje architektury platform analitycznych, w których istotną rolę odgrywają hurtownia danych i data mart. Ponadto omówiono dwie główne strategie budowy tego rodzaju struktur, wskazując na zalety i wady każdej z nich.

W rozdziale czwartym szczegółowo zaprezentowano kolejne etapy działań prowadzących do zbudowania hurtowni danych lub data mart jako głównego elementu warstwy danych aplikacji analitycznej. W tej części Czytelnik zapozna się z istotą modelowania wielowymiarowego, podstawowymi pojęciami używanymi w projektowaniu analitycznych baz danych oraz modelami konceptualnym i logicznym hurtowni danych, stanowiącymi podstawę projektu struktur danych.

Trzy podstawowe modele danych, implementowane w hurtowniach danych i data marts – tj. model gwiazdy, płatka śniegu i konstelacji faktów – przedstawiono w rozdziale piątym. Wskazano w nim również te przypadki ich użycia, które zapewniają czytelność i poprawność odwzorowania analizowanego zagadnienia oraz wysoką efektywność zaspokajania informacyjnych potrzeb użytkownika biznesowego.

Rozdział szósty zawiera treści opisujące techniki modelowania faktów w bazie danych, z uwzględnieniem różnorodności analizowanych zagadnień. Zobrazowano i szczegółowo omówiono w nim różne struktury tabel faktów, zapewniające ich adekwatność do rodzaju wymaganego opisu i poziomu jego szczegółowości.

W rozdziale siódmym zaprezentowano techniki modelowania wymiarów, a w szczególności omówiono różne rodzaje tabel wymiarów, których struktury zapewniają poprawność tworzenia kontekstu analizowanego zagadnienia, zapobiegają nadmiernemu i niepotrzebnemu wzrostowi rozmiaru bazy danych oraz jednocześnie umożliwiają osiągnięcie wysokiej wydajności zapytań SQL. Treść tego rozdziału dotyczy także tzw. wymiarów wolnozmiennych i sposobów ich opisu za pomocą odpowiednich struktur danych oraz hierarchii wymiarów jako istotnego aspektu związanego z drążeniem danych, który należy uwzględnić podczas projektowania tabel wymiarów.

W ostatnim rozdziale zaprezentowano podstawowe techniki tworzenia kodu źródłowego w języku SQL, umożliwiającego eksploatowanie danych zgromadzonych w przykładowej data mart. W tej części opracowania omówiono rozszerzenia instrukcji *SELECT* w postaci dodatkowych klauzul i funkcji, pozwalających na analizę danych.

Wszystkie treści przedstawione w rozdziałach od czwartego do ósmego dodatkowo zobrazowano przykładami, w których wykorzystano analityczną bazę danych AdventureWorks firmy Microsoft. W rozdziałach od drugiego do ósmego sformułowano także zadania do samodzielnego rozwiązania, które mają na celu utrwalenie zaprezentowanych treści i rozwinięcie umiejętności ich implementacji w praktyce.

Niniejsza książka jest przeznaczona dla Czytelników posiadających wiedzę i umiejętności w zakresie projektowania relacyjnych baz danych oraz posługiwania się językiem SQL i zainteresowanych zdobyciem nowych kompetencji umożliwiających tworzenie hurtowni danych lub data mart jako źródeł danych wykorzystywanych w działaniach realizowanych przy użyciu narzędzi analitycznych klasy business intelligence czy też data science. Jej adresatami mogą być zarówno analitycy systemowi, odpowiedzialni za transformację informacyjnych potrzeb biznesu na język technologiczny, jak i deweloperzy budujący analityczne platformy składowania danych oraz kierowane do nich zapytania SQL. Publikacja ta jest także skierowana do studentów kierunków

informatycznych, którzy poszukują kompendium wiedzy na temat projektowania hurtowni danych i data mart oraz eksploatacji zgromadzonych w nich danych.

Wszystkie przedstawione fizyczne modele danych skonstruowano przy użyciu narzędzia Oracle SQL Developer Data Modeler Version 24.3.1.347.1153, natomiast przykładowe kody źródłowe w języku SQL utworzono przy wykorzystaniu narzędzia Oracle SQL Developer Version 24.3.1.347.1826. Instrukcje w języku SQL uruchamiano w środowisku bazy danych Oracle Database 23ai, a następnie poddawano testom semantycznym oraz syntaktycznym, które potwierdziły poprawność każdego polecenia.

Zaprezentowane w opracowaniu przykłady kodów SQL zostały umieszczone na stronie WWW Katedry Informatyki Politechniki Lubelskiej pod adresem: <https://cs.pollub.pl/staff/piotrm/podrecznik-kody-sql/>.

Piotr Muryjas
Grzegorz Kozieł

1. Analityka i jej znaczenie w organizacji

Współczesne przedsiębiorstwa, działające w dynamicznie zmieniającym się otoczeniu, muszą nie tylko szybko reagować na stale pojawiające się szanse i zagrożenia, ale przede wszystkim podejmować wyzwanie właściwego przewidywania nowych trendów i budowania strategii gwarantujących sukces w perspektywie długoterminowej. Jednak analiza bieżącej sytuacji i prognozowanie przyszłości stają się coraz trudniejsze i bardziej złożone. Wiąże się to z ogromnym wzrostem ilości danych generowanych zarówno w samej organizacji, jak i jej otoczeniu.

Obecnie analiza danych w procesie zarządzania nowoczesną firmą stała się ważniejsza niż kiedykolwiek. Charakter danych wykorzystywanych w tym procesie – tj. ich ilość, złożoność i różnorodność – powoduje, że muszą zostać przekształcone do postaci przydatnej w ocenie wybranego obszaru rzeczywistości. Ta transformacja wymaga wykorzystania różnego rodzaju technik analitycznych i zaawansowanej technologii informatycznej (IT), która pozwala efektywniej i skuteczniej znajdować zależności między różnymi danymi opisującymi działania wewnątrz firmy oraz zachowania jej klientów.

Uzyskanie wysokiej skuteczności w osiąganiu celów oraz zdobycie znaczącej i trwałej przewagi konkurencyjnej wymaga wdrożenia w firmie nowego podejścia, określanego jako zarządzanie zorientowane na dane (ang. *data-driven management*), czyli – inaczej mówiąc – podejmowania decyzji z użyciem

wyników analizy danych. Podejście takie staje się wymogiem współczesności i zapewnia wysoką efektywność funkcjonowania organizacji [19]. Potencjał ukryty w powszechnym stosowaniu analizy danych umożliwia przedsiębiorstwu odkrywanie wielu nowych informacji i stwarza warunki do zwiększania własnej wartości na rynku [3]. Dlatego też analiza danych, czyli przekształcanie surowych danych w formę umożliwiającą podejmowanie decyzji, ma kluczowe znaczenie w dobie cyfryzacji wszelkich procesów biznesowych [16]. Natomiast całokształt działań, określanych jako analityka, należy postrzegać jako szeroką kategorię aplikacji, technologii i procesów, które umożliwiają pozyskiwanie, gromadzenie, udostępnianie i analizę danych oraz wspierają decydentów w zarządzaniu jednostkami [32].

Posiadanie danych transakcyjnych czy też analitycznych nie generuje jeszcze wartości dla ich właścicieli. Aby uzyskać informacje potrzebne do oceny sposobu funkcjonowania firmy, konieczna jest wizualizacja danych w odpowiedni, czytelny i zrozumiały dla człowieka sposób. Można to zrobić za pomocą narzędzi business intelligence (BI), których użycie pozwoli jeszcze lepiej uwydatnić wartość biznesową danych i wzmocni konieczność implementacji idei aktywnego posługiwania się nimi w zarządzaniu współczesnym przedsiębiorstwem.

Realizacja koncepcji zarządzania przez dane wymaga wykorzystywania technologii informatycznych, a w szczególności analitycznych platform składowania danych i narzędzi analizy danych. Istotne znaczenie ma także właściwy sposób modelowania zagadnienia będącego przedmiotem analizy. W procesie tym należy uwzględnić te wszystkie aspekty, które w późniejszym projekcie bazy danych znajdą swoje odzwierciedlenie w postaci tabel oraz związków między nimi. Szczególnie ważne jest precyzyjne i jednoznaczne zdefiniowanie pojęć użytych jako deskryptory oraz miary. Pierwsze z nich będą określały kontekst analizowanego zagadnienia, natomiast drugie będą miały charakter opisu ilościowego, tzn. wyrażonego zawsze liczbami. Ich poprawna identyfikacja na etapie modelowania rzeczywistości przyczyni się do sprawnego zbudowania analitycznej bazy danych i szybkiego wprowadzenia do użytkowania tworzonej platformy wspomagania procesów decyzyjnych.

Zapewnienie wysokiej użyteczności analitycznej bazy danych i zintegrowanych z nią narzędzi analitycznych wymaga właściwego wyboru miar i wskaźników, które dostarczą wszystkich wymaganych informacji do oceny

bieżącej sytuacji czy też przewidywania kierunków jej rozwoju. Jednak nadmierna liczba tych wielkości znacząco utrudnia to zadanie. Dlatego też w literaturze przedmiotu [21] proponuje się ich pogrupowanie w następujący sposób:

- kluczowe wskaźniki wyników (KRI – ang. *Key Results Indicators*) dostarczają zarządowi ogólnych podsumowań wyników przedsiębiorstwa;
- wskaźniki wyników (RI – ang. *Results Indicators*) informują menedżerów o rezultatach łączenia się zespołów w celu realizacji określonych zadań;
- wskaźniki wydajności (PI – ang. *Performance Indicators*) informują menedżerów o efektywności pracy poszczególnych zespołów;
- kluczowe wskaźniki wydajności (KPI – ang. *Key Performance Indicators*) informują menedżerów, jak firma radzi sobie 24 godziny na dobę, 7 dni w tygodniu, codziennie lub co tydzień w zakresie kluczowych czynników sukcesu.

W celu efektywnego wykorzystania informatycznych platform analitycznych zaleca się zachowanie właściwych proporcji między liczbą poszczególnych miar. Zaleca się używanie około 10 KRI, do 80 RI i PI oraz 10 KPI [21]. Z kolei Robert Kaplan i David Norton zalecają nie więcej niż 20 KPI, a Jeremy Hope i Robin Fraser sugerują mniej niż 10 KPI [13].

Patrząc na przedmiot analizy z punktu widzenia czasu, można wyróżnić miary do opisu przeszłości, teraźniejszości i przyszłości. Według Davida Parmentera właściwy zestaw powinien zawierać mieszankę 60 proc. miar przeszłych, 20 proc. bieżących i 20 proc. zorientowanych na przyszłość [21].

Niezależnie od rodzaju miar i wskaźników podczas budowy i implementacji analitycznego modelu rzeczywistości należy uwzględnić takie ich rodzaje, które zapewnią realizację trzech podstawowych funkcji [11]:

- kontrola – wskaźniki pozwalają menedżerom i pracownikom na ocenę i kontrolę wydajności zasobów, którymi mają zarządzać;
- komunikacja – wskaźniki przekazują informacje o wydajności wewnętrznym pracownikom i menedżerom, a także zewnętrznym interesariuszom;
- poprawa – wskaźniki identyfikują luki pomiędzy wydajnością a celami, które wskazują drogę do możliwych działań naprawczych; rozmiar tych luk i ich kierunek (np. pozytywny lub negatywny) można wykorzystać do dostosowania/zaplanowania działań naprawczych.

Zestaw miar, tj. ich liczba i rodzaje, wykorzystywanych w ocenie procesów i osób, a także w procesie podejmowania decyzji jest silnie powiązany

z obszarem działalności firmy i jej wielkością. Duże przedsiębiorstwa stosują znacznie więcej miar niż te należące do grupy MŚP. Jednak w każdym przypadku podczas projektowania miar warto unikać błędów, które czynią analizę bezużyteczną i prowadzą do błędnych wniosków. Do najczęściej popełnianych należą: wykonywanie pomiarów zbyt wielu działań i zjawisk, traktowanie wszystkich miar jako KPI, utożsamianie rezultatów końcowych z wynikami pomiaru operacyjnego, brak powiązania wybranych miar z krytycznymi czynnikami sukcesu organizacji oraz zbytne poleganie na przeszłych miarach [21].

Właściwy wybór miar jest kluczowy dla powodzenia realizacji projektów informatycznych mających na celu budowę i wdrożenie analitycznych platform wspomagania decyzji, a tym samym dla sukcesu i użyteczności implementacji analityki we współczesnym przedsiębiorstwie.

2. Analityczne źródła danych w organizacji

Współczesne przedsiębiorstwa wykorzystują wiele systemów informatycznych, które zazwyczaj wspomagają bieżącą działalność operacyjną poprzez gromadzenie danych opisujących pojedyncze procesy biznesowe. Jednak z analitycznego punktu widzenia mają one pewne wady, które uniemożliwiają ich efektywne zastosowanie jako wsparcie w podejmowaniu decyzji. W niniejszym rozdziale zostaną przedstawione różne rodzaje źródeł danych wykorzystywanych przez narzędzia business intelligence czy data mining, za pomocą których realizowane są procesy eksploatacji danych, będące istotnym czynnikiem wpływającym na jakość i skuteczność działań umożliwiających osiągnięcie przyjętych celów.

2.1. Systemy klasy ERP

Tradycyjne systemy transakcyjne, określane także jako systemy klasy ERP, są najczęstszym narzędziem technologii informatycznych (IT), wspomagającym funkcjonowanie niemal wszystkich współczesnych organizacji biznesowych, publicznych czy administracji państwowej lub lokalnej. Ich główne przeznaczenie to ewidencjonowanie różnych zdarzeń i procesów zachodzących w codziennej działalności firmy. Z tego względu realizowane są w nich przede wszystkim operacje wstawiania, modyfikowania i usuwania danych. W celu zapewnienia

ich wysokiej efektywności w systemach tej klasy implementuje się relacyjny model danych, który dodatkowo eliminuje redundancję danych i zapewnia ich spójność. Dane te pozyskiwane są w czasie rzeczywistym, dzięki czemu tworzą najbardziej aktualny opis przebiegu procesów biznesowych. Wydaje się zatem, że systemy transakcyjne mogą stanowić doskonałe źródło danych wspomagających podejmowanie decyzji. Pewnym potwierdzeniem tego faktu są różne moduły analityczne, wbudowane w większość systemów klasy ERP, umożliwiające bieżące raportowanie. Jednak wielokryterialna ocena tego rodzaju rozwiązań wskazuje, że nie są one odpowiednie jako analityczne źródła danych [18]. Jako uzasadnienie tego stwierdzenia podaje się następujące powody:

- zbyt duża szczegółowość danych jak na potrzeby analityczne (wiele danych transakcyjnych jest zbędnych w analitycznej bazie danych);
- większość zgromadzonych danych opisuje rzeczywistość w krótkim horyzoncie czasowym (zazwyczaj jednego roku);
- duża zmienność danych, utrudniająca lub nawet uniemożliwiająca identyfikację wzorców czy trendów;
- występowanie problemów z poprawnością i jakością danych (mechanizmy walidacji działające wyłącznie na poziomie wierszy opisujących pojedyncze transakcje, a nie wielu systemów informatycznych; występowanie błędnych danych lub ich braki);
- koncentracja funkcjonalności systemów ERP na pojedynczych procesach, a nie na zagadnieniach, które mają być analizowane;
- długi czas wykonywania zapytań analitycznych, spowodowany normalizacją danych i dużą liczbą złączeń między tabelami;
- bardzo ograniczone raportowanie analityczne ze względu na możliwość tworzenia stosunkowo prostych zapytań do bazy danych.

Mając świadomość wymienionych ograniczeń systemów transakcyjnych, należy podchodzić ze znaczną rezerwą do rozwiązań, w których różnego rodzaju analizy, szczególnie te wspierające decyzje średnio- i długoterminowe, wykonywane są z wykorzystaniem tych źródeł danych.

Implementacja takiego podejścia obarczona będzie znacznym ryzykiem posługiwania się ograniczonymi lub nieprawidłowymi danymi, których wykorzystanie w procesach decyzyjnych może doprowadzić do wystąpienia negatywnych zjawisk w funkcjonowaniu organizacji. Dlatego też konieczne jest

zastosowanie źródeł danych o innym charakterze, lepiej dostosowanych do analitycznych wymagań użytkowników końcowych.

2.2. Hurtownia danych i data mart

Ograniczenia systemów klasy ERP w kontekście ich użycia jako analitycznego źródła danych wynikają z ich natury i przeznaczenia. Pozyskiwanie i gromadzenie danych, opisujących określoną rzeczywistość, są konieczne, aby w przyszłości można było je wykorzystać podczas tworzenia analitycznych platform. Jednak z uwagi na przedstawione w poprzednim rozdziale zastrzeżenia, dotyczące użycia systemów transakcyjnych jako jedyne wsparcia procesów decyzyjnych, warto rozważyć możliwość stworzenia takiego rozwiązania, które umożliwi wyeliminowanie tych wad oraz lepsze dostosowanie struktury i kontentu zasobów danych do analitycznych wymagań użytkowników.

W latach 90. ubiegłego wieku w Stanach Zjednoczonych zaprezentowano nowoczesną jak na ówczesne czasy koncepcję analitycznej bazy danych, całkowicie zmieniającą dotychczasowe, tj. relacyjne, podejście do sposobu gromadzenia danych. Za pionierów i wizjonerów tej myśli uznaje się Williama Inmona i Ralpha Kimballa, którzy stworzyli pojęcie i zasady budowy hurtowni danych jako centralnego zasobu danych, umożliwiającego zaspokojenie informacyjnych potrzeb wszystkich użytkowników podejmujących decyzje na różnych szczeblach organizacji. Podstawowe założenia tego podejścia definiowane są następująco [12]:

- zorientowanie tematyczne – dane w hurtowni opisują zagadnienia i funkcje biznesowe, a nie procesy realizowane w organizacji;
- jednolita postać danych – w wyniku integracji danych z różnych źródeł są one zapisane w sposób spójny i zrozumiały dla każdego użytkownika, niewymagający translacji za pomocą dodatkowych zasobów danych;
- niezmiennosc danych – raz wprowadzone do hurtowni dane pozostają w niej w pierwotnej postaci i nie podlegają modyfikacji czy usunięciu, co zapewnia możliwość oceny zmian zachodzących zarówno w organizacji, jak i jej otoczeniu;

- czas jako konieczna cecha opisu – dane charakteryzujące każde zagadnienie w hurtowni są zawsze oznaczone czasem, który wskazuje moment ich powstania, dzięki czemu możliwa jest ocena ich zmienności na przestrzeni kolejnych okresów.

Główna różnica między koncepcją reprezentowaną przez W. Inmona i R. Kimballa polega na sposobie dochodzenia do finalnego rozwiązania, jakim jest hurtownia danych. W. Inmon proponuje, by obejmowała ona swoim zakresem całą organizację i była zbudowana bezpośrednio z wykorzystaniem wszystkich zasobów danych istniejących w organizacji, bez potrzeby tworzenia pośrednich warstw [28]. Z kolei R. Kimball uważa, że w pierwszej kolejności należy zbudować departamentowe hurtownie danych, które określił mianem *data marts*, a następnie zintegrować je w jeden centralny zasób [15]. W konsekwencji tych dwóch odmiennych podejść proces projektowania i implementacji hurtowni danych będzie realizowany według różnych strategii, które opisano w trzecim rozdziale.

Wśród głównych zalet podejścia proponowanego przez R. Kimballa wymieniamy się [15]:

- krótki czas budowy rozwiązania analitycznego, wspierającego użytkownika końcowego w procesach decyzyjnych;
- znacznie niższy koszt projektowania i implementacji *data mart* w porównaniu z hurtownią danych;
- krótki czas generowania odpowiedzi na różnego rodzaju zapytania analityczne.

W przypadku koncepcji W. Inmona wskazuje się następujące istotne korzyści jej implementacji [12]:

- stworzenie jednego centralnego źródła prawdy na temat funkcjonowania organizacji w wielu obszarach;
- zapewnienie możliwości raportowania i wykonywania złożonych analiz, obejmujących swoim zakresem wiele departamentów.

Wymienione powyżej zalety wdrożenia obydwu idei wskazują, że stworzenie analitycznej bazy danych – czy to w postaci hurtowni danych, czy też *data mart* – jest działaniem zapewniającym efektywne wykorzystanie danych zgromadzonych w systemach transakcyjnych. Decydując się na wybór jednej z nich, należy wziąć pod uwagę także wady tych rozwiązań. Główne zastrzeżenia związane z budową *data mart* dotyczą koncentrowania się wyłącznie na

pojedynczych zagadnieniach, często niepowiązanych ze sobą, oraz możliwości wystąpienia niespójnych danych, dotyczących tych samych zagadnień, w różnych zasobach tego rodzaju. W odniesieniu do hurtowni danych jako wady wymienia się bardzo wysoki koszt jej budowy i wdrożenia, dużą złożoność administrowania tym zasobem oraz dłuższy, w porównaniu z data mart, czas wykonywania zapytań do bazy danych [17]. W celu lepszego uwidocznienia różnic pomiędzy hurtownią danych i data mart w tabeli 2.1 zestawiono ich cechy – zarówno te o charakterze organizacyjnym, jak i technicznym.

Tab. 2.1. Cechy data mart i hurtowni danych

Cecha	Data mart	Hurtownia danych
Przeznaczenie	Wsparcie decyzji taktycznych (np. wdrożenie programu lojalnościowego, nowa kampania reklamowa)	Wsparcie decyzji strategicznych (np. wprowadzenie nowego produktu, nowe rynki zbytu)
Zakres	Koncentracja na zagadnieniach w obrębie jednego działu firmy (np. zarządzanie produkcją)	Koncentracja na wielu zagadnieniach we wszystkich działach organizacji
Złożoność projektu budowy	Średnia	Bardzo duża
Model danych	Gwiazda, płatek śniegu	Konstelacja faktów
Rodzaj składowanych danych	Większość danych w postaci transakcyjnej lub lekko zagregowanej	Dane o różnym stopniu szczegółowości, duża ilość danych transakcyjnych, także dane średnio i silnie zagregowane
Źródło danych	Pojedyncze systemy transakcyjne, źródła zewnętrzne lub hurtownia danych	Wiele systemów transakcyjnych, data marts i źródeł zewnętrznych
Rozmiar bazy danych	Zbiory danych rzędu kilkudziesięciu GB	Olbrzymie zbiory danych rzędu kilkuset GB lub wielu TB
Czas budowy i wdrożenia	Od kilku tygodni do kilku miesięcy	Od wielu miesięcy do nawet kilku lat
Koszty budowy i wdrożenia	Niskie lub średnie (znacznie niższe niż dla hurtowni danych)	Bardzo wysokie
Wydajność zapytań	Bardzo wysoka (mniejsze zbiory danych, zapytania o niskim lub średnim stopniu złożoności)	Niższa w porównaniu z data mart (z uwagi na rozmiar danych i znaczną złożoność zapytań)

Źródło: opracowanie własne.

Mimo upływu lat koncepcje W. Inmona i R. Kimballa wciąż stanowią podstawę budowy wszelkiego rodzaju analitycznych baz danych. Wprawdzie

podane przez obydwu pionierów zasady projektowania i wdrożenia takich rozwiązań nie zawierają gotowych procedur działania, to jednak dostarczają wiedzy na temat powodów i intencji zastosowania hurtowni danych czy data mart we współczesnych organizacjach. Świadomość istnienia tych zasad z pewnością ułatwi realizację projektu budowy tego typu analitycznej platformy składowania danych oraz pozwoli zwiększyć prawdopodobieństwo jej sukcesu i wartość biznesową.

2.3. Zadania do samodzielnego wykonania

1. Proszę wskazać istotne różnice między hurtownią danych i data mart.
2. Dla przykładowej firmy, która świadczy na całym świecie usługi typu bike-sharing, tj. wypożyczenia rowerów w obrębie miasta, proszę wskazać te obszary jej funkcjonowania, w których wdrożenie data mart przyniesie wymierne korzyści, a także określić przykładowe korzyści w poszczególnych obszarach.
3. Proszę wskazać te obszary działalności Narodowego Funduszu Zdrowia, w których wdrożenie data mart przyniesie wymierne korzyści, oraz wymienić przykładowe korzyści w poszczególnych obszarach.

3. Architektura platform analitycznych

Budowa analitycznej bazy danych jako rozwiązania, które integruje w sobie dane pochodzące z wielu źródeł znajdujących się zarówno w dotychczasowej infrastrukturze IT firmy, jak i w jej otoczeniu, jest procesem bardzo złożonym i wymaga olbrzymiego doświadczenia w obszarze zarządzania projektami, zarządzania danymi oraz samego projektowania i implementacji systemów informatycznych. Zanim rozpocznie się jakiegokolwiek prace związane z tworzeniem takiej bazy danych, konieczne jest określenie architektury, która zdefiniuje związek tego rodzaju zasobu z innymi systemami IT organizacji oraz sposób jego zasilania danymi.

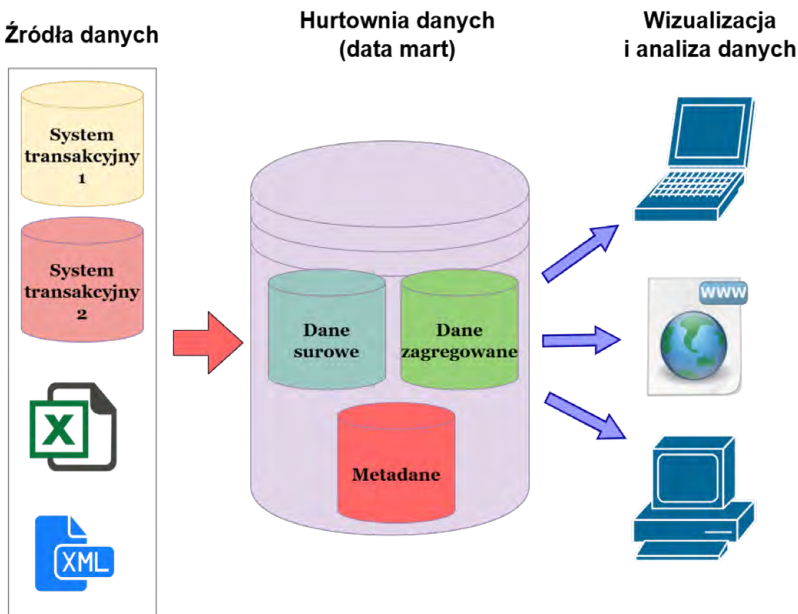
3.1. Rodzaje architektury platform analitycznych

Wraz z rozwojem technologii bazodanowych, rosnącą ilością danych i złożonością procesów ich transformacji oraz pojawieniem się nowoczesnych koncepcji zarządzania przez dane zmieniały się założenia budowy systemu informacyjnego organizacji, w którym analityczna baza danych będzie odgrywała kluczową rolę jako wsparcie w procesie podejmowania decyzji. Na przestrzeni wielu lat istnienia hurtowni danych i data mart architektura, w której rozwiązania te są implementowane, zmieniała się w taki sposób, aby zapewnić

użytkownikom końcowym jak najlepsze wykorzystanie posiadanych zasobów danych w działaniach zapewniających osiągnięcie przyjętych celów funkcjonowania organizacji.

Na rysunku 3.1 zobrazowano podstawową architekturę platformy analitycznej, w której występują trzy warstwy [23]:

- źródła danych;
- hurtownia danych lub data mart;
- wizualizacja i analiza danych.



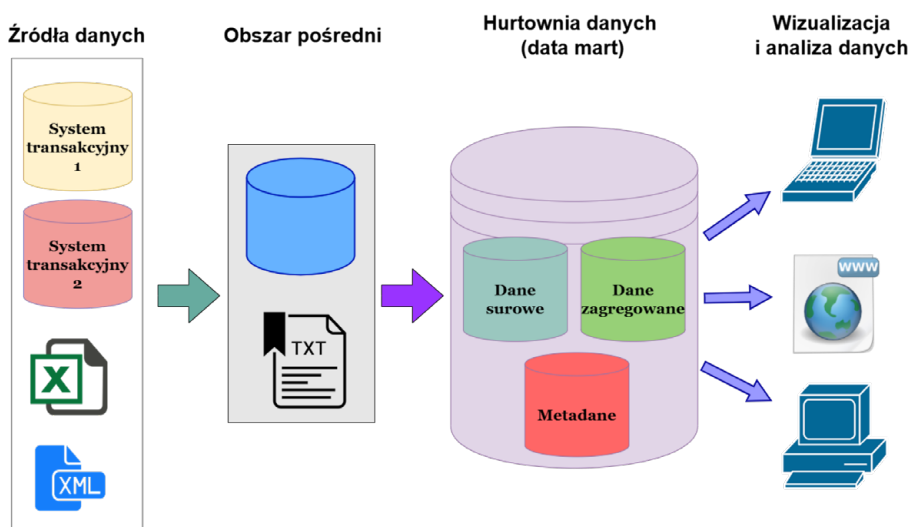
Rys. 3.1. Podstawowa architektura platformy analitycznej

Źródło: opracowanie własne na podstawie [23].

Warstwa źródeł danych obejmuje wszystkie zasoby danych, jakimi dysponuje organizacja, oraz te, do których ma dostęp w swoim otoczeniu. Zazwyczaj są to wszelkiego rodzaju systemy klasy ERP, które wspomagają bieżącą działalność operacyjną, bądź też własne lub obce pliki płaskie (np. arkusze kalkulacyjne, tekstowe csv, json czy xml), udostępniane poprzez API ze źródeł zewnętrznych. Dane te poddawane są przetwarzaniu w procesach ETL (ang. *Extract, Transform, Load*), których rezultatem jest ich załadowanie do docelowej analitycznej bazy danych. Ze względu na różnorodność potrzeb informacyjnych użytkowników

końcowych w bazie tej alokowane są zarówno oczyszczone dane surowe, często przypominające dane transakcyjne, jak i dane o pewnym stopniu zagregowania. Dodatkowo składowane są tzw. metadane, które opisują m.in. związek między danymi transakcyjnymi i analitycznymi, ich pochodzenie i autorów czy też zasady dostępu. Trzecia warstwa architektury reprezentuje użytkowników końcowych i ich działania podejmowane z użyciem dedykowanych narzędzi eksploatacji danych, tj. analizy wizualnej i zaawansowanych technik data science w celu wygenerowania i upublicznienia wyników tych operacji w różnych formatach i na różnych urządzeniach.

W celu optymalizacji działań, związanych z zasilaniem hurtowni danych czy data mart, definiuje się zmodyfikowaną architekturę, którą przedstawiono na rysunku 3.2 [23].



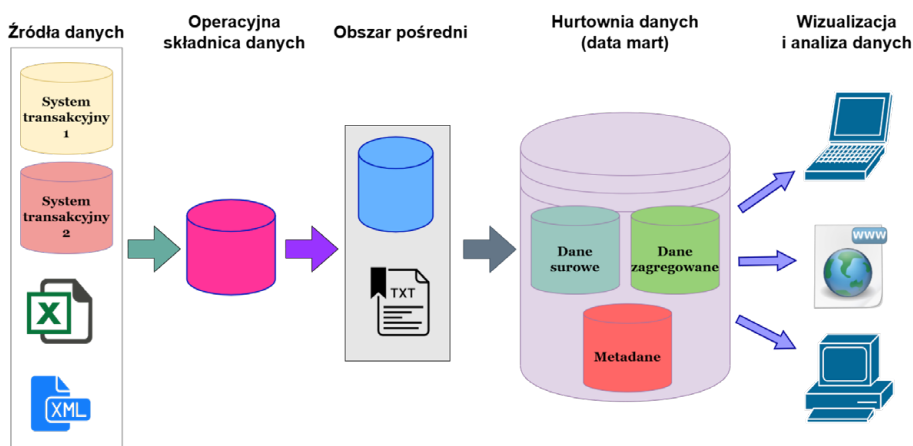
Rys. 3.2. Architektura platformy analitycznej z obszarem pośrednim

Źródło: opracowanie własne na podstawie [23].

W tej koncepcji występuje tzw. obszar pośredni (ang. *staging area*), będący tymczasową przestrzenią, w której odbywa się oczyszczanie i walidacja danych źródłowych przed poddaniem ich ostatecznej transformacji i załadowaniu do docelowej analitycznej bazy danych. Jest to zazwyczaj dodatkowy serwer bazodanowy, pełniący funkcję kontenera, w którym umieszcza się kopię wymaganych danych transakcyjnych oraz dane wykorzystywane wielokrotnie

w kolejnych zasilaniach hurtowni. Jego obecność znacząco skraca czas dostępu do danych źródłowych i realizacji procesów ETL. Alternatywną postacią formatu przechowywania oczyszczonych danych w obszarze pośrednim są pliki płaskie, które mogą być efektywnie przetwarzane w narzędziach integracji danych.

Kolejnym rodzajem architektury platformy analitycznej jest wariant, w którym występuje warstwa tzw. operacyjnej składnicy danych (ang. ODS – *Operational Data Store*) [29]. Architekturę w tej postaci zaprezentowano na rysunku 3.3.



Rys. 3.3. Architektura platformy analitycznej z obszarem pośrednim i ODS

Źródło: opracowanie własne na podstawie [29].

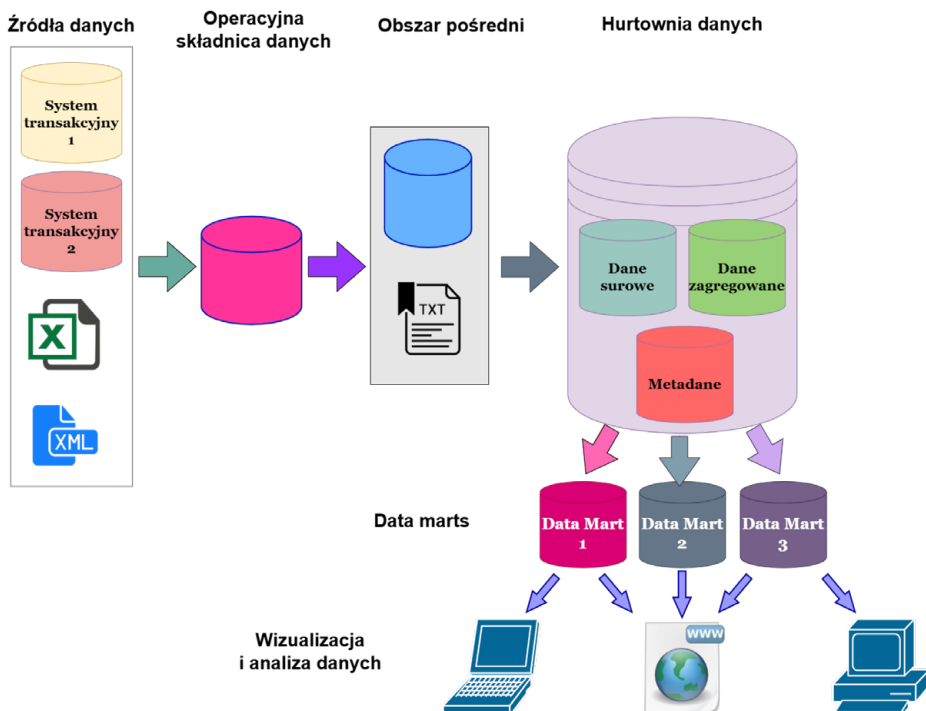
Operacyjna składnica danych to miejsce zgromadzenia danych pobranych bezpośrednio z wielu źródłowych systemów transakcyjnych. Mają one postać identyczną jak w miejscach, z których pochodzą, tj. są w postaci znormalizowanej z zachowaniem ich natywnych formatów i znaczenia, jednak zostały już wstępnie oczyszczone. Niekiedy dokonuje się także ich niewielkiej agregacji w celu wstępnego przygotowania dla przyszłych działań. Zasób ten przeznaczony jest do operacyjnego raportowania o znacznie szerszym zakresie informacyjnym w porównaniu z tym, jaki można uzyskać w jednostkowych systemach źródłowych, co znacząco zwiększa jego wartość biznesową [5]. Zapewnia on pozyskanie bieżącego i całościowego obrazu sytuacji w organizacji, w którym uwzględnia się wieloprzekrojowe widoki na realizowane w niej procesy.

Operacyjna składnica danych jest zasilana w prostych procesach ETL – mniej złożonych niż te, które zasilają hurtownię danych. Także zakres i ilość tych danych jest znacznie mniejsza w porównaniu z bazą analityczną, dzięki czemu zapytania kierowane do ODS mają prostszą strukturę i są wykonywane znacznie szybciej. Ponieważ głównym celem istnienia ODS jest wspieranie raportowania dotyczącego aktualnej sytuacji w firmie, dane w niej składowane mają krótki czas życia i są nadpisywane nowymi partiami pojawiającymi się w krótkich okresach czasu, zazwyczaj tygodniowych lub maksymalnie miesięcznych. Operacyjną składnicę danych należy postrzegać jako istotną warstwę w architekturze platformy analitycznej, gdyż pozwala użytkownikowi końcowemu dostrzec jej wartość w procesach decyzyjnych już na wczesnym etapie jej tworzenia. Tej użyteczności nie posiada obszar pośredni, gdyż zgromadzone w nim dane mają postać tymczasową, będącą wynikiem kolejnych procesów ETL. Obecność ODS ma jeszcze jedną zaletę – pozwala zwiększyć efektywność procesów ETL, gdyż są one realizowane częściej i obejmują swoim zakresem mniejsze ilości danych, co pozwala uniknąć zakłóceń bieżącej pracy systemów ERP.

Kolejny rodzaj architektury platformy analitycznej (rys. 3.4) łączy w sobie pewne uprzednio przedstawione warstwy oraz nową, stanowiącą pewnego rodzaju ekstrakcję zasobów zgromadzonych w hurtowni danych [23].

Wyodrębnienie jednostkowych data marts ma na celu ułatwienie różnym grupom użytkowników końcowych korzystanie z zasobów danych dedykowanych wyłącznie zaspokojeniu ich potrzeb informacyjnych, z jednoczesnym zachowaniem możliwości utworzenia dowolnego widoku przekrojowego, obejmującego swoim zakresem wszystkie obszary działalności. Architektura, w której z centralnej hurtowni danych wydzielane są data marts, zaproponował W. Inmon. Implementacja tej koncepcji odbywa się w ramach realizacji strategii określanej mianem Top-Down. Jej zalety są następujące:

- wspomaganie procesów decyzyjnych we wszystkich obszarach biznesu poprzez stworzenie jednego i spójnego obrazu organizacji;
- synergia korzyści eksploatacji hurtowni w różnych obszarach biznesu;
- możliwość łatwego tworzenia (ekstrakcji) kolejnych hurtowni tematycznych;
- spójność struktur danych i samych danych w obrębie całej organizacji;
- łatwość zarządzania jednym centralnym zasobem danych.



Rys. 3.4. Architektura platformy analitycznej z obszarem pośrednim, ODS i data marts (strategia Top-Down)

Źródło: opracowanie własne na podstawie [23].

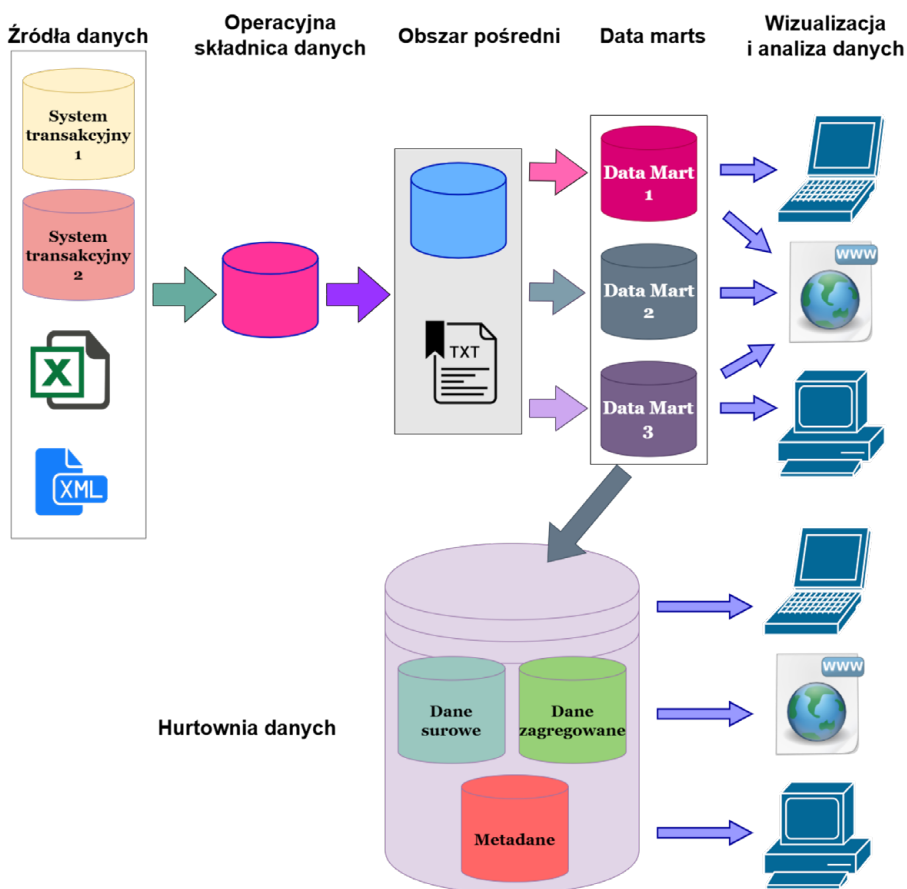
Z punktu widzenia technicznych aspektów eksploatacji i utrzymania hurtowni wśród korzyści implementacji podejścia Top-Down wymienia się [1]:

- 40% redukcję nadmiarowości danych;
- 30% poprawę spójności danych;
- 25% szybszą implementację raportowania w całej organizacji;
- 35% lepsze dostosowanie do prawnych wymogów regulacyjnych.

Jednak realizacja wielu projektów, w których zastosowano taką strategię, ujawniła również wiele jej wad, do których należy zaliczyć [22]:

- długi czas budowy rozwiązania;
- wyższe ryzyko nieosiągnięcia celów stawianych hurtowni danych;
- konieczność wprowadzania dużych zmian w projekcie hurtowni danych w przypadku zmiany sposobu realizacji danej funkcji biznesowej;
- wysoki koszt realizacji przedsięwzięcia;
- znaczny wzrost wielkości zasobów danych.

Alternatywne podejście do budowy hurtowni danych reprezentuje R. Kimball, który zaproponował odmienną strategię, określaną jako Bottom-Up, zwiększającą prawdopodobieństwo odniesienia sukcesu budowy i wdrożenia analitycznej bazy danych. W strategii tej wykorzystuje się architekturę przedstawioną na rysunku 3.5.



Rys. 3.5. Architektura platformy analitycznej z obszarem pośrednim, ODS i data marts (strategia Bottom-Up)

Źródło: opracowanie własne na podstawie [23].

W tym przypadku zakłada się budowę wielu odrębnych data marts, które docelowo będą zintegrowane do centralnej hurtowni danych. Jako zalety tego rozwiązania należy wymienić:

- krótszy całkowity czas realizacji projektu niż w przypadku strategii Top-Down;
- krótszy czas uzyskania wymiernych korzyści biznesowych przez użytkowników końcowych;
- niższe ryzyko niepowodzenia przedsięwzięcia;
- uniknięcie błędów podczas realizacji kolejnych projektów budowy data mart (uczenie się zespołu);
- wysoką skalowalność i adaptowalność data mart do zmian zachodzących w organizacji;
- zwiększenie elastyczności tworzenia różnych przekrojów informacyjnych przez użytkowników biznesowych poprzez większą ziarnistość (poziom szczegółowości) danych w data mart.

Wśród innych korzyści związanych z realizacją strategii Bottom-Up wskazuje się również te, które mają związek z postrzeganiem data mart jako istotnej inwestycji, służącej wzrostowi konkurencyjności firmy na rynku. Należą do nich [1]:

- 45% szybsza początkowa implementacja;
- 50% niższe koszty początkowego rozwoju;
- 60% wyższe wskaźniki adopcji przez użytkowników końcowych;
- 35% bardziej elastyczna adaptacja do zmian.

Wiele firm, tworząc analityczne platformy IT wspierające procesy decyzyjne, doceniło zalety strategii Bottom-Up i zdecydowało się ją wdrożyć podczas realizacji swoich projektów. Jednak ich uczestnicy zwracają uwagę także na pewne wady tego podejścia, do których zaliczono [22]:

- wysokie ryzyko popełnienia błędów podczas realizacji pierwszych projektów (pierwszy data mart jest pewnego rodzaju prototypem);
- możliwość wystąpienia niespójności danych zgromadzonych w data mart z danymi znajdującymi się w zasobach innych działów organizacji;
- trudności w utrzymaniu wysokiej jakości danych i ich jednolitego znaczenia w odrębnych data marts;
- wysoki poziom trudności i złożoności późniejszej integracji pojedynczych data marts do postaci centralnej hurtowni danych;

- konieczność zarządzania odrębnymi środowiskami bazodanowymi.

Implementacja jednej z wyżej opisanych strategii uzależniona jest od wielu czynników, zarówno o charakterze organizacyjnym, jak i technicznym. Dlatego też sugeruje się zastosowanie podejścia Top-Down w tych organizacjach, dla których najistotniejsze są [1]:

- efektywne zarządzanie zgodnością z obowiązującymi przepisami prawnymi;
- globalna standaryzacja danych;
- wielofunkcyjność rozwiązania wsparta rozległą integracją danych;
- efektywne i skuteczne wsparcie długoterminowego planowania strategicznego.

Natomiast strategię Bottom-Up zaleca się realizować w przypadku, gdy [1]:

- wymagane jest uzyskanie szybkich efektów wdrożenia data mart;
- istniejące systemy ERP nie są ze sobą zintegrowane i nie umożliwiają międzyplatformowej wymiany danych;
- poszczególne działy firmy są autonomiczne i nie wykorzystują zasobów informacyjnych pochodzących z innych departamentów;
- w firmie realizowane są zwinne procesy biznesowe generujące dane, które należy efektywnie alokować w analitycznych bazach danych.

Warto jednak podkreślić, że – niezależnie od przyjętego podejścia – wdrożenie hurtowni danych czy data mart we współczesnej organizacji stanowi działanie ukierunkowane na monetyzację posiadanych danych, generujące wymierne korzyści biznesowe i techniczne, a przede wszystkim zwiększające jej konkurencyjność na rynku.

3.2. Zadania do samodzielnego wykonania

1. Proszę wskazać różnice między obszarem pośrednim i operacyjną składnicą danych oraz określić rolę każdego z tych rozwiązań w architekturze platformy analitycznej.
2. Proszę scharakteryzować architekturę platformy analitycznej, będącą rezultatem implementacji strategii jej budowy typu Top-Down.
3. Proszę wskazać zalety i wady architektury platformy analitycznej zbudowanej w ramach realizacji strategii Bottom-Up.

4. Modelowanie wielowymiarowe w projektowaniu hurtowni danych

Modelowanie hurtowni danych czy data mart to działanie mające na celu zdefiniowanie logicznej i fizycznej struktury danych służących jako wsparcie w procesie podejmowania decyzji. Kluczowe etapy obejmują analizę wymagań informacyjnych użytkownika biznesowego, zbudowanie modeli: konceptualnego i logicznego, oraz ich implementację w wybranym środowisku bazodanowym. Proces tworzenia modeli może być realizowany z wykorzystaniem różnych metodyk, takich jak modelowanie wielowymiarowe czy Data Vault 2.0. Zastosowanie jednego z tych podejść zależy od jakości, wielkości i złożoności danych oraz sposobu ich wykorzystania.

W tym rozdziale Czytelnik zapozna się z metodyką projektowania analitycznej bazy danych, określaną jako modelowanie wielowymiarowe. Przedstawiono w nim podstawowe pojęcia stosowane podczas tworzenia takiego modelu oraz kolejne fazy jego budowy. Dla pełnego zrozumienia omawianych treści przytoczono przykłady obrazujące rezultaty wykonania poszczególnych etapów działań. Opis metodyki Data Vault 2.0, ze względu na obszerność tej tematyki, zostanie zamieszczony w odrębnym opracowaniu.

4.1. Istota modelowania wielowymiarowego

Funkcjonowanie współczesnych organizacji charakteryzuje się dużą zmiennością i złożonością procesów zachodzących zarówno w jej strukturach wewnętrznych, jak i otoczeniu. W związku z tym potrzeby informacyjne użytkowników końcowych aplikacji analitycznych są bardzo zróżnicowane i wymagają przyjęcia różnorodnych punktów widzenia na określony obszar działania. Dodatkowo ocena wybranych zagadnień, związanych z funkcjonowaniem przedsiębiorstwa, wymusza konieczność określenia takiego poziomu szczegółowości ich opisu, który zapewni uzyskanie wartościowych rezultatów analiz. Znajomość tego poziomu przekłada się na zakres danych, jakie są niezbędne w bazie danych, aby skutecznie wspierać procesy decyzyjne za pomocą tworzonych aplikacji.

Głównym celem modelowania wielowymiarowego jest uzyskanie optymalnej struktury danych w hurtowni danych czy data mart, która zapewni krótki czas wykonywania analitycznych zapytań do tabel. Zapytania takie będą agregować dane z użyciem wielu kolumn, zazwyczaj pochodzących z różnych tabel, i jednocześnie realizować typowe operacje analityczne, takie jak sumowanie, zliczanie, wyznaczanie wartości średniej, maksymalnej czy minimalnej. Analizując plany wykonania tego rodzaju zapytań, można zauważyć, że charakteryzują się one długim czasem uzyskania odpowiedzi. Świadomość tego powinna być istotnym argumentem dla projektanta hurtowni danych do poszukiwania takiego rozwiązania, które zapewni znaczącą redukcję tego czasu.

Zasadnicza różnica między modelowaniem wielowymiarowym a relacyjnym polega na tym, że to pierwsze zorientowane jest na wspieranie procesu dostarczania danych zagregowanych. Natomiast modelowanie relacyjne ma na celu wspomaganie procesów ewidencjonowania danych. Konsekwencją odmienności celów zastosowania tych podejść jest obecność znacznej redundancji danych w analitycznych bazach danych, podczas gdy w relacyjnych bazach dane występują w znormalizowanej postaci w celu uniknięcia ich nadmiarowości. Rezultat modelowania wielowymiarowego zapewni krótki czas wykonywania operacji dominującej w hurtowni danych, czyli instrukcji *SELECT*. Natomiast model relacyjny będzie efektywnie wspierał operacje *INSERT*, *UPDATE* i *DELETE*.

Modelowanie wielowymiarowe jest nie tylko metodyką projektowania struktur analitycznej bazy danych, lecz także istotnym wsparciem w budowie tego rodzaju platform. Podejście takie często implementuje się już na etapie opracowania koncepcji hurtowni czy data mart podczas tworzenia jej modelu konceptualnego i logicznego. Na rysunku 4.1 przedstawiono schemat obrazujący kolejne etapy, jakie należy zrealizować, aby poprawnie zaprojektować bazę danych zorientowaną na składowanie danych analitycznych.



Rys. 4.1. Etapy projektowania analitycznej bazy danych

Źródło: opracowanie własne.

W modelowaniu wielowymiarowym konieczne jest posługiwanie się pewnymi pojęciami, charakterystycznymi dla tej domeny rozwiązań IT. Realizacja etapów drugiego, trzeciego, czwartego i piątego wymaga znajomości takich pojęć, jak: fakt, wymiar, atrybut wymiaru, miara.

Jako fakt należy rozumieć pewien zawężony obszar rzeczywistości bądź też zagadnienie, które będą poddane analizie umożliwiającej wygenerowanie danych wspomagających realizację procesów decyzyjnych. Pamięając o tym, że hurtownia danych i data mart są zorientowane tematycznie, a nie procesowo, faktem może być zarówno działanie – np. *Sprzedaż online* lub *Sprzedaż*

stacjonarna, Obsługa klienta czy Produkcja – jak i pewne zagadnienie związane z tymi działaniami – np. *Jakość obsługi klientów, Satysfakcja klientów czy Uciążliwość produkcji dla otoczenia*.

Wymiar definiowany jest jako pewien punkt widzenia na określony fakt, tworzy zatem kontekst opisu analizowanego obszaru rzeczywistości. Wymiary należy identyfikować odrębnie dla każdego faktu, pamiętając o tym, że muszą one być zgodne ze sposobem patrzenia na ten fakt przez przyszłego użytkownika aplikacji analitycznej. Przykładami wymiarów dla faktu *Obsługa klienta* mogą być: *Czas, Kanał komunikacji, Pracownik* czy też *Powód komunikacji*. Z kolei dla faktu *Satysfakcja klienta* można wskazać takie wymiary, jak *Czas, Klient* czy *Kryterium satysfakcji*. W praktyce często okazuje się, że ten sam wymiar może opisywać wiele faktów, co spowoduje w etapie szóstym uproszczenie modelu danych poprzez zmniejszenie liczby tabel zawierających dane charakterystyczne dla tych wymiarów.

Znajomość wymiarów prowadzi do określenia szeroko pojętego kontekstu analizy faktu. Jednak tak ogólne stwierdzenia, jak np. *Czas* i *Lokalizacja klienta*, opisujące zachowania zakupowe klienta, czy też *Kanał komunikacji* lub *Klient*, charakterystyczne dla opisu obsługi klienta, nie pozwalają zrozumieć, co tak naprawdę one oznaczają. Wymiar *Czas* może być rozumiany jako rok, miesiąc lub konkretny dzień miesiąca czy tygodnia, wymiar *Klient* – jako konkretna osoba prawna (firma) lub fizyczna (klient indywidualny) bądź też grupa takich osób. Dlatego też konieczne jest zdefiniowanie tzw. atrybutu wymiaru, oznaczającego pewną jego szczegółową cechę charakterystyczną. Zazwyczaj wymiar opisany jest wieloma atrybutami, uwzględniającymi wieloznaczność danego pojęcia. Na przykład dla wymiaru *Czas* można zdefiniować takie atrybuty, jak *Rok, Kwartał, Miesiąc, Dzień* czy *Pora dnia*, a dla wymiaru *Klient* – *Rodzaj klienta, Nazwa klienta, Status, Kraj klienta* czy *Branża*.

Ostatnim z podstawowych pojęć, używanych w projektowaniu analitycznych baz danych, jest miara, która opisuje w sposób ilościowy dany fakt. Każda miara zawsze jest liczbą, a jej właściwy wybór musi zapewnić możliwość oceny stopnia osiągnięcia celu, z którym powiązany jest dany fakt. Biorąc pod uwagę przykładowy fakt, jakim jest *Satysfakcja klientów*, można zdefiniować takie miary, jak: *Liczba klientów dokonujących zakupu, Liczba klientów, którzy wystawili opinię po zakupie, Liczba klientów, którzy wystawili pozytywną opinię po zakupie* czy *Liczba klientów deklarujących ponowny zakup w sklepie*.

Szczegółowa charakterystyka ww. pojęć, ich przykłady odnoszące się do wybranej rzeczywistości oraz praktyczne wskazówki dotyczące sposobu ich wyboru zostaną przedstawione w dalszej części tego rozdziału.

Podsumowując dotychczasowe rozważania dotyczące idei modelowania wielowymiarowego, należy stwierdzić, że znajomość opisanych powyżej podstawowych pojęć jest istotnym krokiem ułatwiającym realizację niemal wszystkich jego etapów. Ważne też, aby w procesie identyfikacji i definicji opisanych tutaj podstawowych pojęć w odniesieniu do wybranej rzeczywistości uczestniczyły osoby, które będą reprezentować przyszłych użytkowników biznesowych. Jest to warunek konieczny poprawności tworzonego wielowymiarowego modelu analitycznej bazy danych.

4.2. Identyfikacja obszaru analizy

Projektowanie analitycznej bazy danych należy rozpocząć od określenia obszary będące przedmiotem analizy. Poprawne wskazanie tych obszarów wymaga, aby uprzednio określić cel budowy data mart czy hurtowni danych. W niniejszej publikacji przyjęto założenie, iż analityczna baza danych jest tworzona dla firmy biznesowej, której działalność polega na sprzedaży produktów będących rowerami, akcesoriami rowerowymi oraz odzieżą sportową. Biorąc to pod uwagę, takim celem będzie na przykład zwiększenie rentowności sprzedaży, optymalizacja procesów logistycznych, zwiększenie odsetka zadowolonych klientów, zwiększenie skuteczności kampanii marketingowych czy zapewnienie pełnej dostępności sklepu internetowego. Należy jednak pamiętać, aby cel zdefiniować bardzo precyzyjnie, zgodnie z zasadą SMART [9, 31], co zapewni w piątym etapie procesu modelowania wielowymiarowego poprawne określenie miar dla poszczególnych faktów. Zachowując ww. zasadę, poniżej przedstawiono uszczegółowione definicje uprzednio wspomnianych celów:

- zwiększenie rentowności sprzedaży o 25% w ciągu najbliższych trzech lat;
- optymalizacja procesów logistycznych poprzez redukcję o 20% wykorzystywanych zasobów w ciągu trzech kolejnych lat;
- zwiększenie odsetka zadowolonych klientów, którzy dokonali zakupu, do poziomu 96% w ciągu najbliższych pięciu lat;

- zwiększenie skuteczności kampanii marketingowych do poziomu 75% w okresie trzech kolejnych lat;
- zapewnienie dostępności dla klientów internetowej platformy sprzedażowej do poziomu 99,96% w okresie 12 miesięcy.

Istotnym działaniem na etapie określenia obszarów analizy jest dekompozycja celów głównych do postaci celów szczegółowych. Tego rodzaju podejście znacznie ułatwi realizację drugiego etapu modelowania wielowymiarowego, tj. identyfikację faktów. Wykorzystując definicję uprzednio określonych celów ogólnych, można zdefiniować następujące przykładowe cele szczegółowe dla każdego z nich (zachowując odpowiednio ten sam horyzont czasu):

- zwiększenie rentowności sprzedaży (cel ogólny):
 - zwiększenie wartości sprzedaży online o 30%,
 - zwiększenie wartości sprzedaży stacjonarnej o 20%,
 - obniżenie kosztów sprzedaży online o 12%,
 - obniżenie kosztów sprzedaży stacjonarnej o 20%,
 - redukcja kosztów operacyjnych o 15%,
 - zmniejszenie wskaźnika porzuconych koszyków w zakupach internetowych do poziomu 20%;
- optymalizacja procesów logistycznych poprzez redukcję o 20% wykorzystywanych zasobów (cel ogólny):
 - skrócenie czasu magazynowania produktów do maksymalnie dwóch miesięcy (zwiększenie rotacji produktów),
 - zmniejszenie ryzyk błędów i uszkodzeń produktów w magazynowych procesach logistycznych do 10% ogółu operacji logistycznych,
 - redukcja kosztów magazynowania produktów o 30%,
 - skrócenie czasu realizacji zamówień do 24 godzin od momentu potwierdzenia zamówienia,
 - zapewnienie dostępności produktów w ciągu maksymalnie 48 godzin;
- zwiększenie odsetka zadowolonych klientów (cel ogólny):
 - zwiększenie odsetka zadowolonych klientów korzystających z mobilnych kanałów zakupowych do poziomu 98%,
 - zwiększenie odsetka zadowolonych klientów dokonujących zakupów stacjonarnych do poziomu 94%,
 - obniżenie odsetka transakcji, które związane są z reklamacją lub zwrotem zakupionych towarów, do 5% w ciągu kolejnych pięciu lat,

- zwiększenie odsetka zadowolonych klientów z kontaktu z firmą do poziomu 98%;
- zwiększenie skuteczności kampanii marketingowych (cel ogólny):
 - zwiększenie skuteczności kampanii online (SEM, e-mail, media społecznościowe, afiliacje, banery) do poziomu 80%,
 - zwiększenie skuteczności kampanii offline (reklama w radio, telewizji, prasie, ambientowa) do poziomu 40%,
 - zwiększenie wskaźnika responsywności klientów na kampanie do poziomu 45%,
 - zwiększenie wskaźnika konwersji w lejku sprzedażowym do poziomu 15%;
- zapewnienie dostępności dla klientów internetowej platformy sprzedażowej na poziomie 99,96%:
 - wyeliminowanie 99% zagrożeń związanych z atakami na stronę WWW i stronę sklepu internetowego,
 - redukcja błędów działania platformy wynikających z niepoprawnej implementacji jej funkcjonalności do poziomu 3%,
 - zapewnienie skuteczności składania zamówień internetowych na poziomie 99,75%,
 - zapewnienie pozytywnego doświadczenia użytkownika, związanego z używaniem platformy sprzedażowej, na poziomie 97%.

Tak szczegółowa identyfikacja obszarów analizy jest konieczna, gdyż jej rezultaty przesądzą o jakości wyników działań w kolejnych etapach modelowania wielowymiarowego. Błędy popełnione w tej fazie spowodują utworzenie w hurtowni danych struktur danych niespójnych z rzeczywistością. Będzie to skutkować brakami właściwych danych, uniemożliwiającymi skuteczne i efektywne wsparcie prawidłowych decyzji, do podjęcia których ma być wykorzystana projektowana hurtownia danych.

4.3. Definicja faktów

Konwersja celu głównego budowy hurtowni danych na zbiór celów szczegółowych umożliwia w kolejnym etapie modelowania stworzenie definicji faktów. Z każdym takim celem można skojarzyć jeden lub wiele faktów, w zależności

od tego, jak szczegółowo określono ten cel. Identyfikując fakt, należy pamiętać, aby zapewnić możliwość ilościowego (liczbowego) opisu analizowanego zagadnienia. Dla lepszego zrozumienia tej zależności w tabeli 4.1 przedstawiono przykładowe cele cząstkowe, skojarzone z nimi fakty i ich krótki opis.

Każdy z faktów opisuje określone zagadnienie lub zjawisko, pozostające w istotnym związku z uprzednio zdefiniowanym celem. Analiza tych faktów umożliwi w przyszłości ocenę stopnia jego osiągnięcia. Dlatego tak ważne jest umiejętne identyfikowanie faktów, aby nie popełnić błędów polegających na uproszczeniu znaczenia analizowanych zjawisk. Typowe błędy na tym etapie polegają na posługiwaniu się pojęciami zaczerpniętymi z mowy potocznej – np. stwierdzenie, że analiza ma dotyczyć klientów, jest niepoprawne, ponieważ przedmiotem analizy nie są klienci sami w sobie. Poprawne jest natomiast sformułowanie, że analizowane będą zachowania klientów, ich responsywność na działania ze strony otoczenia, aktywność zakupowa, satysfakcja czy lojalność wobec firmy. Podobnie nie można analizować sklepu internetowego, jego funkcjonalności czy stron webowych – ale atrakcyjność jego oferty, poprawność realizacji funkcjonalności e-sklepu czy też widzialność stron internetowych.

Reasumując rozważania dotyczące identyfikacji faktów, należy podkreślić, że jest to jeden z kluczowych etapów działań w ramach modelowania wielowymiarowego, gdyż decyduje zarówno o wartości biznesowej tworzonej hurtowni danych, jak i poprawności jej struktur danych.

Każdy z faktów opisuje określone zagadnienie lub zjawisko, pozostające w istotnym związku z uprzednio zdefiniowanym celem. Analiza tych faktów umożliwi w przyszłości ocenę stopnia jego osiągnięcia. Dlatego tak ważne jest umiejętne identyfikowanie faktów, aby nie popełnić błędów polegających na uproszczeniu znaczenia analizowanych zjawisk. Typowe błędy na tym etapie polegają na posługiwaniu się pojęciami zaczerpniętymi z mowy potocznej – np. stwierdzenie, że analiza ma dotyczyć klientów, jest niepoprawne, ponieważ przedmiotem analizy nie są klienci sami w sobie. Poprawne jest natomiast sformułowanie, że analizowane będą zachowania klientów, ich responsywność na działania ze strony otoczenia, aktywność zakupowa, satysfakcja czy lojalność wobec firmy. Podobnie nie można analizować sklepu internetowego, jego funkcjonalności czy stron webowych – ale atrakcyjność jego oferty, poprawność realizacji funkcjonalności e-sklepu czy też widzialność stron internetowych.

Tab. 4.1. Wybrane fakty dla przykładowych celów cząstkowych

Cel cząstkowy	Fakt	Opis faktu
Zwiększenie wartości sprzedaży online	Sprzedaż online	Opisuje przychody uzyskane ze sprzedaży online w różnych kanałach sprzedażowych
Zwiększenie wartości sprzedaży stacjonarnej	Sprzedaż stacjonarna	Opisuje przychody uzyskane ze sprzedaży w sklepach stacjonarnych
Obniżenie kosztów sprzedaży stacjonarnej	Koszty sprzedaży stacjonarnej	Opisuje różnego rodzaju koszty generowane w różnych miejscach w związku ze sprzedażą stacjonarną
Zmniejszenie wskaźnika porzuconych koszyków internetowych	Porzucanie koszyków zakupowych	Opisuje działanie klientów polegające na porzuceniu koszyka zakupowego w sklepie internetowym
Skrócenie czasu magazynowania produktów	Rotacja produktów w magazynie	Opisuje ścieżkę logistyczną, jaką przechodzą produkty od momentu ich przyjęcia na magazyn do chwili wydania (wysyłki) do klienta
Zmniejszenie ryzyka błędów i uszkodzeń produktów w procesach logistycznych	Ryzyko błędów w procesach logistycznych	Opisuje błędy popełnione w procesach logistycznych realizowanych w magazynie i ich skutki
	Uszkodzenia produktów w procesach logistycznych	Opisuje uszkodzenia produktów na ich ścieżce logistycznej
Skrócenie czasu realizacji zamówień	Realizacja zamówień	Opisuje poszczególne etapy realizacji zamówień
Zapewnienie dostępności produktów	Dostępność produktów	Opisuje zagadnienie dostępności produktów w sklepie i uzupełniania stanów magazynowych
	Dostawy produktów	Opisuje proces składania i realizacji zamówień na produkty u ich dostawców (producentów), włącznie z ich dostawą do własnego magazynu
Zwiększenie odsetka zadowolonych klientów z zakupów mobilnych	Satysfakcja klientów z zakupów mobilnych	Opisuje poziom satysfakcji klientów z zakupów mobilnych oraz źródła tej satysfakcji
Zwiększenie odsetka zadowolonych klientów z kontaktu z firmą	Ogólna satysfakcja klientów	Opisuje zagadnienie satysfakcji klientów, której źródłem są wszelakie kontakty z firmą
	Zarządzanie relacjami z klientami	Opisuje tworzenie, utrzymanie i rozwój relacji z klientami
	Obsługa posprzedażna klientów	Opisuje zagadnienie obsługi klientów po zakupieniu przez nich produktów
Obniżenie odsetka transakcji reklamowanych lub związanych ze zwrotem towaru	Reklamacje produktów	Opisuje zagadnienie reklamowania zakupionych produktów przez klientów oraz powody takiego działania
	Zwroty produktów	Opisuje zagadnienie zwracania produktów przez kupujących i przyczyny zaistnienia zwrotów
Zwiększenie skuteczności kampanii marketingowej online	Skuteczność kampanii e-marketingowej	Opisuje poziom osiągnięcia celu kampanii prowadzonej w różnych kanałach online

cd. Tab. 4.1. Wybrane fakty dla przykładowych celów cząstkowych

Cel cząstkowy	Fakt	Opis faktu
Zwiększenie wskaźnika responsywności klientów na kampanie	Responsywność klientów w kampanii marketingowej	Opisuje zachowania klientów w odpowiedzi na kierowany do nich kontent marketingowy
Zapewnienie dostępności platformy internetowej	Dostępność sklepu internetowego	Opisuje możliwość logowania się do sklepu internetowego i skorzystania z jego funkcjonalności
	Ruch w witrynie WWW	Opisuje wizyty klientów w witrynie WWW
	Zagrożenia atakami sklepu internetowego	Opisuje występowanie różnych ataków na sklep internetowy i sposoby odpowiedzi na te zdarzenia
	Błędne działanie platformy	Opisuje występowanie błędów funkcjonalnych podczas korzystania z platformy internetowej
	Skuteczność złożenia zamówień online	Opisuje stopień realizacji procesu składania zamówień online

Źródło: opracowanie własne.

Reasumując rozważania dotyczące identyfikacji faktów, należy podkreślić, że jest to jeden z kluczowych etapów działań w ramach modelowania wielowymiarowego, gdyż decyduje zarówno o wartości biznesowej tworzonej hurtowni danych, jak i poprawności jej struktur danych.

4.4. Definicja szczegółowości faktów

Opisana w poprzednim rozdziale identyfikacja faktów jest rezultatem działań podejmowanych zarówno przez analityka systemowego, a więc osobę z obszaru IT, jak i przez reprezentanta klienta biznesowego, który potwierdzi prawidłowość wyboru faktów i ich spójność z celami budowy hurtowni danych. Jednak na potrzeby przyszłych działań deweloperskich, związanych z tworzeniem struktury tabel faktów, konieczne jest doprecyzowanie znaczenia każdego faktu i określenie, co dokładnie będzie rozumiane jako wiersz w takiej tabeli. W zależności od poziomu ogólności analizy, często także określanej mianem jej ziarnistości, rekord w tabeli faktu może zawierać zarówno dane bardzo szczegółowe, jak i powstałe w wyniku agregacji. W tabelach 4.2–4.20 przedstawiono przykłady różnych poziomów szczegółowości dla faktów uprzednio wymienionych w tabeli 4.1.

Tab. 4.2. Szczegółowość faktu *Sprzedaż online*

Szczegółowość	Opis faktu
Niska	Zbiór wielu zamówień internetowych
Średnia	Pojedyncze zamówienie internetowe
Bardzo duża	Pojedyncza pozycja zamówienia internetowego

Źródło: opracowanie własne.

Tab. 4.3. Szczegółowość faktu *Koszty sprzedaży stacjonarnej*

Szczegółowość	Opis faktu
Niska	Całkowite koszty sprzedaży stacjonarnej
Średnia	Grupa rodzajowa kosztów sprzedaży stacjonarnej
Bardzo duża	Pojedynczy określony koszt sprzedaży stacjonarnej

Źródło: opracowanie własne.

Tab. 4.4. Szczegółowość faktu *Porzucanie koszyków zakupowych*

Szczegółowość	Opis faktu
Niska	Wiele koszyków zakupowych
Średnia	Pojedynczy koszyk zakupowy traktowany jako całość
Bardzo duża	Pojedynczy produkt z koszyka zakupowego

Źródło: opracowanie własne.

Tab. 4.5. Szczegółowość faktu *Rotacja produktów w magazynie*

Szczegółowość	Opis faktu
Niska	Wiele ścieżek logistycznych wielu produktów
Średnia	Wiele ścieżek logistycznych pojedynczego produktu
Duża	Pojedyncza ścieżka logistyczna danego produktu
Bardzo duża	Pojedynczy etap ścieżki logistycznej danego produktu

Źródło: opracowanie własne.

Tab. 4.6. Szczegółowość faktu *Ryzyko błędów w procesach logistycznych*

Szczegółowość	Opis faktu
Bardzo niska	Wiele błędów w wielu procesach logistycznych
Średnia	Wiele błędów w pojedynczym procesie logistycznym
Duża	Pojedynczy błąd w wielu procesach logistycznych
Bardzo duża	Pojedynczy błąd w jednym procesie logistycznym

Źródło: opracowanie własne.

Tab. 4.7. Szczegółowość dla faktu *Realizacja zamówienia*

Szczegółowość	Opis faktu
Niska	Realizacja wielu zamówień
Średnia	Realizacja pojedynczego zamówienia
Duża	Pojedynczy etap realizacji zamówienia

Źródło: opracowanie własne.

Tab. 4.8. Szczegółowość dla faktu *Dostawy produktów*

Szczegółowość	Opis faktu
Niska	Wiele dostaw wielu produktów
Średnia	Wiele dostaw tego samego produktu
Duża	Pojedyncza dostawa konkretnego produktu traktowana jako całość
Bardzo duża	Pojedyncza operacja realizowana w ramach pojedynczej dostawy konkretnego produktu

Źródło: opracowanie własne.

Tab. 4.9. Szczegółowość faktu *Satysfakcja klientów z zakupów mobilnych*

Szczegółowość	Opis faktu
Niska	Wyrażenie opinii przez grupę klientów
Średnia	Wyrażenie pojedynczej opinii przez danego klienta traktowane jako całość
Duża	Wyrażenie opinii dotyczącej jednego kryterium satysfakcji przez danego klienta

Źródło: opracowanie własne.

Tab. 4.10. Szczegółowość faktu *Zarządzanie relacjami z klientami*

Szczegółowość	Opis faktu
Niska	Wiele kontaktów z wieloma klientami
Średnia	Wiele kontaktów z tym samym klientem
Duża	Pojedynczy kontakt z tym samym klientem

Źródło: opracowanie własne.

Tab. 4.11. Szczegółowość faktu *Obsługa posprzedażna klientów*

Szczegółowość	Opis faktu
Niska	Wiele obsług posprzedażnych wielu klientów
Średnia	Wiele obsług posprzedażnych jednego klienta
Duża	Pojedyncza obsługa posprzedażna jednego klienta
Bardzo duża	Pojedyncza czynność w ramach jednej obsługi posprzedażnej jednego klienta

Źródło: opracowanie własne.

Tab. 4.12. Szczegółowość faktu *Reklamacje produktów*

Szczegółowość	Opis faktu
Niska	Wiele reklamacji wielu produktów jako całość
Średnia	Wiele reklamacji pojedynczego produktu jako całość
Duża	Pojedyncza reklamacja pojedynczego produktu
Bardzo duża	Pojedynczy powód jednej reklamacji danego produktu

Źródło: opracowanie własne.

Tab. 4.13. Szczegółowość faktu *Zwrot produktów*

Szczegółowość	Opis faktu
Niska	Wiele zwrotów różnych produktów jako całość
Średnia	Wiele zwrotów tego samego produktu
Duża	Pojedynczy zwrot produktu jako całość
Bardzo duża	Pojedynczy powód jednego zwrotu danego produktu

Źródło: opracowanie własne.

Tab. 4.14. Szczegółowość faktu *Skuteczność kampanii e-marketingowej*

Szczegółowość	Opis faktu
Niska	Kampania e-marketingowa jako całość w wielu kanałach
Średnia	Kampania e-marketingowa jako całość w danym kanale
Duża	Pojedyncze działanie określonego rodzaju w ramach kampanii e-marketingowej w danym kanale

Źródło: opracowanie własne.

Tab. 4.15. Szczegółowość faktu *Responsywność klientów w kampanii marketingowej*

Szczegółowość	Opis faktu
Bardzo niska	Reakcje wielu klientów na wiele kampanii
Niska	Reakcje wielu klientów na pojedynczą kampanię
Średnia	Reakcje wielu klientów na pojedyncze działanie w ramach pojedynczej kampanii
Duża	Reakcja konkretnego klienta na pojedynczą kampanię
Bardzo duża	Reakcja konkretnego klienta na pojedyncze działanie w ramach danej kampanii

Źródło: opracowanie własne.

Tab. 4.16. Szczegółowość faktu *Dostępność sklepu internetowego*

Szczegółowość	Opis faktu
Niska	Skorzystanie z wielu różnych funkcjonalności w sklepie internetowym
Średnia	Wiele wizyt w sklepie internetowym jako całość
Duża	Pojedyncza wizyta w sklepie internetowym jako całość
Bardzo duża	Pojedyncza operacja w sklepie internetowym

Źródło: opracowanie własne.

Tab. 4.17. Szczegółowość faktu *Ruch w witrynie WWW*

Szczegółowość	Opis faktu
Niska	Wiele wizyt w witrynie WWW jako całość
Średnia	Pojedyncza wizyta w witrynie WWW jako całość
Duża	Wiele wizyt na konkretnej stronie WWW jako całość
Bardzo duża	Pojedyncza wizyta na konkretnej stronie WWW

Źródło: opracowanie własne.

Tab. 4.18. Szczegółowość faktu *Zagrożenia atakami sklepu internetowego*

Szczegółowość	Opis faktu
Niska	Wiele ataków różnego rodzaju na sklep internetowy
Średnia	Grupa ataków określonego typu na sklep internetowy
Duża	Pojedynczy atak danego rodzaju na sklep internetowy
Bardzo duża	Pojedynczy atak danego rodzaju na dany zasób sklepu internetowego

Źródło: opracowanie własne.

Tab. 4.19. Szczegółowość dla faktu *Błędne działanie platformy*

Szczegółowość	Opis faktu
Niska	Wystąpienie wszystkich błędów podczas korzystania z platformy
Średnia	Wystąpienie błędów określonego typu podczas korzystania z platformy
Duża	Wystąpienie pojedynczego błędu określonego rodzaju podczas korzystania z platformy

Źródło: opracowanie własne.

Tab. 4.20. Szczegółowość dla faktu *Skuteczność złożenia zamówień online*

Szczegółowość	Opis faktu
Niska	Złożenie wielu zamówień widziane jako całość
Średnia	Złożenie pojedynczego zamówienia jako całość
Duża	Realizacja pojedynczego etapu składania zamówienia

Źródło: opracowanie własne.

Zaproponowane w tabelach 4.2–4.20 przykładowe szczególowości dla tego samego faktu stanowią odzwierciedlenie różnych potrzeb informacyjnych końcowego użytkownika biznesowego i wpływają na rodzaj danych, jakie będą przechowywane w analitycznej bazie danych. Jednak deweloper SQL, tworzący tabelę faktu, nie jest osobą upoważnioną do decydowania o szczególowości faktu. Decyzja taka musi być podjęta przez klienta biznesowego, który określi, jak dokładnie zamierza opisywać i analizować rzeczywistość. Przy definiowaniu faktu o dużym stopniu szczególowości – jak np. pojedyncza wizyta na określonej stronie WWW – należy mieć na uwadze, iż takie rozwiązanie spowoduje wydłużony czas agregacji danych i obliczeń analitycznych wykonywanych na wyższym poziomie ogólności, wynikającym z przyjętych wymiarów. Zaletą takiego podejścia jest jednak możliwość uzyskania dokładnych danych, które mogą wyjaśniać określone zachowanie odwiedzającego podczas jego aktywności w każdym miejscu witryny internetowej. Z kolei określenie faktu na zbyt dużym poziomie ogólności – np. wizyta w sklepie internetowym rozumiana jako całość – uniemożliwi poznanie wielu szczegółów tego zdarzenia, za to informacje o aspektach uwzględnionych w tym opisie trafią do użytkownika w znacznie krótszym czasie niż w przypadku dużej ziarnistości faktu. W praktyce znalezienie optymalnego rozwiązania, które zapewni oczekiwany stopień szczególowości faktu i akceptowany czas wykonania zapytań, wymaga odpowiedniego modelowania danych na różnych poziomach ich agregacji.

4.5. Identyfikacja wymiarów i ich atrybutów

Kolejnym etapem modelowania wielowymiarowego jest identyfikacja wymiarów dla poszczególnych faktów. Działanie to ma na celu określenie tych aspektów analizy faktu, które są istotne z punktu widzenia możliwości osiągnięcia przyjętych celów budowy hurtowni danych. Dlatego też rodzaj wymiaru będzie zależał od dwóch czynników. Pierwszym z nich jest znaczenie faktu, do którego opisu zostanie wykorzystany dany wymiar, drugim zaś – jego szczegółowość. Wymiar należy postrzegać jako kontekst analizy zagadnienia, pozwalający dokładnie zrozumieć fakt. Dane numeryczne, charakterystyczne dla faktu, nie mają żadnej użyteczności i nie mogą być poprawnie interpretowane, jeśli nie określi się sposobu ich postrzegania.

Identyfikacja wymiarów jest drugim fundamentalnym działaniem w modelowaniu wielowymiarowym, dlatego tak ważna jest poprawność jego realizacji. W celu ułatwienia wykonania tego zadania należy zadać sobie kilka przykładowych pytań, które brzmią następująco:

- Co?
- Kto?
- Gdzie?
- Kiedy?
- W jaki sposób?

Odpowiedzi na te pytania będą zależeć od faktu, w kontekście którego je zadano. Stosując powyższe podejście, dla przykładowego faktu *Ruch w witrynie WWW*, można zdefiniować następujące wymiary:

- *Strona WWW* (jaką stronę WWW odwiedza użytkownik?);
- *Odwiedzający* (kto odwiedza stronę WWW?);
- *Czas* (o której godzinie i minucie odwiedzono stronę WWW?);
- *Źródło ruchu* (w jaki sposób odwiedzający trafił na stronę WWW?);
- *Platforma odwiedzin* (z jakiej przeglądarki i urządzenia korzystał odwiedzający?).

W tabelach 4.21–4.30 przedstawiono propozycje wymiarów dla wybranych faktów opisanych uprzednio w tabelach 4.2–4.20.

Tab. 4.21. Wybrane wymiary dla faktu *Sprzedaż online*

Szczegółowość faktu	Wymiary
Zbiór wielu zamówień internetowych	Czas, Kanał sprzedaży, Kategoria produktu, Grupa klientów, Miejsce złożenia zamówienia
Pojedyncze zamówienie internetowe	Czas, Kanał sprzedaży, Klient, Miejsce złożenia zamówienia, Metoda dostawy, Miejsce dostawy, Metoda płatności
Pojedyncza pozycja zamówienia internetowego	Czas, Kanał sprzedaży, Produkt, Klient, Miejsce złożenia zamówienia, Metoda dostawy, Miejsce dostawy, Metoda płatności, Kampania marketingowa

Źródło: opracowanie własne.

Tab. 4.22. Wybrane wymiary dla faktu *Koszty sprzedaży stacjonarnej*

Szczegółowość faktu	Wymiary
Całkowite koszty sprzedaży stacjonarnej	Czas, Miejsce powstania kosztu
Grupa rodzajowa kosztów sprzedaży stacjonarnej	Czas, Miejsce powstania kosztu, Rodzaj kosztu
Pojedynczy określony koszt sprzedaży stacjonarnej	Czas, Miejsce powstania kosztu, Rodzaj kosztu, Źródło kosztu

Źródło: opracowanie własne.

Tab. 4.23. Wybrane wymiary dla faktu *Porzucanie koszyków zakupowych*

Szczegółowość faktu	Wymiary
Wiele koszyków zakupowych	Czas, Kanał sprzedaży, Grupa klientów, Etap zamówienia
Pojedynczy koszyk zakupowy traktowany jako całość	Czas, Kanał sprzedaży, Klient, Etap zamówienia
Pojedynczy produkt z koszyka zakupowego	Czas, Kanał sprzedaży, Klient, Produkt, Etap zamówienia

Źródło: opracowanie własne.

Tab. 4.24. Wybrane wymiary dla faktu *Ryzyko błędów w procesach logistycznych*

Szczegółowość faktu	Wymiary
Wiele błędów w wielu procesach logistycznych	Czas, Rodzaj błędu, Miejsce powstania błędu
Wiele błędów w pojedynczym procesie logistycznym	Czas, Rodzaj błędu, Miejsce powstania błędu, Proces logistyczny
Pojedynczy błąd w wielu procesach logistycznych	Czas, Błąd, Miejsce powstania błędu
Pojedynczy błąd w jednym procesie logistycznym	Czas, Błąd, Miejsce powstania błędu, Proces logistyczny

Źródło: opracowanie własne.

Tab. 4.25. Wybrane wymiary dla faktu *Realizacja zamówień*

Szczegółowość faktu	Wymiary
Realizacja wielu zamówień	Czas, Miejsce realizacji
Realizacja pojedynczego zamówienia	Czas, Zamówienie, Miejsce realizacji
Pojedynczy etap realizacji zamówienia	Czas, Zamówienie, Etap realizacji, Miejsce realizacji

Źródło: opracowanie własne.

Tab. 4.26. Wybrane wymiary dla faktu *Dostawy produktów*

Szczegółowość faktu	Wymiary
Wiele dostaw wielu produktów	Czas, Miejsce dostawy, Sposób dostawy
Wiele dostaw tego samego produktu	Czas, Miejsce dostawy, Sposób dostawy, Produkt
Pojedyncza dostawa konkretnego produktu traktowana jako całość	Czas, Miejsce dostawy, Sposób dostawy, Produkt
Pojedyncza operacja realizowana w ramach pojedynczej dostawy konkretnego produktu	Czas, Miejsce dostawy, Sposób dostawy, Produkt, Etap dostawy

Źródło: opracowanie własne.

Tab. 4.27. Wybrane wymiary dla faktu *Satysfakcja klientów z zakupów mobilnych*

Szczegółowość faktu	Wymiary
Wyrażenie opinii przez grupę klientów	Czas, Grupa klientów, Kanał sprzedaży, Kategoria kryterium satysfakcji
Wyrażenie pojedynczej opinii przez konkretnego klienta traktowane jako całość	Czas, Klient, Kanał sprzedaży, Kategoria kryterium satysfakcji
Wyrażenie opinii dotyczącej jednego kryterium satysfakcji przez pojedynczego klienta	Czas, Klient, Kanał sprzedaży, Kryterium satysfakcji

Źródło: opracowanie własne.

Tab. 4.28. Wybrane wymiary dla faktu *Zwrot produktów*

Szczegółowość faktu	Wymiary
Wiele zwrotów różnych produktów jako całość	Czas, Miejsce zwrotu, Kategoria produktowa, Przyczyna zwrotu
Wiele zwrotów tego samego produktu	Czas, Miejsce zwrotu, Produkt, Przyczyna zwrotu
Pojedynczy zwrot produktu jako całość	Czas, Miejsce zwrotu, Produkt, Przyczyna zwrotu
Pojedynczy powód jednego zwrotu danego produktu	Czas, Miejsce zwrotu, Produkt, Przyczyna zwrotu

Źródło: opracowanie własne.

Tab. 4.29. Wybrane wymiary dla faktu *Zagrożenia atakami sklepu internetowego*

Szczegółowość faktu	Wymiary
Wiele ataków różnego rodzaju na sklep internetowy	Czas, Domena, Źródło ataku
Grupa ataków określonego typu na sklep internetowy	Czas, Domena, Źródło ataku, Typ ataku
Pojedynczy atak danego rodzaju na sklep internetowy	Czas, Domena, Źródło ataku, Rodzaj ataku
Pojedynczy atak określonego rodzaju na dany zasób sklepu internetowego	Czas, Domena, Zasób sklepu, Źródło ataku, Rodzaj ataku

Źródło: opracowanie własne.

Tab. 4.30. Wybrane wymiary dla faktu *Błędne działanie platformy*

Szczegółowość faktu	Wymiary
Wystąpienie wszystkich błędów podczas korzystania z platformy	Czas, Domena, Miejsce wystąpienia błędu
Wystąpienie błędów określonego typu podczas korzystania z platformy	Czas, Domena, Typ błędu, Miejsce wystąpienia błędu
Wystąpienie pojedynczego błędu określonego rodzaju podczas korzystania z platformy	Czas, Domena, Rodzaj błędu, Miejsce wystąpienia błędu

Źródło: opracowanie własne.

Analizując treści przedstawione w tabelach 4.21–4.30, można zauważyć bardzo istotną prawidłowość polegającą na obecności wymiaru czasu w opisie każdego faktu. Ponadto ten sam wymiar może być wykorzystany do opisu różnych faktów, a w wielu przypadkach, wraz z rosnącą szczegółowością faktu zwiększa się liczba powiązanych z nim wymiarów.

Identyfikując wymiary dla danego faktu, należy także wziąć pod uwagę funkcjonalności przyszłych aplikacji analitycznych związane z możliwością wykonania takich operacji, jak filtrowanie, grupowanie, rozwijanie i zwijanie danych. Ponieważ są one odzwierciedleniem informacyjnych potrzeb użytkownika końcowego, konieczne jest bardzo precyzyjne definiowanie znaczenia poszczególnych wymiarów, gdyż brak takiej staranności opisu może w przyszłości, podczas tworzenia modelu danych, doprowadzić do niejednoznaczności w interpretacji tabel w bazie danych.

Znajomość wszystkich wymiarów powiązanych z danym faktem umożliwia przejście do kolejnego etapu modelowania wielowymiarowego, jakim jest identyfikacja ich atrybutów, czyli dokładnie charakteryzujących dany wymiar

cech. Jeśli wziąć pod uwagę wymiar czasu, który został użyty do opisu wszystkich faktów w tabelach 4.21–4.30, warto zadać sobie następujące pytanie: Czy dla każdego przypadku dla tego wymiaru zostaną zastosowane te same cechy? Zbiór cech, opisujących czas, jest bardzo bogaty. Czas może być opisany rokiem kalendarzowym, kwartałem, miesiącem, tygodniem, dniem, porą dnia, godziną, a nawet minutą. Są to typowe, naturalne atrybuty tego wymiaru. Ale w tym przypadku mogą także wystąpić inne cechy, których znaczenie wynika z informacyjnych potrzeb użytkownika końcowego, takie jak rok fiskalny, kwartał czy miesiąc fiskalny.

Poprawny dobór atrybutów wymiaru musi uwzględniać szczegółowość, z jaką dany wymiar ma opisywać powiązany z nim fakt. Oznacza to, że fakty opisujące zagadnienie z większym stopniem ogólności kojarzone są bardzo często z wymiarami, których atrybuty charakteryzują się również mniejszą szczegółowością. Dla przykładu, biorąc pod uwagę fakt *Sprzedaż online*, dla którego ziarnistość opisu dotyczy wszystkich zamówień złożonych w każdym kanale sprzedażowym, atrybutami wymiaru *Czas* mogą być *Rok kalendarzowy*, *Kwartał* i *Miesiąc*. Jeśli ten sam fakt będzie interpretowany jako pojedyncze zamówienie, wówczas wśród poprawnych i użytecznych w analizie atrybutów powinny znaleźć się jeszcze *Tydzień*, *Dzień*, *Pora dnia*, a nawet *Godzina*.

Właściwy wybór atrybutów wymiaru jest tylko jednym z warunków koniecznych do spełnienia, aby zapewnić pełną użyteczność rezultatów przyszłych analiz. Drugim warunkiem jest poprawne sprecyzowanie znaczenia każdego atrybutu, czyli nadanie mu jednoznacznej interpretacji, która umożliwi właściwy opis i zrozumienie otrzymanych wyników. Przykładem niejednoznacznego opisu atrybutu wymiaru *Czas* jest *Miesiąc*. Cecha ta może oznaczać numer miesiąca w roku kalendarzowym, numer miesiąca w kwartale, pełną lub skróconą nazwę miesiąca. Podobna sytuacja występuje dla atrybutu *Dzień*, który może być interpretowany jako numer dnia w roku, numer dnia w kwartale, numer dnia w miesiącu, numer dnia w tygodniu czy nazwa dnia tygodnia. W takim przypadku dobrą praktyką jest precyzyjne nazywanie każdego atrybutu, gdyż takie podejście pozwoli uniknąć w przyszłości, podczas modelowania wymiarów, nieporozumień związanych ze strukturą tabel wymiarów oraz zapewni poprawność definiowania szczegółowych transformacji danych w procesach ETL. Inną istotną korzyścią takiego podejścia jest zwiększenie efektywności zapytań w bazie danych oraz narzędziach analitycznych,

gdyż dokładna identyfikacja i znajomość znaczenia atrybutów wyeliminują konieczność przekształcania czy wzbogacania danych do postaci wymaganej w przyszłej analizie.

Przykładowe zestawy atrybutów dla różnych wymiarów faktu *Sprzedaż online*, rozumianego jako pojedyncza pozycja zamówienia, przedstawiono w tabeli 4.31. Każdy z atrybutów nazwano w sposób pozwalający na jednoznaczną interpretację jego znaczenia.

Identyfikacja atrybutów wymiaru jest jednym z kluczowych czynników decydujących o użyteczności analitycznej bazy danych w procesach analitycznych. Zadanie to należy realizować we współpracy z biznesowym użytkownikiem końcowym, który oceni adekwatność atrybutów do przyjętego kontekstu analizy, poprawność ich doboru do wymaganego poziomu szczegółowości opisu wymiaru i faktu oraz możliwość ich wykorzystania w wizualnej analizie danych, wykonywanej z użyciem narzędzi business intelligence.

Tab. 4.31. Przykładowe atrybuty wymiarów dla faktu *Sprzedaż online*

Wymiar	Atrybuty wymiaru (przykłady)
Czas	Kod czasu Pełna data i czas Numer roku kalendarzowego Numer kwartału roku kalendarzowego Nazwa kwartału roku kalendarzowego Numer miesiąca roku kalendarzowego Pełna nazwa miesiąca Skrót nazwy miesiąca Numer dnia miesiąca Numer dnia tygodnia Nazwa dnia tygodnia Typ dnia Nazwa pory dnia Godzina 24H Numer roku fiskalnego Nazwa kwartału roku fiskalnego Numer miesiąca roku fiskalnego
Kanał sprzedaży	Kod kanału sprzedaży Nazwa typu kanału sprzedaży Kod typu kanału sprzedaży Nazwa kanału sprzedaży

cd. Tab. 4.31. Przykładowe atrybuty wymiarów dla faktu *Sprzedaż online*

Wymiar	Atrybuty wymiaru (przykłady)
Produkt	Kod produktu Nazwa produktu Nazwa podkategorii produktowej Nazwa kategorii produktowej Nazwa producenta
Klient	Kod klienta Nazwa klienta Typ klienta Nazwa miasta Nazwa regionu kraju Kod regionu kraju Nazwa kraju Kod kraju
Miejsce złożenia zamówienia	Kod miejsca złożenia zamówienia Nazwa dzielnicy miasta Nazwa miasta Nazwa regionu kraju Kod regionu kraju Nazwa kraju Kod kraju
Metoda dostawy	Kod metody dostawy Nazwa typu metody dostawy Kod typu metody dostawy Nazwa metody dostawy
Miejsce dostawy	Kod miejsca dostawy Nazwa miejsca Kod pocztowy miejsca Nazwa dzielnicy miasta Nazwa miasta Nazwa regionu kraju Kod regionu kraju Nazwa kraju Kod kraju
Metoda płatności	Kod metody płatności Nazwa typu metody płatności Kod typu metody płatności Nazwa metody płatności
Kampania marketingowa	Kod kampanii Nazwa typu kampanii Nazwa kanału kampanii Nazwa kampanii Dzień rozpoczęcia kampanii Dzień zakończenia kampanii

Źródło: opracowanie własne.

4.6. Definicja miar

Wyrazem kwantyfikowalności faktu, czyli jego ilościowego opisu, jest miara, której wartość umożliwia ocenę stopnia osiągnięcia celu skojarzonego z danym faktem. Zazwyczaj dla pojedynczego faktu określa się wiele miar, które tworzą pełny obraz analizowanej rzeczywistości. Źródłem danych dla miary jest pewien pomiar, co oznacza, że jej wartość ma charakter natywny. Przykładami takiej miary są: liczba odsłon strony WWW, liczba sztuk produktu w zamówieniu, cena produktu czy też koszt dostawy zamówienia online. Jednak niektóre wielkości liczbowe opisujące fakt mogą mieć naturę pochodną, czyli nie są bezpośrednio zmierzone, lecz obliczone według przyjętego wzoru (wyrażenia) – np. koszt jednostkowy sprzedaży produktu, zyskowność sklepu stacjonarnego, wartość zamówienia czy porzuconego koszyka. Te wielkości również określa się mianem miar, jakkolwiek niektóre z nich należałoby zaliczać już do wskaźników ze względu na źródło pochodzenia, tj. wynik obliczenia, a nie pomiar. Dla faktów o dużej ogólności miary mogą stać się wskaźnikami o kluczowym znaczeniu dla użytkownika końcowego. W tym przypadku wartość KPI jest rezultatem silnej agregacji danych, a następnie wykonania obliczeń na tak stworzonych agregatach. Wybór właściwych miar będzie zatem uzależniony od tego, z jaką szczegółowością jest opisany fakt, którego one dotyczą.

Biorąc pod uwagę możliwości wykonywania operacji analitycznych – tj. sumowania, zliczania, wyznaczenia średniej, wartości maksymalnej czy minimalnej – z użyciem miar i otrzymania wyników, którym można nadać poprawne znaczenie, wyróżnia się następujące rodzaje miar:

- addytywne;
- semiaddytywne;
- nieaddytywne.

Miary addytywne charakteryzują się następującymi cechami:

- są najbardziej użytecznym sposobem analizy faktu, zapewniającym wysoką elastyczność definiowania wyrażen matematycznych w narzędziach analitycznych;
- zapewniają możliwość ich użycia jako parametru funkcji analitycznej, tworząc nową poprawną wartość opisującą fakt;
- mogą być stosowane do opisu faktu z użyciem zarówno dowolnego wymiaru, jak i dowolnego poziomu hierarchii wymiaru.

Przykładami miar addytywnych są: przychód ze sprzedaży, koszt realizacji zamówienia, liczba sztuk produktu w zamówieniu, kwota VAT czy liczba zamówień.

Nieco inny charakter mają miary semiaddytywne, których wartości mogą być użyte w funkcjach analitycznych tylko dla niektórych wymiarów czy też poziomów ich hierarchii, tworząc w rezultacie wyniki o pewnym znaczeniu. Przykładem takiej miary jest *Stan magazynowy produktu*, który może być sumowany w dowolnym wymiarze (*Magazyn, Produkt, Opakowanie*) z wyjątkiem wymiaru *Czas*.

Istnieją także miary, określane mianem nieaddytywnych, które nie mogą być przetwarzane z użyciem funkcji analitycznych w żadnym wymiarze, gdyż ich rezultat nie jest interpretowalny. Definiują one wszelkiego rodzaju wskaźniki, których wartości są rezultatem obliczeń wykonanych na różnych agregatach danych. Jako przykład miar nieaddytywnych dla faktu sprzedażowego można podać *Stawka VAT, Średnia wartość zamówienia czy Przychód narastająco*. Dla faktu *Satysfakcja klientów* taką miarą nieaddytywną będzie *Odsetek zadowolonych klientów czy Średnia ocena*.

Definiując miary, warto wziąć pod uwagę jeszcze jeden aspekt związany z wydajnością przyszłej aplikacji analitycznej, w której odbywać się będzie wizualizacja danych. Jeżeli na tym etapie zostaną wybrane wyłącznie miary natywne, czyli niewyliczane na podstawie żadnej formuły, a pominie się miary kalkulowane, wówczas ich wartości będą wyznaczane w warstwie narzędzia analitycznego. Takie podejście spowoduje znaczne wydłużenie czasu generowania różnych elementów graficznych w tzw. dashboardach, co negatywnie wpłynie na doświadczenie użytkownika korzystającego z tego rodzaju rozwiązania. Dlatego też tak ważne jest, aby już na etapie typowania miar wskazać te, których wartości będą gromadzone w tabeli faktu w analitycznej bazie danych i wykorzystywane w przyszłych wizualizacjach.

W tabelach 4.32–4.43 zamieszczono przykłady miar dla wybranych faktów wymienionych w tabelach 4.2–4.20. Tak rozbudowane zestawienie ilościowego opisu poszczególnych faktów będzie stanowiło istotną pomoc w identyfikowaniu miar dla wielu różnych rzeczywistości podlegających analizie i ułatwi Czytelnikowi samodzielne dobieranie tych miar adekwatnie do potrzeb informacyjnych.

Tab. 4.32. Przykłady miar dla faktu *Sprzedaż online*

Szczegółowość faktu	Miary (przykłady)
Zbiór wielu zamówień internetowych	Liczba zamówień, wartość zamówień, średnia wartość zamówienia, liczba zamawiających klientów, koszty dostaw, wartość rabatów, średnia wartość rabatu dla zamówienia, przychody narastająco, całkowita zyskowność, średnia zyskowność dla jednego zamówienia
Pojedyncze zamówienie internetowe	Wartość zamówienia, koszt dostawy, koszt ubezpieczenia dostawy, liczba pozycji w zamówieniu, rabat kwotowy dla zamówienia, kwota VAT, zyskowność na zamówieniu
Pojedyncza pozycja zamówienia internetowego	Cena katalogowa produktu, cena transakcyjna, liczba sztuk produktu, wartość pozycji zamówienia, rabat kwotowy dla pozycji zamówienia, rabat procentowy dla pozycji zamówienia, stawka VAT dla produktu, kwota VAT dla produktu, wartość produktu w odniesieniu do wartości całego zamówienia

Źródło: opracowanie własne.

Tab. 4.33. Przykłady miar dla faktu *Porzucanie koszyków zakupowych*

Szczegółowość faktu	Miary (przykłady)
Wiele koszyków zakupowych	Liczba porzuconych koszyków, liczba wszystkich koszyków, wartość wszystkich porzuconych koszyków, średnia wartość porzuconego koszyka, odsetek porzuconych koszyków w odniesieniu do wszystkich koszyków, wskaźnik wartości porzuconych koszyków do wartości wszystkich koszyków
Pojedynczy koszyk zakupowy traktowany jako całość	Wartość porzuconego koszyka, liczba produktów w koszyku
Pojedynczy produkt z koszyka zakupowego	Cena regularna produktu, cena transakcyjna produktu, liczba sztuk produktu w koszyku, wartość produktu w koszyku, wartość produktu odniesiona do wartości koszyka

Źródło: opracowanie własne.

Tab. 4.34. Przykłady miar dla faktu *Satysfakcja klientów z zakupów mobilnych*

Szczegółowość faktu	Miary (przykłady)
Wyrażenie opinii przez grupę klientów	Liczba klientów, którzy dokonali zakupu, liczba klientów, którzy wyrazili opinię, liczba klientów zadowolonych, odsetek klientów zadowolonych, średnia ocena, najniższa ocena, najwyższa ocena
Wyrażenie pojedynczej opinii przez danego klienta traktowane jako całość	Całościowa ocena sklepu
Wyrażenie opinii dotyczącej jednego kryterium satysfakcji przez danego klienta	Ocena sklepu według pojedynczego kryterium

Źródło: opracowanie własne.

Tab. 4.35. Przykłady miar dla faktu *Realizacja zamówienia*

Szczegółowość faktu	Miary (przykłady)
Realizacja wielu zamówień	Średni czas realizacji zamówienia, średnie odchylenie czasu realizacji zamówienia, średni czas realizacji każdego etapu, średnie odchylenie czasu realizacji każdego etapu
Realizacja pojedynczego zamówienia	Czas realizacji zamówienia, różnica między czasem realizacji zamówienia a czasem średnim, czas realizacji każdego etapu zamówienia, odchylenie czasu realizacji każdego etapu od wartości średniej, procentowe odchylenie czasu realizacji zamówienia od wartości średniej, procentowe odchylenie czasu realizacji każdego etapu od wartości średniej
Pojedynczy etap realizacji zamówienia	Czas realizacji jednego etapu zamówienia, odchylenie czasu realizacji etapu od wartości średniej, procentowe odchylenie czasu realizacji etapu od wartości średniej, procentowy udział czasu realizacji etapu w czasie realizacji całego zamówienia

Źródło: opracowanie własne.

Tab. 4.36. Przykłady miar dla faktu *Koszty sprzedaży stacjonarnej*

Szczegółowość faktu	Miary (przykłady)
Całkowite koszty sprzedaży stacjonarnej	Koszty całkowite sprzedaży, koszty stałe sprzedaży, koszty zmienne sprzedaży, udział całkowitych kosztów sprzedaży w kosztach całkowitych działalności firmy
Grupa rodzajowa kosztów sprzedaży stacjonarnej	Całkowite koszty rodzajowe, udział całkowitych kosztów rodzajowych w całkowitych kosztach sprzedaży
Pojedynczy określony koszt sprzedaży stacjonarnej	Pojedynczy koszt danego rodzaju związany ze sprzedażą

Źródło: opracowanie własne.

Tab. 4.37. Przykłady miar dla faktu *Dostawy produktów*

Szczegółowość faktu	Miary (przykłady)
Wiele dostaw wielu produktów	Całkowity koszt dostaw, średni koszt dostawy, całkowity czas dostaw, średni czas dostawy, średni dystans dostaw, mediana czasu dostawy
Wiele dostaw tego samego produktu	Całkowity koszt dostaw danego produktu, średni koszt dostawy danego produktu, średni czas dostawy danego produktu, średni dystans dostaw produktu, mediana czasu dostawy produktu
Pojedyncza dostawa konkretnego produktu traktowana jako całość	Całkowity koszt danej dostawy danego produktu, czas danej dostawy danego produktu, odchylenie czasu danej dostawy od średniego czasu dostawy
Pojedyncza operacja realizowana w ramach pojedynczej dostawy konkretnego produktu	Czas trwania pojedynczej operacji, całkowity koszt wykonania pojedynczej operacji, udział czasu trwania operacji w całkowitym czasie danej dostawy, udział kosztu wykonania operacji w całkowitym koszcie danej dostawy

Źródło: opracowanie własne.

Tab. 4.38. Przykłady miar dla faktu *Reklamacje produktów*

Szczegółowość faktu	Miary (przykłady)
Wiele reklamacji wielu produktów traktowane jako całość	Liczba reklamacji, wartości reklamacji, koszty reklamacji, liczba klientów dokonujących reklamacji, liczba reklamowanych produktów, częstotliwość reklamacji, średni czas obsługi reklamacji, średni koszt obsługi reklamacji
Wiele reklamacji pojedynczego produktu jako całość	Liczba reklamacji, wartość reklamacji, koszty reklamacji, liczba klientów składających reklamację, liczba sztuk reklamowanego produktu, częstotliwość reklamacji, średni czas obsługi reklamacji
Pojedyncza reklamacja pojedynczego produktu	Wartość reklamacji, koszt reklamacji, długość czasu obsługi reklamacji, odchylenie czasu obsługi reklamacji od czasu średniego

Źródło: opracowanie własne.

Tab. 4.39. Przykłady miar dla faktu *Responsywność klientów w kampanii e-marketingowej*

Szczegółowość faktu	Miary (przykłady)
Reakcje wielu klientów na wiele kampanii	Liczba adresatów kampanii, liczba adresatów responsywnych, średni ogólny wskaźnik responsywności klientów, średni czas responsywności klientów, ogólna wartość responsywności
Reakcje wielu klientów na pojedynczą kampanię	
Reakcje wielu klientów na pojedyncze działanie w ramach pojedynczej kampanii	Liczba adresatów działania, liczba adresatów responsywnych, średni ogólny wskaźnik responsywności klientów, średni czas responsywności klientów, ogólna wartość responsywności
Reakcja konkretnego klienta na pojedynczą kampanię	Czas responsywności, wartość responsywności
Reakcja konkretnego klienta na pojedyncze działanie w ramach danej kampanii	Czas responsywności, wartość responsywności

Źródło: opracowanie własne.

Tab. 4.40. Przykłady miar dla faktu *Błędne działanie platformy*

Szczegółowość faktu	Miary (przykłady)
Wystąpienie wszystkich błędów podczas korzystania z platformy	Liczba błędów, częstotliwość wystąpienia błędu, całkowity koszt wystąpienia błędów, średni koszt wystąpienia błędu, całkowity koszt likwidacji błędów, całkowity czas błędnego działania platformy, średni czas naprawy błędu
Wystąpienie błędów określonego typu podczas korzystania z platformy	
Wystąpienie pojedynczego błędu określonego rodzaju podczas korzystania z platformy	Całkowity koszt wystąpienia błędu, całkowity koszt likwidacji błędu, czas błędnego działania platformy, czas naprawy błędu

Źródło: opracowanie własne.

Tab. 4.41. Przykłady miar dla faktu *Ruch w witrynie WWW*

Szczegółowość faktu	Miary (przykłady)
Wiele wizyt w witrynie WWW jako całość	Liczba unikalnych wizyt w witrynie, średni czas spędzony na jednej stronie witryny, minimalny czas spędzony na stronie witryny, maksymalny czas spędzony na stronie witryny, średnia liczba odwiedzonych stron, średnia wartość konwersji wizyty w witrynie
Pojedyncza wizyta w witrynie WWW jako całość	Średni czas spędzony na jednej stronie witryny, minimalny czas spędzony na stronie witryny, maksymalny czas spędzony na stronie witryny, liczba odwiedzonych stron, wartość konwersji wizyty w witrynie
Wiele wizyt na konkretnej stronie WWW jako całość	Średni czas spędzony na danej stronie WWW, minimalny czas spędzony na danej stronie witryny, maksymalny czas spędzony na danej stronie WWW, liczba unikalnych wizyt na danej stronie, całkowita wartość konwersji wizyt na danej stronie
Pojedyncza wizyta na konkretnej stronie WWW	Czas spędzony na danej stronie WWW, liczba wyświetleń danej strony, wartość konwersji wizyty na danej stronie

Źródło: opracowanie własne.

Tab. 4.42. Przykłady miar dla faktu *Zagrożenia atakami sklepu internetowego*

Szczegółowość faktu	Miary (przykłady)
Wiele ataków różnego rodzaju na sklep internetowy	Ogólna liczba ataków, liczba ataków odpartych, liczba rodzajów ataków, wskaźnik skuteczności odparcia ataków, częstotliwość występowania ataków, wartość strat w wyniku ataku, koszty przywrócenia stabilności działania sklepu, średni czas trwania ataku, średni czas przywrócenia stabilności działania sklepu
Grupa ataków określonego typu na sklep internetowy	Liczba ataków danego typu, liczba odpartych ataków danego typu, wskaźnik skuteczności odparcia ataków danego typu, odsetek liczby ataków danego typu w ogólnej liczbie ataków, wartość strat w wyniku ataków, koszty przywrócenia stabilności działania e-sklepu, średni czas trwania ataku, średni czas przywrócenia stabilności działania sklepu
Pojedynczy atak określonego rodzaju na sklep internetowy	Wartość strat w wyniku ataku, koszt przywrócenia stabilności działania e-sklepu, czas trwania ataku, czas przywrócenia stabilności działania sklepu
Pojedynczy atak danego rodzaju na dany zasób sklepu internetowego	Koszt utraconego zasobu, koszt przywrócenia zasobu, czas trwania ataku, czas przywrócenia stabilności działania sklepu

Źródło: opracowanie własne.

Jak widać w tabelach 4.32–4.43, miary natywne są charakterystyczne dla faktów o dużej ziarnistości (szczegółowości), podczas gdy miary kalkulowane, jak wartości całkowite, średnie, minimalne czy maksymalne, są adekwatne do opisu faktów o pewnym stopniu agregacji.

Tab. 4.43. Przykłady miar dla faktu *Skuteczność złożenia zamówień internetowych*

Szczegółowość faktu	Miary (przykłady)
Złożenie wielu zamówień widziane jako całość	Całkowita liczba rozpoczętych zamówień, liczba zamówień skutecznie złożonych, odsetek zamówień złożonych skutecznie, wartość rozpoczętych zamówień, wartość zamówień złożonych skutecznie, udział wartości skutecznie złożonych zamówień w wartości wszystkich rozpoczętych zamówień, średni czas trwania składania zamówienia, mediana czasu trwania składania zamówienia
Złożenie pojedynczego zamówienia jako całość	Wartość zamówienia, czas trwania składania zamówienia, odchylenie czasu trwania składania zamówienia od czasu średniego
Realizacja pojedynczego etapu składania zamówienia	Czas trwania pojedynczej akcji podczas składania zamówienia, odchylenie czasu trwania pojedynczej operacji od czasu średniego trwania tej operacji, udział długości czasu wykonania danej operacji w całkowitym czasie trwania składania zamówienia

Źródło: opracowanie własne.

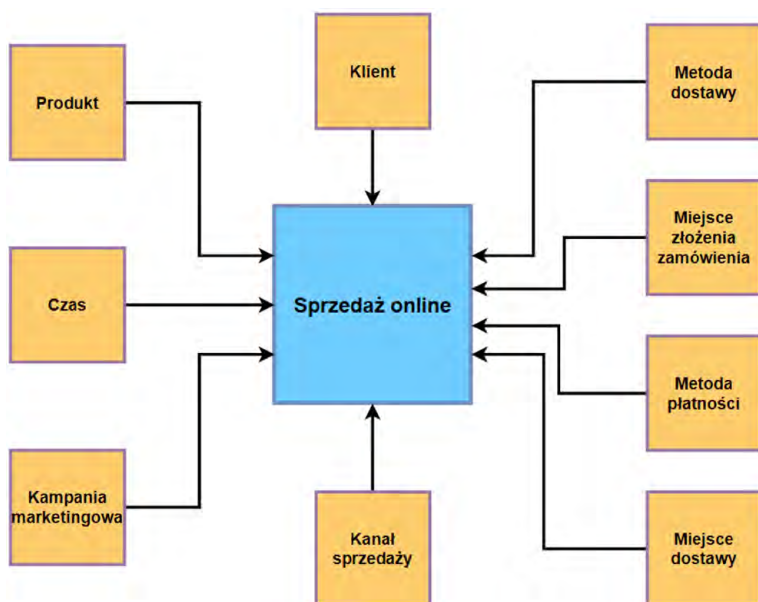
Kończąc prezentację tematyki dotyczącej definiowania miar, warto jeszcze przedstawić Czytelnikowi jedną sugestię, wynikającą z doświadczenia zawodowego Autorów niniejszej publikacji. Zdefiniowanie większej liczby niezbędnych miar już na tym etapie ogranicza liczbę operacji agregacji i kalkulacji w późniejszej analizie danych, co znacząco wpłynie na wzrost wydajności zarówno tworzonej bazy danych, jak i aplikacji wykorzystujących zgromadzone w niej dane.

4.7. Modelowanie hurtowni danych

Wszystkie dotychczas zrealizowane etapy modelowania wielowymiarowego umożliwiły dokładne określenie analizowanego zagadnienia, tj. identyfikację faktu, oraz określenie kontekstu tej analizy, tj. wskazanie wymiarów dla poszczególnych faktów. Dodatkowo dla każdego faktu wskazano miary, czyli ilościowy sposób jego opisu, a dla każdego wymiaru tego faktu podano atrybuty, dzięki którym możliwa będzie poprawna interpretacja tych miar. Rezultaty powyższych działań stanowią podstawę do budowy modeli hurtowni danych (data mart), które z jednej strony umożliwią użytkownikowi biznesowemu lepsze zrozumienie analizowanej rzeczywistości i określenie stopnia poprawności jej odwzorowania, z drugiej zaś – pozwolą na utworzenie w docelowym

środowisku bazodanowym struktur danych, umożliwiającym składowanie danych wykorzystywanych w przyszłej analizie.

Pierwszym naturalnym modelem, mającym na celu zobrazowanie poprawności identyfikacji faktu i jego wymiarów, jest tzw. model konceptualny, którego przykład przedstawiono na rysunku 4.2. Obrazuje on zależności pomiędzy faktem (zagadnieniem) sprzedaży prowadzonej z wykorzystaniem różnych kanałów online a wymiarami opisującymi tego rodzaju sprzedaż.



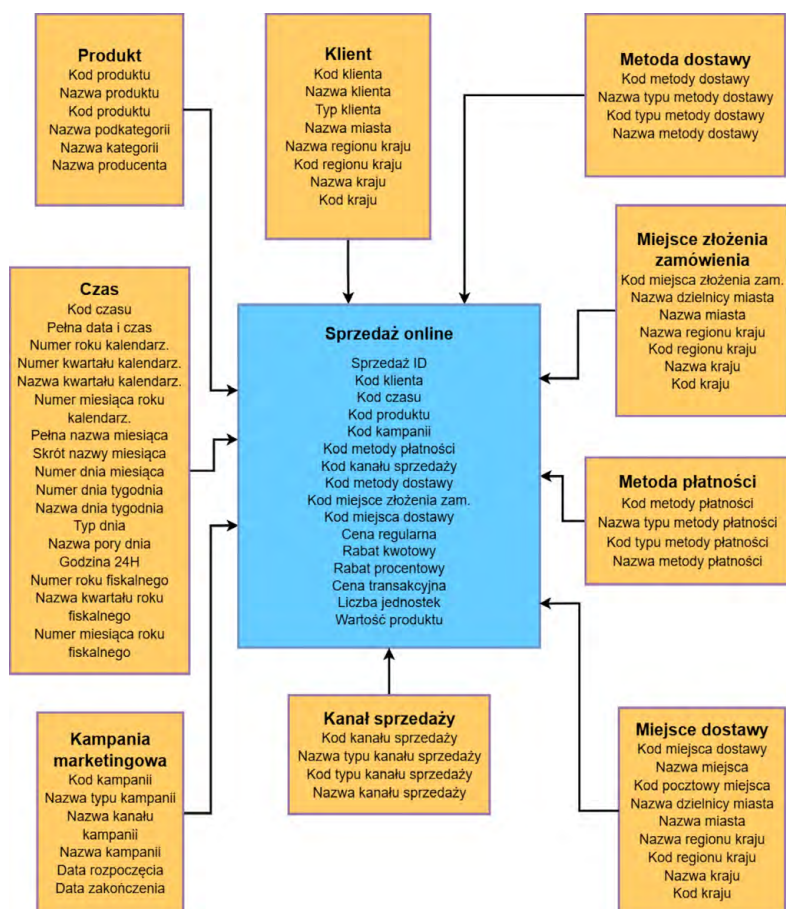
Rys. 4.2. Model konceptualny dla zagadnienia *Sprzedaż online*

Źródło: opracowanie własne.

Model ten, mimo dużej prostoty, jest bardzo czytelny i zrozumiały zarówno dla użytkownika biznesowego, jak i technologicznego. Stanowi pewnego rodzaju metodę weryfikacji właściwego zrozumienia informacyjnych potrzeb użytkownika końcowego, a przyszłemu deweloperowi bazy danych i aplikacji dostarcza informacji o rodzajach źródeł danych, które będą potrzebne podczas implementacji tego rozwiązania.

Znajomość modelu konceptualnego pozwala przejść do kolejnego etapu projektowania hurtowni danych (data mart), w którym następuje dokładne sprecyzowanie pojęć reprezentujących fakt i jego wymiary oraz określenie

szczegółowości ich opisu. Rezultatem tego działania będzie stworzenie modelu logicznego, w którym znajdzie się dokładna specyfikacja atrybutów poszczególnych wymiarów oraz wykaz miar opisujących fakt. Przykład takiego modelu zobrazowano na rysunku 4.3.



Rys. 4.3. Model logiczny dla zagadnienia *Sprzedaż online*

Źródło: opracowanie własne.

Model ten jest rozwinięciem modelu konceptualnego przedstawionego na rysunku 4.2. Analizując wykaz atrybutów kolejnych wymiarów, można zauważyć, że każdy z nich jest opisany z bardzo dużą szczegółowością. Przykładowo wymiar *Produkt* reprezentuje konkretny model produktu, wymiar *Klient* – pojedynczego klienta, wymiar *Miejsce dostawy* – konkretną lokalizację miejsca

odbioru zamówionego produktu, a wymiar *Czas* – pełną godzinę. Znajomość znaczenia każdego z wymiarów umożliwia poprawną interpretację faktu *Sprzedaż online*, który oznacza dokładnie pojedynczą pozycję (produkt) jednej transakcji, opisaną jej identyfikatorem (*Sprzedaż ID*). Potwierdzeniem takiego znaczenia faktu są również miary, które odnoszą się do jednego produktu.

Zarówno model konceptualny, jak i logiczny są istotnymi elementami w projektowaniu analitycznej bazy danych, jednak nie zawierają one w sobie wystarczających informacji, na podstawie których można bezpośrednio przejść do działań deweloperskich związanych z jej tworzeniem. Dlatego też konieczna jest transformacja modelu logicznego do postaci, w której zostaną zaprezentowane konkretne struktury danych i związki między nimi, umożliwiające gromadzenie danych niezbędnych do opisu wymiarów i faktów. Działania te, określane mianem modelowania danych, prowadzą do zbudowania modelu danych hurtowni danych (data mart), którego implementacja w docelowym środowisku bazodanowym spowoduje powstanie fizycznych obiektów, takich jak tabele, widoki czy indeksy. Ze względu na znaczną obszerność tematyki tworzenia modelu danych hurtowni danych (data mart) zagadnienie to zostanie dokładnie zaprezentowane w odrębnym, kolejnym rozdziale niniejszej publikacji.

4.8. Zadania do samodzielnego rozwiązania

Proszę wykonać poniższe zadania związane z projektowaniem analitycznej bazy danych dla przykładowej firmy świadczącej globalnie usługi typu bike-sharing, tj. wypożyczania rowerów w obrębie miasta:

1. Utworzyć model konceptualny data mart, która umożliwi analizę zyskowności usług świadczonych przez przykładową firmę, zidentyfikować różne fakty związane z realizacją usług, określić ich szczegółowość oraz wskazać dla każdego z nich odpowiednie wymiary. Przedstawić model konceptualny data mart w postaci graficznej.
2. Na podstawie modelu konceptualnego utworzonego w zadaniu 1 proszę zbudować model logiczny data mart. Dla każdego wymiaru, skojarzonego z danym faktem, proszę określić odpowiednie atrybuty i nazwać je w zrozumiały sposób, natomiast dla poszczególnych faktów – zdefiniować miary niezbędne w ocenie zyskowności usług. Przedstawić model logiczny data mart w postaci graficznej.

5. Modele danych hurtowni danych

Znajomość faktów i opisujących je wymiarów pozwala w kolejnym etapie tworzenia analitycznej bazy danych przejść do działań związanych ze zbudowaniem jej modelu danych, zawierającego definicję tabel faktów, tabel wymiarów oraz relacji między nimi. Model danych stanowi naturalne rozszerzenie modelu logicznego poprzez jego uszczegółowienie polegające na identyfikacji rodzaju danych (kolumn tabeli i typów danych), które będą opisywać zarówno fakty, jak i wymiary.

W literaturze przedmiotu wyróżnia się dwa podstawowe modele danych hurtowni danych (data mart) [27]:

- model gwiazdy (ang. *star schema*);
- model płatka śniegu (ang. *snowflake schema*).

Możliwe jest także zbudowanie rozszerzonego modelu danych, jakim jest konstelacja, czyli model, w którym relacje istnieją także między tabelami faktów lub tabelami wymiarów opisujących różne fakty. W niniejszym rozdziale szczegółowo zaprezentowano poszczególne modele wraz z przykładami obrazującymi ich implementację. W modelowaniu danych zastosowano podejście relacyjne, przy czym dla lepszej czytelności postanowiono termin „relacja” zastąpić określeniem „tabela”, a termin „krotka” – pojęciem „rekord”.

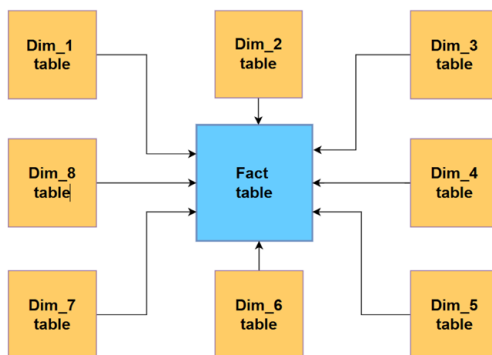
5.1. Model gwiazdy

Podstawowym modelem danych wykorzystywanym w hurtowniach danych jest model gwiazdy. To najprostszy i najbardziej intuicyjny sposób przedstawienia zależności pomiędzy danymi opisującymi jeden fakt a danymi charakterystycznymi dla powiązanych z nim poszczególnych wymiarów. W modelu tym wyróżnia się centralną tabelę faktu oraz otaczające ją tablele wymiarów. Wśród jego charakterystycznych cech należy wymienić następujące [10]:

- każdy wymiar jest opisany dokładnie przez jedną tabelę wymiaru;
- tabela faktu jest powiązana jedną lub wieloma relacjami z każdą tabelą wymiaru;
- tabela faktu zawiera klucze obce odwołujące się wyłącznie do kluczy głównych tabel wymiarów oraz kolumny odpowiadające miarom;
- każda tabela wymiaru jest w relacji typu jeden-do-wielu wyłącznie z tabelą faktu i dostarcza kontekstu do opisu danych numerycznych znajdujących się w tej tabeli;
- nie istnieją relacje pomiędzy tabelami wymiarów;
- tabela wymiaru zawiera kolumny opisujące wszystkie atrybuty wymiaru (jeden atrybut może być opisany wieloma kolumnami);
- tabela wymiaru nie jest znormalizowana;
- hierarchia danego wymiaru tworzona jest z użyciem wyłącznie jednej tabeli opisującej ten wymiar;
- tabela faktu zawiera dane o tym samym stopniu agregacji, tj. stopniu szczegółowości opisu faktu.

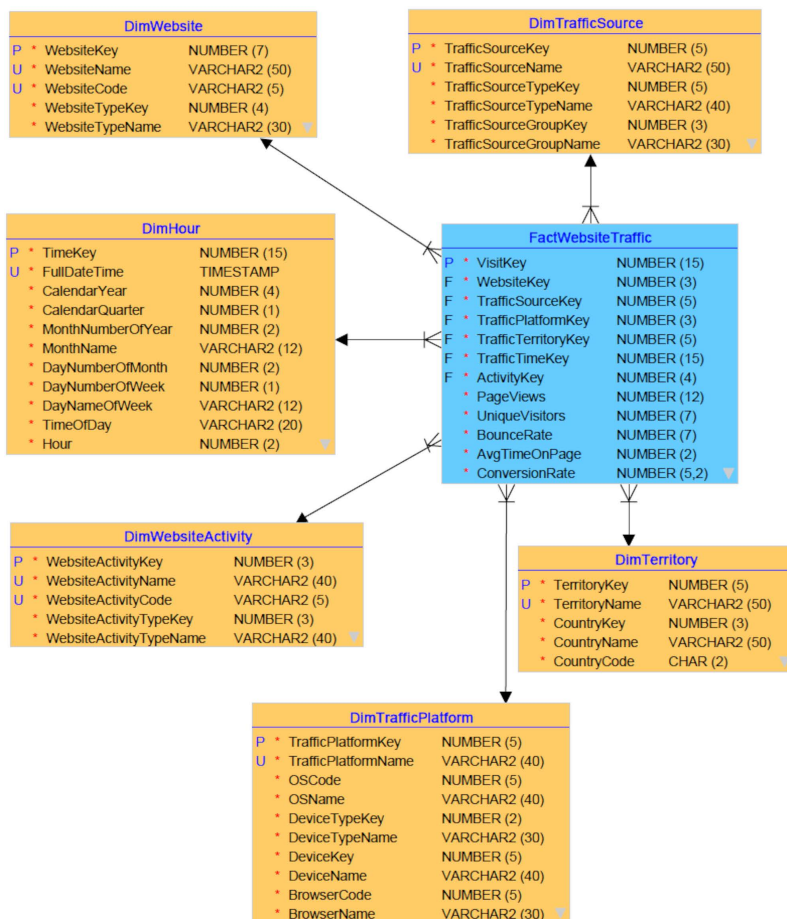
Schemat ideowy modelu gwiazdy przedstawiono na rysunku 5.1.

Przykład implementacji modelu danych typu gwiazda przedstawiono na rysunku 5.2. Tabela faktu *FactWebsiteTraffic* zawiera dane zagregowane do poziomu godziny (kolumna *TrafficTimeKey*), opisujące ruch na konkretnej stronie WWW (kolumna *WebsiteKey*), pochodzący z różnych źródeł (kolumna *TrafficSourceKey*), generowany z wielu krajów (kolumna *TrafficTerritoryKey*) i platform dostępowych (kolumna *TrafficPlatformKey*) oraz rodzaj działania podejmowanego na tej stronie (kolumna *ActivityKey*).



Rys. 5.1. Schemat ideowy modelu gwiazdy

Źródło: opracowanie własne.



Rys. 5.2. Fizyczny model danych data mart z użyciem modelu gwiazdy

Źródło: opracowanie własne.

W modelu gwiazdy tabele wymiarów są szerokie, tzn. zawierają dużą liczbę kolumn, które opisują dany wymiar – np. *DimHour* czy *DimTrafficPlatform*. Niektóre z tabel mogą zawierać niewielką liczbę rekordów, np. tabela *DimTerritory* (około 230 według źródeł ONZ), a inne – bardzo dużą liczbę wierszy, np. tabela *DimHour* (43 800 dla okresu pięciu lat). Struktura każdej tabeli wymiaru nie jest znormalizowana, co sprawia, że będzie ona zawierać dane redundantne przechowywane w niektórych kolumnach. Przykładowo kolumna *MonthName* w tabeli *DimHour* zawiera wielokrotne powtórzenia tej samej nazwy miesiąca dla każdej godziny każdego dnia danego miesiąca. Brak normalizacji danych w takiej tabeli jest celowym zabiegiem, który eliminuje zbędne i kosztowne czasowo złączenia z innymi tabelami wymiaru podczas wykonywania zapytań. Z kolei tabela faktu *FactWebsiteTraffic* zawiera kolumny odpowiadające wszystkim miarom wykorzystywanym do pomiaru ruchu na stronie internetowej (*PageViews*, *UniqueVisitors*, *BounceRate*, *AvgTimeonPage* i *ConversionRate*). Jest to jedyny zbiór danych numerycznych, które będą używane w różnych kalkulacjach analitycznych.

Wśród głównych zalet modelu gwiazdy wymienia się [10]:

- czytelny, intuicyjny i łatwy w zrozumieniu sposób opisu analizowanego obszaru rzeczywistości;
- odzwierciedlenie naturalnego sposobu myślenia użytkownika biznesowego o analizowanym zagadnieniu, ułatwiające późniejsze użycie przez niego danych w działaniach analitycznych;
- niską złożoność metadanych, zapewniającą efektywne wykorzystanie data mart jako źródła dla narzędzi business intelligence;
- bardzo wysoką użyteczność w modelowaniu danych w data mart;
- dużą prostotę zapytań analitycznych kierowanych do bazy danych;
- zapewnienie krótkiego czasu wykonania zapytań do bazy danych ze względu na redundancję danych w tabelach wymiarów i ograniczoną do niezbędnego minimum liczbę złączeń między tabelą faktu i tabelami wymiarów;
- skrócenie czasu ładowania danych do data mart, wynikające z obecności mniejszej liczby tabel w porównaniu z typowym modelem relacyjnym;
- zapewnienie łatwej implementacji wolnozmiennych wymiarów typu 2 i 3 (szczegóły na ten temat Czytelnik znajdzie w podrozdziale 7.2);
- wspieranie tego rodzaju modelu przez wszystkie współczesne narzędzia business intelligence;

- ułatwienie użytkownikom narzędzi business intelligence szybkiego tworzenia raportów analitycznych i dashboardów.

Mimo tak wielu zalet stosowania modelu gwiazdy w literaturze przedmiotu wskazuje się także jego istotne wady, do których należą [2]:

- ograniczone możliwości wykonywania złożonych analiz w dowolnych przekrojach informacyjnych ze względu na narzuconą i sztywną strukturę danych;
- duży rozmiar tabel wymiarów, który wymaga znacznej przestrzeni dyskowej;
- brak możliwości implementacji wolnozmiennych wymiarów typu 4, 5, 6 i 7;
- możliwość wystąpienia niespójności danych w tabelach wymiarów ze względu na nadmiarowość danych składowanych w wielu kolumnach tych tabel;
- możliwość zastosowania modelu głównie w data mart; w przypadku hurtowni danych występują bardzo duże trudności w modelowaniu złożonych zagadnień.

Podsumowując zaprezentowane treści dotyczące modelu gwiazdy, należy uznać jego wysoką użyteczność zarówno w tworzeniu wielowymiarowego opisu przedmiotowego zagadnienia, jak i późniejszej jego analizie. Model ten pozwala odwzorować obszar analizy w sposób zgodny ze sposobem patrzenia na niego przez użytkownika końcowego aplikacji analitycznej oraz osoby podejmującej decyzje, co sprawia, że wartość biznesowa takiego rozwiązania jest bardzo wysoka.

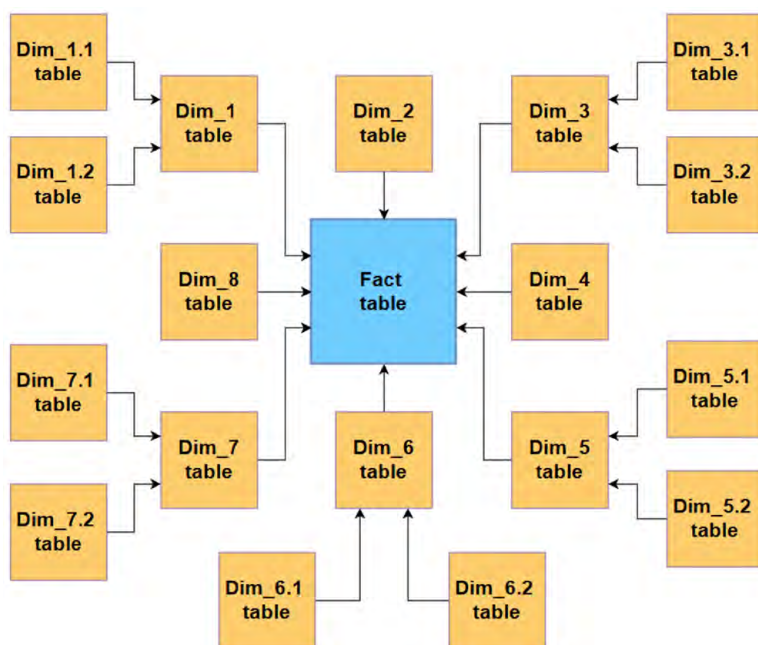
5.2. Model płatka śniegu

Alternatywnym sposobem wielowymiarowego modelowania rzeczywistości jest użycie modelu płatka śniegu, który doskonale sprawdza się w przypadku bardziej złożonych zagadnień analitycznych. Podobnie jak model gwiazdy, płatek śniegu posiada centralną tabelę faktu połączoną z wieloma tabelami wymiarów, jednak niektóre z nich opisywane są kolejnymi tabelami, zawierającymi bardziej szczegółowe dane opisujące wymiar.

Wśród najistotniejszych cech modelu płatka śniegu należy wymienić następujące [23]:

- każdy wymiar może być opisany przez wiele tabel;
- tabela faktu nie jest powiązana relacjami ze wszystkimi tabelami wymiarów, lecz tylko z wybranymi – tj. z tymi, które reprezentują wymaganą szczegółowość opisu faktu;
- tabela faktu zawiera klucze obce odwołujące się do kluczy głównych wybranych tabel wymiarów oraz kolumny odpowiadające miarom;
- dana tabela wymiaru zawiera wyłącznie wybrane atrybuty tego wymiaru, a nie wszystkie, jak w przypadku modelu gwiazdy;
- pomiędzy tabelami wymiarów definiowane są tylko relacje typu jeden-do-wielu;
- niektóre tabele wymiarów mają postać znormalizowaną;
- hierarchia danego wymiaru jest definiowana przez kolumny pochodzące z wielu tabel wymiarów;
- tabela faktu zawiera dane o tym samym stopniu agregacji.

Schemat ideowy modelu płatka śniegu przedstawiono na rysunku 5.3.



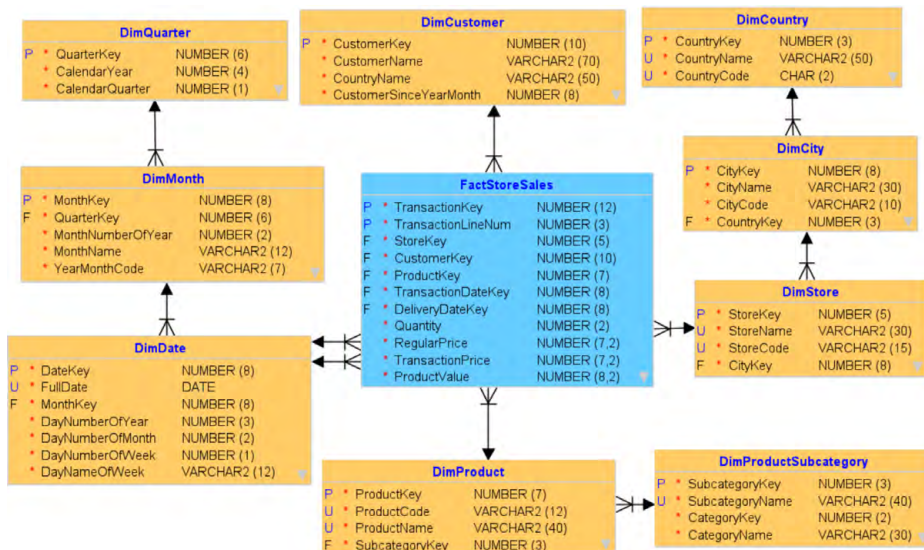
Rys. 5.3. Schemat ideowy modelu płatka śniegu

Źródło: opracowanie własne.

Można zatem zauważyć, że model płatka śniegu znacząco różni się od modelu gwiazdy, gdyż charakteryzuje się większą złożonością, która jednak ma także swoje zalety. Należą do nich:

- możliwość zastosowania do modelowania danych w hurtowni danych, a nie tylko w data mart;
- mniejsza zajętość przestrzeni dyskowej przez tabele wymiarów, które są w postaci znormalizowanej (brak redundancji danych);
- możliwość wielokrotnego wykorzystania tych samych tabel wymiarów do opisu różnych wymiarów;
- większa elastyczność w wykonywaniu zaawansowanych analiz z użyciem dowolnych i złożonych przekrojów informacyjnych.

Na rysunku 5.4 przedstawiono fizyczny model danych data mart, zbudowany dla obszaru sprzedaży stacjonarnej z wykorzystaniem modelu płatka śniegu. Jest to wzorcowy przykład użycia tego rodzaju modelu. Zaprezentowany przykład opisuje sprzedaż produktów w sklepach stacjonarnych zlokalizowanych w różnych miastach w wielu krajach na rzecz klientów zarówno krajowych, jak i zagranicznych.



Rys. 5.4. Fizyczny model danych data mart z użyciem modelu płatka śniegu – wariant podstawowy

Źródło: opracowanie własne.

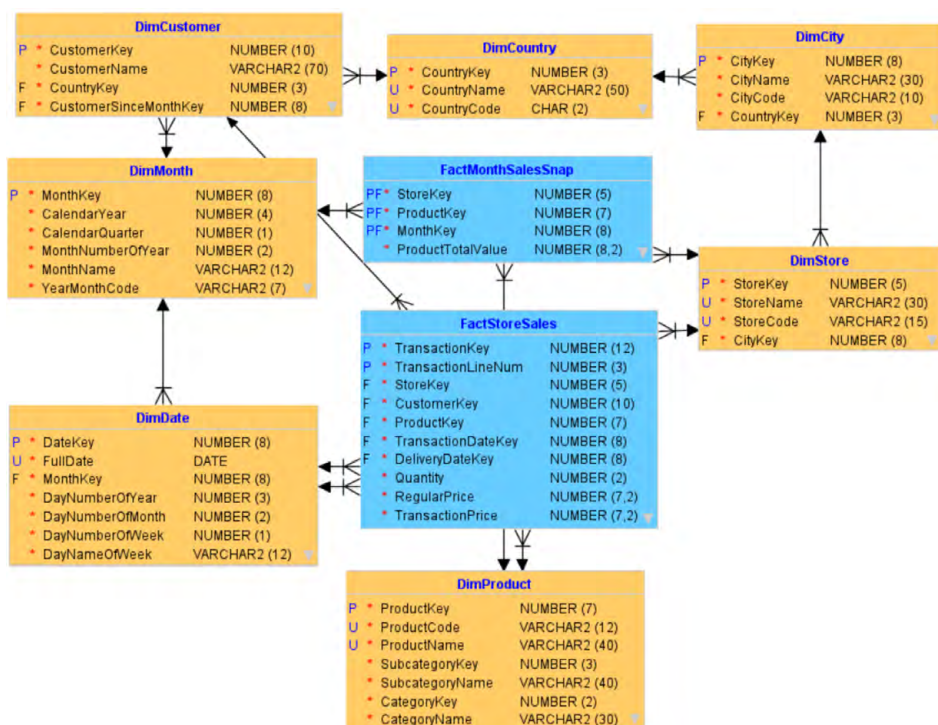
Zgodnie z uprzednio podaną charakterystyką modelu płatka śniegu można zauważyć, że wymiar czasu jest tutaj opisany trzema tabelami, tj. *DimDate*, *DimMonth* i *DimQuarter*, wymiar produktowy – dwoma tabelami, tj. *DimProduct* i *DimProductSubcategory*, a wymiar sklepu – trzema tabelami, tj. *DimStore*, *DimCity* i *DimCountry*. Ponadto wszystkie tabele wymiaru sklepu są znormalizowane i istnieją pomiędzy nimi odpowiednie relacje jeden-do-wielu. Tabela faktu sprzedażowego pozostaje w relacjach tylko z niektórymi tabelami wymiarów, tj. z *DimProduct*, *DimDate*, *DimStore* i *DimCustomer*. Każdy rekord tej tabeli reprezentuje zakup jednego produktu w ramach pojedynczej transakcji.

Analizując struktury danych na tym diagramie data mart, można jednak zadać sobie pytanie: w jakim celu dokonano normalizacji tabel opisujących wymiar czasu, sklepu i po części produktu? Jakie są korzyści implementacji takiego rozwiązania? Koncentrując uwagę wyłącznie na bardzo wąskim obszarze analizy, reprezentowanym przez tabelę faktu *FactStoreSales* i wykorzystaniu wspomnianych tabel wymiarów do opisu tego faktu, można uznać, że takie działanie nie ma uzasadnienia, a wręcz prowadzi do obniżenia wydajności zapytań (duża liczba operacji join), może być też źródłem problemów związanych z integralnością danych – np. w kolumnach *CountryName* i *CustomerSinceYearMonth* tabeli *DimCustomer*. Ponadto zmiana nazwy kraju lub sposobu reprezentacji miesiąca w danym roku (np. z formatu YYYY-MM na MMYYYY) wymusza utworzenie dodatkowych procesów ETL, które umożliwią modyfikację danych w tabelach *DimCountry*, *DimCustomer* czy *DimMonth*. Jeśli jednak rozszerzyć zakres sposobu użycia tabel znormalizowanych, to można zauważyć ich użyteczność w sytuacjach, gdy pojawiają się inne tabele faktów, wymagające opisu za pomocą tych tabel wymiarów. Przypadek taki zilustrowano na rysunku 5.5.

Nowa postać fizycznego modelu danych eliminuje problem potencjalnej niespójności danych poprzez powiązanie tabel *DimCountry* i *DimMonth* z *DimCustomer* oraz ujawnia użyteczność znormalizowanych tabel wymiarów *DimMonth* i *DimStore* do opisu innych faktów, które znajdują się w tej samej data mart. Dodatkowa tabela faktu *FactMonthSalesSnap* została umieszczona na diagramie wyłącznie dla zobrazowania takiej sytuacji i nie stanowi elementu pojedynczego schematu płatka śniegu. Warto także zauważyć, że w tym przypadku tabela wymiaru *DimCountry* została wykorzystana zarówno do opisu wymiaru

klienta, jak i wymiaru sklepu. Podobna sytuacja dotyczy tabeli *DimMonth*, która opisuje zarówno wymiar klienta, jak i wymiar czasu.

W zmodyfikowanej postaci schematu płatka śniegu atrybuty tego samego wymiaru mogą znajdować się w wielu tabelach. Przykładem tego są wymiary czasu, klienta i sklepu. Pierwszy z nich opisywany jest przez dwie tabele, tj. *DimDate* i *DimMonth*. Pozostałe dwa – odpowiednio przez tabele *DimCustomer*, *DimMonth* i *DimCountry* oraz *DimStore*, *DimCity* i *DimCountry*. Patrząc w szerszej perspektywie użycia danych dotyczących czasu, można sobie wyobrazić, że tabela *DimMonth* zostanie poddana normalizacji, co doprowadzi do powstania kolejnych tabel wymiarów, np. *DimQuarter* i *DimYear*. To samo podejście można zastosować do tabeli *DimProduct*, zastępując ją trzema tabelami, tj. *DimProduct*, *DimProductSubcategory* i *DimProductCategory*.



Rys. 5.5. Fizyczny model danych data mart z użyciem modelu płatka śniegu – wariant zaawansowany

Źródło: opracowanie własne.

Decydując się na zastosowanie modelu płatka śniegu do opisu rzeczywistości, warto wziąć pod uwagę jego wady, do których należą:

- duża trudność w zrozumieniu opisywanego zagadnienia, gdyż model nie odzwierciedla naturalnego sposobu myślenia o rzeczywistości przez użytkownika biznesowego;
- duża złożoność zapytań do bazy danych, będąca rezultatem występowania wielu tabel i złączeń między nimi;
- znaczne wydłużenie czasu wykonywania zapytań ze względu na konieczność złączenia wielu tabel oraz zwiększoną liczbę operacji odczytów z dysku;
- wydłużenie czasu realizacji procesów ETL, zasilających hurtownię danych (data mart), w porównaniu z modelem gwiazdy;
- większe trudności w implementacji wolnozmiennych wymiarów, wynikające z konieczności wprowadzenia zmian w wielu tabelach wymiarów;
- duża złożoność metadanych, która utrudnia efektywne wykorzystanie hurtowni (data mart) jako źródła danych dla narzędzi business intelligence.

Decyzja o wyborze schematu gwiazdy lub płatka jako najlepszego sposobu modelowania danych zależy od wielu czynników, takich jak charakter danych, oczekiwania dotyczące zapytań do bazy danych, ograniczenia pamięci używanych do gromadzenia danych czy wydajność przyszłych procesów analitycznych wykonywanych z użyciem takiego źródła danych. Wybierając pomiędzy tymi modelami, należy wziąć pod uwagę przede wszystkim poziom złożoności analizowanego zagadnienia. Schemat gwiazdy jest rekomendowany w przypadku modelowania prostych tematów, które wymagają niewielkiej liczby wymiarów o niskiej kardynalności danych oraz niewielkiej liczbie poziomów hierarchii wymiarów używanych do analizy faktu. Innymi istotnymi czynnikami, przemawiającymi za użyciem schematu gwiazdy, będą duża szybkość generowania odpowiedzi na zapytania analityczne oraz możliwość ich definiowania przez użytkownika biznesowego.

Z kolei schemat płatka śniegu jest najlepszym wyborem w sytuacji, gdy opis zagadnienia wymaga dużej liczby wymiarów, ich kardynalność jest wysoka oraz w analizie faktu wykorzystuje się wiele wielopoziomowych hierarchii wymiarów. Schemat ten powinno się stosować również w sytuacji, gdy użytkownikowi końcowemu zależy na dużej szczegółowości i elastyczności tworzenia zapytań, szczególnie typu ad hoc, czyli takich, które nie są trwale zapisane w aplikacji analitycznej jako jej funkcjonalności czy w bazie danych

jako widoki, lecz są definiowane doraźnie, w miarę pojawiania się nowych analitycznych potrzeb informacyjnych.

Dla podsumowania zaprezentowanych treści na temat podstawowych analitycznych modeli danych w tabeli 5.1 zestawiono ich najważniejsze cechy.

Tab. 5.1. Istotne cechy modelu gwiazdy i płatka śniegu

Model gwiazdy	Model płatka śniegu
Intuicyjny, prosty i czytelny sposób opisu analizowanego zagadnienia	Złożony, trudny w interpretacji opis analizowanego zagadnienia
Stosowany tylko w prostych analizach	Możliwość wykorzystania w analizach zaawansowanych
Duża sztywność (stabilność) struktur danych opisujących zagadnienie	Duża elastyczność w modelowaniu relacji między danymi używanymi w analizie
Zawiera tabelę faktów i tabele wymiarów powiązane wyłącznie z tabelą faktów	Zawiera tabelę faktów i tabele wymiarów, które mogą być powiązane bezpośrednio z tabelą faktów lub innymi tabelami wymiarów
Tabele wymiarów mają postać zdenormalizowaną	Tabele wymiarów mają postać zarówno normalną, jak i zdenormalizowaną
Opis wszystkich atrybutów danego wymiaru znajduje się tylko w jednej tabeli wymiaru	Atrybuty wymiaru są opisane w wielu tabelach o różnych poziomach szczegółowości
Hierarchia wymiaru tworzona z użyciem jednej tabeli wymiaru	Hierarchia wymiaru opisana danymi z wielu tabel wymiarów
Niska złożoność zapytań do bazy danych (mała liczba operacji join między tabelami)	Wysoka złożoność zapytań do bazy danych (duża liczba złączeń między tabelami)
Bardzo krótki czas wykonywania zapytań	Długi czas generowania odpowiedzi na zapytania
Łatwy sposób implementacji modelu w bazie danych	Bardziej skomplikowany, w porównaniu ze schematem gwiazdy, sposób implementacji modelu w bazie danych
Wysoki poziom redundancji danych	Niski poziom redundancji danych
Dużą zajętość przestrzeni dyskowej	Znacznie mniejsza zajętość przestrzeni dyskowej w porównaniu ze schematem gwiazdy
Domyślnie implementowany w narzędziach business intelligence w procesie analizy danych	Konieczność transformacji modelu do postaci gwiazdy w narzędziach business intelligence

Źródło: opracowanie własne.

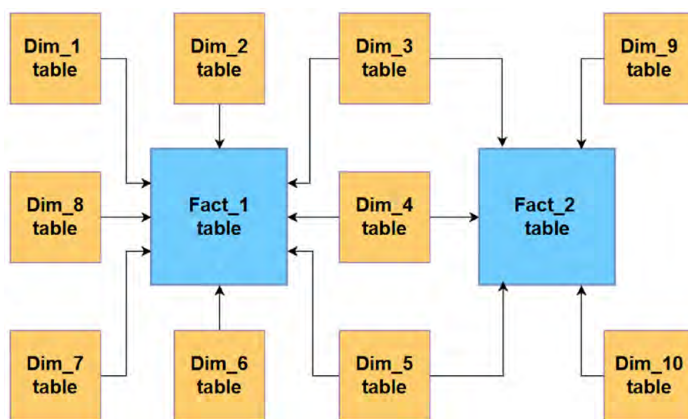
5.3. Model konstelacji

Dotychczas zaprezentowane modele danych, używane w analitycznych bazach danych, charakteryzowały się obecnością wyłącznie jednej tabeli faktu. Jednak w praktyce modelowania wielowymiarowego bardzo często występuje przypadek definiowania wielu tabel faktów, które są opisywane wspólnymi tabelami wymiarów. Taki model nazywany jest konstelacją faktów. Można go postrzegać jako zbiór wielu modeli typu gwiazda, dlatego też niekiedy określany jest także jako galaktyka.

Charakterystyczne cechy modelu konstelacji są następujące:

- zawiera wiele tabel faktów, które opisują to samo zagadnienie;
- istnieją wspólne tabele wymiarów, które są powiązane z wieloma tabelami faktów;
- każda tabela faktu opisuje dane zagadnienie z innego punktu widzenia i zawiera odrębny zestaw miar;
- poszczególne tabele faktów mogą zawierać dane o różnym stopniu szczegółowości (różna ziarnistość danych).

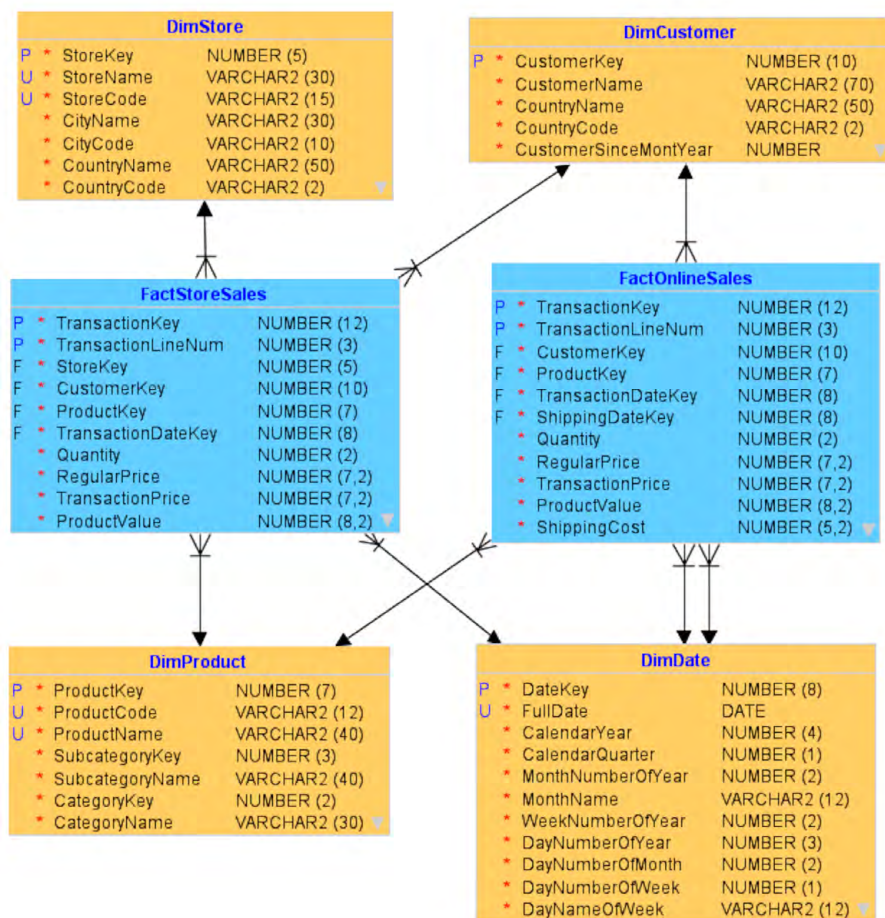
Schemat ideowy modelu konstelacji przedstawiono na rysunku 5.6.



Rys. 5.6. Schemat ideowy modelu konstelacji faktów

Źródło: opracowanie własne.

Model konstelacji faktów jest najbardziej skomplikowanym schematem danych w hurtowniach danych w porównaniu z modelem gwiazdy czy płatka śniegu. Przykład implementacji tego rodzaju modelu przedstawiono na rysunku 5.7.



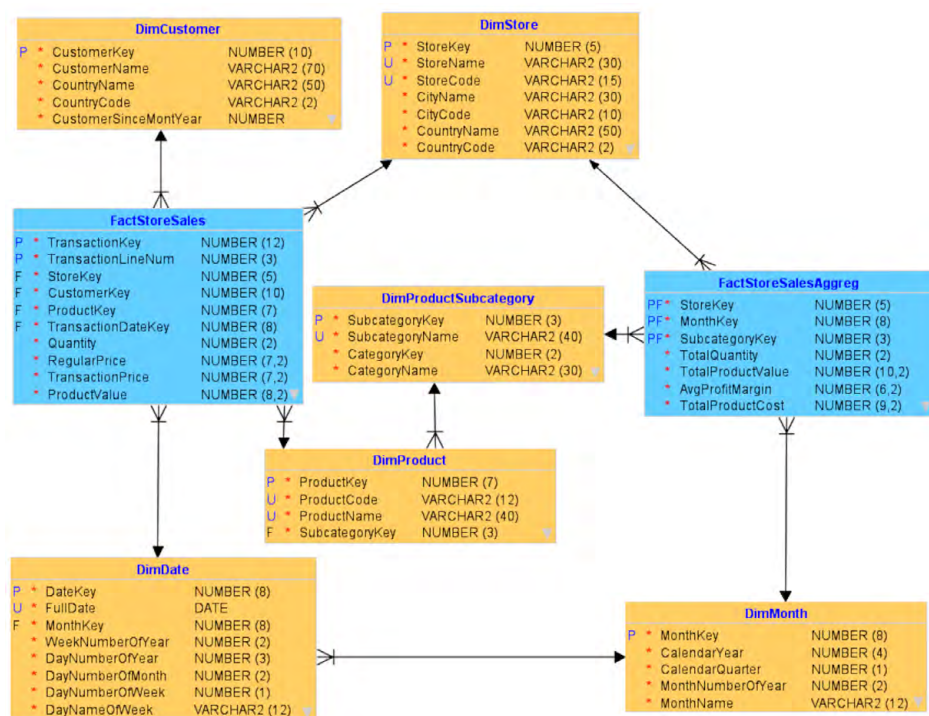
Rys. 5.7. Fizyczny model danych data mart z użyciem modelu konstelacji faktów – wariant podstawowy

Źródło: opracowanie własne.

Obydwie tabele faktów opisują zagadnienie sprzedaży – odpowiednio stacjonarnej (tabela *FactStoreSales*) i online (tabela *FactOnlineSales*). Jedna z tabel wymiarów, tj. *DimStore*, pozostaje w związku wyłącznie z tabelą faktu *FactStoreSales*. Natomiast wszystkie pozostałe tabele wymiarów są wykorzystane do opisu danych w obydwu tabelach faktów. Można zatem powiedzieć, że tabele te są współdzielone w różnych modelach gwiazdy, z których każdy opisuje inny rodzaj sprzedaży.

W przedstawionym przykładzie różne tabele faktów opisane są wymiarami o tym samym poziomie szczegółowości. Każdy rekord w tabelach *FactStoreSales*

i *FactOnlineSales* zawiera dane o sprzedaży danemu klientowi konkretnego produktu w jednym dniu. Jednak w rzeczywistości występują takie sytuacje, w których jedna z tabel faktu pozostaje w relacji z tabelą wymiaru o danym poziomie szczegółowości, np. z tabelą opisującą dzień, a inna tabela faktu jest połączona z tabelą tego samego wymiaru, lecz o wyższym poziomie ogólności, np. opisującą miesiąc. Przypadek taki zilustrowano na rysunku 5.8.



Rys. 5.8. Fizyczny model danych data mart z użyciem modelu konstelacji faktów – wariant zaawansowany

Źródło: opracowanie własne.

W zaawansowanej postaci modelu konstelacji, oprócz uprzednio wymienionej już tabeli *FactStoreSales*, można zauważyć obecność zagregowanej tabeli faktu *FactStoreSalesAggreg*, w której znajdują się dane o wyższym poziomie ogólności, tj. zagregowane do miesiąca i podkategorii produktowej. Kolumny opisujące miary w tej tabeli, tj. *TotalQuantity*, *TotalProductValue*, *AvgProfitMargin* i *TotalProductCost*, mają całkowicie odmienne znaczenie niż kolumny miar w tabeli *FactStoreSales*.

Mimo dużej złożoności modelu jego struktura zapewnia pewne korzyści, wśród których należy wymienić:

- możliwość opisu złożonych zagadnień analitycznych (jest odpowiedni do modelowania danych w hurtowniach danych);
- większą skalowalność modelu, umożliwiającą tworzenie nowych tabel faktów bez konieczności zmian struktur danych w już istniejącym schemacie (wykorzystanie istniejących tabel wymiarów do opisu nowych tabel faktów oraz dodanie nowych tabel wymiarów bez wpływu na już istniejące);
- zmniejszenie rozmiaru bazy danych ze względu na normalizację tabel wymiarów i wielokrotne ich wykorzystanie w opisie różnych tabel faktów,
- możliwość wykonywania złożonych analiz wymagających agregacji o różnym stopniu szczegółowości.

Wybierając pomiędzy modelami gwiazdy, płatków śniegu i konstelacji faktów, należy szukać kompromisu pomiędzy prostotą modelu i jego elastycznością, a także między wydajnością zapytań i funkcjonalnością oczekiwaną przez użytkownika końcowego. Tworząc aplikację analityczną, która będzie wykorzystywała dany model danych, przede wszystkim powinno się mieć na uwadze jego użyteczność i adekwatność do modelowanego zagadnienia.

5.4. Zadania do samodzielnego rozwiązania

Wykorzystując model logiczny data mart, utworzony w podrozdziale 4.8 dla firmy świadczącej usługi bike-sharing w skali globalnej, proszę wykonać następujące zadania:

1. Zbudować model danych data mart typu płatek śniegu, opisujący fakt pojedynczego wypożyczenia roweru i przedstawić go w postaci graficznej, używając dowolnego narzędzia do modelowania baz danych.
2. Zbudować model danych data mart typu konstelacja faktów, opisujący fakt rezerwacji roweru we wskazanej stacji oraz fakt wypożyczenia roweru przez klienta i przedstawić go w postaci graficznej, wykorzystując dowolne narzędzie do modelowania baz danych.

6. Modelowanie faktów

W niniejszym rozdziale dokładnie omówiono różne techniki umożliwiające zbudowanie tabel w analitycznej bazie danych przeznaczonych do składowania danych opisujących fakty.

6.1. Podstawowe rodzaje tabel faktów

Znajomość modelu logicznego data mart czy hurtowni danych oraz rodzajów modeli danych pozwala przejść w kolejnym etapie projektowania analitycznej bazy danych do zamodelowania w niej faktów w postaci pojedynczej tabeli faktu lub wielu tabel faktów. Podczas definiowania struktury takiej tabeli należy zachować następujące zasady:

- struktura tabeli faktu musi umożliwiać opis analizowanego zagadnienia z taką dokładnością, jaka jest istotna z punktu widzenia potrzeb informacyjnych końcowego użytkownika biznesowego;
- kolumny odpowiadające miarom powinny opisywać analizowane zagadnienie możliwie w sposób elementarny, tj. zapewnić addytywność miar; jeśli to konieczne, można także zdefiniować kolumny odpowiadające miarom semiaddytywnym lub nieaddytywnym;

- tabela faktu musi zawierać kolumny opisujące kontekst wielkości mierzalnych, czyli kolumny będące kluczami obcymi do tabel wymiarów; dodatkowo może zawierać kolumny odpowiadające wymiarom zdegenerowanym (np. numer zamówienia dla faktu *Sprzedaż online*, identyfikator odsłony strony WWW dla faktu *Dostępność witryny WWW* czy też identyfikator ataku na zasób sieci dla faktu *Zagrożenia atakami sklepu internetowego*);
- wartości kluczy obcych tabeli faktu nie powinny być równe NULL;
- dopuszcza się wartość NULL dla kolumn opisujących miary;
- nie należy używać znakowych typów danych dla kolumn tabeli ze względu na wzrost jej rozmiaru spowodowany większą liczbą bajtów potrzebnych do reprezentacji danych tego typu. Zaleca się stosowanie wyłącznie typów numerycznych.

W tabeli faktu nie jest konieczne tworzenie odrębnej kolumny, która byłaby kluczem podstawowym – jego funkcję może pełnić kombinacja wszystkich kluczy obcych tabel wymiarów. Oprócz kolumn reprezentujących wymiary i miary w tabeli faktu mogą znaleźć się również kolumny dodatkowo opisujące dany fakt, np. identyfikator transakcji w tabeli *FactOnlineSales* charakteryzującej sprzedaż internetową czy też numer przesyłki w tabeli *FactDelivery* powiązanej z faktem reprezentującym proces dostawy zamówionych produktów. Kolumny te określane są mianem wymiarów zdegenerowanych, które zostaną szczegółowo opisane w podrozdziale 7.1 niniejszej publikacji.

W porównaniu z tabelami wymiarów tabele faktów są wąskie, to znaczy zawierają mniejszą liczbę kolumn. Jednak o ich wielkim rozmiarze decyduje olbrzymia liczba wierszy, znacząco większa niż w tabelach wymiarów. Dlatego też tabele te często są partycjonowane w bazie danych, aby skrócić czas wykonywania do nich zapytań analitycznych [20].

W zależności od tego, co reprezentują sobą wartości kluczy obcych, odpowiadających poszczególnym wymiarom, tabela faktu będzie opisywać analizowane zagadnienie z różnym stopniem szczegółowości. Jeśli np. klucz wymiaru *Produkt* dla faktu *Sprzedaż online* opisuje konkretny produkt, a klucz wymiaru *Czas* jest powiązany z konkretnym dniem, wówczas rekord w tabeli faktu będzie charakteryzował sprzedaż internetową tego produktu w jednym dniu. Jeśli jednak klucz wymiaru *Czas* reprezentuje miesiąc, to zachowując znaczenie klucza dla wymiaru *Produkt*, rekord w tabeli faktu będzie opisywał sprzedaż produktu w okresie jednego miesiąca. Dlatego też, projektując strukturę tabeli

faktu, konieczne jest dokładne określenie pojęcia tego faktu, aby zapewnić możliwość wykonywania analiz z wymaganym poziomem szczegółowości.

Tabele faktów stanowią główne źródło danych używanych w analizie, gdyż wszelkie operacje kalkulacji typu sumowanie, zliczanie czy wyznaczanie wartości średniej wykonywane są z wykorzystaniem danych zawartych w kolumnach odpowiadających poszczególnym miarom.

W zależności od sposobu interpretacji faktu i zastosowanej struktury danych do jego opisu można wyróżnić cztery podstawowe rodzaje tabel faktów [7, 24]:

- transakcyjna tabela faktu (ang. *transactional fact table*);
- tabela okresowych migawek (ang. *periodic snapshot fact table*);
- tabela skumulowanych migawek (ang. *accumulating snapshot fact table*);
- tabela faktu bez faktu (ang. *factless fact table*).

Transakcyjna tabela faktu jest najczęściej spotykaną tabelą w wielu hurtowniach danych czy data marts. Przedstawia ona wybrane zagadnienie z największym stopniem szczegółowości (dużą ziarnistością), odpowiadającym wystąpieniu pojedynczego działania lub zdarzenia w analizowanej rzeczywistości. Kolumny w takiej tabeli opisują fakt w sposób bardzo zbliżony do tego, jaki jest stosowany w systemie transakcyjnym (relacyjnym), co może być źródłem pewnych wątpliwości co do poprawności takiej definicji tabeli. Jednak tego rodzaju tabela faktu pozwala uzyskać bardzo szczegółowe dane w przyszłej analizie, szczególnie podczas wykonywania operacji drążenia danych.

Transakcyjne tabele faktów pozwalają odpowiedzieć na szczegółowe pytania typu „ile?”. Mogą one brzmieć następująco:

- Ile klientów dokonało zakupu w od początku bieżącego miesiąca?
- Ile sztuk konkretnego produktu sprzedano w podanym dniu klientom, którzy opłacili zamówienia za pomocą danej metody płatności?
- Ile wynoszą przychody ze sprzedaży produktów, uzyskane z zamówień złożonych za pomocą aplikacji mobilnej w bieżącym miesiącu?
- O ile zmieniła się liczba unikalnych wizyt na stronie głównej naszej organizacji w porównaniu z ubiegłym miesiącem?
- Ile było incydentów bezpieczeństwa w witrynie naszego sklepu internetowego w bieżącym tygodniu?
- O ile zmieniła się liczba zwrotów produktów zakupionych w sklepie internetowym w porównaniu z tym samym miesiącem poprzedniego roku?

Miary definiowane w tych tabelach mogą być używane w kalkulacjach wykonywanych z użyciem każdego wymiaru powiązanego z danym faktem, co sprawia, że tabele te dostarczają informacji w wielu złożonych przekrojach i umożliwiają zaspokojenie najbardziej wyrafinowanych potrzeb informacyjnych przyszłych użytkowników aplikacji analitycznych.

Przykłady struktur transakcyjnych tabel faktów przedstawiono na rysunkach 6.1 i 6.2. W tabeli *FactStoreSales* (rys. 6.1) jeden jej rekord reprezentuje pojedynczą pozycję (kolumna *SalesLineNumber*) transakcji sprzedaży, opisanej wartością kolumny *StoreSalesKey*, wykonanej w sklepie, którego identyfikator znajduje się w kolumnie *StoreKey*. Zastosowane tutaj miary – np. *CatalogPrice*, *DiscountAmount*, *TransactionPrice*, *Quantity*, *ProductValue* czy *ProductCost* – są charakterystyczne dla jednego produktu (kolumna *ProductKey*) będącego przedmiotem konkretnej transakcji.

FactStoreSales		
P *	StoreSalesKey	VARCHAR2 (10)
P *	SalesLineNumber	NUMBER (3)
*	StoreKey	NUMBER (7)
*	CustomerKey	NUMBER (10)
*	ProductKey	NUMBER (5)
*	PaymentMethodKey	NUMBER (2)
*	PromotionKey	NUMBER (4)
*	OrderDateKey	NUMBER (8)
*	CatalogPrice	NUMBER (7,2)
*	DiscountAmount	NUMBER (7,2)
*	DiscountPctg	NUMBER (2)
*	TransactionPrice	NUMBER (7,2)
*	Quantity	NUMBER (2)
*	ProductValue	NUMBER (8,2)
*	ProductCost	NUMBER (5,2)
*	ProfitMargin	NUMBER (5,2)

Rys. 6.1. Transakcyjna tabela *FactStoreSales* dla faktu *Sprzedaż stacjonarna*

Źródło: opracowanie własne.

Struktura tabeli *FactStoreSales* umożliwia udzielenie odpowiedzi na takie przykładowe pytania, jak:

- Ile wynoszą przychody uzyskane przez poszczególne sklepy w kolejnych dniach określonego roku ze sprzedaży poszczególnych produktów?
- Ile wynosi całkowita kwota udzielonych rabatów w ramach poszczególnych promocji organizowanych w kolejnych miesiącach bieżącego roku?

- Ile wynoszą koszty sprzedaży poszczególnych produktów w bieżącym miesiącu i jak one się zmieniły w porównaniu z tym samym miesiącem ubiegłego roku?
- Ilu klientów opłaciło swoje zakupy w poszczególnych sklepach, korzystając z dostępnych metod płatności w kolejnych latach?
- Ile wynosi średnia wartość zakupu dla poszczególnych sklepów w kolejnych miesiącach bieżącego roku i jak ona się zmieniała w tym czasie?

Tego rodzaju pytań można zdefiniować bardzo dużo, a ich liczba będzie się zwiększała wraz ze wzrostem liczby wymiarów oraz miar opisujących fakt sprzedaży.

Innym przykładem transakcyjnej tabeli faktu jest *FactWebsiteAccess* (rys. 6.2), w której każdy rekord jest opisem jednej wizyty (kolumna *WebsiteAccessKey*) klienta identyfikowanego przez wartość *VisitorKey* na konkretnej stronie WWW (kolumna *WebsiteKey*). Kolumny reprezentujące miary tego faktu, czyli np. *LoadTime*, *TimeOnSite* czy *TimeSinceLastVisit*, będą zawierać wartości specyficzne dla bardzo szczegółowego zdarzenia, jakim jest wejście w określonym czasie (*EntryDatetimeKey*) na daną stronę webową.

FactWebsiteAccess		
P *	SessionKey	NUMBER (15)
P *	WebsiteKey	NUMBER (9)
P *	VisitorKey	NUMBER (12)
P *	VisitorTypeKey	NUMBER (2)
P *	DeviceKey	NUMBER (6)
P *	EntryLocationKey	NUMBER (12)
P *	WebBrowserKey	NUMBER (2)
P *	SourceOfVisitKey	NUMBER (9)
P *	EntryDatetimeKey	NUMBER (15)
P *	ExitDatetimeKey	NUMBER (15)
*	LoadTime	NUMBER (5)
*	TimeOnSite	NUMBER (5)
*	TimeSinceLastVisit	NUMBER (9)
*	VisitSequenceNumber	NUMBER (7)
*	PageSequenceNumber	NUMBER (5)
*	EventsNumberOnPage	NUMBER (3)
*	LCP	NUMBER (6,2)
*	FID	NUMBER (5)
*	CLS	NUMBER (4,2)
*	FCP	NUMBER (6,2)
*	INP	NUMBER (5) ▼

Rys. 6.2. Transakcyjna tabela *FactWebsiteAccess* dla faktu *Dostępność witryny WWW*

Źródło: opracowanie własne.

Wykorzystując tabelę *FactWebsiteAccess*, można zdefiniować następujące przykładowe pytania:

- Ile wynosi średni i całkowity czas trwania wizyty w firmowej witrynie webowej, wyświetlanej na poszczególnych rodzajach urządzeń dostępowych w kolejnych tygodniach bieżącego roku, i jak on się zmienił w porównaniu z tym samym okresem roku ubiegłego?
- Ile wynosi średni czas pomiędzy kolejnymi wizytami na danej stronie WWW, pochodzącymi z poszczególnych źródeł w kolejnych tygodniach poszczególnych miesięcy bieżącego roku?
- Ile wynosi w bieżącym miesiącu średni czas wczytywania poszczególnych stron na różnych urządzeniach dostępowych dla wejść z poszczególnych lokalizacji i jak on się zmienił w porównaniu z tym samym miesiącem ubiegłego roku?
- Ile stron WWW zostało średnio wyświetlonych podczas pojedynczej sesji w firmowej witrynie webowej zainicjowanej na poszczególnych rodzajach urządzeń dostępowych w bieżącym miesiącu i jak zmieniła się ta wartość w porównaniu z ubiegłym miesiącem tego samego roku?
- Ile wynosi średnia wartość wskaźnika INP (*Interaction to Next Paint*) dla poszczególnych stron WWW wczytywanych na różnych urządzeniach dostępowych i jak ona się zmieniła w porównaniu z ubiegłym miesiącem bieżącego roku?

Tak duża szczegółowość opisu tego zdarzenia może wynikać z konieczności analizowania pojedynczych wizyt w celu poprawienia doświadczenia użytkownika korzystającego z firmowej witryny webowej (np. poprzez zwiększenie responsywności stron czy szybkości ich ładowania) i późniejszego profilowania kontentu webowego dostarczanego do danego odwiedzającego.

Podsumowując charakterystykę transakcyjnych tabel faktów, należy stwierdzić, iż duża ziarnistość danych w takiej tabeli sprawia, że zapytania kierowane do niej będą wykonywały się znacznie dłużej w porównaniu z czasem zapytań adresowanych do tabel, w których dane wcześniej zagregowano na wyższych poziomach hierarchii wymiarów. Dużą zaletą tworzenia transakcyjnych tabel faktów jest jednak możliwość identyfikowania wzorców zachowań osób biorących udział w różnych zdarzeniach, wzorców zdarzeń czy trendów dotyczących analizowanych zagadnień. Obecność tego typu tabel

w analitycznej bazie istotnie podnosi wartość tego rozwiązania jako narzędzia wspierającego procesy decyzyjne.

Kolejnym rodzajem tabeli faktu jest tabela okresowych migawek, która posiada następujące cechy:

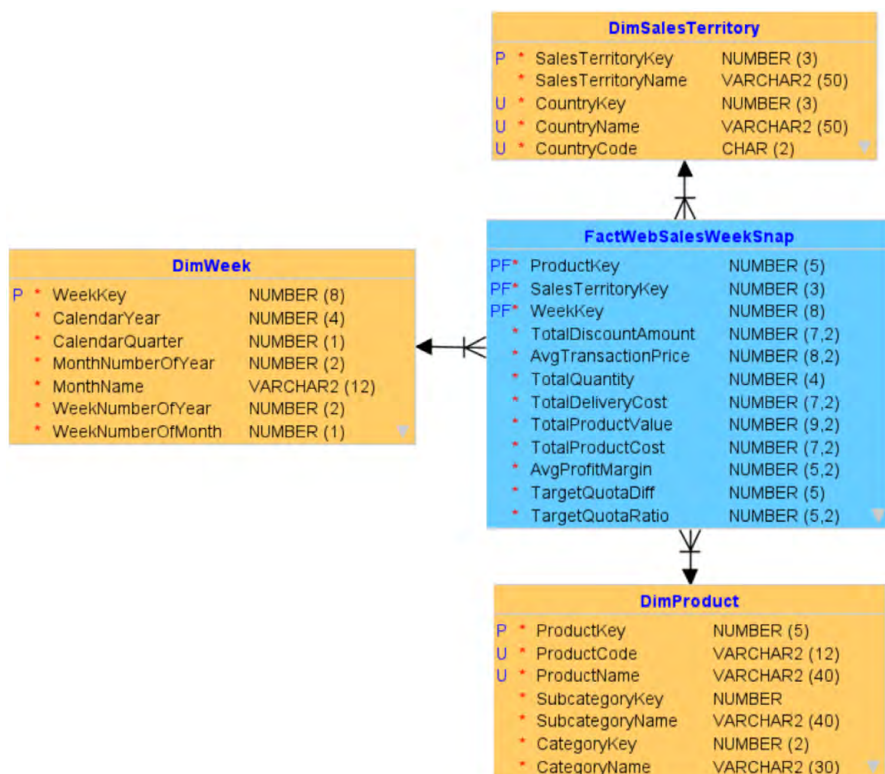
- każdy wiersz opisuje zagadnienie w sposób zagregowany według przyjętego jednego atrybutu wymiaru czasu – np. tygodnia, miesiąca, kwartału czy roku;
- tabela jest gęsta, tj. dla każdej wartości atrybutu opisującego wymiar czasu istnieje wiersz zawierający wartości miar dla tego okresu, nawet jeśli te wartości transakcyjne są równe zero lub NULL;
- struktura tabeli umożliwia skrócenie czasu wykonywania zapytań, które wykorzystują jeden z poziomów hierarchii wymiaru czasu jako kryterium agregacji danych (nie jest konieczna agregacja w locie, gdyż istnieją już wiersze zawierające potrzebne dane).

Tabele tego rodzaju gromadzą dane umożliwiające analizę szeregów czasowych. Zazwyczaj głównym celem ich tworzenia jest zapewnienie możliwości prognozowania przebiegu zjawisk. Dodatkowo są one używane do: wyjaśnienia, jakie czynniki wpływały na określony przebieg zjawiska w przeszłości; identyfikowania wzorców zjawisk; określania trendów i dynamiki zmian zachodzących w analizowanej rzeczywistości; symulowania przebiegu zjawisk w określonych warunkach opisanych wymiarami.

Ze względu na zagregowany charakter wartości miar w tabeli definiuje się kolumny odpowiadające głównie miarom semiaddytywnym lub nieaddytywnym, jakkolwiek istnieją również tabele, w których występują miary addytywne. Przykład tabeli okresowych migawek przedstawiono na rysunku 6.3. W tabeli *FactWebSalesWeekSnap* przechowywane są dane o sprzedaży webowej, zagregowane do poziomu tygodnia. Rekord w tabeli wymiaru *DimWeek* identyfikuje konkretny tydzień, opisany takimi atrybutami, jak numer tygodnia w roku (*WeekNumberOfYear*) czy numer tygodnia w miesiącu (*WeekNumberOfMonth*). Ponadto do opisu tygodnia użyto atrybutów: numer roku kalendarzowego (*CalendarYear*), numer kwartału w roku kalendarzowym (*CalendarQuarter*) czy nazwa miesiąca (*MonthName*). Tabela wymiaru *DimProduct* opisuje produkt będący przedmiotem sprzedaży, a tabela *DimSalesTerritory* – kraj, z którego pochodzi zamówienie. Przykładowa tabela faktu zawiera kolumny reprezentujące zarówno miary addytywne (np. *TotalQuantity*, *TotalDeliveryCost*,

TotalProductValue czy *TotalProductCost*), semiaddytywne (*TargetQuotaDiff*), jak i nieaddytywne (*AvgProfitMargin*, *TargetQuotaRatio*). Na podstawie ich wartości można odpowiedzieć na następujące przykładowe pytania analityczne:

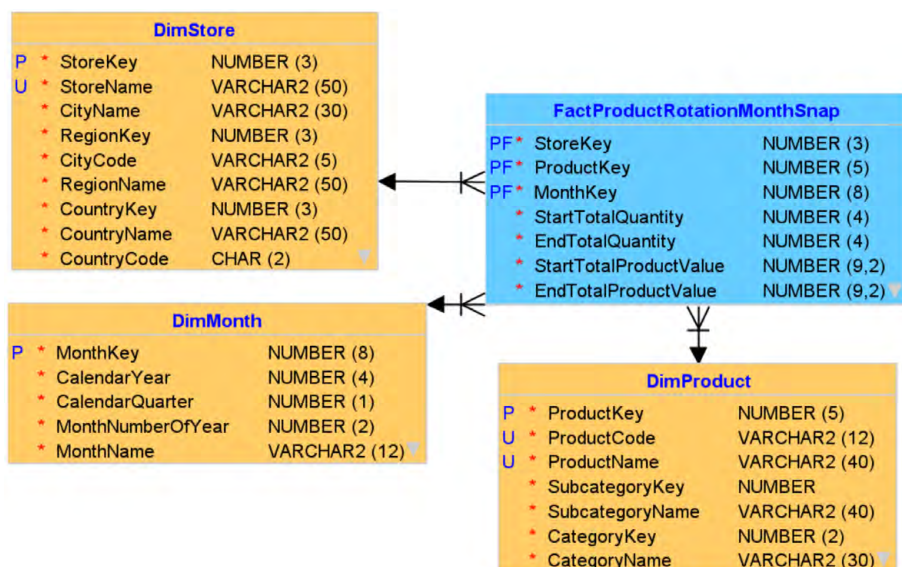
- Ile będą wynosić prawdopodobne przychody ze sprzedaży poszczególnych produktów w kolejnych tygodniach bieżącego roku w poszczególnych krajach oraz ile sztuk produktów można sprzedać w tym okresie?
- Ile wynosiła średnia cena transakcyjna każdego produktu w kolejnych tygodniach danego roku w każdym kraju, z którego pochodzą zamówienia?
- Ile wynosi średnia zyskowność ze sprzedaży produktów w kolejnych tygodniach i jak ona się zmieniała w porównaniu z tym samym okresem roku ubiegłego?
- Ile wynosi odchylenie wykonania sprzedaży od założonych celów w kolejnych tygodniach bieżącego roku i jaka jest dynamika jego zmian w porównaniu z tym samym okresem ubiegłego roku?



Rys. 6.3. Tabela okresowych migawek *FactWebSalesWeekSnap* dla faktu *Sprzedaż webowa*

Źródło: opracowanie własne.

Inny przykład tabeli okresowych migawek przedstawiono na rysunku 6.4. W tym przypadku tabela faktu *FactProductRotationMonthSnap* opisuje rotacje produktów w magazynach w kolejnych miesiącach. Dane ilościowe zawarte w kolumnach *StartTotalQuantity*, *EndTotalQuantity*, *StartTotalProductValue* i *EndTotalProductValue* zostały wyznaczone na poziomie miesiąca, co wynika z definicji wymiaru opisanego tabelą *DimMonth*. W odróżnieniu od poprzedniego przykładu wszystkie miary są nieaddytywne i określają stan magazynowy produktu na początek (*StartTotalQuantity*) i koniec (*EndTotalQuantity*) miesiąca oraz wartość zmagazynowanych produktów w tych momentach czasowych. Kalkulowanie wartości tych miar jest możliwe przy użyciu odpowiedniej transakcyjnej tabeli faktu, jednak ze względu na olbrzymią jej wielkość czas otrzymania wyników byłby nieakceptowalny przez użytkownika końcowego. W takiej sytuacji z pomocą przychodzą tabele okresowych migawek, które eliminują konieczność przetwarzania danych z transakcyjnych tabel faktów, co znacząco wpływa na zwiększenie wydajności zapytań kierowanych do hurtowni danych czy data mart.



Rys. 6.4. Tabela okresowych migawek *FactProductRotationMonthSnap* dla faktu *Rotacja produktów*

Źródło: opracowanie własne.

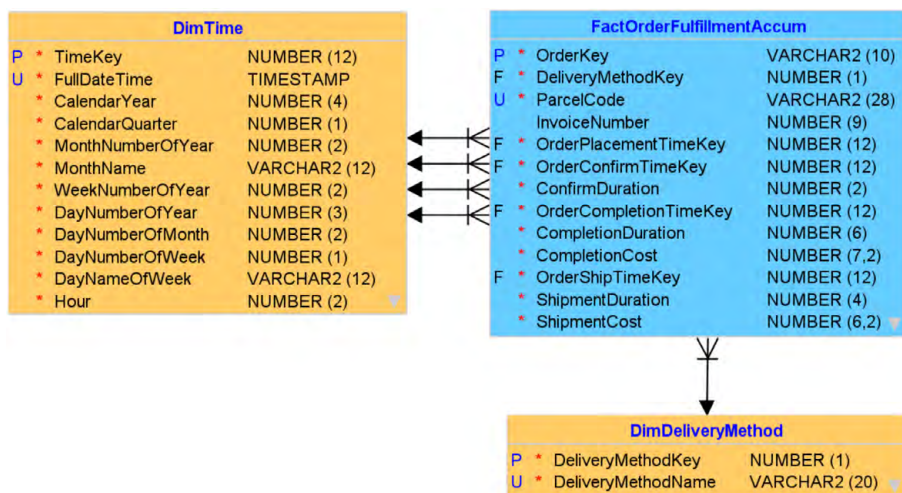
Opisane dotychczas rodzaje tabel faktów doskonale sprawdzają się w przypadku wykonywania analiz typu post (ocena tego, co się już wydarzyło) czy prognozowania bądź identyfikowania wzorców. Ich struktura zapewnia wysoką wydajność zapytań kierowanych do tych zasobów. Jednak w praktyce często okazuje się, że oprócz wyżej wymienionych działań analitycznych konieczne jest także określenie efektywności i skuteczności realizacji różnych procesów czy przebiegu zjawisk, szczególnie takich, które charakteryzują się pewną ustaloną etapowością (sekwencją) czynności. Zaprezentowane uprzednio tabele faktów nie zapewniają możliwości wykonania tego typu oceny w optymalnym czasie. Dlatego też w takiej sytuacji wykorzystuje się tabelę faktu, określaną mianem tabeli skumulowanych migawek, która:

- służy do opisu kolejnych etapów działania lub przebiegu zjawiska; etapy te są niezmiennie w zakresie ich liczby i kolejności realizacji (np. obsługa zamówień, transport przesyłek, obsługa reklamacji);
- zawiera wiele kluczy obcych odwołujących się do wymiaru czasu w celu opisanie chwili rozpoczęcia każdego etapu (liczba takich kluczy obcych jest równa liczbie etapów);
- każdy etap opisuje za pomocą odrębnego zestawu miar (najczęściej jest to ilościowy opis zasobu zużytego do realizacji danego etapu, np. czas i koszt realizacji, czy liczba zaangażowanych osób);
- jej niektóre kolumny kluczy obcych mogą przyjmować wartość NULL;
- dzięki swojej strukturze zapewnia krótki czas pozyskania danych na temat kolejnych etapów działań czy przebiegu zjawiska.

Każdy wiersz tabeli opisuje jeden pełny cykl zjawiska lub procesu. Jest on tworzony w momencie wystąpienia jego pierwszego etapu, a następnie aktualizowany w chwili rozpoczęcia kolejnego etapu. Modyfikacja danego wiersza jest kontynuowana aż do zakończenia pełnego danego cyklu.

Najprostszy przykład takiej tabeli o nazwie *FactOrderFulfillmentAccum* dla faktu *Efektywność realizacji zamówień* zaprezentowano na rysunku 6.5. W modelowanym przykładzie zamówienie jest alokowane w sklepie internetowym, może zawierać wiele pozycji i jest realizowane jako całość, tzn. wszystkie zamówione produkty dostarczane są w jednej przesyłce. Wszystkie etapy cyklu życia takiego zamówienia obejmują: złożenie zamówienia, potwierdzenie przyjęcia zamówienia, kompletację zamówienia oraz wysyłkę.

Rekord w tabeli wymiaru *DimTime* reprezentuje czas (data, godzina, minuty), który zostanie użyty wielokrotnie w tabeli faktu w kolumnach opisujących moment rozpoczęcia kolejnych etapów cyklu życia zamówienia. W momencie wstawienia nowego rekordu w tabeli faktu uzupełnione będą dane określające identyfikator zamówienia, metodę dostawy oraz czas złożenia zamówienia. Pozostałe dane w tym wierszu pojawiają się z pewnym opóźnieniem, co oznacza, że wartości kluczy obcych *OrderConfirmTimeKey*, *OrderCompletionTimeKey* i *OrderShipTimeKey* mogą przyjmować wartość NULL. Czasy trwania poszczególnych etapów zapisane są odpowiednio w kolumnach *ConfirmationDuration*, *CompletionDuration* i *ShipmentDuration*, natomiast koszty realizacji kolejnych etapów – odpowiednio w kolumnach *CompletionCost* i *ShipmentCost*.

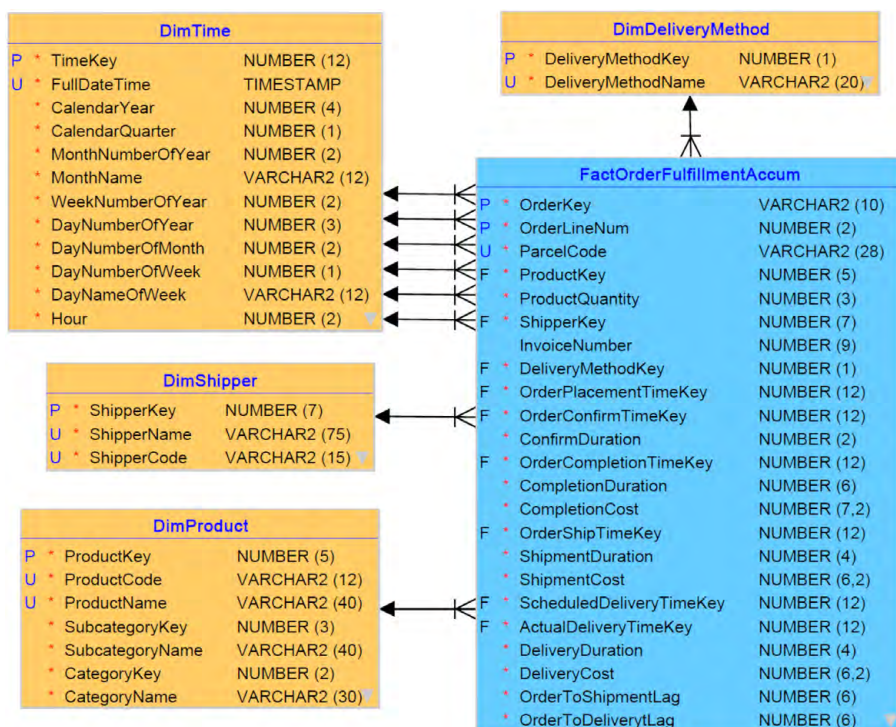


Rys. 6.5. Tabela skumulowanych migawek *FactOrderFulfillmentAccum* dla faktu *Realizacja zamówienia*

Źródło: opracowanie własne.

Znacznie bardziej zaawansowany przykład tabeli skumulowanych migawek przedstawiono na rysunku 6.6. Obrazuje on cykl życia zamówienia złożonego na platformie Allegro lub Amazon, które zawiera wiele pozycji, a każda z nich jest dostarczana od różnego sprzedawcy inną metodą dostawy w różnym dniu w odrębnej przesyłce. W tym przypadku rekord w tabeli faktu opisuje cykl życia pojedynczej pozycji zamówienia (jednego produktu). W porównaniu z poprzednim przykładem cykl ten dodatkowo charakteryzuje się

etapem doręczenia produktu, dla którego zdefiniowano kolumny opisujące planowany (*ScheduledDeliveryTimeKey*) i rzeczywisty (*ActualDeliveryTimeKey*) czas tego zdarzenia. Ponadto wprowadzono dwie miary, określające wielkość opóźnienia zarówno wysłania produktu (*OrderToShipmentLag*), jak i realizacji pełnego cyklu życia jednej pozycji zamówienia (*OrderToDeliveryLag*).



Rys. 6.6. Tabela skumulowanych migawek *FactOrderFulfillmentAccum* dla faktu *Realizacja zamówienia* – wariant zaawansowany

Źródło: opracowanie własne.

Tabela skumulowanych migawek stanowi istotne uzupełnienie dotychczas omówionych typów tabel faktów, dostarczając danych do pełnej oceny zjawisk i ich dynamiki. Jest wykorzystywana do monitorowania procesów oraz optymalizacji ich przebiegu. Implementacja wszystkich trzech typów tabel faktów zapewnia zaspokojenie najbardziej złożonych wymagań biznesowych. Wykorzystanie tylko jednego lub nawet dwóch typów tabel faktów bardzo utrudniłoby, jeśli nie uniemożliwiłoby, przeprowadzenie kompleksowych analiz, których rezultaty mają wspomagać podejmowanie decyzji.

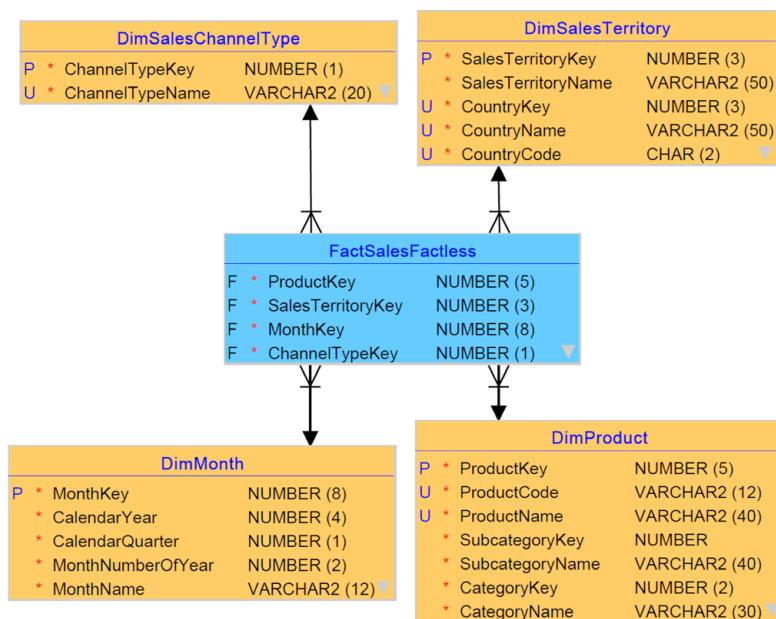
W tabeli 6.1 porównano wybrane cechy omówionych dotychczas tabel faktów. Zestawienie to pozwoli Czytelnikowi dostrzec różnice w przeznaczeniu poszczególnych typów i jednocześnie ułatwi zrozumienie wysokiej użyteczności zbioru tych tabel w tworzeniu pełnego opisu analizowanej rzeczywistości.

Tab. 6.1. Charakterystyka podstawowych rodzajów tabel faktów

Cecha	Transakcyjna	Okresowych migawek	Skumulowanych migawek
Ziarnistość	Jeden rekord opisuje elementarne zdarzenie	Jeden rekord opisuje wiele zdarzeń w przedziale czasu	Jeden rekord opisuje pełny cykl jednego zdarzenia
Wymiar czasu	Najniższy wymagany poziom szczegółowości czasowej	Wymagany analizą przedział czasu	Wielokrotne użycie wymiaru czasu w tabeli
Liczba wymiarów w strukturze tabeli	Większa liczba wymiarów niż w tabeli okresowych migawek	Mniejsza liczba wymiarów niż w tabeli transakcyjnej	Największa liczba wymiarów w porównaniu z innymi rodzajami tabel faktów
Miary	Opisują elementarne zdarzenie (wartości transakcyjne)	Opisują okresowe zdarzenie (agregaty wartości)	Opisują kolejne etapy jednego cyklu zdarzenia (wartości transakcyjne)
Rozmiar tabeli	Bardzo duży rozmiar, największy spośród wszystkich typów tabel faktów	Znacznie mniejszy niż tabeli transakcyjnej ze względu na mniejszą ziarnistość danych	Znacząco mniejszy niż tabeli transakcyjnej i okresowych migawek
Wydajność zapytań do tabeli	Niska ze względu na dużą ziarnistość danych (bardzo duża liczba wierszy)	Wysoka ze względu na użycie agregatów danych zapisanych w tabeli	Wysoka ze względu na opis zdarzenia w jednym rekordzie
Wstawianie rekordów	Możliwe	Możliwe	Możliwe
Modyfikacja rekordów	Brak możliwości	Brak możliwości	Możliwa na początek danego etapu cyklu zdarzenia
Usuwanie rekordów	Brak możliwości	Brak możliwości	Brak możliwości
Wzrost wielkości tabeli	Bardzo szybki i duży	Wolniejszy i mniejszy niż w tabeli transakcyjnej	Wolniejszy i mniejszy niż w tabelach transakcyjnej i okresowych migawek
Potrzeba tworzenia innych tabel faktów z agregatami danych	Bardzo wysoka ze względu na bardzo szczegółowy charakter składowanych danych. Agregacje możliwe na wyższym poziomie dowolnego wymiaru	Brak potrzeby lub niewielka, gdyż dane są już zagregowane na pewnym poziomie. Agregacje możliwe na wyższym poziomie wymiaru czasu	Niewielka potrzeba, gdyż zasadniczo dane powinny być kalkulowane na poziomie dnia. Agregacje możliwe na wyższym poziomie wymiaru czasu

Źródło: opracowanie własne.

Ostatnim podstawowym rodzajem tabeli faktu jest tzw. tabela faktu bez faktu. Jej struktura charakteryzuje się znaczną degeneracją w porównaniu z typowymi tabelami faktów, gdyż występują w niej wyłącznie kolumny będące kluczami obcymi do tabel wymiarów bądź innych tabel faktów. Natomiast nie ma w tej tabeli kolumn odpowiadających miarom. Taka sytuacja może wydawać się zaskakująca, gdyż zazwyczaj fakty charakteryzowane są za pomocą wielkości mierzalnych, umożliwiających ilościową ocenę modelowanego zjawiska czy zagadnienia. Jeśli jednak wziąć pod uwagę różnorodność potrzeb informacyjnych osób podejmujących decyzje, to można sobie wyobrazić przypadek, w którym nie jest istotna informacja opisana wynikiem kalkulacji wykonywanej na kolumnie miary (np. całkowita liczba sprzedanych sztuk produktu), lecz wyłącznie informacja, czy dane zjawisko nastąpiło (np. czy dany produkt został zakupiony). W takim przypadku bardzo pomocna okazuje się tabela faktu bez faktu. Na rysunku 6.7 przedstawiono przykład takiej tabeli o nazwie *FactSalesFactless*, który obrazuje wystąpienie w danym miesiącu (kolumna *MonthKey*) transakcji sprzedaży określonego produktu (kolumna *ProductKey*) dokonanej z danego kraju (kolumna *SalesTerritoryKey*) przy wykorzystaniu wybranego typu kanału sprzedaży (kolumna *ChannelTypeKey*).



Rys. 6.7. Tabela faktu bez faktu *FactSalesFactless* dla faktu *Sprzedaż*

Źródło: opracowanie własne.

Każda nowa operacja sprzedaży w systemie transakcyjnym spowoduje podczas realizacji procesu ETL wstawienie nowego rekordu do tej tabeli, który będzie potwierdzeniem istnienia powiązania pomiędzy określonymi danymi znajdującymi się w poszczególnych tabelach wymiarów. Konsekwencją tego jest brak możliwości utworzenia w tabeli klucza głównego składającego się ze wszystkich kluczy obcych, co było charakterystyczne dla uprzednio omówionych rodzajów tabel faktów.

Mimo braku kolumn reprezentujących miary struktura tabeli *FactSalesFactless* umożliwia uzyskanie odpowiedzi na wiele pytań analitycznych, wśród których można wymienić następujące:

- Ile transakcji sprzedaży zrealizowano w poszczególnych krajach z wykorzystaniem różnych kanałów sprzedażowych w podziale na poszczególne miesiące danego roku?
- Jaka jest różnica między liczbą transakcji w bieżącym miesiącu i liczbą transakcji w tym samym okresie ubiegłego roku?
- W których krajach było najwięcej transakcji sprzedaży określonego produktu w bieżącym roku?
- Który produkt najczęściej sprzedawano w ubiegłym miesiącu bieżącego roku?
- Które produkty sprzedano w kolejnych miesiącach poszczególnych lat w danych krajach dla każdego rodzaju kanału sprzedażowego?
- Które rodzaje kanałów sprzedażowych są najczęściej wykorzystywane w poszczególnych krajach?

Powyższe pytania analityczne nie wyczerpują bogatego ich zestawu, który można zdefiniować, wykorzystując fizyczny model danych przedstawiony na rysunku 6.7. Warto jednak zauważyć, że wszelkie kalkulacje wykorzystane do wygenerowania liczb, będących odpowiedzią na tak postawione pytania, zostaną wykonane na agregatach danych stworzonych z użyciem kolumn wymiarów oraz funkcji *COUNT*, *MIN*, *MAX* czy *AVG*.

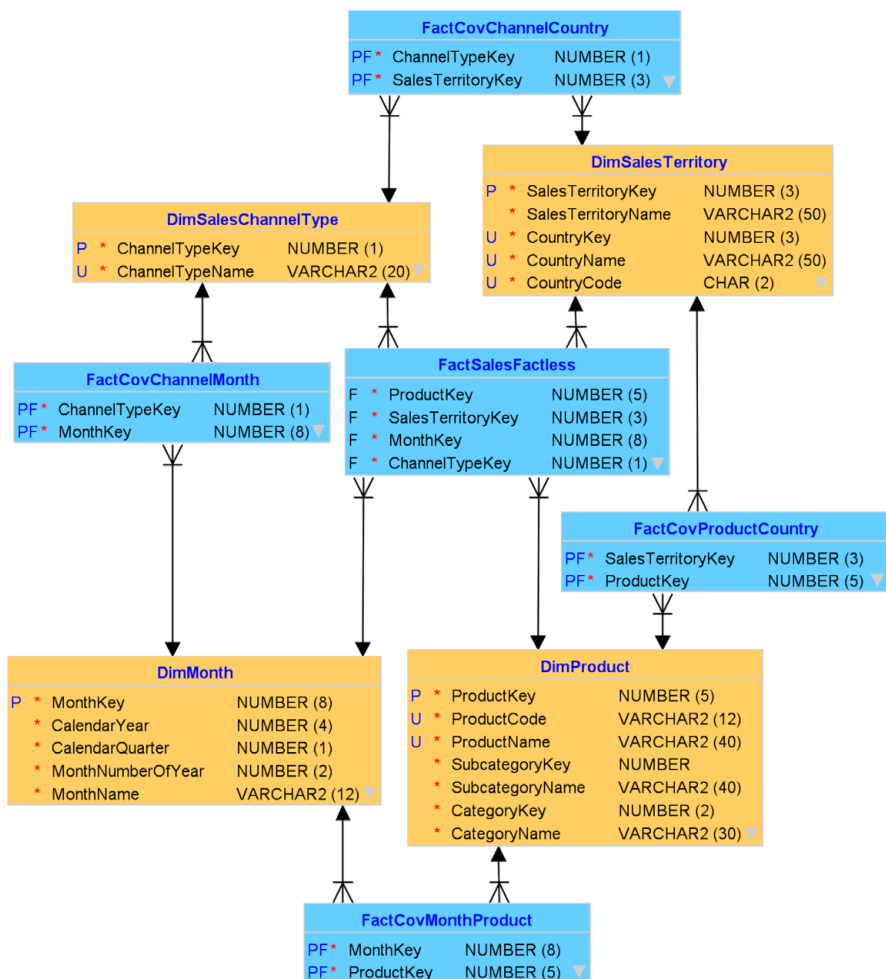
Przedstawiona dotychczas idea użycia tabeli faktu bez faktu pozwala identyfikować występowanie określonego zjawiska w rzeczywistości oraz opisać je dodatkowo za pomocą liczb o odpowiednim znaczeniu. Dlatego też tabela o takiej strukturze często nazywana jest tabelą śledzenia zdarzeń (ang. *event tracking table*). W literaturze przedmiotu [6] wskazuje się ponadto dodatkową korzyść ze stosowania tabel tego rodzaju. Polega ona na identyfikowaniu tych

zjawisk, które nie wystąpiły w analizowanej rzeczywistości, jakkolwiek istniało tego prawdopodobieństwo. Dla lepszego zobrazowania takiego przypadku warto najpierw postawić sobie następujące przykładowe pytania biznesowe:

- Których produktów nie sprzedano w kolejnych miesiącach danego roku?
- Które typy kanałów sprzedażowych nie są wykorzystywane przez klientów z poszczególnych krajów?
- Których produktów nie sprzedano w danych krajach w bieżącym roku?
- W którym miesiącu danego roku nie było żadnej transakcji sprzedaży w poszczególnych kanałach sprzedażowych?
- Których produktów dotychczas nie zamówiono w aktualnym miesiącu bieżącego roku, a sprzedano je w tym samym miesiącu roku ubiegłego?

Aby odpowiedzieć na powyższe pytania, nie wystarczy już posługiwać się tabelą faktu bez faktu o uprzednio przedstawionej strukturze – konieczne jest stworzenie dodatkowej tabeli, zawierającej rekordy reprezentujące iloczyn kartezyjański rekordów pochodzących z odpowiednich tabel wymiarów. Tak powstały zbiór danych określa się mianem tabeli pokrycia (ang. *factless coverage fact table*) [6]. Wykorzystanie obydwu typów tabel bez faktów umożliwi udzielenie odpowiedzi na powyższe pytania. Przykłady tabel pokrycia przedstawiono na rysunku 6.8. Zobrazowane tabele o nazwach *FactCovChannelCountry*, *FactCovChannelMonth*, *FactCovMonthProduct* i *FactCovProductCountry* zawierają wszystkie możliwe kombinacje wartości kluczy opisujących odpowiednie wymiary. Wykorzystanie tabel pokrycia i tabeli *FactSalesFactless* oraz wykonanie operacji różnicy tych zbiorów pozwala na wygenerowanie danych będących odpowiedzią na postawione powyżej pytania.

Przykładowe tabele pokrycia zdefiniowano z użyciem tylko dwóch tabel wymiarów. W praktyce spotyka się struktury bardziej złożone, w których liczba kluczy obcych w takiej tabeli jest znacznie większa w celu uwzględnienia bardziej złożonych przypadków analizy danych. W odróżnieniu od tabeli śledzenia zdarzeń tabela pokrycia może posiadać klucz główny, zdefiniowany przez wszystkie klucze obce w niej występujące, co zwiększy wydajność zapytań kierowanych do takich zasobów danych.

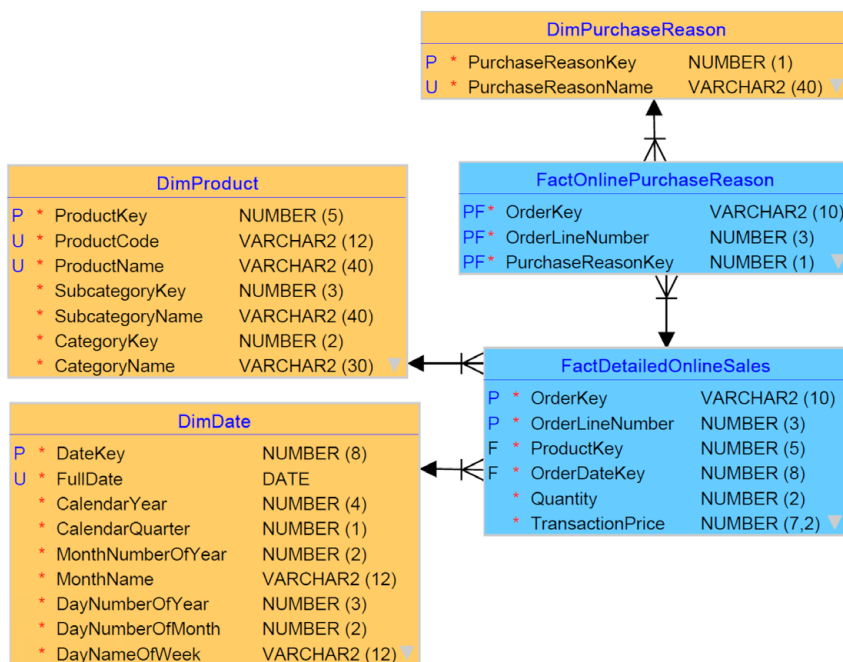


Rys. 6.8. Przykłady tabel pokrycia bez faktu dla faktu *Sprzedaż*

Źródło: opracowanie własne.

Wszystkie omówione dotychczas przykłady tabel faktów bez faktów miały jedną wspólną cechę: występowanie relacji pomiędzy taką tabelą i wyłącznie tabelami wymiarów. Istnieją jednak sytuacje, w których tabela faktu bez faktu pozostaje w relacji z typową tabelą faktu. Przypadek taki przedstawiono na rysunku 6.9. Tabela faktu bez faktu o nazwie *FactOnlinePurchaseReason* opisuje powód zakupu (kolumna *PurchaseReasonKey*) danego produktu (kolumna *OrderLineNumber*) znajdującego się w podanym zamówieniu (kolumna *OrderKey*). Relacja istniejąca pomiędzy obydwoima tabelami faktu umożliwia

uzyskanie tej informacji bez konieczności tworzenia związku między tabelami *FactDetailedOnlineSales* i *DimPurchaseReason*, gdyż do opisu znaczenia faktu powodów zakupu nie są potrzebne żadne kolumny miar.



Rys. 6.9. Przykład relacji między tabelą faktu bez faktu *FactOnlinePurchaseReason* i typową tabelą faktu *FactDetailedOnlineSales*

Źródło: opracowanie własne.

Niezależnie od typu tabeli bez faktów należy mieć świadomość pewnych wyzwań, jakie są związane z ich utrzymaniem w analitycznej bazie danych. Główne z nich to:

- nadmierny wzrost wielkości hurtowni danych (data mart), wynikający z bardzo szybkiego przyrastania rozmiaru tabeli;
- możliwe obniżenie wydajności zapytań kierowanych do tych tabel w przypadku dużej granulacji danych;
- trudności w zrozumieniu i interpretacji zawartości tabeli w wyniku braku miar określających charakter faktu.

Sprostanie tym wyzwaniom wymaga zastosowania kompresowania i partycjonowania tabeli, właściwego doboru rodzajów i struktur indeksów oraz

prowadzenia szczegółowej dokumentacji opisującej znaczenie i zawartość tych tabel.

W uzupełnieniu dotychczas przedstawionych informacji na temat struktury podstawowych typów tabel faktów warto jeszcze wspomnieć o możliwej obecności w nich kolumn opisujących tzw. wymiary zdegenerowane, czyli takie, dla których nie utworzono odrębnych tabel wymiarów. Tego rodzaju podejście zmniejsza rozmiar analitycznej bazy danych, jednak jest także źródłem pewnych wyzwań związanych z koniecznością zapewnienia spójności tych danych i ich jakości w różnych tabelach faktów. Więcej informacji na temat wymiarów zdegenerowanych Czytelnik znajdzie w podrozdziale 7.1.

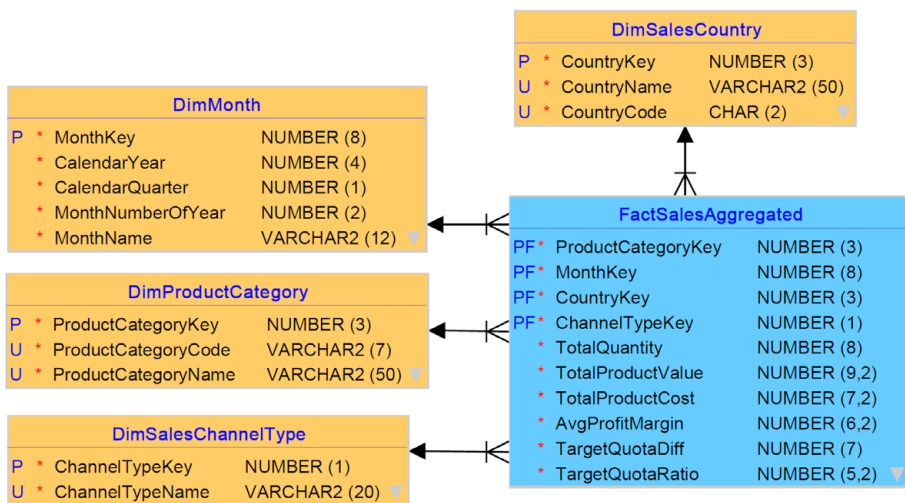
6.2. Dodatkowe rodzaje tabel faktów

Oprócz uprzednio wymienionych podstawowych typów tabel faktów w literaturze przedmiotu [15] wyróżnia się dodatkowe ich rodzaje, które mają umożliwić wyższą wydajność zapytań kierowanych do analitycznej bazy danych bądź też zapewnić wyższą efektywność wykorzystania zgromadzonych danych. Należą do nich:

- zagregowana tabela faktu (ang. *aggregate fact table*);
- skonsolidowana tabela faktu (ang. *consolidated fact table*);
- zgodna tabela faktu (ang. *conformed fact table*).

Zagregowana tabela faktu zawiera dane, które mogą powstać w wyniku agregacji danych pochodzących z transakcyjnej tabeli faktu według dowolnego pojedynczego wymiaru lub wielu wymiarów. Przykładem takiej tabeli jest omówiona uprzednio tabela okresowych migawek, zawierająca dane powstałe w wyniku grupowania wyłącznie według wymiaru czasu. Jeśli jednak zachodzi potrzeba dysponowania agregatami danych, opisanymi wieloma wymiarami, wówczas należy zastosować odrębną, dedykowaną tabelę faktu, w której każdy wiersz reprezentuje wynik kalkulacji wykonanej dla pojedynczej wartości atrybutu użytych wymiarów. W ten sposób powstanie tabela zawierająca rekordy odpowiadające każdej kombinacji wartości atrybutów takich wymiarów. Tego rodzaju konstrukcja, określana często mianem kostki OLAP, zapewnia znaczne skrócenie czasu wykonania operacji drążenia danych. Przykład zagregowanej tabeli faktu o nazwie *FactSalesAggregated*,

opisującej sprzedaż, przedstawiono na rysunku 6.10. Każdy rekord tej tabeli opisuje sprzedaż wszystkich produktów należących do jednej kategorii produktowej (kolumna *ProductCategoryKey*) w jednym miesiącu (kolumna *MonthKey*), w danym kraju (kolumna *CountryKey*), dla określonego rodzaju sprzedaży (kolumna *ChannelTypeKey*). Dane te mają charakter silnie zagregowany i zostały wygenerowane w procesie zasilenia hurtowni danych, co sprawia, że nie ma potrzeby wykonywania tych kalkulacji w czasie rzeczywistym z użyciem danych znajdujących się w odpowiedniej transakcyjnej tabeli faktów. Dzięki istnieniu odrębnej zagregowanej tabeli faktów czas pozyskania danych o wyższym poziomie ogólności jest znacznie krótszy niż w przypadku posługiwania się wyłącznie tabelą transakcyjną.



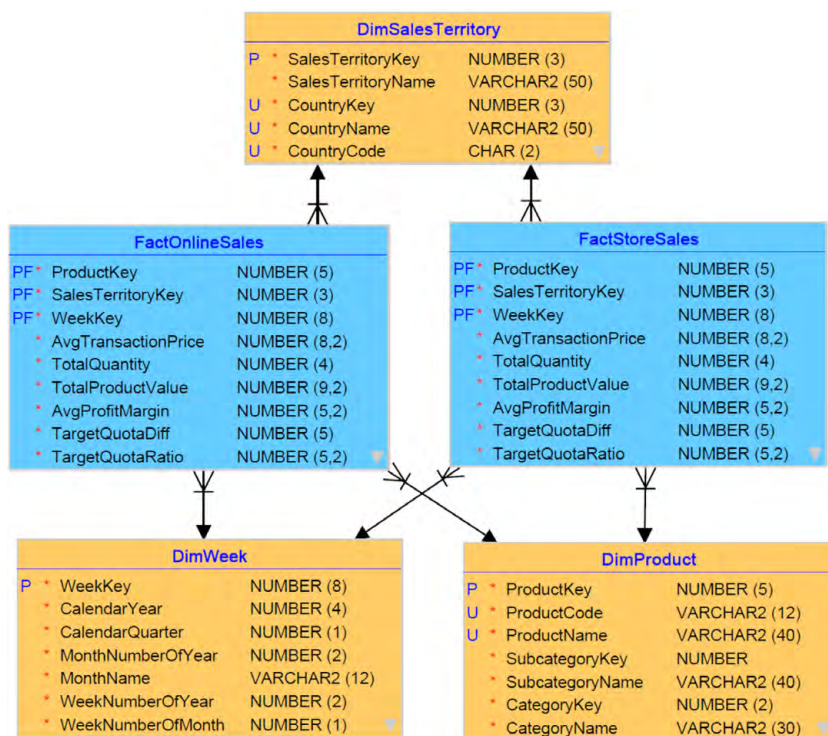
Rys. 6.10. Zagregowana tabela faktów *FactSalesAggregated* dla faktu *Sprzedaż*

Źródło: opracowanie własne.

W dotychczasowych rozważaniach dotyczących struktur tabel faktów przyjęto założenie, że każdy analizowany fakt jest odwzorowany za pomocą odrębnej tabeli zawierającej kolumny charakterystyczne wyłącznie dla danego faktu. Jednak w praktyce modelowania wielowymiarowego istnieją takie przypadki, w których wiele faktów jest opisanych tym samym zbiorem danych, tj. tymi samymi miarami i tymi samymi danymi o wymiarach. W przykładzie przedstawionym na rysunku 6.11 tabele faktów *FactOnlineSales*

i *FactStoreSales* zawierają dane charakteryzujące sprzedaż odpowiednio w kanałach online (mobilny, webowy) oraz w sklepach stacjonarnych. Można jednak zauważyć, że zestaw wymiarów i miar użytych do opisu tych faktów jest identyczny. Zachowanie tradycyjnego podejścia, tj. modelowanie odrębnych faktów różnymi tabelami, spowoduje nadmierny wzrost wielkości analitycznej bazy danych ze względu na duplikowanie danych gromadzonych w różnych tabelach faktów. W takiej sytuacji R. Kimball zaleca zastosowanie skonsolidowanej tabeli faktu, która zastąpi wiele tabel faktów [15]. Utworzenie takiej tabeli będzie możliwe, jeśli:

- poziom szczegółowości opisu różnych faktów jest identyczny dla wszystkich konsolidowanych tabel;
- konsolidowane tabele zawierają te same zestawy miar opisujących dany fakt.

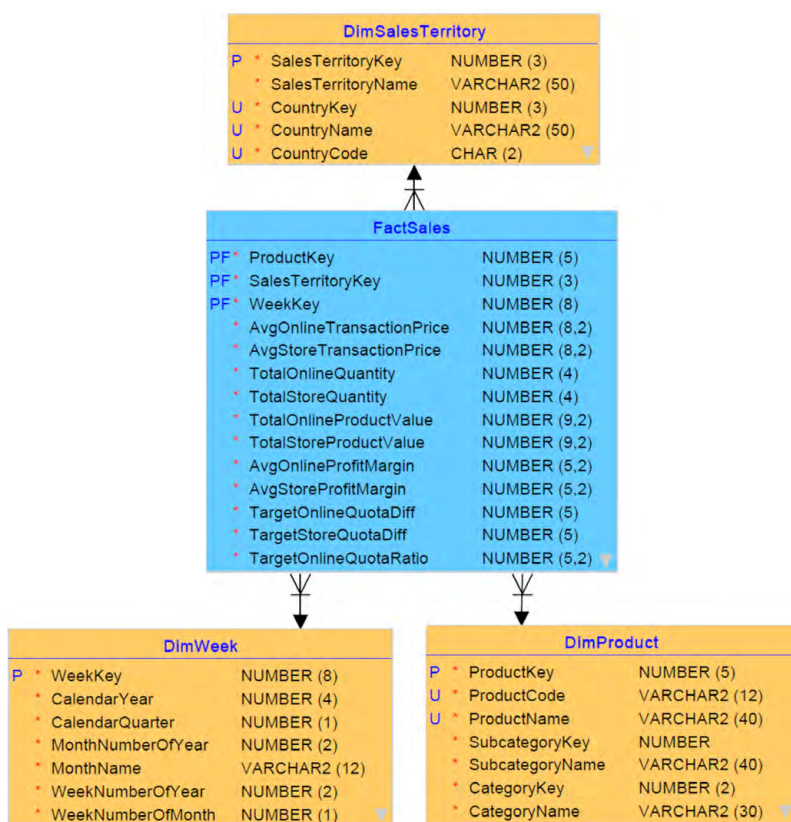


Rys. 6.11. Modelowanie odrębnych faktów za pomocą różnych tabel faktów

Źródło: opracowanie własne.

W rezultacie konsolidacji struktura nowej tabeli będzie składała się z tych samych kolumn opisujących wymiary, natomiast wzrośnie liczba kolumn

odpowiadających miarom. Przykład skonsolidowanej tabeli faktu, po przeprowadzeniu transformacji tabel z rysunku 6.11, zaprezentowano na rysunku 6.12. Definicja nowej tabeli faktu *FactSales*, w porównaniu z pierwotnymi tabelami *FactOnlineSales* i *FactStoreSales*, obejmuje teraz odrębne miary dla sprzedaży online (np. *TotalOnlineQuantity*, *TotalOnlineProductValue*) i odrębne miary dla sprzedaży stacjonarnej (np. *TotalStoreQuantity*, *TotalStoreProductValue*). Dodatkową zaletą tego rozwiązania jest także skrócenie czasu pozyskania danych w przypadku konieczności porównywania wartości miar o tym samym znaczeniu dla różnych faktów, gdyż wielkości te znajdują się w jednym rekordzie tej samej tabeli zamiast w odrębnych tabelach.



Rys. 6.12. Tabela *FactSales* jako przykład skonsolidowanej tabeli faktu

Źródło: opracowanie własne.

Ostatnim prezentowanym rodzajem tabeli faktu jest tzw. zgodna tabela, zawierająca dane opisujące pojedynczy fakt lub wiele faktów występujących

w różnych obszarach analizy. Przykładem zgodnej tabeli faktu jest tabela *FactOnlinePromoSales*, która może wystąpić w data mart zarówno opisującej działania sprzedażowe, jak i dotyczącej obszaru marketingu (rys. 6.13). Innym przykładem zgodnej tabeli jest tabela *FactComplaintsHandling*, istniejąca zarówno w data mart wspomagającej decyzje w obszarze zarządzania relacjami z klientami (CRM – ang. *Customer Relationship Management*), jak i związanej z analizą kosztów. Obydwie przykładowe tabele dopasowane są jednocześnie tabelami okresowych migawek. Tabela *FactOnlinePromoSales* opisuje sprzedaż online w różnych kanałach sprzedaży (kolumna *OnlineChannelKey*) promocyjnych produktów (kolumna *ProductKey*) w każdym dniu (kolumna *DateKey*) określonej promocji (kolumna *PromotionKey*) w poszczególnych krajach (kolumna *SalesTerritoryKey*). Z kolei tabela *FactComplaintsHandling* zawiera dane zagregowane do poziomu miesiąca (kolumna *MonthKey*), charakteryzujące obsługę reklamacji dotyczących określonego produktu (kolumna *ProductKey*), złożonych w danym kraju (kolumna *SalesTerritoryKey*).

FactOnlinePromoSales		
P	* PromotionKey	NUMBER (7)
P	* DayKey	NUMBER (8)
P	* SalesTerritoryKey	NUMBER (3)
P	* ProductKey	NUMBER (5)
P	* OnlineChannelKey	NUMBER (2)
	* NumOfTransactions	NUMBER (7)
	* TotalPromoSalesValue	NUMBER (8,2)
	* TotalQuantity	NUMBER (7)
	* NumOfCustomers	NUMBER (5)
	* PromotionDayNum	NUMBER (2)

FactComplaintsHandling		
P	* ProductKey	NUMBER (5)
P	* SalesTerritoryKey	NUMBER (3)
P	* MonthKey	NUMBER (6)
	* TotalNumOfComplaints	NUMBER (5)
	* TotalNumOfAcceptComplaints	NUMBER (5)
	* AvgTotalComplaintHandlingCost	NUMBER (8,2)
	* AvgShipmentCost	NUMBER (7,2)
	* AvgProductRepairCost	NUMBER (7,2)
	* AvgNumOfDaysComplaintHandling	NUMBER (2)

Rys. 6.13. Przykłady zgodnych tabel faktu

Źródło: opracowanie własne.

Charakterystyczną cechą zgodnej tabeli faktu jest obecność kolumn reprezentujących wymiary i miary, których znaczenie i wartości są takie same we wszystkich modelowanych obszarach analizy. W praktyce oznacza to, że ta sama tabela faktu jest wykorzystywana wielokrotnie w strukturach danych różnych data marts, co zapewnia skrócenie czasu realizacji procesów ETL oraz zmniejszenie wielkości bazy danych, gdyż dane dotyczące tego samego zagadnienia ładowane są tylko do jednej tabeli faktu, a nie do wielu.

Podsumowując rozważania dotyczące różnych tabel faktów, ich przeznaczenia i wynikającej z tego ich struktury, należy zauważyć, że w tej samej

analitycznej bazie może wystąpić jednocześnie kilka rodzajów tabel faktów, gdyż każda z nich dostarcza danych do wykreowania innego spojrzenia na analizowane zagadnienia, jednocześnie zapewniając redukcję czasu uzyskania odpowiedzi na zapytania o różnym stopniu szczegółowości.

6.3. Zadania do samodzielnego rozwiązania

Wykorzystując model danych data mart typu konstelacja faktów, zbudowany w zadaniu 2 w podrozdziale 5.4, proszę wykonać następujące zadania:

1. Dokonać rozbudowy istniejącego modelu danych poprzez utworzenie nowej transakcyjnej tabeli faktu opisującej koszty związane z usunięciem awarii i uszkodzenia roweru. Ziarnistość faktu należy przyjąć na poziomie pojedynczej awarii lub uszkodzenia danego roweru. Wykorzystać istniejące już tabele wymiarów do zdefiniowania relacji między nimi a nową tabelą faktu.
2. Utworzyć tabelę faktu okresowych migawek, która umożliwi ocenę zyskowności usług wynajmu rowerów na poziomie kolejnych miesięcy poszczególnych lat. Zdefiniowane kolumny miar powinny reprezentować wszystkie wielkości niezbędne do określenia zyskowności tego rodzaju usług. Wykorzystać istniejące już tabele wymiarów do zdefiniowania relacji między nimi a nową tabelą faktu.
3. Utworzyć zagregowaną tabelę faktu, umożliwiającą analizę zyskowności usług wynajmu rowerów z uwzględnieniem takich wymiarów, jak czas, stacja rowerowa i rower. Dla wymiaru czasu proszę przyjąć agregację na poziomie miesiąca, dla wymiaru stacja – agregację na poziomie miasta, a dla wymiaru rower – agregację na poziomie typu roweru. Zdefiniowane kolumny miar powinny reprezentować wszystkie wielkości niezbędne do określenia zyskowności usług tego rodzaju. Wykorzystać istniejące już tabele wymiarów do zdefiniowania relacji między nimi a nową tabelą faktu.

7. Modelowanie wymiarów

Koniecznym uzupełnieniem dotychczas opisanych działań projektowych, zmierzających do zbudowania analitycznej bazy danych, jest zamodelowanie wymiarów zidentyfikowanych w modelu koncepcyjnym data mart czy hurtowni danych. Uszczegółowione w modelu logicznym opisy wymiarów należy teraz przetransformować do postaci tabel wymiarów, a atrybuty wymiarów – na odpowiednie kolumny tych tabel.

7.1. Rodzaje tabel wymiarów

Sposób definiowania tabel wymiarów i ich struktura dla poszczególnych wymiarów będą w głównej mierze wynikać z rodzaju wybranego modelu danych. Jednak, niezależnie od modelu, można wskazać pewne uniwersalne reguły tworzenia tabel wymiarów:

- każda tabela wymiaru posiada klucz główny;
- klucz główny tabeli powinien być kluczem surogatowym, tzn. kluczem sztucznym generowanym wyłącznie na potrzeby identyfikacji wierszy;
- jeśli to możliwe, tabela wymiaru powinna zawierać klucz naturalny, niekiedy nazywany kluczem biznesowym (np. NIP klienta biznesowego, kod EAN produktu);

- klucz główny tabeli nie powinien być kluczem głównym pochodzącym z systemu źródłowego;
- kolumny tabeli muszą umożliwiać opis atrybutów wymiaru; jeśli to konieczne, dla jednego atrybutu można utworzyć wiele kolumn;
- struktura tabeli powinna uwzględniać możliwość gromadzenia danych wykorzystywanych w przyszłych działaniach analitycznych bez konieczności wykonywania czasochłonnych transformacji danych, np. w narzędziach business intelligence.

Definiując klucz główny tabeli wymiaru bardzo często wykorzystuje się tzw. klucz surogatowy, który może być również stosowany w relacyjnym modelu danych. Jednak w przypadku hurtowni danych i data mart jego obecność i znaczenie mają całkowicie odmienny charakter. Klucz ten należy stosować, jeśli dane w tabeli wymiaru pochodzą z wielu źródeł transakcyjnych, w których użyto różnych kluczy głównych do opisu tego samego obiektu, albo gdy potrzebna jest wiedza o historii zmian wartości opisujących wymiar. Klucz surogatowy musi być kluczem prostym, tzn. definiowanym wyłącznie za pomocą jednej kolumny tabeli.

W opisie wymiaru niekiedy konieczne jest użycie, oprócz klucza surogatowego, również tzw. klucza biznesowego, którego wartość:

- umożliwia jednoznaczne identyfikowanie i interpretowanie przez użytkownika końcowego każdej instancji określonego wymiaru;
- opisuje w sposób unikalny wystąpienie danej instancji wymiaru we wszystkich obszarach analizy, w których wymiar ten jest używany;
- jest niezmienna w całym cyklu życia danej instancji wymiaru.

Przykładem klucza biznesowego jest REGON, charakteryzujący podmioty gospodarki narodowej w Polsce, NIP (Numer Identyfikacji Podatkowej), będący cechą każdego podatnika w Polsce, bądź też kod EAN, opisujący produkty sprzedawane w Europie.

Przykładową strukturę tabeli dla wymiaru czasu, opisującego większość analizowanych zagadnień, przedstawiono na rysunku 7.1.

DimTime		
P	* TimeKey	NUMBER (15)
U	* FullDateTime	TIMESTAMP
	* Hour	NUMBER (2)
	* TimeOfDay	VARCHAR2 (10)
	* FullDate	DATE
	* DayNumberOfYear	NUMBER (3)
	* DayNumberOfMonth	NUMBER (2)
	* DayNumberOfWeek	NUMBER (1)
	* DayNameOfWeek	VARCHAR2 (12)
	* MonthNumberOfYear	NUMBER (2)
	* FullMonthName	VARCHAR2 (12)
	* ShortMonthName	VARCHAR2 (3)
	* CalendarQuarterNumber	NUMBER (1)
	* QuarterName	VARCHAR2 (2)
	* CalendarYear	NUMBER (4)

Rys. 7.1. Struktura przykładowej tabeli *DimTime* opisującej wymiar czasu

Źródło: opracowanie własne.

Tabela o takiej strukturze doskonale nadaje się do bardzo szczegółowego opisu, z punktu widzenia czasu, różnych działań wykonywanych np. przez klienta korzystającego z usług bankowości online, zdarzeń zachodzących w lokalnej sieci komputerowej czy procesów związanych z realizacją zamówień klientów. Uwzględniono w niej atrybut *FullDateTime* wymiaru czasu, umożliwiający opis zagadnienia z dokładnością do minut czy nawet sekund, a także atrybuty pozwalające na analizy o wyższym stopniu ogólności, tj. miesięcznym, kwartalnym lub rocznym. Zastosowanie wielu kolumn do opisu tego samego poziomu agregacji danych jest zabiegiem celowym, gdyż pozwala identyfikować ten sam atrybut wymiaru na różne sposoby w zależności od bieżących uwarunkowań, związanych głównie z prezentacją tych danych. Dla przykładu: atrybut *Month* może być reprezentowany za pomocą pełnej nazwy miesiąca (*FullMonthName*), skróconej nazwy miesiąca (*ShortMonthName*) bądź też numeru miesiąca w roku kalendarzowym (*MonthNumberOfYear*).

Struktura tabeli wymiaru w głównej mierze zależy od modelu danych, jaki został wybrany do opisu analizowanego zagadnienia. Jak już uprzednio wspomniano, tabele te mogą mieć postać znormalizowaną, częściowo znormalizowaną lub zdenormalizowaną. Umiejętność właściwego doboru poprawnej struktury i dostrzeżenie możliwości wielokrotnego wykorzystania tych samych danych opisujących wymiar w kontekście różnych faktów pozwala znacząco ograniczyć rozmiar bazy danych.

W literaturze przedmiotu wyróżnia się następujące rodzaje tabel wymiarów [14]:

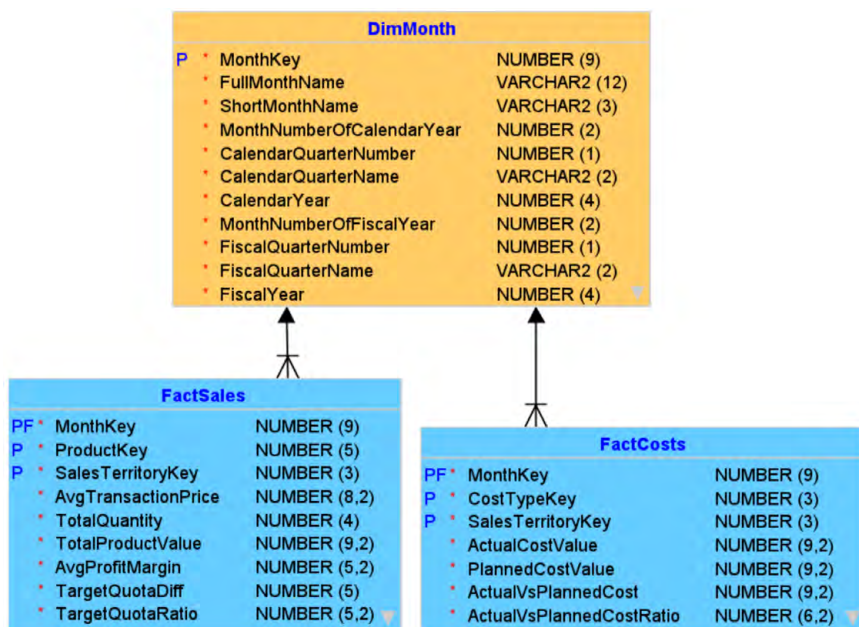
- zgodne (ang. *conformed dimension*), określane również jako dopasowane;
- skurczone (ang. *shrunk dimension*), nazywane także zmniejszonymi;
- odgrywające rolę (ang. *role-playing dimension*);
- zewnętrzne (ang. *outrigger dimension*);
- zdegenerowane (ang. *degenerate dimension*);
- wymienne (ang. *swappable dimension*);
- śmieciowe (ang. *junk dimension*);
- pomostowe (ang. *bridge dimension*).

Wymienione powyżej rodzaje tabel wymiarów nie klasyfikują ich w sposób rozłączny. Oznacza to, że tabela jednego typu w określonych warunkach może być również tabelą innego rodzaju.

Pierwszym prezentowanym typem tabeli wymiaru będzie zgodna tabela, posiadająca taką strukturę, która umożliwia jej zastosowanie do opisu wielu tabel faktów wykorzystywanych w analizie tego samego lub różnych zagadnień. Dane znajdujące się w zgodnej tabeli zawsze mają takie same znaczenie, niezależnie od rodzaju zagadnienia, do opisu którego zostały użyte. Najważniejszą zaletą posługiwania się tego rodzaju tabelami jest zgromadzenie w jednym miejscu wszystkich danych opisujących wymiar, co zapewnia spójność opisu analizowanych zagadnień, gdyż w każdym z obszarów używane dane pochodzą z tej samej tabeli. Ponadto jakakolwiek modyfikacja tych danych, szczególnie w kontekście użycia technik modelowania wymiarów wolnozmiennych, odbywa się tylko w jednym miejscu, co istotnie upraszcza definicję procesów ETL i czas ich realizacji.

Zdefiniowanie poprawnej struktury zgodnej tabeli wymiaru nie jest łatwym zadaniem, gdyż wymaga konsensusu dotyczącego nazewnictwa atrybutów i ich znaczenia we wszystkich obszarach działania organizacji, w których dany wymiar będzie wykorzystany do analizy zagadnień. Jednak wypracowanie pewnego wspólnego zestawu atrybutów umożliwi wszystkim zainteresowanym stronom efektywne wykonywanie wielopłaszczyznowych analiz i poprawną interpretację ich wyników za pomocą danego wymiaru.

Przedstawiona na rysunku 7.1 tabela *DimTime* może być przykładem zgodnej tabeli, ale jej użyteczność ogranicza się tylko do tych data marts, w których wymagany jest bardzo szczegółowy, z dokładnością do minut, opis analizowanych zagadnień. Tabela ta nie może być wykorzystana do analizowania faktu związanego z kosztami funkcjonowania firmy, gdyż te zazwyczaj rozliczane są w okresach miesięcznych. Jeśli jednak przeprowadzi się denormalizację tabeli *DimTime* tak, aby wydzielić z niej miesiące do nowej tabeli *DimMonth*, wówczas ta tabela stanie się już zgodną tabelą wymiaru.



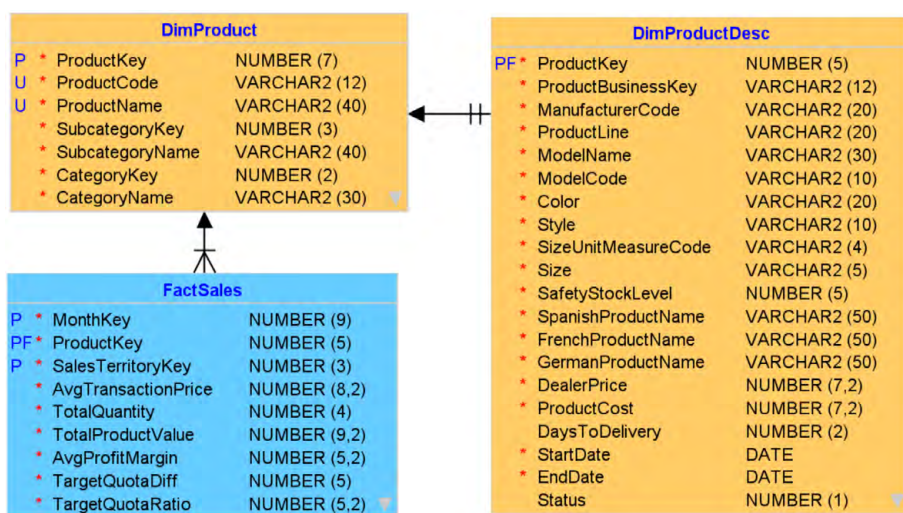
Rys. 7.2. Tabela *DimMonth* jako przykład zgodnej tabeli wymiaru

Źródło: opracowanie własne.

Na rysunku 7.2 przedstawiono przykład struktury takiej tabeli. Zdefiniowanie w tabeli *DimMonth* kolumn opisujących miesiąc, kwartał i rok fiskalny zwiększa możliwość jej użycia w opisie tych wszystkich faktów, które wymagają uwzględnienia aspektów związanych z rozliczeniami z fiskusem. W takiej postaci tabela ta może być wykorzystana w analizie zarówno działań sprzedażowych, marketingowych, produkcyjnych, logistycznych, jak i związanych z zapewnieniem bezpieczeństwa infrastruktury informatycznej. Innym

przykładem zgodnej tabeli wymiaru mogą być tabele opisujące produkt, usługę czy klienta. Każdy z tych wymiarów musi być charakteryzowany takim zestawem atrybutów który pozwala na odwoływanie się do nich jako kontekstu analizowanych danych we wszystkich obszarach funkcjonowania firmy.

Innym rodzajem tabeli wymiaru jest tzw. skurczona tabela (zmniejszona). Zawiera ona podzbiór kolumn pochodzących z pierwotnej tabeli wymiaru i może być powiązana zarówno z inną tabelą wymiaru (jak w przypadku modelu płatka śniegu), jak i z tabelą faktu zawierającą dane o wyższym stopniu agregacji. Zaprezentowana uprzednio tabela *DimMonth* (rys. 7.2) jest jednym z przykładów skurczonej tabeli, gdyż wydzielono ją z pierwotnej tabeli *DimTime* (rys. 7.1). Jednym z powodów takiej ekstrakcji pewnych kolumn do zmniejszonej tabeli jest uproszczenie struktury tabeli wymiaru, w której mogą znaleźć się dziesiątki lub nawet setki kolumn. Ponadto odrębna tabela wymiaru może być wykorzystana jako dzielony zbiór danych pomiędzy różne data marts bądź też do opisu zagregowanych tabel faktów lub tabel okresowych migawek (w przypadku wymiaru czasu) w tej samej data mart.



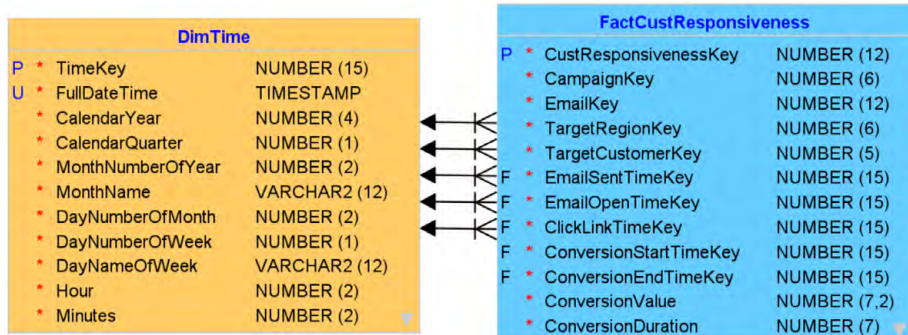
Rys. 7.3. Tabela *DimProductDesc* jako przykład zmniejszonej tabeli wymiaru

Źródło: opracowanie własne.

Innym przykładem zmniejszonej tabeli będzie tabela *DimProductDesc*, której kolumny szczegółowo opisują poszczególne produkty (rys. 7.3). W tym przypadku wydzielenie szczegółowych cech produktu do nowej tabeli wynikało z ich dużej liczby i braku konieczności utrzymywania szerokiej tabeli *DimProduct*, co ma istotne znaczenie szczególnie w aspekcie optymalnego wykorzystania pamięci operacyjnej podczas wykonywania zapytań do takiej tabeli. W wyniku oddzielenia pewnych danych od siebie w tabeli *DimProduct* znajdują się najważniejsze dane kontekstowe w analizie faktu sprzedażowego. Jeśli znajdzie konieczność dodatkowego uzupełnienia opisu wyników analizy o dane szczegółowe, wówczas zostaną one pobrane z tabeli *DimProductDesc*.

Na podstawie obydwu przedstawionych przykładów można zauważyć, że obecność skurczonych tabel jest charakterystyczna dla modelu płątka śniegu lub modelu konstelacji faktów. Ich obecność znacząco uprasza zarządzanie danymi, szczególnie w przypadku okresowego występowania zmian wartości atrybutów opisujących dany wymiar.

Kolejnym rodzajem tabeli wymiaru jest tabela określana jako odgrywająca rolę. W tym przypadku pojęcie roli odnosi się do określonego znaczenia, w jakim dany wymiar jest używany w opisie faktu. Najlepszym przykładem tabeli tego rodzaju jest *DimTime*, przedstawiona na rysunku 7.4. Prezentowany przykład opisuje transakcyjną tabelę faktu responsywności klienta (kolumna *TargetCustomerKey*) na pojedynczy e-mail (kolumna *EmailKey*), wysłany do niego w ramach danej kampanii e-mailingowej (kolumna *CampaignKey*) do regionu określonego przez wartość w kolumnie *TargetRegionKey*. Poszczególne role – takie jak czas wysłania e-maila, czas jego otwarcia, czas kliknięcia w link zamieszczony w e-mailu, czas rozpoczęcia i zakończenia konwersji – zdefiniowane są odpowiednio w kolumnach *EmailSentTimeKey*, *EmailOpenTimeKey*, *ClickLinkTimeKey*, *ConversionStartTimeKey* i *ConversionEndTimeKey* tabeli faktu i opisane danymi pochodzącymi z tabeli *DimTime*. Dlatego też pomiędzy tabelą wymiaru *DimTime* i tabelą faktu *FactCustResponsiveness* istnieje wiele relacji, z których każda skojarzona jest z pojedynczą rolą.



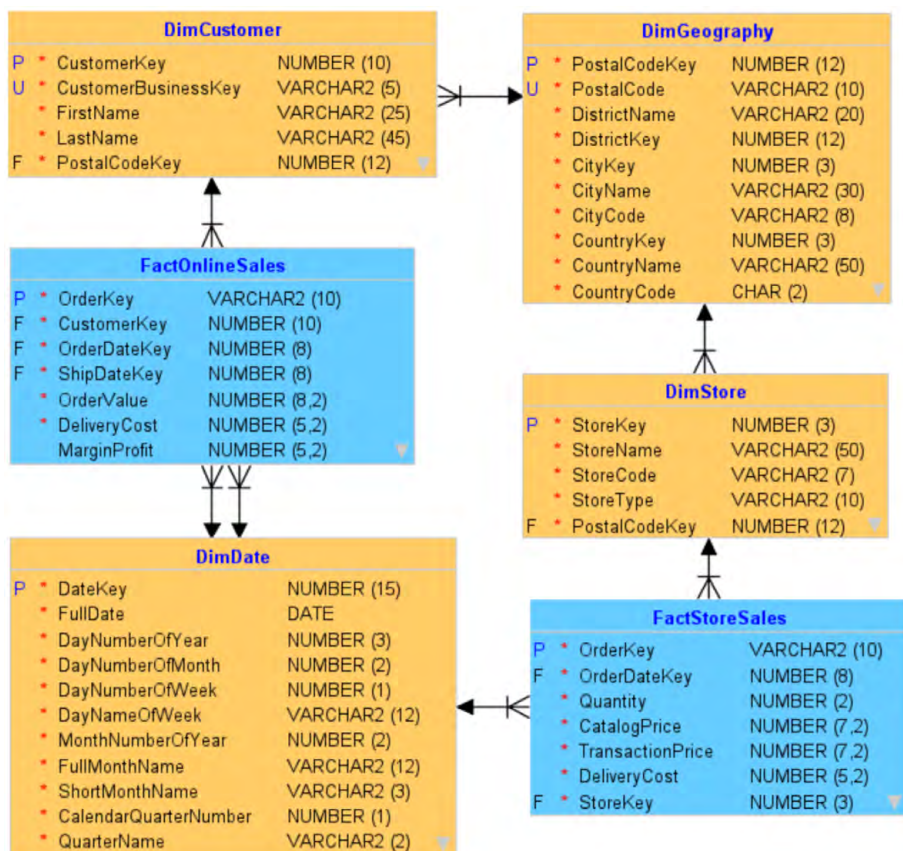
Rys. 7.4. Tabela *DimTime* jako przykład tabeli odgrywającej rolę

Źródło: opracowanie własne.

Główną zaletą posługiwania się tabelą odgrywającą rolę jest możliwość bardzo elastycznego wzbogacania opisu każdej roli poprzez zdefiniowanie dodatkowej kolumny w tabeli wymiaru. Jeśli w przedstawionym przykładzie zajdzie konieczność rozszerzenia opisu czasu o porę dnia, wówczas wystarczy dodać pojedynczą kolumnę do tabeli *DimTime* i wszystkie role będą opisane tym samym nowym atrybutem. Inną zaletą jest wyeliminowanie konieczności tworzenia odrębnych tabel dla poszczególnych ról, co powoduje zmniejszenie rozmiaru bazy danych. Gromadzenie danych w jednej tabeli wymiaru, wykorzystywanej w różnych kontekstach, upraszcza także proces ETL zasilający tę tabelę i skraca czas jego realizacji. Korzystnym aspektem istnienia takiej tabeli w bazie danych jest również możliwość jej bezpośredniego wykorzystania w przypadku pojawienia się w tabeli faktu kolumny opisującej nową rolę. W takiej sytuacji nie ma potrzeby tworzenia kolejnej tabeli wymiaru dla tej kolumny, lecz wystarczy zdefiniować nową relację do tabeli odgrywającej rolę.

Podczas modelowania wymiarów w analitycznej bazie danych pojawia się niekiedy konieczność opisanie pewnych ich atrybutów za pomocą dodatkowych tabel, określanych jako zewnętrzne. Tabele te mogą mieć strukturę znormalizowaną lub częściowo znormalizowaną, w zależności od sposobu ich wykorzystania w opisie innych wymiarów. W odróżnieniu od zgodnych tabel wymiarów zewnętrzne tabele mogą być powiązane tylko z innymi tabelami wymiarów, ale nigdy nie pozostają w relacji z żadną tabelą faktu. Na rysunku 7.5 przedstawiono fizyczny model danych, który zawiera tabelę *DimGeography* pełniącą funkcję zewnętrznej tabeli wymiaru. Zawiera ona szczegółowe dane dotyczące każdego kodu pocztowego. Tabela ta pozostaje w relacji wyłącznie z tabelami

DimCustomer oraz *DimStore*, opisując szczegółowo odpowiednio lokalizację klienta oraz sklepu.



Rys. 7.5. Tabela *DimGeography* jako przykład zewnętrznej tabeli wymiaru

Źródło: opracowanie własne.

Główną zaletą stosowania zewnętrznej tabeli, pozostającej w relacji z innymi tabelami wymiarów, jest wyeliminowanie redundancji danych oraz uproszczenie i skrócenie czasu realizacji procesu ETL, który zasila tę tabelę. Nie bez znaczenie pozostaje także łatwość rozszerzenia jej struktury bez konieczności modyfikowania innych powiązanych tabel. W przedstawionym przykładzie (rys. 7.5) dodanie w tabeli *DimGeography* kolumn *AreaName* i *AreaCode*, charakteryzujących lokalizację kraju na świecie, umożliwi wzbogacenie opisu wyników analiz sprzedażowych, lecz nie wymusi jakichkolwiek zmian w tabelach *DimCustomer* i *DimStore*. Równie istotna wydaje się łatwość

wprowadzania i rejestrowania zmian wartości atrybutów wymiaru geograficznego, szczególnie w przypadku stosowania wymiaru wolnozmiennego typu 2 i 3. Ponadto tabela zewnętrzna stanowi obowiązkowy element modelu danych podczas implementacji takiego wymiaru typu 5 (więcej informacji o wymiarach wolnozmiennych Czytelnik znajdzie w podrozdziale 7.2).

Przedstawione dotychczas rodzaje tabel wymiarów wykorzystywane są do przechowywania wielu wartości poszczególnych ich atrybutów. Konieczność wydzielania tych danych do odrębnej tabeli wynikała z dużej liczby kolumn (odpowiadających atrybutom) oraz wierszy (reprezentujących wartości atrybutów). Jednak niektóre fakty są opisane takimi danymi, które nie mają ani charakteru miary, ani natury typowego wymiaru. Przykładami tego są numer referencyjny przelewu bankowego, identyfikator transakcji płatniczej czy numer przesyłki zawierającej zamówione produkty. Gdyby więc zastosować dotychczasowe podejście polegające na utworzeniu odrębnej tabeli wymiaru, to zawierałaby ona wyłącznie jedną kolumnę i byłaby w relacji jeden-do-jednego z tabelą faktu. Rozwiązanie takie nie przynosi żadnych korzyści, gdyż nie jest optymalne z punktu widzenia wydajności zapytań i nie jest zrozumiałe dla użytkownika końcowego. W takiej sytuacji dokonuje się włączenia opisu tego wymiaru do tabeli faktu jako dodatkowej kolumny. Konsekwencją takiego podejścia będzie posługiwanie się terminem wymiaru zdegenerowanego, a nie zdegenerowanej tabeli wymiaru, gdyż w rzeczywistości tabela taka nie istnieje.

Na rysunku 7.6 przedstawiono przykłady dwóch tabel faktów zawierających wymiary zdegenerowane. W tabeli *FactOnlineSales* są one reprezentowane przez kolumny *OrderRefNumber* (numer zamówienia), *PaymentRefNumber* (identyfikator płatności), *InvoiceNum* (numer faktury) i *ParcelRefNumber* (identyfikator przesyłki z zamówionymi produktami), natomiast w tabeli *FactClaimSettlement* (likwidacja szkody) – przez *ClaimRefNumber* (numer zgłoszenia szkody) oraz *CompensationRefNumber* (identyfikator wypłaty odszkodowania). Wartości w tych kolumnach są unikatowe i jednoznacznie opisują dany fakt.

Występowanie wymiarów zdegenerowanych jest charakterystyczne dla faktów o bardzo dużej szczegółowości (np. pojedyncza pozycja zamówienia) bądź też faktów o niskim poziomie agregacji (np. całe zamówienie). Wymiary tego typu, choć nie reprezentują sobą żadnej wartości biznesowej, są bardzo użyteczne szczególnie w operacjach drążenia danych, kiedy trzeba dotrzeć do szczegółów przechowywanych w systemach źródłowych.

FactOnlineSales		
P *	OrderKey	VARCHAR2 (10)
U *	OrderRefNumber	VARCHAR2 (10)
	CustomerKey	NUMBER (10)
	OrderDateKey	NUMBER (8)
	ShipDateKey	NUMBER (8)
	OrderValue	NUMBER (8,2)
	DeliveryCost	NUMBER (5,2)
	MarginProfit	NUMBER (5,2)
U *	PaymentRefNum	VARCHAR2 (25)
U *	InvoiceNum	VARCHAR2 (10)
U *	ParcelRefNum	VARCHAR2 (26)

FactClaimSettlement		
P *	ClaimSettlementKey	VARCHAR2 (15)
U *	ClaimRefNumber	VARCHAR2 (10)
	CustomerKey	NUMBER (10)
	PolicyNumber	VARCHAR2 (10)
	IncidentTypeKey	NUMBER (6)
	ClaimRegDateKey	NUMBER (8)
	CompensationDateKey	NUMBER (8)
	CompensationAmount	NUMBER (10,2)
U *	CompensationRefNumber	VARCHAR2 (25)

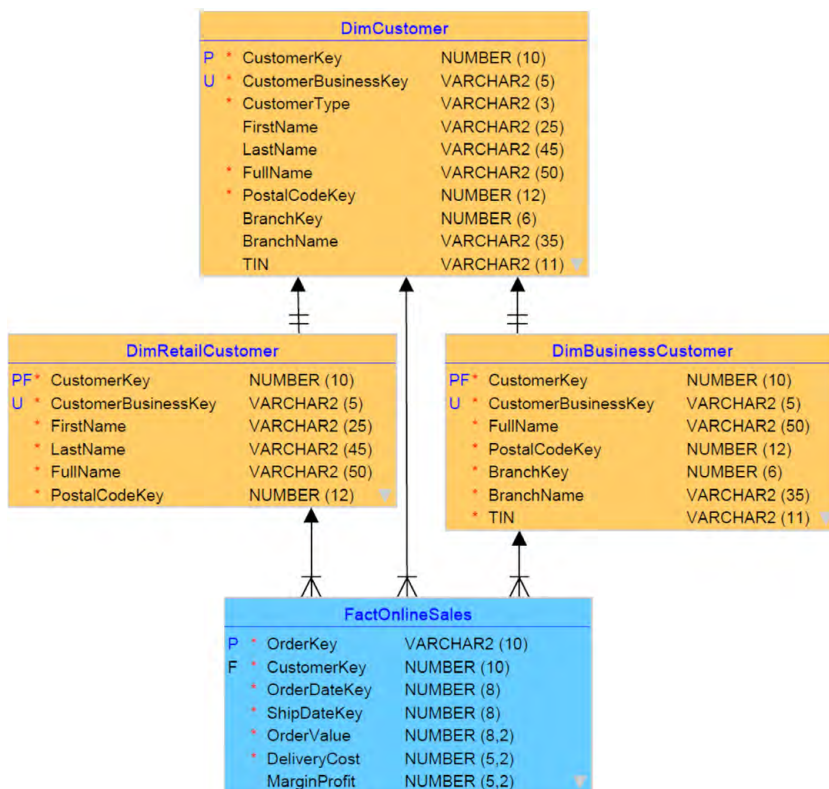
Rys. 7.6. Przykłady tabel faktów zawierających wymiary zdegenerowane

Źródło: opracowanie własne.

Bardzo ciekawym sposobem modelowania wymiaru jest jednocześnie zastosowanie wielu tabel do jego opisu, z których każda reprezentuje różny podzbiór danych wymiaru podstawowego. Zależnie od wartości atrybutu, znajdującego się w pierwotnej tabeli wymiaru i różnicującego odmienne wersje tego opisu, używana będzie ta tabela, która wynika z wybranego kontekstu analizy. Liczba tabel opisujących wymiar tego rodzaju będzie więc zależała od liczby unikalnych wartości atrybutu decydującego o wersji danych. Tabele tego rodzaju wymiaru określane są mianem tabel wymiennych i każda z nich charakteryzuje się następującymi cechami:

- ma inne znaczenie;
- pozwala ograniczyć innym użytkownikom dostęp do danych (kolumn i wierszy) z podstawowej tabeli wymiaru (tworzenie różnych widoków faktu);
- ma strukturę inną od pozostałych tabel tego samego wymiaru, co oznacza, że dostarcza innych danych wyjściowych w zależności od wartości atrybutu różnicującego wersje danych z podstawowej tabeli wymiaru;
- zawiera mniej danych w porównaniu z podstawową tabelą wymiaru (mniejsza liczba wierszy i kolumn);
- jej klucz podstawowy jest tym samym kluczem, który istnieje w podstawowej tabeli wymiaru;
- pozostaje w bezpośredniej relacji z tabelą faktu za pomocą tego samego klucza obcego, który używany jest przez podstawową tabelę wymiaru i inne tabele wymienne;
- może być używana razem z inną wersją tego samego wymiaru do opisu tego samego faktu.

Przykład użycia wymiennych tabel wymiaru przedstawiono na rysunku 7.7.



Rys. 7.7. Przykład tabel wymiennych dla wymiaru *Klient*

Źródło: opracowanie własne.

Pierwotna tabela wymiaru *DimCustomer* zawiera dane zarówno o klientach detalicznych, jak i biznesowych. Znajdują się w niej kolumny charakterystyczne dla osób prywatnych (*FirstName*, *LastName*) i firm (*BranchKey*, *BranchName*, *TIN*) oraz dodatkowa kolumna umożliwiająca identyfikację typu klienta (*CustomerType*). Ponieważ ten sam rekord tabeli będzie reprezentował obydwa typy klientów, w związku z tym niektóre z kolumn będą zawierać wartość NULL (np. kolumna *TIN* opisująca NIP nie ma wartości dla klienta detalicznego) ze względu na brak ich adekwatności w takim opisie. Pozostawienie relacji między tabelą faktu i tabelą *DimCustomer* umożliwi wykonywanie analiz z uwzględnieniem typu klienta. Jeśli jednak zajdzie potrzeba pogłębienia

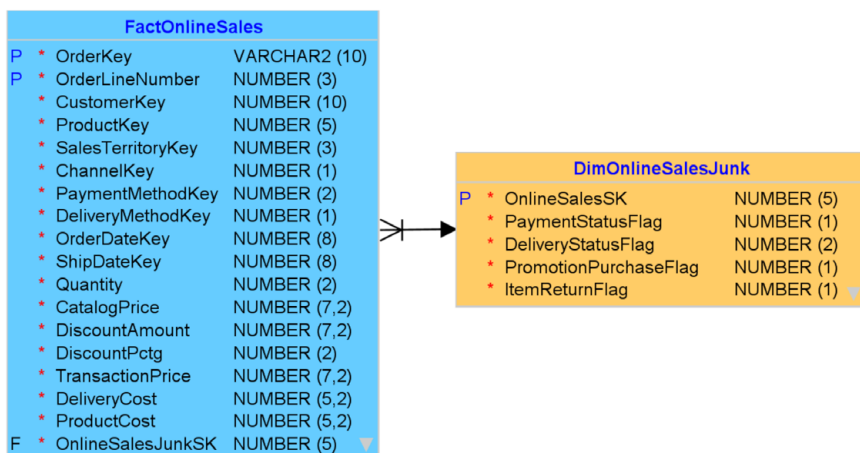
analiz i opisanie ich rezultatów danymi charakterystycznymi dla określonego typu klienta, wówczas podczas wykonywania zapytania dane będą pobierane z tabeli *DimRetailCustomer* bądź też z *DimBusinessCustomer* – zależnie od wartości w kolumnie *CustomerType* podstawowej tabeli wymiaru.

Główną zaletą stosowania wymiennych tabel wymiaru jest możliwość łatwego projektowania i udostępniania różnych widoków danych, adekwatnie do potrzeb informacyjnych użytkownika końcowego.

Jako wadę tego rodzaju rozwiązania można uznać konieczność zarządzania wieloma widokami zamiast jednym i konieczność zapewnienia autoryzacji dostępu do nich przez uprawnione osoby. Ponadto obecność podstawowej tabeli wymiaru i tabel wymiennych wprowadza znaczną redundancję danych, co skutkuje wzrostem rozmiaru bazy danych oraz zwiększa stopień złożoności procesów ETL, które muszą zasilić więcej niż jedną tabelę wymiaru.

W praktyce modelowania wielowymiarowego mogą wystąpić pewne cechy faktu, które wyglądają podobnie jak wymiary zdegenerowane, jednak różnią się od nich kardynalnością, tj. licznością przyjmowanych wartości, można też im przypisać pewne znaczenie biznesowe. Według Margy Ross [26] zbiór takich atrybutów warto wyodrębnić z tabeli faktu do nowej tabeli wymiaru, określanej jako tabela śmieciowych wymiarów. Atrybuty te mogą dotyczyć różnych aspektów związanych z opisem faktu, niekoniecznie powiązanych znaczeniowo ze sobą. Przykładem tego rodzaju opisu są wszelkiego rodzaju wskaźniki (flagi) lub statusy związane z faktem. W odróżnieniu od wymiarów zdegenerowanych dane w tabeli śmieciowych wymiarów mają znaczenie w analizie modelowanego faktu i mogą stanowić istotne uzupełnienie jego opisu, dlatego też nie można ich pominąć. Jednak pozostawienie ich w tabeli faktu może utrudniać poprawne jej interpretowanie, co sprawia, że są one alokowane do odrębnej tabeli.

Przykładami atrybutów, które mogą znaleźć się w tabeli śmieciowych wymiarów, mogą być status dostawy, flaga informująca o promocyjnym zakupie produktu czy też o zwrocie danego produktu z zamówienia. Na rysunku 7.8 przedstawiono fizyczny model danych obrazujący użycie tabeli śmieciowych wymiarów (*DimOnlineSalesJunk*). W tabeli tej znalazły się kolumny opisujące kolejno: status płatności, status dostawy, promocyjny zakup produktu oraz jego opcjonalny zwrot.



Rys. 7.8. Tabela *DimOnlineSalesJunk* jako przykład opisu wymiarów śmieciowych

Źródło: opracowanie własne.

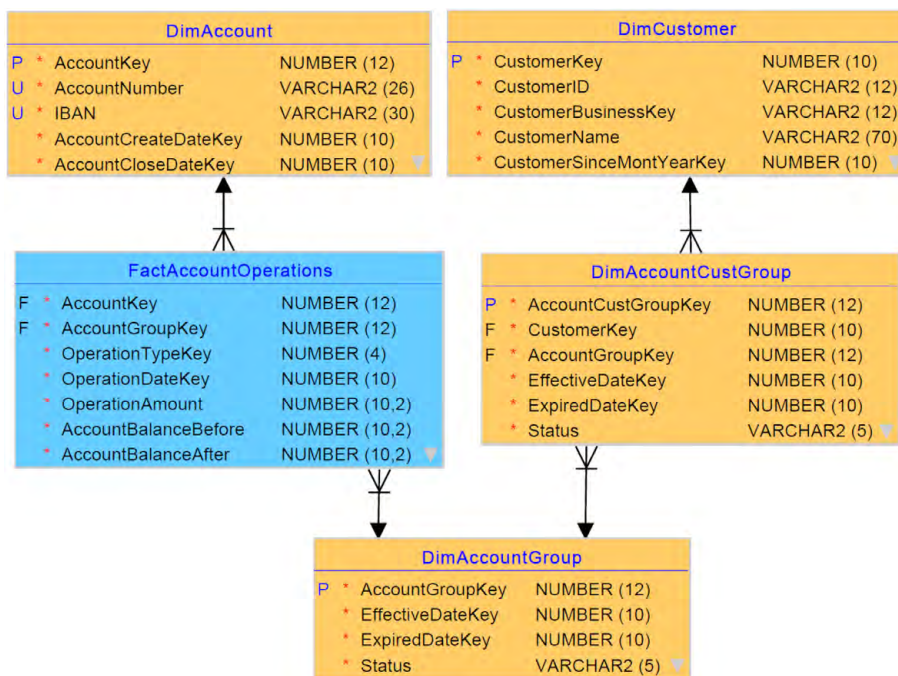
Tabela śmieciowych wymiarów powinna zawierać dane o niskiej kardynalności atrybutów, tj. o niewielkiej liczbie unikalnych wartości występujących w poszczególnych kolumnach. Każdy rekord tabeli reprezentuje pojedynczą kombinację wartości wszystkich odwzorowanych w niej atrybutów. Przykładowo, jeśli tabela zawiera 4 kolumny, a w każdej z nich może wystąpić odpowiednio 5, 10, 15 i 20 unikalnych wartości, to maksymalna liczba wierszy w takiej tabeli jest równa 15 000 ($5 \cdot 10 \cdot 15 \cdot 20 = 15\,000$), co czyni ją stosunkowo dużym zasobem danych jak na tabelę wymiaru.

Główną zaletą posługiwania się tabelą śmieciowych wymiarów jest uproszczenie struktury tabeli faktu i zmniejszenie jej rozmiaru oraz wzrost elastyczności tworzenia zapytań analitycznych z wykorzystaniem danych z tej tabeli. Jednak obecność dodatkowego złączenia z tabelą faktu prowadzi do obniżenia ich wydajności. W miarę pojawiania się nowych wartości atrybutów reprezentowanych przez kolumny tabeli, jej rozmiar może znacząco wzrastać.

Rezultatem dotychczas zaprezentowanych technik modelowania wymiarów była tabela wymiaru, której pojedynczy rekord może być powiązany z wieloma rekordami w tabeli faktu i jednocześnie pojedynczy rekord tabeli faktu opisywany jest danymi z tylko jednego rekordu tabeli wymiaru. Jest to klasyczne podejście charakterystyczne dla modelu gwiazdy, płatka śniegu czy konstelacji faktów, w których obecne są relacje jeden-do-wielu między tabelami. W rzeczywistości występują jednak przypadki opisywania tego samego

faktu za pomocą wielu wartości tego samego atrybutu wymiaru. Przykładami obrazującym taką sytuację są np.: realizacja pojedynczego zamówienia, w którą zaangażowanych jest wielu pracowników; wystąpienie incydentu bezpieczeństwa, którego skutkiem jest wiele szkód wyrządzonych w zasobach lokalnej sieci komputerowej; bądź też diagnoza pacjenta, podczas której zidentyfikowano wiele objawów chorobowych.

Tego rodzaju złożone scenariusze wymagają odmiennego podejścia w projektowaniu struktury tabel wymiarów i zastosowania tzw. tabel pomostowych, które umożliwią odwzorowanie związku wiele-do-wielu pomiędzy tabelą faktu i tabelą wymiaru bądź też pomiędzy tabelami wymiaru. Na rysunku 7.9 przedstawiono fizyczny model danych, który opisuje fakt przeprowadzania operacji na rachunku bankowym z wykorzystaniem tabeli pomostowej.



Rys. 7.9. Tabela *DimAccountCustGroup* jako przykład tabeli pomostowej

Źródło: opracowanie własne.

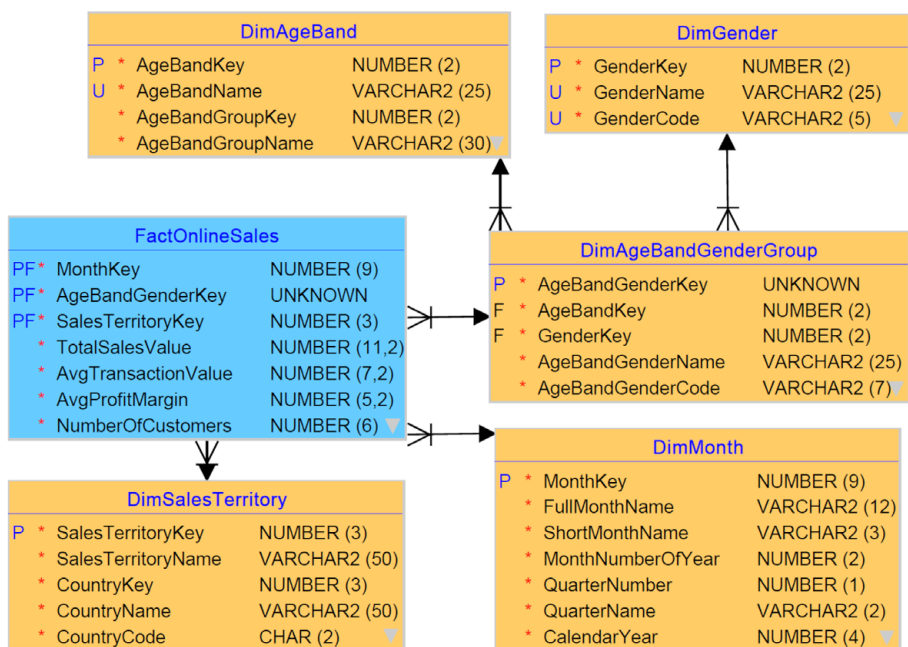
Każdy rekord tabeli faktu *FactAccountOperation* odpowiada jednemu rodzajowi operacji wykonanej w tym samym dniu na danym rachunku bankowym przez jego właściciela lub pełnomocników. Ponieważ między tabelą faktu i pierwotną tabelą wymiaru *DimCustomer* występuje związek wiele-do-wielu, konieczne stało się wprowadzenie tabeli pomostowej *DimAccountCustGroup*, która opisuje wszystkie osoby upoważnione do różnych operacji na rachunku. Osoby te zaliczono do jednej grupy zdefiniowanej w tabeli *DimAccountGroup*. W tabeli pomostowej pojawiają się dodatkowe kolumny charakteryzujące przynależność danego klienta do określonej grupy, którymi są *EffectiveDateKey* (data zapisania do grupy), *ExpiredDateKey* (data wypisania z grupy) i *Status* (aktualny lub historyczny członek grupy). Bez tych kolumn tabela pomostowa byłaby tabelą faktu bez faktu, gdyż zawierałaby wyłącznie klucze obce *CustomerKey* i *AccountGroupKey*, odwołujące się odpowiednio do tabel *DimCustomer* i *DimAccountGroup*.

Innym przykładem wykorzystania tabeli pomostowej w modelowaniu wymiarów jest scenariusz, w którym konieczne jest uwzględnienie w opisie faktu iloczynu kartezyjańskiego danych pochodzących z różnych tabel wymiarów. Przypadek taki zilustrowano na rysunku 7.10, gdzie tabela *DimAgeBandGenderGroup* zawiera rekordy będące produktem kartezyjańskim wierszy z tabel *DimAgeBand* i *DimGender*. Tabela faktu *FactOnlineSales* umożliwia opis zachowań zakupowych klientów różnej płci w różnych grupach wiekowych. Zastosowanie tabeli pomostowej *DimAgeBandGenderGroup* ułatwia wykonanie tego typu analiz, ponieważ zawiera ona możliwe pary wartości płeć-grupa wiekowa, a przy tym jest bezpośrednio powiązana z tabelą faktu. W ten sposób znika konieczność odrębnego grupowania danych według wymiarów na etapie analizy, gdyż odpowiednie ich agregaty zostały już przygotowane w procesie ETL i załadowane do tabeli pomostowej oraz powiązanej z nią tabeli faktu.

Tabele pomostowe są często stosowane w modelowaniu wielowymiarowym, ponieważ upraszczają zapytania poprzez zmniejszenie liczby potrzebnych złączeń i filtrów, zwiększają ich wydajność, unikając iloczynów kartezyjańskich i dużych pośrednich zbiorów wynikowych na etapie analizy, a także zwiększają elastyczność tworzenia różnych przekrojów informacyjnych dzięki zastosowaniu różnych sposobów łączenia ze sobą wielu wymiarów. Ponadto tabele pomostowe zapewniają możliwość implementacji złożonych reguł biznesowych i scenariuszy, w których pomiędzy danymi istnieją relacje

wiele-do-wielu, pozwalają też zachowywać integralność i spójność danych, unikając ich duplikacji (szczególnie tych opisujących fakty).

Obecność tabel pomostowych wiąże się także z pewnymi wyzwaniami. Wśród nich wymienia się konieczność zapewnienia prostej struktury tych tabel, czytelnego dokumentowania ich znaczenia, poprawnego alokowania w modelu danych – tj. umieszczania jak najbliżej tabel, w których zostaną wykorzystane zgromadzone w nich dane – oraz zarządzania odpowiednimi indeksami i właściwego partycjonowania danych z tych tabel. Uwzględnienie tych aspektów stworzy optymalne warunki do wykonywania zaawansowanych analiz, zapewniając ich wysoką wydajność i możliwość tworzenia najbardziej złożonych przekrojów informacyjnych.



Rys. 7.10. Tabela *DimAgeBandGenderGroup* jako tabela pomostowa opisująca iloczyn kartezyński między tabelami wymiarów

Źródło: opracowanie własne.

7.2. Wymiary wolnozmiennie

W zaprezentowanych dotychczas technikach modelowania wymiarów przyjęto założenie o stałości wartości atrybutu opisującego dany wymiar – np. nazwa danego kraju w kolumnie *CountryName* w tabeli *DimSalesTerritory* (rys. 6.11 lub rys. 6.12) pozostaje niezmienna w czasie. Jednak takie założenie nie jest w pełni zgodne z rzeczywistością, gdyż nazwa kraju może ulec zmianie z powodów politycznych lub kulturowych. Przykładem takiej sytuacji jest Holandia (Holland), której rząd zdecydował w 2020 roku o zmianie oficjalnej nazwy kraju na Niderlandy (Netherlands). Następstwem tego powinna być modyfikacja odpowiedniego rekordu w tabeli *DimSalesTerritory*. Wykonując taką operację, należy jednak mieć świadomość jej konsekwencji, a będą one bardzo poważne z analitycznego punktu widzenia. Nadpisanie poprzedniej nazwy spowoduje bowiem, że we wszystkich wykorzystujących ją raportach analitycznych pojawi się nowa wartość – niezależnie od tego, czy dane te dotyczą okresu sprzed zmiany, czy po niej. Innymi słowy, wszystkie dane historyczne pochodzące z tabeli faktu zostaną opisane nową nazwą kraju, co należy uznać za istotny błąd prowadzący do wyświetlania niepoprawnych danych analitycznych. Jak zatem sprawić, by użytkownik końcowy widział dane historyczne zgodnie ze stanem faktycznym (poprzednią nazwę w przeszłości i aktualną w teraźniejszości)? Rozwiązaniem jest zastosowanie tzw. wymiaru wolnozmiennego (SCD – ang. *slowly changing dimension*), umożliwiającego posługiwanie się atrybutami, których wartość może ulegać zmianie w czasie przy jednoczesnym zachowaniu pełnej ścieżki historii tych zmian.

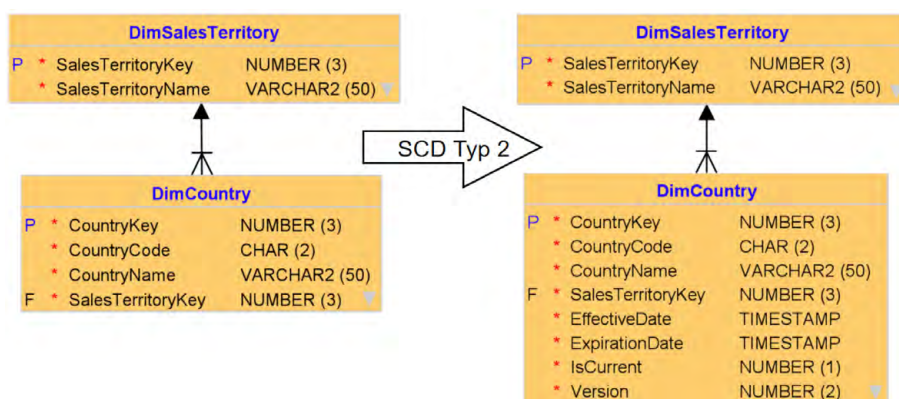
W literaturze przedmiotu [15] wymienia się następujące podstawowe techniki modelowania SCD:

- Typ 1 – nadpisanie poprzedniej wartości atrybutu nową wartością;
- Typ 2 – utworzenie nowego wiersza w tabeli wymiaru, zawierającego nową wartość atrybutu (technika najczęściej stosowana);
- Typ 3 – dodanie nowej kolumny w tabeli wymiaru, która będzie przechowywała poprzednią wartość atrybutu, a nowa wartość zostanie umieszczona w pierwotnej kolumnie opisującej dany wymiar.

Typ 1 jest najprostszy w implementacji, jednak jego zastosowanie prowadzi do utraty informacji o wartości atrybutu w przeszłości. Tym samym wszystkie dane znajdujące się w tabeli faktu zawsze będą opisane najnowszą

wartością atrybutu danego wymiaru. Ten typ SCD stosuje się w sytuacji, gdy wartości historyczne atrybutu nie są istotne w analizie danych. Przykładem użycia SCD Typu 1 może być zmiana charakterystyki produktu w systemie ERP, która spowoduje nadpisanie tego opisu w hurtowni danych, przy założeniu, że nie jest konieczne śledzenie tych zmian w środowisku analitycznym.

Znacznie lepszym rozwiązaniem jest zastosowanie SCD Typu 2, który umożliwia zapisanie w tabeli wymiaru pełnej historii zmian wartości atrybutu użytego do jego opisu wymiaru. Aby w pełni wykorzystać zalety tego typu SCD, przykładowa tabela *DimSalesTerritory*, w której nazwa kraju może ulegać zmianie, powinna być znormalizowana, tj. jej struktura musi być podzielona na dwie odrębne tabele wymiarów, np. *DimSalesTerritory* i *DimCountry*. Dodatkowo w tabeli *DimCountry* konieczne jest utworzenie nowych kolumn, których wartości będą opisywać historię zmian wartości atrybutu *CountryName*. Na rysunku 7.11 przedstawiono przykład struktury tabeli *DimCountry* przed zastosowaniem modelowania SCD Typu 2 oraz po zmianach wynikających z tej operacji.



Rys. 7.11. Modelowanie wymiaru wolnozmiennego typu 2

Źródło: opracowanie własne.

Zmodyfikowana struktura tabeli *DimCountry* musi zawierać co najmniej dwie dodatkowe kolumny, opisujące czas wprowadzenia nowej wartości (kolumna *EffectiveDate*) i czas zakończenia jej obowiązywania (kolumna *ExpirationDate*). Opcjonalnie można utworzyć kolumnę *IsCurrent*, przyjmującą wartości 0 (rekord z archiwalną nazwą kraju) lub 1 (rekord z bieżącą nazwą kraju), oraz kolumnę *Version*, opisującą numer wersji rekordu dla danego kraju. Szczególnie użyteczna jest pierwsza z tych kolumn, gdyż może być wykorzystana w warunku *WHERE* polecenia *SELECT* do szybszego wybrania bieżącej nazwy kraju.

Rejestrowanie historii zmian wartości atrybutu *CountryName*, opisującego nazwę kraju, w przypadku użycia SCD Typu 2 polega na:

- utworzeniu w tabeli wymiaru *DimCountry* nowego rekordu i wygenerowaniu w nim unikalnej wartości w kolumnie *CountryKey*;
- wstawieniu nowej wartości w kolumnie *CountryName* pochodzącej z systemu transakcyjnego oraz skopiowaniu do kolumny *CountryCode* wartości klucza biznesowego z rekordu archiwalnego;
- zmodyfikowaniu wartości w nowym rekordzie w kolumnach *EffectiveDate* (wstawienie daty wprowadzenia zmiany), *IsCurrent* (wstawienie wartości 1) i *Version* (inkrementowana wartość z tej kolumny z rekordu archiwalnego);
- zmodyfikowaniu w rekordzie archiwalnym wartości kolumn *ExpirationDate* (wstawienie daty końca obowiązywania wartości poprzedniej) i *IsCurrent* (wstawienie wartości 0).

Na rysunku 7.12 przedstawiono przykład obrazujący sposób modyfikacji zawartości tabeli *DimCountry* w przypadku zmiany nazwy kraju z Holland na Netherlands. W przytoczonym przykładzie rekord z aktualną nazwą kraju ma wartość NULL w kolumnie *ExpirationDate*. W praktyce spotyka się również przypadki, gdzie wartość w tej kolumnie ma charakter całkowicie abstrakcyjny, np. 31-12-9999 lub 31-12-4999. Obydwa podejścia są jednak prawidłowe, gdyż do wskazania tego rekordu jako zawierającego bieżącą nazwę kraju nie używa się wartości z kolumny *ExpirationDate*, lecz kolumny *IsCurrent*.

Aby zachować możliwość identyfikacji danego kraju mimo zmiany jego nazwy, konieczne jest zastosowanie atrybutu mającego stałą wartość w czasie. W prezentowanym przykładzie takim atrybutem jest *CountryCode*. Jego wartość pozostaje niezmienna w nowym rekordzie. Kolumna ta jest kluczem

biznesowym w tabeli *DimCountry*, gdyż ma pewne znaczenie biznesowe lub administracyjne, jest także wykorzystywana w systemach ERP do jednoznacznego opisu rekordów, co umożliwia powiązanie danych między transakcyjną i analityczną bazą danych podczas realizacji procesów ETL.

Rekord przed zmianą nazwy kraju

CountryKey	CountryName	CountryCode	SalesTerritoryKey	EffectiveDate	ExpirationDate	IsCurrent	Version
9	Holland	NL	3	01-01-1965	null	1	1

Rekord po zmianie nazwy kraju

CountryKey	CountryName	CountryCode	SalesTerritoryKey	EffectiveDate	ExpirationDate	IsCurrent	Version
9	Holland	NL	3	01-01-1965	31-12-2019	0	1
15	Netherlands	NL	3	01-01-2020	null	1	2

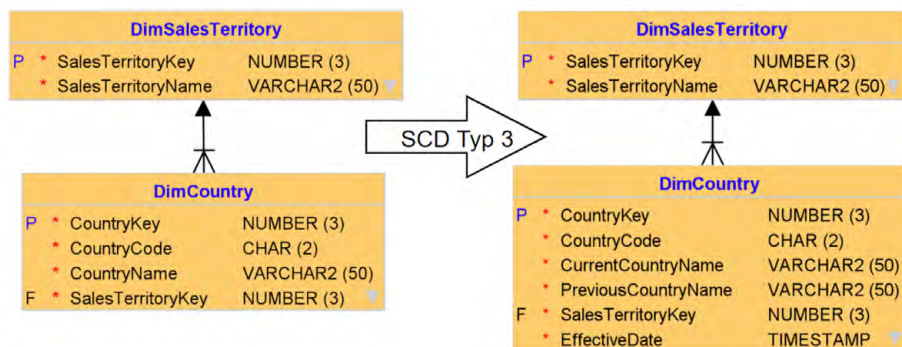
Rys. 7.12. Przykład implementacji SCD Typu 2 dla tabeli *DimCountry*

Źródło: opracowanie własne.

Tworząc nowy rekord w tabeli wymiaru *DimCountry*, należy pamiętać, aby reprezentował on de facto nowe państwo. W tym celu definiuje się tzw. klucz surogatowy (w przykładzie jest to kolumna *CountryKey*), którego wartości są unikatowe i posłużą jako wartości klucza obcego w powiązanej tabeli faktu, umożliwiając w ten sposób poprawny sposób interpretacji danych analitycznych wygenerowanych po zmianie nazwy kraju. Klucz surogatowy jest sztucznym identyfikatorem bez merytorycznego znaczenia, stosowanym wyłącznie w celu utworzenia relacji między tabelami. Zaleca się, aby był kluczem prostym, tj. składającym się wyłącznie z jednej kolumny, gdyż pozwala to uniknąć nadmiernego wzrostu złożoności danych i ogranicza niepotrzebny przyrost danych w tabelach wymiarów.

Implementacja SCD Typu 2 jest najbardziej popularnym sposobem śledzenia zmian wartości atrybutu wymiaru. Wysoką użyteczność tej techniki potwierdza obecność w narzędziach ETL predefiniowanych procesów wbudowanych, które po odpowiedniej konfiguracji realizują wszystkie zadania niezbędne do utworzenia historii tych zmian, tj. zapewniają identyfikację nowej wartości atrybutu w systemie źródłowym i utworzenie nowego wiersza w tabeli wymiaru z wymaganymi danymi.

Innym sposobem rejestrowania zmian wartości atrybutu wymiaru jest użycie SCD Typu 3. W odróżnieniu od typu 2 technika ta umożliwia zapisanie historii ograniczonej wyłącznie do jednej zmiany tej wartości. Przykład struktury tabeli *DimCountry*, dla której zastosowano SCD Typu 3, przedstawiono na rysunku 7.13.



Rys. 7.13. Modelowanie wymiaru wolnozmennego typu 3

Źródło: opracowanie własne.

W zmodyfikowanej strukturze tabeli *DimCountry* musi wystąpić dodatkowa kolumna *PreviousCountryName*, w której zostanie zapisana nazwa kraju przed zmianą. Ponadto można utworzyć kolumnę *EffectiveDate* w celu zapisania daty wprowadzenia tej zmiany. Kolumna *CurrentCountryName* będzie przechowywać aktualną nazwę kraju.

Implementacja SCD Typu 3 dla tabeli *DimCountry* jest znacznie prostsza w porównaniu z SCD Typu 2 i obejmuje:

- modyfikację rekordu opisującego dany kraj poprzez wstawienie w kolumnie *CurrentCountryName* nowej nazwy kraju;
- aktualizację wspomnianego rekordu w kolumnie *PreviousCountryName* poprzez wprowadzenie archiwalnej nazwy kraju oraz daty dokonania zmiany w kolumnie *EffectiveDate*.

Przykład prezentujący rezultat wykonania powyższych działań przedstawiono na rysunku 7.14. Główną zaletą użycia SCD Typu 3 jest brak wzrostu rozmiaru tabeli, gdyż cała operacja utrwalenia historii zmian przeprowadzana jest wyłącznie na jednym rekordzie. Jednak znaczne ograniczenie możliwości śledzenia tych zmian sprawia, że rozwiązanie to stosuje się wyłącznie do opisu takiego wymiaru, którego atrybut zmienia swoją wartość tylko jeden raz, lub gdy operacje analityczne wymagają porównywania wyników kalkulacji tylko dla dwóch kolejnych wartości atrybutu.

Rekord przed zmianą nazwy kraju

CountryKey	CurrentCountryName	PreviousCountryName	CountryCode	SalesTerritoryKey	EffectiveDate
9	Holland	Holland	NL	3	01-01-1965

Rekord po zmianie nazwy kraju

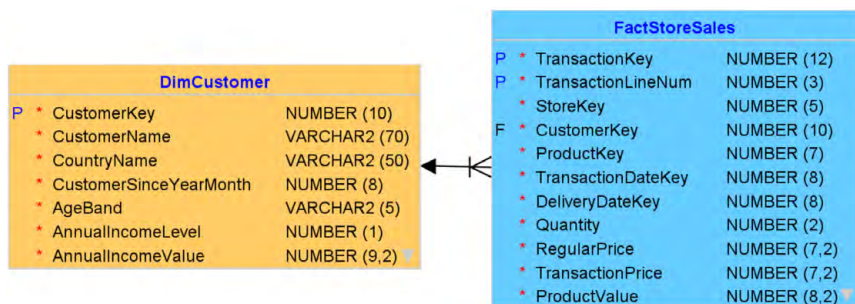
CountryKey	CurrentCountryName	PreviousCountryName	CountryCode	SalesTerritoryKey	EffectiveDate
9	Netherlands	Holland	NL	3	01-01-2020

Rys. 7.14. Przykład implementacji SCD Typu 3 dla tabeli *DimCountry*

Źródło: opracowanie własne.

Przedstawione powyżej sposoby śledzenia zmian wartości atrybutu wymiaru nie wyczerpują pełnej listy tego rodzaju technik. Margy Ross, liderka w Kimball Group i DecisionWorks Consulting, zaproponowała dodatkowe podejścia do realizacji idei tworzenia śladu zmian tych wartości. Zostały one określone jako SCD Typu 4, SCD Typu 5, SCD Typu 6 i SCD Typu 7 [25].

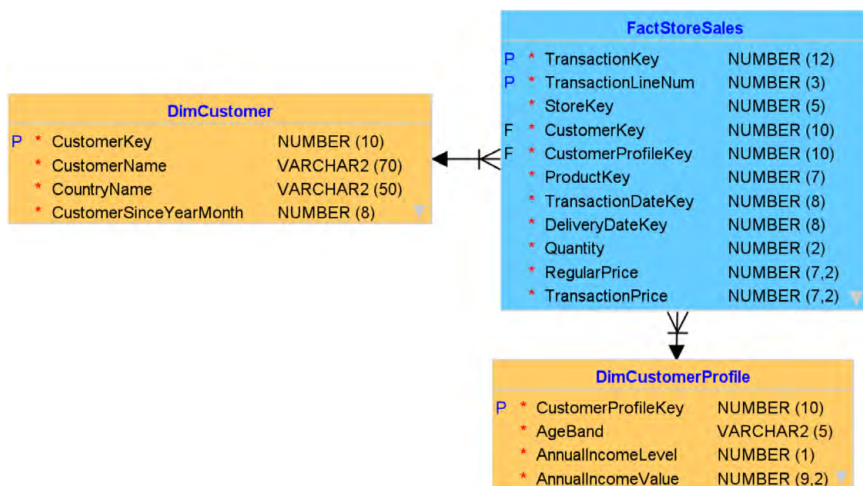
Technika SCD Typu 4 polega na ekstrakcji grupy atrybutów, których wartości będą się zmieniać, do nowej dodatkowej tabeli wymiaru, a następnie zdefiniowaniu odrębnych relacji między poszczególnymi tabelami wymiarów i tabelą faktu. W celu zobrazowania sposobu implementacji tej techniki na rysunku 7.15 przedstawiono strukturę tabeli *DimCustomer*, w której znajdują się m.in. kolumny opisujące przedział wiekowy klienta (*AgeBand*), poziom jego rocznych dochodów (*AnnualIncomeLevel*) oraz ich wysokość (*AnnualIncomeValue*).



Rys. 7.15. Model danych przed zastosowaniem SCD Typu 4 do tabeli *DimCustomer*

Źródło: opracowanie własne.

Wartości w tych kolumnach mogą ulegać częstszym zmianom w porównaniu z omawianą uprzednio nazwą kraju w tabeli *DimCountry*. Dlatego też w tym przypadku Margy Ross zaleca użycie techniki SCD Typu 4, rezultatem czego będzie struktura danych zaprezentowana na rysunku 7.16. Z pierwotnej tabeli wymiaru *DimCustomer* wydzielono do nowej tabeli *DimCustomerProfile* kolumny *AgeBand*, *AnnualIncomeLevel* i *AnnualIncomeValue*. W tabeli tej utworzono kolumnę *CustomerProfileKey*, będącą kluczem surogatowym wykorzystanym do zdefiniowania relacji pomiędzy tą tabelą i tabelą faktu *FactStoreSales*. Każda zmiana przedziału wiekowego klienta lub jego rocznych dochodów spowoduje wstawienie w tabeli *DimCustomerProfile* nowego rekordu, pod warunkiem że rekord z takimi wartościami kolumn jeszcze nie istnieje.



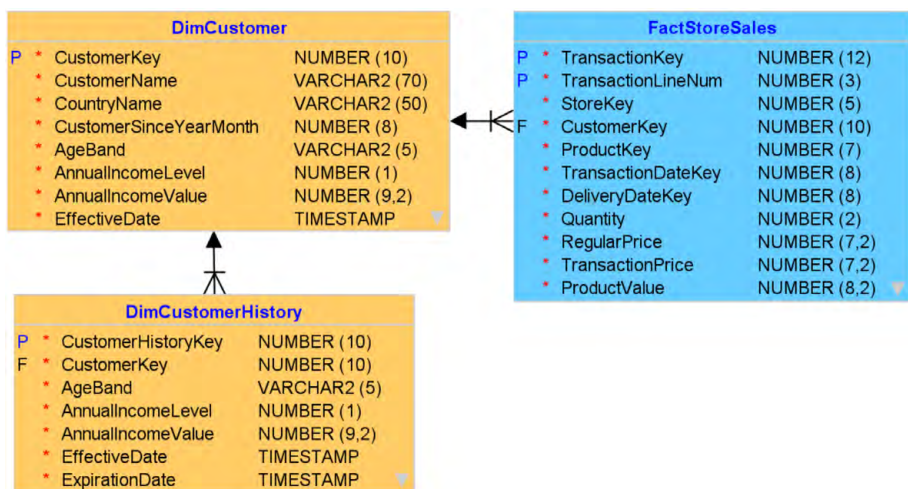
Rys. 7.16. Model danych po użyciu SCD Typu 4 do tabeli *DimCustomer* – wariant 1

Źródło: opracowanie własne.

Analizując diagram na rysunku 7.16, można zauważyć brak relacji pomiędzy tabelami *DimCustomer* i *DimCustomerProfile*. Sytuacja taka uniemożliwia bezpośrednie śledzenie zmian wartości w kolumnach wydzielonych do tabeli *DimCustomerProfile* dla konkretnego klienta. Jednak z analitycznego punktu widzenia istotna jest możliwość określenia wpływu tych zmian na zachowania zakupowe klienta, opisane w tabeli faktu *FactStoreSales*, a będzie

ona zapewniona, gdyż dane w tej tabeli opisane są wartościami kluczy obcych z obydwu tabel wymiarów. Dlatego też schemat przedstawiony na rysunku 7.16 należy uznać za poprawny.

Jeśli jednak istnieją powody, by zapamiętać pełną historię zmian wartości atrybutu wymiaru w powiązaniu z konkretnym rekordem z pierwotnej tabeli wymiaru, wówczas konieczne jest zastosowanie innego sposobu modelowania SCD Typu 4. Przykładowe rozwiązanie przedstawiono na rysunku 7.17. Implementacja tego typu SCD stanowi kombinację działań charakterystycznych dla typu 1 i typu 2 i polega na nadpisaniu wartości w kolumnach *AgeBand*, *AnnualIncomeLevel* lub *AnnualIncomeValue* w tabeli *DimCustomer* oraz na wstawieniu rekordu archiwalnego do tabeli *DimCustomerHistory* z poprzednimi wartościami tych kolumn. Rekord w tej tabeli jest dodatkowo opisany czasem obowiązywania tych wartości, co umożliwia analizę zagadnienia z tabeli faktu z wykorzystaniem historycznych danych na temat klienta.



Rys. 7.17. Model danych po użyciu SCD Typu 4 do tabeli *DimCustomer* – wariant 2

Źródło: opracowanie własne.

Przykład prezentujący rezultat wykonania działań w ramach implementacji SCD Typu 4 w wariantcie umożliwiającym zapisanie historii zmian powiązanej z konkretnym klientem przedstawiono na rysunku 7.18. W przykładzie tym zmianie ulegają wartości w kolumnach *AnnualIncomeLevel*

i *AnnualIncomeValue* w tabeli *DimCustomer*. Dotychczasowe wartości tych kolumn dla danego klienta zostają nadpisane, a w tabeli *DimCustomerHistory* wstawiona jest kopia rekordu sprzed zmiany z tabeli *DimCustomer*, z przypisaną wartością 1100 dla klucza surogatowego *CustomerHistoryKey*.

Rekord przed zmianą wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerKey</i>	<i>CustomerName</i>	<i>CountryName</i>	<i>CustomerSince</i> <i>YearMonth</i>	<i>AgeBand</i>	<i>AnnualIncome</i> <i>Level</i>	<i>AnnualIncome</i> <i>Value</i>	<i>EffectiveDate</i>
9	Muryjas Mateusz	Poland	01-2019	30-40	4	475 000,00	01-01-2020

Zawartość tabeli *DimCustomerHistory* przed zmianą w tabeli *DimCustomer*

<i>CustomerHistoryKey</i>	<i>CustomerKey</i>	<i>AgeBand</i>	<i>AnnualIncome</i> <i>Level</i>	<i>AnnualIncome</i> <i>Value</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>
1000	9	30-40	3	350 000,00	01-01-2019	31-12-2019

Rekord po zmianie wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerKey</i>	<i>CustomerName</i>	<i>CountryName</i>	<i>CustomerSince</i> <i>YearMonth</i>	<i>AgeBand</i>	<i>AnnualIncome</i> <i>Level</i>	<i>AnnualIncome</i> <i>Value</i>	<i>EffectiveDate</i>
9	Muryjas Mateusz	Poland	01-2019	30-40	5	525 000,00	23-09-2021

Zawartość tabeli *DimCustomerHistory* po zmianie w tabeli *DimCustomer*

<i>CustomerHistoryKey</i>	<i>CustomerKey</i>	<i>AgeBand</i>	<i>AnnualIncome</i> <i>Level</i>	<i>AnnualIncome</i> <i>Value</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>
1100	9	30-40	4	475 000,00	01-01-2020	22-09-2021
1000	9	30-40	3	350 000,00	01-01-2019	31-12-2019

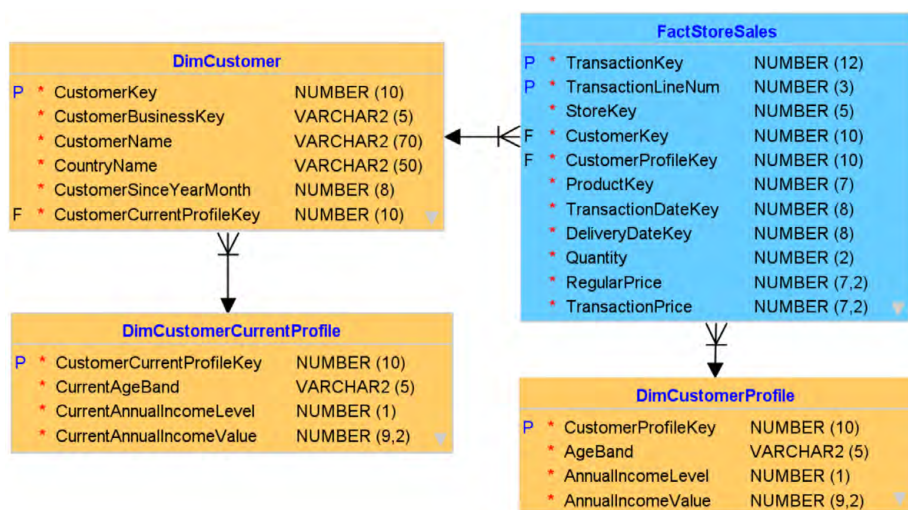
Rys. 7.18. Przykład implementacji SCD Typu 4 wariant 2 dla tabeli *DimCustomer*

Źródło: opracowanie własne.

SCD Typu 4 jest najczęściej stosowaną techniką w przypadku modelowania tych wymiarów, dla których wartości atrybutów zmieniają się bardzo często. Uzyskanie poprawnych rezultatów zastosowania tego podejścia wymaga jednak zaawansowanych umiejętności projektowania złożonych procesów ETL, w których będą identyfikowane zmiany w systemach źródłowych oraz wykonywane operacje *UPDATE* i *INSERT* w odpowiednich tabelach wymiarów. Ponadto posługiwanie się dodatkową tabelą z danymi historycznymi powoduje wzrost objętości bazy danych, co może być postrzegane jako istotna wada tego rozwiązania.

Innym sposobem modelowania historii zmian wartości atrybutu wymiaru jest SCD Typu 5, który łączy w sobie działania typu 1 i typu 4. Można powiedzieć, że typ 5 stanowi rozszerzenie typu 4 o dodatkową zewnętrzną tabelę wymiaru, przechowującą bieżące wartości atrybutów wymiaru. Na rysunku 7.19 przedstawiono przykład zastosowania SCD Typu 5 dla wymiaru klienckiego opisującego sprzedaż stacjonarną. Podobnie jak w przypadku SCD Typu 4, zakłada się, że zmianie mogą ulegać wartości atrybutów opisujących przedział wiekowy klienta (*AgeBand*), poziom jego dochodów (*AnnualIncomeLevel*)

lub ich wartość (*AnnualIncomeValue*). Charakterystyczną cechą SCD Typu 5 jest obecność tabeli *DimCustomerCurrentProfile*, w której istnieje dokładnie jeden rekord powiązany z każdym rekordem klienta w tabeli *DimCustomer*. Jednocześnie w technice tej pozostawiono tabelę *DimCustomerProfile*, służącą do bezpośredniego opisu faktu sprzedażowego.



Rys. 7.19. Model danych po użyciu SCD Typu 5 do tabeli *DimCustomer*

Źródło: opracowanie własne.

Podczas implementacji SCD Typu 5 w procesie zasilenia hurtowni danych wykonywane są następujące operacje, jeśli zostanie wykryta w systemie transakcyjnym zmiana wartości jednego z uprzednio wymienionych atrybutów wymiaru klienckiego:

- opcjonalne wykonanie instrukcji *INSERT* w tabeli *DimCustomerProfile*, jeśli nie istnieje w niej rekord zawierający nowe wartości atrybutów *AgeBand*, *AnnualIncomeLevel* i *AnnualIncomeValue*;
- modyfikacja rekordu w tabeli *DimCustomerCurrentProfile*, odpowiadającego rekordowi klienta, dla którego zmieniły się wartości w systemie źródłowym;
- wstawianie nowego rekordu w tabeli *DimCustomer* z taką samą wartością klucza biznesowego klienta, nową wartością klucza surogatowego *CustomerKey* i uprzednio istniejącą wartością klucza obcego *CustomerCurrentProfileKey*.

W celu zobrazowania sposobu zmiany zawartości tabel wymiarów użytych do modelowania SCD Typu 5 dla tabeli *DimCustomer* na rysunku 7.20 przedstawiono wiersze z tych tabel przed zmianą wartości opisujących dochód klienta, natomiast na rysunku 7.21 uwidoczniło zmianę zawartości tych tabel po modyfikacji w systemie źródłowym danych o tym dochodzie.

Rekord tabeli *DimCustomer* przed zmianą wartości atrybutów *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>CustomerCurrentProfileKey</i>
9	CS345	...	1

Rekord tabeli *DimCustomerCurrentProfile* przed zmianą wartości atrybutów *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerCurrentProfileKey</i>	<i>CurrentAgeBand</i>	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>
1	30-40	3	350 000,00

Rekordy tabeli *DimCustomerProfile* przed zmianą wartości atrybutów *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerProfileKey</i>	<i>AgeBand</i>	<i>AnnualIncomeLevel</i>	<i>AnnualIncomeValue</i>
1	30-40	3	350 000,00
2	30-40	4	475 000,00

Rys. 7.20. Przykład implementacji SCD Typu 5 dla tabeli *DimCustomer* – widok przed zmianą wartości atrybutów opisujących dochody klienta

Źródło: opracowanie własne.

Rekordy tabeli *DimCustomer* po zmianie wartości atrybutów *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>CustomerCurrentProfileKey</i>
9	CS345	...	3
99	CS345	...	3

Zawartość tabeli *DimCustomerCurrentProfile* po zmianie wartości atrybutów *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerCurrentProfileKey</i>	<i>CurrentAgeBand</i>	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>
3	30-40	5	590 000,00

Zawartość tabeli *DimCustomerProfile* po zmianie wartości atrybutów *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerProfileKey</i>	<i>AgeBand</i>	<i>AnnualIncomeLevel</i>	<i>AnnualIncomeValue</i>
1	30-40	3	350 000,00
2	30-40	4	475 000,00
3	30-40	5	590 000,00

Rys. 7.21. Przykład implementacji SCD Typu 5 dla tabeli *DimCustomer* – widok po zmianie wartości atrybutów opisujących dochody klienta

Źródło: opracowanie własne.

Zdefiniowanie relacji pomiędzy tabelami *DimCustomerCurrentProfile* i *DimCustomer* umożliwia szybsze uzyskanie informacji o bieżącym profilu klienta bez konieczności odwoływania się do tabeli faktu. Jest to bardzo istotna zaleta tego rozwiązania, gdyż pozwala zawsze pozyskać dane o bieżących wartościach w kolumnach charakteryzujących aktualny profil klienta, tj. *CurrentAgeBand*, *CurrentAnnualIncomeLevel* i *CurrentAnnualIncomeValue*, nawet gdy nie dokonał on jeszcze zakupu po zmianie danych o jego dochodach lub grupie wiekowej.

Główną wadą implementacji SCD Typu 5 jest znaczny wzrost stopnia złożoności procesu ETL, który musi zapewnić poprawną realizację działań w ramach SCD Typu 1 i 4, czyli właściwą aktualizację zawartości kilku tabel, tj. tabeli bazowej (np. *DimCustomer*), tabeli z bieżącymi wartościami atrybutów (np. *DimCustomerCurrentProfile*), tabeli z danymi historycznymi (np. *DimCustomerHistory*) bądź też tabeli wyszukiwania (np. *DimCustomerProfile*). W przypadku znaczącego obniżenia wydajności zapytań z użyciem zewnętrznej tabeli wymiaru (np. *DimCustomerCurrentProfile*) zaleca się modyfikację struktury diagramu z rysunku 7.19 poprzez usunięcie tej tabeli i umieszczenie jej kolumn w tabeli bazowej *DimCustomer*.

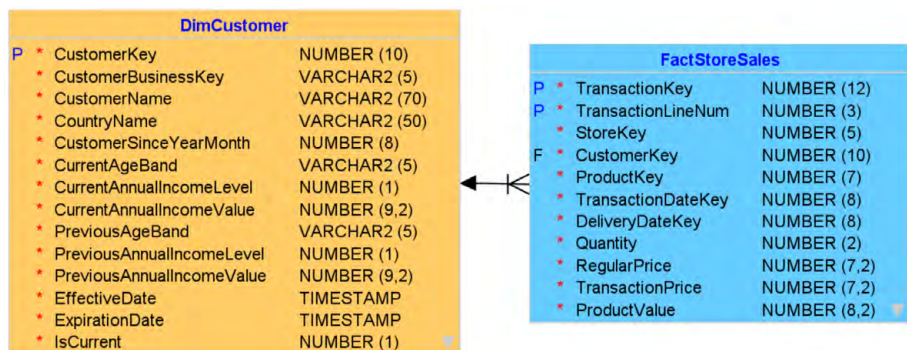
Bardzo ciekawym sposobem modelowania wymiaru wolnozmiennego jest podejście określane jako typ 6, łączące w sobie techniki pochodzące z typu 1, 2 i 3. To hybrydowe rozwiązanie okazuje się szczególnie użyteczne w sytuacjach, gdy zachodzi konieczność śledzenia w tym samym czasie bieżących, historycznych i statycznych danych opisujących wymiar. SCD Typu 6 pozwala znacząco rozszerzyć możliwości analizowania wpływu zmian wartości atrybutów wymiaru na dane znajdujące się w tabeli faktu.

Zaimplementowanie SCD Typu 6 wymaga zmiany struktury bazowej tabeli wymiaru poprzez utworzenie w niej dodatkowych kolumn umożliwiających przechowywanie historycznych wartości atrybutów, dat ich obowiązywania oraz identyfikację ich bieżących wartości. Dla przykładowej tabeli *DimCustomer*, zobrazowanej na rysunku 7.15, dokonano modyfikacji jej struktury na potrzeby SCD Typu 6, polegających na:

- utworzeniu kolumny *CustomerBusinessKey* umożliwiającej identyfikację tego samego klienta niezależnie od tego, jaka jest dla niego aktualna wartość *CustomerKey*;
- zmianie nazw kolumn *AgeBand*, *AnnualIncomeLevel* i *AnnualIncomeValue* odpowiednio na *CurrentAgeBand*, *CurrentAnnualIncomeLevel* i *Current-*

AnnualIncomeValue w celu intuicyjnego wskazania miejsca przechowywania aktualnych wartości odpowiednich atrybutów;

- utworzeniu dodatkowych kolumn *CustomerBusinessKey*, *PreviousAgeBand*, *PreviousAnnualIncomeLevel*, *PreviousIncomeValue*, *EffectiveDate*, *ExpirationDate*, w których zapisane będą historyczne wartości odpowiednich atrybutów, oraz kolumny *IsCurrent* do wskazania rekordu z aktualną ich wartością. Strukturę tabeli *DimCustomer* po wprowadzeniu opisanych wyżej zmian zobrazowano na rysunku 7.22.



Rys. 7.22. Model danych po użyciu SCD Typu 6 do tabeli *DimCustomer*

Źródło: opracowanie własne.

Na rysunku 7.23 przedstawiono przykład ilustrujący zmianę zawartości tabeli wymiaru *DimCustomer*, dla której zastosowano SCD Typu 6, w momencie wykrycia w systemie transakcyjnym dla klienta o kluczu biznesowym CS345 nowych wartości opisujących jego roczne dochody, tj. innych niż te występujące dotychczas w kolumnach *CurrentAnnualIncomeValue* i *CurrentAnnualIncomeLevel* tabeli *DimCustomer*.

W pierwszej tabeli znajduje się rekord klienta zawierający dane przed zmianą wartości jego rocznych dochodów. W momencie wykrycia ich zmiany z 350 000 do poziomu 475 000 w tabeli *DimCustomer* następuje:

- nadpisanie w rekordzie o *CustomerBusinessKey* równym CS345 i *CustomerKey* równym 9 wartości w kolumnach *CurrentAnnualIncomeLevel* i *CurrentAnnualIncomeValue* nowymi wartościami, odpowiednio 4 i 475 000 (działanie SCD Typu 1);
- utworzenie nowego rekordu z tą samą wartością klucza biznesowego CS345 i nową wartością *CustomerKey* równą 99, w którym zapisane zostają aktualne

wartości opisujące poziom dochodów (kolumna *CurrentAnnualIncomeLevel*) i ich wartość (kolumna *CurrentAnnualIncomeValue*); dodatkowo w rekordzie tym zapisywane są daty obowiązywania nowych wartości (kolumny *EffectiveDate* i *ExpirationDate*) oraz wskaźnik bieżącego rekordu (kolumna *IsCurrent*), a w rekordzie archiwalnym, dla którego *CustomerKey* równa się 9, modyfikowane są wartości w kolumnach *ExpirationDate* i *IsCurrent* (działanie SCD Typu 2);

- zapisanie w nowym rekordzie w kolumnach *PreviousAnnualIncomeLevel* i *PreviousAnnualIncomeValue* aktualnych wartości odpowiednio z kolumn *CurrentAnnualIncomeLevel* i *CurrentAnnualIncomeValue* (działanie SCD Typu 3).

Rekord przed zmianą wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>	<i>PreviousAnnualIncomeLevel</i>	<i>PreviousAnnualIncomeValue</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>	<i>IsCurrent</i>
9	CS345	...	3	350 000,00	3	350 000,00	01-01-2022	31-12-9999	1

Rekord po pierwszej zmianie wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>	<i>PreviousAnnualIncomeLevel</i>	<i>PreviousAnnualIncomeValue</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>	<i>IsCurrent</i>
9	CS345	...	4	475 000,00	3	350 000,00	01-01-2022	31-12-2022	0
99	CS345	...	4	475 000,00	4	475 000,00	01-01-2023	31-12-9999	1

Rekord po drugiej zmianie wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>	<i>PreviousAnnualIncomeLevel</i>	<i>PreviousAnnualIncomeValue</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>	<i>IsCurrent</i>
9	CS345	...	5	590 000,00	3	350 000,00	01-01-2022	31-12-2022	0
99	CS345	...	5	590 000,00	4	475 000,00	01-01-2023	31-12-2023	0
999	CS345	...	5	590 000,00	5	590 000,00	01-01-2024	31-12-9999	1

Rys. 7.23. Przykład implementacji SCD Typu 6 dla tabeli *DimCustomer*

Źródło: opracowanie własne.

W momencie wystąpienia w systemie ERP kolejnej zmiany rocznych dochodów klienta o podanym kluczy biznesowym CS345 z wartości 475 000 na 590 000 w tabeli *DimCustomer* zostaną wykonane następujące działania:

- aktualizacja wartości w kolumnach *CurrentAnnualIncomeLevel* i *CurrentAnnualIncomeValue* we wszystkich rekordach, w których wartość klucza biznesowego jest równa CS345 (działanie SCD Typu 1);
- wstawienie nowego rekordu z wartością 999 w kolumnie *CustomerKey* i CS345 w kolumnie *CustomerBusinessKey*, wprowadzenie odpowiednich nowych wartości w kolumnach *CurrentAnnualIncomeLevel* i *CurrentAnnualIncomeValue*, ustawienie dat ważności tego rekordu i wskaźnika bieżącego rekordu, a w rekordzie historycznym, dla którego

CustomerKey równa się 99, modyfikacja daty końca jego ważności (kolumna *ExpirationDate*) i wskaźnika aktualności *IsCurrent* (działanie SCD Typu 2);

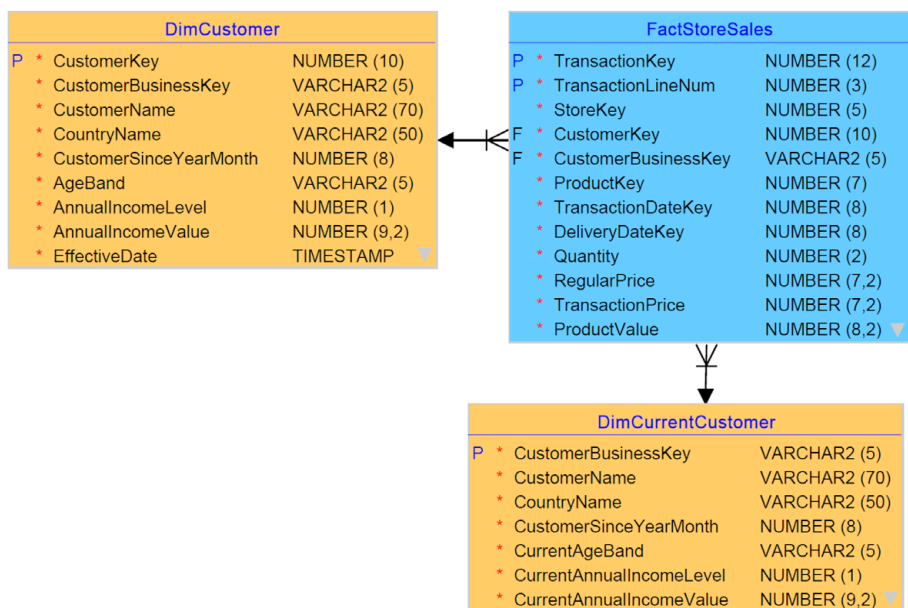
- modyfikacja w nowym rekordzie wartości w kolumnach *PreviousAnnualIncomeLevel* i *PreviousAnnualIncomeValue*, aby były one zgodne z odpowiednimi wartościami bieżącymi w tym rekordzie (działanie SCD Typu 3).

Przedstawiony przykład aktualizacji tabeli wymiaru o strukturze zgodnej z SCD Typu 6 może wydawać się dosyć skomplikowany. Jednak warto dostrzec w tym rozwiązaniu pewną istotną zaletę, którą jest znaczne skrócenie czasu wykonania zapytań mających na celu porównanie bieżących wartości atrybutów z ich wartościami archiwalnymi. Taką możliwość stwarza alokowanie w każdym rekordzie o tym samym kluczu biznesowym zarówno wartości aktualnej, jak i historycznej, do której przypisano znacznik czasowy jej obowiązywania.

Podstawową wadą modelowania SCD Typu 6 jest nadmierny wzrost wielkości tabeli wymiaru wynikający z większej liczby kolumn przeznaczonych do przechowywania danych archiwalnych oraz coraz większej liczby wierszy, w których będą zapisywane nowe wartości atrybutów. Problem ten staje się bardziej istotny, gdy dany wymiar jest opisany wieloma atrybutami, których wartości mogą się zmieniać w czasie i konieczne jest śledzenie tych zmian odrębnie dla każdego atrybutu. W takiej sytuacji tabela wymiaru staje się coraz szersza, gdyż dla każdego atrybutu konieczne jest alokowanie dwóch kolumn do przechowywania wartości bieżącej i archiwalnej.

Ostatnim sposobem modelowania wymiarów wolnozmiennych jest technika określana jako SCD Typu 7, polegająca na utworzeniu dodatkowej tabeli wymiaru, w której zapisane zostaną wyłącznie nowe (bieżące) wartości atrybutów, oraz powiększeniu dotychczasowej tabeli o rekordy z archiwalnymi wartościami atrybutów. Tego rodzaju podejście można interpretować jako podział tabeli z zaimplementowanym SCD Typu 6 na dwie odrębne tabele, z których pierwsza zawiera historyczne wartości atrybutów, a druga – ich wartości aktualne. Przykładowy model danych dla tabeli *DimCustomer*, dla której zastosowano SCD Typu 7, przedstawiono na rysunku 7.24. Podstawowa tabela *DimCustomer* będzie zawierać zarówno dane historyczne, jak i bieżące dotyczące danego klienta. Natomiast w tabeli *DimCurrentCustomer* znajdzie się dokładnie jeden rekord opisujący tego klienta z nowymi wartościami atrybutów.

Warto zauważyć, że pierwotna tabela *DimCustomer* ma strukturę charakterystyczną dla SCD Typu 2 i jest w relacji z tabelą faktu *FactStoreSales*, zdefiniowanej przy użyciu klucza surogatowego *CustomerKey*. Dodatkowa tabela *DimCurrentCustomer* musi także pozostawać w relacji z tabelą faktu. Związek ten jest zbudowany przy użyciu klucza biznesowego opisującego klienta (kolumna *CustomerBusinessKey*).



Rys. 7.24. Model danych po użyciu SCD Typu 2 do tabeli *DimCustomer*

Źródło: opracowanie własne.

Na rysunku 7.25 przedstawiono sposób modyfikacji zawartości tabel *DimCustomer* i *DimCurrentCustomer* w sytuacji wystąpienia zmian wartości opisujących roczne dochody klienta i ich poziom. W momencie wykrycia zmiany wartości dochodu z 350 000 do 475 000 następuje modyfikacja rekordu danego klienta (*CustomerBusinessKey* = CS345) w tabeli *DimCurrentCustomer* poprzez wprowadzenie w nim nowych wartości w kolumnach *CurrentAnnualIncomeLevel* i *CurrentAnnualIncomeValue*. Jednocześnie w tabeli *DimCustomer* wstawiany jest nowy rekord z bieżącymi wartościami *AnnualIncomeLevel* i *AnnualIncomeValue* oraz modyfikowane są odpowiednio daty w kolumnach *EffectiveDate* i *ExpirationDate* rekordów archiwalnego i bieżącego.

Rekord tabeli *DimCustomer* przed zmianą wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>AnnualIncomeLevel</i>	<i>AnnualIncomeValue</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>
9	CS345	...	3	350 000,00	01-01-2022	31-12-9999

Rekord tabeli *DimCurrentCustomer* przed zmianą bieżących wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerBusinessKey</i>	...	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>
CS345	...	3	350 000,00

Rekordy tabeli *DimCustomer* po pierwszej zmianie wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue*

<i>CustomerKey</i>	<i>CustomerBusinessKey</i>	...	<i>AnnualIncomeLevel</i>	<i>AnnualIncomeValue</i>	<i>EffectiveDate</i>	<i>ExpirationDate</i>
9	CS345	...	3	350 000,00	01-01-2022	31-12-2022
99	CS345	...	4	475 000,00	01-01-2023	31-12-9999

Rekord tabeli *DimCurrentCustomer* po zmianie dotychczasowych wartości w kolumnach *AnnualIncomeLevel* i *AnnualIncomeValue* tabeli *DimCustomer*

<i>CustomerBusinessKey</i>	...	<i>CurrentAnnualIncomeLevel</i>	<i>CurrentAnnualIncomeValue</i>
CS345	...	4	475 000,00

Rys. 7.25. Przykład implementacji SCD Typu 7 dla tabeli *DimCustomer*

Źródło: opracowanie własne.

Jedną z istotnych zalet SCD Typu 7 jest możliwość gromadzenia danych historycznych oddzielnie od bieżących, z zachowaniem pełnej historii ich zmian, co umożliwia łatwiejsze zarządzanie zawartością tych tabel. Obecność relacji między tabelą faktów i tabelą aktualnych wartości przyspiesza ponadto odnalezienie bieżącej wartości atrybutu wymiaru bez konieczności posługiwania się kluczem surogatowym i datą wskazującą aktualny rekord w tabeli historycznej. Nie bez znaczenia jest także uproszczenie procesu ETL, w którym generowane są dane zagregowane z wykorzystaniem danego wymiaru, gdyż najnowsze wartości atrybutów opisujących ten wymiar pobierane będą z tabeli aktualnych danych, a nie z tabeli archiwalnej, co znacząco skróci czas realizacji tego procesu.

W celu ułatwienia Czytelnikowi porównania cech wszystkich technik modelowania wymiarów wolnozmiennych zestawiono je w tabeli 7.1 z uwzględnieniem działań wykonywanych w tabeli wymiaru oraz ich wpływu na dane w tabeli faktów.

Tab. 7.1. Cechy charakterystyczne różnych typów SCD

Typ SCD	Operacja w tabeli wymiaru	Wpływ na tabelę faktu
Typ1	Nadpisanie wartości atrybutu	Rekordy powiązane tylko z bieżącą wartością atrybutu zapisaną w jednym wierszu podstawowej tabeli wymiaru
Typ 2	Dodanie nowego wiersza z bieżącą wartością atrybutu	Rekordy powiązane zarówno z bieżącą, jak i archiwalną wartością atrybutu, zapisaną w różnych wierszach tabeli wymiaru
Typ 3	Wstawienie nowych kolumn umożliwiających zapamiętanie bieżących i archiwalnych wartości atrybutu	Rekordy powiązane z bieżącą i archiwalną wartością atrybutu, zapisanymi w jednym wierszu podstawowej tabeli wymiaru
Typ 4	Utworzenie nowej tabeli wymiaru, w której zapisane będą wartości zmieniających się atrybutów	Rekordy powiązane z bieżącą wartością atrybutu zapisaną w odrębnej tabeli
Typ 5	Utworzenie dodatkowej tabeli wymiaru, w której zapisane będą aktualne wartości atrybutu, oraz powiązanie jej z podstawową tabelą wymiaru. Możliwa dodatkowa aktualizacja zawartości tabeli wymiaru z aktualnymi i archiwalnymi wartościami atrybutu	Rekordy powiązane z bieżącą wartością atrybutu w podstawowej tabeli wymiaru oraz z dodatkową tabelą wymiaru zawierającą aktualne i archiwalne wartości zmieniającego się atrybutu
Typ 6	Wstawienie nowych kolumn i wierszy w celu zapamiętania bieżących i archiwalnych wartości atrybutu oraz modyfikacja wszystkich wierszy w kolumnie zawierającej bieżącą wartość atrybutu	Rekordy powiązane z bieżącą i archiwalną wartością atrybutu, zapisaną w tym samym wierszu tabeli wymiaru, oraz z wartościami archiwalnymi, utrwalonymi w różnych wierszach tej tabeli
Typ 7	Utworzenie odrębnej tabeli do przechowywania tylko bieżącej wartości atrybutu. Pełna historia zmian wartości atrybutu zapisana w podstawowej tabeli wymiaru	Rekordy powiązane z bieżącą wartością atrybutu zapisaną w odrębnej tabeli oraz z wartościami archiwalnymi z podstawowej tabeli wymiaru

Źródło: opracowanie własne na podstawie [25].

Podsumowując prezentację różnych technik modelowania wymiarów wolnozmiennych, warto podkreślić istotność klucza biznesowego tabeli wymiaru w ich implementacji. Jego wysoka użyteczność wynika przede wszystkim z możliwości użycia tego rodzaju klucza do wykrywania różnic pomiędzy wartościami atrybutu w systemie transakcyjnym i hurtowni danych. Ponadto jest on wykorzystywany do identyfikacji tej samej instancji wymiaru mimo zmiany wartości opisujących ją atrybutów. Dlatego też konieczne jest staranne wyselekcjonowanie takiej cechy wymiaru, która zapewni łatwość i jednoznaczność jej interpretacji, a także niezmiennosc wartości niezależnie od źródła jej pochodzenia.

7.3. Hierarchie wymiaru

Identyfikując atrybuty wymiaru, a następnie modelując je w postaci kolumn tabeli wymiaru, należy wziąć pod uwagę możliwość ich wykorzystania podczas analizowania danych z coraz większą szczegółowością. Proces ten bardzo wspiera konstrukcja nazywana hierarchią wymiaru, umożliwiającą zorganizowanie jego atrybutów w strukturę drzewiastą. Każda gałąź tego drzewa, tj. poziom hierarchii, jest opisana adekwatnym atrybutem wymiaru, którego wartości determinują liczbę węzłów na danym poziomie.

Tworzenie hierarchii wymaga wskazania jej poziomów, zależności między nimi, a następnie przypisania każdemu z nich odpowiedniego atrybutu. Działania te mogą być realizowane w dwojaki sposób. Pierwsza strategia, określana jako Top-Down, polega na definicji najbardziej ogólnego poziomu, który będzie pierwszym w hierarchii, a następnie identyfikowaniu kolejnych podrzędnych poziomów i opisujących je atrybutów. Z kolei w podejściu Bottom-Up najpierw definiuje się najniższy poziom hierarchii, by w kolejnym kroku, dokonując agregacji danych z niższego poziomu, określić poziom nadrzędny i powtarzać to działanie aż do momentu dotarcia do poziomu najwyższego.

We wszystkich hierarchiach związki między poziomami muszą być zdefiniowane w taki sposób, aby każda wartość na wyższym poziomie była bezpośrednio powiązana z wartościami na podrzędnym poziomie, stanowiącymi naturalne i logiczne uszczegółowienie wartości nadrzędnej. Należy w tym miejscu podkreślić, że taka relacja nie musi dotyczyć każdego rodzaju hierarchii, co zostanie zobrazone podczas prezentacji tzw. hierarchii poszarpanej.

W zależności od modelu danych atrybuty opisujące kolejne poziomy hierarchii mogą znajdować się w jednej lub wielu tabelach wymiaru. Jeśli hierarchia stworzona jest dla wymiaru występującego w modelu gwiazdy, to wszystkie jej poziomy będą opisane atrybutami reprezentowanymi przez kolumny znajdujące się w tej samej tabeli wymiaru. W przypadku tworzenia hierarchii dla wymiaru w modelu płatka śniegu lub konstelacji faktów poziomy hierarchii mogą być charakteryzowane przez dane pochodzące z jednej lub wielu tabel, w zależności od stopnia ich normalizacji.

Definiowanie hierarchii jest bardzo ważnym etapem modelowania wielowymiarowego, gdyż jego rezultat wpływa na użyteczność zarówno samej hurtowni

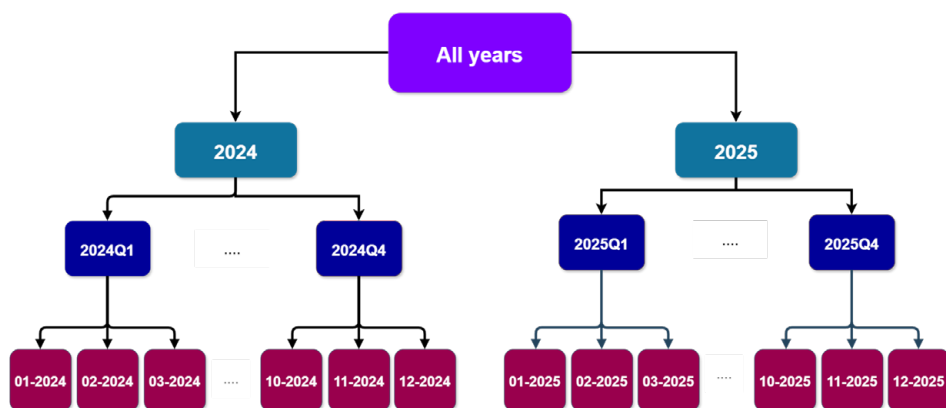
danych (data mart), jak i aplikacji analitycznej oraz umożliwia wykonywanie operacji analitycznych z różną szczegółowością, wynikającą z określonego poziomu takiej hierarchii. Posługiwanie się hierarchiami wymiarów pozwala na lepsze zrozumienie danych i zależności między nimi, tworzenie wielopoziomowych widoków na ten sam fakt, ułatwia też drążenie danych (od danych ogólnych do szczegółowych) oraz ich zwijanie (od danych szczegółowych do bardziej zagregowanych). Istotną korzyścią dla końcowego użytkownika biznesowego jest również możliwość tworzenia wielu hierarchii dla tego samego wymiaru, co zwiększa liczbę przekrojów informacyjnych dotyczących analizowanego zagadnienia. Innym pozytywnym aspektem wynikającym z obecności hierarchii jest wzrost efektywności zapytań, w których wykorzystywane są kolumny opisujące kolejne jej poziomy.

Niezależnie od rodzaju modelu danych można wyróżnić następujące rodzaje hierarchii wymiaru [15]:

- zrównoważona (ang. *balanced hierarchy*);
- niezrównoważona (ang. *unbalanced hierarchy*);
- poszarpana (ang. *ragged hierarchy*).

Najbardziej popularnym i powszechnym sposobem odwzorowania zależności pomiędzy atrybutami wymiaru jest hierarchia zrównoważona. Charakteryzuje ją występowanie takiej samej liczby poziomów w każdej gałęzi drzewa oraz istnienie bezpośredniej relacji wyłącznie między poziomem podrzędnym i bezpośrednio nadrzędnym. W tego typu hierarchii do opisu jej poziomów wykorzystuje się dane znajdujące się w tym samym wierszu (dla modelu gwiazdy) bądź też w wielu wierszach z różnych tabel (dla modelu płata śniegu lub konstelacji). Zazwyczaj wymiar, dla którego utworzono hierarchię tego rodzaju, opisuje fakt z dokładnością odpowiadającą jej najniższemu poziomowi, aczkolwiek jest również możliwe, by poziom ten odpowiadał pewnemu stopniowi uogólnienia danych z tabeli faktu. Przykład zrównoważonej hierarchii wymiaru czasu przedstawiono na rysunku 7.26.

Zaprezentowana hierarchia czasu ma trzy poziomy, zdefiniowane odpowiednio przez rok, kwartał i miesiąc. Każda z gałęzi ma tę samą głębokość. Pierwszy poziom jest charakteryzowany numerem roku kalendarzowego, drugi – numerem tegoż roku i nazwą kwartału, a trzeci – numerem miesiąca w roku i numerem roku. Konieczność tak szczegółowego doprecyzowania znaczenia kolejnych atrybutów wynika z możliwości występowania w opisie tego wymiaru również roku fiskalnego.



Rys. 7.26. Przykład hierarchii zrównoważonej dla wymiaru czasu

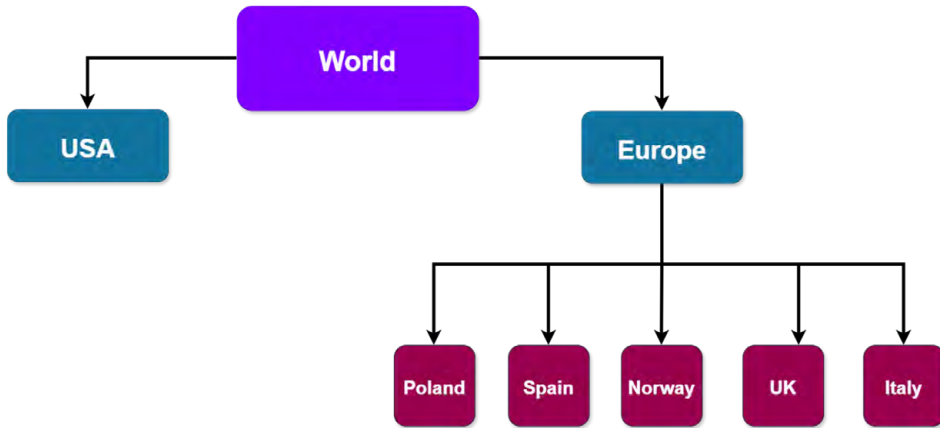
Źródło: opracowanie własne.

W hierarchii zrównoważonej do opisu jej poziomów wykorzystuje się dane zarówno z tego samego rekordu, jak i z wielu tabel. Ponieważ dla tego samego wymiaru można zdefiniować wiele hierarchii, na dodatek każdą o innej głębokości i strukturze, to analiza danych – czyli operacje rozwijania i zwiżania – będzie wykonywana z różnym stopniem szczegółowości. Istotną zaletą hierarchii zrównoważonej jest łatwość definiowania zapytań wykonujących agregacje i obliczenia na każdym jej poziomie. Ponadto efektywność jej użycia wzrasta, jeśli platforma bazodanowa wspiera możliwość tworzenia hierarchii jako obiektu w schemacie bazy danych.

Zastosowanie tylko trzech poziomów w przykładowej hierarchii nie oznacza, że w tabeli wymiaru czasu znajdują się kolumny zawierające dane dotyczące wyłącznie roku, kwartału czy miesiąca kalendarzowego, ale także opisujące tydzień, dzień, porę dnia, a nawet konkretne godziny. Jednak potrzeby informacyjne użytkownika końcowego mogą determinować hierarchię płytszą niż wynikałoby to z liczby atrybutów danego wymiaru. Warto w tym miejscu podkreślić, że dany atrybut nie musi należeć do żadnej hierarchii, może też być wykorzystany w opisie więcej niż jednej hierarchii tego samego wymiaru.

Kolejnym rodzajem drzewa hierarchicznego, stosowanego w praktyce modelowania wielowymiarowego, jest hierarchia nie zrównoważona. Charakteryzuje się ona występowaniem przynajmniej jednej gałęzi o mniejszej liczbie poziomów niż pozostałe. Oznacza to, że każda gałąź może mieć inną głębokość.

Na rysunku 7.27 przedstawiono przykład prostej hierarchii tego rodzaju dla wymiaru geograficznego.



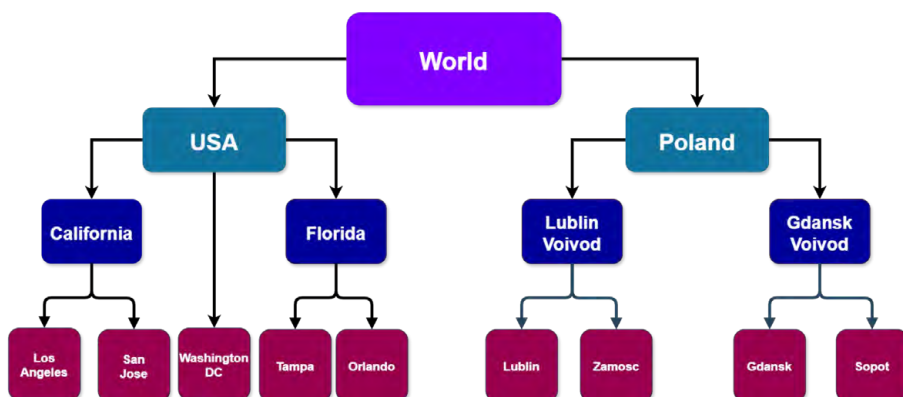
Rys. 7.27. Przykład hierarchii nie zrównoważonej dla wymiaru geograficznego

Źródło: opracowanie własne.

W hierarchii nie zrównoważonej kolejne poziomy nie są opisane danymi z kolumn tego samego rekordu, lecz z różnych wierszy. Dlatego też liczba poziomów takiej hierarchii nie jest określona liczbą kolumn tabeli wymiaru, lecz liczbą wierszy zawierających powiązane dane. Innym przykładem hierarchii nie zrównoważonej jest struktura przedstawiająca zależności pomiędzy dyrektorami, kierownikami sprzedaży i przedstawicielami handlowymi. W takim przypadku niektórzy kierownicy sprzedaży mogą być skojarzeni z dyrektorami nadrzędnymi w hierarchii, ale nie muszą istnieć podlegli im przedstawiciele handlowi, natomiast inni kierownicy mogą być osobami przełożonymi dla wybranych przedstawicieli handlowych.

Implementacja hierarchii nie zrównoważonej jest dosyć złożona w zapytaniach i prowadzi do obniżenia efektywności ich wykonania. Dlatego też często zaleca się jej naturalizację, czyli transformację do postaci kolumn tabeli wymiaru [4].

Ostatnim podstawowym rodzajem hierarchii wymiaru jest tzw. hierarchia poszarpana, która umożliwia modelowanie zależności między jego atrybutami w sytuacji, kiedy wartość atrybutu na poziomie podrzędnym nie jest powiązana z wartością atrybutu na poziomie bezpośrednio nadrzędnym, lecz z wartością atrybutu znajdującego się na wyższym poziomie tej hierarchii. Przykład hierarchii poszarpanej dla wymiaru geograficznego zobrazowano na rysunku 7.28.



Rys. 7.28. Przykład hierarchii poszarpanej dla wymiaru geograficznego

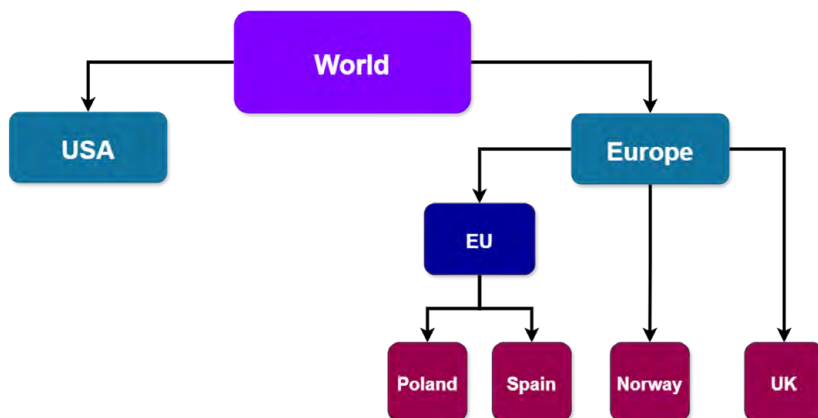
Źródło: opracowanie własne.

W zobrazowanej hierarchii występują trzy poziomy: kraj, region kraju i miasto. Większość miast jest przypisana do określonego regionu, tj. stanu, prowincji czy województwa. Jednak w USA miasto Washington DC, będące stolicą tego kraju, nie należy do żadnego stanu. Dlatego też w hierarchii tego wymiaru miasto to jest powiązane bezpośrednio z poziomem kraju.

W uzupełnieniu dotychczasowego opisu rodzajów hierarchii wymiaru warto jeszcze zaprezentować rozwiązanie będące połączeniem hierarchii nie-zrównoważonej i poszarpanej. Przykład taki przedstawiono na rysunku 7.29. W tym przypadku niezrównoważenie hierarchii wynika z większej liczby poziomów w gałęzi Europe niż w gałęzi USA. Ponadto w gałęzi Europe występuje dodatkowy poziom opisany wartością EU (European Union), do którego przypisane są niektóre kraje, jak Poland czy Spain. Jednak nie każdy kraj europejski, np. Norway czy UK, należy do Unii Europejskiej, dlatego też występuje tutaj zjawisko hierarchii poszarpanej.

W literaturze przedmiotu [15] wymienia się również inną klasyfikację hierarchii wymiaru, w której rozróżnia się hierarchie naturalne i sztuczne. Pierwsza z nich charakteryzuje się występowaniem poziomów zgodnych z istniejącymi w rzeczywistości. Przykładem tego typu hierarchii jest omówiona uprzednio zależność między krajem, regionem w kraju i miastem czy też rokiem, kwartałem i miesiącem kalendarzowym. Natomiast hierarchie sztuczne wynikają z potrzeb i preferencji biznesowych w odniesieniu do analizowanego zagadnienia. Tego

rodzaju hierarchia może wyglądać następująco: kategoria produktowa, podkategoria, producent i model produktu czy też rok, sezon i miesiąc w sezonie.



Rys. 7.29. Przykład hierarchii nie zrównoważonej i poszarpanej dla wymiaru geograficznego

Źródło: opracowanie własne.

Niezależnie od przyjętej klasyfikacji hierarchii i ich rodzaju wszystkie z nich mają jedną wspólną cechę – brak sposobności trwałego zapisania jej definicji w tabeli wymiaru. Ponieważ dla tego samego wymiaru można zdefiniować wiele hierarchii, a każda z nich może być opisana danymi z kolumn pochodzących z różnych tabel tego samego wymiaru, to nie ma możliwości, aby informacje o strukturze hierarchii utrwalić w takiej tabeli. Jednak niektóre bazy danych umożliwiają stworzenie w swoim schemacie obiektu będącego hierarchią (np. polecenie *CREATE HIERARCHY* w SQL Oracle), co znacząco skraca czas wykonania późniejszych zapytań. Definicja hierarchii wymiaru jest także generowana w narzędziach analitycznych, w których dokonuje się mapowania zależności pomiędzy poszczególnymi jej poziomami i opisuje je odpowiednimi kolumnami ze źródeł danych.

Podsumowując tematykę hierarchii wymiaru, należy podkreślić bardzo wysoką użyteczność tego rodzaju rozwiązania w procesach drążenia i związania danych, które są najczęściej wykonywanymi operacjami analitycznymi. Definiując strukturę poszczególnych tabel wymiarów i relacje między nimi, należy wziąć pod uwagę nie tylko informacyjne potrzeby biznesowe związane z wymaganiem opisem analizowanego zagadnienia, ale także aspekt wydajności zapytań, które będą generować dane na różnych poziomach szczegółowości.

7.4. Zadania do samodzielnego rozwiązania

Wykorzystując model danych zbudowany w wyniku realizacji zadań w podrozdziale 6.3, proszę:

1. Zmodyfikować model danych dotyczący faktu opisującego koszty związane z usunięciem awarii i uszkodzenia roweru (zadanie 1 w podrozdziale 6.3), wprowadzając niezbędne brakujące wymiary związane z rodzajem kosztów i rodzajem awarii lub uszkodzenia roweru.
2. Zastosować wymiar odgrywający rolę do opisu czasu wypożyczenia i czasu zwrotu roweru, określonego w transakcyjnej tabeli faktu utworzonej jako rezultat zadania 1 w podrozdziale 5.4 i reprezentującej pojedynczą usługę wypożyczenia roweru.
3. Dokonać analizy struktury istniejących tabel wymiarów i zmodyfikować dotychczasowy model danych, definiując odpowiednie zgodne i zewnętrzne tabele wymiarów.
4. Dokonać modyfikacji modelu danych opisujący wymiar kliencki w taki sposób, aby można było rejestrować historię zmian miasta przypisanego do klienta w jego profilu.
5. Zdefiniować hierarchię dla wymiaru opisującego stację rowerów, tj. miejsce wypożyczenia i zwrotu rowerów. W opisie jej poziomów należy uwzględnić aspekt prowadzenia usług wypożyczenia rowerów w skali globalnej. Proszę nazwać zrozumiale poszczególne poziomy tej hierarchii i wskazać kolumny z tabel wymiarów, które będą opisywać te poziomy. Przedstawić hierarchię w formie graficznej.

8. Eksploatacja danych w języku SQL

Wykonanie projektu analitycznej bazy danych, jej implementacja w docelowym środowisku bazodanowym i zasilenie danymi w procesach ETL to działania, które doprowadzą do zbudowania wartościowego źródła danych na temat różnych obszarów działania organizacji. Istnienie jednego, spójnego i wspólnego zasobu dla wszystkich potencjalnych beneficjentów tego rozwiązania sprawia, że każdy z nich ma dostęp do identycznego opisu tych samych zagadnień, jakkolwiek może tworzyć własne, profilowane przekroje informacyjne, które będą zaspokajać jego potrzeby informacyjne. Działanie to określane jest mianem eksploatacji danych, która poprzez transformację danych, ich analizę i wizualizację jej rezultatów prowadzi do pogłębienia wiedzy lub odkrywania nowej o przedmiotowym obszarze funkcjonowania firmy [8].

W eksploatacji danych często wykorzystuje się dedykowane narzędzia analityczne, np. klasy business intelligence (Power BI, Tableau Desktop) czy data science (KNIME, Rapid Miner) bądź też biblioteki dla języka Python (Pandas, Polars, NumPy, PySpark, Matplotlib czy Seaborn) i pakiety dla języka R, zawarte w środowisku tidyverse. Oferują one wiele zawansowanych wbudowanych rozwiązań ułatwiających obliczenia i wizualizację ich wyników. Analityczne przetwarzanie danych może być także realizowane przy wykorzystaniu technologii natywnych dla środowisk, w których składowane są dane. Mogą to być język MDX, używany m.in. w środowiskach Oracle

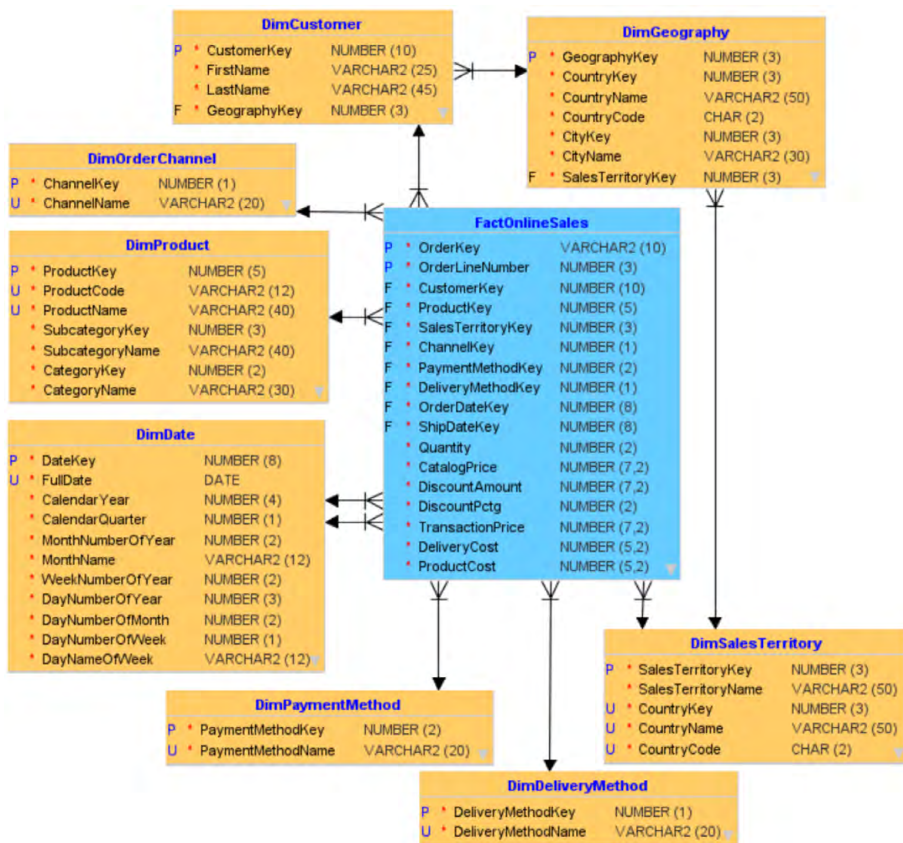
(Oracle Essbase) i Microsoft (SQL Server Analysis Services, Azure Analysis Services), język PL/SQL dla platformy Oracle czy PL/pgSQL dla PostgreSQL. Bardzo ciekawymi rozwiązaniami umożliwiającymi eksploatację danych są również języki GraphQL czy SPARQL. Obydwie te technologie zapewniają możliwość definiowania zapytań nie tylko do relacyjnych baz danych, lecz także do baz NoSQL, które są coraz częściej używane do składowania danych w systemach źródłowych.

W niniejszej publikacji zdecydowano się zaprezentować techniki eksploatacji danych, które można implementować w języku SQL. Uzasadnieniem takiego wyboru jest powszechność stosowania tego języka w zarządzaniu każdą bazą danych, w której dane składowane są w architekturze wykorzystującej tabele i relacje między nimi. W literaturze przedmiotu [30] także podkreśla się korzyści użycia języka SQL w eksploatacji danych, wskazując, że zawiera on wiele przydatnych funkcji i wyrażeń, które mogą przyspieszyć i zwiększyć dokładność analiz.

Wybór języka SQL podyktowany był również możliwością jego wykorzystania także w chmurowych hurtowniach danych, takich jak Amazon Redshift (<https://aws.amazon.com/redshift/>), Snowflake (<https://www.snowflake.com/en/product/platform/>), Google BigQuery (<https://cloud.google.com/bigquery>) czy Microsoft Azure Synapse Analytics (<https://azure.microsoft.com/pl-pl/products/synapse-analytics>). Są to rozwiązania analityczne należące do światowej czołówki, dlatego też Autorzy zachęcają Czytelnika do odrębnego zapoznania się z tymi technologiami.

W przypadku hurtowni danych struktura tych baz jest dostosowana do potrzeb analitycznego przetwarzania określanego jako OLAP (*Online Analytical Processing*), co zapewnia wysoką wydajność zapytań kierowanych do takiego źródła danych. Zapytania te wykorzystują agregaty danych, które mogą być zbudowane np. za pomocą klauzuli *GROUP BY*. Obliczenia wykonywane na takich agregatach przy użyciu funkcji *COUNT*, *SUM*, *AVG*, *MIN* czy *MAX* są identyczne w swojej naturze jak w przypadku bazy relacyjnej, jednak czas ich wykonania w hurtowni danych jest znacznie krótszy niż w środowisku relacyjnym, co wynika z odmiennego modelu danych, zastosowanego podczas projektowania struktury tabel i związków między nimi.

W niniejszym rozdziale omówiono różne techniki eksploatacji danych w języku SQL, a sposób ich implementacji zobrazowano za pomocą definicji zapytań kierowanych do przykładowej analitycznej bazy danych, wspomagającej decyzje w obszarze sprzedaży online. Fizyczny model danych tej bazy przedstawiono na rysunku 8.1.



Rys. 8.1. Fizyczny model danych przykładowej analitycznej bazy danych

Źródło: opracowanie własne.

Znaczenie danych w poszczególnych tabelach jest następujące:

- *FactOnlineSales* – pojedyncza pozycja zamówienia online;
- *DimCustomer* – klient dokonujący zamówienia online;
- *DimSalesTerritory* – kraj, z którego pochodzi zamówienie;
- *Dim Geography* – miasto, z którego pochodzi klient;
- *DimDeliveryMethod* – metoda dostawy zamówionych produktów;

- *DimChannel* – kanał sprzedaży online, z którego pochodzi zamówienie;
- *DimProduct* – produkt zamówiony przez klienta;
- *DimDate* – dzień złożenia zamówienia lub wysłania produktów;
- *DimPaymentMenthod* – metoda płatności użyta do opłacenia zamówienia.

Każdy rekord tabeli faktu zawiera kolumny miar, z których dwie, tj. *TransactionPrice* i *Quantity*, będą używane w dalszych przykładach.

8.1. Tworzenie kostek analitycznych – ROLLUP i CUBE

Grupowanie danych przy wykorzystaniu klauzuli *GROUP BY* w instrukcji *SELECT*, a następnie użycie funkcji agregacji *COUNT*, *SUM*, *MIN*, *MAX* czy *AVG* dokonujących kalkulacji z użyciem tych danych prowadziło do wygenerowania wyniku charakterystycznego dla całej takiej grupy wierszy. Liczba takich grup, a tym samym liczba wyników obliczeń, jest równa liczba unikalnych wartości kolumny, będącej kryterium grupowania, jeśli po klauzuli *GROUP BY* wskazano tylko jedną kolumnę. Obliczenia te są wykonywane na najniższym poziomie szczegółowości wynikającym z liczby kolumn umieszczonych po klauzuli grupowania. Jednak w rzeczywistości często zachodzi potrzeba, aby w zbiorze wynikowym znalazły się również wyniki kalkulacji na innych, nadrzędnych poziomach szczegółowości. W takim przypadku podczas określania kryteriów grupowania stosuje się dodatkową klauzulę *ROLLUP*, która umożliwi wygenerowanie podsumowań na innych niż domyślny poziomach grupowania danych. Ogólną składnię polecenia *SELECT* zawierającego klauzulę *ROLLUP* przedstawiono na rysunku 8.2.

```
1 | SELECT column_1, column_2, aggregate_function(column_3)
2 | FROM data_source
3 | GROUP BY ROLLUP(column_1, column_2);
```

Rys. 8.2. Składnia instrukcji *SELECT* z klauzulą *ROLLUP*

Źródło: opracowanie własne.

Sposób wykonywania obliczeń w takim zapytaniu wygląda następująco:

- w pierwszym kroku funkcja agregacji operuje na najbardziej szczegółowych grupach wierszy, utworzonych za pomocą wszystkich kolumn obecnych po klauzuli *GROUP BY*;

- następnie usuwana jest druga kolumna z definicji grupy i ponownie ta sama funkcja agregacji wykonuje obliczenia;
- ostatnia kalkulacja jest wykonywana na całym zbiorze wierszy określonym po klauzuli *FROM*.

Przykład użycia klauzuli *ROLLUP* przedstawiono na rysunku 8.3.

```

1 SELECT COALESCE(TO_CHAR(calendaryear), 'All years') AS Year,
2        COALESCE(channelname, 'All channels') AS Channel,
3        COUNT(DISTINCT orderkey) AS Orders_Num
4 FROM dimorderchannel doc INNER JOIN factonlinesales fos
5      ON doc.channelkey = fos.channelkey
6      INNER JOIN dimdate dd
7      ON dd.datekey = fos.orderdatekey
8 GROUP BY ROLLUP(TO_CHAR(calendaryear), channelname) ;

```

Rys. 8.3. Przykład użycia klauzuli *ROLLUP* w zapytaniu

Źródło: opracowanie własne.

W zapytaniu tym dodatkowo zastosowano funkcję *COALESCE* w celu konwersji wartości *NULL*, która wystąpi w kolumnie *Channel* podczas grupowania według roku oraz na poziomie całego przetwarzanego zbioru danych. Rezultat wykonania zapytania (rys. 8.4) prezentuje liczbę zamówień w kolejnych latach dla poszczególnych kanałów sprzedażowych.

YEAR	CHANNEL	ORDERS_NUM
2021	Mobile channel	1714
2021	Web channel	1748
2022	Mobile channel	2910
2022	Web channel	2864
2023	Mobile channel	2783
2023	Web channel	2778
2024	Mobile channel	4521
2024	Web channel	4505
2021	All channels	3462
2022	All channels	5774
2023	All channels	5561
2024	All channels	9026
All years	All channels	23823

Rys. 8.4. Rezultat zapytania z rysunku 8.3 wykorzystującego klauzulę *ROLLUP*

Źródło: opracowanie własne.

W przytoczonym przykładzie zapytania klauzula *ROLLUP* obejmowała swoim zakresem wszystkie kolumny będące kryterium grupowania. Istnieje jednak możliwość tworzenia grup tylko na wybranych poziomach, co skutkuje wykonywaniem obliczeń na mniejszej liczbie grup danych niż wynika to z użytych kryteriów grupowania. Przykład takiego zapytania zaprezentowano na rysunku 8.5.

```

1 SELECT calendaryear AS Year,
2    COALESCE(countryname, 'All countries') AS Country,
3    COALESCE(channelname, 'All channels') AS Channel,
4    COUNT(DISTINCT orderkey) AS Orders_Num
5 FROM dimorderchannel doc
6    INNER JOIN factonlinesales fos
7        ON doc.channelkey = fos.channelkey
8    INNER JOIN dimdate dd
9        ON dd.datekey = fos.orderdatekey
10   INNER JOIN dimsalesterritory dst
11       ON dst.salesterritorykey = fos.salesterritorykey
12 WHERE salesterritoryname = 'NACA' AND calendaryear IN (2023, 2024)
13 GROUP BY calendaryear, countryname, ROLLUP(channelname) ;

```

Rys. 8.5. Zapytanie ze zmniejszoną liczbą poziomów obliczeń dla klauzuli *ROLLUP*

Źródło: opracowanie własne.

Analizując rezultat wykonania tego zapytania (rys. 8.6), można zauważyć, że nie występuje w nim poziom określający rok, wszystkie kraje i wszystkie kanały oraz poziom całej tabeli źródłowej. Jest to spowodowane zastosowaniem klauzuli *ROLLUP* tylko do kolumny *ChannelName*.

YEAR	COUNTRY	CHANNEL	ORDERS_NUM
2023	Canada	Mobile channel	499
2023	Canada	Web channel	547
2023	United States	Mobile channel	430
2023	United States	Web channel	388
2024	Canada	Mobile channel	905
2024	Canada	Web channel	933
2024	United States	Mobile channel	675
2024	United States	Web channel	703
2023	Canada	All channels	1046
2023	United States	All channels	818
2024	Canada	All channels	1838
2024	United States	All channels	1378

Rys. 8.6. Rezultat wykonania zapytania z rysunku 8.5

Źródło: opracowanie własne.

Innym sposobem grupowania danych podczas wykonywania obliczeń analitycznych jest użycie klauzuli *CUBE*, która spowoduje wygenerowanie wszystkich możliwych kombinacji wartości występujących w kolumnach będących kryterium grupowania oraz dodatkowe podsumowania na poszczególnych poziomach szczegółowości danych. Ogólna składnia zapytania, w którym wykorzystuje się klauzulę *CUBE*, wygląda tak jak na rysunku 8.7.

```
1 SELECT column_1, column_2, aggregate_function(column_3)
2 FROM data_source
3 GROUP BY CUBE(column_1, column_2);
```

Rys. 8.7. Składnia instrukcji *SELECT* z klauzulą *CUBE*

Źródło: opracowanie własne.

Działania w takim zapytaniu wykonywane są w następujący sposób:

- w pierwszym kroku generowane są wszystkie możliwe kombinacje wartości wszystkich kolumn zdefiniowanych jako kryteria grupowania;
- w kolejnym etapie funkcja agregacji wykonuje obliczenia w każdej grupie wierszy opisanej pojedynczą kombinacją wartości utworzoną w poprzednim kroku;
- w ostatnim kroku kalkulacja jest wykonywana na całym zbiorze wierszy określonym po klauzuli *FROM*.

Przykładowe zapytanie obrazujące sposób użycia klauzuli *CUBE* przedstawiono na rysunku 8.8. Ma ono niemal identyczną postać jak to umieszczone na rysunku 8.3, jednak tutaj zastosowano klauzulę *CUBE*. W jego zbiorze wynikowym (rys. 8.9) pojawiły się dwa nowe wiersze: *All years* i *Mobile channel* oraz *All years* i *Web channel*, reprezentujące wszystkie lata i każdy z kanałów sprzedażowych.

```
1 SELECT COALESCE(TO_CHAR(calendaryear), 'All years') AS Year,
2        COALESCE(channelname, 'All channels') AS Channel,
3        COUNT(DISTINCT orderkey) AS Orders_Num
4 FROM dimorderchannel doc INNER JOIN factonlinesales fos
5      ON doc.channelkey = fos.channelkey
6      INNER JOIN dimdate dd
7      ON dd.datekey = fos.orderdatekey
8 GROUP BY CUBE(TO_CHAR(calendaryear), channelname);
```

Rys. 8.8. Przykład użycia klauzuli *CUBE* w zapytaniu

Źródło: opracowanie własne.

YEAR	CHANNEL	ORDERS_NUM
2021	Mobile channel	1714
2022	Mobile channel	2910
2023	Mobile channel	2783
2024	Mobile channel	4521
2021	Web channel	1748
2022	Web channel	2864
2023	Web channel	2778
2024	Web channel	4505
All years	Mobile channel	11928
All years	Web channel	11895
2021	All channels	3462
2022	All channels	5774
2023	All channels	5561
2024	All channels	9026
All years	All channels	23823

Rys. 8.9. Rezultat zapytania z rysunku 8.8 wykorzystującego klauzulę *CUBE*

Źródło: opracowanie własne.

Taki sam rezultat uzyska się przy dwukrotnym użyciu w zapytaniu klauzuli *ROLLUP*, co ilustruje kod na rysunku 8.10.

```

1 SELECT COALESCE(TO_CHAR(calendaryear), 'All years') AS Year,
2         COALESCE(channelname, 'All channels') AS Channel,
3         COUNT(DISTINCT orderkey) AS Orders_Num
4 FROM dimorderchannel doc INNER JOIN factonlinesales fos
5      ON doc.channelkey = fos.channelkey
6      INNER JOIN dimdate dd
7      ON dd.datekey = fos.orderdatekey
8 GROUP BY ROLLUP(TO_CHAR(calendaryear)), ROLLUP(channelname) ;

```

Rys. 8.10. Alternatywne użycie klauzuli *ROLLUP* zamiast *CUBE*

Źródło: opracowanie własne.

Liczba dodatkowych wierszy, tworzonych w wyniku użycia klauzuli *CUBE*, zależy od liczby kryteriów grupowania, liczby unikalnych wartości dla każdego z nich oraz liczby kolumn będących parametrami tej funkcji. Dlatego też można uznać, że *CUBE* – w porównaniu z *ROLLUP* – dostarcza znacznie szerszego widoku danych. Na rysunku 8.11 przedstawiono zapytanie, w którym działanie klauzuli *CUBE* ograniczono wyłącznie do kolumny *CountryName*.

```

1 SELECT calendaryear AS Year,
2     COALESCE(countryname, 'All countries') AS Country,
3     COALESCE(channelname, 'All channels') AS Channel,
4     COUNT(DISTINCT orderkey) AS Orders_Num
5 FROM dimorderchannel doc
6     INNER JOIN factonlinesales fos
7         ON doc.channelkey = fos.channelkey
8     INNER JOIN dimdate dd
9         ON dd.datekey = fos.orderdatekey
10    INNER JOIN dimsalesterritory dst
11        ON dst.salesterritorykey = fos.salesterritorykey
12 WHERE salesterritoryname = 'NACA' AND calendaryear IN (2023, 2024)
13 GROUP BY calendaryear, CUBE(countryname), channelname ;

```

Rys. 8.11. Zastosowanie klauzuli *CUBE* do wybranych kryteriów grupowania

Źródło: opracowanie własne.

Taka konstrukcja polecenia spowodowała, że w zbiorze wynikowym, zobrazowanym na rysunku 8.12, nie znalazły się podsumowania dla wszystkich lat i wszystkich kanałów sprzedażowych, natomiast występują podsumowania dla wszystkich krajów w poszczególnych latach.

YEAR	COUNTRY	CHANNEL	ORDERS_NUM
2023	Canada	Mobile channel	499
2023	United States	Mobile channel	430
2023	Canada	Web channel	547
2023	United States	Web channel	388
2024	Canada	Mobile channel	905
2024	United States	Mobile channel	675
2024	Canada	Web channel	933
2024	United States	Web channel	703
2023	All countries	Mobile channel	929
2023	All countries	Web channel	935
2024	All countries	Mobile channel	1580
2024	All countries	Web channel	1636

Rys. 8.12. Zapytanie ze zmniejszoną liczbą poziomów obliczeń dla klauzuli *CUBE*

Źródło: opracowanie własne.

Na podstawie zaprezentowanych przykładów można zauważyć, że klauzule *ROLLUP* i *CUBE* znacząco rozszerzają zakres analizy danych poprzez dostarczenie dodatkowych poziomów agregacji danych, na których wykonywane są obliczenia. Ponadto zmiana liczby parametrów tych funkcji oraz miejsca ich

zastosowania po klauzuli *GROUP BY* tworzy warunki do bardzo elastycznego kształtowania zawartości informacyjnej zbioru wynikowego.

Wykorzystanie klauzul *ROLLUP* i *CUBE* do generowania podsumowań wymaga zastosowania czytelnego opisu tych wierszy, w których te wartości zostaną umieszczone. Dotychczas we wszystkich przykładach stosowano funkcję *COALESCE*, która umożliwiała identyfikację takich wierszy poprzez kontrolę istnienia w nich wartości *NULL* w kolumnach określających poziom agregacji. Alternatywnym sposobem wykonania tego zadania jest użycie funkcji *GROUPING*. Funkcja ta zwraca wartość 1, jeśli wiersz wynikowy jest podsumowaniem na poziomie określonym przez parametr tej funkcji. W przeciwnym razie zwraca wartość 0. Przykład kodu SQL, w którym użyto funkcji *GROUPING*, przedstawiono na rysunku 8.13.

```
1 SELECT COALESCE(TO_CHAR(calendaryear), 'All years') AS Year,  
2         COALESCE(channelname, 'All channels') AS Channel,  
3         COUNT(DISTINCT orderkey) AS Orders_Num,  
4         GROUPING(channelname) AS SubTotal_Channel,  
5         GROUPING(TO_CHAR(calendaryear)) AS SubTotal_Year  
6 FROM dimorderchannel doc INNER JOIN factonlinesales fos  
7      ON doc.channelkey = fos.channelkey  
8      INNER JOIN dimdate dd  
9      ON dd.datekey = fos.orderdatekey  
10 GROUP BY ROLLUP(TO_CHAR(calendaryear), channelname) ;
```

Rys. 8.13. Zastosowanie funkcji *GROUPING* w zapytaniu analitycznym

Źródło: opracowanie własne.

Poprawność syntaktyczna tego polecenia wymaga, aby parametrem funkcji była kolumna znajdująca się po klauzuli *GROUP BY*. Rezultat wykonania zapytania zobrazowano na rysunku 8.14.

YEAR	CHANNEL	ORDERS_NUM	SUBTOTAL_CHANNEL	SUBTOTAL_YEAR
2021	Mobile channel	1714	0	0
2021	Web channel	1748	0	0
2022	Mobile channel	2910	0	0
2022	Web channel	2864	0	0
2023	Mobile channel	2783	0	0
2023	Web channel	2778	0	0
2024	Mobile channel	4521	0	0
2024	Web channel	4505	0	0
2021	All channels	3462	1	0
2022	All channels	5774	1	0
2023	All channels	5561	1	0
2024	All channels	9026	1	0
All years	All channels	23823	1	1

Rys. 8.14. Rezultat zastosowanie funkcji *GROUPING* w zapytaniu analitycznym

Źródło: opracowanie własne.

W kolumnie *Subtotal_Channel* w wierszach opisanych jako *All channels* dla kolejnych lat znajdują się wartości 1, gdyż wiersze te zawierają podsumowania dla wszystkich kanałów sprzedażowych. Natomiast wartość 1 w kolumnie *Subtotal_Year* w ostatnim wierszu oznacza, że reprezentuje on podsumowanie dla wszystkich lat.

W uzupełnieniu dotychczas zaprezentowanych treści na temat sposobów tworzenia agregatów analitycznych warto jeszcze omówić jedną metodę, w której użytkownik końcowy może dowolnie kształtować strukturę takiej kostki OLAP poprzez użycie klauzuli *GROUPING SETS* w instrukcji *SELECT*. Jej ogólną składnię w takim przypadku przedstawiono na rysunku 8.15.

```

1 SELECT column_1, column2, aggregate_function(column_3)
2 FROM data_source
3 GROUP BY
4   GROUPING SETS ( (column_1, column_2),
5                   (column_1),
6                   (column_2),
7                   ()
8                   );

```

Rys. 8.15. Składnia polecenia *SELECT* z klauzulą *GROUPING SETS*

Źródło: opracowanie własne.

Obecność klauzuli *GROUPING SETS* umożliwia określenie różnych struktur grup wierszy, w których wykonywane będą obliczenia za pomocą funkcji agregacji. Grupy te mogą być zdefiniowane przez grupy kolumn, pojedyncze kolumny bądź też całą tabelę. W przykładzie na rysunku 8.16 zaimplementowano taki scenariusz, w którym powstają cztery rodzaje grup, tj.: według roku i kanału sprzedaży (linia 9 kodu), według roku (linia 10), według kanału sprzedaży (linia 11) i cała tabela (linia 12). W obrębie tych grup wyznacza się liczbę zamówień (linia 3).

```

1 SELECT COALESCE(TO_CHAR(calendaryear), 'All years') AS Year,
2         COALESCE(channelname, 'All channels') AS Channel,
3         COUNT(DISTINCT orderkey) AS Orders_Num
4 FROM dimorderchannel doc INNER JOIN factonlinesales fos
5         ON doc.channelkey = fos.channelkey
6         INNER JOIN dimdate dd
7         ON dd.datekey = fos.orderdatekey
8 WHERE calendaryear IN (2023, 2024)
9 GROUP BY GROUPING SETS ( (TO_CHAR(calendaryear), channelname),
10                          (TO_CHAR(calendaryear)),
11                          (channelname),
12                          ()
13                          )
14 ORDER BY 1, 3 ;

```

Rys. 8.16. Zastosowanie klauzuli *GROUPING SETS* w zapytaniu analitycznym

Źródło: opracowanie własne.

W wyniku wykonania zapytania otrzymano dane przedstawione na rysunku 8.17. Zastosowanie klauzuli *GROUPING SETS* spowodowało wyznaczenie wartości funkcji *COUNT* dla poszczególnych lat i kanałów (wiersze 1, 2, 4 i 5), wszystkich kanałów w kolejnych latach (wiersze 3 i 6), wszystkich lat w poszczególnych kanałach (wiersze 7 i 8) oraz dla całego zbioru źródłowego (wiersz 9).

	YEAR	CHANNEL	ORDERS_NUM
1	2023	Web channel	2778
2	2023	Mobile channel	2783
3	2023	All channels	5561
4	2024	Web channel	4505
5	2024	Mobile channel	4521
6	2024	All channels	9026
7	All years	Web channel	7283
8	All years	Mobile channel	7304
9	All years	All channels	14587

Rys. 8.17. Rezultat zastosowanie klauzuli *GROUPING SETS* w zapytaniu analitycznym

Źródło: opracowanie własne.

Przedstawione w tym podrozdziale sposoby tworzenia grup wierszy znacznie rozszerzają – w porównaniu z użyciem tylko klauzuli *GROUP BY* – zakresy danych, w których można wykonywać obliczenia analityczne. Szczególnie wartościowe wydają się podsumowania tworzone dla różnych kolumn, będących kryteriami grupowania, gdyż w ten sposób można istotnie zwiększyć użyteczność otrzymanych wyników zapytań w procesach decyzyjnych.

8.2. Partycjonowanie danych – *OVER* i *PARTITION BY*

W dotychczas prezentowanych przykładach analiza danych wymagała stworzenia agregatów analitycznych za pomocą klauzuli *GROUP BY*. Agregaty te określały zakres danych, na których wykonywano operacje z użyciem funkcji agregacji (np. *COUNT* czy *SUM*). Istnieje jednak możliwość zdefiniowania dodatkowych podzbiorów danych, które będą podlegać analizie, przy użyciu klauzul *OVER* i *PARTITION BY*. Zbiory te tworzą pewnego rodzaju zakresy danych wyznaczających obszar wykonywania kalkulacji. Klauzula *OVER* pozwala zdefiniować tzw. okna danych, w obrębie których można wykonywać dalsze operacje partycjonowania danych lub kalkulacje z użyciem funkcji agregacji. Zastosowanie tych klauzul w poleceniu *SELECT* umożliwi wykonanie m.in. takich operacji analitycznych, jak wskazanie największej wartości sprzedaży w roku, średniej wartości sprzedaży we wskazanym okresie czasu czy też wyznaczenie udziału miesięcznej wartości sprzedaży w rocznej wartości sprzedaży.

Podstawową składnię polecenia *SELECT*, w którym zostaną użyte klauzule *OVER* i *PARTITION*, przedstawiono na rysunku 8.18.


```

1 SELECT column_1, column_2,
2     window_function(col_parameter) OVER (PARTITION BY column_1)
3 FROM data_source
4 WHERE ...

```

Rys. 8.18. Składnia polecenia *SELECT* z użyciem klauzul *OVER* i *PARTITION BY*

Źródło: opracowanie własne.

Na liście kolumn polecenia *SELECT* należy wskazać rodzaj funkcji okna (*window_function*), wykonującej obliczenia z użyciem kolumny (*col_parameter*) znajdującej się w źródle danych (*data_source*). Następnie po klauzuli *OVER* definiuje się zbiór lub zbiory danych (*PARTITION BY*), w których wskazana funkcja będzie realizować kalkulacje. Do wykonywania kalkulacji w oknach i partycjach danych używa się funkcji okna, których nazwy są identyczne jak nazwy funkcji agregacji, stosowanych podczas grupowania wierszy za pomocą klauzuli *GROUP BY*. Różnica między tymi typami funkcji polega na tym, że funkcje okna wykonują obliczenia dla każdego wiersza okna i partycji, podczas gdy funkcje agregacji operują na całej grupie rekordów.

Przykład użycia klauzul tworzących okno i partycje danych przedstawiono na rysunku 8.19. Zdefiniowano w nim dwa okna danych. Pierwsze z nich (linia 2) zawiera partycje składające się z wierszy dotyczących tego samego roku. Drugie okno (linia 3) obejmuje swym zakresem wszystkie rekordy utworzone podzapytaniem. Następnie w każdym z tych okien zastosowano funkcję *SUM* w celu wyznaczenia wartości sprzedaży.

```

1 SELECT Year,
2     ROUND(SUM(Sales) OVER (PARTITION BY Year),0)/1000 AS Y_Sales,
3     ROUND(SUM(Sales) OVER(),0)/1000 AS All_Y_Sales
4 FROM (
5     SELECT calendaryear Year, SUM(transactionprice*quantity) AS Sales
6     FROM factonlinesales fos INNER JOIN dimdate dd
7         ON fos.orderdatekey= dd.datekey
8     GROUP BY calendaryear
9 )
10 ORDER BY 1 ;

```

Rys. 8.19. Przykład definiowania okna danych przy użyciu klauzuli *OVER*

Źródło: opracowanie własne.

Rezultat wykonania zapytania przedstawiono na rysunku 8.20.

YEAR	Y_SALES	ALL_Y_SALES
2021	4665.621	31429.013
2022	7106.274	31429.013
2023	7785.257	31429.013
2024	11871.862	31429.013

Rys. 8.20. Zbiór wynikowy zapytania z rysunku 8.19

Źródło: opracowanie własne.

Powtarzalność tych samych wartości w kolumnie *All_Y_Sales* wynika z faktu, że dla każdego roku wyznaczana jest wartość sprzedaży na podstawie pełnego zbioru danych, tj. zamówień ze wszystkich lat. Natomiast wartości w kolumnie *Y_Sales* kalkulowane są tylko dla zamówień z danego roku.

W tym samym poleceniu *SELECT* można wymienić wiele różnych funkcji okna, które będą wykonywać kalkulacje z użyciem wierszy znajdujących się w zdefiniowanych podzbiorach danych. Na rysunku 8.21 zilustrowano przykład, w którym kalkulacje wykonywane są za pomocą funkcji *SUM* (linie 2 i 3) i *AVG* (linia 4). Kod ten generuje dane określające: wartość sprzedaży w danym kwartale roku (*Q_Sales*), kwartalną liczbę klientów (*Q_Cust*), całkowitą roczną wartość sprzedaży (*Y_Sales*), liczbę klientów w roku (*Y_Cust*) oraz średnią liczbę klientów w kwartale danego roku (*Avg_Q_Cust*).

```

1 SELECT Year, Quarter, ROUND(Sales,0)/1000 Q_Sales, CustNum Q_Cust,
2     ROUND(SUM(Sales) OVER (PARTITION BY Year),0)/1000 AS Y_Sales,
3     SUM(CustNum) OVER (PARTITION BY Year) AS Y_Cust,
4     AVG(CustNum) OVER (PARTITION BY Year) AS Avg_Q_Cust
5 FROM (
6     SELECT calendaryear Year, calendarquarter Quarter,
7         SUM(transactionprice*quantity) AS Sales,
8         COUNT(DISTINCT customerkey) AS CustNum
9     FROM factonlinesales fos INNER JOIN dimdate dd
10    ON fos.orderdatekey= dd.datekey
11    WHERE calendaryear = 2024
12    GROUP BY calendaryear, calendarquarter
13 )
14 ORDER BY 1 ;

```

Rys. 8.21. Zastosowanie wielu funkcji agregacji w oknie danych

Źródło: opracowanie własne.

Wynik wykonania tego zapytania uwidoczniiono na rysunku 8.22.

YEAR	QUARTER	Q_SALES	Q_CUST	Y_SALES	Y_CUST	AVG_Q_CUST
2024	1	2084.346	1503	11871.862	8604	2151
2024	2	4667.85	2947	11871.862	8604	2151
2024	3	4467.619	3637	11871.862	8604	2151
2024	4	652.046	517	11871.862	8604	2151

Rys. 8.22. Rezultat wykonania kodu z rysunku 8.21

Źródło: opracowanie własne.

W przedstawionych dotychczas przykładach partycjonowanie danych ze zbioru wynikowego podzapytania odbywało się z użyciem pojedynczej kolumny, co skutkowało wykonywaniem obliczeń wyłącznie w obrębie jednorodnych partycji. Istnieje jednak możliwość, by operację tę realizować w partycjach zdefiniowanych przez różne kolumny dostępne w tabeli źródłowej. Przykładem takiego rozwiązania jest kod zaprezentowany na rysunku 8.23.

```

1 SELECT Quarter, Month, ROUND(Sales,0)/1000 M_Sales,
2     ROUND(SUM(Sales) OVER (PARTITION BY Quarter),0)/1000 AS Q_Sales,
3     ROUND(SUM(Sales) OVER (PARTITION BY Year),0)/1000 AS Y_Sales
4 FROM (
5     SELECT calendaryear Year, calendarquarter Quarter,
6           monthnumberofyear Month,
7           SUM(transactionprice*quantity) AS Sales,
8           COUNT(DISTINCT customerkey) AS CustNum
9     FROM factonlinesales fos INNER JOIN dimdate dd
10    ON fos.orderdatekey= dd.datekey
11    WHERE calendaryear = 2024
12    GROUP BY calendaryear, calendarquarter, monthnumberofyear
13 )
14 ORDER BY 1 ;

```

Rys. 8.23. Definiowanie wielu partycji i rodzajów kalkulacji w obrębie okna

Źródło: opracowanie własne.

W tabeli generowanej przez podzapytanie znajdują się kolumny opisujące rok, kwartał, miesiąc, wartość sprzedaży i liczbę klientów (dane dotyczą tylko roku 2024). W zapytaniu nadrzędnym wykonano kalkulacje z wykorzystaniem partycji utworzonej według kwartału (linia 2) oraz według roku (linia 3). W obydwu tych partycjach za pomocą funkcji *SUM* obliczono wartość sprzedaży w kwartale (*Q_Sales*) i roku (*Y_Sales*). Dodatkowo w zbiorze wynikowym znajdują się także informacje o wielkości sprzedaży dla każdego miesiąca (*M_Sales*).

Wynik wykonania zapytania definiującego odrębne partycje danych i wykonywane w nich kalkulacje przedstawiono na rysunku 8.24.

QUARTER	MONTH	M_SALES	Q_SALES	Y_SALES
1	1	502.758	2084.346	11871.862
1	2	610.584	2084.346	11871.862
1	3	971.004	2084.346	11871.862
2	4	1082.719	4667.85	11871.862
2	5	1512.922	4667.85	11871.862
2	6	2072.209	4667.85	11871.862
3	7	1009.845	4467.619	11871.862
3	8	2297.984	4467.619	11871.862
3	9	1159.79	4467.619	11871.862
4	10	184.045	652.046	11871.862
4	11	231.948	652.046	11871.862
4	12	236.053	652.046	11871.862

Rys. 8.24. Rezultat wykonania polecenia z rysunku 8.23

Źródło: opracowanie własne.

Tworzenie grup wierszy przy użyciu klauzul *OVER* i *PARTITION BY* prowadzi do powstania nieposortowanych zbiorów danych. Jednak w analizie danych występują takie sytuacje, w których kolejność wierszy w podzbiorze wynikowym ma znaczenie – np. podczas obliczania w kolejnych wierszach wartości w sposób narastający. Inną sytuacją, wskazującą na konieczność posortowania danych przed wykonaniem obliczeń analitycznych, jest porównywanie wartości w bieżącym wierszu z wartością z wiersza poprzedzającego. Poprawne wykonanie obliczeń w wyżej opisanych przypadkach wymaga zastosowania klauzul *OVER* i *ORDER BY*. Dodatkowo po wykonaniu operacji sortowania istnieje możliwość wyznaczenia pierwszej, ostatniej lub n-tej wartości w takim zbiorze, odpowiednio za pomocą funkcji *FIRST_VALUE*, *LAST_VALUE* lub *NTH_VALUE*. Bardzo ciekawym obszarem wykorzystania tak posortowanych zbiorów danych jest także wyznaczenie tzw. średniej kroczącej w kolejnych okresach czasu.

Ogólną składnię polecenia *SELECT* z użyciem klauzul *OVER* i *ORDER BY* przedstawiono na rysunku 8.25.

```

1 | SELECT column_1, column_2,
2 |     window_function(col_parameter) OVER (ORDER BY column_1)
3 | FROM data_source
4 | WHERE ...

```

Rys. 8.25. Sortowanie danych w podzbiorze utworzonym za pomocą klauzuli *OVER*

Źródło: opracowanie własne.

Klauzula *ORDER BY* spowoduje podczas wykonywania zapytania posortowanie wierszy (domyślnie rosnąco) z tabeli źródłowej (*data_source*), natomiast funkcja okna (*window_function*) wykona obliczenia z użyciem wszystkich, kolejnych wierszy tej tabeli. Przykład zapytania wyznaczającego wartość sprzedaży w kolejnych latach oraz wartość narastającą zobrazowano na rysunku 8.26.

```

1 SELECT Year, OrdersNum AS Y_Orders,
2     SUM(OrdersNum) OVER (ORDER BY Year) AS Running_Y_Orders
3 FROM (
4     SELECT calendaryear AS Year,
5           COUNT(DISTINCT orderkey) AS OrdersNum
6     FROM factonlinesales fos INNER JOIN dimdate dd
7     ON fos.orderdatekey = dd.datekey
8     GROUP BY calendaryear
9 )
10 ORDER BY 1, 2;

```

Rys. 8.26. Sortowanie wierszy w oknie danych z użyciem jednej kolumny

Źródło: opracowanie własne.

Tabela źródłowa generuje dane na temat liczby zamówień (*OrdersNum*) w kolejnych latach. Następnie wiersze te są sortowane podczas wykonywania kalkulacji (linia 2) i dla każdego z nich obliczana jest suma wartości w kolumnie *OrdersNum* dla wiersza bieżącego i wszystkich wierszy poprzedzających. W rezultacie wynik w kolumnie *Running_Y_Orders* opisuje sumę narastającą liczby zamówień w kolejnych latach (rys. 8.27).

YEAR	Y_ORDERS	RUNNING_Y_ORDERS
2021	3462	3462
2022	5774	9236
2023	5561	14797
2024	9026	23823

Rys. 8.27. Zbiór wynikowy po wykonaniu zapytania z rysunku 8.26

Źródło: opracowanie własne.

Sortowanie wierszy w oknie danych może odbywać się także z użyciem wielu kolumn, co pozwala zwiększyć szczegółowość analizy o dodatkowe wielkości charakteryzujące wymiar. Na rysunku 8.28 przedstawiono kod zapytania, w którym oprócz roku umieszczono także kwartał, co spowodowało, że obliczenia wykonywane za pomocą funkcji *SUM* (linia 2) są teraz realizowane

z użyciem wartości opisujących liczbę zamówień w kolejnych kwartałach roku. Wartość występująca w kolumnie *Running_Q_Orders* danego wiersza jest sumą wartości z kolumny *Q_Orders* tego wiersza i wszystkich wierszy poprzedzających.

```

1 SELECT Year, Quarter, OrdersNum Q_Orders,
2     SUM(OrdersNum) OVER (ORDER BY Year, Quarter) AS Running_Q_Orders
3 FROM (
4     SELECT calendaryear AS Year, calendarquarter AS Quarter,
5           COUNT(DISTINCT orderkey) AS OrdersNum
6     FROM factonlinesales fos INNER JOIN dimdate dd
7         ON fos.orderdatekey = dd.datekey
8     INNER JOIN dimproduct dp ON dp.productkey = fos.productkey
9     WHERE categoryname = 'Bikes' AND calendaryear IN (2023, 2024)
10    GROUP BY calendaryear, calendarquarter
11 )
12 ORDER BY 1, 2 ;

```

Rys. 8.28. Sortowanie wierszy w oknie danych z użyciem wielu kolumn

Źródło: opracowanie własne.

Analizując wynik wykonania zapytania (rys. 8.29), można zauważyć, że wartości w kolumnie *Running_Q_Orders* wyznaczone są w sposób ciągły dla wszystkich kolejnych kwartałów poszczególnych lat i nie występuje efekt kalkulowania tych wartości tylko w obrębie jednego roku. Tego rodzaju obliczenia są możliwe do zdefiniowania w instrukcji *SELECT* i zostaną zaprezentowane w dalszej części rozdziału.

YEAR	QUARTER	Q_ORDERS	RUNNING_Q_ORDERS
2023	1	518	518
2023	2	1145	1663
2023	3	867	2530
2023	4	513	3043
2024	1	693	3736
2024	2	1593	5329
2024	3	1526	6855
2024	4	251	7106

Rys. 8.29. Wynik zapytania zdefiniowanego na rysunku 8.28

Źródło: opracowanie własne.

Utworzenie okna danych za pomocą klauzuli *OVER*, a następnie posortowanie w nim wierszy stwarza możliwość wykonania dodatkowych operacji,

w których wykorzystane zostaną określone wiersze. Przykładem takiego działania jest porównanie wartości w bieżącym wierszu z wartością znajdującą się w pierwszym lub ostatnim wierszu okna danych. Kod źródłowy dla takiego przypadku użycia przedstawiono na rysunku 8.30.

```

1 SELECT Year, Month, OrdersNum M_Orders,
2     OrdersNum - FIRST_VALUE(OrdersNum)
3         OVER(ORDER BY Year, Month) AS Diff_M_Orders
4 FROM (
5     SELECT calendaryear AS Year, monthnumberofyear AS Month,
6         COUNT(DISTINCT orderkey) AS OrdersNum
7     FROM factonlinesales fos INNER JOIN dimdate dd
8         ON fos.orderdatekey = dd.datekey
9     INNER JOIN dimsalesterritory dst
10        ON dst.salesterritorykey = fos.salesterritorykey
11     WHERE calendaryear = 2024 AND dst.salesterritoryname = 'EMEA'
12     GROUP BY calendaryear, monthnumberofyear
13 )
14 ORDER BY 1, 2;
```

Rys. 8.30. Zastosowanie funkcji *FIRST_VALUE* w oknie danych

Źródło: opracowanie własne.

Funkcja *FIRST_VALUE* zwraca wartość z kolumny *OrdersNum* pierwszego wiersza okna danych. Jest to liczba zamówień w pierwszym miesiącu 2024 roku w regionie o nazwie EMEA. Wyrażenie zdefiniowane w liniach 2 i 3, opisane aliasem *Diff_M_Orders*, wyznacza różnicę między liczbą zamówień w bieżącym wierszu okna i liczbą zamówień w styczniu 2024 roku. Warunkiem poprawnego wyznaczenia różnicy jest zastosowanie sortowania rosnącego według miesiąca. Rezultat wykonania zapytania przedstawiono na rysunku 8.31.

YEAR	MONTH	M_ORDERS	DIFF_M_ORDERS
2024	1	216	0
2024	2	293	77
2024	3	478	262
2024	4	489	273
2024	5	638	422
2024	6	821	605
2024	7	865	649
2024	8	1094	878
2024	9	563	347
2024	10	115	-101
2024	11	103	-113
2024	12	126	-90

Rys. 8.31. Wynik zapytania zdefiniowanego na rysunku 8.30

Źródło: opracowanie własne.

Tego rodzaju kalkulacje, w których porównuje się wartość bieżącą i wartość przeszłą w określonym momencie czasu, są bardzo popularną techniką analizy danych, gdyż umożliwiają ocenę aktualnej sytuacji w porównaniu z początkiem okresu.

Jedną z najbardziej użytecznych i najczęściej stosowanych wielkości w analizie szeregów czasowych są różnego rodzaju średnie kroczące (ruchome), które reprezentują wartość średnią z okresu bieżącego i przeszłych okresów. Umożliwiają one wyodrębnienia wahań sezonowych, krótkookresowych, koniunkturalnych i przypadkowych czy też śledzenie trendów. Jednym z rodzajów takiej średniej jest wielkość obliczana na podstawie danych z okresu bieżącego i wszystkich okresów poprzedzających. W celu jej obliczenia za pomocą polecenia *SELECT* konieczne jest użycie w definicji okna danych dodatkowej klauzuli *ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW*. Kod przedstawiony na rysunku 8.32 ilustruje sposób implementacji takiego rozwiązania. Obliczenia wykonywane w oknie danych, tworzonym w zapytaniu nadrzędnym, będą wykonywane z użyciem tabeli wygenerowanej przez podzapytanie. W tabeli tej znajdują się dane opisujące wartość sprzedaży w kolejnych miesiącach 2024 roku. Następnie w liniach od 2 do 4 zapytania głównego zastosowano funkcję *AVG*, której zakres działania określono za pomocą dodatkowej klauzuli, definiującej zbiór składający się z wiersza bieżącego i wszystkich wierszy poprzedzających w oknie.

```
1 SELECT Month, ROUND(Sales,0)/1000 AS M_Sales,  
2     ROUND(AVG(Sales)  
3         OVER (ORDER BY Month ROWS BETWEEN UNBOUNDED PRECEDING  
4             AND CURRENT ROW),0)/1000 AS Moving_Avg_M_Sales  
5 FROM (  
6     SELECT monthnumberofyear AS Month,  
7         ROUND(SUM(transactionprice*quantity),0) AS Sales  
8     FROM factonlinesales fos INNER JOIN dimdate dd  
9         ON fos.orderdatekey= dd.datekey  
10    WHERE calendaryear = 2024  
11    GROUP BY monthnumberofyear  
12 )  
13 ORDER BY 1, 2 ;
```

Rys. 8.32. Zapytanie zawierające definicję średniej kroczącej

Źródło: opracowanie własne.

Rezultat wykonania tego zapytania przedstawiono na rysunku 8.33.

MONTH	M_SALES	MOVING_AVG_M_SALES
1	502.758	502.758
2	610.584	556.671
3	971.004	694.782
4	1082.719	791.766
5	1512.922	935.997
6	2072.209	1125.366
7	1009.845	1108.863
8	2297.984	1257.503
9	1159.79	1246.646
10	184.045	1140.386
11	231.948	1057.801
12	236.053	989.322

Rys. 8.33. Wynik zapytania wyznaczającego średnią kroczącą

Źródło: opracowanie własne.

Kolumna *M_Sales* zawiera dane o wartości sprzedaży w poszczególnych miesiącach, natomiast w kolumnie *Moving_Avg_M_Sales* znajduje się wartość średniej kroczącej. W celu zweryfikowania poprawności danych w ostatniej kolumnie można przeprowadzić proste obliczenia, np. dla trzeciego miesiąca roku. Suma wartości w kolumnie *M_Sales* od stycznia do marca wynosi 2084,346. Średnia wyliczona z tej wartości za trzy miesiące to 694,782. Jest to ta sama wartość, która występuje w kolumnie *Moving_Avg_M_Sales* dla trzeciego miesiąca.

Szczególne znaczenie w analizie szeregów czasowych ma również n-okresowa średnia krocząca, gdzie okres może być rozumiany jako dzień, tydzień czy miesiąc. Wielkość ta umożliwia np. określenie dla danego miesiąca wartości średniej z trzech ostatnich miesięcy. Przykład polecenia *SELECT*, w którym wyznaczono taką średnią, zilustrowano na rysunku 8.34. Natomiast wynik wykonania tego zapytania przedstawiono na rysunku 8.35.

```

1 SELECT Month, ROUND(Sales,0)/1000 AS M_Sales,
2     ROUND(AVG(Sales)
3         OVER (ORDER BY Month ROWS 2 PRECEDING),0)/1000
4         AS Moving_3_M_Avg_Sales
5 FROM (
6     SELECT monthnumberofyear Month,
7         ROUND(SUM(transactionprice*quantity),0) AS Sales
8     FROM factonlinesales fos INNER JOIN dimdate dd
9         ON fos.orderdatekey= dd.datekey
10    WHERE calendaryear = 2024
11    GROUP BY monthnumberofyear
12 )
13 ORDER BY 1, 2 ;

```

Rys. 8.34. Definicja zapytania wyznaczającego 3-miesięczną średnią kroczącą

Źródło: opracowanie własne.

MONTH	M_SALES	MOVING_3_M_AVG_SALES
1	502.758	502.758
2	610.584	556.671
3	971.004	694.782
4	1082.719	888.102
5	1512.922	1188.882
6	2072.209	1555.95
7	1009.845	1531.659
8	2297.984	1793.346
9	1159.79	1489.206
10	184.045	1213.94
11	231.948	525.261
12	236.053	217.349

Rys. 8.35. Wynik zapytania wyznaczającego 3-miesięczną średnią kroczącą

Źródło: opracowanie własne.

Określenie liczby przeszłych okresów odbywa się za pomocą klauzuli *ROWS PRECEDING*, umieszczonej w definicji okna (linia 3 kodu). Podobnie jak w poprzednim przykładzie, kolumna *M_Sales* zawiera dane o sprzedaży w danym miesiącu 2024 roku. Natomiast w kolumnie *Moving_3_M_Avg_Sales* znajdują się wartości odpowiadające średniej z trzech ostatnich miesięcy, która uwzględnia dany miesiąc i dwa okresy wstecz.

W celu lepszego zrozumienia idei kalkulowania 3-miesięcznej średniej kroczącej zweryfikowano wartości w kolumnie *Moving_3_M_Avg_Sales* dla czwartego miesiąca. W tym celu, wykorzystując kolumnę *M_Sales*, wyznaczono średnią z wartości sprzedaży od miesiąca drugiego do czwartego, otrzymując wartość 888,102. Taka sama wartość znajduje się w wierszu tego miesiąca w kolumnie *Moving_3_M_Avg_Sales*, co oznacza poprawność kalkulacji wykonanej w instrukcji *SELECT*.

Ciekawym rozszerzeniem dotychczas zaprezentowanych treści jest prezentacja możliwości jednoczesnego użycia klauzul *PARTITION BY* i *ORDER BY* podczas grupowania danych przy użyciu klauzuli *OVER*. Tego rodzaju działanie umożliwia wykonywanie obliczeń analitycznych wyłącznie na grupach wierszy zbudowanych za pomocą klauzuli *PARTITION BY*, które zostaną posortowane w każdej partycji przy użyciu klauzuli *ORDER BY*. Jej uprzednie wykorzystanie w klauzuli *OVER* prowadziło do posortowania wierszy wyłącznie w poszczególnych oknach danych i kalkulacji ciągłych bez uwzględnienia faktu, iż w rzeczywistości wiersze te mogą należeć do różnych grup. Taki przypadek zilustrowano na rysunkach 8.28 i 8.29, gdzie sumę narastająco wyznaczono dla wszystkich kwartałów w kolejnych latach zamiast dla kwartałów jednego roku.

Zastosowanie klauzul *PARTITION BY* i *ORDER BY* w definicji okna danych, umieszczonej w tej samej instrukcji *SELECT*, pozwala wygenerować bardzo ciekawe dane analityczne obrazujące wartości skumulowane w kolejnych okresach czasu, np. wartość narastającą sprzedaży w kolejnych miesiącach danego roku, czy też dane opisujące zmianę wartości sprzedaży w porównaniu z początkiem okresu (kwartału lub roku). Ogólną składnię polecenia *SELECT*, zawierającego definicję okna oraz partycji z posortowanymi wierszami, przedstawiono na rysunku 8.36.

```

1 SELECT column_2, column_1,
2     window_function(col_parameter)
3     OVER (PARTITION BY column_2 ORDER BY column_1)
4 FROM data_source
5 WHERE ...

```

Rys. 8.36. Sortowanie danych w partycji okna utworzonego klauzulą *OVER*

Źródło: opracowanie własne.

Na rysunku 8.37 zaprezentowano przykład będący modyfikacją kodu z rysunku 8.28, który umożliwia wyznaczenie narastającej liczby zamówień w kolejnych kwartałach tylko w zakresie jednego roku.

```

1 SELECT Year, Quarter, OrdersNum Q_Orders,
2     SUM(OrdersNum) OVER (PARTITION BY Year
3     ORDER BY Quarter) AS Running_Q_Orders
4 FROM (
5     SELECT calendaryear AS Year, calendarquarter AS Quarter,
6     COUNT(DISTINCT orderkey) AS OrdersNum
7     FROM factonlinesales fos INNER JOIN dimdate dd
8     ON fos.orderdatekey = dd.datekey
9     INNER JOIN dimproduct dp ON dp.productkey = fos.productkey
10    WHERE categoryname = 'Bikes' AND calendaryear IN (2023, 2024)
11    GROUP BY calendaryear, calendarquarter
12 )
13 ORDER BY 1, 2 ;

```

Rys. 8.37. Przykład zapytania z sortowaniem danych w partycji okna danych

Źródło: opracowanie własne.

W zapytaniu tym utworzono partycje, zdefiniowane przez numer roku. W obrębie każdej partycji wiersze sortowane są według kwartału. Natomiast

funkcja *SUM* wykonuje obliczenia nie w całym oknie danych, lecz tylko w poszczególnych partycjach. W rezultacie suma narastająco jest wyznaczana tylko dla każdego roku, co w przykładzie skutkuje tym, iż dla pierwszego kwartału 2024 roku liczba zamówień w tym kwartale (*Q_Orders*) jest równa wartości narastającej w tym roku (*Running_Q_Orders*). Wynik wykonania zapytania, wyznaczającego sumę narastającą w partycji, przedstawiono na rysunku 8.38.

YEAR	QUARTER	Q_ORDERS	RUNNING_Q_ORDERS
2023	1	518	518
2023	2	1145	1663
2023	3	867	2530
2023	4	513	3043
2024	1	693	693
2024	2	1593	2286
2024	3	1526	3812
2024	4	251	4063

Rys. 8.38. Wynik zapytania z rysunku 8.37

Źródło: opracowanie własne.

Podsumowując rozważania dotyczące tworzenia okien i partycji danych oraz zastosowania w nich różnych kalkulacji, warto jeszcze zaprezentować przykład zaawansowanej postaci instrukcji *SELECT*, który zobrazuje możliwość wykonania w tym samym poleceniu obliczeń z użyciem wierszy pochodzących z różnych zakresów danych.

```

1 SELECT Year, Month, OrderNum AS M_Orders,
2    SUM(OrderNum) OVER (PARTITION BY Year ORDER BY Month)
3    AS Run_M_Orders,
4    100*ROUND(OrderNum /
5    SUM(OrderNum) OVER (PARTITION BY Year),4) AS Pctg_Y_Orders,
6    100*ROUND(SUM(OrderNum) OVER (PARTITION BY Year ORDER BY Month) /
7    SUM(OrderNum) OVER (PARTITION BY Year),4) AS Pctg_Run_M_Orders
8 FROM ( SELECT calendaryear Year, monthnumberofyear Month,
9    COUNT(DISTINCT orderkey) AS OrderNum
10    FROM factonlinesales fos INNER JOIN dimdate dd
11    ON fos.orderdatekey= dd.datekey
12    GROUP BY calendaryear, calendarquarter, monthnumberofyear
13 )
14 ORDER BY 1, 2, 3 ;

```

Rys. 8.39. Złożone zapytanie operujące na wierszach z różnych zakresów danych

Źródło: opracowanie własne.

Na rysunku 8.39 zdefiniowano zapytanie, które wyznacza dla każdego miesiąca liczbę zamówień (*M_Orders*), narastającą liczbę zamówień (*Run_M_Orders*)

w roku, procentowy udział miesięcznej liczby zamówień w rocznej liczbie zamówień (*Pctg_Y_Orders*) oraz procentowy udział narastającej miesięcznej liczby zamówień w rocznej liczbie zamówień (*Pctg_Run_M_Orders*).

W poleceniu tym w linii 2 wyznaczono narastającą liczbę zamówień w kolejnych miesiącach każdego roku. Wykonanie tej kalkulacji wymagało stworzenia partycji według roku i posortowania w niej wierszy z użyciem kolumny opisującej miesiąc. W linii 5 kodu źródłowego obliczono liczbę zamówień w danym roku, a następnie wykorzystano tę wartość w mianowniku wyrażenia opisanego aliasem *Pctg_Y_Orders*. Wyznaczenie wartości wyrażenia w kolumnie oznaczonej jako *Pctg_Run_M_Orders* odbywa się w liniach 6 i 7. Ponieważ wszystkie kalkulacje w poleceniu należało wykonać dla danego roku, konieczne było zawężenie dla nich zbioru danych do partycji określonej przez rok (kolumna *Year*). Wynik zapytania zaprezentowano na rysunku 8.40. Prezentację danych ograniczono tylko do roku 2021. W celu potwierdzenia poprawności otrzymanego wyniku można dla danego miesiąca wykonać sumowanie wartości w kolumnie *Pctg_Y_Orders* od początku roku, a wynik tej operacji będzie równy wartości w kolumnie *Pctg_Run_M_Orders* rekordu tego miesiąca.

YEAR	MONTH	M_ORDERS	RUN_M_ORDERS	PCTG_Y_ORDERS	PCTG_RUN_M_ORDERS
2021	1	72	72	2.08	2.08
2021	2	90	162	2.6	4.68
2021	3	74	236	2.14	6.82
2021	4	369	605	10.66	17.48
2021	5	587	1192	16.96	34.43
2021	6	107	1299	3.09	37.52
2021	7	165	1464	4.77	42.29
2021	8	501	1965	14.47	56.76
2021	9	496	2461	14.33	71.09
2021	10	416	2877	12.02	83.1
2021	11	270	3147	7.8	90.9
2021	12	315	3462	9.1	100

Rys. 8.40. Wynik wykonania zapytania z rysunku 8.39

Źródło: opracowanie własne.

Na podstawie przykładów zaprezentowanych w tym podrozdziale można zauważyć, jak bardzo użyteczne w analizie danych są możliwości tworzenia okien i partycji danych, dzięki którym zwiększa się szczegółowość otrzymywanych wyników oraz tworzone są warunki do wykonywania coraz bardziej zaawansowanych kalkulacji, co zostanie uwidocznione w kolejnym podrozdziale dotyczącym tworzenia rankingów.

8.3. Generowanie rankingów – RANK i DENSE_RANK

Bardzo popularną operacją wykonywaną w analizie danych jest tworzenie różnego rodzaju rankingów, których celem jest wskazanie najlepszych lub najgorszych wartości z punktu widzenia określonego wymiaru. Przykładowe obszary zastosowania tej operacji to identyfikacja najlepiej sprzedających się produktów, wskazanie kraju generującego najwyższe przychody czy kanału zakupowego, z którego korzysta największa liczba klientów.

W operacji rankingowania można wykorzystywać zarówno okna, jak i partycje danych, utworzone klauzulami *OVER* i *PARTITION BY*, aby doprecyzować zakres danych, w obrębie którego taki ranking jest tworzony. Wyznaczenie pozycji w rankingu odbywa się za pomocą funkcji okna *RANK*, której przykład użycia zaprezentowany na rysunku 8.41.

```
1 SELECT calendaryear Year, countryname, CustNum AS YC_Cust,  
2     RANK() OVER (ORDER BY CustNum DESC) AS Rank_Position  
3 FROM (  
4     SELECT calendaryear, countryname,  
5           COUNT(DISTINCT customerkey) AS CustNum  
6     FROM factonlinesales fos INNER JOIN dimdate dd  
7          ON fos.orderdatekey= dd.datekey  
8     INNER JOIN dimsalesterritory dst  
9          ON dst.salesterritorykey = fos.salesterritorykey  
10    WHERE calendaryear IN (2023, 2024) AND monthnumberofyear = 5  
11    GROUP BY calendaryear, countryname  
12 )  
13 ORDER BY 4 ;
```

Rys. 8.41. Tworzenie rankingu w oknie danych

Źródło: opracowanie własne.

Jest to instrukcja tworząca ranking krajów według liczby klientów, którzy dokonali zakupu z danego państwa w piątym miesiącu 2023 lub 2024 roku. Kryterium, według którego tworzony jest ranking i sposób przypisania w nim pozycji, określono odpowiednio przez kolumnę występującą po klauzuli *ORDER BY* i słowo kluczowe *DESC*. Składnia przykładowego zapytania spowoduje, że pozycja numer 1 w rankingu zostanie przypisana do kraju z największą liczbą klientów.

Wynik wykonania powyższego kodu uwidoczniiono na rysunku 8.42.

YEAR	COUNTRYNAME	YC_CUST	RANK_POSITION
2024	France	406	1
2024	Canada	225	2
2023	Canada	170	3
2024	United States	160	4
2023	Nederland	148	5
2023	United States	134	6
2023	France	127	7
2023	Poland	106	8
2023	Italy	95	9
2023	Spain	83	10
2024	Poland	56	11

Rys. 8.42. Rezultat zapytania z rysunku 8.41

Źródło: opracowanie własne.

Liczba klientów została umieszczona w kolumnie *YC_Cust*, natomiast pozycja kraju w rankingu – w kolumnie *Rank_Position*. W tym przypadku okno danych jest utworzone przez rekordy pochodzące ze wszystkich lat wskazanych w podzapytaniu i dlatego wartości w kolumnie *CountryName* nie są uporządkowane w obrębie jednego roku. W celu utworzenia rankingu w obrębie partycji okna danych konieczne jest wskazanie kolumny, której unikalne wartości umożliwią zdefiniowanie dodatkowych podzbiorów danych.

W zapytaniu umieszczonym na rysunku 8.43 utworzono partycje według roku, a następnie w każdej z nich wyznaczono ranking kanałów sprzedażowych według liczby złożonych w nich zamówień.

```

1  SELECT calendaryear Year, channelname, ordernum AS YC_Orders,
2         RANK() OVER (PARTITION BY calendaryear
3                        ORDER BY OrderNum DESC) AS Rank_Position
4  FROM (
5      SELECT calendaryear, channelname,
6             COUNT(DISTINCT orderkey) AS OrderNum
7      FROM factonlinesales fos INNER JOIN dimdate dd
8         ON fos.orderdatekey= dd.datekey
9      INNER JOIN dimorderchannel dc
10         ON fos.channelkey = dc.channelkey
11      GROUP BY calendaryear, channelname
12  )
13 ORDER BY 1 DESC, 4 ;

```

Rys. 8.43. Tworzenie rankingu w partycjach okna danych

Źródło: opracowanie własne.

W wyniku wykonania zapytania został wygenerowany zbiór wyników przedstawiony na rysunku 8.44. Jak można zauważyć, ranking kanałów odbywa się w każdym roku niezależnie, a efekt ten uzyskano dzięki obecności klauzuli *PARTITION BY calendaryear* w definicji okna danych (linia 2 kodu źródłowego).

YEAR	CHANNELNAME	YC_ORDERS	RANK_POSITION
2024	Mobile channel	4521	1
2024	Web channel	4505	2
2023	Mobile channel	2783	1
2023	Web channel	2778	2
2022	Mobile channel	2910	1
2022	Web channel	2864	2
2021	Web channel	1748	1
2021	Mobile channel	1714	2

Rys. 8.44. Rezultat rankingowania w partycjach okna danych

Źródło: opracowanie własne.

Tworzenie listy rankingowej może w niektórych przypadkach prowadzić do sytuacji, w której na tej samej pozycji rankingu znajdzie się wiele różnych danych, np. produktów, krajów, klientów czy kanałów sprzedaży. Jest to spowodowane tym, że wartość przypisana do danej pozycji w rankingu jest charakterystyczna dla wielu elementów, co w konsekwencji sprawia, że kolejny numer pozycji rankingowej nie będzie większy o 1 od numeru poprzedniej pozycji. Przypadek taki zobrazowano na rysunku 8.45. Jest to ranking miesięcy w 2022 roku według liczby klientów, którzy złożyli zamówienia w danym okresie czasu. Na jego pierwszej pozycji znalazł się miesiąc z najmniejszą taką liczbą. Efekt ten uzyskano, stosując domyślny rosnący sposób sortowania danych w oknie (linia 2 kodu źródłowego).

```

1 SELECT Year, Month, CustNum AS M_Cust,
2     RANK() OVER (PARTITION BY Year ORDER BY CustNum) Rank_Position
3 FROM ( SELECT calendaryear Year, monthnumberofyear Month,
4         COUNT(DISTINCT customerkey) AS CustNum
5     FROM factonlinesales fos INNER JOIN dimdate dd
6         ON fos.orderdatekey= dd.datekey
7     WHERE calendaryear = 2022
8     GROUP BY calendaryear, calendarquarter, monthnumberofyear
9 )
10 ORDER BY 4, 2 ;

```

Rys. 8.45. Kod tworzenia rankingu miesięcy w roku według liczby klientów

Źródło: opracowanie własne.

Jak można zauważyć na rysunku 8.46, obrazującym wynik powyższego zapytania, w rankingu tym występują dwa miesiące, tj. styczeń i luty, z taką samą liczbą klientów równą 357. Miesiącům tym przypisano identyczną pozycję rankingową równą 3, w rezultacie czego następna pozycja, tj. miesiąc grudzień, nie ma wartości 4, tylko 5.

YEAR	MONTH	M_CUST	RANK_POSITION
2022	11	334	1
2022	10	346	2
2022	1	357	3
2022	2	357	3
2022	12	404	5
2022	3	465	6
2022	4	501	7
2022	7	527	8
2022	8	547	9
2022	5	573	10
2022	9	593	11
2022	6	693	12

Rys. 8.46. Ranking miesięcy w roku według liczby klientów (funkcja *RANK*)

Źródło: opracowanie własne.

W takim przypadku możliwe jest zastosowanie tzw. gęstego rankingu (ang. *dense ranking*), który umożliwia zachowanie ciągłości numerowania kolejnych jego pozycji. Operację tę realizuje się za pomocą funkcji *DENSE_RANK*, którą należy umieścić w zapytaniu w miejscu funkcji *RANK*. Wówczas w zbiorze wynikowym zmodyfikowanego zapytania w kolumnie *Rank_Position* wystąpią wartości ciągłe, co widać na rysunku 8.47.

YEAR	MONTH	M_CUST	RANK_POSITION
2022	11	334	1
2022	10	346	2
2022	1	357	3
2022	2	357	3
2022	12	404	4
2022	3	465	5
2022	4	501	6
2022	7	527	7
2022	8	547	8
2022	5	573	9
2022	9	593	10
2022	6	693	11

Rys. 8.47. Ranking miesięcy w roku według liczby klientów (funkcja *DENSE_RANK*)

Źródło: opracowanie własne.

Użycie funkcji *DENSE_RANK* spowodowało, że grudzień nie jest już na piątej pozycji rankingu, lecz na czwartej. W ten sposób taka lista jest czytelniejsza dla użytkownika końcowego i łatwiejsza w interpretacji.

W uzupełnieniu dotychczas prezentowanych treści dotyczących funkcji rankingowania warto jeszcze wskazać możliwość wykorzystania ich rezultatu do ograniczenia liczby pozycji rankingu występujących w zbiorze wynikowym. Tego rodzaju potrzeba bardzo często pojawia się w praktyce, a jej wyrazem są np. pytania o trzech najbardziej wartościowych klientów, najbardziej dochodowy kraj czy najlepiej sprzedający się produkt.

Przykład zapytania wykonującego filtrowanie pozycji w rankingu przedstawiono na rysunku 8.48.

```
1 SELECT calendaryear AS Year, channelname AS Channel, countryname,
2     ROUND(Sales/1000,3) Total_Sales, Rank_Position
3 FROM ( SELECT calendaryear, channelname, countryname, Sales,
4     DENSE_RANK() OVER (PARTITION BY calendaryear, channelname
5     ORDER BY Sales DESC) AS Rank_Position
6     FROM (SELECT calendaryear, channelname, countryname,
7         SUM(transactionprice*quantity) AS Sales
8         FROM factonlinesales fos INNER JOIN dimdate dd
9         ON fos.orderdatekey= dd.datekey
10        INNER JOIN dimorderchannel doc
11        ON doc.channelkey = fos.channelkey
12        INNER JOIN dimsalesterritory dst
13        ON dst.salesterritorykey = fos.salesterritorykey
14        WHERE monthnumberofyear = 1
15        AND calendaryear IN (2023, 2024)
16        GROUP BY calendaryear, channelname, countryname ) )
17 WHERE Rank_Position IN (1, 2, 3)
18 ORDER BY 1, 2 ;
```

Rys. 8.48. Zapytanie wykonujące filtrowanie pozycji w rankingu

Źródło: opracowanie własne.

Polecenie to ma określić trzy najlepsze kraje z punktu widzenia wartości sprzedaży dla poszczególnych kanałów sprzedażowych w styczniu 2023 i 2024 roku. Źródłem danych do wyznaczenia rankingu jest podzapytanie (od linii 6 do 16 kodu), w którym obliczana jest wartość sprzedaży dla poszczególnych lat, kanałów oraz krajów w podanym okresie czasu. Następnie w linii 17 kodu

zdefiniowano warunek filtrowania w celu ograniczenia liczby pozycji w rankingu do trzech. Rezultat wykonania tego zapytania zobrazowano na rysunku 8.49.

YEAR	CHANNEL	COUNTRYNAME	TOTAL_SALES	RANK_POSITION
2023	Mobile channel	Nederland	37.692	1
2023	Mobile channel	Spain	28.421	2
2023	Mobile channel	Italy	24.832	3
2023	Web channel	Nederland	53.762	1
2023	Web channel	Spain	33.36	2
2023	Web channel	Italy	32.084	3
2024	Mobile channel	Canada	60.554	1
2024	Mobile channel	Nederland	37.188	2
2024	Mobile channel	Poland	33.639	3
2024	Web channel	France	40.838	1
2024	Web channel	United States	37.243	2
2024	Web channel	Poland	35.718	3

Rys. 8.49. Rezultat filtrowania pozycji w rankingu

Źródło: opracowanie własne.

Zaprezentowane treści dotyczące generowania różnego rodzaju rankingów w obrębie okien i partycji danych uwidaczniają rosnącą złożoność polecenia *SELECT* w miarę rozszerzania się informacyjnych potrzeb użytkownika końcowego. Jednak bardzo duże znaczenie takiej konstrukcji, jak ranking w procesach decyzyjnych, wymusza konieczność tworzenia tego rodzaju zapytań i używania ich rezultatów w ocenie analizowanych zagadnień.

8.4. Zaawansowane funkcje okna – LAG i LEAD

Analiza szeregów czasowych często wymaga porównywania wartości charakterystycznych dla bieżącego lub innego danego okresu – np. roku, kwartału czy miesiąca – oraz wartości dla jednego lub wielu poprzednich lub przyszłych takich samych okresów czasu. Taka operacja pozwala ocenić dynamikę zmian wartości w czasie, zidentyfikować trend tych zmian oraz dostrzec wahania cykliczne czy sezonowe. Dla przykładu: menedżer odpowiedzialny za sprzedaż jest zainteresowany nie tylko bezwzględną wartością sprzedaży w danym miesiącu, ale także jej porównaniem z wartością z miesiąca poprzedniego, wartością z pierwszego miesiąca danego roku bądź też wartością dla danego miesiąca roku poprzedniego. Powyższe rodzaje działań są typowe dla działów sprzedaży czy obsługi klienta i powszechnie stosowane w wielu organizacjach biznesowych. Dlatego też różne technologie programowania oraz narzędzia

analizy danych oferują wbudowane funkcjonalności, które wspierają tego rodzaju operacje. Przykładem takiego rozwiązania są funkcje wbudowane *LAG* i *LEAD*, dostępne na wszystkich wiodących platformach bazodanowych (Oracle, MySQL, PostgreSQL, MS SQL Server), a także w językach skryptowych Python (biblioteka *pandas*) czy R.

Pierwszą z omawianych funkcji okna jest *LAG*, która zapewnia dostęp do wierszy poprzedzających wiersz bieżący w danym oknie lub partycji danych. Podstawową składnię polecenia *SELECT* zawierającego tę funkcję przedstawiono na rysunku 8.50.

```
1 SELECT column_1, column_2,  
2    LAG(parameter offset_num ) OVER (PARTITION BY column_1  
3    ORDER BY column_2)  
4 FROM data_source  
5 WHERE ... ;
```

Rys. 8.50. Składnia instrukcji *SELECT* z funkcją *LAG*

Źródło: opracowanie własne.

Parametrem funkcji *LAG* może być kolumna lub wyrażenie kalkulowane w zapytaniu, natomiast *offset_num* określa, który wiersz poprzedzający wiersz bieżący będzie uwzględniony w obliczeniach. Domyślna wartość *offset_num* wynosi 1, tj. wiersz występujący bezpośrednio przed wierszem bieżącym. Przykład użycia funkcji *LAG* w oknie danych zaprezentowano na rysunku 8.51.

```
1 SELECT monthnumberofyear AS Month,  
2    COUNT(DISTINCT customerkey) AS Cust_Num,  
3    COUNT(DISTINCT customerkey) -  
4    LAG(COUNT(DISTINCT customerkey),1)  
5    OVER (ORDER BY monthnumberofyear) AS Diff_MtM  
6 FROM factonlinesales fos INNER JOIN dimdate dd  
7    ON fos.orderdatekey= dd.datekey  
8 WHERE calendaryear = 2024  
9 GROUP BY monthnumberofyear  
10 ORDER BY 1 ;
```

Rys. 8.51. Przykład użycia funkcji *LAG* w oknie danych

Źródło: opracowanie własne.

W zapytaniu tym wyznaczana jest liczba klientów (*Cust_Num*), którzy złożyli zamówienia w kolejnych miesiącach 2024 roku, oraz różnica między liczbą

klientów w danym miesiącu i miesiącu poprzednim (*Diff_MtM*). Widok zbioru wynikowego zapytania przedstawiono na rysunku 8.52. Dla pierwszego miesiąca wartość w kolumnie *Diff_MtM* jest równa *NULL*, gdyż w tym przypadku w funkcji *LAG* następuje odwołanie wykraczające poza zakres okna danych.

MONTH	CUST_NUM	DIFF_MTM
1	345	(null)
2	468	123
3	711	243
4	751	40
5	1011	260
6	1246	235
7	1298	52
8	1650	352
9	824	-826
10	167	-657
11	163	-4
12	193	30

Rys. 8.52. Rezultat wykonania zapytania z rysunku 8.51

Źródło: opracowanie własne.

Ciekawym przykładem kalkulacji z użyciem funkcji *LAG* jest kod zamieszczony na rysunku 8.53. Obliczenia wykonywane w tym poleceniu należą do jednych z najbardziej popularnych w analizie szeregów czasowych i umożliwiają porównanie wartości z danego miesiąca bieżącego roku (*Orders_Num*) z wartością w tym samym miesiącu roku poprzedniego (*Orders_Num_Y_Prev*). W kodzie tym wyznaczana jest liczba zamówień złożonych w dwóch pierwszych kwartałach w latach 2023 i 2024.

```

1 SELECT calendaryear AS Year, monthnumberofyear AS Month,
2        SUM(NumOrders) AS Orders_Num,
3        LAG(SUM(NumOrders),1,0) OVER (PARTITION BY monthnumberofyear
4                                     ORDER BY monthnumberofyear) AS Orders_Num_Y_Prev
5 FROM (SELECT calendaryear, monthnumberofyear,
6             COUNT(DISTINCT orderkey) AS NumOrders
7        FROM factonlinesales fos INNER JOIN dimdate dd
8             ON fos.orderdatekey= dd.datekey
9        WHERE calendaryear IN (2023 ,2024) AND calendarquarter IN (1, 2)
10       GROUP BY calendaryear, monthnumberofyear
11      )
12 GROUP BY calendaryear, monthnumberofyear
13 ORDER BY 1, 2;
```

Rys. 8.53. Przykład użycia funkcji *LAG* w partycji danych

Źródło: opracowanie własne.

Szczegółowo identyfikując wartości w zbiorze wynikowym (rys. 8.54), można zauważyć, że wartość w kolumnie *Orders_Num_Y_Prev* w wierszu 7 jest taka sama jak w kolumnie *Orders_Num* w wierszu 1. Wiersze te odpowiadają temu samemu miesiącowi, ale w różnych latach. Warto także zauważyć wpływ trzeciego parametru funkcji *LAG*, tutaj równemu 0, który wykorzystano w sytuacji braku wartości dla poprzedniego okresu (w przykładzie braku wartości dla tego samego miesiąca poprzedniego roku).

	YEAR	MONTH	ORDERS_NUM	ORDERS_NUM_Y_PREV
1	2023	1	158	0
2	2023	2	172	0
3	2023	3	188	0
4	2023	4	316	0
5	2023	5	936	0
6	2023	6	571	0
7	2024	1	347	158
8	2024	2	471	172
9	2024	3	720	188
10	2024	4	764	316
11	2024	5	1026	936
12	2024	6	1275	571

Rys. 8.54. Rezultat wykonania zapytania z rysunku 8.53

Źródło: opracowanie własne.

W kolejnym przykładzie omówiono scenariusz umożliwiający porównanie wartości reprezentujących liczbę zamówień z danego kwartału z wartościami w jednym i dwóch tych samych okresach, czyli kwartałach, poprzedzających tego samego roku. Definiując w linii 5 kodu źródłowego (rys. 8.55) dodatkowy parametr funkcji *LAG*, określający offset równy 2, uzyskano dostęp do wiersza z podzapytania zawierającego dane dla wymaganego kwartału.

```

1 SELECT calendaryear AS Year, calendarquarter AS Quarter,
2      SUM(NumOrders) AS Orders_Num,
3      LAG(SUM(NumOrders),1,0) OVER (PARTITION BY calendaryear
4      ORDER BY calendarquarter) AS Orders_Num_Q_Prev,
5      LAG(SUM(NumOrders),2,0) OVER (PARTITION BY calendaryear
6      ORDER BY calendarquarter) AS Orders_Num_Q_2Prev
7 FROM (SELECT calendaryear, calendarquarter,
8      COUNT(DISTINCT orderkey) AS NumOrders
9      FROM factonlinesales fos INNER JOIN dimdate dd
10     ON fos.orderdatekey= dd.datekey
11     WHERE calendaryear IN (2023, 2024)
12     GROUP BY calendaryear, calendarquarter
13 )
14 GROUP BY calendaryear, calendarquarter
15 ORDER BY 1, 2;

```

Rys. 8.55. Przykład użycia funkcji *LAG* w partycji danych z opcją offsetu

Źródło: opracowanie własne.

Zbiór wynikowy powyższego zapytania przedstawiono na rysunku 8.56.

	YEAR	QUARTER	ORDERS_NUM	ORDERS_NUM_Q_PREV	ORDERS_NUM_Q_2PREV
1	2023	1	518	0	0
2	2023	2	1823	518	0
3	2023	3	1997	1823	518
4	2023	4	1223	1997	1823
5	2024	1	1538	0	0
6	2024	2	3065	1538	0
7	2024	3	3899	3065	1538
8	2024	4	524	3899	3065

Rys. 8.56. Rezultat wykonania zapytania z rysunku 8.55

Źródło: opracowanie własne.

W celu weryfikacji poprawności danych w tym zbiorze można przeanalizować zawartość wiersza numer 3, tj. trzeci kwartał 2023 roku. W kolumnie *Orders_Num_Q_Prev* występuje wartość 1823 i jest to liczba zamówień w kwartale poprzedzającym, czyli drugim tego samego roku (*Orders_Num*). Jednocześnie wartość w kolumnie *Orders_Num_Q_2Prev* tego wiersza wynosi 518 i jest to ta sama liczba, która występuje w pierwszym kwartale 2023 roku w kolumnie *Orders_Num*.

Zaprezentowana w tym rozdziale funkcja *LAG* umożliwia analizę szeregów czasowych z uwzględnieniem okresów przeszłych. Komplementarną do niej funkcją jest *LEAD*, która pozwala wziąć pod uwagę okresy przyszłe względem

bieżącego, gdyż zapewnia ona dostęp do następnych (kolejnych) rekordów okna lub partycji danych względem aktualnego wiersza. Składnia polecenia *SELECT*, w którym występuje funkcja *LEAD*, jest identyczna jak ta uwidoczniiona na rysunku 8.50, lecz w miejscu funkcji *LAG* należy użyć *LEAD*. W przypadku tej funkcji wielkość offsetu wskazywać będzie na n-ty rekord następujący po wierszu bieżącym. Przykład użycia funkcji *LEAD* do porównania wartości w danym miesiącu z wartością z miesiąca następnego obrazuje kod na rysunku 8.57.

```

1 SELECT monthnumberofyear AS Month,
2      COUNT(DISTINCT customerkey) AS Cust_Num,
3      COUNT(DISTINCT customerkey) -
4      LEAD(COUNT(DISTINCT customerkey) )
5      OVER (ORDER BY monthnumberofyear) AS Diff_MtM
6 FROM factonlinesales fos INNER JOIN dimdate dd
7      ON fos.orderdatekey= dd.datekey
8 WHERE calendaryear = 2024
9 GROUP BY monthnumberofyear
10 ORDER BY 1 ;

```

Rys. 8.57. Przykład użycia funkcji *LEAD* w oknie danych

Źródło: opracowanie własne.

W drugiej linii tego kodu wyznaczono liczbę klientów (*Cust_Num*), którzy złożyli zamówienie w danym miesiącu 2024 roku. Natomiast w liniach 4 i 5 obliczono liczbę klientów w miesiącu następującym po bieżącym i odjęto ją od wartości uprzednio wyznaczonej w linii 3 (*Diff_MTM*). Poprawność powyższego kodu można sprawdzić, analizując jego wynik przedstawiony na rysunku 8.58.

	MONTH	CUST_NUM	DIFF_MTM
1	1	345	-123
2	2	468	-243
3	3	711	-40
4	4	751	-260
5	5	1011	-235
6	6	1246	-52
7	7	1298	-352
8	8	1650	826
9	9	824	657
10	10	167	4
11	11	163	-30
12	12	193	(null)

Rys. 8.58. Rezultat wykonania zapytania z rysunku 8.57

Źródło: opracowanie własne.

Na przykład w ósmym wierszu zbioru wynikowego liczba klientów jest równa 1650, natomiast różnica między liczbą klientów w tym miesiącu i następnym wynosi 826. Rezultat identycznych działań na każdym z wierszy wynikowych potwierdza prawidłowość sformułowania zapytania.

W przedstawionym uprzednio kodzie źródłowym funkcja *LEAD* operowała z domyślną wartością offsetu, umożliwiając w ten sposób wykonanie porównania miesiąc do miesiąca. Istnieje jednak możliwość, aby w takim porównaniu uwzględnić dowolny następny miesiąc poprzez jawne podanie wartości offsetu. Przypadek taki zilustrowano w kodzie znajdującym się na rysunku 8.59.

```
1 SELECT calendaryear AS Year, calendarquarter AS Quarter,  
2 SUM(NumOrders) AS Orders_Num,  
3 LEAD(SUM(NumOrders),1,0) OVER (PARTITION BY calendarquarter  
4 ORDER BY calendarquarter) AS Orders_Num_YQ_Next,  
5 LEAD(SUM(NumOrders),2,0) OVER (PARTITION BY calendarquarter  
6 ORDER BY calendarquarter) AS Orders_Num_YQ_2Next  
7 FROM (SELECT calendaryear, calendarquarter,  
8 COUNT(DISTINCT orderkey) AS NumOrders  
9 FROM factonlinesales fos INNER JOIN dimdate dd  
10 ON fos.orderdatekey= dd.datekey  
11 WHERE calendaryear IN (2022, 2023, 2024)  
12 GROUP BY calendaryear, calendarquarter  
13 )  
14 GROUP BY calendaryear, calendarquarter  
15 ORDER BY 1, 2;
```

Rys. 8.59. Przykład użycia funkcji *LEAD* w partycji danych z opcją offsetu

Źródło: opracowanie własne.

Kod ten ma na celu wyznaczenie liczby zamówień w danym kwartale podanego roku (*Orders_Num*), liczbę zamówień w tym samym kwartale roku następnego (*Orders_YQ_Next*) oraz liczbę zamówień w tym samym kwartale dwa lata później (*Orders_YQ_2Next*). Obliczenie tej ostatniej wartości wymaga użycia w piątej linii kodu parametru funkcji *LEAD* określającego offset równy 2. Rezultat wykonania powyższego zapytania przedstawiono na rysunku 8.60.

	YEAR	QUARTER	ORDERS_NUM	ORDERS_NUM_YQ_NEXT	ORDERS_NUM_YQ_2NEXT
1	2022	1	1190	518	1538
2	2022	2	1780	1823	3065
3	2022	3	1705	1997	3899
4	2022	4	1099	1223	524
5	2023	1	518	1538	0
6	2023	2	1823	3065	0
7	2023	3	1997	3899	0
8	2023	4	1223	524	0
9	2024	1	1538	0	0
10	2024	2	3065	0	0
11	2024	3	3899	0	0
12	2024	4	524	0	0

Rys. 8.60. Rezultat wykonania zapytania z rysunku 8.59

Źródło: opracowanie własne.

W celu weryfikacji poprawności rozwiązania można dokonać analizy danych np. w czwartym wierszu, czyli czwartym kwartale 2022 roku. Liczba zamówień w tym kwartale wynosi 1099 (*Orders_Num*). W tym samym wierszu znajduje się również informacja o liczbie zamówień w czwartym kwartale 2023 roku (*Orders_YQ_Next*) i 2024 roku (*Orders_YQ_2Next*). Wartości te można porównać z wartościami dla przyszłych okresów, znajdującymi się w kolumnie *Orders_Num* odpowiednio w wierszach 8 i 12. Jak można zauważyć, wszystkie te liczby są odpowiednio sobie równe, a zatem należy uznać, że zapytanie skonstruowano poprawnie.

Zarówno funkcja *LAG*, jak i *LEAD* są bardzo wydajną techniką dostępu do rekordów w tej samej tabeli bez konieczności tworzenia dodatkowych wewnętrznych złączeń. Możliwość ich wykorzystania w analizie szeregów czasowych sprawia, że są one wyjątkowo użyteczne jako narzędzie wspierające decyzje w obszarze zarządzania finansami, zarządzania sprzedażą czy kontroli kosztów funkcjonowania organizacji.

8.5. Tworzenie tabel przestawnych – PIVOT

Dotychczasowe wyniki analiz były prezentowane w postaci tradycyjnej tabelki, w której nazwy kolumn opisywały znaczenie danych. Zazwyczaj efekt ten jest osiągany poprzez zdefiniowanie odpowiedniego aliasu kolumny. Istnieje jednak możliwość, aby nazwa kolumny oraz wiersz były opisane wartościami pochodzącymi z bazy danych, co znacznie ułatwi interpretację

wartości odczytywanych na przecięciu określonego wiersza i kolumny. Taką postać prezentacji danych określa się mianem tabeli przestawnej (ang. *pivot table*) i jest ona bardzo popularnym sposobem analizowania danych w praktyce biznesowej.

Posługiwanie się tabelą przestawną w języku SQL wymaga użycia klauzuli *PIVOT* w instrukcji *SELECT*. Ogólną składnię najprostszego takiego polecenia przedstawiono na rysunku 8.61.

```
1 SELECT column_1, column_2, ...
2 FROM data_source
3 PIVOT (aggregate_function FOR pivot_column
4       IN pivot_column_values )
5 WHERE .....
```

Rys. 8.61. Składnia instrukcji *SELECT* z klauzulą *PIVOT*

Źródło: opracowanie własne.

Zbudowanie tabeli przestawnej za pomocą polecenia *SELECT* wymaga:

- podania rodzaju funkcji analitycznej (*aggregate_function*), jaka zostanie użyta do wykonania obliczeń;
- wskazania kolumny tabeli źródłowej (*pivot_column*), której wartości posłużą do opisu kolumn tabeli przestawnej;
- wskazania listy wartości (*pivot_column_values*) dla kolumny *pivot_column*, które zostaną użyte do opisu kolumn tabeli przestawnej.

Przykład obrazujący generowanie tabeli przestawnej w języku SQL za-prezentowano na rysunku 8.62.

```
1 SELECT * FROM (
2     SELECT calendaryear Year, monthnumberofyear Month,
3           ROUND(SUM(transactionprice*quantity),0)/1000 AS Sales
4     FROM factonlinesales fos INNER JOIN dimdate dd
5     ON fos.orderdatekey = dd.datekey
6     GROUP BY calendaryear, monthnumberofyear
7           ) asq
8 PIVOT ( SUM(Sales) FOR Year IN ( 2021, 2022, 2023, 2024) )
9 ORDER BY 1;
```

Rys. 8.62. Przykład tworzenia prostej tabeli przestawnej

Źródło: opracowanie własne.

Podzapytanie wygeneruje tabelę, w której znajdują się trzy kolumny opisujące odpowiednio rok, miesiąc i wartości sprzedaży w danym miesiącu danego roku. Następnie na podstawie tego zbioru wynikowego tworzona jest tabela przestawna, której kolumny będą opisane wartościami znajdującymi się w kolumnie *CalendarYear*, natomiast wiersze – wartościami z kolumny o aliasie *Month*. Na przecięciu miesiąca i roku zostanie umieszczona wartość sprzedaży dla tego okresu czasu. Rezultat wykonania zapytania z rysunku 8.62 przedstawiono na rysunku 8.63.

MONTH	2021	2022	2023	2024
1	202.209	475.06	419.407	502.758
2	236.885	511.254	462.303	610.584
3	168.827	597.136	503.531	971.004
4	550.813	599.566	660.063	1082.719
5	752.817	876.757	1357.141	1512.922
6	266.946	1000.261	892.102	2072.209
7	442.035	344.347	688.634	1009.845
8	418.196	700.8	628.918	2297.984
9	514.192	785.787	830.388	1159.79
10	432.458	357.154	499.184	184.045
11	298.898	297.94	314.128	231.948
12	381.345	560.212	529.458	236.053

Rys. 8.63. Rezultat wykonania kodu z rysunku 8.62

Źródło: opracowanie własne.

W tym przypadku opis kolumn tabeli wynikowej, reprezentujący rok sprzedaży, jest czytelny i zrozumiały dla użytkownika końcowego. Jeśli jednak dane znajdujące się w kolumnie tabeli źródłowej, użytej do opisu kolumn tabeli przestawnej, nie mają jednoznacznej interpretacji, wówczas konieczne staje się wprowadzenie dla nich odpowiednich aliasów. Przykład zastosowania tego rodzaju techniki zamieszczono na rysunku 8.64.

W tabeli *DimDate*, będącej źródłem danych dotyczących czasu, wartości opisujące kwartał są liczbami od 1 do 4. W celu uniknięcia błędnej interpretacji tej wartości (jako numeru miesiąca w roku) każdą kolumnę reprezentującą kwartał roku opisano jego nazwą (11 linia kodu źródłowego), zawierającą znak Q i numer kwartału w roku kalendarzowym. Tego rodzaju niejednoznaczności danych można zapobiec już na etapie projektowania tabeli wymiaru, definiując w niej kolumnę o nazwie czytelnej i poprawnie rozumianej przez użytkownika, np. *QuarterName*, w której kwartał będzie reprezentowany w postaci takich wartości, jak Q1 2025...Q4 2025 lub Q1/25...Q4/25.


```

1 SELECT * FROM (
2     SELECT calendaryear Year, calendarquarter Quarter ,
3           salesterritoryname Region,
4           ROUND (SUM(transactionprice*quantity),0)/1000 AS Sales
5     FROM factonlinesales fos INNER JOIN dimdate dd
6         ON fos.orderdatekey = dd.datekey
7     INNER JOIN dimsalesterritory dst
8         ON dst.salesterritorykey = fos.salesterritorykey
9     GROUP BY calendaryear, calendarquarter, salesterritoryname
10    ) asq
11 PIVOT ( SUM(Sales) FOR Quarter IN ( 1 AS Q1, 2 AS Q2, 3 AS Q3, 4 AS Q4) )
12 ORDER BY 1, 2 ;

```

Rys. 8.64. Przykład tworzenia tabeli przestawnej z aliasami kolumn

Źródło: opracowanie własne.

Rezultat wykonania powyższego zapytania przedstawiono na rysunku 8.65.

YEAR	REGION	Q1	Q2	Q3	Q4
2021	APAC	(null)	0.03	(null)	0.259
2021	EMEA	484.413	1145.501	938.48	736.03
2021	NACA	123.508	425.046	435.943	376.411
2022	APAC	(null)	(null)	3.983	(null)
2022	EMEA	1059.142	1623.339	1246.996	807.44
2022	NACA	524.308	853.244	579.955	407.866
2023	APAC	0.799	2.651	0.008	(null)
2023	EMEA	1131.922	2088.211	1518.738	867.952
2023	NACA	252.52	818.443	629.194	474.817
2024	APAC	(null)	2.431	7.348	4.029
2024	EMEA	1445.957	3027.81	2999.496	411.063
2024	NACA	638.389	1637.609	1460.776	236.955

Rys. 8.65. Rezultat wykonania kodu z rys. 8.64

Źródło: opracowanie własne.

Analizując zbiór wynikowy zapytania (rys. 8.65), można zauważyć w tabeli przestawnej występowanie wartości NULL, oznaczającej brak wartości określonej funkcji agregacji *SUM* dla danej kombinacji wartości opisujących wiersz i kolumnę, co w praktyce oznacza brak sprzedaży w danym regionie i kwartale danego roku kalendarzowego.

Bardzo ciekawym rozwiązaniem podczas tworzenia tabeli przestawnej jest użycie w jej definicji wielu funkcji agregacji (tych samych lub różnych), co zobrazowano na rysunku 8.66.

```

1 SELECT * FROM (
2     SELECT calendaryear Year, calendarquarter Quarter,
3         salesterritoryname Region,
4         ROUND(SUM(transactionprice*quantity),0)/1000 AS Sales,
5         COUNT(DISTINCT orderkey) NumOrders
6     FROM factonlinesales fos INNER JOIN dimdate dd
7         ON fos.orderdatekey = dd.datekey
8     INNER JOIN dimsalesterritory dst
9         ON dst.salesterritorykey = fos.salesterritorykey
10    GROUP BY calendaryear, calendarquarter , salesterritoryname
11        ) asq
12    PIVOT ( SUM(Sales) AS Total_Sales, SUM(NumOrders) AS Total_Orders
13        FOR Quarter IN ( 3 AS Q3, 4 AS Q4) )
14 ORDER BY 1, 2;

```

Rys. 8.66. Przykład wielokrotnego użycia tej samej funkcji agregacji w tabeli przestawnej

Źródło: opracowanie własne.

W przykładzie tym dwukrotnie wykorzystano funkcję *SUM*. Po raz pierwszy zastosowano ją do wyznaczenia całkowitej wartości sprzedaży (*Total_Sales*), po raz drugi – do obliczenia liczby zamówień (*Total_Orders*) w trzecim i czwartym kwartale kolejnych lat w poszczególnych regionach sprzedaży. Na rysunku 8.67 zaprezentowano rezultat wykonania zapytania umieszczonego na rysunku 8.66.

YEAR	REGION	Q3_TOTAL_SALES	Q3_TOTAL_ORDERS	Q4_TOTAL_SALES	Q4_TOTAL_ORDERS
2021	APAC	(null)	(null)	0.259	2
2021	EMEA	938.48	761	736.03	618
2021	NACA	435.943	401	376.411	381
2022	APAC	3.983	1	(null)	(null)
2022	EMEA	1246.996	1133	807.44	707
2022	NACA	579.955	571	407.866	392
2023	APAC	0.008	1	(null)	(null)
2023	EMEA	1518.738	1289	867.952	770
2023	NACA	629.194	707	474.817	453
2024	APAC	7.348	6	4.029	1
2024	EMEA	2999.496	2522	411.063	344
2024	NACA	1460.776	1371	236.955	179

Rys. 8.67. Rezultat wykonania kodu z rysunku 8.66

Źródło: opracowanie własne.

Przykład użycia różnych funkcji agregacji w definicji tabeli przestawnej zobrazowano na rysunku 8.68, natomiast na rysunku 8.69 uwidoczniono wynik

wykonania tego zapytania. Funkcje *MIN* i *MAX*, użyte w tym kodzie, umożliwiają wyznaczenie daty pierwszego i ostatniego zakupu w trzecim i czwartym kwartale kolejnych lat w kraju o nazwie *Germany*.

```

1 SELECT * FROM (
2     SELECT calendaryear Year, calendarquarter Quarter, fulldate
3     FROM factonlinesales fos INNER JOIN dimdate dd
4         ON fos.orderdatekey = dd.datekey
5     INNER JOIN dimsalesterritory dst
6         ON dst.salesterritorykey = fos.salesterritorykey
7     WHERE countryname = 'Germany'
8     ) asq
9 PIVOT ( MIN(fulldate) AS First_Purchase_Date,
10         MAX(fulldate) AS Last_Purchase_Date
11         FOR Quarter IN ( 3 AS Q3, 4 AS Q4 ) )
12 ORDER BY 1, 2;

```

Rys. 8.68. Przykład użycia różnych funkcji agregacji w tabeli przestawnej

Źródło: opracowanie własne.

YEAR	Q3_FIRST_PURCHASE_DATE	Q3_LAST_PURCHASE_DATE	Q4_FIRST_PURCHASE_DATE	Q4_LAST_PURCHASE_DATE
2021	01-07-2021	22-09-2021	18-11-2021	20-12-2021
2022	01-07-2022	23-09-2022	17-10-2022	27-12-2022
2023	03-07-2023	23-09-2023	17-10-2023	28-12-2023
2024	01-07-2024	27-09-2024	09-10-2024	31-12-2024

Rys. 8.69. Rezultat wykonania zapytania z rysunku 8.68

Źródło: opracowanie własne

W dotychczasowych przykładach definicji tabeli przestawnej wykorzystano tylko jedną kolumnę po klauzuli *FOR*. W wyniku takiej składni każda kolumna tej tabeli była opisana danymi reprezentującymi wyłącznie jeden wymiar. Istnieje jednak możliwość użycia wielu wymiarów, których wartości atrybutów będą przypisane do kolejnych kolumn takiej tabeli. Przykład taki przedstawiono na rysunku 8.70.

```

1 SELECT * FROM (
2     SELECT calendaryear Year, calendarquarter Quarter,
3           categoryname Category, channelkey Channel,
4           ROUND(SUM(transactionprice*quantity),0)/1000 AS Sales
5     FROM factonlinesales fos INNER JOIN dimdate dd
6        ON fos.orderdatekey = dd.datekey
7     INNER JOIN dimsalesterritory dst
8        ON dst.salesterritorykey = fos.salesterritorykey
9     INNER JOIN dimproduct dp ON dp.productkey = fos.productkey
10    GROUP BY calendaryear, calendarquarter, categoryname, channelkey
11    ) asq
12 PIVOT ( SUM(Sales) AS Sales FOR (Channel, Quarter)
13        IN ( (1,1) AS Web_Q1, (2,1) AS Mobile_Q1,
14            (1,2) AS Web_Q2, (2,2) AS Mobile_Q2))
15 ORDER BY 1, 2 ;

```

Rys. 8.70. Przykład użycia wielu wymiarów do opisu kolumny tabeli przestawnej

Źródło: opracowanie własne.

Definicja tabeli przestawnej zawiera w tym przypadku kolumny dwóch wymiarów. Kolumna *Channel* reprezentuje kanał złożenia zamówienia przez klienta, a kolumna *Quarter* kwartał, w którym to się stało. W celu poprawnej identyfikacji kolumny tabeli konieczne jest przypisanie aliasu do każdej kombinacji wartości, jakie mogą znaleźć się w kolumnach oznaczonych jako *Channel* i *Quarter*. W prezentowanym kodzie aliasu *Web_Q1* użyto do opisu kombinacji wartości (1,1), gdzie pierwsza wartość oznacza stronę webową, a druga – pierwszy kwartał.

Rezultat omawianego zapytania, przedstawiony na rysunku 8.71, ograniczono do pierwszego i drugiego kwartału kolejnych lat oraz kanałów sprzedażowych o identyfikatorach 1 (strona webowa) i 2 (aplikacja mobilna).

YEAR	CATEGORY	WEB_Q1_SALES	MOBILE_Q1_SALES	WEB_Q2_SALES	MOBILE_Q2_SALES
2021	Accessories	(null)	(null)	17.752	17.414
2021	Bikes	330.685	277.236	756.247	762.659
2021	Clothing	(null)	(null)	8.903	7.601
2022	Accessories	31.261	34.488	48.703	47.713
2022	Bikes	765.747	720.662	1157.515	1174.987
2022	Clothing	16.541	14.752	22.658	25.007
2023	Accessories	(null)	(null)	29.696	31.255
2023	Bikes	666.7	718.541	1356.868	1464.695
2023	Clothing	(null)	(null)	12.886	13.906
2024	Accessories	41.289	42.658	82.209	80.72
2024	Bikes	958.629	1001.204	2123.617	2303.929
2024	Clothing	19.761	20.805	38.741	38.634

Rys. 8.71. Rezultat wykonania kodu z rysunku 8.70

Źródło: opracowanie własne.

Przedstawione techniki eksploatacji danych w języku SQL mogą wydawać się dosyć trudne ze względu na wyższy stopień złożoności zapytań. Nie bez znaczenia dla użytkownika końcowego jest także czas ich wykonania. Dlatego też w praktyce definicję zapytania analitycznego zapisuje się w postaci standardowego widoku, używając polecenia *CREATE VIEW*. Takie podejście zawsze zapewnia wygenerowanie wyniku na podstawie aktualnie posiadanych danych. Jeśli jednak analiza dotyczy danych historycznych, szczególnie dużych ich ilości, wówczas najlepszym rozwiązaniem zapewniającym skrócenie czasu wykonania zapytania jest zapisanie jego definicji jako obiektu w bazie danych w postaci widoku zmaterializowanego za pomocą polecenia *CREATE MATERIALIZED VIEW*. W ten sposób zarówno składnia instrukcji *SELECT*, jak i dane będące rezultatem jej wykonania będą trwale składowane w bazie danych, niezależnie od tabel źródłowych użytych w tym poleceniu.

8.6. Zadania do samodzielnego rozwiązania

1. Wykorzystując fizyczny model danych przykładowej analitycznej bazy danych, przedstawiony na rysunku 8.1, proszę napisać kod w języku SQL, który umożliwi wykonanie następujących zadań:
 - Określić wartość sprzedaży w kolejnych latach i kategoriach produktowych. Dodatkowo należy podać informację o tej wartości zarówno dla każdego roku, jak i dla wszystkich lat. Zbiór wynikowy powinien zawierać dane o numerze roku, nazwie kategorii i wartości sprzedaży. Proszę uporządkować ten zbiór rosnąco według roku i kategorii produktowej.
 - Wyznaczyć dla każdego roku liczbę klientów, którzy złożyli zamówienia z poszczególnych krajów, oraz określić procentowy udział tej liczby w całkowitej rocznej liczbie klientów danego roku. W zbiorze wynikowym należy umieścić kolumny opisujące numer roku, nazwę kraju, liczbę klientów z danego kraju oraz opisany wyżej wskaźnik procentowego udziału. Proszę uporządkować ten zbiór według roku i nazwy kraju.
 - Zbudować ranking trzech najlepiej sprzedających się produktów w poszczególnych miesiącach w kolejnych latach. Jako kryterium rankingu należy przyjąć liczbę sprzedanych sztuk. W zbiorze wynikowym należy umieścić informację o numerze roku, numerze miesiąca w roku,

nazwie produktu oraz liczbie sprzedanych sztuk. Proszę uporządkować ten zbiór rosnąco według roku, miesiąca i pozycji w rankingu.

- Wyznaczyć liczbę zamówień w kolejnych miesiącach poszczególnych lat dla poszczególnych kanałów sprzedaży i porównać ją z liczbą zamówień złożonych w tym samym miesiącu roku ubiegłego w tym samym kanale. Zbiór wynikowy powinien zawierać kolumny opisujące numer roku, numer miesiąca w roku, nazwę kanału sprzedaży, liczbę zamówień w danym miesiącu, liczbę zamówień dla tego miesiąca roku ubiegłego oraz procentową zmianę tych wartości (miesiąc roku bieżącego do miesiąca roku ubiegłego). Proszę uporządkować ten zbiór według roku, miesiąca i nazwy kanału.
 - Zbudować tabelę przestawną, przedstawiającą informacje o liczbie klientów i wartości sprzedaży w poszczególnych latach i kanałach sprzedaży. Informację o numerze roku należy umieścić w wierszach, natomiast o kanałach sprzedaży – w kolumnach. Do opisu kanału proszę użyć jego nazwy. Proszę uporządkować zbiór wynikowy według roku.
2. Proszę przeanalizować kod źródłowy w języku SQL, przedstawiony na rysunku 8.72.

```
1 SELECT DISTINCT Year, Quarter,
2     ROUND(AVG(OrderValue)
3           OVER (PARTITION BY Quarter,2) AS Avg_Q_OrderValue,
4     ROUND(AVG(OrderValue)
5           OVER (PARTITION BY Year,2) AS Avg_Y_OrderValue
6 FROM ( SELECT calendaryear Year, calendarquarter AS Quarter,
7           orderkey AS OrderID,
8           SUM(transactionprice*quantity) AS OrderValue
9 FROM factonlinesales fos INNER JOIN dimdate dd
10      ON fos.orderdatekey= dd.datekey
11      GROUP BY calendaryear, calendarquarter, orderkey )
12 ORDER BY 1 ;
```

Rys. 8.72. Przykład #1 kodu SQL do modyfikacji

Źródło: opracowanie własne.

Następnie proszę odpowiedzieć na poniższe pytania:

- Jak należy zinterpretować dane znajdujące się w zbiorze wynikowym tego zapytania?

- Co należy zmienić w tym kodzie, aby zbiór wynikowy zapytania opisywał średnią wartość zamówienia w poszczególnych kwartałach kolejnych lat?
3. Proszę dokonać analizy kodu źródłowego w języku SQL, przedstawionego na rysunku 8.73.

```

1 SELECT calendarquarter AS Quarter, monthnumberofyear AS Month,
2     COUNT(DISTINCT orderkey) AS Orders_Num,
3     COUNT(DISTINCT orderkey) -
4     LAG(COUNT(DISTINCT orderkey),1,0)
5     OVER (ORDER BY monthnumberofyear) AS Diff_MtM
6 FROM factonlinesales fos INNER JOIN dimdate dd
7     ON fos.orderdatekey= dd.datekey
8 WHERE calendaryear = 2024
9 GROUP BY calendarquarter, monthnumberofyear
10 ORDER BY 1, 2 ;

```

Rys. 8.73. Przykład #2 kodu SQL do modyfikacji

Źródło: opracowanie własne.

Kod ten wyznacza liczbę zamówień w kolejnych miesiącach roku 2024 (kolumna *Orders_Num*) oraz różnicę między liczbą zamówień w miesiącu bieżącym i miesiącu bezpośrednio go poprzedzającym (kolumna *DiffMtM*). Proszę zmodyfikować ten kod poprzez dodanie nowej kolumny o nazwie *DiffMtFirstM* w taki sposób, aby możliwe było wyznaczenie w obrębie danego kwartału różnicy między liczbą zamówień w danym miesiącu i liczbą zamówień w pierwszym miesiącu tego kwartału. Zbiór wynikowy zmodyfikowanego zapytania powinien wyglądać tak, jak to przedstawiono na rysunku 8.74.

QUARTER	MONTH	ORDERS_NUM	DIFF_MTM	DIFF_MTFIRSTM
1	1	347	347	0
1	2	471	124	124
1	3	720	249	373
2	4	764	44	0
2	5	1026	262	262
2	6	1275	249	511
3	7	1353	78	0
3	8	1707	354	354
3	9	839	-868	-514
4	10	167	-672	0
4	11	163	-4	-4
4	12	194	31	27

Rys. 8.74. Oczekiwana zawartość zbioru wynikowego dla zmodyfikowanego przykładu #2 kodu SQL

Źródło: opracowanie własne.

Zakończenie

Celem publikacji było przedstawienie metodyki projektowania hurtowni danych i data mart oraz podstawowych technik eksploatacji zgromadzonych w nich danych za pomocą polecenia *SELECT* języka SQL. Cel ten osiągnięto poprzez prezentację pełnego cyklu działań, które pozwalają dokonać transformacji informacyjnych potrzeb użytkownika biznesowego do postaci analitycznej platformy składowania danych zapewniającej efektywne zaspokojenie tych potrzeb.

Niniejsze opracowanie stanowi kompendium wiedzy w zakresie projektowania analitycznych baz danych typu hurtownia danych i data mart oraz przetwarzania zgromadzonych w nich danych, zorientowanego na wspomaganie procesów decyzyjnych. Jego Autorzy mają jednak świadomość, że proces projektowania tego rodzaju rozwiązania do składowania danych jest tylko jednym z wielu etapów prowadzących do zbudowania w pełni funkcjonalnej platformy analitycznej, w której zostaną uwzględnione wszystkie elementy jej architektury i procesy nadające im wartość. Przedstawione w publikacji techniki eksploatacji danych również nie obejmują wszystkich rodzajów działań analitycznych, a jedynie umożliwiają wykonanie podstawowych operacji, których rezultaty mogą stanowić przyczynek do wykonania pogłębionej eksploracji danych. W tym kontekście Autorzy dostrzegają potrzebę rozszerzenia w kolejnych opracowaniach zaprezentowanych tutaj treści o zagadnienia związane

z projektowaniem i realizacją działań ETL, analizą wizualną w narzędziach business intelligence czy też drążeniem danych w technologiach data science, które to działania umożliwią monetyzację posiadanych danych – zarówno tych transakcyjnych, jak i analitycznych.

Autorzy mają nadzieję, że treści zawarte w publikacji będą stanowiły dla Czytelnika wartościowy zasób informacyjny, który może być wykorzystany w rozwoju lub pozyskaniu nowych kompetencji związanych z tworzeniem rozwiązań informatycznych wspierających funkcjonowanie nowoczesnych organizacji biznesowych.

Bibliografia

- [1] ACI INFOTECH, Comparing Top-Down and Bottom-Up Data Model Designs: Benefits and Use Cases, ACI Insights Hub, 8.11.2024. [Online]. Available: <https://www.linkedin.com/pulse/comparing-top-down-bottom-up-data-model-designs-benefits-use-ektvcl/>. [Data uzyskania dostępu: 20.04.2025].
- [2] Adamson C., Star schema. The complete reference, New York: McGraw-Hill Companies, 2010.
- [3] Anderson C., Creating a Data-Driven Organization: Practical Advice from the Trenches. 1st Edition, O'Reilly Media, Inc., 2015.
- [4] Assaf W., Gunda S., Myers P., Dimensional modeling in Microsoft Fabric Warehouse: Dimension tables, Microsoft, 6.04.2025. [Online]. Available: <https://learn.microsoft.com/en-us/fabric/data-warehouse/dimensional-modeling-dimension-tables>. [Data uzyskania dostępu: 10.05.2025].
- [5] Baragoin C., Marini M., Morgan C., Mueller O., Perkins A., Yim K., Building the Operational Data Store on DB2 UDB Using IBM Data Replication, WebSphere MQ Family, and DB2 Warehouse Manager, San Jose, California: IBM Corporation, International Technical Support Organization, 2001.
- [6] Becker B., Design Tip #133 Factless Fact Tables for Simplification, Kimball Group, 5.04.2011. [Online]. Available: <https://www.kimballgroup.com/2011/04/design-tip-133-factless-fact-tables-for-simplification/>. [Data uzyskania dostępu: 7.05.2025].
- [7] Becker B., Design Tip #167 Complementary Fact Table Type, Kimball Group, 17.06.2014. [Online]. Available: <https://www.kimballgroup.com/2014/06/design-tip-167-complementary-fact-table-types/>. [Data uzyskania dostępu: 3.05.2025].
- [8] Bikakis N., Papastefanatos G., Papaemmanouil O., Big Data Exploration, Visualization and Analytics, *Big Data Research*, vol. 18, p. 100123, 12.2019.
- [9] Bjerke M.B., Renger R., Being smart about writing SMART objectives, *Evaluation and Program Planning*, vol. 61, pp. 125–127, 2017.

- [10] Corr L., Stagnitto J., *Agile Data Warehouse Design: Collaborative Dimensional Modeling, from Whiteboard to Star Schema*, Leeds, UK: DecisionOne Press, 2013.
- [11] Franceschini F., Galetto M., Maisano D., *Designing Performance Measurement Systems. Theory and Practice of Key Performance Indicators*, Cham: Springer Nature Switzerland AG, 2019.
- [12] Inmon W., *Building the Data Warehouse*, 4th Edition, Indianapolis: Wiley Publishing, Inc., 2005.
- [13] Kaplan R.S., Norton D.P., *The Balanced Scorecard: Translating Strategy into Action*, Boston: Harvard Business Review Press, 1996.
- [14] Kimball R., Ross M., Becker B., Mundy J., Thornthwaite W., *The Kimball Group Reader. Relentlessly Practical Tools for Data Warehousing and Business Intelligence*, Indianapolis, Indiana: Wiley Publishing, Inc., 2010.
- [15] Kimball R., Ross M., *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, 3rd Edition, Indianapolis: John Wiley & Sons, Inc., 2013.
- [16] Liberatore M., Luo W., *The Analytics Movement: Implications for Operations Research*, *Interfaces*, vol. 40, no. 4, pp. 313–324, 2010.
- [17] Lu T., *Data Mart vs Data Warehouse: a Detailed Comparison*, Datacamp, 15.01.2025. [Online]. Available: <https://www.datacamp.com/blog/data-mart-vs-data-warehouse>. [Data uzyskania dostępu: 25.04.2025].
- [18] Marquardt A., *Data Warehouse vs. Operational Database! What? How? Which One?*, Airbyte, 16.12.2022. [Online]. Available: <https://airbyte.com/blog/data-warehouse-vs-operational-database>. [Data uzyskania dostępu: 25.04.2025].
- [19] Morrison R., *Data Driven Organisation Design: Sustaining the Competitive Edge through Organisational Analytics*. 2nd Edition, London: Kogan Page, 2021.
- [20] Oracle, *Get the best out of Oracle Partitioning. A practical guide and reference*. Version 19c., 04.2020. [Online]. Available: <https://www.oracle.com/docs/tech/partitioning-guide-2703320.pdf>. [Data uzyskania dostępu: 3.05.2025].
- [21] Parmenter D., *Key performance indicators: developing, implementing, and using winning KPIs*, Fourth Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2019.
- [22] Paulraj P., *Data Warehousing Fundamentals For IT Professionals*, Second Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2010.
- [23] Potineni P., *Oracle® Database. Data Warehousing Guide*, 21c, 02.2022. [Online]. Available: <https://docs.oracle.com/en/database/oracle/oracle-database/21/dwhsg/database-data-warehousing-guide.pdf>. [Data uzyskania dostępu: 20.04.2025].
- [24] Rajasekaran K., Saravanan P., Prabakaran P., *Comparative study on data warehouse tables and schema – an overview*, *Adalya Journal*, vol. 9, no. 4, pp. 126–132, April 2020.
- [25] Ross M., *Design Tip #152 Slowly Changing Dimension Types 0, 4, 5, 6 and 7*, DecisionWorks Consulting, 5.02.2013. [Online]. Available: <https://decisionworks.com/2013/02/design-tip-152-slowly-changing-dimension-types-0-4-5-6-7/>. [Data uzyskania dostępu: 10.05.2025].
- [26] Ross M., *Design Tip #48: De-Clutter With Junk (Dimensions)*, DecisionWorks Consulting, 7.08.2003. [Online]. Available: <https://decisionworks.com/2003/08/design-tip-48-de-clutter-with-junk-dimensions-by-margy/>. [Data uzyskania dostępu: 7.05.2025].

- [27] Serra J., Deciphering Data Architectures: Choosing Between a Modern Data Warehouse, Data Fabric, Data Lakehouse, and Data Mesh, Sebastopol, CA: O'Reilly Media, Inc., 2024.
- [28] Silvers F., Building and Maintaining a Data Warehouse, Boca Raton, FL: Taylor & Francis Group, LLC, 2008.
- [29] Sinha R., Analytical Study of Data Warehouse, *International Journal of Management, IT & Engineering*, vol. 8, no. 1, pp. 105–115, 2019.
- [30] Tanimura C., SQL for Data Analysis. Advanced Techniques for Transforming Data into Insights, Sebastopol, CA: O'Reilly Media, Inc., 2021.
- [31] Univeristy of California, Office of the President, Human Resources, 30.04.2025. [Online]. Available: https://www.ucop.edu/local-human-resources/_files/performance-appraisal/How+to+write+SMART+Goals+v2.pdf. [Data uzyskania dostępu: 5.05.2025].
- [32] Watson H., Tutorial: Business Intelligence – Past, Present, and Future, *Communications of the Association for Information Systems*, vol. 25, pp. 487–510, 2009.