

Marco Russo
Alberto Ferrari

Microsoft SQL Server 2016
Analysis Services

Modelowanie tabelaryczne

Przekład: Jakub Niedźwiedź, Witold Sikorski, Marek Włodarz

APN Promise, Warszawa 2017

Microsoft SQL Server 2016 Analysis Services: Modelowanie tabelaryczne

Authorized Polish translation of the English language edition entitled
Tabular Modeling in Microsoft SQL Server Analysis Services, Second Edition, by
Marco Russo and Alberto Ferrari

ISBN: 978-1-5093-0277-2

Copyright © 2017 by Marco Russo and Alberto Ferrari

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

Polish language edition published by APN PROMISE SA Copyright © 2017

Autoryzowany przekład z wydania w języku angielskim, zatytułowanego:
Tabular Modeling in Microsoft SQL Server Analysis Services, Second Edition, by
Marco Russo and Alberto Ferrari

ISBN: 978-1-5093-0277-2

Wszystkie prawa zastrzeżone. Żadna część niniejszej książki nie może być powielana ani rozpowszechniana w jakiegokolwiek formie i w jakikolwiek sposób (elektroniczny, mechaniczny), włącznie z fotokopiowaniem, nagrywaniem na taśmy lub przy użyciu innych systemów bez pisemnej zgody wydawcy.

APN PROMISE SA, ul. Domaniewska 44a, 02-672 Warszawa
tel. +48 22 35 51 600, fax +48 22 35 51 699
e-mail: mspress@promise.pl

Książka ta przedstawia poglądy i opinie autora. Przykłady firm, produktów, osób i wydarzeń opisane w niniejszej książce są fikcyjne i nie odnoszą się do żadnych konkretnych firm, produktów, osób i wydarzeń, chyba że zostanie jednoznacznie stwierdzone, że jest inaczej. Ewentualne podobieństwo do jakiegokolwiek rzeczywistej firmy, organizacji, produktu, nazwy domeny, adresu poczty elektronicznej, logo, osoby, miejsca lub zdarzenia jest przypadkowe i niezamierzone.

Microsoft oraz znaki towarowe wymienione na stronie <http://www.microsoft.com/about/legal/en/us/IntellectualProperty/Trademarks/EN-US.aspx> są zastrzeżonymi znakami towarowymi grupy Microsoft. Wszystkie inne znaki towarowe są własnością ich odpowiednich właścicieli.

APN PROMISE SA dołożyła wszelkich starań, aby zapewnić najwyższą jakość tej publikacji. Jednakże nikomu nie udziela się rękojmi ani gwarancji.
APN PROMISE SA nie jest w żadnym wypadku odpowiedzialna za jakiegokolwiek szkody będące następstwem korzystania z informacji zawartych w niniejszej publikacji, nawet jeśli APN PROMISE została powiadomiona o możliwości wystąpienia szkód.

ISBN: 978-83-7541-301-4

Przekład: Jakub Niedźwiedź, Witold Sikorski, Marek Włodarz
Redakcja: Marek Włodarz
Korekta: Ewa Swędrowska
Skład i łamanie: MAWart Marek Włodarz

Spis treści

<i>Przedmowa</i>	xiii
<i>Wprowadzenie</i>	xv
1 Wprowadzenie do modelu tabelarycznego	1
Model semantyczny w Analysis Services	1
Czym jest usługa Analysis Services i dlaczego należy jej używać?	1
Krótka historia Analysis Services	3
Model tabelaryczny i wielowymiarowy	4
Model tabelaryczny	4
Model wielowymiarowy	7
Po co są dwa modele?	8
Przyszłość Analysis Services	10
Azure Analysis Services	10
Wybór odpowiedniego modelu dla naszego projektu	11
Licencjonowanie	11
Aktualizacja poprzednich wersji Analysis Services	12
Prostota korzystania	12
Kompatybilność z Power Pivot	13
Kompatybilność z Power BI	13
Cechy wydajności zapytań	13
Cechy wydajności przetwarzania	14
Uwarunkowania sprzętowe	14
BI czasu rzeczywistego	15
Narzędzia klienckie	16
Porównanie funkcji	16
Porównanie DAX i MDX	18
Język DAX	18
Język MDX	19
Wybieranie języka zapytań dla modelu tabelarycznego	20
Wprowadzenie do silników kalkulacyjnych modelu tabelarycznego	20
Wprowadzenie do VertiPaq	21
Wprowadzenie DirectQuery	22
Poziomy zgodności modelu tabelarycznego (1200 kontra 110x)	24

Analysis Services i Power BI	26
Podsumowanie	27

2 Rozpoczynanie pracy w modelu tabelarycznym..... 29

Budowanie środowiska deweloperskiego	29
Komponenty środowiska deweloperskiego	29
Licencje	32
Proces instalacji.....	32
Korzystanie z SQL Server Data Tools	41
Tworzenie nowego projektu.....	41
Konfiguracja nowego projektu.....	43
Import z PowerPivot	49
Importowanie z Power BI	50
Import wdrożonego projektu z Analysis Services.....	51
Zawartość projektu tabelarycznego	51
Budowa prostego modelu tabelarycznego.....	53
Ładowanie danych do tabel.....	53
Praca w widoku diagramu	64
Nawigowanie przy użyciu Tabular Model Explorer	68
Wdrażanie modelu tabelarycznego	70
Odpytywanie modelu tabelarycznego przy użyciu programu Excel	71
Połączenie z modelem tabelarycznym.....	72
Korzystanie z tabel przestawnych	75
Korzystanie z fragmentatorów.....	76
Sortowanie i filtrowanie wierszy i kolumn	79
Używanie formuł kostek Excela	81
Odpytywanie modelu tabelarycznego przy użyciu Power BI Desktop ...	83
Tworzenie połączenia z modelem tabelarycznym	83
Budowanie podstawowego raportu Power BI.....	85
Dodawanie wykresów i fragmentatorów	86
Interakcja z raportem	88
Korzystanie z SQL Server Management Studio	89
Importowanie z Power Pivot	92
Importowanie z Power BI Desktop	93
Używanie DAX Studio jako alternatywy dla SSMS	93
Podsumowanie	94

3 Ładowanie danych do modelu tabelarycznego	95
Istota źródła danych	95
Personifikacja	97
Poświadczenia po stronie serwera i po stronie klienta	99
Praca z dużymi tabelami	100
Ładowanie z serwera SQL Server	101
Ładowanie z listy tabel	104
Ładowanie z zapytania SQL	107
Ładowanie z widoków	108
Otwieranie istniejących połączeń	109
Ładowanie z programu Access	110
Ładowanie z Analysis Services	111
Korzystanie z edytora MDX	112
Ładowanie z tabelarycznej bazy danych	113
Ładowanie z pliku Excela	116
Ładowanie z pliku tekstowego	118
Ładowanie ze schowka	120
Ładowanie z raportu usług Reporting Services	123
Ładowanie danych przy użyciu źródła danych raportu	123
Ładowanie z raportów przy użyciu strumieniowych źródeł danych	129
Ładowanie ze strumieniowych źródeł danych	131
Ładowanie z SharePoint	133
Wybór właściwej metody ładowania danych	133
Podsumowanie	135
4 Wprowadzenie do DAX	137
Wprowadzenie do języka DAX	137
Składnia DAX	139
Typy danych języka DAX	140
Operatory DAX	143
Odwoływanie się do kolumn i miar	145
Funkcje agregujące	145
Funkcje tablicowe	146
Kontekst wykonania	147
CALCULATE i CALCULATETABLE	149
Zmienne	152
Miary	152
Kolumny obliczane	154

Tabele obliczane	156
Tworzenie zapytań w języku DAX	158
Formatowanie kodu DAX	159
DAX Formatter, DAX Studio i DAX Editor	161
Podsumowanie	162
5 Tworzenie hierarchii	163
Hierarchie podstawowe	163
Czym są hierarchie?	163
Kiedy budować hierarchie	165
Tworzenie hierarchii	166
Najlepsze praktyki projektowania hierarchii	167
Hierarchie obejmujące wiele tabel	168
Naturalne i nienaturalne hierarchie	170
Hierarchie rodzic-dziecko	171
Czym są hierarchie rodzic-dziecko?	171
Konfigurowanie hierarchii rodzic-dziecko	172
Operatory jednoargumentowe	177
Podsumowanie	182
6 Modelowanie danych w modelu tabelarycznym	183
Różne techniki modelowania danych	184
Korzystanie z bazy danych OLTP	186
Praca z modelami wymiarowymi	187
Praca z powoli zmieniającymi się wymiarami	188
Praca ze zdegenerowanymi wymiarami	192
Korzystanie z migawkowych tabel faktów	193
Korzystanie z widoków w celu odseparowania się od bazy danych	195
Typy relacji	197
Kardynalność relacji	198
Propagowanie filtru w relacjach	204
Stan aktywny relacji	208
Implementowanie relacji w języku DAX	210
Normalizacja kontra denormalizacja	211
Kolumny obliczane kontra zewnętrzny proces ETL	215
Odwołania cykliczne przy korzystaniu z tabel obliczanych	218
Podsumowanie	219

7	Tabular Model Scripting Language (TMSL)	221
	Definiowanie obiektów w TMSL	221
	Obiekt Model	223
	Obiekt DataSource	226
	Obiekt Table	228
	Obiekt Relationship	235
	Obiekt Perspective	237
	Obiekt Culture	238
	Obiekt Role	240
	Polecenia TMSL	242
	Operacje na obiektach w TMSL	243
	Operacje odświeżania danych i zarządzania bazą danych w TMSL	244
	Skrypty TMSL	245
	Podsumowanie	246
8	Warstwa prezentacji modelu tabelarycznego	247
	Ustawianie metadanych dla tabeli kalendarzowej	247
	Nazwy, sortowanie i formatowanie	248
	Nazywanie obiektów	248
	Ukrywanie kolumn i miar	251
	Porządkowanie miar i kolumn	251
	Sortowanie danych w kolumnie	253
	Formatowanie	256
	Perspektywy	259
	Właściwości związane z Power View	262
	Domyślny zestaw pól	262
	Właściwości sterujące zachowaniem tabeli	264
	Kluczowe wskaźniki wydajności (KPI)	265
	Translacje	268
	Tworzenie pliku translacji	270
	Zapisywanie tłumaczeń nazw w pliku translacji	271
	Wybieranie edytora dla plików translacji	273
	Importowanie pliku translacji	275
	Testowanie tłumaczeń przy użyciu narzędzi klienckich	276
	Usuwanie tłumaczenia	277
	Najlepsze praktyki dotyczące tłumaczeń	278
	Wybieranie ustawień językowych w modelu tabelarycznym	279

Zmienianie ustawień językowych przy korzystaniu ze zintegrowanego obszaru roboczego	281
Zmienianie ustawień językowych przy korzystaniu z serwera obszaru roboczego	282
Podsumowanie	282
9 Korzystanie z DirectQuery	283
Konfigurowanie DirectQuery	284
Ustawianie DirectQuery w środowisku deweloperskim	284
Włączanie trybu DirectQuery po wdrożeniu	291
Ograniczenia modelu tabelarycznego dla DirectQuery	294
Obsługiwane źródła danych	294
Ograniczenia dla źródeł danych	294
Ograniczenia modelowania danych	295
Ograniczenia dotyczące formuł DAX	295
Ograniczenia występujące w formułach MDX	297
Dostrajanie ograniczeń zapytań	298
Wybieranie pomiędzy trybem DirectQuery a VertiPaq	300
Podsumowanie	301
10 Bezpieczeństwo	303
Uwierzytelnianie użytkowników	303
Łączenie się z serwerem Analysis Services spoza domeny	304
Kerberos i problem podwójnego przeskoku	305
Role	306
Tworzenie ról bazodanowych	307
Członkostwo w wielu rolach	309
Zabezpieczenia administracyjne	310
Przyznawanie uprawnień poprzez rolę Server Administrator	310
Role bazodanowe i uprawnienia administracyjne	311
Zabezpieczenia danych	312
Podstawowe zabezpieczenia danych	313
Testowanie ról zabezpieczeń	314
Zaawansowane wyrażenia filtra wierszy	319
Zabezpieczenia w kolumnach i tabelach obliczanych	324
Korzystanie z tabeli uprawnień	325
Szacowanie wpływu zabezpieczeń danych na wydajność	326
Tworzenie zabezpieczeń dynamicznych	327

Funkcje DAX związane z zabezpieczeniami dynamicznymi	327
Implementowanie zabezpieczeń dynamicznych przy użyciu CUSTOMDATA	328
Implementowanie zabezpieczeń dynamicznych przy użyciu USERNAME	329
Zabezpieczenia w trybie DirectQuery	332
Zabezpieczenia i personifikacja w DirectQuery	332
Zabezpieczenia na poziomie wierszy w wersjach SQL Server wcześniejszych niż 2016	334
Monitorowanie zabezpieczeń	335
Podsumowanie	337
11 Przetwarzanie i partycjonowanie modelu tabelarycznego	339
Automatyzowanie wdrożenia na serwerze produkcyjnym	339
Partycjonowanie tabel	341
Definiowanie strategii partycjonowania	341
Definiowanie partycji tabeli w modelu tabelarycznym	344
Zarządzanie partycjami tabeli	348
Opcje przetwarzania	351
Dostępne operacje przetwarzania	352
Definiowanie strategii przetwarzania	358
Wykonywanie przetwarzania	361
Automatyzowanie przetwarzania	365
Korzystanie z poleceń TMSL	366
Korzystanie z SQL Server Integration Services	374
Korzystanie z bibliotek Analysis Management Objects (AMO) i Tabular Object Model (TOM)	377
Korzystanie z PowerShell	380
Przykładowe skrypty przetwarzania	381
Przetwarzanie bazy danych	381
Przetwarzanie tabel	382
Przetwarzanie partycji	383
Partycje kroczące	384
Podsumowanie	388
12 Wewnątrz VertiPaq	389
Struktury VertiPaq	390
Istota magazynu kolumnowego	390
Kodowanie wartości kontra kodowanie skrótów	394

Kompresja RLE	397
Kontrolowanie kodowania kolumn	399
Hierarchie i relacje	400
Segmentacja i partycjonowanie	402
Odczytywanie wewnętrznych metadanych VertiPaq	404
Korzystanie z DMV w celu sprawdzenia użycia pamięci przez VertiPaq	404
Interpretowanie raportów VertiPaq Analyzer	406
Użycie pamięci VertiPaq	410
Użycie pamięci na dane	410
Użycie pamięci podczas przetwarzania	412
Użycie pamięci w zapytaniach	413
Opcje przetwarzania	414
Co dzieje się podczas przetwarzania	415
Dostępne opcje przetwarzania	416
Podsumowanie	418
13 Programowy interfejs modelu tabelarycznego	419
Wprowadzenie do bibliotek AMO oraz TOM	419
Wprowadzenie do AMO	420
Wprowadzenie do TOM	422
Wprowadzenie do poleceń TMSL	428
Programowe tworzenie bazy danych	430
Automatyzowanie odświeżania danych i partycjonowania	434
Analizowanie metadanych	434
Manipulowanie modelem danych	437
Automatyzowanie wdrożenia projektu	439
Kopiowanie tej samej bazy danych na różne serwery	439
Wdrażanie pliku model.bim z wyborem nazwy bazy danych i serwera	440
Podsumowanie	442
14 Monitorowanie i dostrajanie usługi tabelarycznej	443
Znajdowanie procesu usług Analysis Services	443
Zasoby zużywane przez Analysis Services	445
CPU	445
Pamięć	447
Operacje I/O	449

Zrozumienie konfiguracji pamięci	450
Korzystanie z liczników wydajnościowych związanych z pamięcią	455
Korzystanie z dynamicznych widoków zarządzania	460
Widoki DMV przydatne do monitorowania usługi tabelarycznej	462
Automatyzowanie gromadzenia informacji monitorowania i dzienników	464
Liczniki wydajności	464
SQL Server Profiler	465
ASTrace	469
Flight Recorder	469
Zdarzenia rozszerzone	470
Inne narzędzia komercyjne	471
Monitorowanie odświeżania danych (przetwarzania)	471
Monitorowanie zapytań	475
Podsumowanie	478
15 Optymalizowanie modeli tabelarycznych	479
Optymalizowanie użycia pamięci	479
Usuwanie nieużywanych kolumn	480
Zmniejszanie wielkości słownika	480
Wybieranie typu danych	484
Redukowanie wielkości bazy danych poprzez wybór kolejności sortowania	486
Ulepszanie kodowania i rozmiaru bitowego	489
Optymalizowanie wielkich wymiarów	490
Projektowanie modeli tabelarycznych dla wielkich baz danych	495
Optymalizowanie kompresji poprzez podział kolumny	495
Optymalizowanie czasu przetwarzania wielkich tabel	497
Agregowanie tabel faktów na różnych poziomach szczegółowości ..	498
Projektowanie modeli tabelarycznych dla rozwiązań niemal czasu rzeczywistego	503
Wybieranie pomiędzy DirectQuery a VertiPaq	503
Wykorzystanie partycji	504
Redukowanie czasu ponownych obliczeń	507
Zarządzanie blokadami podczas przetwarzania	509
Podsumowanie	510

16 Wybieranie sprzętu i wirtualizacja	511
Wymiarowanie sprzętu	511
Zegar i model procesora	513
Prędkość i wielkość pamięci	516
Architektura NUMA	517
Dysk i I/O	519
Wymagania sprzętowe dla DirectQuery	520
Optymalizowanie konfiguracji sprzętowej	521
Ustawienia zasilania	521
Hyper-threading	523
Ustawienia NUMA	523
Wirtualizacja	524
Dzielenie węzłów NUMA pomiędzy różne maszyny wirtualne	524
Przydzielanie pamięci dla maszyny wirtualnej	525
Skalowalność rozwiązania tabelarycznego SSAS	526
Skalowalność dla pojedynczej bazy danych (wielki rozmiar)	526
Skalowalność dla wielkiej liczby użytkowników	527
Podsumowanie	527
<i>Indeks</i>	529
<i>O autorach</i>	556

Przedmowa

Większości osób, które już pracowały z Analysis Services, zapewne nie trzeba wyjaśniać, kim są Marco Russo i Alberto Ferrari. Pracowali oni nad niektórymi z najbardziej wymagających projektów Analysis Services, napisali wiele książek o tym produkcie i zamieszczali – razem i osobno – fascynujące wpisy na blogu o najlepszych praktykach i innych tematach technicznych. Niezależnie od tego, często występują jako prezenterzy na konferencjach branżowych i prowadzą popularne kursy szkoleniowe na szeroki zakres tematów powiązanych z analizami biznesowymi (*business intelligence*) i Analysis Services. Osobiście spotkałem Alberto i Marco wiele razy w ciągu minionych lat i podziwiam ich nieustającą pasję i prawdziwe umiłowanie uczenia się i nauczania.

Jako wieloletni członek zespołu projektowego Analysis Services, pracowałem nad wieloma aspektami silników SSAS, a także częściowo nad Power BI. Naprawdę lubiłem budowanie Analysis Services. Silny i entuzjastyczny zespół inżynierski w połączeniu ze wspianymi partnerami i klientami – to rzecz niezapomniana!

Projektując i programując różne funkcjonalności Analysis Services w tak wielu wydaniach czasami sędzę, że dokładnie wiem, czego klienci potrzebują i czego oczekują od produktu. Jednak moje rozmowy z Marco i Alberto zwykle przypominają mi, jak bardzo większa jest ich wiedza na temat BI w rzeczywistym świecie. Nasze dyskusje są zawsze fascynujące i prowokujące do myślenia, gdyż obydwaj mają skłonność do przedstawiania nieoczekiwanych punktów widzenia, które wstrząsały moimi wcześniejszymi przekonaniami. Pytania są zaskakujące i z szerokiego zakresu, dyskusje gorące, ale ich konsekwencją są zawsze pozytywne zmiany, zarówno dla samego produktu, jak i naszych klientów.

Każdy zespół projektowy bywa niekiedy oskarżany o „życie w wieżach z kości słoniowej” i ignorowanie tego, co jest ważne dla końcowych odbiorców. Dzięki reakcjom i wskazówkom naszych MVP i ekspertów, często wylewających kubły zimnej wody na nasze błędne koncepcje, dostajemy od nich lepsze pomysły, często jeszcze cenniejsze, niż oni sami się spodziewali. Jednak moim zdaniem największą wartość, jaką wnieśli oni do świata Analysis Services, jest ich działanie jako pośrednicy i tłumaczenie naszej dokumentacji i innych przekazów (które niekiedy, o ile nie zazwyczaj są zbyt techniczne lub abstrakcyjne dla nie-programistów). Szczególnie cenny jest ich talent do tworzenia przykładów i rozwiązań pokazujących, jak coś naprawdę powinno być zrobione. Tak książka jest kolejnym doskonałym przykładem.

Jak zwykle, Marco i Alberto włożyli ogromną pracę w poznanie nowego wydania Analysis Services 2016. Teraz Czytelnik może skorzystać z ich wiedzy i ciężkiej pracy i użyć wszystkich lekcji, których nauczyli się o tym nowym produkcie.

Jestem osobiście bardzo dumny z wydania Analysis Services 2016, zawierającego tak wiele nowych funkcji i usprawnień wydajnościowych, że mogę wymienić tylko niektóre z moich ulubionych: metadane tabelaryczne, model obiektowy TOM, SuperDAX, Parallel Partition Processing, BiDirectional CrossFiltering (i wiele innych). Po przejrzaniu wielu rozdziałów tej nowej książki jestem przekonany, że będzie naprawdę użytecznym i pouczającym dodatkiem do produktu, zaś czytelnicy będą szybko mogli wykorzystać cały potencjał tej nowej wersji Analysis Services.

Z niecierpliwością czekam na przyszłą współpracę z Marco i Alberto i życzę im wielkiego sukcesu przy kolejnych nowych książkach!

Akshai Mirchandani
Principal Software Engineer
Microsoft Corporation

Wprowadzenie

Pierwsze wydanie tej książki zostało opublikowane w roku 2012, gdy firma Microsoft udostępniła pierwszą wersję SQL Server Analysis Services (SSAS) pracującą w trybie tabelarycznym (Tabular). Wcześniej SSAS wykorzystywał inny mechanizm, obecnie nazywany trybem wielowymiarowym (Multidimensional). Od roku 2012 użytkownicy mają wybór, którą opcję chcą zainstalować. W roku 2016 firma Microsoft wypuściła drugie główne wydanie Analysis Services Tabular, wprowadzające wiele nowych funkcjonalności i ważne usprawnienia. Z tego powodu zdecydowaliśmy o konieczności napisania drugiego wydania naszej książki na temat SSAS Tabular – właśnie tego, które Czytelnik ma teraz w dłoniach.

Choć numer wersji Analysis Services jest obecny w tytule książki, w zasadzie nie powinno go tam być. Wynika to z faktu, że sprawy dzieją się coraz szybciej i szybciej. W chwili pisania tych słów używamy wersji 2016 usług SSAS, ale wstępna wersja techniczna kolejnej wersji jest już dostępna. Czy to znaczy, że książka ta jest już przestarzała, choć czuć jeszcze zapach farby drukarskiej? Nie. Podjęliśmy to wyzwanie i dołączyliśmy uwagi dotyczące tych funkcjonalności, które mogą się wkrótce zmienić. Są to jednak wyjątki. Zapewne zobaczymy nowe funkcje i właściwości dodane do produktu, ale raczej niewiele zmian w tych już istniejących.

Jeśli ktoś czytał już wcześniejsze wydanie tej książki, czy warto czytać to nowe? Tak. Zawiera ona wiele nowej treści i uaktualnień. Zaprawdę, należy przeczytać od nowa niemal wszystkie rozdziały, gdyż zaktualizowaliśmy całą treść zgodnie z nową wersją Analysis Services. Co więcej, w tym wydaniu zdecydowaliśmy się skupić wyłącznie na SSAS. Usunęliśmy z niej zaawansowane rozdziały na temat języka DAX, dodając w zamian nowe rozdziały i rozszerzając istniejące w celu omówienia nowych funkcji i zapewnienia pogłębionego obrazu silnika SSAS. Wykorzystaliśmy też doświadczenia zdobyte w ciągu tych lat podczas pomagania wielu klientom na całym świecie przy wdrażaniu rozwiązań bazujących na Analysis Services Tabular. Na wypadek, gdyby komuś brakowało części poświęconej DAX, napisaliśmy oddzielną książkę tylko na temat tego języka, DAX. Kompletny przewodnik, w której Czytelnik znajdzie wszystko, co potrzebne dla opanowania tego pięknego języka – znacznie więcej, niż zawarliśmy we wcześniejszym wydaniu tej książki.

Na koniec, jeśli Czytelnik jest nowym projektantem, dlaczego miałby zainwestować w naukę Analysis Services Tabular? W dzisiejszych czasach Power BI wydaje się być dobrą alternatywą dla mniejszych modeli, jest łatwiejszy do opanowania, no i jest

darmowy. Może jednak przyjść taki dzień, gdy rozwiązanie oparte na Power BI będzie musiało zostać przeskalowane, obsłużyć wielu użytkowników, pomieścić więcej informacji i urosnąć zarówno pod względem rozmiarów, jak i złożoności. Gdy to nastąpi, naturalnym ruchem będzie migracja do pełnego rozwiązania tabelarycznego. Silnik używany w Power BI i Power Pivot jest zasadniczo taki sam, jak w SSAS Tabular, zatem im więcej będziemy o nim wiedzieć, tym lepiej.

Mamy nadzieję, że książka ta okaże się użyteczna i że Czytelnik skorzysta na jej lekturze.

Kto powinien przeczytać tę książkę

Książka ta skierowana jest do profesjonalnych projektantów rozwiązań analiz biznesowych (BI): konsultantów lub członków wewnętrznych zespołów projektanckich BI, którzy przymierzają się do projektów wykorzystujących model tabelaryczny.

Zacznijmy od podstaw trybu tabelarycznego, zatem pod tym względem jest to książka dla początkujących. Zakładamy jednak, że Czytelnik zna już pewne kluczowe koncepcje BI, takie jak modelowanie wielowymiarowe i projektowanie hurtowni danych. Wiedza na temat relacyjnych baz danych, a konkretnie SQL Server, będzie przydatna, gdy przyjdzie czas na poznawanie struktury modelu tabelarycznego i jak ładować dane do tego modelu, a także dla takich tematów, jak DirectQuery. Wcześniejsze doświadczenie z trybem Analysis Services Multidimensional nie jest wymagane, ale ponieważ wiemy, że większość czytelników tej książki już jakieś posiada, od czasu do czasu odwołujemy się do jego cech i porównujemy z odpowiadającymi funkcjonalnościami w trybie tabelarycznym.

Kto nie powinien czytać tej książki

Żadna książka nie jest właściwa dla każdej możliwej publiczności i ta również nie stanowi wyjątku. Osoby nie mające doświadczenia w dziedzinie BI szybko poczną się na „zbyt głębokiej wodzie”, podobnie jak menedżerowie nie posiadający technicznego przygotowania.

Organizacja książki

Książka składa się z następujących rozdziałów:

- Rozdział 1, „Wprowadzenie do modelu tabelarycznego”, przedstawia podstawy modelu tabelarycznego – czym jest i kiedy powinien (a kiedy nie powinien) być używany.
- Rozdział 2, „Rozpoczynanie pracy w modelu tabelarycznym” oraz 3, „Ładowanie danych w modelu tabelarycznym” omawiają podstawy konstruowania modelu tabelarycznego.
- Rozdział 4, „Wprowadzenie do DAX”, przedstawia podstawy języka DAX, pozwalające na tworzenie prostych kalkulacji.
- Rozdziały 5, „Budowanie hierarchii”, 6, „Modelowanie danych w trybie tabelarycznym”, 8, „Warstwa prezentacji modelu tabelarycznego”, 10, „Bezpieczeństwo” oraz 15, „Optymalizowanie modeli tabelarycznych” poświęcone są licznym zagadnieniom projektu tabelarycznego, takim jak hierarchie, relacje, narzędzia klienckie i zabezpieczenia.
- Rozdział 7, „Tabular Model Scripting Language (TMSL)” zawiera omówienie języka TMSL, zaś 13, „Programowy interfejs modelu tabelarycznego” udostępnia szczegóły dodatkowych metod programistycznego dostępu do modeli tabelarycznych.
- Rozdziały 9, „Korzystanie z DirectQuery” oraz 12, „Wewnątrz VertiPac” udostępniają wiele szczegółów na temat silników magazynowych DirectQuery i VertiPac.
- Rozdziały 11, „Przetwarzanie i partycjonowanie modelu tabelarycznego”, 14, „Monitorowanie i dostrajanie usługi tabelarycznej” oraz 16, „Wybieranie sprzętu i wirtualizacji” poświęcone są zagadnieniom operacyjnym, takim jak wymiarowanie sprzętu i konfiguracji, partycjonowaniu tabel i monitorowania działania modelu.

Konwencje używane w tej książce

Książka przedstawia informacje przy użyciu pewnych konwencji mających na celu poprawę czytelności informacji i łatwości podążania za przekazem:

- Opatrzone ikonami elementy w ramkach, takie jak „Uwaga”, zawierają dodatkowe informacje lub alternatywne metody realizacji opisywanego zadania.
- Tekst do wpisania przez Czytelnika (poza blokami kodu) lub nazwy klikanych poleceń są wytłuszczone.
- Znak plus (+) pomiędzy nazwami dwóch klawiszy oznacza, że klawisze te muszą zostać wciśnięte równocześnie. Dla przykładu „Naciśnij Alt+Tab” oznacza, że należy przytrzymać klawisz Alt przy wciskaniu klawisza Tab.

- Definicje miar, kolumn obliczanych i tabel obliczanych używają poniższego standardu, który nie odpowiada ściśle składni używanej w Visual Studio, ale jest wygodniejszą konwencją w przypadku książki (więcej szczegółów na ten temat zawiera rozdział 4):

```
Tabela[NazwaMiary] := <wyrażenie>
Tabela[NazwaKolumnyObliczanej] = <wyrażenie>
ObliczanaTabela = < wyrażenie>
```

- Nazwy opcji menu, okien dialogowych i poleceń podawane są w wersji anglojęzycznej. Przy pierwszym użyciu dodawana jest polska wersja, o ile istnieje. W przypadku braku danego elementu w wersji polskojęzycznej podane zostało tłumaczenie danego terminu, pisane małą literą.

Wymagania systemowe

Do zainstalowania przykładów kodu i bazy danych używanych w tej książce potrzebny będzie sprzęt i oprogramowanie spełniając poniższe wymagania minimalne:

- System operacyjny Windows 7, Windows Server 2008 SP2 lub późniejsze. Akceptowane są zarówno wydania 32-, jak i 64-bitowe.
- Co najmniej GB wolnego miejsca na dysku.
- Co najmniej 4 GB pamięci RAM.
- Procesor 2.0GHz x86 lub x64 albo lepszy.
- Instancja SQL Server Analysis Services 2016 Tabular z zainstalowanymi komponentami klienckimi.

Pełne instrukcje instalowania tego ostatniego wymagania zawiera rozdział 2, „Rozpoczynanie pracy w modelu tabelarycznym”.

Przykłady kodu

Bazy danych użyte w przykładach przedstawionych w książce oparte są na udostępnianych przez firmę Microsoft przykładowych bazach danych Adventure Works 2012 DW oraz ContosoDW. Wszystkie przykładowe projekty i bazy danych można pobrać z poniższej strony:

<https://aka.ms/tabular/downloads>

Wykonaj poniższe kroki, aby zainstalować przykłady kodu na swoim komputerze, tak by móc wykonywać ćwiczenia w trakcie lektury:

1. Rozpakuj pobrane pliki przykładów na dysku twardym.
2. Odtwórz dwie bazy danych SQL Server z plików .bak znajdujących się w podkatalogu Databases. Szczegółowe instrukcje można znaleźć pod adresem <http://msdn.microsoft.com/en-us/library/ms177429.aspx>.
3. Każdy rozdział ma swój własny podkatalog zawierający przykłady kodu w katalogu Models. W wielu przypadkach przyjmują one formę projektu, który należy otworzyć w narzędziu SQL Server Data Tools (SSDT). Pełna instrukcja instalowania SSDT zawarta jest w rozdziale 2.
4. Skrypty PowerShell oraz TMSL zostały umieszczone w podkatalogach Script PowerShell oraz Script TMSL, odpowiednio.

Podziękowania

Następującym osobom należą się podziękowania za ich pomoc i rady: Bret Grinslade, Christian Wade, Cristian Petculescu, Darren Gosbell, Jeffrey Wang, Kasper de Jonge, Marius Dumitru, Kay Unkroth oraz TK Anand.

Odrębne wyrazy wdzięczności należą się Akshai Mirchandani za wspieranie robotę, jaką wykonał odpowiadając na nasze pytania i recenzję techniczną, a zwłaszcza za przedmowę do tej książki.

Na koniec chcemy podziękować Edowi Price i Kate Shoup, którzy pracowali jako redaktorzy recenzenci. Dzięki ich pracy książka zawiera znacznie mniej błędów. Wina za te, które pozostały (mamy nadzieję, że bardzo nieliczne) w całości spoczywa na nas.

Errata i wsparcie dla książki

Autorzy dołożyli wszelkich starań, aby zapewnić dokładność książki i jej kontekstu. Wszelkie błędy, które zostały zgłoszone po publikacji książki, podano na naszej witrynie Microsoft Press:

<https://aka.ms/tabular/errata>

Każdy, kto znajdzie błąd, którego nie ma na liście, może go zgłosić na tejże stronie.

Ewentualną dodatkową pomoc można uzyskać, pisząc e-mail do Microsoft Press Book Support, pod adres mspinput@microsoft.com.

Pod powyższymi adresami nie można uzyskać wsparcia dla oprogramowania firmy Microsoft.

Czekamy na kontakt

W Microsoft Press satysfakcja Czytelników jest podstawowym priorytetem, a kontakty od klientów najcenniejszym zasobem. Czekamy na komentarze na temat tej książki pod adresem:

<https://aka.ms/tellpress>

Ankieta jest krótka i czytamy każdy komentarz i pomysł. Z góry dziękujemy za Wasz wkład!

Pozostańmy w kontakcie

Niech trwa wymiana poglądów. Jesteśmy też na Twitterze: <http://twitter.com/MicrosoftPress>

ROZDZIAŁ 1

Wprowadzenie do modelu tabelarycznego

Rozdział ten ma na celu wprowadzenie SQL Server Analysis Services (SSAS) 2016, krótki przegląd koncepcji modelu tabelarycznego (ang. *Tabular model*) oraz analiza jego związków z modelem wielowymiarowym (ang. *Multidimensional model*), całością SSAS 2016 oraz z całym zestawem narzędzi firmy Microsoft do analizy biznesowej (BI). Rozdział ten pomoże nam także w podjęciu najważniejszej zapewne decyzji w cyklu życia naszego projektu: czy należy korzystać z modelu tabelarycznego. Na koniec zamieściliśmy krótkie omówienie głównych różnic dotyczących modelu tabelarycznego pomiędzy SSAS 2016 a wersjami wcześniejszymi.

Co nowego w SSAS 2016 Nowe poziomy kompatybilności oraz nowa implementacja DirectQuery, zwiększająca wydajność i usuwająca ograniczenia występujące we wcześniejszych wersjach.



Modele semantyczne w Analysis Services

W ekosystemie firmy Microsoft, BI nie jest oddzielnym produktem, ale jest to raczej zestaw funkcji rozdzielonych pomiędzy kilka produktów. W kolejnych punktach wyjaśnimy rolę SSAS w tym ekosystemie. Model tabelaryczny jest jednym z dwóch typów modeli semantycznych, które można utworzyć w SSAS (drugim jest model wielowymiarowy).

Czym jest usługa Analysis Services i dlaczego należy jej używać?

Analysis Services to baza danych OLAP (*online analytical processing*, przetwarzanie analityczne online), rodzaj bazy danych w wysokim stopniu zoptymalizowanej do wykonywania zapytań i obliczeń powszechnie stosowanych w środowisku analizy biznesowej. Wykonuje wiele działań zbliżonych do relacyjnej bazy danych, ale różni się od niej w wielu elementach. W większości przypadków rozwiązanie biznesowe BI jest łatwiej uzyskać, korzystając z usługi Analysis Services w połączeniu z relacyjną bazą

danych, taką jak Microsoft SQL Server, niż korzystając z samego SQL Server. Trzeba jednak podkreślić, że SSAS nie zastępuje potrzeby korzystania z relacyjnej bazy danych ani odpowiednio zaprojektowanej hurtowni danych.

Usługi Analysis Services można traktować jako dodatkową warstwę metadanych lub model semantyczny, umieszczony ponad hurtownią danych w relacyjnej bazie danych. Ta dodatkowa warstwa zawiera informacje o tym, jak powinny być połączone tabele faktów i tabele wymiarów, jak agregować miary, jakie możliwości eksploracji danych w ramach hierarchii powinni mieć użytkownicy, definicje podstawowych obliczeń i tak dalej. Warstwa ta obejmuje jeden lub więcej modeli zawierających logikę biznesową naszej hurtowni danych – zaś użytkownicy końcowi wykonują zapytania w ramach tych modeli, a nie w zawartej w nich relacyjnej bazie danych. Dzięki temu, że wszystkie informacje znajdują się w jednym centralnym położeniu i są współdzielone przez użytkowników, zapytania pisane przez użytkowników stają się znacznie prostsze. W większości przypadków zapytanie musi jedynie opisać, które wiersze i kolumny są potrzebne, zaś model zastosuje odpowiednią logikę biznesową, aby zwracane wartości były sensowne. Co najważniejsze, niemożliwe staje się napisanie zapytania, które zwraca „niepoprawny” wynik z powodu błędu użytkownika końcowego, takiego jak niewłaściwe powiązanie dwóch tabel lub sumowanie kolumny, której nie można sumować. To z kolei oznacza, że narzędzia raportowania i analizy muszą wykonać znacznie mniejszą pracę i mogą dostarczyć prostszy graficzny interfejs do budowania zapytań przez użytkownika końcowego. Oznacza to także, że do tego samego modelu można dołączać różne narzędzia, nadal otrzymując spójne wyniki.

Innym sposobem myślenia o usługach Analysis Services jest traktowanie ich jako rodzaju pamięci podręcznej używanej do przyspieszenia raportowania. W większości scenariuszy, w których korzystamy z Analysis Services, ładuje się do nich kopię danych z hurtowni danych. W konsekwencji wszystkie zapytania związane z raportami i analizą są realizowane poprzez Analysis Services, a nie w ramach relacyjnej bazy danych. Choć nowoczesne relacyjne bazy danych są mocno zoptymalizowane i zawierają wiele funkcji nakierowanych konkretnie na raporty BI, Analysis Services to baza danych zaprojektowana specjalnie dla tego typu prac i może w większości przypadków dać znacznie lepszą efektywność wykonywania zapytań. Optymalizacja działania zapytania jest niezwykle ważna dla użytkowników końcowych, gdyż pozwala im na przeglądanie danych bez długiego oczekiwania na realizację raportów oraz bez przerw w łańcuchu rozumowania.

Z punktu widzenia działu IT największą korzyścią jest możliwość przeniesienia ciężaru tworzenia raportów na użytkowników końcowych. Częstym problemem projektów BI, które nie korzystają z OLAP, jest fakt, że wydział IT musi tworzyć zarówno hurtownię danych, jak i powiązane z nią raporty. Zwiększa to konieczny wysiłek i nakład czasu, często powodując frustrację pracowników biznesowych w sytuacji, gdy dział IT nie potrafi zrozumieć wymagań związanych z tworzeniem raportów i reagować na nie tak szybko, jak się tego wymaga. Gdy używana jest baza danych OLAP, taka

jak Analysis Services, dział IT może udostępniać użytkownikom końcowym używane przez nią modele, pozwalając im na samodzielne budowanie raportów za pomocą takiego narzędzia, jakie im najbardziej odpowiada. Najpopularniejszym narzędziem klienckim jest Microsoft Excel. Już od czasu Office 2000 tabele przestawne Excela używały możliwość bezpośredniego połączenia z modelami wielowymiarowymi Analysis Services (nazywanych kostkami – *cubes*), a Excel 2016 ma niezwykle silne możliwości jako klient Analysis Services.

Podsumowując, Analysis Services nie tylko ogranicza nakłady pracy działu IT, lecz także zwiększa satysfakcję użytkownika końcowego, gdyż użytkownicy mogą sami budować raporty zgodnie ze swoimi chęciami i przeglądać dane we własnym tempie bez pośredników.

Krótką historia Analysis Services

Usługi SQL Server Analysis Services – inaczej usługi OLAP, jak były nazywane przy udostępnieniu ich po raz pierwszy w roku 1998 wraz z wydaniem SQL Server 7.0 – stanowiły pierwsze wejście firmy Microsoft na rynek BI. Po ich wprowadzeniu wiele osób uznało, że pokazały one, iż oprogramowanie BI może wyjść z niszy i wejść na masowy rynek. Sukces Analysis Services i pozostałych produktów BI firmy Microsoft, jaki miał miejsce w ciągu ostatnich 16 lat, potwierdził te przewidywania.

SQL Server Analysis Services 2000 stanowiły pierwszą wersję Analysis Services, która zyskała duże znaczenie na rynku. Analysis Services 2005 szybko stały się najlepiej sprzedającym się narzędziem OLAP, gdy zaś Analysis Services 2008 i 2008 R2 jeszcze poprawiły skalowalność i wydajność, coraz więcej firm zaczęło je przyjmować jako podstawę swojej strategii BI. W roku 2010 terabajtowe kostki stały się dość częste. Znanym przykładem jest tu kostka Yahoo! o rozmiarach 24 TB, pokazująca, ile można osiągnąć.

Microsoft Analysis Services 2012 wykorzystywały istniejącą infrastrukturę w celu wprowadzenia nowego silnika i nowego typu modelu danych, w istocie stając się produktem typu „dwa w jednym”. Nadal zawierały wersję Analysis Services pochodzącą z wydania SQL Server 2008 R2 i wcześniejszych, ale obecnie mechanizm ten znany jest pod nazwą *modelu wielowymiarowego* (Multidimensional). Choć w Analysis Services pojawiły się pewne usprawnienia od czasu wersji 2008 R2, dotyczące głównie wydajności, skalowalności i narzędzi zarządzania, nie pojawiła się od tego czasu żadna nowa główna funkcjonalność. Jednocześnie jednak Analysis Services 2012 zawiera nowy mechanizm modelowania danych i nowy silnik, który w dużym stopniu odpowiada sposobom działania i modelowania danych w Power Pivot i Power BI. Mechanizm ten otrzymał nazwę *modelu tabelarycznego* (Tabular).

Następna wersja SQL Server nie wprowadziła nowych funkcji BI, zatem nie ma żadnych różnic pomiędzy Analysis Services 2012 a 2014, przy założeniu, że zainstalowane zostały bieżące pakiety serwisowe i aktualizacje zbiorcze. Natomiast w wersji

Analysis Services 2016 wprowadzono wiele nowych funkcji i usprawnień w modelu tabelarycznym. Zmiany są na tyle istotne, że uznaliśmy za konieczne napisanie nowej książki na ten temat.

Model tabelaryczny w Analysis Services 2016 jest podstawowym tematem tej książki. Omówimy w niej problemy migracji z modeli tabelarycznych utworzonych we wcześniejszych wersjach Analysis Services, ale jeśli ktoś nie planuje aktualizacji do wersji 2016, zalecamy raczej zapoznanie się z naszą wcześniejszą książką, *Microsoft SQL Server 2012 Analysis Services: The BISM Tabular Model**.

Model tabelaryczny i wielowymiarowy

W tej części rozdziału skrótowo wyjaśnimy architekturę Analysis Services, która od wersji SQL Server 2012 jest podzielona na dwa modele.

Podczas instalacji Analysis Services trzeba wybrać, czy instalujemy instancję, która pracuje w trybie tabelarycznym, czy w trybie wielowymiarowym. Więcej szczegółów na temat instalacji zawiera rozdział 2, „Rozpoczęcie pracy w modelu tabelarycznym”. Instancja tabelaryczna może obsługiwać tylko bazy danych zawierające modele tabelaryczne, zaś instancja wielowymiarowa może obsługiwać jedynie bazy danych zawierające modele wielowymiarowe. Choć obie części Analysis Services mają wiele wspólnego kodu, w wielu aspektach muszą być traktowane jak oddzielne produkty. Koncepcje dotyczące projektowania obu typów modeli bardzo się różnią. Nie można przekształcić tabelarycznej bazy danych w bazę wielowymiarową i odwrotnie, bez przebudowania wszystkiego do nowa. Po tym stwierdzeniu trzeba podkreślić, że z punktu widzenia użytkownika końcowego oba modele wykonują niemal te same zadania i wydają się identyczne, jeśli używane są z poziomu narzędzia klienckiego, takiego jak Excel.

W kolejnych podrozdziałach porównamy dostępne funkcje modeli tabelarycznego i wielowymiarowego oraz zdefiniujemy niektóre ważne terminy używane w dalszej części tej książki.

Model tabelaryczny

W modelu tabelarycznym obiektem najwyższego poziomu jest baza danych. Przypomina to koncepcję bazy danych w relacyjnej bazie danych SQL Server. Instancja Analysis Services może zawierać wiele baz danych, zaś każda baza danych może być traktowana jak zawarta w niej kolekcja obiektów i danych odnoszących się do jednego rozwiązania biznesowego. Jeśli przy pisaniu raportów lub analizowaniu danych okazuje się, że trzeba wykonywać zapytania względem wielu baz danych, zapewne popełniliśmy gdzieś błąd, gdyż wszystko, czego potrzebujemy, powinno być zawarte

* Wydanie polskie: *Microsoft SQL Server 2012 Analysis Services: Model tabelaryczny BISM*, APN Promise, Warszawa 2012.

w jednej bazie danych. Jeśli napotkamy sytuację, w której dane są rozproszone pomiędzy wiele tabelarycznych baz danych, należy rozważyć przekonstruowanie tych modeli analitycznych w jeden. Omawianie właściwego projektu analitycznych baz danych i leżących w tle hurtowni danych wykracza poza tematykę tej książki. Więcej informacji na ten temat można znaleźć w dokumencie „SQLBI Methodology” (<http://www.sqlbi.com/articles/sqlbi-methodology/>).

Projektowanie modeli tabelarycznych wykonujemy za pomocą narzędzi *SQL Server Data Tools* (SSDT), zaś projekt w SSDT jest odwzorowany na bazę danych w *Analysis Services*. Po utworzeniu projektu w SSDT musimy go wdrożyć w instancji *Analysis Services*. Oznacza to, że SSDT wykonuje liczne polecenia, aby utworzyć nową bazę danych w *Analysis Services* lub zmienić strukturę istniejącej bazy danych. Można również użyć narzędzia *SQL Server Management Studio* (SSMS), używanego do zarządzania już wdrożonymi bazami danych oraz do wykonywania zapytań do baz danych.

Bazy danych składają się z jednej lub większej liczby tabel danych. Ponownie *tabela* w modelu tabelarycznym jest bardzo podobna do tabeli w relacyjnych bazach danych. Jest ona zwykle ładowana z pojedynczej tabeli w relacyjnej bazie danych lub z wyników instrukcji SQL `SELECT`. Tabela ma ściśle ustaloną liczbę *kolumn*, które są definiowane w momencie projektowania. Może mieć zmienną liczbę *wierszy*, zależnie od liczby danych do niej załadowanych. Każda kolumna ma ustalony typ. Dla przykładu, kolumna może zawierać tylko wartości całkowite, tylko teksty lub tylko wartości dziesiętne. Ładowanie danych do tabeli jest określane jako *przetwarzanie* tabeli.

Podczas projektowania można też zdefiniować *relacje* między tabelami. W przeciwieństwie do SQL, nie można definiować relacji podczas realizacji zapytań; wszystkie zapytania muszą korzystać z wcześniej istniejących relacji. Jednak relacje między tabelami mogą być oznaczone jako *aktywne* lub *nieaktywne*, zaś podczas zapytania można wybrać, które relacje między tabelami będą używane. Wewnątrz zapytań i obliczeń można też symulować efekty relacji, które nie istnieją. Wszystkie relacje są typu jeden-do-jeden albo jeden-do-wielu i muszą obejmować tylko po jednej kolumnie z każdej z dwóch tabel. Każda relacja może propagować filtry w jednym lub obu kierunkach. Nie jest możliwe zaprojektowanie relacji opartych na więcej niż jednej kolumnie z jednej tabeli ani relacji rekurencyjnych, łączących tabelę z nią samą.

Model tabelaryczny wykorzystuje wyłącznie mechanizm oparty na pamięci i przechowuje na dysku tylko kopię swoich danych, tak więc żadne dane nie zostaną utracone, jeśli usługa zostanie zrestartowana. Podczas gdy model wielowymiarowy, jak większość relacyjnych mechanizmów baz danych, przechowuje swoje dane w formacie opartym na wierszach, model tabelaryczny korzysta z bazy danych opartej na kolumnach, noszącej nazwę *pamięciowego mechanizmu analitycznego* (*in-memory analytics engine*). W większości przypadków zapewnia to znaczące polepszenie wydajności zapytań. (Więcej szczegółów na temat bazy danych opartej na kolumnach można znaleźć w http://en.wikipedia.org/wiki/Column-oriented_DBMS).



UWAGA Pamięciowy mechanizm analityczny przed wydaniem Analysis Services 2012 był znany jako mechanizm *Vertipaq*. W dokumentacji, postach na blogach i w innych materiałach online pozostało wiele odwołań do nazwy *Vertipaq*. Pozostaje ona nawet wewnątrz samego produktu, w nazwach własnościowych oraz zdarzeniach aplikacji Profiler. Z tych powodów, a także dla skrócenia zapisu, używamy w tej książce terminu *VertiPaq*, gdy odnosimy się do pamięciowego mechanizmu analitycznego.

W modelu tabelarycznym zapytania i obliczenia są zdefiniowane w języku *Data Analysis eXpressions* (DAX), natywnym języku dla baz danych modelu tabelarycznego, Power Pivot i Power BI. Zapytania i obliczenia w modelu wielowymiarowym wykorzystują język *Multi Dimensional eXpressions* (MDX). Narzędzia klienckie mogą generować zapytania w DAX albo MDX do wydobywania danych z modelu semantycznego, niezależnie od tego, czy jest to model tabelaryczny, czy wielowymiarowy. Oznacza to, że model tabelaryczny jest wstecznie kompatybilny z wieloma istniejącymi narzędziami klienckimi Analysis Services, takimi jak Excel i SQL Server Reporting Services, oraz narzędziami innych producentów oprogramowania, które wykorzystują MDX do odpytywania modelu. Jednocześnie model wielowymiarowy jest kompatybilny z nowymi narzędziami klienckimi, takimi jak Power BI, które generują zapytania w języku DAX.

Do tabeli w modelu tabelarycznym można dodawać kolumny pochodne, określane jako *kolumny obliczane* (*calculated column*). Wykorzystuje się w nich wyrażenia DAX, aby zwracać wartości na podstawie danych znajdujących się w innych kolumnach tej samej lub innej tabeli w obrębie tej samej bazy danych Analysis Services. Można również tworzyć tabele pochodne, nazywane *tabelami obliczanymi* (*calculated tables*) w modelu tabelarycznym. Wykorzystują one wyrażenia tablicowe DAX do zwracania wartości na podstawie danych już obecnych w innych tabelach w tej samej bazie danych Analysis Services. Obliczane kolumny i tabele są wypełniane podczas przetwarzania bazy danych (ładowania danych). Po zakończeniu przetwarzania kolumny i tabele obliczane zachowują się dokładnie tak samo jak zwykłe kolumny i tabele.

Za pomocą wyrażeń DAX na tabelach można także definiować *miary*. Miarami nazywamy wyrażenia DAX, które zwracają pewną zagregowaną wartość na podstawie danych z jednej lub wielu kolumn. Prostym przykładem miary jest wartość sumy wszystkich wartości z kolumny zawierającej wielkości sprzedaży.

Kluczowe wskaźniki wydajności (KPI) są bardzo podobne do miar, jednak stanowią kolekcję obliczeń, dzięki której można określić stosunek miary do wartości docelowej oraz to, czy w miarę upływu czasu zbliża się ona do wyznaczonego celu.

W większości narzędzi użytkowników, takich jak Excel, do zapytań w modelu tabelarycznym stosuje się zasady podobne do tych w tabelach przestawnych: kolumny z różnych tabel można przeciągać względem osi wierszy i kolumn tabeli przestawnej, tak aby określone wartości z tych kolumn stały się poszczególnymi wierszami i kolumnami tabeli przestawnej, a miary wewnątrz tabeli pokazywały zagregowane wartości liczbowe. Końcowy efekt przypomina nieco działanie klauzuli GROUP BY w SQL, ale

definicja sposobu agregacji danych jest z góry zdefiniowana wewnątrz miar i nie musi być podawana w samym zapytaniu.

Aby poprawić wrażenia użytkownika, można także zdefiniować dla tabel *hierarchie* wewnątrz modelu tabelarycznego. Tworzą one wielopoziomowe, predefiniowane ścieżki drażenia danych. *Perspektywy* mogą ukrywać niektóre części złożonego modelu, co zwiększa wartości użytkowe, zaś *role zabezpieczeń* mogą być wykorzystane do odmowy dostępu do określonych wierszy danych z tabeli dla określonego użytkownika. Jednak perspektyw nie należy mylić z zabezpieczeniami; nawet jeśli obiekt jest ukryty w perspektywie, nadal można wysłać do niego zapytanie, zaś samej perspektywy nie można zabezpieczyć.

Model wielowymiarowy

Na swoim najwyższym poziomie model wielowymiarowy jest bardzo podobny do modelu tabelarycznego: dane są zorganizowane w bazy danych, zaś bazy danych są zaprojektowane w SSDT (wcześniej w BI Development Studio lub w BIDS) i zarządzane przez SQL Server Management Studio.

Różnice stają się widoczne poniżej poziomu bazy danych, gdzie koncepcje wielowymiarowe przeważają nad relacyjnymi. W modelu wielowymiarowym dane są modelowane jako seria *kostek* i *wymiarów*, a nie tabel. Na każdą kostkę składa się jedna lub więcej *grupa miar*, zaś każda grupa miar jest kostką odwzorowaną na jedną tablicę faktów w hurtowni danych. Grupa miar zawiera jedną lub więcej *miar*, zaś miary są bardzo podobne do miar w modelu tabelarycznym. Kostka ma dwa lub więcej wymiarów: jeden specjalny wymiar, *wymiar miar*, który zawiera wszystkie miary z każdej grupy miar oraz różne inne wymiary, takie jak czas, produkt, położenie geograficzne, klient i tak dalej, które mają odwzorowanie w wymiarach logicznych w modelu wymiarowym. Każdy z wymiarów niebędący miarą składa się z jednego lub więcej *atrybutów* (na przykład w wymiarze Data mogą być to takie atrybuty, jak *Dzień*, *Miesiąc* i *Rok*), zaś atrybuty te mogą być z kolei używane w jednopoziomowych hierarchiach lub tworzyć wielowymiarowe *hierarchie użytkownika*. Hierarchię można stosować przy budowaniu zapytań. Użytkownicy rozpoczynają od analizy danych na wysoce zagregowanym poziomie, jak poziom roku w wymiarze czasu, a potem mogą przechodzić na niższe poziomy, jak kwartały, miesiące i dni, aby zobaczyć trendy i interesujące anomalie.

Jak można się spodziewać, ponieważ model wielowymiarowy jest bezpośrednim następcą poprzednich wersji Analysis Services, obejmuje on bogaty i dojrzały zestaw funkcji reprezentujący owoce ponad dekady rozwoju, choć niektóre z tych funkcji używane są niezbyt często. Większość funkcjonalności dostępnych w modelu tabelarycznym istnieje także w modelu wielowymiarowym, lecz model wielowymiarowy ma także wiele funkcji, które nie zostały jeszcze zaimplementowane w modelu tabelarycznym. Szczegółowe porównanie funkcji w obu modelach znajduje się w dalszej części rozdziału.

W modelu wielowymiarowym można przechowywać dane na trzy sposoby:

- **OLAP wielowymiarowy (Multidimensional OLAP, MOLAP)** W tym modelu dane są przechowywane we własnym formacie Analysis Services opartym na pamięci dyskowej.
- **OLAP relacyjny (Relational OLAP, ROLAP)** W tym trybie Analysis Services działa jako warstwa metadanych i żadne dane nie są przechowywane w samej usłudze Analysis Services. Zamiast tego, gdy do kostki kierowane są zapytania, w źródłowej relacyjnej bazie danych są wykonywane zapytania SQL.
- **OLAP hybrydowy (Hybrid OLAP, HOLAP)** Taki sam jak ROLAP, ale zawiera nieco wstępnie zagregowanych wartości zapisanych w MOLAP.

Tryb MOLAP jest używany w przeważającej większości implementacji, choć niekiedy, gdy potrzebne jest tak zwane BI czasu rzeczywistego, używana jest wersja ROLAP. Tryb HOLAP prawie nigdy nie jest używany.

Obszarem, w którym modele wielowymiarowy i tabelaryczny różnią się zasadniczo, są obsługiwane języki zapytań i obliczeń. Podstawowym językiem modelu wielowymiarowego jest MDX i tylko w nim można definiować zapytania i obliczenia. Język MDX odniósł sukces i jest obsługiwany przez wiele narzędzi klienckich Analysis Services innych producentów. Był on także promowany jako pół otwarty standard przez grono producenckie pod nazwą XMLA Council (teraz w zasadzie nie działające), a w rezultacie został także przyjęty przez wiele innych narzędzi OLAP, które są bezpośrednimi konkurentami Analysis Services. Jednak problemem MDX jest to, co stanowi ogólnie problem wielu użytkowników z modelem wielowymiarowym: choć ma on wielkie możliwości, wielu specjalistów BI miało trudności z nauczeniem się go, gdyż używane w nim pojęcia, takie jak wymiary i hierarchie, znacznie różnią się od tych, do których przyzwyczaił nas język SQL.

Po co są dwa modele?

Dlaczego nastąpił ten podział? Istnieje wiele powodów.

- Wielowymiarowa wersja Analysis Services starzeje się. Została zaprojektowana w erze 32-bitowych serwerów z jednym lub dwoma procesorami i pamięci RAM mniejszej od 1 GB, zaś przechowywanie danych na dyskach stanowiło dla baz danych jedyną opcję. Czasy się zmieniły, a nowy sprzęt uległ radykalnej zmianie. Nowa generacja kolumnowych baz danych opartych na pamięci ustanowiła standard dla wydajności zapytań z analitycznymi zadaniami, więc Analysis Services musi się dostosować do nowej technologii. Wsteczne wbudowanie nowego pamięciowego mechanizmu analitycznego do istniejącego modelu wielowymiarowego nie jest prostym zadaniem, więc konieczne stało się wprowadzenie nowego modelu tabelarycznego, który w pełni wykorzysta mechanizm VertiPaq.

- Pomimo sukcesu wielowymiarowych Analysis Services, panowała opinia, że trudno się go nauczyć. Niektórzy specjaliści od baz danych, przyzwyczajeni do relacyjnego modelowania baz danych, mieli trudności z opanowaniem koncepcji wielowymiarowych, zaś ci, którzy je opanowali, stwierdzali, że droga nauki jest trudna. Jeśli więc Microsoft chce dotrzeć z BI do szerszej publiczności, musi uprościć proces tworzenia – stąd przejście od złożonego świata modelu wielowymiarowego do względnie prostych i znanych koncepcji modelu tabelarycznego.
- Firma Microsoft postrzega samoobsługową BI jako potencjalne źródło znacznego wzrostu, zaś Power Pivot i Power BI to jej wejście na rynek. (Samoobsługowa BI pozwala mniej technicznym użytkownikom biznesowym budować własne rozwiązania analityczne). Jest istotne, aby zachować spójność między narzędziami samoobsługowymi i korporacyjnymi. Jeśli więc usługi Analysis Services muszą zostać zmodernizowane, powinny być kompatybilne z Power Pivot i Power BI, zapewniając podobne zasady projektowania, aby modele samoobsługowe mogły być łatwo zaktualizowane do pełnowymiarowych rozwiązań korporacyjnych.
- Niektóre typy danych są modelowane w bardziej odpowiedni i prostszy sposób za pomocą podejścia tabelarycznego, a innym typom danych bardziej odpowiada podejście wielowymiarowe. Dwa oddzielne modele dają programistom wybór takiego podejścia, które lepiej pasuje do ich potrzeb w danych okolicznościach.

Czym jest model semantyczny BI?

Jednym z terminów, który wywołał wiele dyskusji o Analysis Services 2012, jest *semantyczny model BI* (BISM, Business Intelligence Semantic Model). Ten termin nie odnosi się ani do modelu wielowymiarowego, ani do tabelarycznego, lecz opisuje działanie Analysis Services w zestawie Microsoft BI: działa on jako warstwa semantyczna leżąca ponad relacyjną hurtownią danych, dodając bogatą warstwę metadanych obejmującą hierarchie, miary i obliczenia. W tym zakresie termin ten jest bardzo podobny do UDM (Unified Dimensional Model), który był używany w czasach wprowadzania SQL Server 2005. W niektórych przypadkach określenie BISM odnosiło się tylko do modelu tabelarycznego, ale nie jest to właściwe. Ponieważ ta książka jest poświęcona przede wszystkim modelowi tabelarycznemu, nie będziemy często używać tego terminu, jednak uważamy, że trzeba dokładnie rozumieć, czym on jest i jak powinien być używany.

Przyszłość Analysis Services

Fakt, że wewnątrz Analysis Services istnieją dwa modele oraz dwa języki zapytań i obliczeń oznacza, że rozpoczynając projekt musimy wybrać, którego modelu będziemy używać. Problem w tym, że wtedy możemy jeszcze za mało wiedzieć, aby rozstrzygnąć, który z nich jest odpowiedni. To zagadnienie omówimy w następnym punkcie. Oznacza to także, że każdy, kto zdecyduje się specjalizować w Analysis Services, musi nauczyć się dwóch technologii albo zdecydować się na specjalizację tylko w jednym.

Firma Microsoft jasno twierdzi, że model wielowymiarowy nie jest deprecjonowany, zaś model tabelaryczny nie ma go zastąpić. Możliwe, że w kolejnych wersjach Analysis Services pojawią się nowe funkcje przeznaczone dla modelu wielowymiarowego. Fakt, że modele tabelaryczny i wielowymiarowy mają wspólną część kodu sugeruje, że niektóre funkcje mogą być z łatwością rozwijane jednocześnie dla obu modeli. W bardzo długiej perspektywie można spodziewać się, że obydwa modele zleją się ze sobą i będą oferować niemal identyczną funkcjonalność, zatem decyzja o wyborze modelu będzie bazować na tym, czy projektant woli używać wielowymiarowej, czy relacyjnej metody modelowania danych. Dla przykładu, w aktualizacji Analysis Services 2012 (Cumulative Update 3 for Service Pack 1) zostało dodane wsparcie dla zapytań DAX w modelu wielowymiarowym, co stanowi znaczący krok w tym kierunku. W istocie to dzięki tej funkcjonalności klient Power BI może ustanowić aktywne połączenie z modelem wielowymiarowym.

Azure Analysis Services

Analysis Services są również dostępne jako usługa w Azure (rozwiązaniu chmurowym firmy Microsoft), pod nazwą Azure Analysis Services (Azure AS). Można utworzyć instancję Azure AS, która wdrażana jest w bardzo krótkim czasie, przy czym opłaty ponosi się tylko za czas, gdy instancja ta jest aktywna. Gdy wstrzymamy działanie usługi, nie ma opłat, nawet jeśli dane nadal są załadowane do instancji i gotowe do odpytania, gdy tylko zrestartujemy usługę. Według stanu z lutego 2017 tylko model tabelaryczny dostępny jest w Azure. Na potrzeby tej książki można traktować instancję Azure AS jako równoważną instalacji Analysis Services w siedzibie. Czytelnik może więc po prostu pominąć wszystkie szczegóły dotyczące instalowania usług, gdyż ich konfigurowanie odbywa się automatycznie i jest zarządzane przez infrastrukturę Azure.

Wybór odpowiedniego modelu dla naszego projektu

Może wydawać się dziwne, że w tym miejscu książki zajmujemy się tematem, czy model tabelaryczny jest odpowiedni dla naszego projektu, choć nic jeszcze nie wiemy o samym modelu. Jednak na takie pytanie trzeba sobie odpowiedzieć na podobnie wczesnym etapie naszego projektu BI. Z grubsza można stwierdzić, że dla 60-70 procent projektów każdy z modeli działa podobnie, ale dla pozostałych 30-40 procent prawidłowy wybór modelu jest bardzo ważny.

Jak napisaliśmy wcześniej, po rozpoczęciu prac w jednym modelu Analysis Services nie ma możliwości przejścia na drugi. Trzeba zaczynać wszystko od początku, zapewne marnując wiele cennego czasu. Tym samym dokonanie prawidłowej decyzji powinno nastąpić możliwie wcześnie. Podczas podejmowania decyzji trzeba wziąć pod uwagę wiele czynników. W tej części omawiamy wszystkie elementy na rozsądnym poziomie szczegółowości. Warto mieć w pamięci te czynniki podczas czytania dalszych części książki. Po jej zakończeniu będziemy wiedzieć, czy mamy korzystać z modelu tabelarycznego, czy wielowymiarowego.

Licencjonowanie

Pakiet Analysis Services 2016 jest dostępny w wersjach SQL Server Standard oraz SQL Server Enterprise. W wydaniu SQL Server Standard dostępne są obydwa modele, wielowymiarowy i tabelaryczny, choć z określonymi ograniczeniami dotyczącymi liczby obsługiwanych rdzeni, wielkości pamięci i dostępnych funkcjonalności. Oznacza to, że wiele ważnych funkcji niezbędnych dla skalowania modelu, takich jak partycjonowanie, nie jest dostępnych w tym wydaniu SQL Server. Poniżej przedstawiamy skrótowe omówienie ograniczeń wydania Standard (pełne omówienie zawiera oficjalna dokumentacja licencyjna firmy Microsoft; pamiętajmy też, że wydanie Enterprise nie ma tych ograniczeń).

- **Pamięć** Instancja w modelu wielowymiarowym może alokować do 128 gigabajtów (GB), podczas gdy instancja w modelu tabelarycznym może alokować jedynie do 16 GB. Ograniczenie to znacząco wpływa na modele tabelaryczne. Ponieważ wszystkie dane muszą znaleźć się w pamięci, skompresowana baza danych musi zużyć nie więcej niż 16 GB. Uwzględniając stopień kompresji oraz zapotrzebowanie na pamięć podczas wykonywania zapytań, ograniczenie to odpowiada z grubsza nieskompresowanej relacyjnej bazie danych o wielkości od 100 do 150 GB. (Faktyczny współczynnik kompresji zależy od wielu czynników. Można zwiększyć kompresję, używając najlepszych praktyk opisanych w rozdziale 12, „Wewnątrz VertiPaq” i 15, „Optymalizowanie modeli tabelarycznych”).

- **Rdzenie** Nie można użyć więcej niż 24 rdzeni. Biorąc pod uwagę ograniczenia wielkości bazy danych, ten limit nie powinien mieć większego wpływu na sprawność działania, a w każdym razie nie większy, niż ograniczenie wielkości pamięci.
- **Partycje** Nie można podzielić tabeli na wiele partycji bez względu na to, czy używamy modelu wielowymiarowego, czy tabelarycznego. Ma to wpływ zarówno na wstępne przetwarzanie i wydajność zapytań w modelu wielowymiarowym, ale tylko na wydajność wstępnego przetwarzania modelu tabelarycznego. Zazwyczaj partycji używa się, aby przetwarzać tylko część wielkiej tabeli – na przykład tylko bieżące (z ostatniego miesiąca) wpisy tabeli transakcji obejmującej wiele lat.
- **DirectQuery** Nie można korzystać z DirectQuery – funkcji, która transformuje zapytanie wysłane do modelu semantycznego na jedno lub więcej zapytań do działającej w tle relacyjnej bazy danych – w modelu tabelarycznym. Odpowiadającą temu funkcjonalnością w modelu wielowymiarowym jest tryb ROLAP, obsługiwany w wydaniu Standard. Ograniczenie to wpływa na modele semantyczne, które muszą eksponować zmiany danych w czasie rzeczywistym.
- **Perspektywy** Nie można używać perspektyw, bez względu na wybrany model.



UWAGA W wersjach Analysis Services 2012 i 2014 funkcje pozwalające na przesyłanie zapytań DAX do modelu wielowymiarowego były dostępne tylko w wydaniach Enterprise oraz Business Intelligence. W wydaniu Analysis Services 2016 funkcja ta jest dostępna również w wydaniu Standard. Azure Analysis Services obsługuje wszystkie funkcje wydania Enterprise.

Aktualizacja poprzednich wersji Analysis Services

Jak już wspomnieliśmy, nie ma prostej możliwości przekształcenia modelu wielowymiarowego w model tabelaryczny. Na rynku istnieją narzędzia, które obiecują takie przekształcenie za pomocą kilku kliknięć myszą, ale w praktyce działają poprawnie tylko wobec bardzo prostych modeli wielowymiarowych i nie oszczędzają nam wiele czasu. Dlatego, jeśli mamy zaawansowaną implementację modelu wielowymiarowego oraz umiejętności w zakresie jego obsługi, nie ma zapewne sensu porzucania go na rzecz modelu tabelarycznego, o ile nie mamy konkretnych problemów z modelem wielowymiarowym, które model tabelaryczny mógłby rozwiązać, takich jak wiele miar opartych na zliczaniu unikatowych wartości.

Prostota korzystania

Jeśli z kolei zaczynamy korzystać z Analysis Services 2012 lub późniejszej wersji bez żadnego doświadczenia z modelem wielowymiarowym lub OLAP, zapewne stwierdzimy, że model tabelaryczny jest znacznie prostszy do nauki niż model wielowymiarowy. Nie tylko sama koncepcja jest prostsza do zrozumienia, zwłaszcza jeśli korzystaliśmy

z relacyjnych baz danych, lecz także proces budowy projektu jest bardziej oczywisty, a do nauczenia się jest znacznie mniej funkcji. Budowa pierwszego modelu tabelarycznego jest znacznie szybsza i prostsza niż budowa pierwszego modelu wielowymiarowego. Można też stwierdzić, że łatwiej nauczyć się języka DAX niż języka MDX, zwłaszcza w zakresie podstawowych obliczeń, jednak w rzeczywistości oba te języki są tak samo mylące dla osób przyzwyczajonych do SQL.

Kompatybilność z Power Pivot

Model tabelaryczny oraz Power Pivot są niemal identyczne, jeśli chodzi o sposób projektowania; interfejsy użytkownika są w obu przypadkach praktycznie jednakowe, oba też korzystają z języka DAX. Modele Power Pivot mogą być importowane do narzędzi danych SQL Server w celu wygenerowania modelu tabelarycznego, choć procesu tego nie można przeprowadzić w kierunku przeciwnym – modelu tabelarycznego nie można przekształcić na model Power Pivot. Jeśli więc zależy nam na użytkowaniu samoobsługowej BI z wykorzystaniem Power Pivot, sens ma korzystanie z modelu tabelarycznego w naszych korporacyjnych modelach BI, gdyż projekty i kod można między nimi przenosić.

Kompatybilność z Power BI

Pomimo tego, że modele tabelaryczne AS i Power BI są identyczne i używają tego samego podstawowego silnika, Power BI używa funkcji równoważnej Power Query do importowania danych do modelu danych, a funkcja ta nie jest obsługiwana w Analysis Services 2016 w ich pierwszym wydaniu. Można oczekiwać, że funkcja ta zostanie dodana w którymś z pakietów aktualizacyjnych. Gdy tylko się pojawi, możliwe będzie zaimportowanie do Analysis Services 2016 modelu Power BI.

Cechy wydajności zapytań

Wprawdzie niebezpieczne jest tu generalizowanie na temat wydajności zapytań, ale można stwierdzić, że w modelu tabelarycznym będą w większości przypadków działać nie gorzej niż w wielowymiarowym, zaś przy niektórych określonych scenariuszach będą znacznie lepsze. Na przykład pewne miary zliczające, które są słabością modelu wielowymiarowego, działają bardzo dobrze w modelu tabelarycznym. Konkretnie dowody także wskazują na to, że zapytania względem szczegółowych raportów (na przykład zapytania zwracające wiele wierszy oraz dane na poziomie tabeli faktów) będą działać w modelu tabelarycznym znacznie lepiej, o ile napiszemy je w języku DAX, a nie w MDX. Niestety przy bardziej skomplikowanych obliczeniach i technikach modelowania, jak w relacjach wiele-do-wielu, znacznie trudniej jest stwierdzić, czy lepiej działa model wielowymiarowy, czy tabelaryczny. Tylko sprawdzenie koncepcji może nam wskazać, czy wydajność danego modelu będzie spełniać wymagania.

Cechy wydajności przetwarzania

Porównanie wydajności wstępnego przetwarzania w modelach wielowymiarowym i tabelarycznym jest trudne. Liczba wierszy surowych danych, które można przetworzyć na sekundę dla pojedynczej partycji będzie zapewne podobna dla obydwu modeli – o ile pominiemy różne, nieporównywalne operacje, które każdy model wykonuje podczas przetwarzania danych, takie jak budowanie agregacji i indeksów w modelu wielowymiarowym.

Jednak model tabelaryczny ma następujące znaczące przewagi nad modelem wielowymiarowym, jeśli chodzi o przetwarzanie:

- W modelu tabelarycznym nie ma agregacji. Oznacza to przynajmniej jedno czasochłonne zadanie, którego nie trzeba wykonywać podczas wstępnego przetwarzania.
- Przetwarzanie jednej tabeli w modelu tabelarycznym nie ma bezpośredniego wpływu na żadną z pozostałych tabel modelu, podczas gdy w modelu wielowymiarowym przetwarzanie wymiarów ma znaczące efekty. Wykonywanie pełnego przetwarzania wymiaru w tym modelu oznacza, że musimy wykonać pełne przetwarzanie wszystkich kostek, w których wymiar ten jest użyty. Nawet wykonywanie tylko aktualizacji wymiaru wymaga przetworzenia indeksów kostki w celu odbudowania agregacji.

Obydwa te czynniki mogą być bolesne w dużych wdrożeniach trybu wielowymiarowego, szczególnie w sytuacjach, gdy okno czasowe dostępne dla przetwarzania jest niewielkie. Podobnym wąskim gardłem w modelu tabelarycznym są kolumny obliczane, które zawsze są rekonstruowane dla całej tabeli, nawet jeśli odświeżamy tylko jedną partycję. Z tego względu nie należy używać kolumn obliczanych w wielkich tabelach faktów modelu danych.

Uwarunkowania sprzętowe

Modele wielowymiarowy i tabelaryczny mają także bardzo różne wymagania sprzętowe. Dyskowe magazynowanie modelu wielowymiarowego oznacza, że ważne są dyski o dużej wydajności i o bardzo dużej pojemności. Potrzebna jest także pamięć podręczna, więc korzystne jest posiadanie odpowiedniej pamięci RAM, choć nie jest to wymóg konieczny. W modelu tabelarycznym wydajność pamięci dyskowej nie jest priorytetowa, gdyż baza danych przechowywana jest w pamięci. Dlatego ważne jest posiadanie pamięci RAM wystarczającej do przechowywania bazy danych i do obsługi szczytów wykorzystania pamięci, które mają miejsce podczas realizacji zapytań lub przetwarzania danych.

Wymagania dyskowe modelu wielowymiarowego będą zapewne łatwiejsze do zaspokojenia niż zapotrzebowanie na pamięć modelu tabelarycznego. Zakup dużych pamięci dyskowych dla serwera jest stosunkowo tani i prosty dla działu IT. Wiele firm

wykorzystuje sieci magazynowe (SAN), które może nie dają znacznej wydajności, ale pozwalają łatwo zapewnić odpowiednią pojemność pamięci (lub ją powiększać). Natomiast zakup dużych ilości pamięci RAM dla serwera może być trudny. Gdy powiemy, że potrzebujemy pół terabajta pamięci RAM dla serwera, zapewne wywołamy zdumienie. Ponadto, jeśli okaże się, że potrzebujemy więcej pamięci RAM niż przewidywano na początku, zwiększenie dostępnej ilości może być kłopotliwe. Z doświadczenia wiadomo, że można zacząć pracę z pojemnością pamięci RAM, która wydaje się rozsądna, ale wraz z rozbudową tabeli faktów, dodawaniem nowych danych do modelu i coraz bardziej skomplikowanymi zapytaniami, można zacząć napotykać błędy braku pamięci. Ponadto w przypadku niektórych bardzo wielkich, obejmujących kilka terabajtów danych implementacji Analysis Services, zakup serwera z pamięcią wystarczającą na przechowywanie modelu może okazać się niemożliwy, a wtedy model wielowymiarowy będzie jedyną opcją możliwą do realizacji.

BI czasu rzeczywistego

Temat nie jest już tak popularny jak kilka lat temu, ale wymagania dotyczące czasu rzeczywistego lub warunków bliskich czasowi rzeczywistemu mogą w projektach BI stać się coraz częstsze. BI czasu rzeczywistego odnosi się zwykle do potrzeby użytkowników końcowych, aby zapytania i analiza danych były możliwe natychmiast po załadowaniu danych do hurtowni danych, bez długiego oczekiwania na załadowanie danych do Analysis Services.

Model wielowymiarowy może obsłużyć ten problem na jeden z dwóch sposobów:

- Poprzez wykorzystanie trybu MOLAP i podział danych tak, aby wszystkie nowe dane w hurtowni danych wędrowały do jednej, stosunkowo niewielkiej partycji, którą można szybko przetwarzać.
- Poprzez wykorzystanie trybu ROLAP i wyłączenie buforowania, aby model wielowymiarowy wydawał zapytania SQL przy każdym przekazaniu zapytania.

Pierwsza z tych opcji jest zwykle preferowana, choć może być trudna do implementacji, zwłaszcza gdy tabele wymiarów i tabele faktów ulegają zmianie. Aktualizacja danych w wymiarze może być powolna i wymagać przebudowy agregacji. Pamięć ROLAP w modelu wielowymiarowym może często – w przypadku dużych ilości danych – powodować niską wydajność, więc czas na realizację zapytania w trybie ROLAP może być dłuższy niż czas do ponownego przetworzenia partycji MOLAP w pierwszej z opcji.

Model tabelaryczny oferuje w zasadzie te same dwie opcje, ale mają one mniej wad niż ich odpowiedniki w modelu wielowymiarowym. Jeśli dane są przechowywane w mechanizmie pamięciowym, aktualizacja danych w jednej tabeli nie ma wpływu na dane w pozostałych tabelach, więc czasy przetwarzania mogą być szybsze, zaś implementacja znacznie prostsza. Jeśli dane mają pozostać w mechanizmie relacyjnym,

to podstawową różnicę stanowi odpowiednik trybu ROLAP, określany jako Direct-Query. Pełny opis konfiguracji trybu DirectQuery zawiera rozdział 9, „Korzystanie z DirectQuery”.

Narzędzia klienckie

W wielu przypadkach sukces lub porażka projektu BI zależy od jakości narzędzi używanych przez użytkowników końcowych do analizy dostarczanych danych. Dlatego ważna jest odpowiedź na pytanie, jakie narzędzia klienckie obsługuje każdy z modeli.

Zarówno model tabelaryczny, jak i wielowymiarowy, obsługują zapytania MDX i DAX, więc teoretycznie narzędzia klienckie Analysis Services powinny obsługiwać oba modele. Niestety w praktyce nie jest to prawdą. Choć niektóre narzędzia klienckie, takie jak Excel i Power BI, działają tak samo dobrze w obu przypadkach, może być konieczna aktualizacja niektórych narzędzi klienckich innych dostawców, zaś nadal stosowane niektóre starsze narzędzia mogą w ogóle nie działać poprawnie. W ogólności, narzędzia zaprojektowane do generowania zapytań MDX (takie jak Excel) działają lepiej z modelem wielowymiarowym, zaś te generujące zapytania DAX (jak Power BI) sprawdzą się lepiej w modelu tabelarycznym, mimo tego, że wsparcie dla obu języków zapytań gwarantuje działanie wszelkich kombinacji.

Porównanie funkcji

Kolejną sprawą do rozważenia przy wyborze modelu są funkcje dostępne w modelu wielowymiarowym, które nie mają odpowiedników w modelu tabelarycznym lub są tylko w nim częściowo implementowane. Jednak nie wszystkie funkcje są ważne przy wszystkich projektach, więc trzeba stwierdzić, że w wielu scenariuszach w modelu tabelarycznym można przybliżyć niektóre funkcje modelu wielowymiarowego za pomocą sprytnego wykorzystania DAX w obliczanych kolumnach i miarach. W każdym przypadku, jeśli nie mamy doświadczenia z modelem wielowymiarowym, nie będzie nam brakować funkcji, z których nigdy nie korzystaliśmy.

Oto lista najważniejszych funkcji, których brak w modelu tabelarycznym:

- **Zapis zwrotny (Writeback)** Możliwość zapisania przez użytkownika końcowego wartości do wielowymiarowej bazy danych. Może to być na przykład bardzo istotne w aplikacjach finansowych, gdzie użytkownik wprowadza dane budżetowe.
- **Zabezpieczenia miar w wymiarze** Funkcjonalność pozwalająca nadawać lub odbierać prawa do dostępu do pojedynczej miary.
- **Bezpieczeństwo komórek** Jest to powszechnie używana technika pozwalająca nadać lub odebrać prawo do dostępu do pojedynczych komórek. Nie ma możliwości implementacji takiej funkcjonalności w modelu tabelarycznym, ale funkcja ta używana jest rzadko w modelu wielowymiarowym.

- **Zmienne hierarchie** Często używana technika zapobiegająca korzystaniu z hierarchii dziecko/rodzic. W modelu wielowymiarowym hierarchia użytkownika może się upodobnić do hierarchii rodzic/dziecko poprzez ukrycie jej elementów w razie spełnienia pewnych warunków; na przykład, gdy element ma taką samą nazwę jak jego rodzic. Jest to określane jako zmienna hierarchia (*ragged hierarchy*). W modelu tabelarycznym nie ma niczego podobnego.
- **Wymiary pełniące rolę** Są tworzone i przetwarzane tylko raz, a potem pojawiają się wiele razy w tym samym modelu pod różnymi nazwami i w różnych relacjach z grupami miar. W modelu wielowymiarowym są to *wymiary pełniące rolę* (*role playing dimensions*). Podobna konstrukcja jest możliwa w modelu tabelarycznym, dzięki czemu można tworzyć wiele relacji między dwiema tabelami (patrz rozdział 3, „Ładowanie danych w modelu tabelarycznym”). Choć funkcjonalność ta jest bardzo użyteczna, nie wykonuje dokładnie tego samego zadania jak wymiar pełniący rolę. Jeśli w modelu tabelarycznym chcemy zobaczyć tę samą tabelę w dwóch miejscach jednocześnie, musimy ją dwukrotnie załadować, co wydłuża czas przetwarzania i utrudnia utrzymanie bazy.

Tym niemniej, prawdą jest również to, że używanie wymiarów pełniących rolę nie jest najlepszą praktyką z punktu widzenia użyteczności. Wynika to z faktu, że nazwy atrybutów i hierarchii nie mogą być przemianowywane dla różnych ról. Może to prowadzić do nieporozumień przy prezentowaniu danych, jeśli używamy kilku ról tego samego wymiaru w raporcie.
- **Przypisania w zakresach oraz działania jednoargumentowe** Są to funkcje zaawansowanych obliczeń, dostępne w MDX w modelu wielowymiarowym, ale są niemożliwe lub bardzo trudne do odtworzenia w języku DAX w modelu tabelarycznym. Te rodzaje obliczeń są często stosowane w aplikacjach finansowych, więc to oraz brak zapisu zwrotnego i obsługi prawdziwych hierarchii rodzic/dziecko oznacza, że model tabelaryczny nie jest odpowiedni dla tego typu zastosowań.

Wymienione poniżej funkcje są w modelu tabelarycznym obsługiwane tylko częściowo:

- **Obsługa hierarchii rodzic/dziecko** W modelu wielowymiarowym jest specjalnym typem hierarchii zbudowanym na podstawie tabeli wymiaru z samopołączeniem, dzięki któremu każdy wiersz tabeli reprezentuje jeden element hierarchii i ma połączenie z innym wierszem reprezentującym jego element rodzicielski w hierarchii. Hierarchie rodzic/dziecko mają w modelu wielowymiarowym wiele ograniczeń i mogą powodować problemy z wydajnością zapytań. Jednak są bardzo wygodne do modelowania hierarchii, takich jak struktury organizacyjne w przedsiębiorstwie, gdyż deweloper nie musi podczas projektowania znać maksymalnej głębokości hierarchii. Model tabelaryczny implementuje podobną funkcję za pomocą funkcji języka DAX, takich jak `PATH` (patrz rozdział 5, „Budowanie

hierarchii”), ale deweloper musi zdecydować o maksymalnej głębokości hierarchii w czasie projektowania.

- **Szczegółowe przeglądanie** (*drillthrough*) – funkcja, dzięki której użytkownik może kliknąć komórkę, aby zobaczyć wszystkie szczegółowe dane zagregowane w celu otrzymania danej wartości. Funkcja ta jest obsługiwana w obu modelach, ale w modelu wielowymiarowym można określić, które kolumny z których wymiarów i grup miar są zwracane. W modelu tabelarycznym w narzędziach danych SQL Server nie ma interfejsu pozwalającego na określenie takiego wyboru, więc funkcja szczegółowego przeglądania zwraca domyślnie wszystkie kolumny z zaangażowanej tabeli lub tabel.

Porównanie DAX i MDX

W modelu tabelarycznym wszystkie kalkulacje są definiowane przy użyciu języka DAX. Tym niemniej, można wykonywać zapytania do modelu tabelarycznego przy użyciu zarówno języka DAX, jak i MDX. W ogólności używanie DAX jako języka zapytań jest wydajniejsze, ale wsparcie dla MDX jest ważne dla zapewnienia kompatybilności z wieloma istniejącymi narzędziami klienckimi, zaprojektowanymi dla modelu wielowymiarowego Analysis Services (przypomnijmy, że wersje Analysis Services wcześniejsze od 2012 obsługiwały tylko model wielowymiarowy). W tym podrozdziale skrótowo przedstawimy podstawowe koncepcje obydwu języków, co może pomóc w wyborze języka zapytań (i narzędzi klienckich) do konsumowania danych z modelu tabelarycznego.

Język DAX

DAX jest językiem funkcyjnym, który manipuluje wyrażeniami tablicowymi i skalarnymi. Zapytanie w DAX zawsze zwraca tabelę złożoną ze zmiennej liczby wierszy (zależnie od danych) oraz ustalonej liczby typizowanych kolumn (zależnie od wyrażenia zapytania). Z tego punktu widzenia rezultat zapytania DAX jest bardzo podobny do wyniku zapytania SQL i można zauważyć, że DAX jest bardziej podobny do SQL niż do MDX, jeśli chodzi o podstawowe cechy. Większą różnicą jest to, że zapytania SQL zawsze jawnie prezentują dowolne relacje pomiędzy tabelami, podczas gdy DAX niejawnie używa relacji istniejących w modelu danych, przez co konieczne jest przeanalizowanie definicji modelu, aby zrozumieć sens zapytania.

Podobnie jak SQL, DAX nie zawiera żadnej semantyki działającej na hierarchiach. DAX pozwala manipulować tylko tabelami, wierszami, kolumnami i relacjami. Nawet jeśli model tabelaryczny może zawierać metadane definiujące hierarchie, język DAX nie potrafi wykorzystać tych informacji (choć mogą one być użyte przez narzędzia klienckie do wyświetlania danych, o ile narzędzia te posługują się wyrażeniami MDX).

Jako czysty język funkcyjny, DAX nie zawiera wyrażeń imperatywnych, ale wykorzystuje specjalne funkcje nazywane iteratorami, które wykonują określone wyrażenie dla każdego wiersza w przekazanym wyrażeniu tablicowym. Argumenty te przypominają wyrażenia lambda w innych językach funkcyjnych i mieszanych. Jednak istnieją ograniczenia dotyczące sposobów łączenia ich ze sobą, zatem nie można powiedzieć, aby odpowiadały ogólnej definicji wyrażeń lambda. Niezależnie od swojej natury funkcyjnej, DAX nie pozwala definiować nowych funkcji i nie udostępnia mechanizmów rekurencyjnych.

Język MDX

Język MDX przetwarza wyrażenia w przestrzeni wielowymiarowej. Zapytanie w MDX może zwrócić inną tabelę wielowymiarową, gdzie liczba członków na każdej z osi wyniku może zależeć od danych i nie może być ograniczony do określonej wartości. Mimo że w większości przypadków wynik wyrażenia MDX jest konstruowany przy użyciu dwóch osi (wierszy i kolumn), można łatwo zbudować zapytanie zwracające zmienną liczbę kolumn w zależności od kontekstu i użytych filtrów. Wynik zapytania MDX jest przeznaczony do analizy przez narzędzia klienckie, takie jak tabele przestawne Excela. Upewnienie się, że wynik zapytania MDX będzie pasować do wstępnie zdefiniowanej struktury o określonej liczbie (i typach) kolumn, może wymagać dodatkowego wysiłku.

Każda kolumna w tabeli modelu tabelarycznego odpowiada atrybutowi wymiaru pojedynczej kostki, zaś sama kostka odpowiada całemu modelowi tabelarycznemu. Każda hierarchia w modelu tabelarycznym odpowiada hierarchii użytkownika w MDX, zaś każda miara w modelu tabelarycznym odpowiada mierze w MDX należącej do grupy miar o tej samej nazwie, jak tabela zawierająca definicję miary. Relacje w modelu są niejawnie uważane za relacje pomiędzy grupami miar i wymiarami. Jeśli ktoś używa schematów gwiazdzystych, które są najlepszym modelem relacyjnym tak dla wielowymiarowych, jak i tabelarycznych baz danych, zauważy bardzo podobne wyniki po zaimportowaniu tych samych danych do trybu Multidimensional i Tabular. Tym niemniej, obecność hierarchii użytkownika wewnątrz tabeli umożliwia wykorzystanie funkcji MDX do nawigowania przez hierarchie, czemu nie odpowiada żadna funkcjonalność w DAX.

Zapytanie MDX może używać zakresu i lokalnych miar zdefiniowanych w MDX, a także lokalnych miar zdefiniowanych w DAX. Stwierdzenie przeciwne nie jest prawdziwe. Nie można obliczać wyrażeń MDX w zapytaniu DAX. MDX umożliwia kalkulacje rekurencyjne, choć ta technika może być limitowana, gdy zostanie użyta wobec modelu tabelarycznego, gdyż definiowanie miar MDX osadzonych w modelu danych nie jest obsługiwane.

Wybieranie języka zapytań dla modelu tabelarycznego

W ogólności, domyślnym wyborem dla zapytań do modelu tabelarycznego powinien być DAX. Zapewnia najlepszą wydajność, niezależnie od szczegółowości kalkulacji (agregacje lub poziom liści). Tym niemniej, większość istniejących narzędzi klienckich dla Analysis Services generuje zapytania w MDX, zatem ważne jest określenie, jakie ograniczenia wiążą się z używaniem MDX wobec modelu tabelarycznego.

Dowolna kalkulacja na poziomie liścia (indywidualnych danych), taka jak mnożenie wykonywane wiersz po wierszu w tabeli, będzie miała lepszą definicję (i plan zapytania) w języku DAX. Nawigacja przez hierarchię użytkownika ma prostszą składnię w MDX, podczas gdy wymaga skomplikowanej konstrukcji w DAX. Z punktu widzenia wydajności, zazwyczaj MDX nie zapewnia realnych korzyści, pomimo tego, że udostępnia pamięć podręczną drugiego poziomu (w silniku formuł), która nie jest dostępna w DAX.

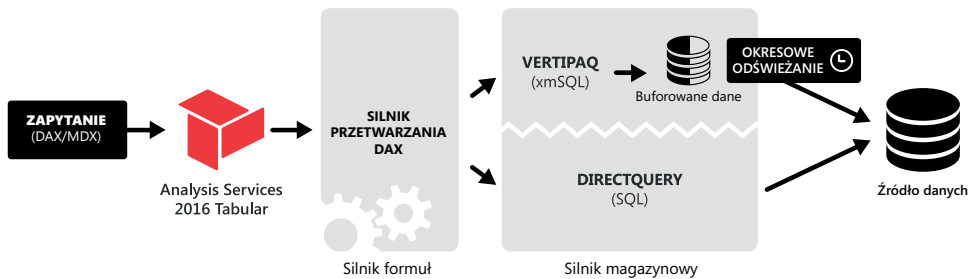
Jeśli konieczne jest wybranie języka do odpytywania danych dla raportu opartego na tabelach (takiego jak tworzone przez Reporting Services), należy użyć DAX. Język MDX może być właściwym wyborem, jeśli generujemy wyniki w postaci dynamicznej macierzy o zmiennej liczbie wierszy i kolumn z zagnieżdżonymi poziomami agregacji i sum częściowych (jak w tabeli przestawnej). Jeśli wybierzemy klienta generującego zapytania MDX, trzeba mieć świadomość możliwych problemów wydajnościowych, gdy umieścimy dużą liczbę atrybutów w wierszach i kolumnach zbioru wynikowego. Jest to typowy problem występujący przy używaniu tabel przestawnych programu Excel do odpytywania tabelarycznego lub wielowymiarowego modelu, gdy próbujemy uzyskać tabelę ze szczegółowymi informacjami z wieloma opisowymi i grupującymi atrybutami. DAX jest znacznie bardziej wydajniejszy dla tego typu zapytań.

Wprowadzenie do silników kalkulacyjnych modelu tabelarycznego

Każde zapytanie wysłane do modelu tabelarycznego w Analysis Services 2016 jest wykonywane przez dwie kolejne warstwy silników kalkulacyjnych. Analysis Services przetwarza zarówno zapytania DAX, jak i MDX, transformując je w plany zapytania wykonywane przez silnik formuł, który potrafi wykonać każdą funkcję lub operację obu języków. By odczytać surowe dane i wykonać obliczenia, silnik formuł wykonuje jedno lub więcej wywołań silnika magazynowego, którym może być albo pamięciowy mechanizm analityczny (VertiPaq), albo zewnętrzna relacyjna baza danych (DirectQuery). Każdy model danych (odpowiadający bazie danych w Analysis Services) definiuje, którego silnika magazynowego użyć. Jak widać na rysunku 1-1, silnik VertiPaq zawiera kopię danych odczytanych ze źródła podczas odświeżania modelu danych, podczas gdy DirectQuery przesyła zapytania do zewnętrznego źródła danych w razie

potrzeby, zmniejszając opóźnienie pomiędzy aktualizacją danych w źródle i dostępnością nowych danych w Analysis Services. Silnik VertiPaq otrzymuje zapytania w postaci wewnętrznej struktury binarnej (zewnętrznie opisywanej przy użyciu czytelnego dla człowieka formatu o nazwie *xmSQL*), zaś zapytania przesyłane przez DirectQuery są sformułowane w języku SQL obsługiwany przez źródło danych.

W tym podrozdziale zawarliśmy wprowadzenie do obydwu silników magazynowych, dzięki czemu będzie można dokonać wstępnego wyboru dla tworzonego modelu danych. W dalszej części książki znajdują się dedykowane rozdziały poświęcone każdemu z tych silników z praktycznymi wskazówkami ich wykorzystania i technikami optymalizacji: rozdziały 9 i 12.



RYСУNEK 1-1 Silniki formuł i magazynu w modelu tabelarycznym Analysis Services

Wprowadzenie do VertiPaq

Pamięciowy mechanizm analityczny używany w modelu tabelarycznym, znany również pod nazwą silnika VertiPaq, jest kolumnową bazą danych przechowywaną w pamięci (*in-memory*). Umieszczenie w pamięci oznacza, że wszystkie dane wchodzące w skład modelu rezydują w RAM. Bycie kolumnową oznacza, że dane są uporządkowane w strukturze oddzielnych kolumn, zoptymalizowane pod kątem pionowego skanowania i wymagające większego wysiłku, jeśli konieczne jest zmaterializowanie całego wiersza ze wszystkimi jego kolumnami. VertiPaq nie zawiera dodatkowych struktur optymalizujących zapytania, takich jak indeksy w relacyjnych bazach danych. Tym samym dla dowolnego zapytania wymagane jest pełne logiczne skanowanie kolumny. Dane w pamięci są również skompresowane (przy użyciu algorytmów, które pozwalają na szybkie operacje skanowania), co redukuje czas skanowania i wymaganą wielkość pamięci.

Silnik VertiPaq jest tylko jedną z części mechanizmu wykonawczego, który zapewnia dostarczenie wyników zapytań i wyrażeń DAX i MDX. W istocie VertiPaq jest jedynie silnikiem magazynowym, który ma fizyczny dostęp do skompresowanych danych i wykonuje podstawowe agregacje, filtrowanie i złączenia pomiędzy tabelami. Realizacja bardziej złożonych kalkulacji wyrażonych w języku DAX lub MDX spoczywa

na silniku formuł, który odbiera pośrednie rezultaty od silnika magazynowego (VertiPaq lub DirectQuery) i wykonuje pozostałe kroki potrzebne w kalkulacjach. Silnik formuł często okazuje się wąskim gardłem przy powoli działających zapytaniach używających silnika magazynowego VertiPaq (w przypadku DirectQuery sytuacja może być odwrotna). Wynika to z faktu, że silnik formuł zazwyczaj przetwarza zapytanie w jednym wątku (choć obsługuje żądania od różnych użytkowników równolegle, jeśli jest taka potrzeba). Dla kontrastu, VertiPaq może i używa wielu rdzeni, jeśli baza danych jest wystarczająco duża, aby usprawiedliwić użycie wielu wątków (zazwyczaj wymagając minimum 16 milionów wierszy w tabeli, ale liczba ta zależy od rozmiaru segmentu użytego w czasie przetwarzania).

Przetwarzanie w silniku VertiPaq opiera się na kilku algorytmach kompresujących: kodowaniu skrótowym (*hash encoding*), kodowaniu wartości oraz kodowaniu RLE (*run-length encoding*). Każda wartość obecna w kolumnie jest zawsze mapowana na 32-bitową liczbę całkowitą. Mapowanie to może zostać wykonane jedną z dwóch metod: poprzez kodowanie wartości lub kodowanie skrótowe. Kodowanie wartości wykorzystuje dynamiczne przekształcenia arytmetyczne, aby przekształcić wartość rzeczywistą na całkowitą i odwrotnie. Kodowanie skrótowe wstawia nowe wartości do tablicy skrótów. 32-bitowe wartości całkowite są następnie kompresowane przed umieszczeniem w kolumnach. Przy korzystaniu z RLE silnik sortuje dane, tak by powstały ciągłe obszary wierszy o tych samych wartościach. Następnie zapisywana jest liczba wierszy o zduplikowanych wartościach, zamiast powtarzania tej samej liczby wiele razy.



UWAGA To, czy należy wybrać kodowanie wartości, czy skrótowe, zależy od wielu czynników, wyjaśnionych szczegółowo w rozdziale 12. W tym rozdziale pokażemy również, jak można poprawić kompresję, zmniejszyć zużycie pamięci i usprawnić szybkość odświeżania danych.

Wprowadzenie DirectQuery

Pamięciowy mechanizm analityczny (VertiPaq) jest domyślnym wyborem dla każdego tworzonego modelu tabelarycznego. Tym niemniej istnieje również opcja uniknięcia przechowywania kopii danych w pamięci. W tym celu można użyć alternatywnego podejścia, które konwertuje zapytanie do modelu tabelarycznego na jedno lub więcej zapytań SQL kierowanych do źródła danych bez wykorzystywania silnika VertiPaq. Opcja ta nosi nazwę *DirectQuery*. W tym podrozdziale poznamy jej przeznaczenie. Sposoby wykorzystywania tego mechanizmu w modelu tabelarycznym przedstawimy w rozdziale 9.

Główną korzyścią z korzystania DirectQuery jest zagwarantowanie, że dane zwracane przez zapytanie są zawsze aktualne. Co więcej, ponieważ Analysis Services nie przechowuje kopii bazy danych w pamięci, rozmiar bazy może być większy niż pojemność pamięci serwera. Wydajność zapewniana przez DirectQuery znacząco zależy od sprawności relacyjnej bazy danych używanej jako źródło danych i optymalizacji w niej zastosowanych. Dla przykładu, jeśli używamy Microsoft SQL Server, możemy posłużyć się indeksami magazynu kolumnowego, aby uzyskać krótsze czasy odpowiedzi. Jednak błędem byłoby zakładanie, że typowa istniejąca relacyjna baza danych może zapewnić lepszą wydajność, niż dobrze dostrojony serwer Analysis Services dla modelu tabelarycznego. Zazwyczaj powinniśmy rozważać używanie DirectQuery albo dla małych, często aktualizowanych baz danych, albo dla bardzo wielkich baz, które nie mogą być umieszczone w pamięci. Jednak w tym drugim przypadku należy oczekiwać czasu wykonywania zapytań mierzonego w sekundach (lub więcej). Znacznie obniża to wrażenia użytkowników podczas interaktywnego przeglądania danych.

DirectQuery wspierane jest przez ograniczoną liczbę relacyjnych baz danych: Microsoft SQL Server (wersja 2008 i późniejsze), Microsoft Azure SQL Database, Microsoft Azure SQL Data Warehouse, Microsoft Analytics Platform System (APS), Oracle (wersja 9i i późniejsze) oraz Teradata (V2R6 i późniejsze). Inne relacyjne bazy danych lub wersje są wspierane przy użyciu dodatkowych niestandardowych dostawców, które mogą wymagać oddzielnej instalacji. Najświeższa lista obsługiwanych baz danych i ich wersji zawiera dokumentacja firmy Microsoft dostępna pod adresem <https://msdn.microsoft.com/en-us/library/gg492165.aspx>.

DirectQuery nie wspiera wszystkich funkcjonalności modelu tabelarycznego ani języków MDX lub DAX. Z punktu widzenia projektanta modelu danych głównym ograniczeniem DirectQuery jest to, że nie wspiera obliczanych tabel. Nadal można korzystać z obliczanych kolumn, choć z pewnymi ograniczeniami (które opiszemy później).

Z punktu widzenia języka DAX istnieje wiele funkcji, które mają odmienną semantykę, gdy zostaną przekonwertowane na odpowiadające im wyrażenia SQL, zatem można zaobserwować niespójne zachowanie dla różnych platform przy używaniu analiz czasowych i funkcji statystycznych. Jeśli chodzi o MDX, istnieją liczne ograniczenia wpływające tylko na styl kodowania MDX. Na przykład nie można używać względnych nazw, wyrażań MDX zakresu sesji ani krotek z członkami z różnych poziomów w klazulach podrzędnych MDX. Jednak istnieje jedno ograniczenie, które ma wpływ na projekt modelu danych: nie można odwoływać się do hierarchii zdefiniowanych przez użytkownika w zapytaniach MDX wysyłanych do modelu poprzez DirectQuery. To znacznie ogranicza użyteczność DirectQuery z programu Excel, gdyż taka funkcjonalność działa bez żadnych zakłóceń, gdy używamy VertiPac jako silnika magazynowego.



UWAGA DirectQuery ma dużo więcej ograniczeń w wydaniach Analysis Services 2012/2014. Dla przykładu, działa tylko z Microsoft SQL Server, język MDX zupełnie nie jest obsługiwany, a każde zapytanie DAX jest konwertowane na pełne zapytanie SQL, bez korzystania z silnika formuł modelu tabelarycznego Analysis Services. Ponadto nie są obsługiwane takie funkcjonalności, jak kolumny obliczane i funkcje analizy czasowej. Z tych powodów lista zastrzeżeń dla korzystania z DirectQuery była znacznie dłuższa. Jednak DirectQuery został całkowicie przekonstruowany w Analysis Services 2016. W tej książce zawarliśmy tylko informacje o DirectQuery używanym w Analysis Services na nowym poziomie zgodności, który stanowi treść następnego podrozdziału.

Poziomy zgodności modelu tabelarycznego (1200 kontra 110x)

Analysis Services 2016 wprowadzają nowy poziom zgodności, odnoszący się do zbioru funkcjonalności i zachowań specyficznych dla poszczególnych wydań silnika Analysis Services. Bazy danych utworzone w poprzednich wersjach Analysis Services mają inne poziomy zgodności i Analysis Services 2016 wspierają je wszystkie. Oznacza to, że można łatwo przemieścić bazy danych utworzone we wcześniejszych wersjach do Analysis Services 2016, ale musimy to zrobić tylko wtedy, jeśli chcemy użyć określonych funkcji dostępnych tylko w tym nowym wydaniu. Możliwe jest również tworzenie nowych baz danych przy użyciu starszych poziomów zgodności, jeśli potrzebujemy użyć określonych funkcjonalności dostępnych na wcześniejszym poziomie (nawet jeśli oficjalnie nie są one wspierane w nowej wersji), co pokażemy w dalszej części podrozdziału.

Dostępne są następujące poziomy zgodności modeli tabelarycznych:

- **1100** Jest to poziom dostępny w SQL Server 2012, SQL Server 2014 oraz SQL Server 2016.
- **1103** Poziom dostępny w SQL Server 2012 SP1, SQL Server 2014 i SQL Server 2016.
- **1200** Poziom dostępny w SQL Server 2016 oraz Azure Analysis Services.

Bazę danych na dowolnym poziomie zgodności można utworzyć i wdrożyć przy użyciu SQL Server Data Tools (SSDT) dla Visual Studio (VS) 2015. W przypadku korzystania z SQL Server Data Tools for Business Intelligence (SSDT-BI) dla VS 2012 lub 2013 można użyć tylko poziomów zgodności 1100 lub 1103.

Nowy poziom zgodności 1200 zapisuje model danych przy użyciu formatu JSON, podczas gdy wcześniejsze poziomy 110x używały do tego celu formatu wywodzącego się z XML, odziedziczonych po wersji Multidimensional Analysis Services. Wariant ten został zaadaptowany dla modelu tabelarycznego w pierwszej wersji (Analysis

Services 2012), ale wymagał niepotrzebnego nadkładu i nadmiarowości kodu. Miał również liczne efekty uboczne wpływające na wydajność (dłuższy czas wdrażania modelu danych) i zarządzanie (małe zmiany w modelu powodowały bardzo liczne różnice w plikach, komplikując porównywanie różnych wersji i korzystanie z systemów kontroli źródeł).

Wiele nowych funkcjonalności w Analysis Services 2016 dostępnych jest tylko przy używaniu nowego poziomu zgodności. Tym samym w ogólności należy używać poziomu 1200 przy tworzeniu nowego projektu lub migrowaniu istniejącego. Ten poziom zgodności jest również wymagany, jeśli zamierzamy wdrożyć bazę danych w Azure Analysis Services. Tym niemniej, niektóre narzędzia i rozszerzenia, które były dostępne w SSDT-BI dla VS 2012 i VS 2013, nie są już dostępne w SSDT for VS 2015. Jeśli używane było którekolwiek z nich, należy starannie rozważyć, czy wykonywać aktualizację modelu na nowy poziom zgodności i do nowej wersji Visual Studio.

Zmiana ta najbardziej wpłynęła na dwa narzędzia open-source:

- **BIDS Helper** Nie można więcej używać narzędzia BIDS Helper (<http://bidshelper.codeplex.com/>) do edytowania właściwości *Actions* lub *HideMemberIf*, gdyż te funkcje nie są już wspierane na poziomie zgodności 1200. Jeśli korzystano z BIDS Helper do edytowania folderów wyświetlanych lub translacji, należy wykorzystać nowe, równoważne funkcjonalności dla modelu tabelarycznego, obecnie oferowane w Analysis Services 2016.
- **DAX Editor** Jeśli używany jest DAX Editor (<http://www.sqlbi.com/tools/dax-editor/>) do edytowania listy miar w modelu tabelarycznym w pojedynczym pliku tekstowym, należy zadbać o pobranie najnowszej wersji (przynajmniej 2.0 lub nowszej), która jest kompatybilna zarówno z formatem XML, jak i JSON.

Nadal można używać BIDS Helper przy wdrażaniu modelu w Analysis Services 2016, wykorzystując nowy silnik i funkcje DAX dostępne w tej wersji. Jednak konieczne jest posługiwanie się narzędziem SSDT-BI for VS 2012 lub VS 2013, bez dostępu do funkcjonalności dostępnych w nowym modelu danych. Jeśli zdecydujemy się na ten wybór, należy traktować go jako sytuację tymczasową i przygotować plan przyszłej migracji na nowy poziom zgodności.

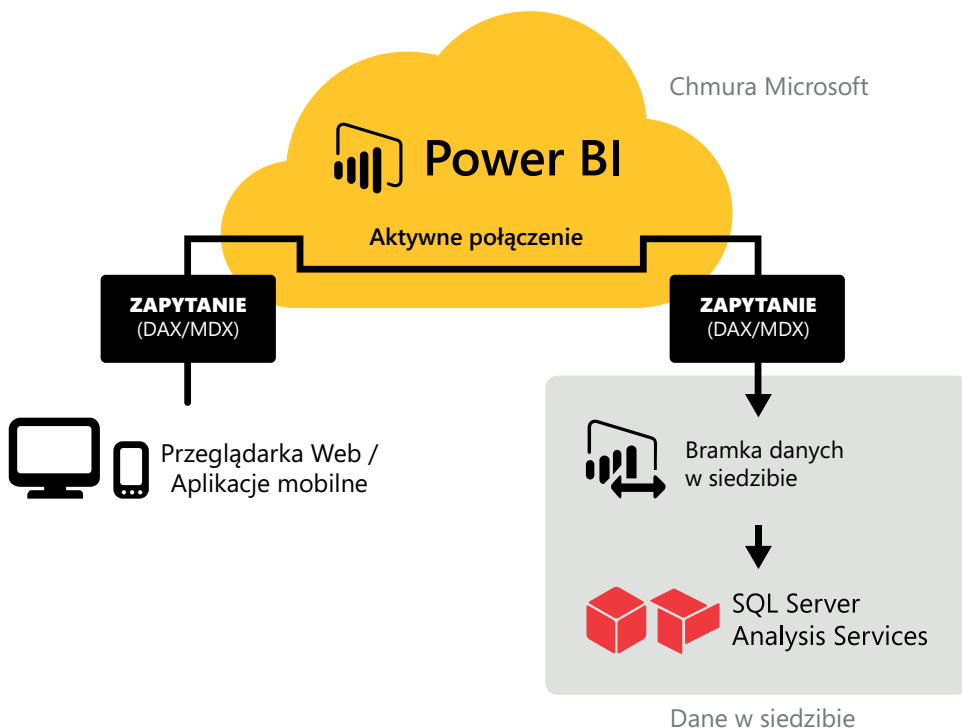
UWAGA Ta książka omawia funkcjonalności dostępne na nowym poziomie zgodności (1200). Jeśli ktoś chce tworzyć model przy użyciu poziomu zgodności 110x, może napotkać liczne różnice w funkcjach, wydajności i interfejsie użytkownika narzędzi projektowych. W takim przypadku sugerujemy, aby polegać na dokumentacji dostępnej dla Analysis Services 2012/2014 i sięgnąć po naszą poprzednią książkę, *Microsoft SQL Server 2012 Analysis Services: Model tabelaryczny BISM*.



Analysis Services i Power BI

Power BI jest wspaniałym narzędziem klienckim dla Analysis Services. Jak dotąd, jest jedynym szeroko dostępnym klientem, który generuje zapytania w języku DAX, dzięki czemu można w pełni wykorzystać model tabelaryczny. W istocie silnik używany wewnętrznie przez Power BI jest tym samym, którego Analysis Services 2016 używa dla modeli tabelarycznych. Power BI jest dostępny online, umożliwiając tworzenie raportów poprzez przeglądarkę Web lub urządzenia mobilne. Jest również dostępny jako bezpłatna aplikacja pulpitu o nazwie Power BI Desktop.

Narzędzie Power BI Desktop można połączyć z Analysis Services przy użyciu trybu Connect Live w module Get Data. Aby móc użyć tej samej bazy danych w trybie online, konieczne jest zainstalowanie On-Premises Data Gateway (bramy danych w siedzibie), tworzącej most pomiędzy klientem online a naszą bazą danych Analysis Services zainstalowaną w siedzibie firmy, jak na rysunku 1-2.



RYСУNEK 1-2 SQL Server Analysis Services połączone z Power BI poprzez On-Premises Data Gateway

Choć można połączyć dowolną wersję modelu tabelarycznego Analysis Services z Power BI, należy mieć na uwadze, że Power BI tworzy optymalizowane zapytania

DAX dla Analysis Services 2016, używając wielu nowych funkcjonalności DAX zaprojektowanych specjalnie w celu poprawienia wydajności Power BI. Gdy Power BI łączy się z Analysis Services 2012/2014, musi używać starszej wersji języka DAX, tworząc mniej wydajne zapytania, co prowadzi do wolniejszych reakcji na interakcję użytkownika w przypadku złożonych raportów. Ta różnica wydajności nie jest związana z poziomem zgodności. Bazy danych wdrożone ze starszym poziomem zgodności mogą w pełni korzystać z nowej składni DAX dostępnej w Analysis Services 2016. W takim przypadku warto rozważyć aktualizację serwera do wersji Analysis Services 2016 dla istniejącego modelu danych, jeśli chcemy się do niego odwoływać poprzez Power BI – również wtedy, gdy nie chcemy lub nie możemy zaktualizować samego modelu na nowy poziom zgodności.

Podsumowanie

W tym rozdziale zobaczyliśmy, czym są modele tabelaryczny i wielowymiarowy w Analysis Services, jakie są ich zalety i wady oraz kiedy należy ich używać. Kluczowym elementem do zapamiętania jest fakt, że bardzo się one różnią – w zasadzie są to oddzielne produkty. Nie należy podejmować decyzji o korzystaniu z modelu tabelarycznego w projekcie bez sprawdzenia, czy dobrze pasuje on do naszych wymagań. W następnym rozdziale postawimy pierwsze kroki przy budowie modeli tabelarycznych.

ROZDZIAŁ 2

Rozpoczynanie pracy w modelu tabelarycznym

Po wstępnym zaprezentowaniu Analysis Services 2016 i modelu tabelarycznego, w tym rozdziale pokażemy, jak rozpocząć budowanie modelu. Poznamy przebieg instalacji Analysis Services, sposoby pracy z projektami przy użyciu narzędzia SQL Server Data Tools, podstawowe moduły modelu tabelarycznego oraz techniki tworzenia, wdrażania i odpytywania bardzo prostego modelu tabelarycznego.

Co nowego w SSAS 2016 Dostępne są nowe funkcje modelu i projektowania, takie jak zintegrowana baza danych obszaru roboczego w SQL Server Data Tools oraz tabele obliczane. Dodatkowo rozdział został uzupełniony o omówienie Power BI jako narzędzia do testowania modeli oraz wprowadzenie do DAX Studio i innych bezpłatnych narzędzi deweloperskich.



Budowanie środowiska deweloperskiego

Przed rozpoczęciem pracy z modelem tabelarycznym trzeba przygotować swoje środowisko deweloperskie.

Komponenty środowiska deweloperskiego

Środowisko deweloperskie zawiera trzy komponenty logiczne:

- deweloperska stacja robocza
- serwer deweloperski
- baza danych obszaru roboczego, która może być hostowana na oddzielnym serwerze obszaru roboczego

Każdy z tych komponentów można zainstalować na oddzielnych komputerach lub na jednej maszynie. Każdy komponent pełni swoją rolę i musimy te role dobrze zrozumieć.

Deweloperska stacja robocza

Model tabelaryczny będziemy implementować na stacji roboczej. Jak wiemy, modele tabelaryczne są projektowane przy użyciu narzędzi SQL Server Data Tools (SSDT). Zasadniczo jest to Visual Studio 2015 plus liczne szablony projektów związanych z SQL Server. SSDT można pobrać i zainstalować z witryny firmy Microsoft (<https://msdn.microsoft.com/en-us/library/mt204009.aspx>). Nie jest potrzebna oddzielna licencja na Visual Studio.

Aby móc utworzyć i modyfikować model tabelaryczny, SSDT potrzebuje bazy danych obszaru roboczego (*workspace database*). Jest to tymczasowa baza danych, którą można utworzyć na tej samej deweloperskiej stacji roboczej w trybie Integrated Workspace albo we wskazanej instancji Analysis Services (więcej informacji na ten temat zawiera podrozdział „Instalowanie serwera bazy danych obszaru roboczego” w dalszej części rozdziału).

Po zakończeniu projektowania naszego modelu tabelarycznego za pomocą SSDT trzeba zbudować i wdrożyć nasz projekt. Budowa projektu przypomina kompilację kodu: w procesie budowy następuje translacja wszystkich informacji zapisanych w plikach projektu na język definicji danych nazywany *Tabular Model Scripting Language* (TMSL). Wdrożenie obejmuje wykonanie tego kodu TMSL w instancji Analysis Services w trybie tabelarycznym na serwerze deweloperskim. W rezultacie albo powstanie nowa baza danych, albo istniejąca baza zostanie zmodyfikowana.



UWAGA Wcześniejsze wersje Analysis Services używały języka XML for Analysis (XMLA) jako języka definicji danych. Analysis Services 2016 wprowadza nowy język, TMSL, bazujący na notacji JSON zamiast na XML. Tym niemniej skrypt oparty na JSON nadal jest przesyłany do Analysis Services przy użyciu protokołu XMLA (metoda **XMLA.Execute** akceptuje zarówno definicje sformułowane w TMSL, jak i w XMLA).

Serwer deweloperski

Serwer deweloperski to serwer z zainstalowaną instancją usług Analysis Services działających w trybie tabelarycznym, którego można używać do przechowywania swoich modeli podczas ich tworzenia. W roli serwera deweloperskiego można również użyć instancji Azure Analysis Services (Azure AS). Projekt wdrażamy na serwerze deweloperskim ze swojej stacji roboczej.

Serwer powinien znajdować się w tej samej domenie co stacja robocza wdrażania projektu. Po wdrożeniu projektu na serwerze, my i każda osoba przez nas upoważniona możemy widzieć swój model tabelaryczny i wysyłać do niego zapytania. Jest to szczególnie ważne dla innych członków naszego zespołu, którzy tworzą raporty lub inne fragmenty naszego rozwiązania BI.

Nasza stacja robocza i serwer, związane z tworzeniem projektu, mogą znajdować się na oddzielnych komputerach lub obie funkcje może spełniać jeden komputer. Z wielu powodów jest jednak lepiej, żeby funkcję serwera deweloperskiego spełniała oddzielna, dedykowana jednostka.

- Prawdopodobnie dedykowany serwer będzie miał znacznie lepsze parametry niż stacja robocza. Jak wkrótce zobaczymy, szczególnie ilość dostępnej pamięci jest bardzo ważna podczas tworzenia modelu tabelarycznego. Wymagania dotyczące pamięci oznaczają, że konieczne jest użycie systemu 64-bitowego. Obecnie wersja 64-bitowa jest już standardem, ale nadal można spotkać starsze stacje robocze z zainstalowaną 32-bitową wersją systemu Windows.
- Korzystanie z oddzielnego serwera ułatwi także przydzielanie innym deweloperom, testerom i użytkownikom praw dostępu do naszych modeli tabelarycznych jeszcze w trakcie pracy nad nim. Umożliwi im to uruchamianie własnych zapytań i budowanie raportów bez przeszkadzania nam. Niektóre zapytania mogą wymagać wielu zasobów, a nie chcielibyśmy, aby nasza stacja robocza zaczęła się niespodziewanie zacinać, gdy ktoś uruchomi duże zapytanie. Ponadto oczywiście nikt nie będzie w stanie uruchomić zapytania na naszej stacji roboczej, gdy ją wyłączymy i pójdziemy do domu.
- Dedykowany serwer umożliwi nam także przetwarzanie modeli podczas wykonywania innej pracy. Jak wspominaliśmy wcześniej, przetwarzanie dużego modelu wymaga wielu zasobów i może trwać kilka godzin. Próba wykonywania tego na własnej stacji roboczej może uniemożliwić jakąkolwiek inną pracę.
- Dedykowany serwer deweloperski będzie miał (zapewne) regularnie wykonywaną kopię zapasową, co zmniejszy prawdopodobieństwo utraty wyników pracy w przypadku awarii sprzętowej.

Istnieje kilka przypadków, gdy można rozważyć pracę *bez* dedykowanego serwera deweloperskiego: jeśli brakuje nam sprzętu lub gdy nie pracujemy nad oficjalnym projektem, lecz sprawdzamy model tabelaryczny lub dokonujemy instalacji w celu kształcenia się.

Serwer bazy danych obszaru roboczego

Jednym ze sposobów, jakimi model tabelaryczny ułatwia nam pracę, jest podejście WYSIWYG (*what you see is what you get* – dostajesz to, co widzisz), więc przy każdej zmianie modelu ma ona natychmiast swoje odbicie w danych, które widzimy w SSDT, bez konieczności zapisywania lub wdrażania czegokolwiek. Jest to możliwe, gdyż SSDT ma swoją własną, prywatną tabelaryczną bazę danych, określaną jako baza danych obszaru roboczego, w której może automatycznie dokonać wdrożenia przy każdej wprowadzonej zmianie. Można traktować tę bazę danych jak rodzaj bazy danych na czas trwania projektowania.

Ważne jest, aby nie pomylić bazy danych obszaru roboczego z tworzoną bazą danych. Tworzoną bazę danych można współdzielić z całym zespołem deweloperskim i można ją aktualizować raz, dwa razy dziennie. Natomiast baza obszaru roboczego nie powinna być nigdy przeszukiwana ani zmieniana inaczej niż przez używaną przez nas instancję SSDT. Choć tworzona baza danych może nie zawierać wszystkich danych, które docelowo będą używane w wersji produkcyjnej, będzie zapewne zawierać reprezentatywną próbkę dość sporych rozmiarów. Baza danych obszaru roboczego musi być bardzo często zmieniana i może zawierać tylko niewielką ilość danych. Wreszcie, jak już wiemy, istnieje wiele powodów, dla których tworzona produkcyjna baza danych powinna znajdować się na oddzielnym serwerze. Jest jednak kilka istotnych powodów, dla których serwer bazy danych obszaru roboczego powinien być umieszczony na tej samej maszynie, na której tworzymy bazę.

Baza danych obszaru roboczego dla projektu tabelarycznego może występować w jednej z dwóch możliwych konfiguracji:

- **Zintegrowany obszar roboczy** W tej konfiguracji SSDT uruchamia prywatną instancję Analysis Services (instalowaną przez proces instalacyjny SSDT), hostującą bazę danych obszaru roboczego.
- **Serwer obszaru roboczego** W tej konfiguracji baza danych obszaru roboczego jest hostowana na jawnie wskazanej instancji Analysis Services, które są usługą Windows instalowaną przy użyciu procedury instalacyjnej SQL Server.



UWAGA Wcześniejsze wersje Analysis Services wymagały jawnej instancji Analysis Services dla bazy danych obszaru roboczego. Produkt SSDT udostępnił opcję zintegrowanego obszaru roboczego w październiku 2016 roku. Gdy używamy zintegrowanego obszaru roboczego, SSDT uruchamia dodatkowy 64-bitowy proces Analysis Services, używając tych samych poświadczeń użytkownika, co użyte do uruchomienia Visual Studio.

Licencje

Wszystkie instalacje w środowisku tworzenia projektu powinny korzystać z wersji SQL Server Developer Edition. Wersja ta ma wszystkie funkcje wersji Enterprise Edition, ale jest bezpłatna! Jedyną niedogodnością takiej licencji jest to, że nie można z niej korzystać w środowisku produkcyjnym. Szczegółowe porównanie wszystkich wydań zawiera strona <https://www.microsoft.com/en-us/cloud-platform/sql-server-editions>.

Proces instalacji

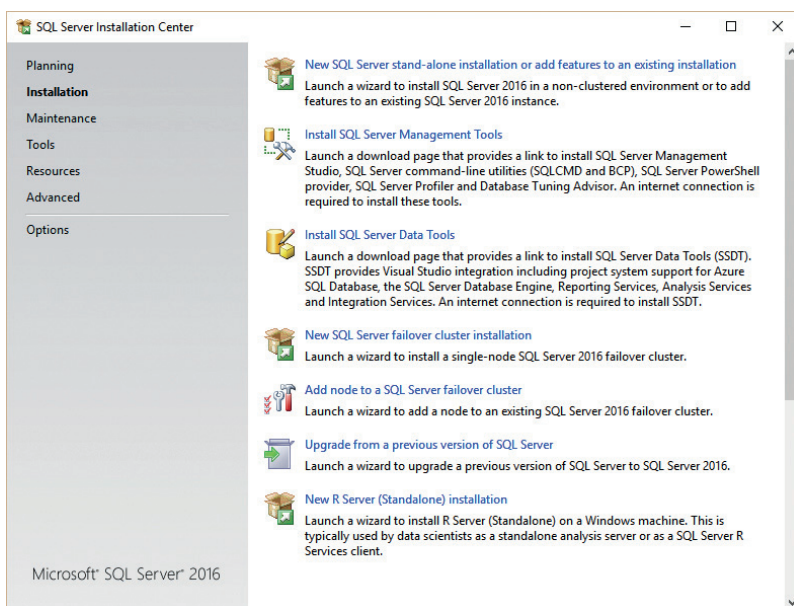
W tym podrozdziale nauczymy się, jak instalować różne komponenty środowiska deweloperskiego. Jeśli Czytelnik używa Azure AS, może pominąć kolejny punkt „Instalacja serwera deweloperskiego” i przejść bezpośrednio do punktu „Instalacja deweloperskiej stacji roboczej”. Szczegóły na temat konfigurowania instancji Azure

AS można znaleźć pod adresem <https://azure.microsoft.com/en-us/documentation/services/analysis-services/>.

Instalacja serwera deweloperskiego

Aby na swoim serwerze deweloperskim zainstalować instancję Analysis Services w trybie tabelarycznym, trzeba wykonać następujące działania:

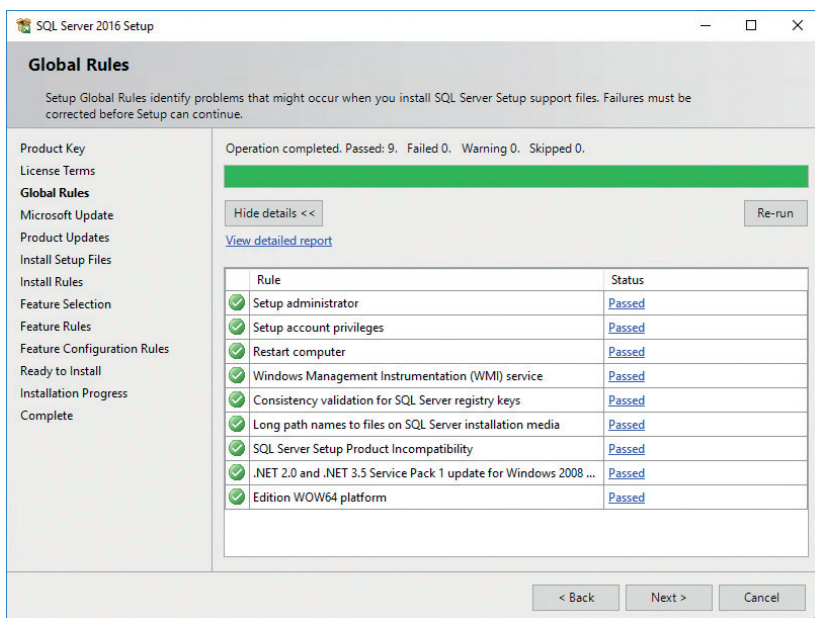
1. Upewnij się, że jesteś zalogowany jako użytkownik z uprawnieniami administracyjnymi.
2. Podwójnie kliknij SETUP.EXE, aby uruchomić SQL Server Installation Center.
3. W lewym panelu okna SQL Server Installation Center kliknij **Installation**, jak na rysunku 2-1.



RYSUNEK 2-1 Okno SQL Server Installation Center

4. Kliknij pierwszą opcję w prawej części okna, **New SQL Server Stand-Alone Installation or Add Features to an Existing Installation** (nowa samodzielna instalacja SQL Server lub dodanie funkcji do istniejącej instalacji).
5. Kreator otworzy okno SQL Server 2016 Setup. Na stronie Product Key zaznacz opcję Enter a Product Key i wpisz klucz dla swojej licencji SQL Server Developer. Alternatywnie, jeśli zamierzasz zainstalować wersję testową, zaznacz opcję Evaluation w sekcji Specify a Free Edition (wskaz wydanie bezpłatne). Następnie kliknij Next.

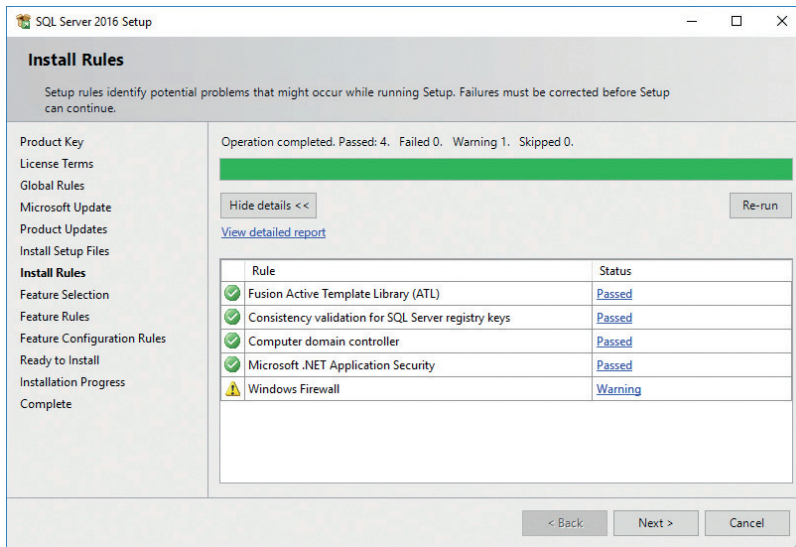
- Na stronie License Terms (warunki licencji) zaznacz pole wyboru I Accept the License Terms (akceptuję warunki licencji) i kliknij Next.
- Kreator uruchomi narzędzie Setup Global Rules w celu sprawdzenia, czy występują jakieś warunki, które mogłyby uniemożliwić poprawną instalację. Jeśli je znajdzie, zostaną wyświetlone na stronie Global Rules, pokazanej na rysunku 2-2 (strona nie jest wyświetlana, jeśli nie wystąpią żadne ostrzeżenia albo błędy). Jeśli istnieją jakieś problemy, trzeba je wyeliminować, zanim możliwe będzie kontynuowanie instalacji. Po wykonaniu tych działań kliknij Next.



RYСУNEK 2-2 Strona Global Rules

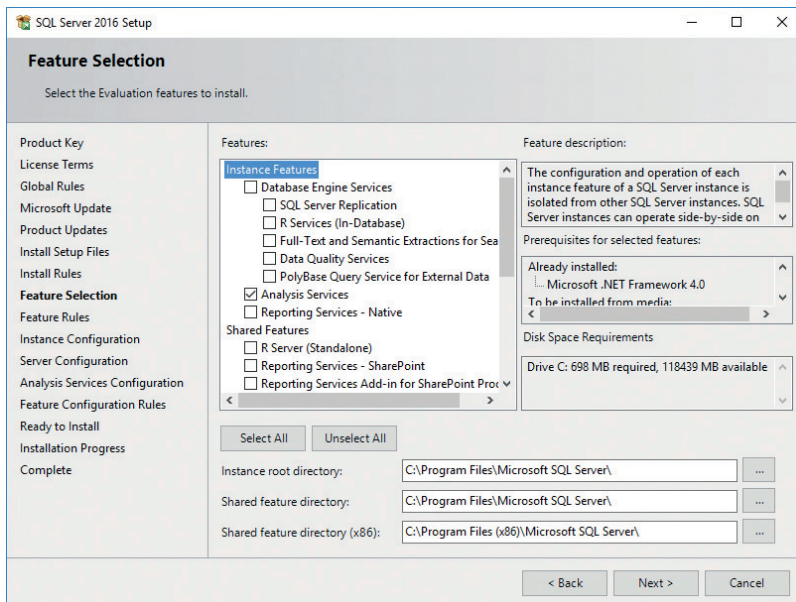
- Na stronie Microsoft Update zaznacz pole wyboru Use Microsoft Update to Check for Updates (użyj Microsoft Update, aby sprawdzić aktualizacje), o ile jesteś podłączony do Internetu, po czym kliknij Next. Kreator sprawdzi dostępność aktualizacji dla SQL Server, takich jak pakiety serwisowe, które również powinny być zainstalować.
- Strony Product Updates (aktualizacje produktu) oraz Install Setup Files (instalowanie plików) wyświetlą informacje o postępie przygotowań do instalacji. Następnie kreator uruchomi moduł Install Rules (reguły instalacji), aby sprawdzić, czy występują jakieś warunki, które mogą uniemożliwić wykonanie instalacji. Jeśli je znajdzie, wyświetli stronę Install Rules, pokazaną na rysunku 2-3 (nie zobaczysz tej strony, jeśli nie ma żadnych ostrzeżeń ani błędów). Jeśli pojawią się jakieś błędy, trzeba je wyeliminować, zanim instalacja może być kontynuowana. Ostrzeżenia

(oznaczone ikoną żółtego trójkąta) można zignorować, jeśli wydają się nieistotne. Po wykonaniu niezbędnych działań kliknij Next.



RYSUNEK 2-3 Strona Install Rules

10. Na stronie Feature Selection (wybór funkcji) zaznacz pole wyboru Analysis Services na liście Features, jak na rysunku 2-4.



RYSUNEK 2-4 Wybieranie Analysis Services na stronie Feature Selection



UWAGA Przy tych wyborach SQL Server 2016 Setup pominie stronę Feature Rules i przejdzie do strony Instance Configuration.

11. Na stronie Instance Configuration (konfiguracja instancji), pokazanej na rysunku 2-5, wybierz albo przycisk opcji **Default instance** (instancja domyślna), albo **Named instance** (instancja nazwana), aby utworzyć albo domyślną instancję (bez nazwy), albo nazwaną. Preferowany jest wybór instancji nazwanej ze zrozumiałą nazwą (na przykład TABULAR, jak na rysunku 2-5), gdyż jeśli później zdecydujemy się na zainstalowanie innej instancji Analysis Services (na przykład działającej w trybie wielowymiarowym) na tym samym serwerze, znacznie łatwiejsze będzie rozpoznanie, z którą instancją się łączymy. Następnie kliknij **Next**.

SQL Server 2016 Setup

Instance Configuration

Specify the name and instance ID for the instance of SQL Server. Instance ID becomes part of the installation path.

Product Key
License Terms
Global Rules
Microsoft Update
Product Updates
Install Setup Files
Install Rules
Feature Selection
Feature Rules
Instance Configuration
Server Configuration
Analysis Services Configuration
Feature Configuration Rules
Ready to Install
Installation Progress
Complete

☐ Default instance

☒ Named instance:

Instance ID:

Analysis Services directory: C:\Program Files\Microsoft SQL Server\MSAS13.TABULAR

Installed instances:

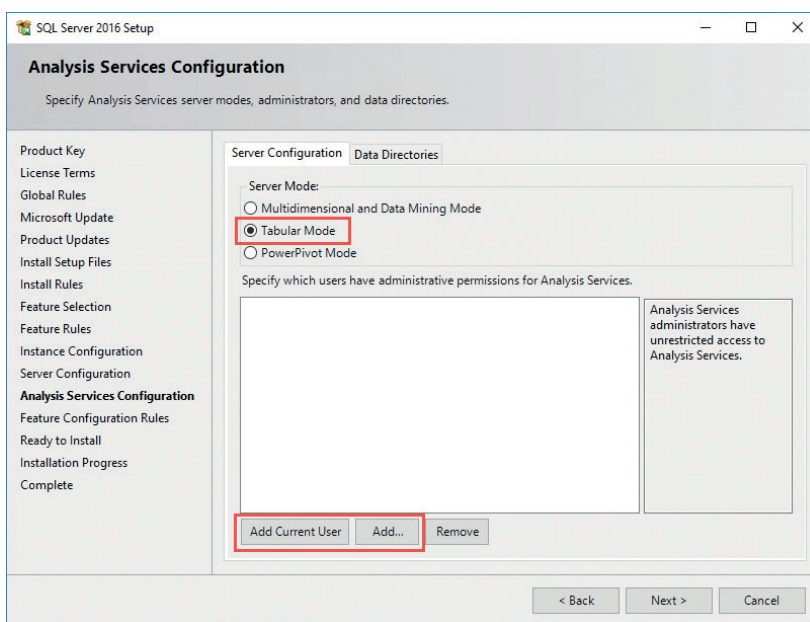
Instance Name	Instance ID	Features	Edition	Version
---------------	-------------	----------	---------	---------

< Back Next > Cancel

RYСУNEK 2-5 Wybieranie instancji na stronie Instance Configuration

12. Na stronie Server Configuration w zakładce Service Accounts (konta usługi) wpisz nazwę użytkownika i hasło konta, w którego kontekście ma działać usługa Analysis Services. Powinno być to konto domenowe utworzone specjalnie w tym celu.
13. Kliknij kartę Collation (sortowanie) i wybierz typ sortowania, którego chcesz używać. Sugerujemy, aby nie używać typu rozróżniającego wielkość liter (*case-sensitive*). W przeciwnym razie konieczne będzie pamiętanie o używaniu właściwej wielkości liter przy pisaniu zapytań i kalkulacji. Kliknij **Next**.

14. Na stronie Analysis Services Configuration w sekcji Server Mode (tryb serwera) zakładki Server Configuration zaznacz przycisk opcji Tabular Mode (tryb tabelaryczny), jak na rysunku 2-6. Następnie kliknij albo przycisk Add Current User (dodaj bieżącego użytkownika), albo przycisk Add (oba zostały zaznaczone na rysunku 2-6), aby dodać użytkownika pełniącego rolę administratora Analysis Services. W tym miejscu trzeba nominować przynajmniej jednego użytkownika.



RYSUNEK 2-6 Wybieranie opcji Tabular Mode

15. Kliknij zakładkę Data Directories (katalogi danych), aby wskazać katalogi, których Analysis Services będą używać jako katalogów Data, Log, Temp i Backup. Zalecane jest utworzenie nowych katalogów specjalnie do tych celów i umieszczenie ich na dedykowanym dysku z dużą ilością wolnego miejsca (*nie* na dysku C:). Użycie dedykowanego dysku ułatwi odszukanie tych katalogów, jeśli zechcemy sprawdzić ich zawartość i rozmiar. Następnie kliknij Next. (SQL Server 2016 Setup pominie stronę Feature Configuration Rules).
16. Na stronie Ready to Install (gotowy do instalacji) kliknij Install, aby rozpocząć proces instalacji. Po zakończeniu zamknij kreatora.

UWAGA Jest bardzo prawdopodobne, że potrzebny będzie również dostęp do instancji relacyjnego silnika bazy danych SQL Server w pracy projektowej. Można zainstalować ją na tym samym serwerze deweloperskim.



Instalacja deweloperskiej stacji roboczej

Na deweloperskiej stacji roboczej trzeba zainstalować następujące aplikacje:

- SQL Server Data Tools i SQL Server Management Studio
- System kontroli źródeł
- Inne użyteczne narzędzia deweloperskie, takie jak DAX Studio, DAX Editor, OLAP PivotTable Extensions, BISM Normalize oraz BIDS Helper.

Instalowanie SQL Server Data Tools i SQL Server Management Studio

Komponenty wymagane dla deweloperskiej stacji roboczej można zainstalować przy użyciu instalatora SQL Server, jak poniżej:

1. Powtórz kroki 1–3 z procedury „Instalowanie serwera deweloperskiego”.
2. W oknie SQL Server Installation Center (patrz rysunek 2-1) wybierz **Install SQL Server Management Tools**. Następnie postępuj zgodnie z instrukcjami, aby pobrać i zainstalować najnowsze wersje SQL Server Management Studio, SQL Server Profiler i innych narzędzi.
3. Ponownie w oknie SQL Server Installation Center zaznacz **Install SQL Server Data Tools**. Następnie postępuj zgodnie z instrukcjami, aby pobrać i zainstalować najnowszą wersję SQL Server Data Tools (SSDT) for Visual Studio 2015. Jeśli nie masz Visual Studio 2015, SSDT zainstaluje zintegrowaną powłokę Visual Studio 2015.

Instalowanie systemu kontroli kodu źródłowego

W tym miejscu trzeba dysponować jakąś kontrolą kodu źródłowego, która daje się dobrze zintegrować z Visual Studio, jak Team Foundation Server, aby sprawdzać tworzone projekty za pomocą SSDT. Tworzenie rozwiązania BI dla Analysis Services nie różni się od innych form prac deweloperskich, więc jest niezwykle ważne, aby nasz kod źródłowy zawarty w projekcie SSDT był zapisany bezpiecznie, a także by dało się wrócić do wcześniejszych wersji po dokonaniu w projekcie zmian.

Instalowanie innych narzędzi

Gdy już wdrożymy model tabelaryczny, nie można przeglądać go poprzez SSDT. Aby móc przeglądać wdrożony model, powinniśmy zainstalować Microsoft Excel na deweloperskiej stacji roboczej. Jak zobaczymy później w tym rozdziale, SSDT będzie próbował uruchomić program Excel, gdy będziemy na to gotowi. Przeglądarka zawarta w SQL Server Management Studio ma bardzo ograniczoną funkcjonalność, jako że jest oparta na kontrolce generatora zapytań MDX pochodzącej z SQL Server Reporting Services.