Microsoft

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

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

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

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

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 wspaniałymi partnerami i klientami – to rzecz niezapomniana!

Projektując i programując różne funkcjonalności Analysis Services w tak wielu wydaniach czasami sądzę, 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 przejrzeniu 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.

Zaczniemy 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 poczują 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 VertiPaq” udostępniają wiele szczegółów na temat silników magazynowych DirectQuery i VertiPaq.
- 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 wspaniałą 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 hurtownie 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 uzyskały 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.

Krótka 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 intormacji 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 niżjednej 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 *hierarchię* wewnątrz modelu tabelarycznego. Tworzą one wielopoziomowe, predefiniowane ścieżki drążenia 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 niedziałające), 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 odpytywania, 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 tabelarycznych, 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 zapytańw 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 oszczędzą 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. Konkretne 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 przetwarzania 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 kłopotliwie. 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 DirectQuery. 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 taka 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 użytkowalności. 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 poniższej 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 pozwolającego 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 wyrażeńimperatywnych, 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 gwiaździstych, 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 zapytańdo 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.

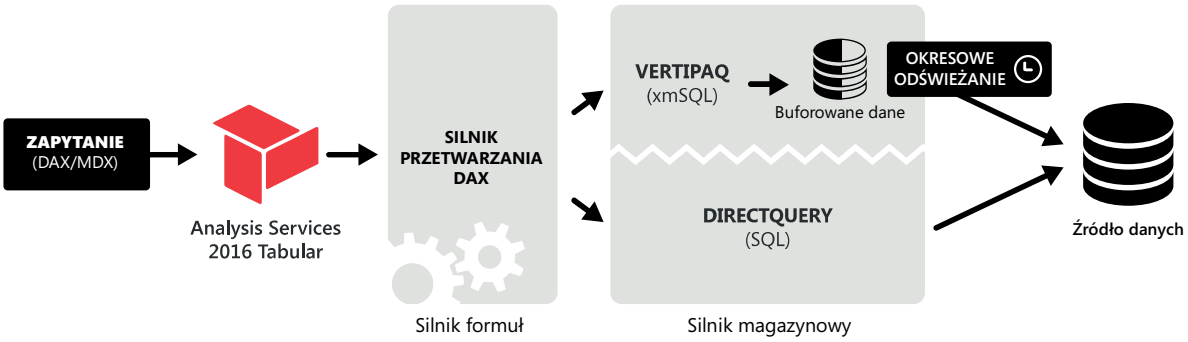
Jeśli konieczne jest wybranie języka do odpytywania danych dla raportu opartego na tabelach (takiego jak tworzone przez Reporting Services), należy użyć DAX. Język MDX może być właściwym wyborem, jeśli generujemy wyniki w postaci dynamicznej macierzy o zmiennej liczbie wierszy i kolumn z zagnieżdżonymi 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 wywołańsilnika 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 żądania do zewnętrznego źródła danych w razie

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



Rysunek 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 wyrażeńDAX 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