

Rob Collie, Bill Jelen

Power Pivot dla Excela

ZAAWANSOWANE MOŻLIWOŚCI



Helion 

Tytuł oryginału: PowerPivot Alchemy: Patterns and Techniques for Excel

Tłumaczenie: Andrzej Watrak
Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

ISBN: 978-83-283-0443-7

© 2014 Robert Collie and Tickling Keys, Inc.

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 or storage retrieval system without permission from the publisher.

Polish edition copyright © 2015 by Helion S.A.
All rights reserved.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock Images LLC.

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
<ftp://ftp.helion.pl/przyklady/ppexzm.zip>

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

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Podziękowania	13
Przedmowa	15
Wprowadzenie	17
Niedoszacowanie — główny problem z każdą książką	17
Głupie zasady są po to, aby je łamać!	18
O tym, jak Bill zmylił mnie po raz drugi	18
Wprowadzenie II	21
Rozdział 1. Sztuczki z raportami i techniki wizualizacji danych	25
Umieszczenie informacji „Data ostatniej aktualizacji”	25
Krok 1. Umieszczenie pola obliczeniowego OstatnieOdświeżenie	26
Krok 2. Użycie pola obliczeniowego w funkcji modułowej	27
Normalizowanie na wykresach pól obliczeniowych według pierwszej/średniej/maksymalnej wartości	29
Słowo z łoży szyderców na temat wykresów	30
Liczby to liczby, a obrazy to obrazy — każdy lubi po swojemu	30
Formuły	31
Alternatywne formuły wykorzystujące średnią	32
Tworzenie wykresów dynamicznie normalizowanych według pierwszej wartości	33
Wydobywanie wyjątków w „problemie Sarah”	35
Zacznijmy od pola „problem Sarah”	35
Wracamy do formuł	36
Wydobywanie rankingów i wyjątków do poziomu sum częściowych	38
W rzeczywistości nie jest to prawdziwe „wydobywanie” danych, ale praca w nadgodzinach	40

Własne etykiety ekranowe w raportach	40
Sztuczka: hiperłącza prowadzące donikąd	41
Lepszy sposób tworzenia etykietek	42
Pytania do tej sztuczki	44
Nazwane zestawy danych i „asymetryczne” tabele przestawne:	
różne pola obliczeniowe dla różnych lat	44
Powtarzalne! Przenośne!	47
Nazwane zestawy nie działają w „tradycyjnych” tabelach przestawnych	48

Rozdział 2. Fragmentatory: przepustka do interaktywności	49
„Inicjalizacja” fragmentatorów zawierających zbyt wiele pozycji do przewijania	53
Wingdings i inne czcionki symboliczne we fragmentatorach	55
Czcionki symboliczne dostępne we fragmentatorach	55
Rozszerzone znaki	57
Jak umieścić symbole we fragmentatorze	59
Utworzenie fragmentatorów	60
Zmiana czcionki fragmentatora	60
Osobne style dla różnych czcionek	63
Ukrywanie nagłówka	63
Zastosowanie makr do zmiany czcionek fragmentatorów	64
Obiekt ActiveSlicer	64
A teraz... makro	64
Jeżeli chcesz zmienić również czcionkę nagłówka... ..	65
Prosta sztuczka na „przestarzałe” fragmentatory	65
Utworzenie „zastępczej” kolumny z etykietami fragmentatora	67
A jeżeli kalendarz nie jest „przycięty”?	69
Przyjazne sortowanie w raportach z fragmentatorami	70
Utworzenie sztucznych tabel dla fragmentatorów	71
Utworzenie pól obliczeniowych	72
Umieszczenie pola UkrytaKolejnośćSortowania w tabeli i sortowanie danych	73
Ukrycie kolumny UkrytaKolejnośćSortowania w arkuszu	73
Sztuczka z sortowaniem według województw za pomocą fragmentatora	74
Sposób 1.: tekstowe pola obliczeniowe	74
Sposób 2.: dodanie nowej kolumny do tabeli Województwa	75
Dynamiczne raporty „Pierwszych N wartości” w Power Pivot	77
Utworzenie dwóch odłączonych fragmentatorów	79
Zakulisowe pola obliczeniowe: odczytywanie wyboru z fragmentatorów	79
Użycie funkcji RANKX()	80
Pole obliczeniowe UwzględniciKlienta	82
Zagwarantowanie unikatowych nazw klientów	83
Napisy w raporcie	83

Formatowanie warunkowe sterowane za pomocą fragmentatorów	85
Inna technika z użyciem odłączonych tabel	86
Użycie nowego fragmentatora i pola obliczeniowego do sterowania formatowaniem warunkowym	87
Utworzenie nowej reguły ze skalą kolorów	88
A teraz sztuczka	89
Korekta błędów	91
Określenie progów zielonego	92
Określenie progów żółtego i czerwonego	93
Użycie w formule wybranej opcji fragmentatora	94
Użycie funkcji ZESTAW.MODUŁÓW()	94
Użycie funkcji USZEREGOWANY.ELEMENT.MODUŁU()	95
Zastosowanie dodatkowych reguł formatowania	96
Obsługa wycyzszczonego wyboru	96

Rozdział 3. Najczęstsze wyzwania związane z kolumnami obliczeniowymi 99

Podręcznik do kolumn obliczeniowych dla specjalisty od Excela	100
Osobliwości funkcji TEXT() i FORMAT()	100
Gdzie jest funkcja WYSZUKAJ.PIONOWO()?	100
Nie można wskazać kolumny za pomocą klawiszy strzałek	100
Sumy pośrednie w kolumnach obliczeniowych i funkcja EARLIER()	101
Odwołania do poprzedniego wiersza i temu podobne obliczenia	104
Odwołania do wierszy „z zakresu” bieżącego wiersza	105
Krzyżowe przeszukiwanie tabel: funkcja LOOKUPVALUE() i inne techniki	107
Co się stanie, jeżeli funkcja LOOKUPVALUE() znajdzie więcej niż jedną wartość?	109
A jeżeli funkcja LOOKUPVALUE() nie spełni Twoich oczekiwań?	110
Przeszukiwanie danych na podstawie daty początkowej i końcowej	111
Przeszukiwanie tylko na podstawie daty początkowej	112
Sumowanie tabel danych w tabelach przeszukiwanych	114
Funkcja CONTAINSX: wyszukiwanie wspólnych wartości w dwóch tabelach	116
Tworzenie własnych funkcji X	117
Formuła kojarząca wartości z dwóch tabel	117
Relacje nie są potrzebne	118
Miliony odmian	118
Ponowne spojrzenie na funkcję CONTAINSX: jaka jest wspólna wartość?	119
Gdzie można użyć tej sztuczki?	121
Najczęstsze wyzwania związane z kalendarzem	121
Kolumna DataPrzyszła	122
Kolumna DzieńTygodnia	123
Kolumna RokMiesiąc	123
Kolumna Dzień roboczy/weekend	123
Kolumna RokKwartałSortowanie	124

Wyzwania związane z niestandardowym kalendarzem	124
Określenie numeru dnia w danym okresie	125
Określenie numeru tygodnia w przedziale czasu	126
Stale rosnący i „gładki” numer tygodnia	126

Rozdział 4. Modelowanie danych i formuły przenośne	127
Nowa, lepsza rzeczywistość	128
Do czego służą „formuły przenośne”?	130
Prosty przykład przenośności	130
Integracja danych o różnej „granulacji”	133
Czy nie można było dodać do tabeli Sprzedaż kolumny OkresID i na tym poprzestać?	136
Rozszerzenie reguły na inne pola niż OkresID	138
Upierz, wypłucz, powtórz	139
Nadrzędne tabele filtrujące w przypadku pojedynczej tabeli	140
Zadziwiająca i rewelacyjne rozwiązanie problemu relacji wiele-do-wielu	140
Na czym polega straszny problem z relacjami wiele-do-wielu?	141
Pozdrawiam Cię, Gerhardzie Bruecklu!	141
Rozwiązanie Gerharda problemu relacji wiele-do-wielu	142
Obliczanie w Power Pivot sprzedaży w tym samym sklepie	143
Utworzenie tabeli Sklepy	144
„Surowe” formuły	144
Obliczenie liczby transakcji	144
Filtrowanie danych sprzedaży	145
Implementacja ostatniego pola obliczeniowego	145
Ciąg dalszy tematu sprzedaży w tych samych sklepach:	
zastosowanie dat otwarcia i zamknięcia sklepów	146
Utworzenie pól obliczeniowych	147
Obliczenie daty zamknięcia sklepu	149
Test całości	149
Ale to pole ma długą nazwę!	150
Jedna zaleta sposobu „czy była wtedy sprzedaż?”	151
Analiza kampanii reklamowych: okresowo dopasowywane pola obliczeniowe	151
Tworzenie tabeli przestawnej z profilem sprzedaży	152
Obliczenie tygodniowego współczynnika korygującego	152
Import współczynników korygujących do modelu danych Power Pivot	153
Utworzenie relacji z pozostałymi tabelami w modelu danych	154
Tworzenie pola obliczeniowego z okresowym współczynnikiem korygującym	155
Pola obliczeniowe uwzględniające okresowość sprzedaży	157
Inne zastosowania pól obliczeniowych	
uwzględniających okresowość sprzedaży	158

Analiza kampanii A/B z użyciem fragmentatorów	
z datami początkową i końcową	159
Utworzenie fragmentatorów i pól obliczeniowych z datami	160
Utworzenie pola obliczeniowego z danymi sprzedaży	
przefiltrowanymi według pól z datami	162
Utworzenie pól z dzienną sprzedażą	163
Inny sposób: użycie tabeli Kampanie	164
Inne ciekawe tematy	166
Określenie odsetka powracających klientów i gości strony WWW	166
Dodanie do tabeli Klienci kolumny „Rok pierwszego zakupu”	167
Utworzenie pola obliczeniowego „Liczba klientów”	168
Utworzenie pola obliczeniowego „Aktywni klienci”	168
Utworzenie pola obliczeniowego z odsetkiem klientów	169
Uwaga: unikaj „nakładania się” lat	169
Zliczanie nowych klientów	170
Pole obliczeniowe oparte na polach pośrednich	171
Wyliczenie średniego „wieku” klienta	171
Wyszukiwanie „najlepszych rzeczy”	172
Średnia ważona: inne zastosowanie funkcji SUMX()	176
Nadanie wag odpowiednich do populacji	177
Ruchoma średnia, suma itp.	178
Suma ruchoma	180
Średnia ruchoma: pierwsze podejście	180
Średnia ruchoma: korekta	181
Inne odmiany metody	181
Średnia ruchoma sterowana za pomocą fragmentatora	181
Utworzenie selektywnego pola obliczeniowego	185
Korekta błędu	186
Tytuł wykresu	186
Zliczanie rzeczy, które nie miały miejsca	187
Kiedy ten sposób nie działa?	188
Uwaga 1.: uważaj na etykiety i wiersze!	189
Uwaga 2.: ten przykład dotyczy tylko produktów	190

Rozdział 5. Power Query

Łączenie kilku arkuszy lub skoroszytów w jedną tabelę Power Pivot	191
Usunięcie wiersza nagłówka	192
Zapisanie każdego arkusza w pliku CSV	193
Otwarcie okna wiersza poleceń	193
Przejdźcie do folderu zawierającego pliki CSV	194
Połączenie wszystkich plików w jeden plik	194
Import złączonego pliku CSV do Power Pivot	195
Bonus: makro do zapisywania wszystkich skoroszytów w folderze do pliku CSV ...	196

Złączenie plików CSV w Power Query	196
Odczytanie pierwszego pliku CSV	197
Dodanie niestandardowej kolumny z „oznaczeniem” pliku	197
Odczytanie drugiego pliku CSV	201
Odczytanie trzeciego pliku CSV	201
Czas połączyć dane!	202
Zachowanie tylko zapytania łączącego tabele	204
Testowanie odświeżania danych	206
Dlaczego jest to takie wyjątkowe?	206
Zastosowanie Power Query do anulowania przestawienia tabeli	207
Wykorzystanie wiersza nagłówka	208
Anulowanie przestawienia kolumn	209
Dlaczego jest to takie wyjątkowe	211
Tworzenie tabeli przeszukiwanej z tabeli danych za pomocą Power Query	212
Metody tworzenia tabel przeszukiwanych	212
Wyniki operacji	214
Zaawansowane zastosowanie Power Query: tworzenie tabeli kalendarza	216
Zaraz, nie widzę przycisku „Utwórz kalendarz”!	216
W skrócie	217
Jak wpisywać formuły	219
Wyniki	219
Usuwanie duplikatów	221

Rozdział 6. Power View 223

Jak korzystać z Power View	223
Wstawienie raportu Power View	224
Domyślna pierwsza tabela Power View	224
Tworzenie nowego elementu panelu	226
Ukryte kontrolki elementu	226
Tworzenie nowego elementu przez przeciągnięcie pola	227
Lista Pola programu Power View	227
Zmiana tabeli w wykres	227
Opcje formatowania	228
Hierarchia danych	229
Nowe fragmentatory jawne i niejawne	230
Dwa rodzaje filtrów	234
Kafelki	236
Zastosowanie zwielokrotnienia pionowego do tworzenia wielu wykresów	239
Tworzenie map	240
Tworzenie animowanych wykresów XY	242

Rozdział 7. Power Map	243
Kodowanie geograficzne	243
Importowanie danych zakodowanych geograficznie	244
Power Map i precyzyjne dane lokalne	246
Poruszanie się po mapach Power Map	246
Wyświetlanie mapy terenu i zdjęć satelitarnych	247
Zmiana etykiety czasu na mapie	247
Trzy metody wyświetlania czasu	248
Rysowanie linii pomiędzy dwoma punktami	248
Wskazówki dotyczące tworzenia przewodnika	250
Porada dotycząca powiększania widoku	252
Porada dotycząca pola tekstowego	252
Przejścia i efekty	252
 Skorowidz	 255

Rozdział 3

Najczęstsze wyzwania związane z kolumnami obliczeniowymi

Od lat powtarzamy, że kolumny obliczeniowe nie stanowią o przewadze dodatku Power Pivot nad zwykłymi funkcjami Excela. Z pewnością jednak możliwość tworzenia kolumn obliczeniowych w tabelach zawierających ponad milion wierszy jest istotna. Definiowaliśmy kolumny obliczeniowe w tabelach o ponad *100 milionach* wierszy. A to już jest coś.

Ale jeżeli pominię się tę jedną nową cechę, wtedy okaże się, że kolumny obliczeniowe w dodatku Power Pivot nie pozwalają na zrobienie niczego więcej niż to, co można osiągnąć w zwykłym Excelu. W końcu ta funkcjonalność dodatku Power Pivot umożliwia utworzenie nowej kolumny zapełnionej jakimiś wartościami — tak samo jak w Excelu.

Pola obliczeniowe natomiast z całą pewnością wyposażą Cię w narzędzie pozwalające uzyskać zdecydowanie bardziej elastyczne, przejrzyste i użyteczne rezultaty niż te uzyskane za pomocą zwykłego Excela. Jeśli chodzi o formuły Power Pivot, to pola obliczeniowe są ich główną atrakcją, a kolumny obliczeniowe jeszcze bardziej zwiększają ich znaczenie.

Kolumny obliczeniowe w dodatku Power Pivot w większym stopniu niż pola obliczeniowe wymagają od doświadczonych użytkowników Excela zmiany przyzwyczajeń. Brak adresacji typu „A1” w formułach DAX jest bardziej widoczny w kolumnach niż w polach obliczeniowych, co często irytuje nowicjuszy. („W Excelu to jest takie proste, a nie można tego zrobić w Power Pivot”).

UWAGA

Opisane kłopoty pojawiają się w chwili, gdy pierwszy raz trzeba w formule odwołać się do jednej lub kilku wartości znajdujących się w innym niż bieżący wierszu tabeli. W praktyce w każdym innym przypadku kolumny obliczeniowe Power Pivot są w prostsze w obsłudze niż zwykłe kolumny z funkcjami w Excelu, ponieważ składnia formuł wykorzystuje nazwy kolumn. Na przykład obliczenia w ramach bieżącego wiersza są dziecinnie proste, ale coś, co my nazywamy „obliczeniami krzyżowymi”, wymaga już pewnej gimnastyki umysłowej.

Ale ja (Rob) odwiedziłem już „kraję problemów z obliczeniami krzyżowymi”. Moja miłość do kolumn obliczeniowych Power Pivot rosła w miarę upływu czasu nawet w przypadku obliczeń krzyżowych. Dlatego celem tego rozdziału jest pomóc Ci w szybkim dotarciu do szczęśliwego końca tej drogi.

Podręcznik do kolumn obliczeniowych dla specjalisty od Excela

Jeżeli jesteś specjalistą od Excela usiłującym opanować kolumny obliczeniowe w dodatku Power Pivot, napotkasz na swej drodze kilka przeszkód, o które możesz się potknąć. W Excelu dostępnych jest około 400 różnych funkcji. Tylko jedną piątą tych funkcji znajdziesz w Power Pivot, ale za to jest to grupa zdecydowanie najlepszych funkcji. Jeżeli chcesz użyć MID(), LEFT(), RIGHT(), IF() lub innej popularnej funkcji, nic nie stoi na przeszkodzie... Będą działać zgodnie z oczekiwaniami.

Osobliwości funkcji TEXT() i FORMAT()

Niektóre funkcje działają tak samo w dodatku Power Pivot, jak i w Excelu, jednak jest kilka wyjątków. Funkcja TEXT() (w wersji polskiej TEKST()) w Excelu przydaje się do konwersji dat na nazwy miesięcy lub dni tygodnia i na pewno będziesz chciał z niej korzystać. Jednak z jakiegoś powodu zespół rozwijający dodatek Power Pivot postanowił do powyższych celów przeznaczyć funkcję VBA FORMAT(), a nie TEXT(). Lubimy żartować, że jest to zwykła literówka — zamiast T-E-X-T wyszło F-O-R-M-A-T, ale w rzeczywistości te dwie funkcje istotnie różnią się między sobą.

Jeżeli zazwyczaj w Excelu używasz funkcji =TEKST(A2, "MMMM"), to w Power Pivot przekłada się ona na =FORMAT([Data]; "MMMM"). Obie funkcje jako argument przyjmują pole z datą i zwracają nazwę miesiąca, na przykład „styczeń”.

Funkcja FORMAT() dopuszcza dwa kody formatu, których nie obsługuje funkcja TEXT(). Na przykład wartość Q powoduje zwrócenie numeru kwartału (od 1 do 4), a WW numeru tygodnia (od 1 do 52). Te różnice stanowią o przewadze funkcji FORMAT() nad TEXT(), ale osoba niewtajemniczona może o tym nie wiedzieć.

Gdzie jest funkcja WYSZUKAJ.PIONOWO()?

Funkcją numer jeden dla specjalistów od Excela jest WYSZUKAJ.PIONOWO(). Nie jest ona dostępna w Power Pivot z ważnego powodu: w większości przypadków nie jest potrzebna! Relacje umożliwiają połączenie pól tabeli A z polami tabeli B bez konieczności „scalania” obu tabel. Jednak w przypadku, gdy w kolumnie obliczeniowej naprawdę potrzebne jest umieszczenie odwołań do innej tabeli, Power Pivot oferuje kilka doskonałych funkcji, na przykład RELATED() (opisaną w książce *DAX Formula for Power Pivot*) i LOOKUPVALUE() (opisaną w tym rozdziale).

Nie można wskazać kolumny za pomocą klawiszy strzałek

Nie ma na to sposobu. Specjaliści od Excela powinni być wdzięczni Robowi Colliemu za to, że pracował w zespole rozwijającym dodatek Power Pivot. Ale Rob jest młody. Młodszy specjaliści od Excela tworzą formuły, klikając kolumny myszą, inni w całości je wpisują. Starsi specjaliści często wskazują kolumnę za pomocą klawiszy strzałek. Ta technika pochodzi jeszcze z czasów programu Lotus 1-2-3. Na przykład chcąc wpisać w komórce E2 formułę =B2*C2/D2, możesz użyć tej techniki

w następujący sposób: naciśnij klawisz =, następnie strzałkę w lewo, strzałkę w lewo, strzałkę w lewo, *, strzałkę w lewo, strzałkę w lewo, /, strzałkę w lewo, Enter. Opis brzmi strasznie, ale gdy już się opanuje tę technikę, okazuje się ona bardzo szybka. Na naszych seminariach 70% uczestników używa myszy, 10% wpisuje formuły w całości, a 20% korzysta z klawiszy strzałek. Siatka w oknie Power Pivot pozwala korzystać z myszy lub wpisywać formuły ręcznie, ale nie umożliwia użycia sposobu z klawiszami strzałek.

Sumy pośrednie w kolumnach obliczeniowych i funkcja EARLIER()

Załóżmy, że masz bardzo prostą tabelę, na przykład taką:

Klient	Liczba
a	1
a	2
b	3
b	4

Rysunek 3.1.

Chcesz do niej dodać trzecią kolumnę zawierającą sumę wartości dla każdego klienta:

Klient	Liczba	Suma dla klienta
a	1	3
a	2	3
b	3	7
b	4	7

Rysunek 3.2.

Tak wygląda formuła w trzeciej kolumnie obliczeniowej:

```
=CALCULATE(SUM(Tabela[Liczba]);
    FILTER(Tabela; Tabela[Klient]=EARLIER(Tabela[Klient]))
)
```

Jeżeli nie znasz dokładnie kontekstu wiersza (opisanego dwa akapity dalej), postępuj po prostu według przedstawionego schematu i dodaj trzecią kolumnę. Proponujemy, aby początkujący użytkownicy wykorzystali powyższą formułę jako wzorzec, podstawiając w nim nazwy własnych tabel i kolumn. Później można przejść dalej, nie przejmując się niewiedzą o tym, co naprawdę zostało zrobione. Nie szkodzi. Masz na to nasze pozwolenie i zachętę, szczególnie na początku.

A poważnie, możesz przejść do następnej sztuczki opisanej w tym rozdziale. Musisz tylko przeczytać pozostałą część tego podrozdziału, jeżeli zdecydujesz się osiągnąć wyższy stopień wtajemniczenia.

O, jednak tu jesteś? Zatem dobrze, oto wyjaśnienie, o co chodzi w kolumnie obliczeniowej z sumami pośrednimi. Ponieważ tabela ma cztery wiersze, formuła w kolumnie obliczeniowej musi być „uruchomiona” cztery razy — po jednym razie dla każdego wiersza, w którym będzie umieszczony wynik (w tym przypadku będą to liczby 3, 3, 7 i 7, jak na rysunku 3.2). W każdym z tych czterech kroków odwołanie do kolumny zostanie przekształcone na wartość zapisaną w wskazanej kolumnie i w „bieżącym” wierszu. Tak więc na przykład w drugim wierszu tabeli odwołanie `Tabela[Liczba]` zwróci wartość 2, zgodnie z oczekiwaniami.

Jeżeli opis wydaje się skomplikowany, chcielibyśmy pokazać, że nie ma tu nic skomplikowanego, ponieważ przedstawiony proces przebiega dokładnie tak, jak się tego oczekuje. Załóżmy teraz, że tworzysz kolejną kolumnę obliczeniową, tym razem zawierającą następującą formułę:

```
=CONCATENATE(Tabela[Klient]; Tabela[Liczba])
```

Otrzymasz oczywiście dokładnie to, czego oczekujesz:

Klient	Liczba	Suma dla klienta	CalculatedColumn1
a	1	3	a1
a	2	3	a2
b	3	7	b3
b	4	7	b4

Rysunek 3.3.

Jest tak dlatego, że w każdym z czterech „kroków” wykonywanych w kolumnie obliczeniowej (po jednym kroku na każdy wiersz) odwołanie do kolumny jest z zasady ograniczone do bieżącego wiersza. Byłoby dość dziwne, gdyby w pierwszym wierszu odwołanie do kolumny `[Liczba]` zwróciło wartość 2. Oczywiście oczekujemy, że zwrócona zostanie wartość 1.

Ponieważ formuła w kolumnie obliczeniowej wyliczana jest jeden raz dla każdego wiersza, istnieje wymyślny termin na określenie koncepcji „bieżącego wiersza w każdym kroku”. Jest to *kontekst wiersza*. Gdybyśmy użyli tego terminu w opisie przedstawionego wcześniej prostego procesu, brzmiałoby to tak: „podczas wyliczania kolumny obliczeniowej w czterowierszowej tabeli mechanizm DAX analizuje cztery różne konteksty wiersza i wylicza formułę jeden raz dla każdego kontekstu”. Jak na razie wygląda to niezłe, prawda? Teraz weź głęboki wdech i powoli wypuść powietrze...

Problem polega na tym, że formuła w kolumnie `Suma dla klienta` zawiera funkcję `FILTER()`, która jest dość podstępna. Formuła w kolumnie obliczeniowej jest wyliczana jeden raz w każdym kontekście wiersza, a funkcja `FILTER('Tabela', ...)` dodatkowo za każdym razem przegląda wszystkie wiersze w tabeli, zupełnie tak samo jak formuła!

Jedną z kluczowych, charakterystycznych cech funkcji `FILTER()` polega na pomijaniu kontekstu wiersza w kolumnie obliczeniowej i operowaniu na całej tabeli. Może to być mylące (na początku), ale też wspaniałe (gdy zrozumiesz jej działanie).

Ponieważ tabela ma cztery wiersze, za każdym razem, gdy wyliczana jest funkcja `FILTER('Tabela', ...)`, wykonywane są cztery osobne kroki. W każdym kroku funkcja sprawdza wiersz i pyta: „Hej, mam uwzględnić ten wiersz czy go odrzucić?”.

W sumie podczas tworzenia kolumny obliczeniowej jest więc wykonywanych 16 kroków:

Klient	Liczba	Suma dla klienta
a	1	3
a	2	3
b	3	7
b	4	7

Klient	Liczba	
a	1	F1
a	2	F2
b	3	F3
b	4	F4

Rysunek 3.4.

Na powyższym rysunku symbol *KO1* oznacza „kontekst pierwszego wiersza, w którym wykonywana jest formuła kolumny obliczeniowej”, *F1* natomiast oznacza „kontekst pierwszego wiersza, w którym wykonywana jest funkcja `FILTER()`”.

Możesz sobie wyobrazić „nazwy” wszystkich 16 kontekstów, w których wykonywana jest formuła z kolumny obliczeniowej. Nazwy mają postać *KO1/F1*, *KO1/F2* itd. aż do *KO4/F3* i *KO4/F4*. (Te nazwy są zupełnie nieoficjalne. Konteksty w formułach DAX nie mają swoich nazw. Używamy ich tylko na potrzeby niniejszego opisu).

Są zatem cztery „konteksty wierszy dla kolumn obliczeniowych”, a wewnątrz każdego z nich są schowane cztery „konteksty wierszy dla funkcji `FILTER()`” — jest to zewnętrzna i wewnętrzna pętla z czterema kontekstami wierszy każda.

Teraz wewnątrz funkcji `FILTER()` wykonywane jest „podstawowe” odwołanie do kolumny zgodnie z kontekstem wiersza z funkcją, a nie z kolumną obliczeniową. Czy to jest zrozumiałe? Wewnątrz funkcji `FILTER()` odwołanie do kolumny 'Tabela' [Klient] jest realizowane zgodnie z „wewnętrzną” pętlą odliczającą konteksty wierszy *F1*, *F2* itd., z całkowitym pominięciem „zewnętrznej” pętli odliczającej konteksty *KO1* itd.! (Pamiętaj, że funkcja `FILTER()` musi sprawdzić każdy wiersz w tabeli, aby ocenić, czy ma go uwzględnić w obliczeniach, czy odrzucić). Dlatego nawet gdy formuła jest wyliczana w „zewnętrznym” kontekście *KO2* (w którym [Klient]=„a”), to „podstawowe” odwołanie do kolumny 'Tabela' [Klient] wewnątrz funkcji `FILTER()` niekiedy spowoduje, że zwrócona zostanie wartość „b” — w kontekstach *KO2/F3* i *KO2/F4*.

W jaki sposób wewnątrz funkcji `FILTER()` można „cofnąć się” do zewnętrznej pętli w celu sprawdzenia wartości pola [Klient] w kontekście *KO2*? Korzystając z funkcji `EARLIER()`! W skrócie, funkcja ta działa następująco: „podczas odczytywania wartości w określonej kolumnie wyjdź poza bieżący kontekst wiersza i wejdź w kontekst zewnętrzny (wcześniejszy, ang. *earlier*)”. Jak się przekonasz, równie często będziemy używać funkcji `EARLIER()`, jak i `CURRENTROW()` (bieżący wiersz). Funkcje te użyte w funkcji `FILTER()` w kolumnie obliczeniowej umożliwiają odczytywanie wartości z każdego wiersza w tabeli i porównanie ich z wartościami z bieżącego wiersza. W ten sposób możesz wziąć jeden wiersz tabeli, a następnie przejrzeć wszystkie pozostałe, pytając „Hej, jesteście podobni do tego wiersza? Jeżeli tak, chcę was policzyć”.

Poza tym może (choć bardzo rzadko) wystąpić sytuacja, w której będą użyte więcej niż dwie zagnieżdżone pętle odliczające konteksty wierszy. Dlatego funkcja `EARLIER()` ma opcjonalny drugi argument `Number` do określenia liczby „zewnętrznych” pętli, poza które chcesz wyjść. Jeżeli

pominiesz ten argument, zostanie przyjęta domyślna wartość 1. Jest to też powód, dla którego istnieje również funkcja EARLIEST(). Jej przeznaczeniem jest wyjście do *najbardziej* zewnętrznego kontekstu wiersza, bez względu na to, jak głęboko pętle będą zagnieżdżone.

UWAGA

Do chwili pisania tej książki Twoi drodzy autorzy nigdy nie musieli wychodzić dalej niż jeden krok na zewnątrz pętli i dlatego też nigdy nie używali argumentu Number funkcji EARLIER() ani funkcji EARLIEST(). Jak dotąd nawet nie odczuliśmy potrzeby, aby tak robić.

Dawno temu, gdy używałem (ja, Bill) dodatku Power Pivot w firmie Data Analyst, desperacko próbowałem wymyślić przykład zastosowania funkcji EARLIER() lub EARLIEST(). Dzisiaj, po kilku latach, używamy funkcji EARLIER() w zasadzie tylko z funkcją SUMIF(), co jest bardzo proste i przydatne.

Odwołania do poprzedniego wiersza i temu podobne obliczenia

Oto bardzo często zadawane pytanie w świecie kolumn obliczeniowych: „Jak odwołać się do wiersza umieszczonego tuż »nad« danym wierszem”?

Na przykład jak utworzyć kolumnę obliczeniową taką jak przedstawiona niżej?

Data	Wartość	Wartość z wczoraj
2014-01-01	60	
2014-01-02	57	60
2014-01-03	68	57
2014-01-04	56	68
2014-01-05	64	56
2014-01-06	64	64
2014-01-07	57	64

Rysunek 3.5.

Jak to zwykle bywa, odpowiedź jest bardzo podobna do tej z omówionego przed chwilą przykładu z sumami pośrednimi:

```
=CALCULATE(SUM('PatrzacWstecz'[Wartość]);
    FILTER('PatrzacWstecz';
        'PatrzacWstecz'[Data] = EARLIER('PatrzacWstecz'[Data])-1
    )
)
```

W rzeczywistości jedyna różnica polega na wykonaniu działania -1 na końcu trzeciego wiersza formuły!

Zwróć uwagę na następujące kwestie:

- Kolejność sortowania tabeli nie ma wpływu na wynik formuły. Przedstawiona formuła wyszukuje wiersz (wiersze), w którym wartość w kolumnie Data jest wcześniejsza o jeden dzień niż data w bieżącym wierszu, a następnie sumuje wartości z kolumny Wartość.
- Jeżeli chcesz sięgnąć do następnego, a nie poprzedniego wiersza (wierszy), zmień działanie z -1 na $+1$.
- Analogicznie, jeżeli chcesz cofnąć się o tydzień, zmień -1 na -7 .
- Opisana technika sprawdza się również w przypadku kolumn zawierających inne dane niż daty. Jeżeli w tabeli znajduje się na przykład kolumna NumerMiesiąca, odjęcie od niej wartości 1 także jest możliwe.
- Często zamiast kolumny obliczeniowej takiej jak powyższa lepiej jest utworzyć pole obliczeniowe. Jednak dokładne omówienie tego zagadnienia wykracza poza zakres niniejszej książki. (Przykłady użycia takiego pola znajdziesz w dalszej części tego rozdziału i w rozdziale 4.)
- Sytuacje, w których potrzebne są kolumny obliczeniowe, na pewno się zdarzają, dlatego została tu opisana ta technika.

Odwołania do wierszy „z zakresu” bieżącego wiersza

Czytając opis powyższej sztuczki, zapewne dziwiłeś się, dlaczego w formule została użyta funkcja `SUM()`, skoro trzeba było sięgnąć tylko do poprzedniego wiersza. Cóż, dodatek Power Pivot „nie dowierza”, że przy użyciu funkcji `FILTER()` chcesz znaleźć tylko jeden wiersz. Zawsze przyjmuje, że wyszukanych ma być więcej wierszy. Z tego też powodu zawsze wymagane jest użycie funkcji agregującej, na przykład `SUM()`. Oczywiście jeżeli znajdowany jest tylko jeden wiersz, to wybór funkcji agregującej często nie ma znaczenia. Wszystkie funkcje takie jak `SUM()`, `AVERAGE()`, `MAX()` itp. zwrócą ten sam wynik.

Przyjrzyjmy się kilku przykładom. Zauważ, że w tabeli przedstawionej poniżej znajdują się wiersze zawierające tę samą datę:

Data	Wartość
2014-01-01	54
2014-01-01	67
2014-01-01	50
2014-01-02	51
2014-01-02	55
2014-01-02	63
2014-01-03	65
2014-01-03	60
2014-01-03	53

Rysunek 3.6.

Poniżej widać sumy z poprzedniego dnia:

Data	Wartość	Suma z wczoraj
2014-01-01	54	
2014-01-01	67	
2014-01-01	50	
2014-01-02	51	171
2014-01-02	55	171
2014-01-02	63	171
2014-01-03	65	169
2014-01-03	60	169
2014-01-03	53	169

Rysunek 3.7.

Formuła użyta w tej tabeli jest dokładnie taka sama jak w poprzedniej sztuczce. Zmieniona została jedynie nazwa tabeli:

```
=CALCULATE(SUM('WieleWartości'[Wartość]);
    FILTER('WieleWartości';
        'WieleWartości'[Data]=EARLIER('WieleWartości'[Data])-1
    )
)
```

A tu jest przykład sumy ruchomej uwzględniającej bieżącą datę:

Data	Wartość	Suma z wczoraj	Suma ruchoma
2014-01-01	54		171
2014-01-01	67		171
2014-01-01	50		171
2014-01-02	51	171	340
2014-01-02	55	171	340
2014-01-02	63	171	340
2014-01-03	65	169	518
2014-01-03	60	169	518
2014-01-03	53	169	518

Rysunek 3.8.

Formuła wygląda jak niżej:

```
=CALCULATE(SUM('WieleWartości'[Wartość]);
    FILTER('WieleWartości';
        'WieleWartości'[Data] <= EARLIER('WieleWartości'[Data])
    )
)
```

UWAGA

Aby wykluczyć z obliczeń wiersze z bieżącą datą, zmień operator <= na <.

Idąc dalej tym tropem, poniżej przedstawiamy trzydniową sumę ruchomą obejmującą bieżący, poprzedni i następny dzień:

Data	Wartość	Suma z wczoraj	Suma ruchoma	Suma ruchoma 3-dniowa
2014-01-01	54		171	340
2014-01-01	67		171	340
2014-01-01	50		171	340
2014-01-02	51	171	340	518
2014-01-02	55	171	340	518
2014-01-02	63	171	340	518
2014-01-03	65	169	518	546
2014-01-03	60	169	518	546
2014-01-03	53	169	518	546
2014-01-04	69	178	717	534
2014-01-04	65	178	717	534
2014-01-04	65	178	717	534
2014-01-05	54	199	874	535

Rysunek 3.9.

A tak wygląda formuła:

```
=CALCULATE(SUM('WieleWartości'[Wartość]);
    FILTER('WieleWartości';
        'WieleWartości'[Data] >= EARLIER('WieleWartości'[Data])-1 &&
        'WieleWartości'[Data] <= EARLIER('WieleWartości'[Data])+1
    )
)
```

UWAGA

Oczywiście możesz zamienić funkcję SUM() na przykład na MAX() i wyszukać wartość z „najlepszego” spośród trzech dni. Przypomnijmy jednak jeszcze raz: często, chociaż *nie* zawsze, zamiast tworzyć kolumny obliczeniowe takie jak powyższe, lepiej jest utworzyć pole obliczeniowe.

Krzyżowe przeszukiwanie tabel: funkcja LOOKUPVALUE() i inne techniki

Wszystkie przykłady opisane dotąd w tym rozdziale dotyczą przeszukiwania wierszy tej samej tabeli. Ale co zrobić, jeżeli trzeba pobrać wartości z innych tabel? Przyjrzyjmy się przykładowemu problemowi. Mamy zatem tabelę Sprzedaż:

Data	Produkt	Liczba
2014-01-01	A	37
2014-01-01	B	47
2014-01-01	A	46
2014-01-01	B	41
2014-01-02	A	45
2014-01-02	B	20
2014-01-02	A	36
2014-01-02	B	27
2014-01-03	A	38
2014-01-03	B	41
2014-01-03	A	30
2014-01-03	B	38

Rysunek 3.10.

UWAGA

Tabela Sprzedaż zawiera wiele wierszy z takimi samymi parami wartości w kolumnach Data i Produkt.

Jest również tabela Ceny:

Data	Produkt	Cena jednostkowa
2014-01-01	A	1,50 zł
2014-01-01	B	3,10 zł
2014-01-02	A	1,60 zł
2014-01-02	B	3,25 zł
2014-01-03	A	2,00 zł
2014-01-03	B	2,75 zł

Rysunek 3.11.

UWAGA

W tabeli Ceny jest tylko jeden wiersz z daną parą wartości w kolumnach Data i Produkt.

Teraz chcemy w kolumnie obliczeniowej w tabeli Sprzedaż uwzględnić odpowiednią wartość pola Cena jednostkowa z tabeli Ceny.

„Najprostszym” rozwiązaniem byłoby użycie funkcji RELATED(). Jeżeli pomiędzy tymi dwiema tabelami jest utworzona relacja, to sytuacja wygląda całkiem niezłe. Ale niestety, w tym przypadku nie ma relacji. Co więcej, nie ma kolumny, której można byłoby użyć do jej utworzenia. Aby utworzyć relację, trzeba zdefiniować nowe kolumny w obu tabelach (Sprzedaż i Ceny) zawierające związane wartości z kolumn Data i Produkt. Potem, używając tych „hybrydowych” kolumn, będzie można utworzyć relację. Ten sposób wymaga mnóstwa pracy. Ponadto utworzenie kolumny obliczeniowej powoduje, że znacznie powiększa się rozmiar pliku, szczególnie jeżeli łączone tabele są duże.

Poniżej natomiast przedstawiona jest formuła umożliwiająca odczytanie wartości z kolumny Cena jednostkowa bez uprzedniego tworzenia relacji:

```
=LOOKUPVALUE(Ceny[Cena_jednostkowa];
              Ceny[Data]; 'Sprzedaż'[Data]; Ceny[Produkt]; 'Sprzedaż'[Produkt])
```

Formuła jest bardzo prosta. Pierwszy argument funkcji LOOKUPVALUE() określa kolumnę, z której chcesz odczytywać wartości. Następnie wskazywana jest para kolumn — najpierw z tabeli „źródłowej” i odpowiadająca jej kolumna z tabeli „docelowej”. Praktycznie można wskazać dowolną liczbę par kolumn. I to działa:

Data	Produkt	Liczba	Efektywna cena jednostkowa
2014-01-01	A	37	1,50 zł
2014-01-01	B	47	3,10 zł
2014-01-01	A	46	1,50 zł
2014-01-01	B	41	3,10 zł
2014-01-02	A	45	1,60 zł
2014-01-02	B	20	3,25 zł
2014-01-02	A	36	1,60 zł
2014-01-02	B	27	3,25 zł
2014-01-03	A	38	2,00 zł
2014-01-03	B	41	2,75 zł
2014-01-03	A	30	2,00 zł
2014-01-03	B	38	2,75 zł

Rysunek 3.12.

Co się stanie, jeżeli funkcja LOOKUPVALUE() znajdzie więcej niż jedną wartość?

Tak, to dobre pytanie. Aby sprawdzić, jak to wygląda w praktyce, umieść w tabeli Ceny jeszcze jeden wiersz z produktem A i datą 1 stycznia. Tym razem wprowadź taką samą cenę (1,50 zł) jak w oryginalnym wierszu:

Data	Produkt	Cena jednostkowa
2014-01-01	A	1,50 zł
2014-01-01	B	3,10 zł
2014-01-01	A	1,50 zł

Rysunek 3.13.

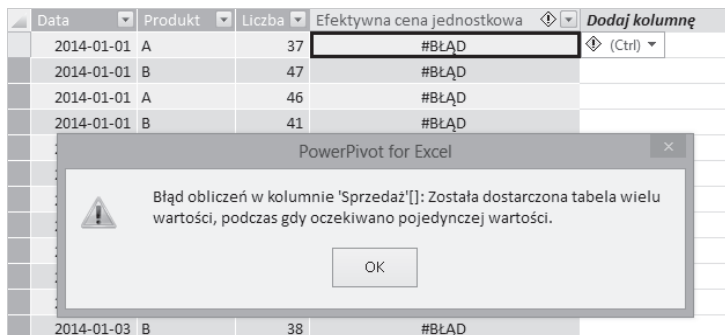
Następnie sprawdź zawartość kolumny obliczeniowej z funkcją LOOKUPVALUE():

Data	Produkt	Liczba	Efektywna cena jednostkowa
2014-01-01	A	37	1,50 zł
2014-01-01	B	47	3,10 zł
2014-01-01	A	46	1,50 zł
2014-01-01	B	41	3,10 zł
2014-01-02	A	45	1,60 zł
2014-01-02	B	20	3,25 zł
2014-01-02	A	36	1,60 zł
2014-01-02	B	27	3,25 zł
2014-01-03	A	38	2,00 zł
2014-01-03	B	41	2,75 zł
2014-01-03	A	30	2,00 zł
2014-01-03	B	38	2,75 zł

Rysunek 3.14.

A to dopiero, to i tak działa! Fajnie, formuła wykryła, że mimo iż znalazła kilka wierszy, wszystkie zwróciły tę samą wartość (1,50 zł), dzięki czemu nie wystąpił błąd. Wyrazy uznania dla programistów za dobrą robotę!

Jeżeli cofniesz się i zmienisz wartość w „nowym” wierszu na 1,60 zł, co spowoduje konflikt wartości, wtedy rzeczywiście pojawi się błąd:



Rysunek 3.15.

A jeżeli funkcja LOOKUPVALUE() nie spełni Twoich oczekiwań?

A jeżeli poprzedni przykład z dwoma wierszami w tabeli Ceny z tą samą parą Data/Produkt, ale różną ceną, będzie jak najbardziej uzasadniony? Co wtedy trzeba zrobić? Cóż, trzeba wrócić do naszych starych znajomych, czyli funkcji CALCULATE() i FILTER():

```
=CALCULATE(AVERAGE(Ceny[Cena_jednostkowa]);
            FILTER(Ceny;
                  Ceny[Data]='Sprzedaż'[Data] &&
                  Ceny[Produkt]='Sprzedaż'[Produkt]
            )
)
```

Funkcje działają niezawodnie:

Średnia z 1,50 zł i 1,60 zł

Data	Produkt	Liczba	Efektywna cena jednostkowa	Efektywna średnia cena jedn.
2014-01-01	A	37	#BŁĄD	1,55 zł
2014-01-01	B	47	#BŁĄD	3,10 zł
2014-01-01	A	46	#BŁĄD	1,55 zł
2014-01-01	B	41	#BŁĄD	3,10 zł
2014-01-02	A	45	#BŁĄD	1,60 zł
2014-01-02	B	20	#BŁĄD	3,25 zł
2014-01-02	A	36	#BŁĄD	1,60 zł
2014-01-02	B	27	#BŁĄD	3,25 zł
2014-01-03	A	38	#BŁĄD	2,00 zł
2014-01-03	B	41	#BŁĄD	2,75 zł
2014-01-03	A	30	#BŁĄD	2,00 zł
2014-01-03	B	38	#BŁĄD	2,75 zł

Rysunek 3.16.

UWAGA

Jako pierwszego argumentu funkcji CALCULATE() nie musisz używać funkcji AVERAGE(). Możesz zastosować MIN(), MAX(), SUM() lub dowolną inną funkcję agregującą. W przypadkach „wielodopasowań” Twój wybór funkcji agregującej oczywiście ma *bardzo* istotne znaczenie.

Teraz przyjrzyjmy się kilku podobnym, ale bardziej złożonym przypadkom, które spotyka się od czasu do czasu.

Przeszukiwanie danych na podstawie daty początkowej i końcowej

Tematem niniejszego rozdziału są najczęstsze wyzwania, a opisany niżej problem całkowicie się do nich kwalifikuje. Poniżej przedstawiona jest tabela zawierająca mnóstwo danych:

Data	Przedstawiciel
2014-01-01	Agnieszka
2014-01-02	Robert
2014-01-03	Robert
2014-01-04	Kasia
2014-01-05	Paweł
2014-01-06	Kasia
2014-01-07	Robert
2014-01-08	Robert
2014-01-09	Paweł
2014-01-10	Paweł

Rysunek 3.17.

Oprócz tego jest jeszcze innego rodzaju tabela z cenami. Tabela Ceny2 nie podaje cen z każdego dnia osobno, ale zawiera kolumny Początek i Koniec określające przedziały czasu, w których obowiązywała podana cena:

Początek	Koniec	Cena
2014-01-01	2014-02-07	100,00 zł
2014-02-07	2014-05-25	150,00 zł
2014-05-25	2014-10-12	200,00 zł
2014-10-12	2014-12-31	150,00 zł

Rysunek 3.18.

Aby w tabeli Planowanie umieścić obowiązującą w danym dniu cenę, użyj następującej formuły:

```
=CALCULATE(MAX(Ceny2[Cena]);
    FILTER(Ceny2;
        Ceny2[Początek] <= Planowanie[Data] &&
        Ceny2[Koniec] >= Planowanie[Data]
    )
)
```

Wynik wygląda tak:

Zmiana ceny 7 lutego

Data	Przedstawiciel	Cena efektywna (Ceny2)
2014-02-04	Paweł	100,00 zł
2014-02-05	Paweł	100,00 zł
2014-02-06	Paweł	100,00 zł
2014-02-07	Kasia	150,00 zł
2014-02-08	Agnieszka	150,00 zł
2014-02-09	Kasia	150,00 zł

Rysunek 3.19.

Zmiana ceny 7 lutego jest poprawna. Zadanie wykonane.

Przeszukiwanie tylko na podstawie daty początkowej

Załóżmy teraz, że mamy ten sam problem, ale tym razem tabela nie zawiera kolumny Koniec, jedynie kolumnę Początek. Przyjmijmy, że mamy tabelę Ceny3, która jest w rzeczywistości tabelą Ceny2 z usuniętą kolumną Koniec:

Początek	Cena
2014-01-01	100,00 zł
2014-02-07	150,00 zł
2014-05-25	200,00 zł
2014-10-12	150,00 zł

Rysunek 3.20.

I co teraz? W zasadzie musisz znaleźć w tej tabeli „ostatni” wiersz, zawierający datę wcześniejszą lub taką samą jak bieżąca. Jest to dość trudne. Jeden ze sposobów polega na odtworzeniu w tabeli Ceny3 kolumny obliczeniowej Koniec, jak poniżej:

```
=CALCULATE(MIN([Początek]);
    FILTER(Ceny3;
        [Początek] > EARLIER([Początek])
    )
)-1
```

Wynik jest następujący:

Początek	Cena	Koniec
2014-01-01	100,00 zł	2014-02-06
2014-02-07	150,00 zł	2014-05-24
2014-05-25	200,00 zł	2014-10-11
2014-10-12	150,00 zł	1899-12-29

Rysunek 3.21.

Teraz, wyposażony z powrotem w kolumnę *Koniec*, możesz zastosować technikę opisaną w poprzedniej części niniejszego rozdziału. Jednak gwoili uzupełnienia poniżej przedstawiamy rozwiązanie problemu niewymagające odtwarzania kolumny *Koniec*. W tabeli *Planowanie* utwórz następującą kolumnę obliczeniową:

```
=CALCULATE(MAX(Ceny3[Cena]);
    TOPN(1;
        FILTER(Ceny3;
            Ceny3[Początek] <= Planowanie[Data]
        );
        Ceny3[Początek]
    )
)
```

Powyższa kolumna obliczeniowa w ogóle nie odwołuje się do kolumny *Koniec*! Funkcja *FILTER()* odrzuca wszystkie wiersze z tabeli *Ceny3* zawierające daty późniejsze niż bieżąca data *Planowanie[Data]*, po czym funkcja *TOPN()* pobiera ostatni ze zwróconych wierszy! I oczywiście zwraca taki sam wynik jak w poprzednim przykładzie:

Data	Przedstawiciel	Cena efektywna (Ceny2)	Cena efektywna (Ceny3)
2014-02-04	Paweł	100,00 zł	100,00 zł
2014-02-05	Paweł	100,00 zł	100,00 zł
2014-02-06	Paweł	100,00 zł	100,00 zł
2014-02-07	Kasia	150,00 zł	150,00 zł
2014-02-08	Agnieszka	150,00 zł	150,00 zł
2014-02-09	Kasia	150,00 zł	150,00 zł

Rysunek 3.22.

UWAGA

Użycie funkcji *FILTER()* wewnątrz *TOPN()* może przyprawić Cię o ból głowy, co nam też się zdarzyło! „Normalnie” stosujemy funkcję *FILTER()* bezpośrednio w argumencie funkcji *CALCULATE()*, ale tutaj używamy jej w przypadku, gdy funkcja *TOPN()* wymaga podania tabeli. Jest to sytuacja, w której możesz przekonać się o istnieniu dwóch różnych, ale powiązanych ze sobą zastosowań funkcji takich jak *FILTER()* (czy też *TOPN()* i *DATESYTD()*) w formułach DAX. Ich przeznaczeniem jest filtrowanie danych (gdy zostaną użyte bezpośrednio w argumencie funkcji *CALCULATE()*) lub zwrócenie tabeli (jeżeli zostaną użyte jako argument tablicowy innej funkcji).

Gdy już sobie to uświadomisz, zalecamy Ci opanowanie chichotu (czy też maniakałnego śmiechu). Jest to moment, w którym poczujesz niemal nieograniczoną siłę, i to nie tylko w kwestii kolumn obliczeniowych. W praktyce funkcje te okazują się jeszcze bardziej przydatne w polach obliczeniowych! Poczujesz moment, w którym to sobie uświadomisz, uwierz nam. Nie da się go nie zauważyć. Wszyscy w biurze odwrócą się i zaczną dziwnie Ci się przyglądać, ale Tobie będzie wszystko jedno.

W zasadzie, jeżeli chcesz kojarzyć wartości z różnych tabel, najlepiej jest zacząć od funkcji *LOOKUPVALUE()* (przy założeniu, że użycie funkcji *RELATED()* nie jest możliwe ani praktyczne). Jeżeli to się nie uda, zajmij się funkcjami *CALCULATE()* i *FILTER()* czy nawet takimi kombinacjami jak *CALCULATE(..., TOPN(..., FILTER(...)))*.

UWAGA

Jest to temat, w którym „ruchomy obraz” jest wart więcej niż tysiąc obrazów albo milion słów. Chcemy uniknąć wypisywania w tym miejscu miliona słów, więc idźmy dalej. Jeżeli potrzebujesz obszerniejszych informacji na ten temat, weź udział w szkoleniu *PowerPivotPro University*, na którym znajdziesz ten i wiele innych tematów ilustrowanych i wyjaśnionych za pomocą animowanych prezentacji.

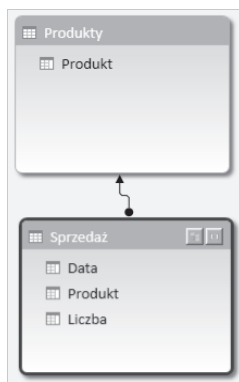
Sumowanie tabel danych w tabelach przeszukiwanych

Oto inny często spotykany problem. Załóżmy, że mamy tę samą co wcześniej tabelę Sprzedaż, która nie jest powiązana z tabelą Produkty. Sprzedaż to tabela danych, a Produkty to tabela przeszukiwana.

UWAGA

Dokładne wyjaśnienie terminów *tabela danych* i *tabela przeszukiwana* jest zawarte w książce *DAX Formulas for Power Pivot* i na stronie *PowerPivotPro University* (w jęz. angielskim).

Zgodnie z przyjętą przez nas konwencją tabela przeszukiwana jest umieszczona w widoku diagramu powyżej tabeli z danymi:



Rysunek 3.23.

A tak wyglądają obie tabele w widoku danych. Pierwsza z nich to tabela Produkty. Jest bardzo prosta, zawiera tylko dwa wiersze:

Produkt
A
B

Rysunek 3.24.

Dla przypomnienia, tak wygląda tabela Sprzedaż:

Data	Produkt	Liczba
2014-01-01	A	30
2014-01-01	B	32
2014-01-01	A	30
2014-01-01	B	46
2014-01-02	A	27
2014-01-02	B	20
2014-01-02	A	20
2014-01-02	B	31
2014-01-03	A	24
2014-01-03	B	26
2014-01-03	A	48
2014-01-03	B	23

Rysunek 3.25.

W tabeli Produkty jest potrzebna kolumna Liczba sprzedanych sztuk, jak poniżej:

Produkt	Liczba sprzedanych sztuk
A	179
B	178

Rysunek 3.26.

Jest kilka sposobów na popętnienie błędu podczas tworzenia formuły i kilka sposobów na osiągnięcie celu. Poniżej przedstawiona jest jedna z poprawnych formuł:

```
=CALCULATE(SUM('Sprzedaż'[Liczba]))
```

Co takiego? Funkcja CALCULATE() tylko z jednym argumentem? O co tu chodzi? Na ten temat moglibyśmy rozpisać się na wiele stron. Albo możemy po prostu powiedzieć, że funkcja CALCULATE() pobiera kontekst wiersza i przekształca go w kontekst filtru. Tak brzmi „poprawne” wyjaśnienie, ale powtórzmy, to jest jeden z tych przypadków, w których musimy wybierać między milionem słów a animacją. Albo mówiąc językiem programistów, możemy potraktować ten przykład jako wzorzec i nie troszczyć się o to, co się dzieje w środku.

Nawiasem mówiąc, okazuje się, że w tym przypadku użyteczna może być również funkcja SUM() bez funkcji CALCULATE():

```
=SUM('Sprzedaż'[Liczba])
```

Tak wygląda wynik:

Produkt	Liczba sprzedanych sztuk	Całkowita liczba sprzedanych sztuk
A	179	357
B	178	357

Rysunek 3.27.

Bardzo zgrabny sposób. Być może użyjesz czasami tej funkcji w mianowniku jakiegoś ułamka. Ponadto taka „surowa suma” *nie* opiera się na relacji między tabelami. Zawsze sumuje wartości ze wszystkich wierszy innej tabeli. (W praktyce często przydaje się nam funkcja MAX() użyta z kolumną z datami — patrz podrozdział „Najczęstsze wywołania związane z kalendarzem” poniżej).

Do kompletu poniżej przedstawiona jest pierwsza (prawidłowa) formuła, napisana w inny sposób:

```
=SUMX(RELATEDTABLE('Sprzedaż'); 'Sprzedaż'[Liczba])
```

Funkcja RELATEDTABLE() działa z definicji „odwrotnie” do funkcji RELATED(), ponieważ śledzi relację w przeciwnym kierunku. Funkcja RELATED() zawsze kojarzy jeden wiersz z tabeli przeszukiwanej z jednym wierszem z tabeli danych, RELATEDTABLE() natomiast kojarzy wszystkie wiersze z tabeli danych na potrzeby agregacji danych w tabeli przeszukiwanej.

Zwróć uwagę, że powyższa formuła zwraca te same wyniki co pierwsza:

Produkt	Liczba sprzedanych sztuk	Całkowita liczba sprzedanych sztuk	Funkcje SUMX i RELATEDTABLE
A	179	357	179
B	178	357	178

Rysunek 3.28.

Warto wiedzieć, że wyliczanie sum częściowych w tabeli przeszukiwanej, jak w tym przypadku, jest bardzo często lepiej zaimplementowane niż wyliczanie pól obliczeniowych. Jednym z częstych „uzasadnionych” powodów użycia kolumny obliczeniowej jest potrzeba utworzenia kolumny „grupującej”, na przykład Maksimum / Średnia / Minimum, w której produkty (lub inne elementy) są grupowane w poszczególnych kategoriach. Kolumny grupujące możesz umieścić w obszarach *Wiersze*, *Kolumny*, *Filtry* lub we fragmentatorze tabeli przestawnej i odpowiednio prezentować dane.

Funkcja CONTAINSX: wyszukiwanie wspólnych wartości w dwóch tabelach

Zamiast wyszukiwać czy agregować dane z różnych tabel, możesz niekiedy potrzebować sprawdzić, czy wartość w jednej tabeli ma swój odpowiednik w innej tabeli. Na przykład możesz chcieć wyróżnić wiersze w tabeli *Firmy* po lewej stronie poniższego rysunku, zawierające słowa kluczowe znajdujące się w tabeli *Słowa kluczowe* po prawej (zobacz rysunek 3.29).

Potrzeba wyszukiwania wspólnych wartości w dwóch tabelach pojawia się całkiem często. Czasami możesz to osiągnąć w dodatku Power Pivot, bardzo łatwo tworząc relację pomiędzy dwiema tabelami i kolumną obliczeniową z funkcją RELATED() w jednej z nich, i sprawdzać w ten sposób, czy w drugiej tabeli jest odpowiednia wartość. Ale niekiedy zdarza się, że ten sposób nie działa, na przykład gdy nie szukasz dokładnego, tylko „częściowego” odpowiednika.

Nazwa	Liczba	To jest spółka przemysłowa	Klucz
Jastrzębska Spółka Węglowa	363	TAK!	węglowy
PKO BP	267	Chyba nie	węglowa
GAZ-SYSTEM	950	TAK!	miedź
Tauron	703	Chyba nie	stal
Zakłady Metalurgiczne "POMET"	363	TAK!	gaz
Polska Grupa Energetyczna	763	Chyba nie	metal
KGHM Polska Miedź	751	TAK!	
Zakłady Metalurgiczne "POMET"	847	TAK!	
Grupa Lotos	177	Chyba nie	
PZU	645	Chyba nie	
ThyssenKrupp Energostal	313	TAK!	
PZU	859	Chyba nie	

Rysunek 3.29.

Tworzenie własnych funkcji X

Prawdopodobnie jedyną rzeczą, która mogłaby mnie (Roba) uszczęśliwić bardziej niż nowa funkcja X, byłaby możliwość „robienia” nowej funkcji (na przykład PRODUCTX() — patrz <http://pptv.pro/PRODUCTX>).

UWAGA

Ja (Bill) wciąż uparcie domagam się funkcji CONCATENATEX() (do łączenia wartości) lub ROMANX() (do rzymskiego zapisu liczb)!

Pewnego dnia patrzyłem na model danych Power Pivot i pomyślałem: „No tak, miło byłoby mieć funkcję CONTAINSX()”. Okazuje się, że można „zrobić” własną funkcję CONTAINSX() za pomocą funkcji SUMX().

Formuła kojarząca wartości z dwóch tabel

Poniżej przedstawiona jest formuła, której możesz użyć do sprawdzenia, czy w dwóch tabelach pojawia się wspólna wartość:

```
[To jest spółka przemysłowa] =
=IF(SUMX('Słowa kluczowe';
    FIND(UPPER('Słowa kluczowe'[Klucz]);
        UPPER(Firmy[Nazwa]));
    ; 0
)
) > 0;
"TAK!";
"Chyba nie"
)
```

UWAGA

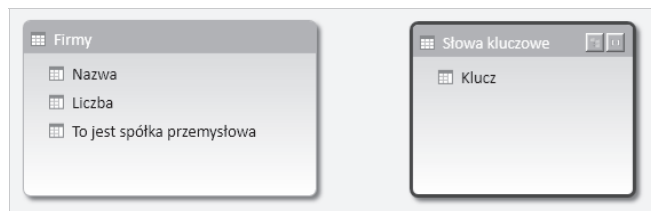
Powyższą formułę równie dobrze można zdefiniować na wiele innych sposobów.

Oto co robi ta formuła:

- Funkcja SUMX() przegląda każdy wiersz w tabeli Słowa kluczowe. W każdym wierszu wykonywana jest funkcja FIND(), która zwraca liczbę. Następnie funkcja SUMX() sumuje wszystkie liczby zwrócone przez funkcję FIND(). (Sumowanych jest pięć wartości, ponieważ tabela Słowa kluczowe zawiera sześć wierszy).
- Funkcja FIND() sprawdza również, czy ciąg znaków z wiersza tabeli Słowa kluczowe zawiera się w bieżącym wierszu tabeli Firmy. Jeżeli tak, wówczas funkcja FIND(), tak jak zwykle, zwraca numer pozycji, na której znajduje się ten ciąg. Jeżeli ciąg nie zostanie znaleziony, funkcja zwróci wartość 0.
- Funkcje UPPER() powodują, że funkcja FIND() nie uwzględnia wielkości znaków. Jeżeli powinna uwzględniać, należy usunąć funkcje UPPER().
- Jeżeli funkcja SUMX() zwróci wartość 0, oznacza to, że dopasowanie nie zostało znalezione. Wtedy funkcja IF() zwraca ciąg Chyba nie. Zwrócona jakakolwiek wartość różna od 0 oznacza, że zostało znalezione przynajmniej jedno (albo więcej) dopasowanie, i wtedy funkcja IF() zwraca ciąg Tak!

Relacje nie są potrzebne

Opisana technika nie wymaga relacji:



Rysunek 3.30.

Miliony odmian

Opisany wyżej przykład możesz przekształcić na wiele praktycznych sposobów. Możesz obliczać liczbę dopasowań, zamiast zwracać ciągi *Tak* lub *Nie*. Możesz nie uwzględniać wielkości znaków. Możesz stosować wyszukiwanie typu „zaczyna się od”, „kończy się na” lub wyszukiwać dokładne dopasowania.

Ponowne spojrzenie na funkcję CONTAINSX: jaka jest wspólna wartość?

Dowiedziałeś się właśnie, jak sprawdzić, czy istnieje wspólna wartość w dwóch tabelach. Możesz również sprawdzić, jaka to wartość:

Poprzednia technika		Obecna technika	
Nazwa	Liczba	To jest spółka przemysłowa	Wspólne słowo
Jastrzębska Spółka Węglowa	363	TAK!	węglowa
PKO BP	267	Chyba nie	
GAZ-SYSTEM	950	TAK!	gaz
Tauron	703	Chyba nie	
Zakłady Metalurgiczne "POMET"	363	TAK!	metal
Polska Grupa Energetyczna	763	Chyba nie	
KGHM Polska Miedź	751	TAK!	miedź
Zakłady Metalurgiczne "POMET"	847	TAK!	metal
Grupa Lotos	177	Chyba nie	
PZU	645	Chyba nie	
ThyssenKrupp Energostal	313	TAK!	stal
PZU	859	Chyba nie	

Rysunek 3.31.

Jak widziałeś przed chwilą, utworzenie formuły zwracającej informację o istnieniu wspólnej wartości jest całkiem proste. Ale co robić, gdy zechcesz dowiedzieć się, jakie słowo jest wspólne? Taką formułę również bardzo łatwo utworzyć, szczególnie w dodatku Power Pivot.

UWAGA

Określenie wspólnego słowa kluczowego jest przykładem zadania, które łatwiej jest zrealizować w dodatku Power Pivot niż w zwykłym Excelu. Gdy opanujesz definiowanie kolumn obliczeniowych w dodatku Power Pivot, prawdopodobnie stwierdzisz, że rozwiązywanie takich samych problemów w Excelu jest nieporęczne. Na początku niniejszego rozdziału wspomnieliśmy, że użycie funkcji RELATED() jest prostsze niż funkcji WYSZUKAJ.PIONOWO(). W tym przypadku implementowana jest funkcja WYSZUKAJ.PIONOWO() z symbolami wieloznacznymi. Oczywiście to samo jest możliwe w Excelu za pomocą formuły WYSZUKAJ.PIONOWO("&A1&"), ale 99% użytkowników Excela nigdy nie widziało takiego zastosowania funkcji WYSZUKAJ.PIONOWO().

Aby dowiedzieć się, jakie jest wspólne słowo kluczowe, utwórz kolumnę Wspólne słowo (zobacz rysunek 3.32).

Tak wygląda formuła dla tej kolumny:

```
[Wspólne słowo]=
FIRSTNONBLANK(FILTER(
    VALUES('Słowa kluczowe'[Klucz]);
    SEARCH('Słowa kluczowe'[Klucz]; Firmy[Nazwa]; 1; 0)
);
1
)
```

Nazwa	Liczba	To jest spółka przemysłowa	Wspólne słowo
Jastrzębska Spółka Węglowa	363	TAK!	węglowa
PKO BP	267	Chyba nie	
GAZ-SYSTEM	950	TAK!	gaz
Tauron	703	Chyba nie	
Zakłady Metalurgiczne "POMET"	363	TAK!	metal
Polska Grupa Energetyczna	763	Chyba nie	
KGHM Polska Miedź	751	TAK!	miedź
Zakłady Metalurgiczne "POMET"	847	TAK!	metal
Grupa Lotos	177	Chyba nie	
PZU	645	Chyba nie	
ThyssenKrupp Energostal	313	TAK!	stal
PZU	859	Chyba nie	

Rysunek 3.32.

Przyjrzyj się pierwszemu (pogrubionemu) fragmentowi funkcji `FIRSTNONBLANK()`. Jest to przykład niestandardowego użycia tej funkcji. Często możesz „nadużywać” funkcji `FIRSTNONBLANK()` w pożyteczny sposób, jak w tym przypadku.

Widzisz pogrubiony fragment 1 formuły? Jest to drugi argument funkcji `FIRSTNONBLANK()`, który w „standardowych” przypadkach jest zazwyczaj wyrażeniem. Na przykład możesz poszukiwać pierwszej daty, kiedy dany klient kupił coś, czyli kiedy pole obliczeniowe pierwszy raz zwraca niepustą wartość. W tym przypadku użycie liczby 1 wymusza wynik testu „Czy wartość jest pusta?” i funkcja zwraca pierwszą znaną wartość. Innymi słowy, funkcja `FIRSTNONBLANK()` użyta z drugim argumentem równym 1 jest prowizoryczną funkcją „wyszukaj pierwszą wartość tekstową”.

W powyższej formule fragment `FILTER(VALUES(SEARCH(...)))` (pomiędzy pogrubionymi fragmentami) ma tylko jedno zadanie: zwrócić jedną kolumnę z wartościami tekstowymi pobranymi z kolumny `Słowa kluczowe[Klucz]`. Jednak zwracane są tylko te wartości, które zostały znalezione w bieżącym wierszu kolumny `Fi rmy[Nazwa]`. Innymi słowy, zwracane są tylko znalezione słowa kluczowe.

Cały proces rozpoczyna się od wykonania funkcji `VALUES('Słowa kluczowe'[Klucz])`. Ponieważ nie ma żadnych relacji ani innych zależności pomiędzy tabelami `Fi rmy` a `Słowa kluczowe`, zawsze od razu zostaną zwrócone unikatowe wartości z kolumny `'Słowa kluczowe'[Klucz]`:

Klucz
węglowy
węglowa
miedź
stal
gaz
metal

Rysunek 3.33.

W tym fragmencie formuły kolumny obliczeniowej funkcja 'Słowa kluczowe' [Klucz] zawsze zwraca nieprzefiltrowaną listę wszystkich wartości ze wskazanej kolumny. Następnie do akcji wkracza funkcja FILTER() i pozostawia tylko te wiersze/wartości, dla których wyrażenie w drugim argumencie ma wartość PRAWDA.

Następnie formuła wykorzystuje funkcję SEARCH() jako drugi argument funkcji FILTER(). Ponieważ funkcja SEARCH(), gdy coś znajdzie, zwraca niezerową wartość, funkcja FILTER() potraktuje ją jak wartość PRAWDA i zwróci tylko wyszukane wartości.

Gdzie można użyć tej sztuczki?

Co można zrobić z kolumną obliczeniową pokazaną w tej sztuczce? A gdyby ją umieścić w obszarze *Wiersze* tabeli przestawnej wraz z kilkoma innymi polami obliczeniowymi? Nowa kolumna sprawdza się w takiej konfiguracji całkiem dobrze:

Branża	Liczba spółek	Średnia liczba
(nieokreślona)	379	530,6
gaz	157	586,9
metal	184	566,3
miedź	42	493,6
stal	115	546,1
węglowa	74	573,0
węglowy	49	541,2
Suma końcowa	1000	549,9

Rysunek 3.34.

Zauważ, że zmieniliśmy komórkę *Etykiety wierszy* tabeli przestawnej na *Branża*, a komórkę *(puste)* na *(nieokreślona)*. Możesz również zrobić kilka innych rzeczy:

- Użyć nowej kolumny we fragmentatorze zamiast w obszarze *Wiersze*.
- Użyć nowej kolumny jako filtru w funkcji CALCULATE(). Możesz wykorzystać nowe pole obliczeniowe o nazwie na przykład [Średnia liczba dla firm metalurgicznych] z formułą CALCULATE([Średnia Liczba]; Firmy[Wspólne słowo]="metal").
- Utworzyć nową kolumnę obliczeniową sumującą wartości węglowa i węglowy w jedną wartość, na przykład, hm, węglowa. Potem mógłbyś używać *tej* nowej kolumny do dowolnych opisanych wyżej celów.

Najczęstsze wyzwania związane z kalendarzem

Przegląd najczęściej spotykanych scenariuszy związanych z kolumnami obliczeniowymi nie byłby kompletny bez spędzenia chwili w świecie kalendarza, który będzie naszym ostatnim przystankiem w tym rozdziale.

Kolumna DataPrzyszła

Na poniższym rysunku widać, że ostatnia data w tabeli KontaktySerwisowe to 2014-06-01:

DataKontaktu	ProduktID	KlientID
2014-06-01	816	22117
2014-06-01	251	21216
2014-06-01	658	19090
2014-06-01	164	12702
2014-06-01	978	14599
2014-06-01	369	13034
2014-06-01	839	15478

Rysunek 3.35.

Jednak tabela Kalendarz sięga dalej, do grudnia 2014 roku:

Data	NrDniaTygodnia	Miesiąc
2014-12-26	5	grudzień
2014-12-27	6	grudzień
2014-12-28	7	grudzień
2014-12-29	1	grudzień
2014-12-30	2	grudzień
2014-12-31	3	grudzień

Rysunek 3.36.

Często w tabeli Kalendarz bardzo przydaje się kolumna obliczeniowa, w której znajdują się oznaczenia dla dat z przyszłości w odniesieniu do daty z innej tabeli. Poniżej przedstawiona jest jedna z takich formuł:

```
=IF([Data] > MAX(KontaktySerwisowe[DataKontaktu]); "Tak"; "Nie")
```

Gdy umieścisz tę formułę w tabeli Kalendarz, będzie ona zwracać wartość Tak, począwszy od daty 2014-06-02:

Data	DataPrzyszła
2014-05-30	Nie
2014-05-31	Nie
2014-06-01	Nie
2014-06-02	Tak
2014-06-03	Tak

Rysunek 3.37.

Kolumna DzieńTygodnia

Masz już kolumnę NrDniaTygodnia zawierającą liczby od 1 do 7. Ale teraz chcesz mieć nazwy tych dni. Tak wygląda niesamowicie prosta formuła realizująca to zadanie:

```
=FORMAT([Data]; "dddd")
```

Efekt jest wyjątkowo widowiskowy:

Data	DzieńTygodnia
2014-01-06	poniedziałek
2014-01-07	wtorek
2014-01-08	środa
2014-01-09	czwartek
2014-01-10	piątek
2014-01-11	sobota
2014-01-12	niedziela

Rysunek 3.38.

Kolumna RokMiesiąc

Teraz, gdy poznałeś podstawowe metody podejmowania wyzwań związanych z kalendarzem, zwiększymy trochę tempo i będziemy pokazywać same formuły i wyniki ich działania. Oto formuła RokMiesiąc i jej wynik:

```
=[Rok] & FORMAT([NrMiesiąca]; "00")
```

Data	NrMiesiąca	RokMiesiąc
2014-01-31	1	201401
2014-02-01	2	201402

Rysunek 3.39.

Kolumna Dzień roboczy/weekend

Tak wygląda formuła Dzień roboczy/weekend i jej wynik:

```
=SWITCH([NrDniaTygodnia]; 6; "Weekend"; 7; "Weekend"; "Roboczy")
```

Data	NrDniaTygodnia	DzieńTygodnia	Dzień roboczy/weekend
2014-01-06	1	poniedziałek	Roboczy
2014-01-07	2	wtorek	Roboczy
2014-01-08	3	środa	Roboczy
2014-01-09	4	czwartek	Roboczy
2014-01-10	5	piątek	Roboczy
2014-01-11	6	sobota	Weekend
2014-01-12	7	niedziela	Weekend

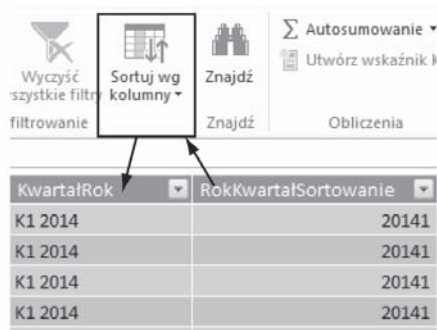
Rysunek 3.40.

Kolumna RokKwartałSortowanie

Tak wygląda formuła RokKwartałSortowanie i jej wynik:

=[Rok] * 10 + [NrKwartału]

Kolumna użyta do sortowania kolumny KwartałRok



Rysunek 3.41.

Wyjątkowo często występującym problemem jest brak możliwości sortowania danych w przedziałach czasu dłuższych niż jeden dzień (tydzień, miesiąc, kwartał itp.) i „zespawanych” z rokiem (na przykład *K2 2014*) według innej kolumny, zazwyczaj dostępnej w tabeli Kalendarz.

Uważamy, że powinna istnieć możliwość sortowania według daty, ale dodatek Power Pivot na to nie pozwala. I oczywiście nie można sortować według kolumny NrKwartału, ponieważ nie będą wtedy brane pod uwagę różne lata.

Dlatego często musimy sami znaleźć rozwiązanie: po prostu mnożymy rok przez 10, aby „zrobić miejsce” na jedną cyfrę oznaczającą numer kwartału i dodać ją. Czy to istotne, że wynik 20142 jest bezużyteczny i nigdzie nie zostanie użyty w tabeli przestawnej? Ważne jest, że dzięki niemu dane są sortowane prawidłowo. Możemy ukryć tę kolumnę przed narzędziami klienta i zapomnieć o niej.

UWAGA

Tę samą technikę możesz zastosować w kolumnie RokMiesiąc z tym wyjątkiem, że rok trzeba przemnożyć przez 100, aby zrobić miejsce na dwie cyfry z numerem miesiąca.

Wyzwania związane z niestandardowym kalendarzem

Wszystkie poprzednie przykłady dotyczyły tzw. „standardowego” kalendarza, czyli takiego, w którym miesiąc ma dokładnie tyle samo dni, ile jest w kalendarzu na ścianie.

W wielu branżach, również w detalicznej, standardowy kalendarz jest niemal bezużyteczny. We wszelkich porównaniach okresów wykorzystywane są niestandardowe kalendarze zdefiniowane przez użytkownika. Bardzo często takie kalendarze tworzone są według schematu 445, w którym pierwszy miesiąc ma cztery tygodnie, drugi też cztery tygodnie, a trzeci pięć tygodni (później ten schemat jest powtarzany).

Nad niektórymi kolumnami obliczeniowymi wymaganymi zazwyczaj w takich tabelach można się nieźle napocić, więc omawiając tę sztuczkę, podamy kilka przykładów.

Określenie numeru dnia w danym okresie

Żałujemy, że masz tabelę kalendarza typu 445 o nazwie `KalendarzHybrydowy`. Wśród jego kolumn są dwie o nazwach `Data` i `445MiesiącID`:

Data	445MiesiącID
2014-01-25	1
2014-01-26	1
2014-01-27	1
2014-01-28	1
2014-01-29	2
2014-01-30	2
2014-01-31	2
2014-02-01	2

Rysunek 3.42.

Kolumna `445MiesiącID` jest podstawą dla wszystkich wyliczeń na poziomie miesiący. Aby cofnąć się do poprzedniego okresu i sprawdzić ubiegły miesiąc, wystarczy wykonać proste odejmowanie -1 z użyciem tej kolumny:

Ale tworząc pole obliczeniowe porównujące okresy czasu w kalendarzu typu 445, szybko stwierdzisz, że musisz znać numer dnia bieżącego okresu, „w którym jesteś”. Ten numer powinien przyjmować wartości od 1 do 28, w następnym miesiącu ponownie od 1 do 28, a w kolejnym (5-tygodniowym) od 1 do 35. Potem schemat powinien się powtarzać. Trzeba by się podrapać po głowie.

Aby oszczędzić Twoją czuprynę, poniżej przedstawiamy gotową formułę:

```
=INT([Data] -
    CALCULATE(FIRSTDATE(KalendarzHybrydowy[Data]);
        FILTER(KalendarzHybrydowy;
            KalendarzHybrydowy[445MiesiącID] =
                EARLIER(KalendarzHybrydowy[445MiesiącID])
        )
    ) + 1
)
```

A tak wygląda jej wynik:

Data	445MiesiącID	DzieńOkresu
2014-01-25		1
2014-01-26		1
2014-01-27		1
2014-01-28		1
2014-01-29		2
2014-01-30		2
2014-01-31		2
2014-02-01		2

Rysunek 3.43.

Określenie numeru tygodnia w przedziale czasu

Inne często spotykane pytanie brzmi: „Jaki jest numer tygodnia w bieżącym okresie?”. Numer powinien przybierać wartości od 1 do 4, potem od 1 do 5, a następnie od 1 do 5.

Ten problem jest łatwiejszy niż określenie numeru dnia, ale i tak znów musisz się podrapać po głowie, jeżeli ktoś patrzy, co robisz. Poniżej jest formuła:

```
=CEILING([DzieńOkresu]/7;1)
```

Stale rosnący i „gładki” numer tygodnia

Pola obliczeniowe typu „Poprzedni tydzień” szybko przekonają Cię, że potrzebna jest kolumna taka jak na poniższym rysunku, w której wartości zwiększają się o 1 z każdym kolejnym tygodniem, ale nigdy nie „resetują się” (nie wracają do 1) z początkiem nowego roku. Jak również nigdy nie ma przerw w wartościach tej kolumny. (Innymi słowy, chcesz, aby pierwszy tydzień następnego roku miał numer dokładnie o 1 większy od numeru ostatniego tygodnia poprzedniego roku).

Data	NrTygodnia
2014-12-15	51
2014-12-16	51
2014-12-17	52
2014-12-18	52
2014-12-19	52
2014-12-20	52
2014-12-21	52
2014-12-22	52
2014-12-23	52
2014-12-24	53
2014-12-25	53
2014-12-26	53

Rysunek 3.44.

Formuła jest niespodziewanie trudna w utworzeniu, nie dlatego, że jest to dodatek Power Pivot, ale ze względu na złożoność obliczeń arytmetycznych, od których mózg może się przegrzać.

Oto co musisz wpisać:

```
=ROUNDDOWN(((445MiesiącID)-1)/3;0)*13 + MOD([445MiesiącID]-1;3)*4 + [IDTygodnia]
```

Gdybyśmy dalej podążali tym totem, wyczerpalibyśmy samych siebie i czytelników. Dlatego zmieniamy bieg i w rozdziale 4. pobawimy się trochę przenośnymi formułami.

Skorowidz

A

adres URL
 obrazu, 224
 w sieci Web, 224
aktualizacja danych, 28
analiza
 kampanii reklamowych, 151, 159
 zachowania klientów, 171
animowane wykresy, 242
anulowanie
 przestawienia kolumn, 209, 211
 przestawienia tabeli, 207

B

baza danych OLAP, 49
błąd, 110, 132, 186

C

czas trwania sceny, 249
czcionka
 Bookshelf Symbol 7, 56
 Marlett, 57
 MS Reference Specialty, 57
 Symbol, 56
 Webdings, 56
 Wingdings, 55, 59
czcionki symboliczne, 55

D

dane
 lokalne, 246
 o populacji, 176

zakodowane geograficznie, 244
znormalizowane, 29

data

 końcowa, 111, 161, 165
 ostatniej aktualizacji, 25
 otwarcia sklepów, 146
 początkowa, 112, 161, 165

dodanie nowej kolumny, 75

dodatek

 Power Map, 243
 Power Pivot, 25
 Power Query, 191
 Power View, 223

dynamiczne

 raporty, 77
 normalizowanie wykresów, 33

E

edytor zaawansowany, 219
edytowanie reguły formatowania, 90, 91
efekty, 252
elementy Power View, 225
etykiety, 42, 189
etykiety ekranowe, 40

F

filtr, 234–236
 główny, 138
 zaawansowany, 236
filtrowanie
 danych sprzedaży, 145
 krzyżowe, 54
 według pól z datami, 162
filtry raportów, 50, 51

format

CSV, 192

daty, 27

DMS, 244

formatowanie warunkowe, 37, 85, 96

formuła współczynnika, 155

formuły, 31

przenośne, 130

statyczne, 197

fragmentatory, 49, 53

czcionki symboliczne, 55

data końcowa, 159, 165

data początkowa, 159, 165

formatowanie warunkowe, 85

jawne, 230

kolumna z etykietami, 67

niejawne, 231

niestandardowy styl, 62

odczytywanie wyboru, 79

pola obliczeniowe, 72

przestarzałe, 65

sterowanie formatowaniem warunkowym, 87

sterowanie średnią ruchomą, 181

stosowanie makr, 64

sztuczne tabele, 71

tworzenie, 60

ukrywanie nagłówka, 63

umieszczanie symboli, 59

zmiana czcionki, 60, 64

funkcja

ALL(), 93

ALLSELECTED(), 35

ALLVALUES(), 35

AVERAGE(), 156

AVERAGEX(), 32

BLANK(), 188

CALCULATE(), 32, 111, 143, 174

CONCATENATEX(), 117

CONTAINSX(), 116–119

DATESBETWEEN(), 162

EARLIER(), 15, 101

FILTER(), 102, 105, 113, 121, 162

FIND(), 118

FIRSTDATE(), 147

FIRSTNONBLANK(), 120

FORMAT(), 67, 100

IF(), 39, 76, 164

JEŻELI.BŁĄD(), 95

LASTDATE(), 26, 186

LOOKUPVALUE(), 107–110, 113

MAX(), 76

MIN(), 79

MINX(), 40

PODAJ.POZYCJĘ(), 82

POZYCJA(), 81

POZYCJA.ŚR(), 81

RANDBETWEEN(), 146

RANKX(), 39, 77, 80, 83

RELATED(), 100, 108, 113, 116

RELATEDTABLE(), 116

SEARCH(), 121

SUM(), 105

SUMMARIZE(), 175

SUMX(), 118, 176, 178, 190

SWITCH(), 77, 79

TEXT(), 100

TOPN(), 93, 113, 174, 175

UPPER(), 118

USZEREGOWANY.ELEMENT.MODUŁU(),
84, 95, 96

VALUES(), 35, 37, 83

WARTOŚĆ.MODUŁU(), 27, 84

WYSZUKAJ.PIONOWO(), 100, 119

ZESTAW.MODUŁÓW(), 84, 94, 95

funkcje X, 117

funkcjonalność Kafelki wg, 239

G

główny filtr, 138

H

hierarchia danych, 229

hiperłącze, 41

I

ikona Uogólnij, 230

ikony, 238

implementacja pola obliczeniowego, 145

import

tabeli, 215

współczynnika korygujących, 153

zakodowanych danych, 244

złączonego pliku CSV, 195

informacje

o aktualizacji, 25

o ważnych lokalizacjach, 244

inicjalizacja fragmentatorów, 53

integracja danych, 133

J

język formuł M, 217

K

kafelki, 236, 239
 kalendarz, 69, 121
 kalendarz niestandardowy, 124
 kampania reklamowa, 151
 kierunek filtrowania danych, 142
 klawisze strzałek, 100
 klient

- nowy, 166, 170
- powracający, 166

 kodowanie geograficzne, 243
 kojarzenie wartości, 117
 kolejność sortowania, 105
 kolumna

- DataPrzyszła, 122
- Dzień roboczy/weekend, 123
- DzieńTygodnia, 123
- obliczeniowa, 183
- RokKwartałSortowanie, 124
- RokMiesiąc, 123
- UkrytaKolejnośćSortowania, 73

 kolumny

- obliczeniowe, 99
- z etykietami, 67

 konflikt wartości, 110
 kontrolki elementu, 226
 konwertowanie współrzędnych, 245
 korekta błędu, 91, 186
 krzyżowe przeszukiwanie tabel, 107

L

limit wierszy, 192
 linia na mapie, 248

Ł

łączenie

- arkuszy w tabelę, 191
- danych, 202
- plików CSV, 195, 196
- skoroszytów w tabelę, 191
- tabel, 204

M

makro

- do zapisywania skoroszytów, 196
- do zmiany czcionek, 64
- Fazzy, 196
- MakroEtykietek(), 42, 43

 mapa, 240
 mapa terenu, 247
 metody

- tworzenia tabel przeszukiwanych, 212
- wyświetlania czasu, 248

 model danych, 179
 modelowanie danych, 127

N

nadrzędne tabele filtrujące, 140
 nagłówek, 192, 208
 nagłówek fragmentatora, 63
 napisy w raporcie, 83
 narzędzie Tablica znaków, 57, 58
 nazwa pola, 150
 nazwane zestawy danych, 44
 niestandardowa kolumna, 197
 normalizowanie

- danych, 29, 31
- pola obliczeniowego, 33

 numer

- dnia, 125
- tygodnia, 126

O

obiekt ActiveSlider, 64
 obliczanie

- daty, 149
- liczby transakcji, 144
- sprzedży, 143
- współczynnika korygującego, 152

 obsługa wyczyszczonego wyboru, 96
 odczytanie

- drugiego pliku CSV, 201
- pierwszego pliku CSV, 197
- trzeciego pliku CSV, 201

 odłączone tabele, 86
 odsetek powracających klientów, 166, 169
 odświeżanie

- danych, 206
- komórki, 28

- odwołania
 - do poprzedniego wiersza, 104
 - do wierszy, 105
- okno
 - Pole obliczeniowe, 157
 - z wierszem poleceń, 193
- okresowe fluktuacje, 152
- okresowo dopasowywane pola, 151
- okresowość sprzedaży, 157, 158
- określenie
 - numeru dnia, 125
 - numeru tygodnia, 126
 - progu, 92, 93
- opcja
 - Liczność, 233
 - Publiczne, 203
 - Załaduj do modelu danych, 200
- opcje
 - formatowania, 228
 - fragmentatora, 94
- operator ||, 163
- opisowe nazwy, 150

P

- pakiet Office 2013 Pro Plus, 243
- panel Zapytania skoroszytu, 202
- pierwszych N wartości, 77
- pliki
 - CSV, 193, 194
 - POI, 244
 - XLSX, 208
- POI, 244
- pola
 - obliczeniowe, 26, 99
 - programu Power View, 227, 229
 - z dzienną sprzedażą, 163
- pole
 - Aktywni klienci, 168
 - Data końcowa, 162, 165
 - Data początkowa, 162, 165
 - DziennaSprzedaż, 164
 - EtykietaFragmentatora, 68
 - Kalendarz[NrTygodnia], 31
 - Kanał, 237
 - KolejnośćSortowania, 72
 - Liczba klientów, 168
 - Liczba sprzedanych sztuk, 187, 188
 - Niesprzedany, 187
 - obliczeniowe, 161
 - OstatnieOdświeżenie, 26
 - problem Sarah, 35
 - Produkt spadkowy, 37

- Ranking, 40
- Ranking zielony, 93
- Region, 237
- selektywne, 185
- SortowanieAlfabetyczne, 76
- SumaSprzedaży, 162
- Średni wiek, 176
- tekstowe, 252
- TrendSprzedaży, 36
- UkrytaKolejnośćSortowania, 73
- UwzględnićKlienta, 82
- WybranyZyskProc, 93
- złożone z dwóch poprzednich, 174
- Zmienna średnia ruchoma, 185
- Zysk dla rankingu zielonego, 93
- połączenie
 - Anuluj przestawienie kolumn, 210
 - Dołącz, 202
 - Duplikuj kolumnę, 220
 - Hiperłącze, 41
 - Odśwież, 206
 - Sortuj, 70
 - Usuń duplikaty, 213, 221
 - Usuń kolumny, 213, 215
 - Wklej i dołącz, 192
 - Załaduj do, 200, 205
 - Zamknij i załaduj do, 214
- porównanie fragmentatorów, 232
- poruszanie się po mapach, 246
- Power Map, 243
- Power Pivot, 25
- Power Query, 191
- Power View, 223
- powiększanie widoku, 252
- precyzyjne dane lokalne, 246
- problem Sarah, 35, 36
- prywatność plików, 203
- przeciąganie pola, 227
- przejścia, 252
- przenośne formuły, 47
- przenośność, 130
- przeszukiwanie danych, 111, 112
- przewodnik, 250

R

- rankingi, 38
- raport, 25
- raporty interaktywne, 223
- reguła ze skalą kolorów, 88
- reguły formatowania, 90, 96
- relacja wiele-do-wielu, 140–142

relacje, 118
 relacje pomiędzy tabelami, 136
 ręczne odświeżanie danych, 207
 rodzaje filtrów, 234
 rozszerzanie reguł, 138
 rozszerzone znaki, 57
 rysowanie linii, 248

S

selektywne pole obliczeniowe, 185
 sortowanie, 70, 72, 74
 sprzedaż, 162

- dzienna, 163
- obliczanie, 143, 163
- okresowość, 157, 158
- w tych samych sklepach, 146

 sterowanie formatowaniem warunkowym, 87
 styl fragmentatora, 62
 style dla czcionek, 63
 suma ruchoma, 180
 sumowanie tabel danych, 114
 sumy

- częściowe, 38
- pośrednie, 101

 szybkość wzrostu populacji, 176

Ś

średnia, 32

- arytmetyczna, 177
- ruchoma, 178
 - korekta, 181
 - pierwsze podejście, 180
 - sterowana, 181
- ważona, 176

T

tabela

- Kalendarz, 68, 162, 166, 216, 220
- Kampanie, 164
- Klienci, 166
- z profilem sprzedaży, 152
- Ranking, 173, 174
- Sklepy, 144
- Sprzedaż, 166
- Współczynniki korygujące, 155, 159

 tabele

- domyślne, 224
- filtrujące, 138, 139

przestawne, 44, 132, 156, 172

- asymetryczne, 44
- tradycyjne, 48

 przeszukiwane, 212
 Tablica znaków, 57
 techniki wizualizacji danych, 25
 tekstowe pola obliczeniowe, 74
 test, 149
 test A/B, 159
 testowanie odświeżania danych, 206
 tworzenie

- animowanych wykresów, 242
- etykietek, 42
- formuł M, 217
- fragmentatora jawnego, 231
- fragmentatora niejawnego, 231
- fragmentatorów, 60
- map, 240
- nowego elementu, 226, 227
- odłączonych fragmentatorów, 79
- pól obliczeniowych, 72, 147
- przewodnika, 250
- reguły, 88
- relacji, 154
- relacji pomiędzy tabelami, 136
- selektywnego pola obliczeniowego, 185
- sztucznych tabel, 71
- tabeli kalendarza, 216
- tabeli przestawnej, 47, 152
- tabeli przeszukiwanej, 212
- wielu wykresów, 239
- własnych funkcji X, 117
- współczynnika okresowości, 158
- wykresów, 33

 typ Data, 160
 tytuł wykresu, 186

U

układ trzech tabel, 135
 ukryte kontrolki elementu, 226
 ukrywanie kolumny, 73
 unikatowe nazwy klientów, 83
 usługa Excel Services, 49
 ustawianie wartości filtra, 236
 usuwanie

- duplikatów, 221
- wiersza nagłówka, 192

 użycie

- fragmentatora, 147
- funkcji
 - RANKX(), 80

użycie

USZEREGOWANY.ELEMENT.MODUŁU(),
95

ZESTAW.MODUŁÓW(), 94

odłączonych tabel, 86

opcji fragmentatora, 94

poła obliczeniowego, 27

trzeciej tabeli, 138

wiersza poleceń, 207

W

wartości filtru, 236

wielokrotność pionowa, 239

wiersz, 189

nagłówka, 192, 208

poleceń, 193

wizualizacja danych, 25

wprowadzanie formuł M, 219

wskazywanie kolumny, 100

wskaźnik biznesowy, 25

współczynnik

korygujący, 152, 153

korygujący okresowy, 155

okresowości sprzedaży, 158

wstawienie raportu, 224

wybór ikony, 238

wydobywanie

rankingów, 38

wyjatków, 35

wyjątki, 35, 38

wykorzystanie wiersza nagłówka, 208

wykres, 227

wykres punktowy XY, 242

wykresy

dynamicznie normalizowane, 33

pól obliczeniowych według, 29

wyszukiwanie

daty największej sprzedaży, 172

największej liczby, 173

wspólnych wartości, 116

wyświetlanie

czasu, 248

danych, 249

mapy terenu, 247

poła obliczeniowego, 27

zdjęć satelitarnych, 247

Z

zagnieżdżanie funkcji, 40

zakładka Komunikat wejściowy, 40

zdjęcia satelitarne, 247

zliczanie

nowych klientów, 170

rzeczy, 187

unikatowych wartości, 233

zmiana

czcionki fragmentatora, 60

etykiety czasu, 247

formatu DMS, 244

tabeli w wykres, 227

znaki, 57

znaki Webdings, 59

zwielokrotnienie pionowe, 239

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



- 1. ZAREJESTRUJ SIĘ**
- 2. PREZENTUJ KSIĄZKI**
- 3. ZBIERAJ PROWIZJĘ**

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

Power Pivot dla Excela

Jeżeli myślisz, że poznałeś już na wylot wszystkie funkcje arkusza kalkulacyjnego Excel, to jesteś w błędzie! Zakres możliwości tego narzędzia można poszerzać dzięki różnorodnym dodatkom. Jednym z nich jest PowerPivot — biblioteka pozwalająca na analizę ogromnych zbiorów danych. Jeżeli chcesz w pełni wykorzystać jej potencjał, to masz w rękach doskonałe źródło informacji!

W trakcie lektury kolejnych rozdziałów nauczysz się korzystać z fragmentatorów, poznasz zaawansowane funkcje wbudowane, a także metody analizy kampanii reklamowych czy wyników testów A/B. Ponadto dowiesz się, jak określić odsetek powracających klientów i gości na stronie WWW, oraz zliczysz zdarzenia, które potencjalnie mogły zajść, ale do nich nie doszło. W tej wyjątkowej książce znajdziesz również bezcenne informacje na temat zastosowania narzędzi Power Query, Power View oraz Power Map. Sięgnij po ten przewodnik i poznaj możliwości Excela w zakresie Business Intelligence!

Dzięki tej książce:

- poznasz zastosowanie fragmentatorów
- przeprowadzisz zaawansowane analizy ogromnych zbiorów danych
- przeanalizujesz wyniki testów A/B
- poznasz możliwości narzędzi Power Query, Power View oraz Power Map
- użyjesz Excela do Business Intelligence

Wykorzystaj program Excel w Business Intelligence!

Rob Collie — były szef zespołu inżynierów w firmie Microsoft. Przedsiębiorca, konsultant, autor książek. Pasjonat Excela. Prowadzi popularną stronę poświęconą dodatkowi PowerPivot.

Bill Jelen — certyfikowany specjalista MVP. Autor 42 książek poświęconych Excelowi. Prowadzi popularną stronę MrExcel.com.

Helion

32496 numer katalogowy

księgarnia internetowa



<http://helion.pl>

zamówienia telefoniczne



0 801 339900



0 601 339900

Sprawdź najnowsze promocje:

🔗 <http://helion.pl/promocje>

📖 Książki najchętniej czytane:

🔗 <http://helion.pl/bestsellery>

📰 Zamów informacje o nowościach:

🔗 <http://helion.pl/nowosci>

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

sięgnij po **WIĘCEJ**



KOD KORZYŚCI

ISBN 978-83-283-0443-7



9 788328 304437

Informatyka w najlepszym wydaniu

cena: 49,00 zł