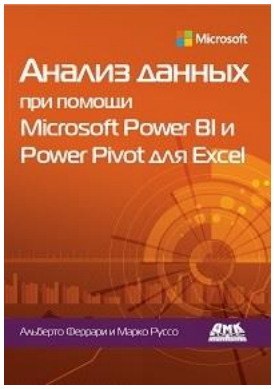
Альберто Феррари, Марко Руссо. Анализ данных при помощи Microsoft Power BI и Power Pivot для Excel

В книге представлена техника моделирования данных в Excel и Power BI. Авторы покажут, почему модель данных необходимо строить на основе нескольких таблиц, научат производить расчеты с календарем, расскажут об использовании снимков для подсчета количества товаров в наличии, о том, как работать с несколькими валютами одновременно, и подробно объяснят на примерах многие другие полезные операции. Если вы не знакомы с Power Pivot и формулами DAX, рекомендую начать с Марк Мур. Power Pivot или Роб Колли. Формулы DAX для Power Pivot.

Альберто Феррари, Марко Руссо. Анализ данных при помощи Microsoft Power BI и Power Pivot для Excel. – М.: ДМК Пресс, 2020. – 288 с.



Купить книгу в Ozon или Лабиринте

Файлы примеров можно скачать с сайта издательства ДМК Пресс. Каждому рисунку соответствует свой файл. Чтобы вы могли следить за изложением и находить правильные файлы, в конспекте я сохранил нумерация рисунков книги.

Глава 1. Введение в моделирование данных

В примерах мы будем использовать базу данных Contoso. Это вымышленная компания, торгующая электроникой по всему миру с использованием различных каналов продаж. В базе данных Contoso таблица продаж содержит 12 млн. записей.

Гранулярность данных

Лучше всего думать о гранулярности как об уровне детализации данных. Чем выше гранулярность, тем более детализированная информация будет доступна для анализа. При недостаточной детализации часть анализа будет невозможна. Например, если у вас нет столбца с указанием цвета товара, анализ по цвету будет недоступен. При избыточной детализации вы столкнетесь с проблемой агрегирования. Поясню. Представьте, что вам нужно получить средний годовой доход клиентов, покупающих определенный товар. Такая информация в таблице присутствует:

	SalesOrde...	Sales...	Sal...	Sales...	Tot...	Firs...	Las...	BirthD...	M..	G...	Yearly...
1	20070101811...	5	1	€ 103,99	€ 66,27	Emily	Miller	10.07.1962	M	F	$20 000,00
2	20070101811...	5	1	€ 103,99	€ 66,27	Deanna	Ramos	04.10.1962	S	F	$10 000,00
3	20070101811...	5	1	€ 103,99	€ 66,27	Nicole	Brown	22.03.1961	M	F	$10 000,00
4	20070101811...	6	1	€ 3,99	€ 2,54	Deanna	Ramos	04.10.1962	S	F	$10 000,00
5	20070101811...	6	1	€ 3,99	€ 2,54	Emily	Miller	10.07.1962	M	F	$20 000,00
6	20070101811...	5	1	€ 3,99	€ 2,54	Melody	Munoz	06.04.1936	M	F	$10 000,00
7	20070101811...	5	1	€ 103,99	€ 66,27	Carla	Raman	24.06.1961	M	F	$10 000,00
8	20070101811...	5	1	€ 29,46	€ 18,78	Shaun	Raji	12.03.1962	M	M	$20 000,00
9	20070101811...	5	1	€ 29,46	€ 18,78	Dennis	She	08.07.1961	S	M	$20 000,00
10	20070101811...	5	1	€ 3,99	€ 2,54	Randy	Zeng	21.08.1936	M	M	$20 000,00
11	20070101811...	6	1	€ 3,99	€ 2,54	Carla	Raman	24.06.1961	M	F	$10 000,00

Рис. 1.4. Информация о покупателях и товарах в одной таблице модели данных Power Pivot; (в файле к рис. 1.4 необходимо открыть окно Power Pivot, чтобы увидеть содержимое таблицы)

В качестве меры для расчета среднего годового дохода покупателя можно попробовать использовать формулу DAX:

AverageYearlyIncome := AVERAGE (Sales[YearlyIncome])

Меру можно вставить в сводную таблицу:

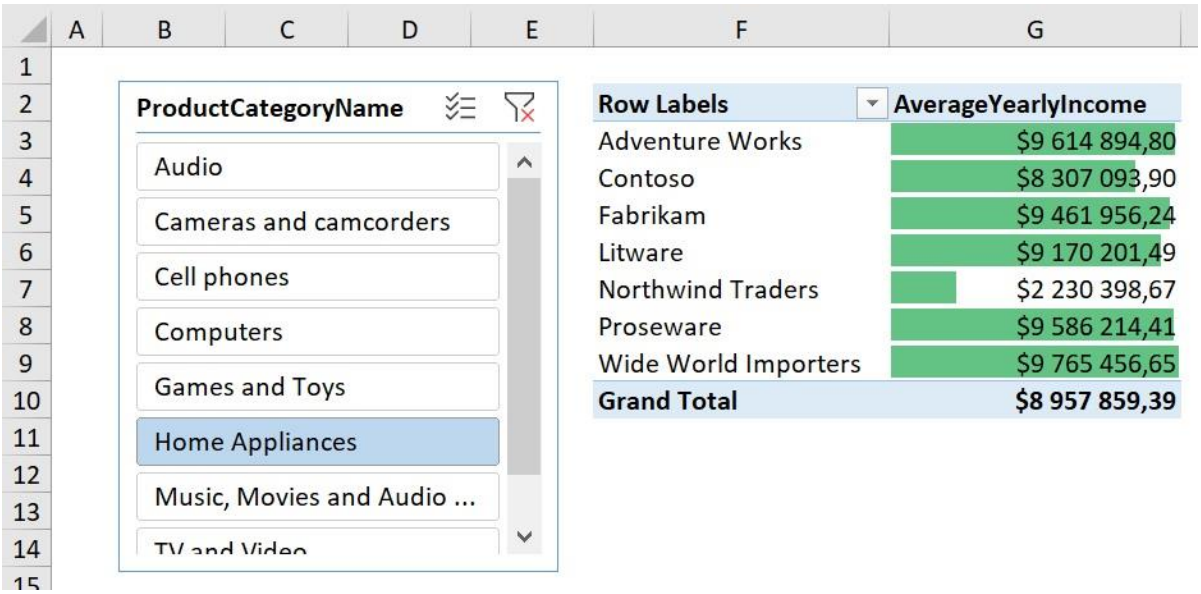


Рис. 1.5. Анализ среднего годового дохода покупателей бытовой техники

Отчет выглядит замечательно, но, к сожалению, цифры в нем не соответствуют действительности – они завышены. Вы вычислили среднее значение по таблице продаж с гранулярностью, установленной на уровне каждой трансакции, а значит, покупатели в ней повторяются. Так, если покупатель приобрел три товара, при подсчете среднего значения годовой доход для него будет учтен трижды.

Чтобы получить корректный результат, необходимо изменить гранулярность до уровня покупателя – либо путем повторной загрузки таблицы в модель данных, либо воспользовавшись более сложной формулой:

```
CorrectAverage := AVERAGEX (
        SUMMARIZE (
                Sales;
                Sales[CustomerKey];
                Sales[YearlyIncome]
        );
        Sales[YearlyIncome]
)
```

Мы применяем функцию SUMMARIZE для предварительной агрегации на уровне покупателя во временной таблице, а затем вычисляем среднее значение по YearlyIncome. Итоги правильного расчета среднего годового дохода сильно отличаются от наших прежних расчетов:

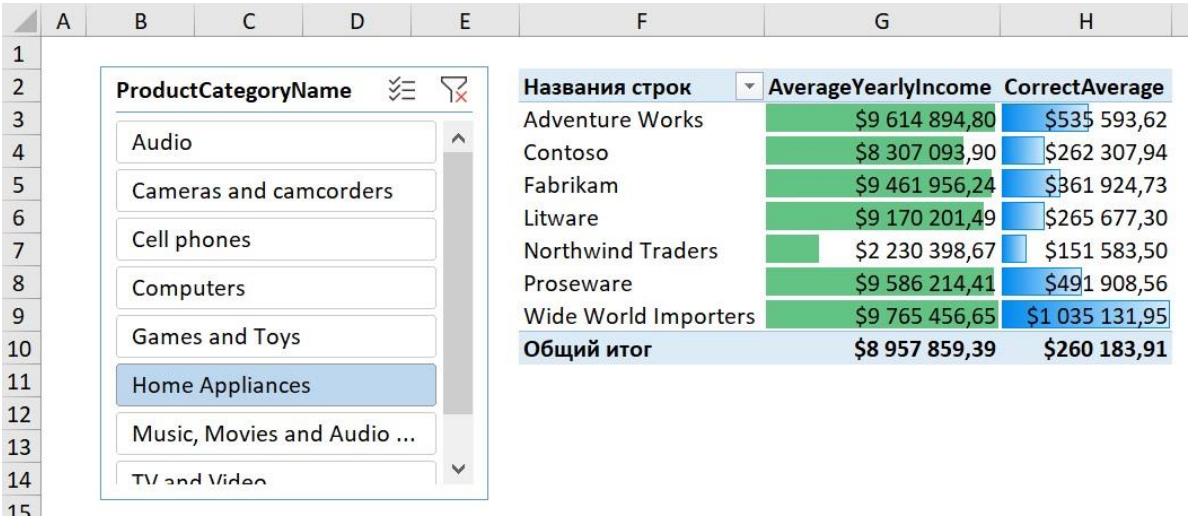


Рис. 1.6. Более сложная формула дает корректный результат

Введение в модель данных

Модель данных, состоящая из одной таблицы, таит в себе проблему в отношении определения правильного уровня гранулярности. Пользователи Excel зачастую применяют такие модели, поскольку до версии Excel 2013 строить сводные таблицы можно было только на их основании. В Excel 2013 компания Microsoft ввела понятие модели данных Excel, чтобы можно было загружать сразу несколько таблиц и создавать связи между ними.

Модель данных – это набор таблиц, объединенных связями. Модель из одной таблицы – тоже модель, хоть и не представляющая большого интереса. Именно связи, объединяющие несколько таблиц в составе единой модели данных, и делают ее столь мощной и удобной для анализа.

Две таблицы можно связать по общему полю, например, ProductKey. В таблице Product этот столбец представляет собой первичный ключ, что предполагает уникальность значений в нем и возможность идентифицировать по ним товары. В таблице Sales один и тот же товар может присутствовать во множестве строчек.

Вы можете создать связь, перетаскивая мышью поля, являющегося внешним ключом, к первичному ключу. Сделав это, вы заметите, что ни Excel, ни Power BI не используют стрелки для обозначения связей. Вместо этого на концах линии, соединяющей таблицы, вы обнаружите единичку (один) и звездочку (многие):

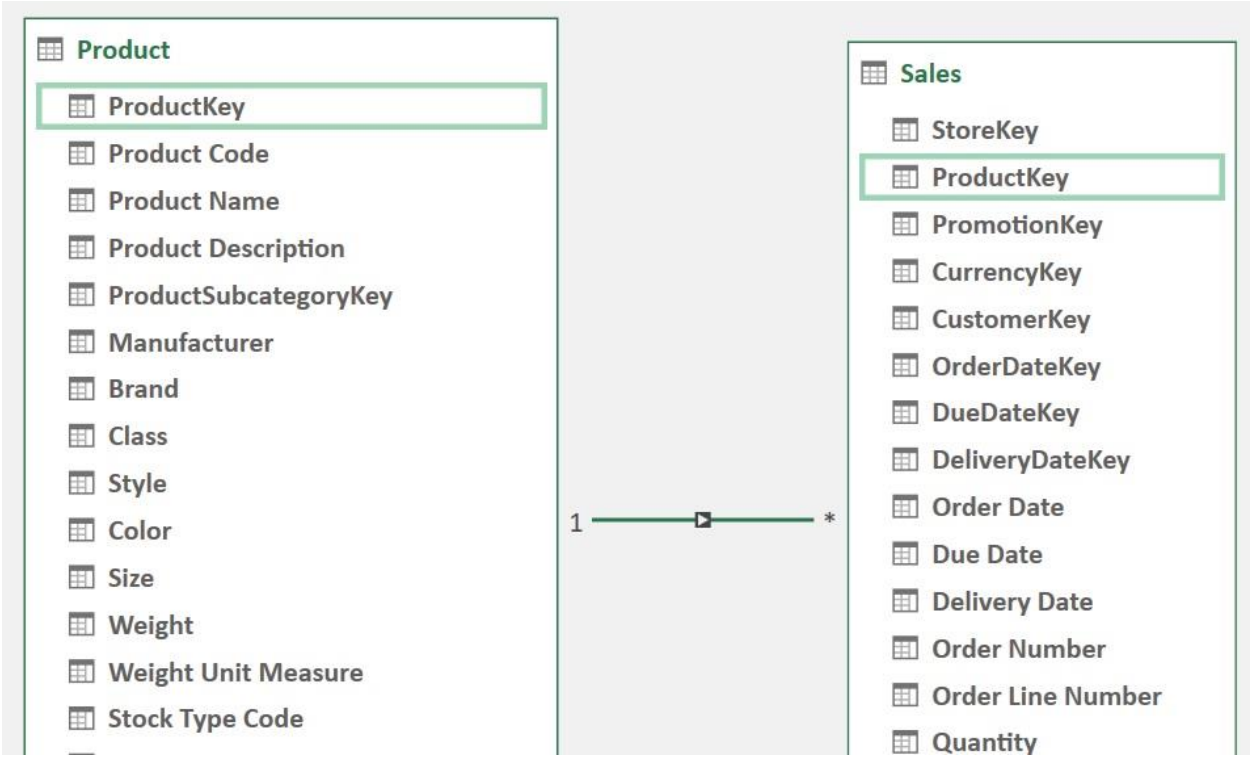


Рис. 1.8. Связь между таблицами представлена линией с индикаторами на концах («1» для одного и «звездочка» для многих)

После связывания таблиц вы можете осуществлять суммирование значений в таблице Sales, делая срезы по столбцам из таблицы Product. Например, вы можете использовать цвет товара в качестве среза при суммировании по количеству проданных товаров:

Row Labels	Sum of Quantity
Azure	60
Black	4307
Blue	985
Brown	453
Gold	155
Green	374
Grey	1551
Orange	179
Pink	600
Purple	10
Red	896
Silver	3604
Silver Grey	143
Transparent	141
White	3746
Yellow	294
Grand Total	**17498**

Рис. 1.9. После связывания таблиц вы можете осуществлять срезы по значениям одной таблицы, используя столбцы из другой

Поскольку теперь у вас в наличии есть две таблицы, то и гранулярностей будет две. В таблице Sales гранулярность установлена на уровне продажи, а в таблице Product – на уровне товара. Фактически гранулярность как концепция относится к таблице, а не к модели данных в целом. Когда в вашей модели несколько таблиц, вы должны позаботиться о том, чтобы в каждой из них была настроена гранулярность. Даже если сценарий с наличием нескольких таблиц кажется вам более сложным по сравнению с единственной таблицей, моделью данных, созданной на их основе, будет гораздо легче управлять, а гранулярность перестанет быть проблемой.

Поместив ключ товара в таблицу Sales, вы можете посредством связи извлекать все атрибуты товаров, включая категорию, цвет и многое другое. Таким образом, отсутствие необходимости хранить категорию товара в таблице продаж практически свело на нет проблему выбора уровня гранулярности. То же самое касается и других атрибутов товара: цвета, цены за единицу, наименования и всех остальных.

Нормализация

Если внимательно посмотреть на таблицу Product, можно заметить, что в ней отсутствуют категория и подкатегория. Зато есть столбец ProductSubcategoryKey, название которого говорит о том, что это внешний ключ, ссылающийся на другую таблицу (где это поле будет первичным ключом) с перечислением подкатегорий товаров.

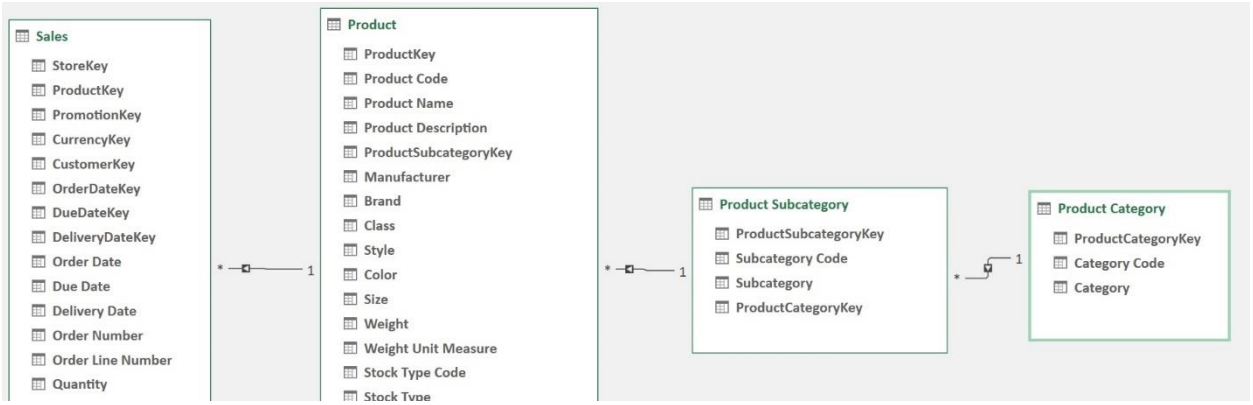


Рис. 1.10. Категории и подкатегории товаров хранятся в разных таблицах, к которым можно обратиться посредством связей

Во-первых, это позволяет сохранить место на диске из-за отсутствия необходимости хранить дублирующую информацию. Во-вторых, при необходимости изменить название категории товара вам нужно будет сделать это всего в одной строчке. Все товары автоматически подхватят новое

наименование посредством связи. У такой техники проектирования модели данных есть свое название – *нормализация*. Говорят, что атрибут таблицы нормализован, если он вынесен в отдельную таблицу, а на его место помещен ключ, ссылающийся на эту таблицу. Обратная техника, заключающаяся в хранении атрибутов в таблице, которой они принадлежат, носит название *денормализация*.

В денормализованной таблице один и тот же атрибут может встречаться множество раз, и при необходимости изменить его название вам придется корректировать все строки, содержащие этот атрибут. К примеру, в нашей модели атрибут цвета товара (Color) денормализован, а значит, значение Red будет повторяться во всех строках с красными товарами. На практике вы будете встречаться как с преимущественно нормализованными, так и с денормализованными моделями в зависимости от особенностей использования базы данных.

Модели с высокой степенью нормализации обычно используются в системах обработки транзакций в реальном времени (online transactional processing systems – OLTP). Нормализация здесь используется как способ повышения эффективности операций вставки и обновления информации, характерных для OLTP-систем. В такой системе все заказы, ссылающиеся на конкретного покупателя, будут обновлены сразу после изменения информации о нем в базе данных. Если бы атрибуты были денормализованы, то обновление адреса покупателя повлекло бы за собой изменение сотен строк в базе данных, что негативно сказалось бы на быстродействии системы.

Но во время анализа данных вы не выполняете операции вставки и обновления. Вас интересует исключительно чтение информации. И в этом случае нормализация таблиц вам ни к чему. Представьте, что вы строите сводную таблицу на основании нашей предыдущей модели данных. В этом случае список полей будет выглядеть примерно так:

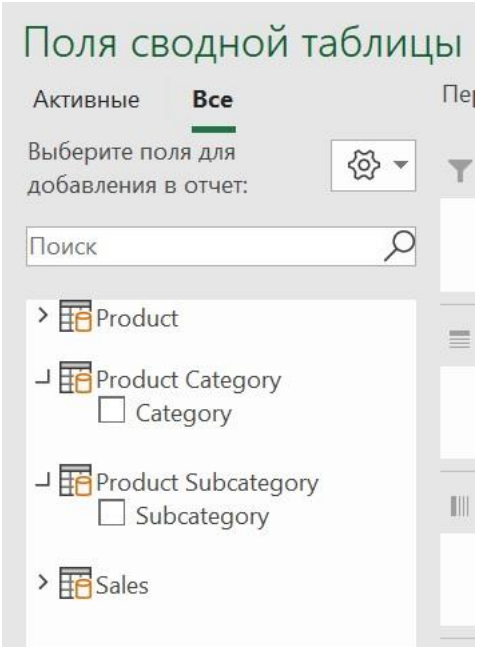


Рис. 1.11. В списке полей сводной таблицы, построенной на основании нормализованной модели данных, слишком много таблиц – легко запутаться

Информация о товарах хранится в трех таблицах, и все они представлены в списке полей сводной таблицы. Хуже того, в таблицах Product Category и Product Subcategory содержится всего по одному столбцу. В этом примере мы намеренно скрыли некоторые бесполезные столбцы вроде первичных ключей, что является хорошей практикой. (Для этого откройте Power Pivot, перейдите в таблицу *Product Category*, выделите столбцы *ProductCategoryKey* и *Category Code*, кликните правой кнопкой мыши, и выберите *Скрыть из набора клиентских средств*.

Введение в схему «звезда»

В распоряжении типичной компании вроде Contoso будет сразу несколько информационных активов, в числе которых товары, склады, сотрудники, покупатели и время. Эти активы взаимодействуют друг с другом и генерируют события. К примеру, в системе приема заявок к

активам могут относиться клиенты, заявки и время, а события генерируются в процессе изменения статуса заявок. Такое разделение делает возможным применение специальной техники моделирования данных, получившей название *схема «звезда»*. В этой схеме все таблицы подразделяются на две категории:

- **измерения**. Измерение является информационным активом: товар, покупатель, сотрудник или пациент. Измерения содержат атрибуты. К примеру, атрибутами товара являются его цвет, категория, подкатегория, производитель и цена. У пациента это имя, адрес и дата рождения;
- **факты**. Факт – это событие, в которое вовлечено несколько измерений. В базе данных Contoso, например, фактом является продажа товара. В этом событии участвуют сам товар, покупатель, дата продажи и другие измерения. В фактах также содержатся меры – числовые показатели, которые можно агрегировать при анализе состояния бизнеса. Это может быть количество или сумма проданного товара, размер скидки и прочее.

Если вы расположите на диаграмме в Power Pivot все измерения вокруг единственной таблицы фактов, то получите типичную форму звезды:

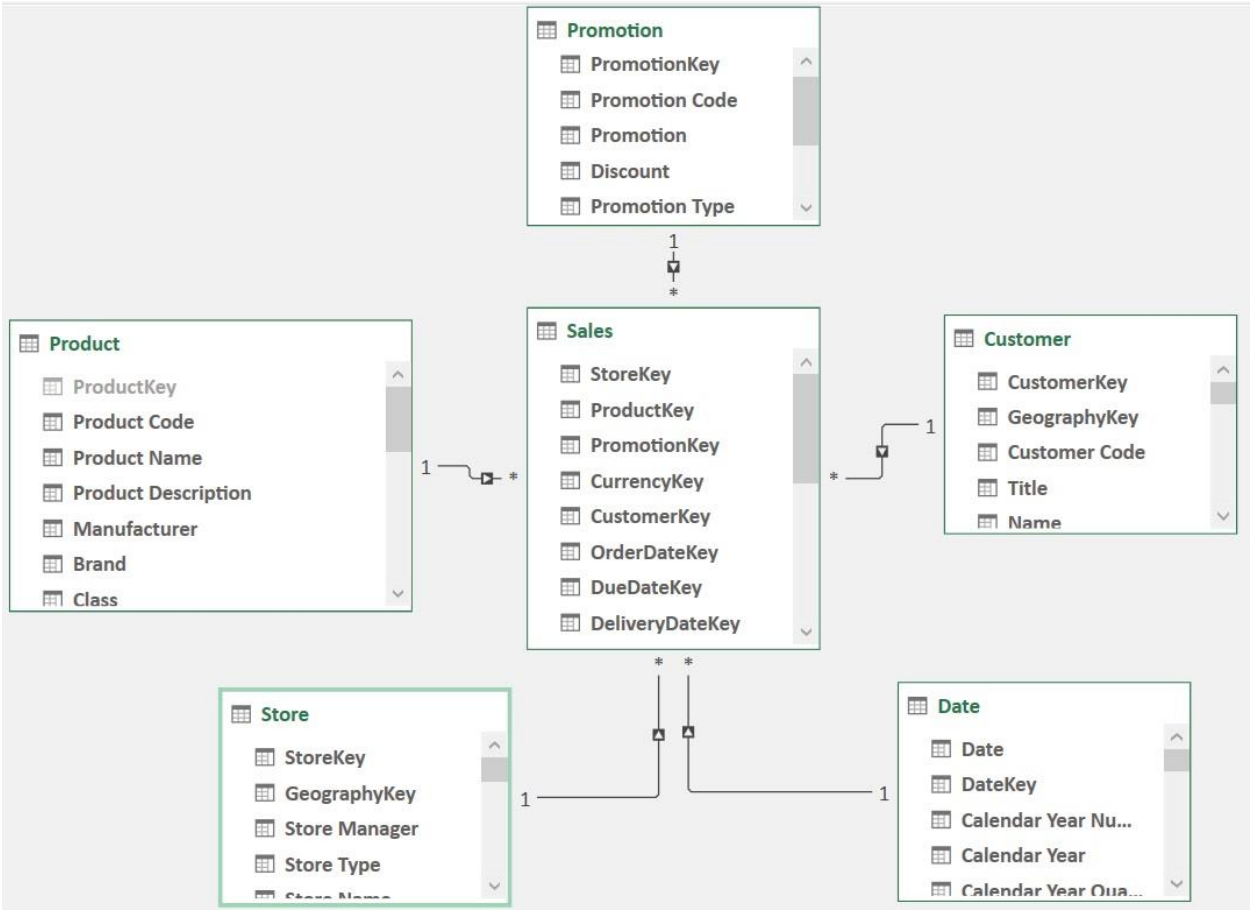


Рис. 1.12. Схема «звезда» приобретает свои очертания после расположения измерений вокруг таблицы фактов

Измерения используются для осуществления срезов данных, тогда как сама агрегация числовых показателей выполняется в таблице фактов. Таблицы измерений содержат не так много строк. Таблицы фактов, напротив, чаще всего очень объемные. Таблицы фактов могут быть объединены связями с измерениями, тогда как измерения не должны быть связаны между собой. В предыдущем примере измерения Store и Customer могут быть объединены связью с Geography. Такая связь вносит неоднозначность. По этой причине ни Excel, ни Power BI не позволят вам создать подобную модель. Как избавиться от неоднозначности? Необходимо провести денормализацию модели – перенести нужные колонки из таблицы Geography в Store и Customer, а само измерение с географией удалить из модели.

Схема «снежинка» является разновидностью «звезды» с тем исключением, что некоторые измерения не связаны с таблицей фактов напрямую. Вместо этого они объединены с ней

посредством других измерений, как на рис. 1.10. Здесь неоднозначность не возникает потому, что таблица измерений связана только с одной другой таблицей измерений, т.е., является дочерней.

Понимание важности именования объектов

Свод правил по именованию таблиц и столбцов:

- наименование измерения должно состоять только из названия актива в единственном числе: Customer или Product;
- если название актива состоит из нескольких слов, используйте для их разделения прописные буквы, не используйте пробел: ProductCategory или CountryShip;
- для имени таблицы фактов используйте название фактической операции во множественном числе: Sales или Purchases. При взгляде на модель данных вам будет представляться один покупатель (из таблицы Customer) со множеством продаж (из таблицы Sales), а природа связи «один ко многим» будет читаться естественным образом;
- избегайте использования слишком длинных и слишком коротких имен объектов;
- ключевой атрибут в измерении должен содержать название таблицы и окончание Key: CustomerKey.

Глава 2. Использование главной и подчиненной таблицы

Модель с главной (header) и подчиненной(detail) таблицами возникает, когда связью объединяются таблицы фактов.

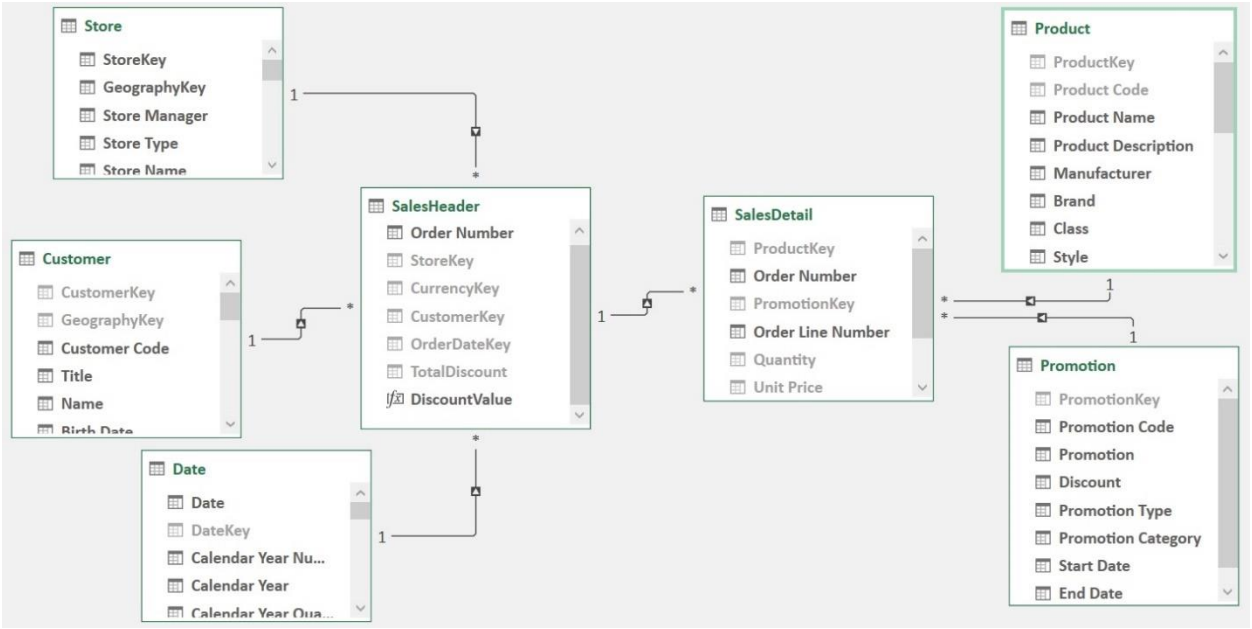
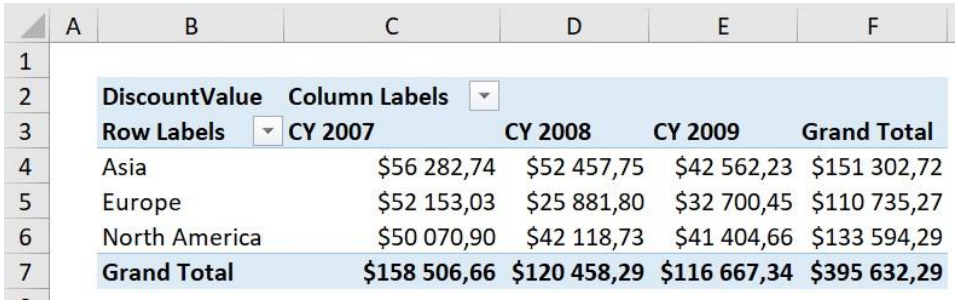


Рис. 2.1. Модель данных с главной (SalesHeader) и подчиненной (SalesDetail) таблицами

Меры вроде общего количества проданных товаров или вырученной суммы на уровне подчиненной таблицы, будут работать прекрасно. Проблемы начинаются, когда появляется необходимость агрегировать меры из главной таблицы. Вы могли бы создать меру для вычисления суммы скидки в главной таблице:

DiscountValue := SUM (SalesHeader[TotalDiscount])

Сумма скидки хранится в главной таблице, поскольку рассчитывается в момент продажи для всего документа в целом. Иными словами, скидка у нас содержится не в каждой отдельной строке заказа, а указывается единой суммой для всей операции. Именно по этой причине ей отведено место в главной таблице. Мера показывает правильные цифры, пока вы осуществляете срезы по измерениям, напрямую связанным с главной таблицей SalesHeader.



DiscountValue	Column Labels			
Row Labels	CY 2007	CY 2008	CY 2009	Grand Total
Asia	$56 282,74	$52 457,75	$42 562,23	$151 302,72
Europe	$52 153,03	$25 881,80	$32 700,45	$110 735,27
North America	$50 070,90	$42 118,73	$41 404,66	$133 594,29
Grand Total	**$158 506,66**	**$120 458,29**	**$116 667,34**	**$395 632,29**

Рис. 2.2. Вы можете делать срезы по континентам и годам

Но как только вы задействуете в срезах любой атрибут измерения, не объединенного с SalesHeader напрямую, мера сломается. Например, фильтр по цветам не работает:

DiscountValue	Column Labels			
Row Labels	CY 2007	CY 2008	CY 2009	Grand Total
Azure	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Black	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Blue	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Brown	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Gold	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Green	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Grey	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Orange	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Pink	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Purple	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Red	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Silver	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Silver Grey	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Transparent	$158 506,66	$120 458,29	$116 667,34	$395 632,29
White	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Yellow	$158 506,66	$120 458,29	$116 667,34	$395 632,29
Grand Total	**$158 506,66**	**$120 458,29**	**$116 667,34**	**$395 632,29**

Рис. 2.3. Если сделать срез по атрибутам товара, сумма скидки во всех строках будет дублироваться

Если вы используете Power BI или Analysis Services Tabular 2016 и выше, вам будет доступна двунаправленная фильтрация. Это значит, что вы сможете установить направление распространения фильтра от таблицы SalesDetail к SalesHeader. В результате фильтр, наложенный на товары или их цвет, будет распространяться как на таблицу SalesDetail, так и на SalesHeader, выбирая только интересующие вас заказы. В Excel двунаправленная фильтрация в модели недоступна.

Чтобы решить проблему, нужно повысить гранулярность подчиненной таблицы, и в итоге вовсе избавиться от главной. Скидка должна храниться не на уровне заказов главной таблицы, а на уровне отдельных строк заказов подчиненной таблицы. Эту операцию можно проделать и с остальными столбцами в главной таблице, такими как StoreKey, PromotionKey, CustomerKey. Подобный предельный уровень денормализации данных называется выравниванием (flattening), поскольку вы постепенно переходите от модели с несколькими таблицами фактов (в нашем случае двумя) к единой таблице фактов, содержащей всю информацию.

Недостатком такой архитектуры является то, что вы повышаете гранулярность для заказов, и агрегирование для заказов потребует более сложных формул. Если бы вы хотели узнать количество заказов в исходной модели данных, вы бы могли создать простую меру:

NumOfOrders := COUNTROWS (SalesHeader)

В выровненной модели данных, чтобы вычислить количество заказов, необходимо будет посчитать количество уникальных значений в столбце Order Number:

NumOfOrders := DISTINCTCOUNT (Sales[Order Number])

Глава 3. Использование множественных таблиц фактов

Рассмотрим ситуацию с несколькими не связанными друг с другом таблицами фактов. Это очень распространенный сценарий. Представьте, что компания отдельно ведет учет продаж и закупок. При этом в таблицах фактов будут как общие активы (например, товары), так и обособленные – такие как покупатели для продаж и поставщики для закупок. Проблемы могут возникнуть, когда для таблиц фактов необходимо использовать перекрестные фильтры.

Начнем с двух таблиц: Sales (продажи) и Purchases (закупки). Они полностью денормализованы:

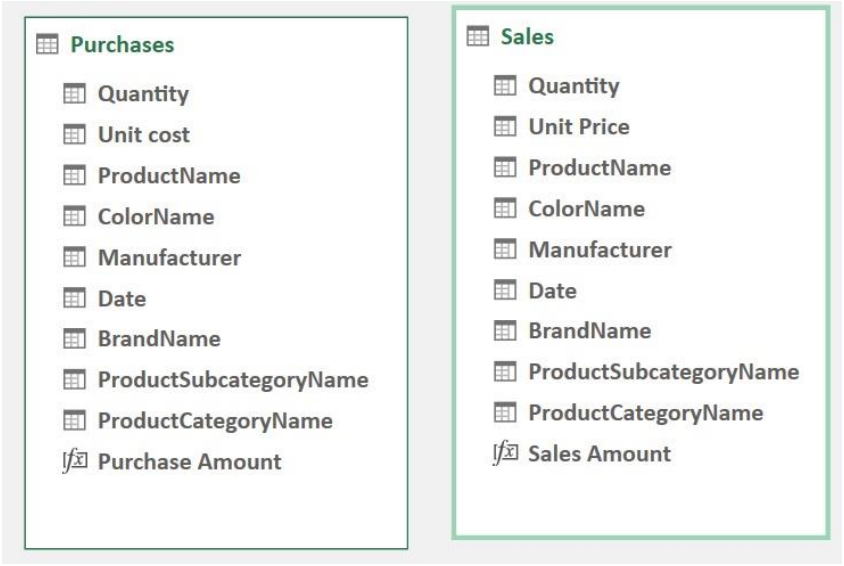


Рис. 3.1. Полностью денормализованные таблицы Sales и Purchases без связей

Определив меры…

Purchase Amount := SUMX (Purchases; Purchases[Quantity] * Purchases[Unit Cost])

Sales Amount := SUMX (Sales; Sales[Quantity] * Sales[Unit Price])

… вы не сможете анализировать продажи и закупки одновременно. Например, сводная по поставщикам… корректна для закупок, но не для продаж:

Row Labels	Purchase Amount	Sales Amount
A. Datum Corporation	2 533 963,42	30 202 685,54
Adventure Works	6 048 167,59	30 202 685,54
Contoso, Ltd	12 314 395,68	30 202 685,54
Fabrikam, Inc.	10 003 071,13	30 202 685,54
Litware, Inc.	6 377 548,93	30 202 685,54
Northwind Traders	1 713 836,80	30 202 685,54
Proseware, Inc.	5 305 305,29	30 202 685,54
Southridge Video	2 199 989,35	30 202 685,54
Tailspin Toys	646 571,47	30 202 685,54
The Phone Company	3 045 608,33	30 202 685,54
Wide World Importers	4 151 139,81	30 202 685,54
Grand Total	**54 339 597,80**	**30 202 685,54**

Рис. 3.2. Вывод мер Sales Amount и Purchase Amount в единой сводной таблице дал неверные результаты

Фильтр по производителю товаров из таблицы Purchases распространяется исключительно на эту таблицу. Он не может повлиять на таблицу Sales, поскольку между этими фактами нет связей.

Более того, вы и не сможете создать связь между ними, поскольку для этого нет подходящих столбцов.

Вы можете обойти данное ограничение путем написания сложной формулы DAX. Следующий код осуществляет фильтрацию по производителю:

```
Sales Amount Filtered :=
CALCULATE (
        [Sales Amount];
        INTERSECT ( VALUES ( Sales[BrandName] ); VALUES ( Purchases[BrandName]
        ) )
)
```

Функция INTERSECT позволяет выбрать значения из столбца Sales[BrandName], содержащиеся в текущем фильтре по Purchases [BrandName]. В результате действие фильтра по Purchases[BrandName] отразится на выборе Sales[BrandName], что, в свою очередь, позволит отфильтровать таблицу Sales:

Названия строк	Purchase Amount	Sales Amount	Sales Amount Filtered
A. Datum Corporation	2 533 963,42	30 202 685,54	1 966 583,30
Adventure Works	6 048 167,59	30 202 685,54	4 022 462,56
Contoso, Ltd	12 314 395,68	30 202 685,54	6 722 804,20
Fabrikam, Inc.	10 003 071,13	30 202 685,54	5 040 864,58
Litware, Inc.	6 377 548,93	30 202 685,54	3 425 045,95
Northwind Traders	1 713 836,80	30 202 685,54	1 205 185,61
Proseware, Inc.	5 305 305,29	30 202 685,54	2 656 623,00
Southridge Video	2 199 989,35	30 202 685,54	1 463 471,36
Tailspin Toys	646 571,47	30 202 685,54	333 143,41
The Phone Company	3 045 608,33	30 202 685,54	1 293 603,00
Wide World Importers	4 151 139,81	30 202 685,54	2 072 898,57
Общий итог	**54 339 597,80**	**30 202 685,54**	**30 202 685,54**

Рис. 3.3. Поле Sales Amount Filtered использует текущий выбор из таблицы Purchases, распространяющийся и на таблицу Sales

Чтобы упростить код, нам необходимо привести модель данных к схеме «звезда», выделив измерения Product, по которому можно будет осуществлять фильтрацию обеих таблиц. Новая схема представляет собой «звезду» в чистом виде – с двумя таблицами фактов и одним измерением:

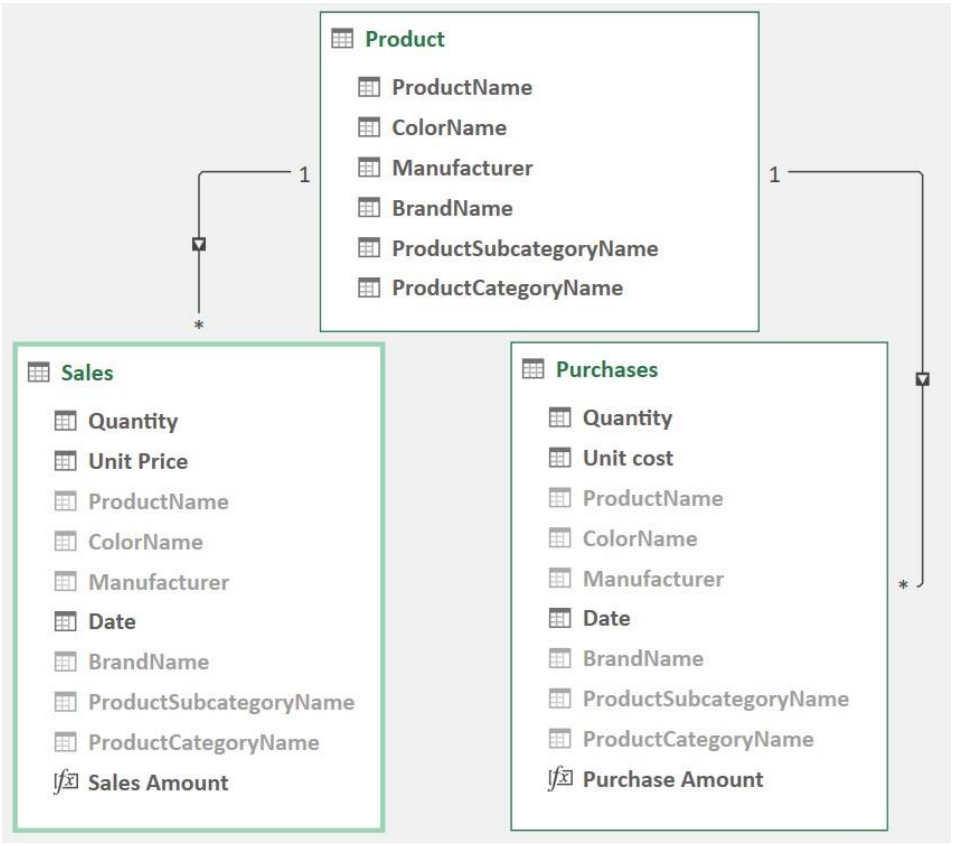


Рис. 3.4. С введением измерения Product модель данных стала проще

Понимание неоднозначности модели данных

Под *неоднозначной моделью* понимается такая модель, в которой допущено несколько путей объединения двух таблиц посредством связей. Простейшая форма неоднозначности модели возникает, когда вы пытаетесь объединить две таблицы более чем одной связью. В таком случае активна будет только одна из связей – по умолчанию та, которую вы создали первой. Остальные связи будут помечены как неактивные. На рис. 3.8 представлен пример такой модели. Из существующих трех связей между таблицами лишь одна обозначена сплошной линией, то есть активна. Оставшиеся две связи отмечены пунктиром, а значит, являются неактивными.

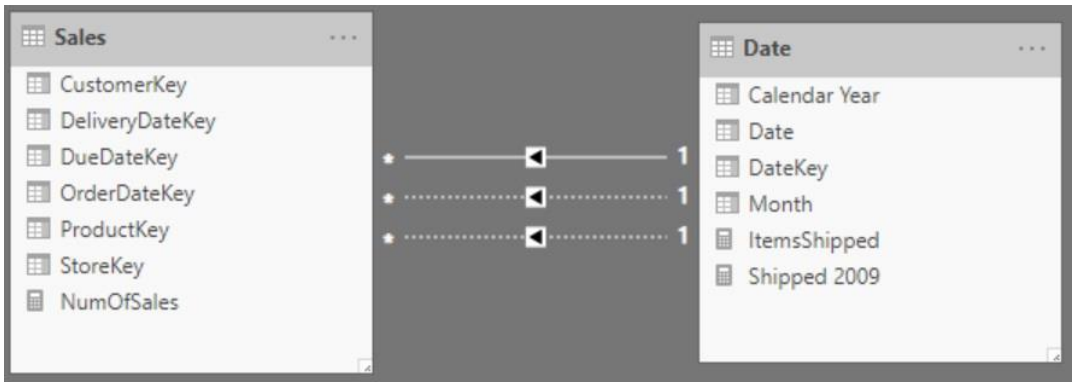


Рис. 3.8. Две таблицы не могут быть объединены более чем одной активной связью

В следующей модели присутствует два столбца с указанием на возраст:

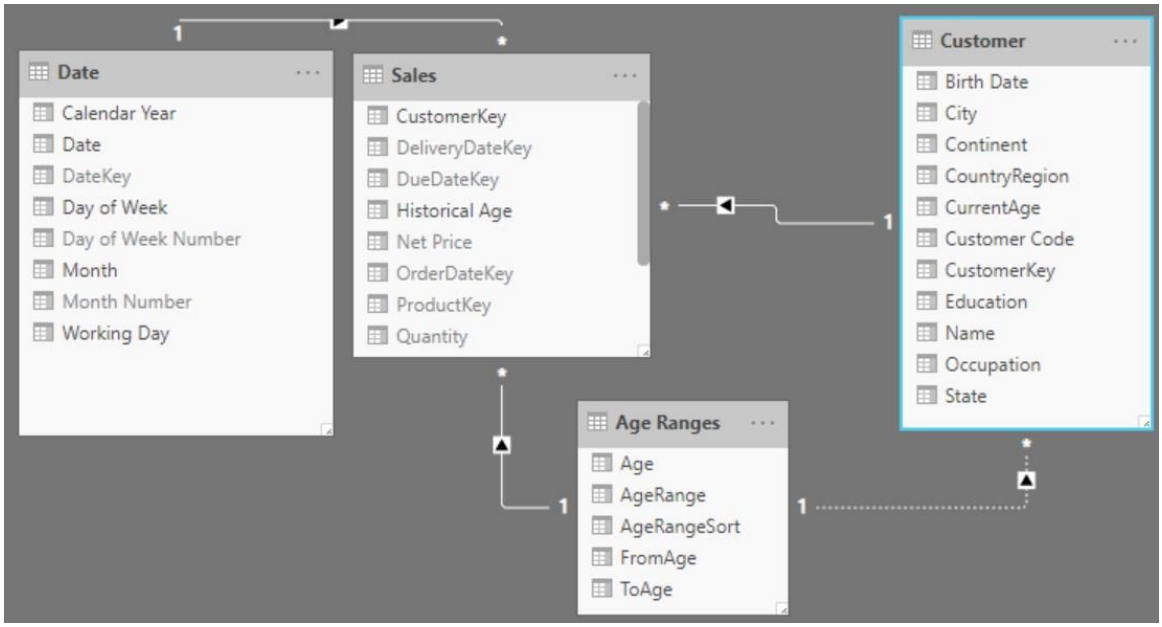


Рис. 3.9. Эта модель также неоднозначна, хотя причина этого не столь очевидна

Один из них – Historical Age – находится в таблице фактов, а второй – CurrentAge – в измерении Customer. Оба поля являются внешними ключами в своих таблицах и ссылаются на таблицу Age Ranges (диапазоны возрастов), но лишь одна из этих связей может быть активна. Другая связь деактивирована. Представьте, что строите сводную таблицу со срезом по таблице Age Ranges. Так какую информацию вы хотите получить? Во сколько лет покупатель приобрел наш товар (поле Historical Age) или сколько ему лет сейчас (поле CurrentAge)? Если бы обе связи оставались активными, системе не удалось бы однозначно ответить на этот вопрос. В результате вы должны либо решить, какую связь оставить активной, либо продублировать таблицу, вносящую неразбериху. Выбрав второй вариант, вы сможете в будущем однозначно указывать, связь с какой из двух таблиц (Current Age Ranges или Historical Age Ranges) вы имеете в виду в своих запросах.

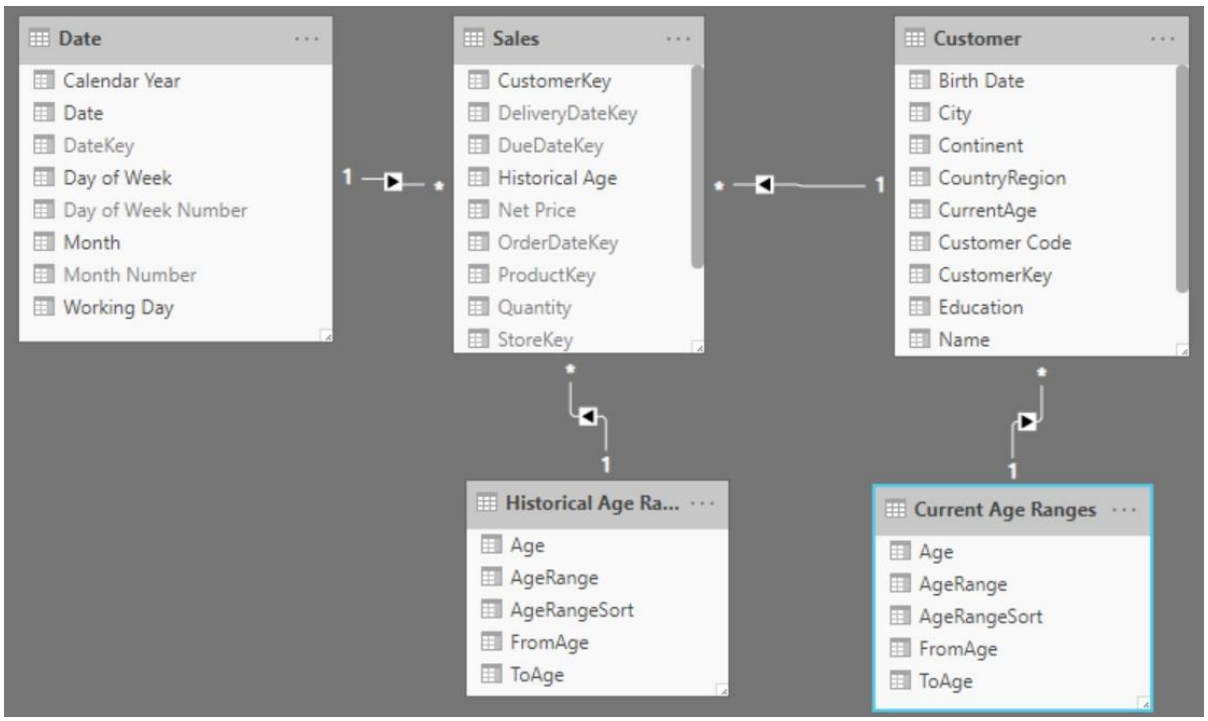


Рис. 3.10. Теперь в нашей модели две таблицы Age Ranges

Связь многие ко многим на основе таблицы-моста

Представьте, что вы получаете заказы от своих покупателей и раз в месяц выписываете счета. Один заказ может быть включен в несколько счетов, и в то же время один счет может распространяться на несколько заказов. Чтобы реализовать этот подход, модель должна содержать дополнительную таблицу с указанием номеров счетов и заказов, а также сумм.

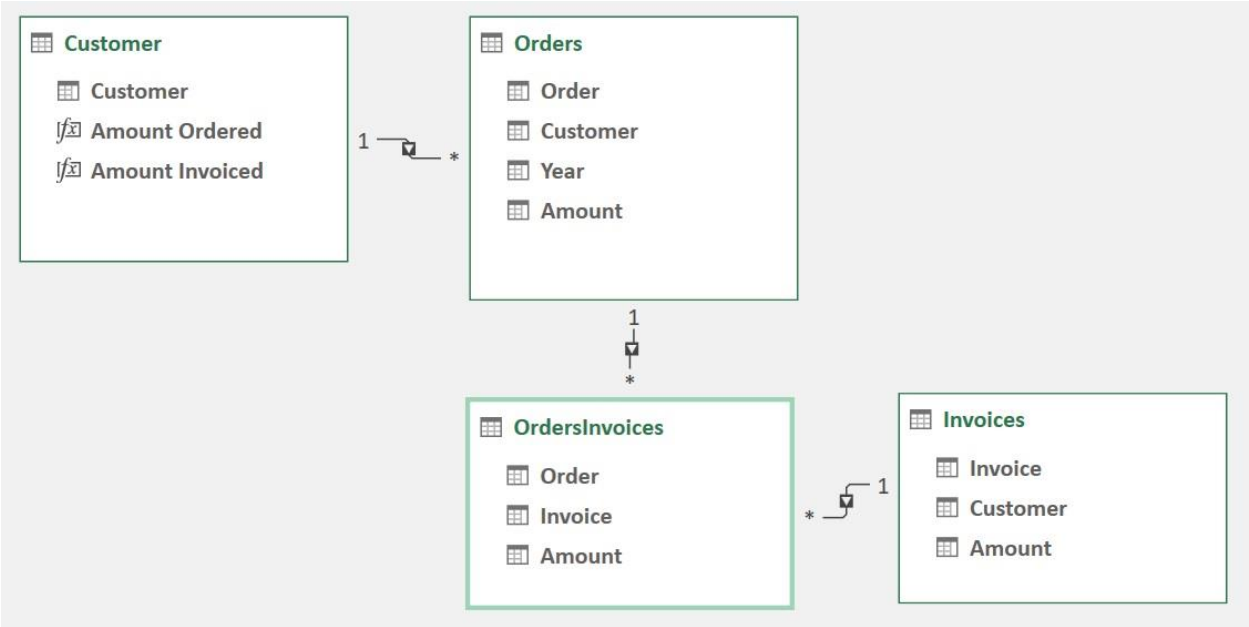


Рис. 3.20. В этой модели появилась возможность включения нескольких заказов в счет и нескольких счетов в заказ

Фактически наша модель включает в себя связь типа «многие ко многим» между таблицами заказов и счетов. Для такой связи мы использовали дополнительную таблицу OrdersInvoices. Эта таблица не является ни измерением, ни таблицей фактов. С фактом ее роднит то, что она содержит меру Amount и объединена связью с измерением Invoices. В то же время она связана и с таблицей Orders, которая сама одновременно является измерением и таблицей фактов. Технически таблицу OrdersInvoices можно назвать таблицей-мостом (bridge table), поскольку она представляет собой мост между таблицами заказов и счетов.

Теперь, когда суммы частичной оплаты заказов хранятся в промежуточной таблице, формула для расчета суммы заказов, включенных в счета, будет выглядеть следующим образом:

```
Amount Invoiced :=
CALCULATE (
        SUM ( OrdersInvoices[Amount] );
         CROSSFILTER ( OrdersInvoices[Invoice]; Invoices[Invoice]; BOTH )
)
```

Мы суммируем столбец Amount в таблице-мосте, тогда как функция CROSSFILTER включает двунаправленную фильтрацию для связи между этой таблицей и Invoices. В результате мы получим отчет, в котором отражены суммы заказов и их полное или частичное включение в счета. Желтым выделены ячейки, в которых счет оплачивает заказ не полностью:

Custome	Orde	Amount Ordere	Amount Invoiced
John	1	100,00	100,00
	4	400,00	400,00
	7	500,00	500,00
	10	100,00	100,00
	13	400,00	400,00
	16	500,00	300,00
John Total		**2 000,00**	**1 800,00**
Melanie	3	500,00	500,00
	6	500,00	500,00
	9	500,00	500,00
	12	500,00	250,00
	15	500,00	100,00
	18	500,00	150,00
Melanie Total		**3 000,00**	**2 000,00**
Paul	2	250,00	250,00
	5	1 000,00	1 000,00
	8	750,00	750,00
	11	250,00	250,00
	14	1 000,00	750,00
	17	750,00	600,00
Paul Total		**4 000,00**	**3 600,00**

Рис. 3.21. Использование таблицы-моста позволило нам получить полную картину по заказам и счетам

Глава 4. Работа с датой и временем

Чисто технически мы говорим о времени как об измерении. Однако время – это не обычное измерение. В Excel 2016 и Power BI Desktop компания Microsoft встроила *автоматическую систему* для работы с датами и временем.

Автоматическая группировка дат в Excel

При работе со сводными таблицами на основании модели данных Excel добавление столбца с датами автоматически приведет к созданию набора вспомогательных столбцов для оперирования датами и временем. Представьте, что вы начали работать с моделью, где в таблице Sales есть один столбец с датами – Order Date.

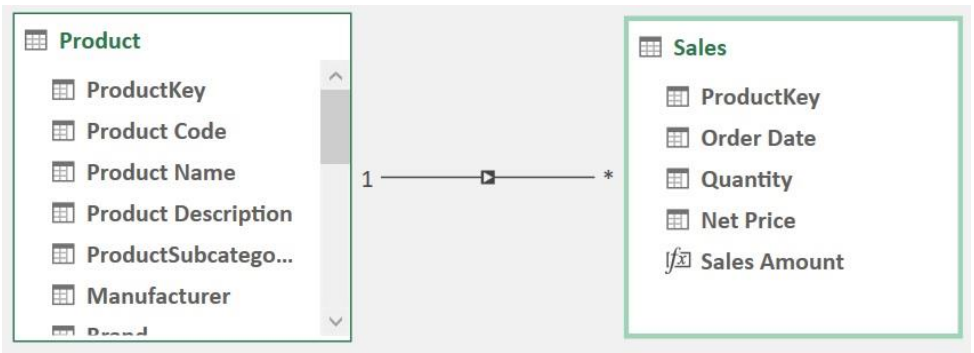


Рис. 4.5. В таблице Sales есть один столбец с датами – Order Date, без столбцов для представления лет и/или месяцев

При создании сводной таблицы с полем Sales Amount в области значений и Order Date в строках вы заметите небольшую задержку, после чего отобразится таблица:

	A	B	C
1			
2		Row Labels	Sales Amount
3		⊟2007	1 459 215,95
4		⊞Qtr1	329 358,16
5		⊞Qtr2	366 206,60
6		⊞Qtr3	382 041,25
7		⊞Qtr4	381 609,95
8		⊞2008	1 122 535,05
9		⊞2009	1 242 534,61
10		Grand Total	3 824 285,61

Рис. 4.6. В сводной таблице сделан срез данных по годам и кварталам, несмотря на то что в модели не было таких столбцов

Чтобы появилась возможность осуществлять срезы по годам, Excel автоматически добавил в таблицу Sales необходимые поля. Вы увидите их, если повторно откроете модель данных:

Рис. 4.7. В таблице Sales содержатся новые столбцы, которые были автоматически созданы в Excel

Если вы проделаете те же операции над другой таблицей фактов, в ней также будут созданы эти столбцы. При этом новые столбцы из обеих таблиц не могут быть использованы в едином перекрестном фильтре. Подробности об этом инструменте см. Time grouping enhancements in Excel 2016. В статье также есть информация о том, какие действия необходимо выполнить в системном регистре, чтобы отключить этот механизм. Если вы работаете с более или менее сложными моделями данных, мы советуем вам отключить этот помощник и научиться управляться со столбцами дат самостоятельно.

Автоматическая группировка дат в Power BI Desktop

Power BI Desktop создает по одной скрытой таблице для каждого столбца с датой и строит все необходимые связи. Когда вы осуществляете срез по дате, Power BI Desktop использует для визуализации календарную иерархию из созданной скрытой таблицы.

Использование нескольких измерений даты и времени

В одной таблице фактов может присутствовать сразу несколько полей с датами. Например, дата заказа, дата оплаты и дата поставки. Можно создать три измерения, т.е., три таблицы дат. Такой подход «раздует» модель, особенно, если таблиц фактом множество. Мы рекомендуем использовать одну таблицу с датами. А для анализа показателей с другим типом дат создавать специальные меры.

Если, к примеру, вам необходимо провести сравнение с участием даты заказа и даты поставки, вы можете сохранить в модели неактивную связь между таблицами Sales и Date по полю DeliveryDateKey и активировать ее программно для вычисления этой специфической меры. После добавления этой связи мы получим модель:

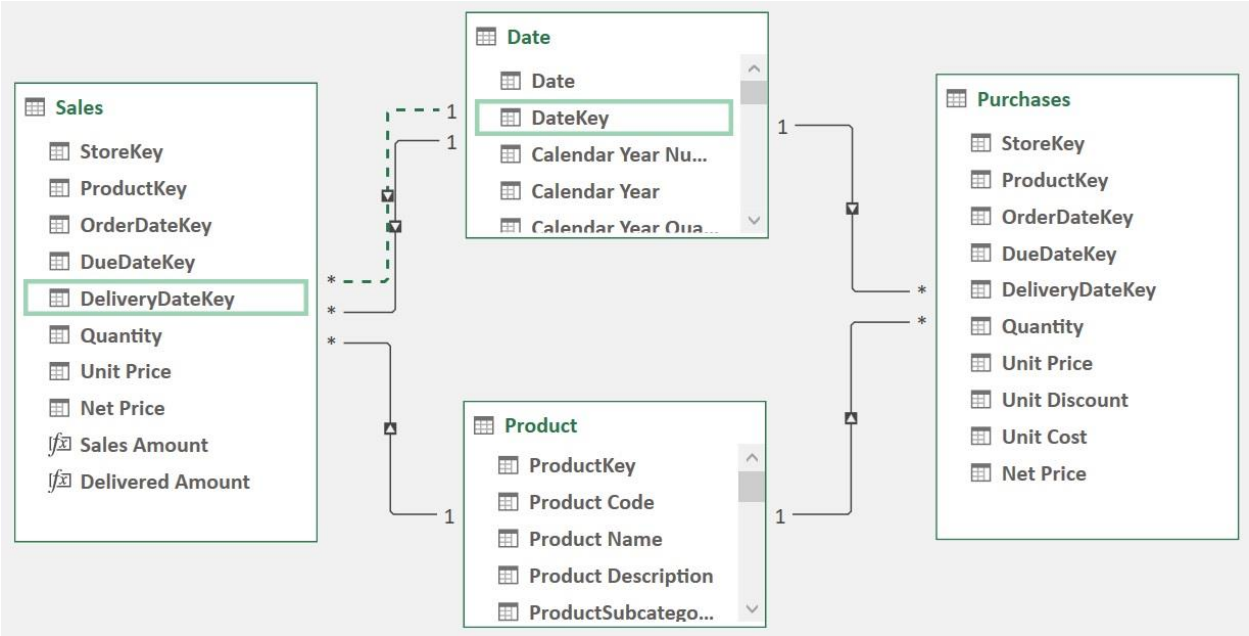


Рис. 4.16. Связь между полями DeliveryDateKey и DateKey присутствует, но она не активна

Теперь вы можете использовать формулу

```
Delivered Amount :=
CALCULATE (
        [Sales Amount];
        USERELATIONSHIP ( Sales[DeliveryDateKey]; 'Date'[DateKey] )
)
```

В этой мере включается неактивная связь между таблицами Sales и Date только на время вычисления. Так что вы можете применять срезы по тому же измерению Date и при этом извлекать информацию, связанную с поставкой:

	A	B	C	D
1				
2		**Row Labels**	**Sales Amount**	**Delivered Amount**
3		CY 2007	1 459 215,95	1 410 787,80
4		CY 2008	1 122 535,05	1 145 421,73
5		CY 2009	1 242 534,61	1 221 566,90
6		CY 2010		46 509,18
7		**Grand Total**	**3 824 285,61**	**3 824 285,61**

Рис. 4.17. Мера Delivered Amount использует связь с датой поставки, а логика расчетов скрыта внутри формулы

В большинстве моделей данных вполне достаточно одного измерения дат. Если вам необходимо произвести расчеты с использованием специфических дат, сделайте это внутри меры, воспользовавшись неактивной связью.

Таблица времени

Если вам необходимо вести учет времени, создайте отдельное от таблицы дат измерение для хранения времени. Как правило, таблица времени содержит 24*60 = 1440 строк – с интервалом 1 минута:

Time	Hour	Minute	TimeIndex	HourMinute
0:00:00	00	00	0	00:00
0:01:00	00	01	1	00:01
0:02:00	00	02	2	00:02
0:03:00	00	03	3	00:03
0:04:00	00	04	4	00:04
0:05:00	00	05	5	00:05
0:06:00	00	06	6	00:06
0:07:00	00	07	7	00:07
0:08:00	00	08	8	00:08
0:09:00	00	09	9	00:09
0:10:00	00	10	10	00:10
0:11:00	00	11	11	00:11
0:12:00	00	12	12	00:12
0:13:00	00	13	13	00:13
0:14:00	00	14	14	00:14

Рис. 4.18. Таблица для хранения времени в Power BI Desktop

Функции для работы с датой и временем

Если ваша модель данных спроектирована правильно, работать с датой и временем можно с использованием большого числа специализированных функций DAX. К примеру, если вам нужно провести вычисления средних продаж за последние 12 месяцев используйте меру:

```
Sales Avg12M :=
CALCULATE (
        [Sales Amount] / COUNTROWS ( VALUES ( 'Date'[Month] ) );
        DATESINPERIOD (
                'Date'[Date];
                MAX ( 'Date'[Date] );
                -12;
                MONTH
        )
)
```

Здесь функция DATESINPERIOD возвращает последние 12 месяцев, используя последнюю дату в контексте фильтра в качестве точки отсчета. Результат расчета средних показателей:

Row Labels	Sales Amount	Sales Avg12M
⊞2007	1 459 215,95	121 601,33
⊞2008	1 122 535,05	93 544,59
⊟2009	1 242 534,61	103 544,55
January	71 828,15	94 146,79
February	59 980,01	94 048,68
March	71 327,93	94 596,90
April	103 551,11	93 559,09
May	160 137,28	96 306,46
June	93 484,82	96 631,05
July	145 604,22	101 094,13
August	98 972,35	97 565,15
Septembe	90 457,03	95 765,51
October	91 665,16	98 482,67
Novembe	133 481,80	100 581,92
Decembe	122 044,75	103 544,55
Grand Total	3 824 285,61	

Рис. 4.20. Мера, вычисляющая средние показатели продаж за последние 12 месяцев

Работа с финансовыми календарями

При работе с финансовыми календарями нет необходимости добавлять специальные столбцы в таблицу фактов. Вместо этого вы вводите новые поля в ваше измерение с датами, чтобы при

желании иметь возможность осуществлять срезы как по обычному, так и по финансовому календарю. Представьте, например, что вам необходимо создать финансовый календарь, в котором первым месяцем будет июль. То есть финансовый год будет продолжаться с 1 июля по 30 июня. В таком случае вам понадобится модифицировать свое измерение, чтобы в нем появились финансовые месяцы, а также вам понадобится дополнительный столбец для их правильной сортировки, чтобы июль открывал год, а июнь – закрывал.

Специальные функции даты и времени ориентированы на работу со стандартным календарем. Но у некоторых из них есть дополнительный параметр, позволяющий взаимодействовать с финансовыми календарями. Например, при работе с финансовым календарем функции DATESYTD необходимо передать второй параметр, указывающий день и месяц окончания года:

```
Sales YTD Fiscal :=
CALCULATE (
        [Sales Amount];
        DATESYTD ( 'Date'[Date]; "06/30" )
)
```

Ниже представлен отчет с нарастающим итогом по обычному и финансовому календарям:

Row Labels	Sales Amoun	Sales YTD	Sales YTD Fiscal
⊟FY 2007	695 564,75	695 564,75	695 564,75
January	101 097,12	101 097,12	101 097,12
February	108 553,20	209 650,32	209 650,32
March	119 707,83	329 358,16	329 358,16
April	121 085,74	450 443,90	450 443,90
May	123 413,41	573 857,31	573 857,31
June	121 707,44	695 564,75	695 564,75
⊟FY 2008	1 286 923,00	523 271,80	1 286 923,00
July	139 381,00	834 945,75	139 381,00
August	87 384,31	922 330,06	226 765,31
September	155 275,94	1 077 606,00	382 041,25
October	99 872,65	1 177 478,64	481 913,89
November	122 522,86	1 300 001,50	604 436,75
December	159 214,45	1 459 215,95	763 651,19
January	64 601,67	64 601,67	828 252,87
February	61 157,39	125 759,06	889 410,25
March	64 749,27	190 508,33	954 159,53
April	116 004,84	306 513,17	1 070 164,36
May	127 168,83	433 682,00	1 197 333,20
June	89 589,80	523 271,80	1 286 923,00
⊞FY 2009	1 159 572,55	560 309,30	1 159 572,55
⊞FY 2010	682 225,31		682 225,31
Grand Total	3 824 285,61		

Рис. 4.22. Продажи в новой колонке обнуляются в июле, как мы и ожидали

Если у вас есть необходимость работать с более сложными календарями, вы можете обратиться к соответствующим шаблонам.

Большой раздел посвящен расчету рабочих дней. Приведена довольно сложная модель, учитывающая, что в разных странах выходными и праздничными будут разные дни.

Продемонстрирована работа с особыми периодами года. Например, с пасхальными и рождественскими периодами. Проблема в том, что Пасха каждый год выпадает на разные даты. Так что вам необходимо учитывать эти изменяющиеся периоды с целью их сравнения.

Также очень полезно уметь создавать отчеты и панели мониторинга (dashboard), содержимое которых обновляется в зависимости от даты формирования. К примеру, у вас есть панель мониторинга для сравнения продаж в текущем месяце с предыдущим. Проблема в том, что определение текущего месяца тесно связано с текущим днем. Сегодня текущим месяцем может

быть апрель, а в этот же день в следующем месяце – май, и нам бы не хотелось обновлять фильтры панели мониторинга каждый месяц.

Описана техника работы с недельными календарями. В качестве примера работы рассмотрены вычисления, основанные на стандарте ISO 8601.

Глава 5. Отслеживание исторических атрибутов

Если вам необходимо хранить состояние тех или иных атрибутов в зависимости от времени, значит, вы имеете дело с так называемыми *историческими атрибутами* (historical attributes), или, говоря техническим языком, *медленно меняющимися измерениями* (slowly changing dimensions – SCD).

Обычно вам нужно отслеживать значения атрибутов в измерениях. Например, может потребоваться узнать предыдущий адрес покупателя, чтобы проанализировать его приобретения до и после смены места жительства. В этом случае в измерении появится две строки для одного покупателя – одна со старым адресом, а вторая – с новым. В то же время в таблице фактов в строках, относящихся к этому покупателю, будет содержаться ссылка на нужную версию в измерении.

Если покупателя могут передавать от одного менеджера другому, то в правильно спроектированной модели вы должны увидеть в измерении Customer следующие два столбца:

- *Historical Manager*. Содержит менеджера, который был ответственным за этого покупателя в момент совершения продажи;
- *Current Manager*. Указывает на менеджера, прикрепленного к этому покупателю в данный момент.

В результате в таблице Customer мы получим несколько строк для каждого покупателя в зависимости от того, сколько менеджеров у него сменилось за все время. В нашей базе насчитывается 18 869 покупателей. Однако в таблице Customer содержится 43 882 строки. И если мы создадим простую меру для подсчета количества покупателей, как в приведенном ниже коде, результат будет неверным:

```
NumOfCustomers := COUNTROWS ( Customer )
```

Чтобы узнать, сколько же у нас покупателей, необходимо посчитать их уникальные коды:

```
NumOfCustomers2 := DISTINCTCOUNT ( Customer[Customer Code] )
```

Actual Manager	NumOfCustomers
Louise	10 901
Mark	26 082
Paul	1 937
Raoul	4 962
Total	**43 882**

Actual Manager	NumOfCustomers2
Louise	3 639
Mark	9 910
Paul	1 937
Raoul	3 383
Total	**18 869**

Рис. 5.8. Слева мера NumOfCustomers, справа – NumOfCustomers2

Загрузка медленно меняющихся измерений

В присутствии медленно меняющегося измерения гранулярность изменится как у таблицы фактов, так и у самого измерения. Без необходимости хранить историю смены менеджеров гранулярность таблицы фактов была установлена на уровне покупателя. Но с введением медленно меняющегося измерения уровень гранулярности повысится до версии покупателя. И именно версии покупателей должны быть связаны с продажами в зависимости от того, когда именно была совершена операция. Необходимо изменить запросы на формирование измерения и таблицы фактов, чтобы их гранулярности совпадали. Нельзя обновить уровень гранулярности только одной таблицы – в этом случае связь будет работать некорректно. Далее подробно описаны шаги для формирования такой модели.

Быстро меняющиеся измерения

Еще один ежегодно меняющийся атрибут, который вам может понадобиться отслеживать, – это возраст покупателя. Мы покажем вам другой способ отслеживать изменение атрибутов и для

этого введем понятие *быстро меняющегося измерения*. Поскольку само измерение не меняется – изменения касаются лишь отдельных атрибутов – лучшим вариантом будет вынести их в отдельное измерение, тем самым удалив из таблицы покупателей. Исходная модель данных, где возраст покупателя хранится в измерении Customer:

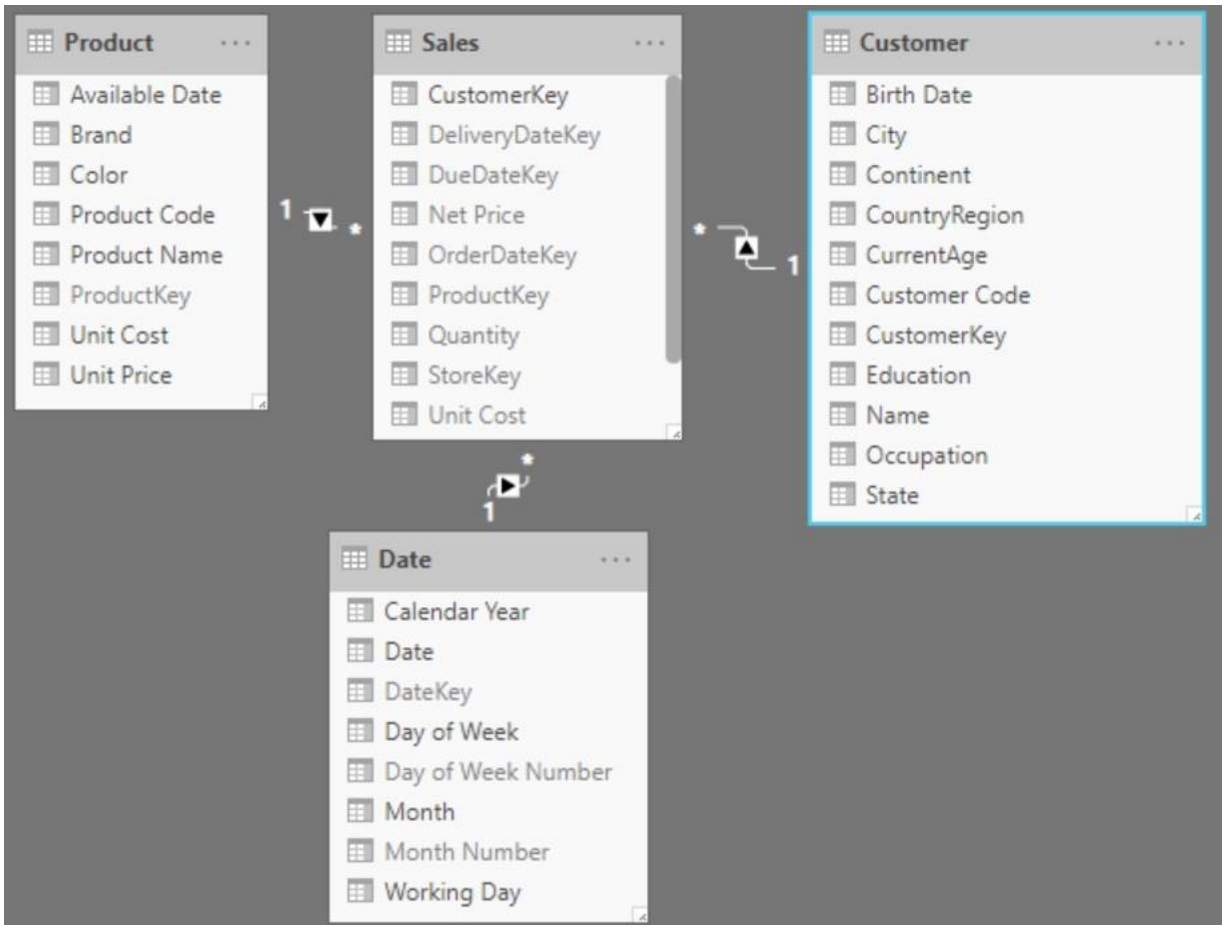


Рис. 5.24. Возраст хранится в качестве атрибута в измерении Customer

Поскольку этот атрибут меняется довольно часто, можно хранить его прямо в таблице продаж при помощи вычисляемого столбца:

```
Sales[Historical Age] =
DATEDIFF (
        RELATED ( Customer[Birth Date] );
        RELATED ( 'Date'[Date] );
        YEAR
)
```

В момент продажи будет вычисляться разница между днем рождения покупателя и текущей датой, и получившееся значение в годах будет сохранено в таблице Sales. Денормализация атрибута в таблице фактов позволяет вам не создавать для него отдельное измерение. Такой подход к хранению возраста покупателей не предусматривает полного перепроектирования модели данных, что понадобилось бы в случае введения медленно меняющегося измерения.

Одного этого дополнительного столбца достаточно, чтобы формировать полезные отчеты. К примеру, мы можем построить гистограмму по данным продаж с разбивкой по возрастам:

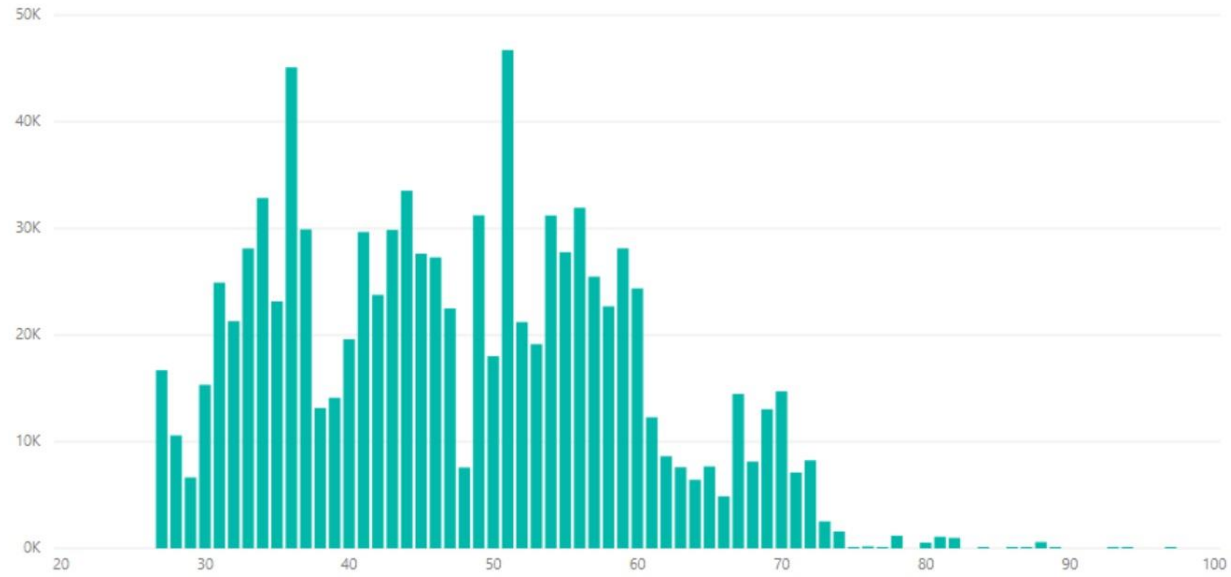


Рис. 5.25. Атрибут с историческим возрастом покупателей хорошо подходит для построения графиков и гистограмм

Если вас интересует объединение возрастов в группы для проведения более глубокого анализа, лучше вынести хранение этого атрибута в отдельное измерение, а возраст в таблице фактов использовать в качестве внешнего ключа. Измененная модель данных:

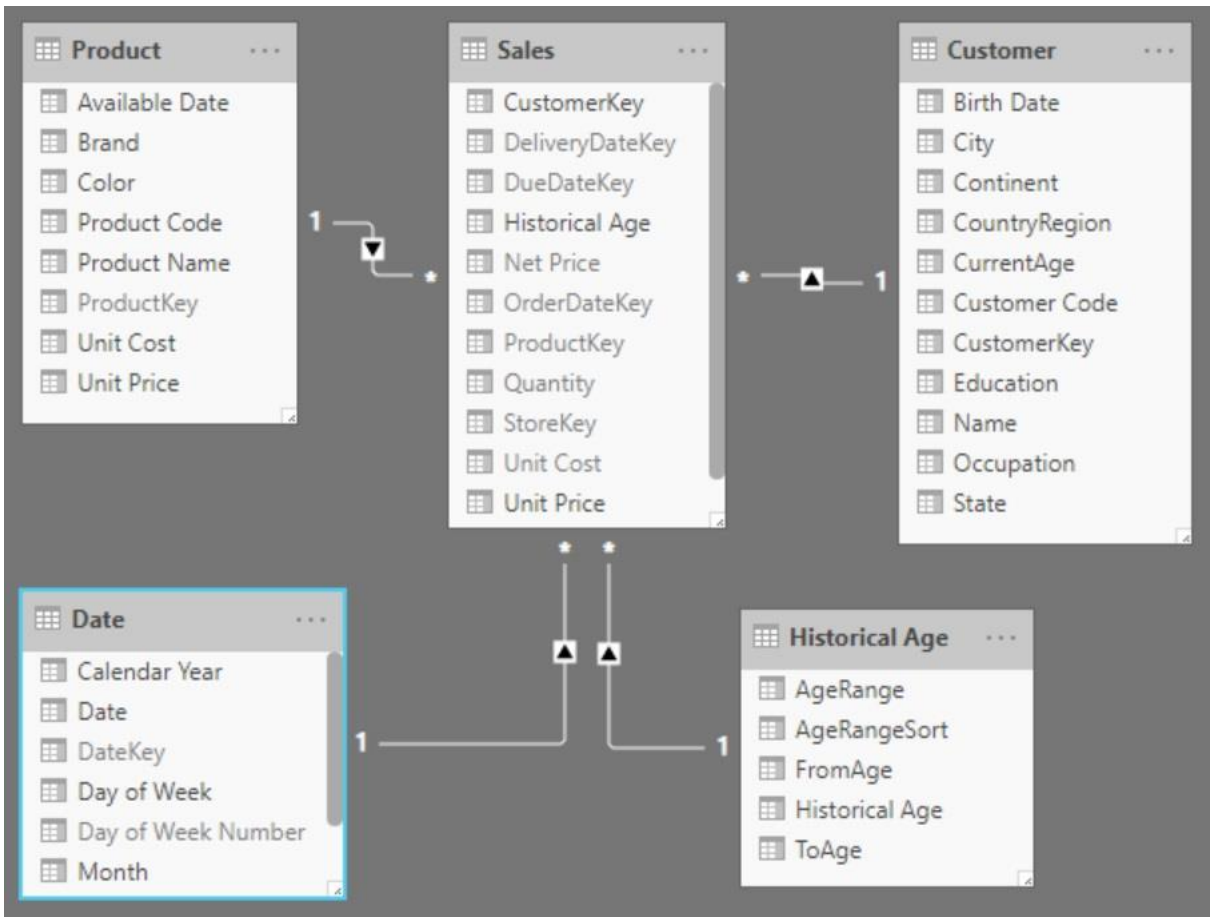


Рис. 5.26. Можно преобразовать возраст покупателя во внешний ключ для связи с измерением возрастов

В измерении Historical Age мы можем хранить диапазоны возрастов и другую информацию. Это позволит нам осуществлять срезы в отчетах не по возрасту, а по целым возрастным группам:

AgeRange	Sales Amount	NumOfCustomers	AverageSale
	2 870 569,69	352	$8 155,03
25-40	281 923,90	615	$458,41
40-50	252 664,85	556	$454,43
50-60	272 339,29	525	$518,74
Over 60	146 787,88	310	$473,51
Total	**3 824 285,61**	**2 353**	**$1 625,28**

Рис. 5.27. С отдельным измерением можно осуществлять срезы по возрастным группам

Выбор оптимальной техники моделирования

Если это возможно, старайтесь выделять меняющийся атрибут (или набор атрибутов) в отдельное измерение. В этом случае вам не нужно будет менять гранулярность таблиц. Однако если таких атрибутов слишком много, лучше пойти по более сложному пути создания полноценного медленно меняющегося измерения.

Глава 6. Использование снимков

Ранее мы говорили о разделении таблиц в модели на измерения и таблицы фактов. На самом деле таблицы фактов не всегда отражают события. Иногда в них содержатся измеряемые показатели вроде температуры двигателя, среднего ежедневного потока покупателей в магазине по месяцам или результатов складской инвентаризации. Во всех этих случаях рассчитанная информация хранится на определенный момент времени и не отражает конкретного события. Обычно такие сценарии моделируются при помощи снимков. Еще один пример снимка – остаток на расчетном счете. Фактом является каждая отдельная транзакция по счету, а снимок отражает, каким был баланс на определенный момент времени.

Снимок – это не факт, а измерение, проведенное в конкретный момент времени. Различия между таблицей фактов и снимком заключаются, скорее, в природе хранимой информации, а не в структуре. Еще один пример снимка – таблица с курсами валют.

Различают следующие разновидности снимков:

естественный снимок. Например, информацию о температуре воды в двигателе на ежедневной основе.

производный снимок. Например, баланс расчетного счета на ежемесячной основе. Баланс счета является производной величиной от суммы всех произведенных транзакций (доходов и расходов) со счетом за предыдущий период. Так что информация здесь лишь хранится как снимок, но также может быть рассчитана путем простой агрегации соответствующих транзакций.

Агрегирование снимков

В качестве примера возьмем еженедельную инвентаризацию товаров в магазинах. Работая со снимками, необходимо помнить, что в них не должны содержаться аддитивные меры. *Аддитивная мера* представляет собой меру, которая может агрегироваться с применением функции SUM по всем измерениям. Мы можем пользоваться функцией суммирования для агрегации остатков по магазинам, но не должны этого делать по измерению времени. Снимки хранят информацию, актуальную на конкретный момент времени. Но при вычислении итоговых показателей по измерению времени обычно не пользуются функцией суммирования. Вместо этого лучше воспользоваться функциями получения последнего доступного значения или вычисления среднего.

Это типичный сценарий для использования *полуаддитивной меры*, показывающей последние доступные данные по времени. Традиционно в полуаддитивной мере применяется функция LASTDATE, извлекающая последнюю дату в заданном интервале. Но и она не всегда будет работать корректно, если в снимке и в таблице дат последние даты периода не совпадают. Например, в снимке данные о инвентаризации есть только на каждый понедельник.

Гранулярность в снимке редко устанавливается на уровне дня. Смена гранулярности вкупе с полуаддитивными мерами может доставлять проблемы. Формулы для подсчета итогов могут оказаться довольно сложными. Для оптимизации кода используйте вычисляемые столбцы в измерении дат, в которых будут предварительно вычисляться даты, представленные в снимках.

Понятие производных снимков

Производным снимком называется предварительно агрегированная таблица, содержащая сжатые данные. В большинстве случаев снимки создаются для повышения производительности модели. Представьте, что вам нужно построить отчет о ежемесячном количестве покупателей с разбивкой на новых и тех, кто уже приобретал у нас товары ранее (будем называть их вернувшимися). Вы можете воспользоваться предварительно рассчитанными данными из таблицы:

DateKey	Customers	NewCustomers	ReturningCustomers
20070131	182	182	0
20080131	40	27	13
20090131	21	10	11
20070228	154	147	7
20080229	54	40	14
20090228	47	39	8
20070331	152	148	4
20080331	61	56	5
20090331	49	42	7
20070430	185	177	8
20080430	100	94	6
20090430	26	19	7
20070531	153	143	10
20080531	43	38	5

Рис. 6.7. В таблице хранится информация о новых и вернувшихся покупателях в виде снимка

Эта предварительно агрегированная таблица может быть добавлена в модель данных и объединена связью с измерением Date. Это позволит вам строить по ней отчеты:

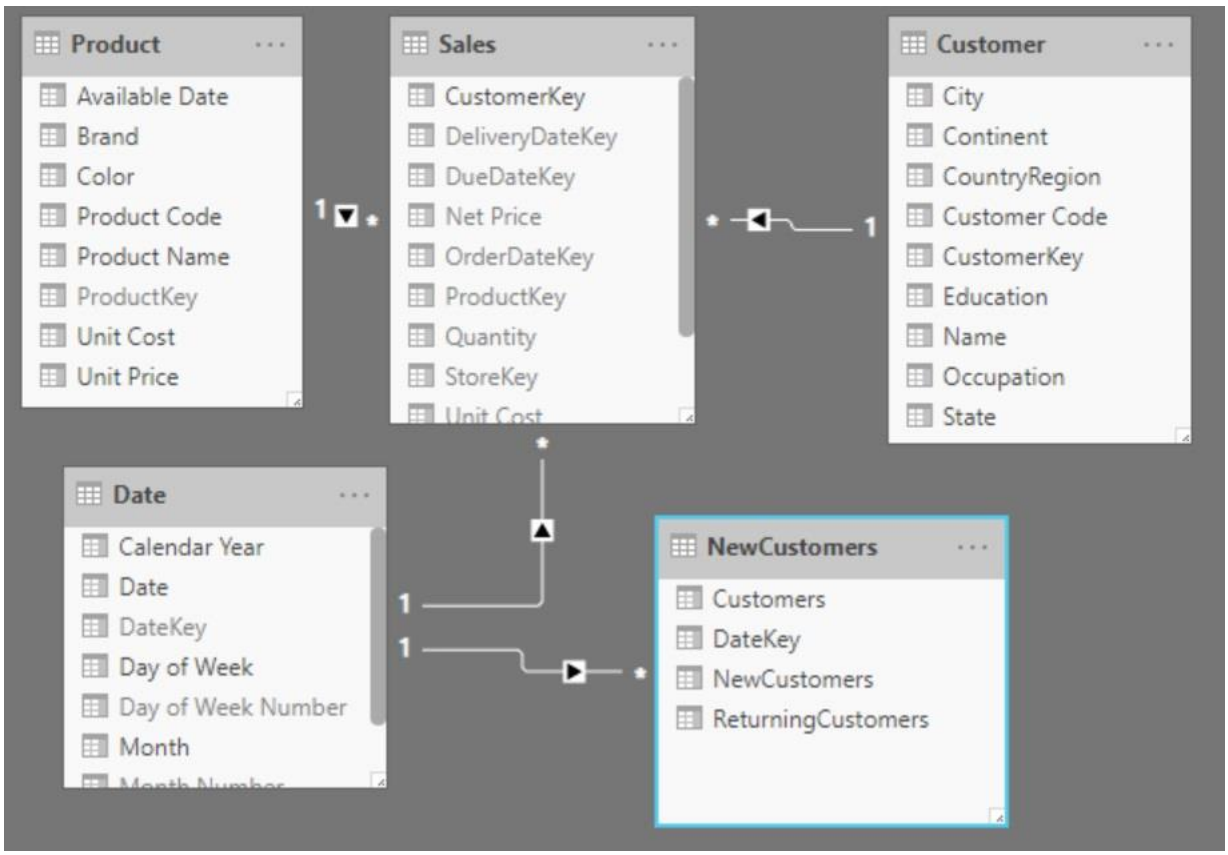


Рис. 6.8. NewCustomers – новая таблица, добавленная в модель посредством связи

В снимке хранится всего по одной записи для каждого месяца. Вы уже сейчас можете построить ежемесячный отчет по покупателям:

Calendar Year	Month	Sales Amount	Customers	NewCustomers	ReturningCustomers
CY 2007	January	101 097,12	182	182	0
CY 2007	February	108 553,20	154	147	7
CY 2007	March	119 707,83	152	148	4
CY 2007	April	121 085,74	185	177	8
CY 2007	May	123 413,41	153	143	10
CY 2007	June	121 707,44	86	75	11
CY 2007	July	139 381,00	113	106	7
CY 2007	August	87 384,31	119	114	5
CY 2007	September	155 275,94	81	73	8
CY 2007	October	99 872,65	70	60	10
CY 2007	November	122 522,86	106	98	8
CY 2007	December	159 214,45	97	86	11
CY 2008	January	64 601,67	40	27	13
CY 2008	February	61 157,39	54	40	14
Total		**3 824 285,61**	**2655**	**2353**	**302**

Рис. 6.9. Простой ежемесячный отчет на основании снимка

Производительность этого отчет высока, поскольку все данные для него заранее агрегированы, и на его формирование потребуется всего несколько миллисекунд. Но за такую скорость приходится платить следующими негативными последствиями:

- вы не сможете рассчитывать подытоги. Вам недоступно агрегирование значений при помощи функции SUM.
- вы не сможете осуществлять срезы по другим атрибутам. Представьте, что вам потребовался такой же отчет, но отфильтрованный по покупателям конкретной категории товаров. В этом случае наш снимок окажется бесполезным. То же самое касается среза по дате и любому другому атрибуту большей детализации, чем месяц.

Рассмотрен пример с использованием матрицы переходов.

Глава 7. Анализ интервалов даты и времени

В главе представлены сценарии, в которых временные показатели будут главным предметом аналитики, а не просто измерением для осуществления срезов. В обычной модели данных фактом является неделимое событие, произошедшее в конкретный момент времени. В этой главе факты рассматриваются как события, обладающие определенной длительностью. Так что в модели мы будем хранить не дату события, а точку во времени, в которой это событие стартовало. Длительность этого события мы будем вычислять с помощью DAX.

В традиционных базах дату и время хранят в одном столбце. Мы советуем при загрузке данных в модель разбивать информацию о дате и времени события на два столбца. В этом случае измерение дат будет хранить данные с гранулярностью до дня, а в измерении времени будет содержаться только время. Таким образом, для хранения событий в интервале десяти лет вам понадобится измерение дат объемом 3650 строк, а в таблице со временем будет находиться 1440 строк, если данные нужны с детализацией до минуты. Если бы дата и время хранились в одной таблице, нам бы потребовалось измерение, содержащее 5 256 000 строк (3650 раз по 1440). Разница в скорости обработки запросов будет существенной.

Работа с интервалами предполагает определенную смену образа мышления и пересмотр самого понятия факта. Вы можете хранить факты вместе с их длительностью, но в этом случае вам необходимо пересмотреть концепцию временных интервалов, поскольку факт может распространяться на несколько периодов.

Когда длительности (или интервалы) в вашей модели пересекаются по времени, вам стоит изменить гранулярность таблицы фактов.

Если сутки в вашей модели данных не заканчиваются в полночь, вы можете осуществить временной сдвиг, чтобы день начинался, к примеру, не с 00:00, а с 02:00. Это полезно для включения ТВ-передач начала ночи в предыдущую дату.

Глава 8. Связи «многие ко многим»

Существуют сценарии, в которых невозможно выразить отношение между двумя сущностями при помощи одной связи. Типичный пример такого сценария – расчетный счет. У счета может быть

сразу несколько владельцев, тогда как у каждого владельца может быть не один расчетный счет. Таким образом, вы не можете добавить поле с ключом покупателя в таблицу Accounts (счета), как не можете хранить ссылку на расчетный счет в таблице Customers. Такой тип отношения по своей природе выражает наличие соответствия многих записей из одной таблицы многим строкам из другой.

Типичным способом работы со связями «многие ко многим» является создание таблицы-моста, содержащей информацию о владельцах счетов. На рис. 8.1 показан пример модели данных, в которой реализована связь «многие ко многим» между клиентами и расчетными счетами.

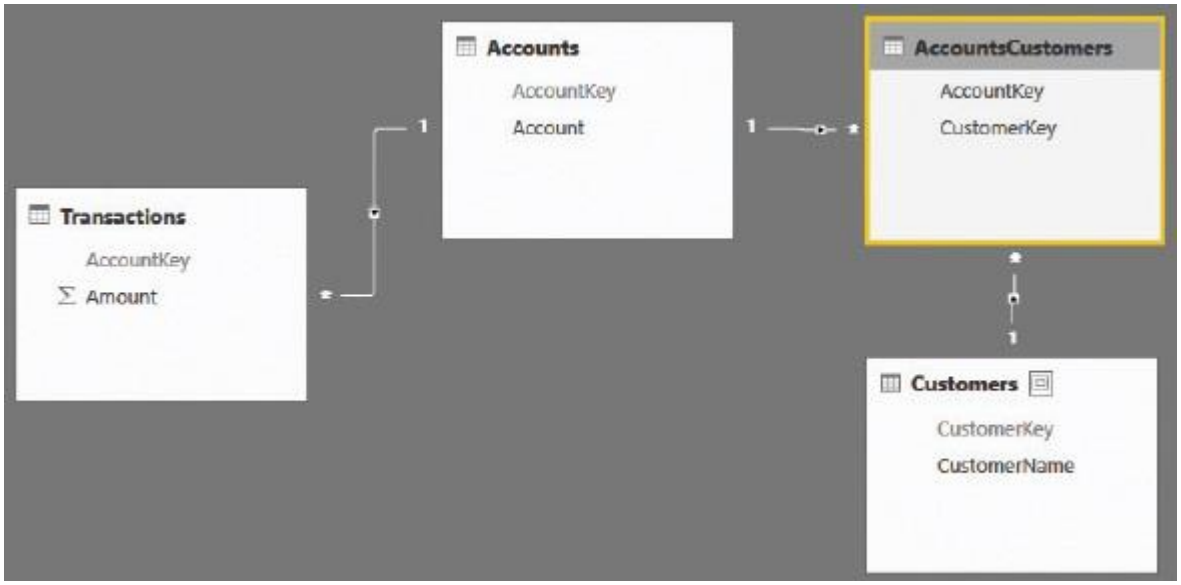


Рис. 8.1. Связь между таблицами Customers и Accounts осуществляется посредством таблицы-моста AccountsCustomers

Первое, что необходимо усвоить касательно связей «многие ко многим», – это то, что связями они называются лишь с точки зрения модели данных, тогда как на практике представляют собой пару обычных связей «один ко многим». Мы рассуждаем и работаем со связью «многие ко многим» как с физическим отношением между таблицами, хотя на самом деле его не существует.

Также стоит отметить, что связи, идущие от таблиц Customers и Accounts к мосту, разнонаправленные. Фактически вектор каждой из этих связей направлен от таблицы-моста к измерению. При этом мост всегда будет находиться на стороне «многих».

По умолчанию фильтр между таблицами распространяется по связи от «од-ного» ко «многим», но не наоборот. Таким образом, если построить отчет и осуществить в нем срез по покупателю, фильтр достигнет таблицы-моста и на этом остановится. А значит, на таблицу Accounts фильтр, установленный в Customers, не распространится, как показано на рис. 8.2.

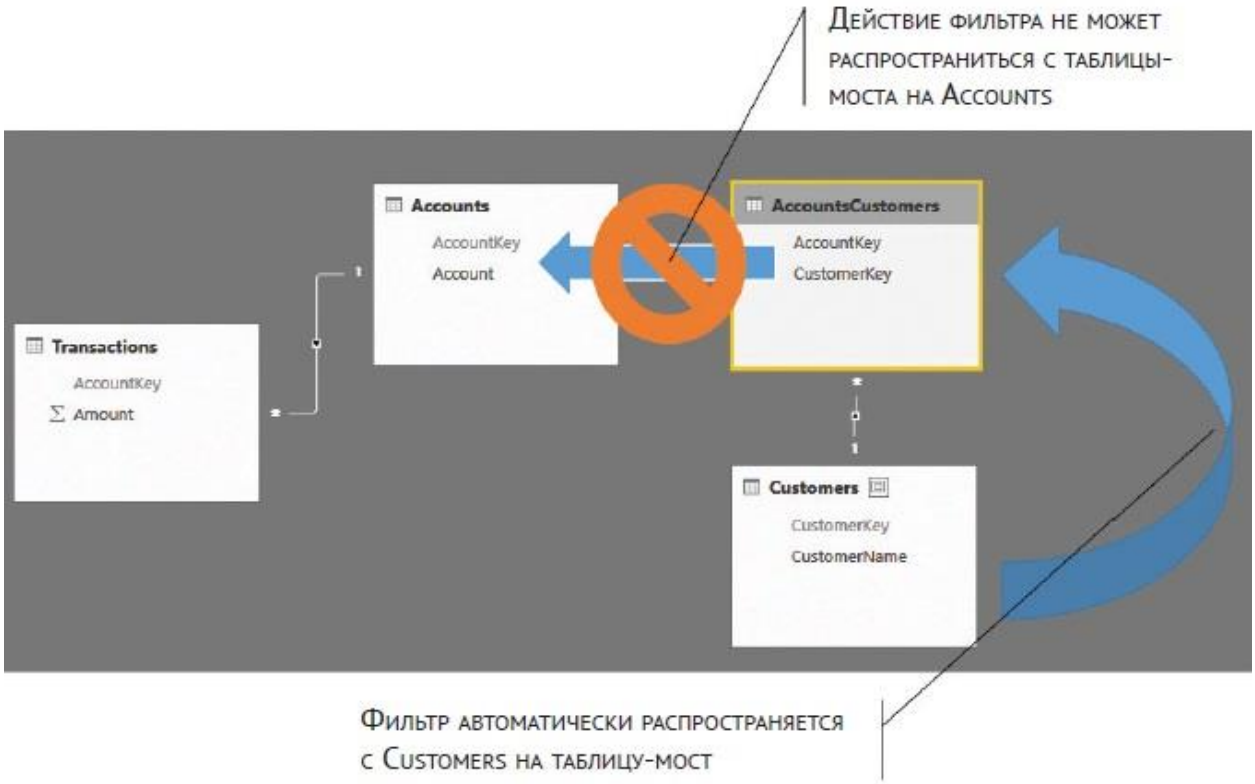


Рис. 8.2. Фильтр может распространяться от «одного» ко «многим», но не наоборот

Если вы в отчете вынесете на строки покупателей, а в единственной колонке в значениях примените функцию SUM к полю Amount из таблицы Transactions (транзакции), то увидите для всех строк одинаковые суммы. Это произошло из-за того, что фильтр, наложенный на Customers, не смог пробиться через таблицу Accounts к транзакциям.

CustomerName	Amount
Luke	€ 5,000
Mark	€ 5,000
Paul	€ 5,000
Robert	€ 5,000
Total	**€ 5.000**

Рис. 8.3. Вы не сможете фильтровать транзакции по покупателям из-за наличия связи «многие ко многим»

Решить эту проблему можно, включив двунаправленную фильтрацию между таблицей-мостом и Accounts. В Power BI такая возможность заложена в саму модель данных, тогда как в Excel вам придется воспользоваться помощью DAX.

Если включить двунаправленную фильтрацию в модели, ее действие будет распространяться на все вычисления. В то же время если активировать соответствующий шаблон при помощи включения функции CROSSFILTER в качестве параметра CALCULATE, его действие будет ограничено только этой инструкцией.

```
SumOfAmount :=
CALCULATE (
        SUM ( Transactions[Amount] );
        CROSSFILTER ( AccountsCustomers[AccountKey]; Accounts[AccountKey]; BOTH )
)
```

В процессе вычисления этой меры фильтр сможет распространять свое действие от таблицы-моста к Accounts, а значит, в вывод попадут только строки, принадлежащие выбранному покупателю.

CustomerName	Amount	SumOfAmount
Luke	$5,000.00	$800.00
Mark	$5,000.00	$2,800.00
Paul	$5,000.00	$1,700.00
Robert	$5,000.00	$1,700.00
Total	**$5,000.00**	**$5,000.00**

Рис. 8.4. Мера SumOfAmount показывает правильные значения, тогда как Amount во всех строках выводит итог

Понятие неаддитивности

Другой важной особенностью применения связей типа «многие ко многим» является то, что меры, агрегируемые посредством таких связей, обычно получаются неаддитивными. Это не ошибка в модели данных, а особенность таких связей. Чтобы лучше понять, о чем речь, посмотрите на матрицу, показанную на рис. 8.6, в которой собраны одновременно данные по таблицам Accounts и Customers.

Account	Luke	Mark	Paul	Robert	**Total**
Luke	$800.00				**$800.00**
Mark		$800.00			**$800.00**
Mark-Paul		$1,000.00	$1,000.00		**$1,000.00**
Mark-Robert		$1,000.00		$1,000.00	**$1,000.00**
Paul			$700.00		**$700.00**
Robert				$700.00	**$700.00**
Total	**$800.00**	**$2,800.00**	**$1,700.00**	**$1,700.00**	**$10,000.00**

Рис. 8.6. Связи «многие ко многие» генерируют неаддитивные меры

Итоги по колонкам показываются правильные, то есть составляют суммы значений по строкам. Однако итоги по строкам заполнены неверно. Это происходит из-за того, что суммы по счетам выводятся для всех их владельцев. Например, счетом Mark-Paul одновременно владеют Марк и Пол. Для каждого из них индивидуально сумма баланса составляет 1000 долларов, но когда мы рассматриваем их вместе, баланс не меняется и по-прежнему равен 1000 долларов.

Неаддитивные меры – это не ошибка. Это характерное поведение для мер, когда вы работаете со связями типа «многие ко многим».

* * *

Три заключительные главы посвящены моделям данных, использующим таблицы разной гранулярности, сегментации данных в моделях и работе с несколькими валютами. В приложении кратко суммированы основные понятия моделирования данных: таблицы, типы данных, связи, фильтрация и перекрестная фильтрация, различные типы моделей (схема «звезда» и «снежинка»), модели с таблицами-мостами, аддитивные, неаддитивные и полуаддитивные меры.