УДК 004.424
ББК 32.372
Ф43

Альберто Феррари и Марко Руссо
Ф43 Анализ данных при помощи Microsoft Power BI и Power Pivot для Excel / пер. с анг. А. Ю. Гинько. – М.: ДМК Пресс, 2020. – 288 с.: ил.

ISBN 978-5-97060-858-6

УДК 004.424
ББК 32.372

В этой книге представлены базовые техники моделирования данных в Excel и Power BI. Авторы, специалисты в области бизнес-аналитики, делают акцент на реальных ситуациях, с которыми регулярно сталкиваются как консультанты. Они продемонстрируют общие техники моделирования, научат читателя 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.

Издание предназначено как для новичков, так и для специалистов в области моделирования данных, желающих получить советы экспертов. Для изучения материала требуется владение Excel на среднем или продвинутом уровне.

ISBN (анг.) 978-1-5093-0276-5
ISBN (рус.) 978-5-97060-858-6

Оглавление

Рецензия

Вы держите в руках уникальную по нескольким причинам книгу.

Во-первых, это первая книга на русском языке по системе бизнес-аналитики Microsoft Power BI. В течение нескольких последних лет, когда слушатели после тренингов по Excel, Power Pivot и Query спрашивали «что мне почитать про Power BI?», я не знал, что ответить. Англоязычной литературы написано по этой теме уже много, но на русском – полный ноль. Теперь уже нет.

Во-вторых, я очень рад, что в качестве первой ласточки издательство «ДМК Пресс» решило перевести именно эту книгу. Альберто Феррари и Марко Руссо однозначно входят в круг самых достойных авторов в этой области. Они щедро делятся своими знаниями в книгах и статьях, выступают на конференциях и проводят тренинги по Power Pivot, DAX и Power BI ещё с самого начала появления этих технологий и знают о них больше, чем кто бы то ни было. Отдельно, как тренер, хочу отметить их преподавательский талант, стройность и логичность объяснений, красоту примеров – это дорогого стоит.

Бизнес-аналитика (Business Intelligence, BI) давно уже перестала быть уделом гиков-айтишников из миллиардных корпораций. Сегодня она способна принести пользу при принятии управленческих решений в компании любого калибра, помочь визуализировать результаты и непрерывно отслеживать их динамику, собирая данные из разных «вселенных»: бухгалтерских программ, баз данных, файлов, интернета. Сегодня каждый может (и должен!) быть «сам себе аналитик». И эта книга – настоящий клад и огромное подспорье для всех, кто встал на этот путь.

Николай Павлов,
Microsoft Certified Trainer, Microsoft Most Valuable Professional,
автор проекта «Планета Excel», www.planetaexcel.ru

Предисловие от издательства

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв прямо на нашем сайте **www.dmkpress.com**, зайдя на страницу книги, и оставить комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу **dmkpress@gmail.com**, при этом напишите название книги в теме письма.

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу **http://dmkpress.com/authors/publish_book/** или напишите в издательство по адресу **dmkpress@gmail.com**.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг – возможно, ошибку в тексте или в коде, – мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от расстройств и поможете нам улучшить последующие версии этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу **dmkpress@gmail.com**, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательство «ДМК Пресс» очень серьезно относится к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или веб-сайта, чтобы мы могли применить санкции.

Пожалуйста, свяжитесь с нами по адресу электронной почты **dmkpress@gmail.com** со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую нам предоставлять вам качественные материалы.

Введение

Пользователи Excel любят цифры. А может, те, кто любят цифры, любят Excel. Как бы то ни было, если вам нравится доходить до самой сути при анализе любых наборов данных, скорее всего, вы провели немало времени, работая с Excel, сводными таблицами и формулами.

В 2015 году увидел свет программный продукт Power BI. И сегодня справедливо будет утверждать, что те, кто любят цифры, любят также Power Pivot для Excel и Power BI. Эти средства имеют много общего – в частности, их объединяет движок баз данных VertiPaq, а также язык DAX, унаследованный от SQL Server Analysis Services.

В прежних версиях Excel процесс анализа информации главным образом основывался на загрузке наборов данных, расчете значений в столбцах и написании формул для построения графиков. При этом в своей работе вы сталкивались с серьезными ограничениями – начиная с размера рабочей книги и заканчивая тем, что язык формул Excel не лучшим образом подходит для решения числовых задач большого объема. Новый движок, лежащий в основе Power BI и Power Pivot, стал огромным шагом вперед. С ним в вашем распоряжении оказался полный функционал баз данных, а также потрясающий язык DAX. Но ведь с большой силой приходит и большая ответственность! И если вы хотите воспользоваться всеми преимуществами этих новых средств, вам придется многому научиться. В частности, необходимо будет познакомиться с основами моделирования данных.

Моделирование данных – это отнюдь не ядерная физика, а лишь набор базовых знаний, которым должен овладеть всякий, кто заинтересован в анализе данных. К тому же если вы любите цифры, то вам непременно придется по душе моделирование данных. Освоить эту науку будет несложно, а вместе с тем вы получите массу удовольствия.

В этой книге вы познакомитесь с базовыми концепциями моделирования данных на практических примерах, с которыми наверняка не раз встречались в жизни. В наши планы не входило написание запутанной книги с подробным описанием комплексных решений, необходимых для реализации сложных систем. Вместо этого мы сосредоточились на реальных ситуациях, с которыми ежедневно сталкиваемся в работе в качестве консультантов. Когда к нам обращались за помощью, а мы видели, что имеем дело с типичной задачей, то отправляли ее прямиком в архив. Позже, открыв заветный ящик, мы получили ценные примеры для книги и расположили их в порядке, пригодном для обучения моделированию данных.

Прочитав эту книгу, вы вряд ли станете гуру в области создания моделей данных, но знаний по этой теме у вас существенно прибавится. И если впоследствии в поиске решения очередной задачи на вычисление нужного вам значения вы допустите мысль об изменении модели данных, значит, мы поработали не зря. Кроме того, вы уверенно вступите на путь становления успешного специалиста в области моделирования данных. Но заключительный шаг к вершине вы сможете сделать, только набравшись практического опыта и набив немало шишек. К сожалению, опыт нельзя приобрести, читая книги.

Для кого предназначена эта книга?

Целевая аудитория книги довольно разнообразна. В нее входят и пользователи Excel, применяющие в своей практике Power Pivot, и специалисты по анализу данных в Power BI, и даже новички в области бизнес-аналитики, желающие познакомиться с основами моделирования данных. Все они потенциальные читатели данной книги.

Заметьте, что мы не включили в этот список тех, кто целенаправленно хочет почитать о создании моделей данных. Изначально мы предполагали, что наш читатель может даже не знать, что ему нужно какое-то моделирование каких-то данных. Наша цель – дать вам понять, что проектирование моделей данных – это как раз то, что вам нужно, и познакомить с базовыми принципами этой прекрасной науки. В общем, если вам интересно, что такое моделирование данных и чем оно так полезно, эта книга для вас.

Как мы представляем себе нашего читателя?

Мы предполагаем, что наш читатель обладает базовыми знаниями в области сводных таблиц Excel и/или имеет опыт использования Power BI в качестве средства отчетности и моделирования. Наличие аналитических навыков также приветствуется. В своей книге мы не затрагиваем вопросы интерфейса Excel или Power BI. Вместо этого мы фокусируем свое внимание исключительно на моделях данных – как проектировать и модифицировать их так, чтобы значительно упростить запросы. Так что наша задача – рассказать вам, что делать, а как это делать, вы уж решите сами. Мы не планировали создавать пошаговое руководство, а хотели максимально простым языком объяснить достаточно сложную тему.

Также мы намеренно обошли вниманием описание языка DAX. Было бы невозможно уместить в одной книге и теорию моделирования данных, и DAX. Если вы уже знакомы с этим языком, вам будет проще разобраться с многочисленными примерами кода на DAX, представленными в данной книге. В противном случае советуем вам прочитать книгу «Подробное руководство по DAX» (The Definitive Guide to DAX), являющуюся полноценным

учебником по этому языку и хорошо сочетающуюся с приведенными в нашей книге примерами.

Структура книги

Книга начинается с пары легких вводных глав, за которыми следуют главы, каждая из которых посвящена отдельному виду модели данных. Предлагаем вам краткое описание:

- глава 1 «Введение в моделирование данных». Является вводной частью в базовые принципы моделирования данных. В ней мы расскажем, что из себя представляет модель данных, начнем говорить о понятии гранулярности, определим понятия основных моделей хранилища данных – «звезда» и «снежинка», – а также поговорим о нормализации и денормализации;
- глава 2 «Использование главной/подчиненной таблицы». Описывает наиболее распространенный сценарий с наличием главной и подчиненной таблиц. В этой главе мы обсудим пример с заказами и строками заказов, размещенными в двух отдельных таблицах фактов;
- глава 3 «Использование множественных таблиц фактов». Описывает сценарии, в которых у вас есть множество таблиц фактов, на основании которых необходимо построить единый отчет. В этой главе мы подчеркнем важность создания корректной многомерной модели для облегчения работы с информацией;
- глава 4 «Работа с датой и временем». Это одна из самых длинных глав книги. В ней затронуты вопросы логики расчетов на основании временных периодов. Мы расскажем, как правильно создать таблицу-календарь и работать с функциями времени (YTD, QTA, PARALLELPERIOD и др.). После этого приведем несколько примеров расчетов на основании рабочих дней, поработаем с особыми периодами года и поясним в целом, как правильно работать с датами;
- глава 5 «Отслеживание исторических атрибутов». В этой главе описываются особенности использования в модели данных медленно меняющихся измерений. Также представлено детальное описание трансформаций, которые необходимо выполнить для отслеживания исторических атрибутов, и даны инструкции по написанию корректного кода на DAX, учитывающего медленно меняющиеся измерения;
- глава 6 «Использование снимков». Описывает любопытные аспекты использования снимков (snapshot). В этой главе вы узнаете, что такое снимки, когда и для чего их необходимо использовать, а также как рассчитывать значения при применении снимков. Кроме того, мы посмотрим, как можно использовать мощную модель с применением матрицы переходов;

- глава 7 «Анализ интервалов даты и времени». В этой главе мы пойдем еще на шаг дальше, чем в главе 5. Мы продолжим заниматься временными вычислениями, но на этот раз обратимся к модели данных, в которой события, хранящиеся в таблице фактов, обладают определенной длительностью, а значит, требуют особого подхода для получения корректных результатов;
- глава 8 «Связи многие ко многим». Описывает характерные особенности использования связей «многие ко многим». Такой тип связи играет важную роль в любой модели данных. Мы рассмотрим обычные связи «многие ко многим», связи с каскадными действиями и их использование с учетом факторов перераспределения и фильтров. Также обсудим вопросы производительности таких связей и способы ее улучшения;
- глава 9 «Работа с разными гранулярностями». В этой главе мы углубимся в работу с таблицами фактов с разными уровнями гранулярности. Мы рассмотрим примеры из области бюджетирования, в которых таблицы фактов будут хранить информацию с разной степенью детализации, и предложим несколько альтернативных способов для решения этих ситуаций как при помощи языка DAX, так и непосредственно в модели данных;
- глава 10 «Сегментация данных в модели». В этой главе мы рассмотрим несколько моделей с применением техники сегментации. Начнем с простой сегментации по цене, после чего перейдем к анализу динамической сегментации с использованием виртуальных связей. В конце главы проведем ABC-анализ средствами DAX;
- глава 11 «Работа с несколькими валютами». В этой главе мы рассмотрим особенности работы с несколькими валютами. Взаимодействуя с курсами валют, важно понимать их специфику и в соответствии с ней строить модель данных. Мы проанализируем несколько сценариев с разными требованиями и для каждого из них выработаем оптимальное решение;
- приложение А «Моделирование данных 101». Это приложение можно рассматривать как справочное руководство. Здесь мы кратко опишем на примерах все базовые концепции, использованные в этой книге. При возникновении вопросов вы всегда можете обратиться к приложению, освежить в памяти соответствующую тему и вернуться к чтению.

Сложность моделей и решений будет возрастать на протяжении всей книги, так что мы советуем читать ее последовательно, а не прыгать от главы к главе. Так вы сможете постепенно идти от простого к сложному и осваивать по одной теме за раз. После прочтения книга может стать для вас справочным руководством, и когда вам потребуется построить ту или иную модель данных, вы можете смело открыть нужную главу и воспользоваться предложенным решением.

Условные обозначения

В этой книге приняты следующие условные обозначения:

- **жирным** помечен текст, который вводите вы;
- *курсив* используется для обозначения новых терминов;
- программный код обозначен в книге `моноширинным шрифтом`;
- первые буквы в названиях диалоговых окон, их элементов, а также команд – прописные. Например, в диалоговом окне **Save As...** (Сохранить как...);
- комбинации нажимаемых клавиш на клавиатуре обозначаются знаком плюс (+) между названиями клавиш. Например, **Ctrl**+**Alt**+**Delete** означает, что вы должны одновременно нажать клавиши **Ctrl**, **Alt** и **Delete**.

Сопутствующий контент

Для подкрепления ваших навыков на практике мы снабдили книгу сопутствующим контентом, который можно скачать по ссылке: https://aka.ms/AnalyzeData/downloads.

Представленный архив содержит файлы в форматах Excel и/или Power BI Desktop для всех примеров из этой книги. Каждому рисунку соответствует отдельный файл, чтобы вы имели возможность анализировать разные шаги и присоединиться к выполнению примера на любой стадии. Для большинства примеров представлены файлы в формате Power BI Desktop, так что мы настоятельно рекомендуем вам установить этот программный пакет с сайта Power BI.

Благодарности

В конце вводной главы мы бы хотели выразить благодарность нашему редактору Кейт Шуп (Kate Shoup), которая помогала нам на протяжении всей книги, и техническому редактору Эду Прайсу (Ed Price). Если бы не их дотошность, читать эту книгу было бы гораздо труднее. Если книга содержит меньше ошибок, чем наша первоначальная рукопись, это только их заслуга. А во всех оставшихся неточностях виноваты лишь мы.

Список опечаток и поддержка

Мы сделали все возможное, чтобы текст и сопутствующий контент к этой книге не содержали ошибок. Все неточности, которые были обнаружены после публикации издания, перечислены на сайте Microsoft Press по адресу: https://aka.ms/AnalyzeData/errata.

Если вы нашли опечатку, которая не указана в перечне, вы можете оповестить нас на той же странице.

Если вам требуется дополнительная помощь, направьте письмо в Microsoft Press Book Support по адресу: mspinput@microsoft.com.

Отметим, что услуги по поддержке программного обеспечения Microsoft по этому адресу не оказываются.

Обратная связь

Ваше удовлетворение от книги – главный приоритет для Microsoft Press, а ваша обратная связь – наш самый ценный актив. Пожалуйста, выскажите свое мнение об этой книге по адресу: https://aka.ms/tellpress.

Пройдите небольшой опрос, и мы прислушаемся ко всем вашим идеям и пожеланиям. Заранее благодарим за ваши отзывы!

Оставайтесь с нами

Давайте продолжим общение! Заходите на наш Twitter: @MicrosoftPress.

Глава 1

Введение в моделирование данных

Книга, которую вы держите в руках, посвящена *моделированию данных* (data modeling). Но перед тем как приступать к чтению, неплохо бы понять, зачем вам вообще нужно изучать моделирование данных. В конце концов, вы можете просто загрузить нужные данные в Excel и построить на их основе сводную таблицу. Так зачем вам еще что-то знать о моделировании данных?

К нам как к консультантам в этой области часто обращаются частные лица и компании, которые не могут рассчитать какие-то нужные им показатели. При этом они понимают, что все исходные данные для расчета у них есть, но либо формула получается чересчур сложной и запутанной, либо цифры не сходятся. В 99 % случаев причиной является неправильно спроектированная *модель данных* (data model). Если ее поправить, формула станет простой и понятной. Так что вам просто необходимо научиться моделировать данные, если вы хотите улучшить свои аналитические навыки и предпочитаете концентрироваться на принятии правильных решений, а не на поиске замысловатой формулы в справочнике по DAX.

Обычно считается, что моделирование данных – непростая тема для изучения. И мы не станем этого отрицать. Это действительно сложная область. Она потребует от вас серьезных усилий, к тому же вам нужно будет постараться перестроить сознание так, чтобы сразу мыслить категориями модели данных, рассуждая о возможных сценариях. Так что да, моделирование данных – тема непростая, ресурсоемкая и требующая немалых усилий в освоении. Иными словами, сплошное удовольствие!

В этой главе мы покажем вам несколько примеров того, как правильно спроектированная модель данных помогает облегчить написание итоговых формул. Конечно, это всего лишь примеры, и они могут не относиться напрямую к стоящим перед вами задачам. Но мы надеемся, что их будет достаточно для понимания того, почему стоит изучать моделирование данных. Быть хорошим специалистом по моделированию данных – значит уметь подгонять актуальную модель под шаблоны, изученные и решенные

другими. Ваша модель данных ничем не отличается от других. Да, в ней есть свои особенности, но высока вероятность, что до вас с подобными задачами уже кто-то сталкивался. Научиться выявлять сходства между вашим примером и моделями, описанными в книге, не так просто, но в то же время очень приятно. Когда вы достигнете успеха в этом, решения задач начнут появляться перед вами сами, а большинство проблем с расчетом нужных вам показателей просто исчезнут.

В основном в своих примерах мы будем использовать базу данных Contoso. Это вымышленная компания, торгующая электроникой по всему миру с использованием различных каналов продаж. Вероятно, вы ведете совершенно иной бизнес – в этом случае вам придется адаптировать отчеты под свои нужды.

Поскольку это первая глава, начнем мы с описания общей терминологии и концепции. Мы расскажем, что такое модель данных и почему в ней так важны связи. Также мы познакомимся с понятиями нормализации/денормализации и схемой «звезда». На протяжении всей книги мы будем описывать новые концепции на примерах, но в первой главе это будет наиболее заметно.

Пристегните ремни! Пришло время узнать все тайны о моделировании данных.

Работа с одной таблицей

Если вы используете Excel и сводные таблицы для анализа данных, велика вероятность, что вы загружаете информацию посредством запроса из какого-то источника – обычно из базы данных. После этого строите сводную таблицу и приступаете к анализу. Разумеется, при этом вы вынуждены мириться с некоторыми ограничениями Excel, главным из которых является лимит на количество строк в таблице, равный одному миллиону. Больше записей просто не поместится на рабочем листе. Честно говоря, в начале своего пути мы не рассматривали эту особенность как серьезный сдерживающий фактор. В самом деле, зачем кому-то может понадобиться загружать в Excel миллион строк, если можно воспользоваться базой данных? Причина может быть в том, что работа с Excel не требует от пользователя знаний в области моделирования данных, а с базой данных – требует.

Так или иначе, эта особенность Excel является существенным ограничением. В базе данных Contoso, которую мы используем в примерах, таблица продаж содержит 12 млн записей. Так что мы не можем просто взять и поместить их все на лист Excel. Но эта проблема легко решается. Вместо того чтобы загружать данные целиком, вы можете сгруппировать их, чтобы сократить количество строк. Если, допустим, вам необходимо проанализировать продажи в разрезе категорий и подкатегорий товаров, вы можете наложить соответствующие группировки, что существенно снизит объем загружаемой информации.

К примеру, разделение исходной таблицы из 12 млн строк на группы по производителю, бренду, категории и подкатегории с сохранением детализации продаж до дня позволило нам сократить количество записей до 63 984, что вполне приемлемо для загрузки на лист Excel. Написание запроса для выполнения подобной группировки – это задача для отдела ИТ или подходящего редактора запросов, если вы, конечно, не знаете язык SQL. Выполнив получившийся запрос, вы можете приступать к анализу. На рис. 1.1 можно видеть первые несколько строк после импорта данных в Excel.

FullDateLabel	Manufacturer	BrandName	ProductSubcategoryName	ProductCategoryName	SalesQuantity	SalesAmount	TotalCost
2007-03-31	Adventure Works	Adventure Works	Coffee Machines	Home Appliances	55	14332.268	7651.84
2008-10-22	Contoso, Ltd	Contoso	Cell phones Accessories	Cell phones	2040	23504.88	12648.94
2009-01-31	Adventure Works	Adventure Works	Televisions	TV and Video	194	51593.106	28146.4
2009-01-21	Fabrikam, Inc.	Fabrikam	Camcorders	Cameras and camcorders	282	163007.2	76709.45
2007-12-31	Adventure Works	Adventure Works	Laptops	Computers	29	14008.43	7944.32
2007-06-22	Contoso, Ltd	Contoso	Cell phones Accessories	Cell phones	680	6107.24	3420.44
2007-06-22	Proseware, Inc.	Proseware	Projectors & Screens	Computers	86	71417.6	30786.94
2007-08-23	Adventure Works	Adventure Works	Laptops	Computers	43	22672.2	9954.6
2009-03-30	The Phone Company	The Phone Company	Touch Screen Phones	Cell phones	198	48500.37	24164.56
2008-03-24	Contoso, Ltd	Contoso	Home & Office Phones	Cell phones	806	7358.594	3914.64
2007-09-30	Fabrikam, Inc.	Fabrikam	Microwaves	Home Appliances	44	4805.604	2824.24
2007-11-13	Adventure Works	Adventure Works	Desktops	Computers	153	47357.97	28256.02
2008-12-06	Contoso, Ltd	Contoso	Projectors & Screens	Computers	32	10790.4	6477.2
2007-11-14	Contoso, Ltd	Contoso	Digital SLR Cameras	Cameras and camcorders	146	55397.5	25876
2009-12-30	Adventure Works	Adventure Works	Desktops	Computers	32	15107.75	7952.97
2009-03-13	Wide World Importers	Wide World Importers	Recording Pen	Audio	42	7990.92	3607.26
2009-08-11	Wide World Importers	Wide World Importers	Recording Pen	Audio	9	1466.1	749.16
2009-09-28	Contoso, Ltd	Contoso	Microwaves	Home Appliances	78	9955.268	5189.27
2008-02-18	A. Datum Corporation	A. Datum	Digital Cameras	Cameras and camcorders	345	70989.93	32872.58
2007-08-15	Litware, Inc.	Litware	Washers & Dryers	Home Appliances	69	112603.8	56472.35

Рис. 1.1. Данные о продажах, сгруппированные для облегчения анализа

После загрузки таблицы в Excel вы можете наконец почувствовать себя как дома, создать сводную таблицу и приступать к анализу. На рис. 1.2 мы представили продажи по производителям для выбранной категории посредством обычной сводной таблицы и среза.

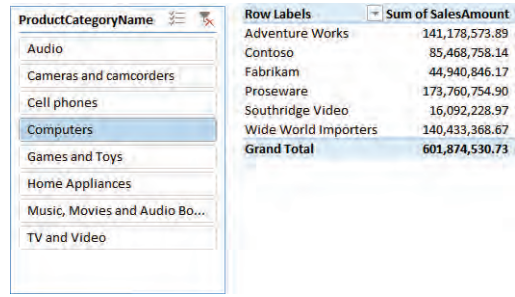


Рис. 1.2. На основании данных в Excel легко можно создать сводную таблицу

Верите вы или нет, но только что вы построили свою первую модель данных. Да, она состоит всего из одной таблицы, но тем не менее это модель данных. А значит, вы можете исследовать ее аналитический потенциал и искать способы для его повышения. У представленной модели есть одно серьезное ограничение – она содержит меньше строк, чем исходная таблица.

Будучи новичком в Excel, вы могли бы подумать, что лимит в миллион строк распространяется только на исходные данные, которые вы загружаете для дальнейшего анализа. И хотя это верно, важно также понимать, что данное ограничение автоматически переносится и на модель данных, что негативно сказывается на аналитическом потенциале отчетов. Фактически, для того чтобы сократить количество строк, вы вынуждены были производить группировку на уровне исходных данных и извлекать продажи, сгруппированные по определенным столбцам.

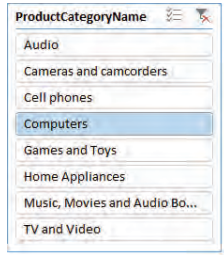
Таким образом, вы косвенно ограничили свои аналитические возможности. К примеру, вы не сможете провести аналитику по цвету товаров на основании полученной таблицы, поскольку информация об этой характеристике просто отсутствует. Добавить столбец к таблице – не проблема. Проблема в том, что при добавлении столбцов будет автоматически увеличиваться размер таблицы как в ширину (в количестве столбцов), так и в длину (в количестве строк). На практике одна строка для отдельной категории – например, аудиотехники (Audio) – превратится в несколько записей, каждая из которых будет содержать свой цвет для этой категории.

А если вы не сможете заранее решить, какие столбцы вам пригодятся для выполнения срезов, то вам придется загружать все 12 млн строк, а с таким объемом Excel не справится. Именно это мы имели в виду, когда говорили, что потенциал Excel в отношении моделирования данных невелик. Ограничение на количество импортируемых строк делает невозможным проведение анализа больших объемов данных.

Здесь вам на помощь приходит Power Pivot. Используя Power Pivot, вы не будете ограничены миллионом строк. Фактически количество записей, загружаемых в таблицу Power Pivot, ничем не ограничено. А значит, вы легко сможете импортировать в свою модель все продажи и проводить на их основании более глубокий анализ.

Примечание. Power Pivot доступен в Excel с версии 2010 в качестве внешней надстройки, а начиная с Excel 2013 включен в основной пакет. В Excel 2016 и следующих версиях Microsoft ввела новый термин для описания моделей Power Pivot: *модель данных Excel* (Excel Data Model). Однако термин Power Pivot по-прежнему широко используется.

Располагая полной информацией о продажах в одной таблице, вы можете проводить более детализированный анализ. К примеру, на рис. 1.3 вы видите сводную таблицу, построенную на основе модели данных Power Pivot со всеми загруженными столбцами. Теперь вы можете осуществлять срезы по категории товара, цвету и году, поскольку вся эта информация находится в модели. Чем больше столбцов, тем выше аналитический потенциал.



Sum of SalesAmount	Column Labels			
Row Labels	2007	2008	2009	Grand Total
Black	$59,783,936.63	$63,073,257.64	$70,489,997.35	$193,347,191.62
Blue	$5,128,405.12	$6,516,691.17	$7,715,161.42	$19,360,257.71
Brown	$6,301,722.60	$7,183,128.75	$4,904,738.33	$18,389,589.68
Gold		$39,185.40	$122,755.63	$161,941.03
Green	$1,747,237.19	$1,566,822.01	$2,159,190.14	$5,473,249.35
Grey	$6,318,419.68	$5,924,762.67	$6,410,704.05	$18,653,886.41
Orange		$12,800.63	$84,766.80	$97,567.43
Pink	$20,078.44	$43,870.83	$228,526.86	$292,476.13
Red	$6,926,045.96	$7,872,382.02	$11,495,613.50	$26,294,041.47
Silver	$43,375,399.72	$39,082,679.67	$36,471,502.90	$118,929,582.29
White	$64,963,894.39	$64,142,854.42	$70,639,615.97	$199,746,364.78
Yellow	$260,512.33	$330,165.82	$537,704.67	$1,128,382.82
Grand Total	$194,825,652.07	$195,788,601.04	$211,260,277.62	$601,874,530.73

Рис. 1.3. Если в модель данных загружены все столбцы, можно строить более интересные сводные таблицы

Этого примера достаточно, чтобы усвоить первый урок, касающийся модели данных: *размер имеет значение, поскольку он напрямую связан с гранулярностью*. Но что такое *гранулярность*? *Гранулярность* (granularity) – одна из важнейших концепций, описываемых в этой книге, и мы постараемся познакомить вас с ней как можно раньше. Далее в книге мы углубимся в изучение этой концепции, а сейчас позвольте дать простое описание термина гранулярность. В первом наборе данных вы сгруппировали информацию по категории и подкатегории, пожертвовав детальными данными ради уменьшения размера таблицы. Говоря техническим языком, вы установили гранулярность таблицы на уровне категории и подкатегории. Можете думать о *гранулярности* как об уровне детализации данных. Чем выше гранулярность, тем более детализированная информация будет доступна для анализа. В последнем рассмотренном наборе данных, загруженном в Power Pivot, гранулярность установлена на уровне товара (на самом деле она даже выше – на уровне каждой отдельной продажи), тогда как в предыдущем примере была на уровне категории и подкатегории. Возможности для детального анализа напрямую связаны с количеством доступных столбцов в таблице, а значит, с ее гранулярностью. Вы уже знаете, что увеличение количества столбцов непременно ведет к увеличению количества строк.

Выбрать правильный уровень гранулярности всегда непросто. При неверном выборе практически невозможно будет извлечь нужную информацию при помощи формул. У вас либо попросту не будет этих данных в таблице (как в примере с отсутствующим цветом товаров), либо эти данные будут разбросаны по всему набору. При этом неправильно будет говорить, что более высокий уровень гранулярности таблицы – это всегда хорошо. Нужно стремиться, чтобы гранулярность была установлена на оптимальном уровне с учетом ваших требований к дальнейшему анализу данных.

Мы уже рассматривали пример с потерянными данными. А что значит выражение «данные разбросаны по всему набору»? Проиллюстрировать такое поведение информации несколько сложнее. Представьте, к примеру, что вам необходимо получить средний годовой доход клиентов, покупаю-

щих определенный набор товаров. Такая информация в таблице присутствует – у нас ведь есть все сведения о наших покупателях. На рис. 1.4 показан фрагмент таблицы с нужными нам столбцами (необходимо открыть окно Power Pivot, чтобы увидеть содержимое таблицы).

ProductCategoryName	ProductSubcategoryName	ProductName	SalesAmount	FirstName	LastName	YearlyIncome
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Katrina	Xie	€ 20,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Seth	Rodriguez	€ 80,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Evelyn	Arun	€ 10,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Christy	Beck	€ 40,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Alejandro	Nara	€ 40,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Leah	Lu	€ 30,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Robyn	Torres	€ 20,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Jimmy	Moreno	€ 30,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Rafael	Cai	€ 20,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Jenny	Ferrier	€ 110,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Levi	Arun	€ 70,000.00
Cameras and camcorders	Digital SLR Cameras	A. Datum SLR Camera X137 Grey	$627.00	Randall	Torres	€ 40,000.00

Рис. 1.4. Информация о покупателях и товарах содержится в одной таблице

В каждой строке таблицы продаж в отдельном столбце указывается величина годового дохода клиента, купившего этот товар. В попытке вычислить средний годовой доход покупателя мы можем попробовать создать меру при помощи следующего кода на DAX:

```
AverageYearlyIncome := AVERAGE ( Sales[YearlyIncome] )
```

Созданная мера отлично работает, и вы можете использовать ее в сводной таблице, как это показано на рис. 1.5. Здесь мы видим средний годовой доход покупателей бытовой техники (Home Appliances) разных брендов.

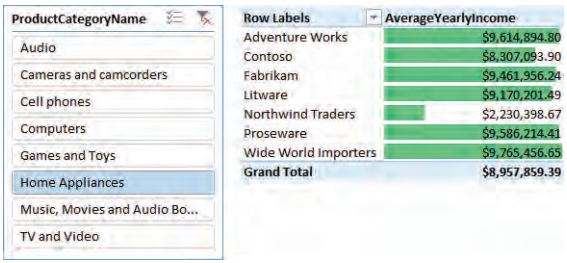


Рис. 1.5. Анализ среднего годового дохода покупателей бытовой техники

Отчет выглядит замечательно, но, к сожалению, цифры в нем не соответствуют действительности – они чересчур завышены. Фактически вы вычисляете среднее значение по таблице продаж с гранулярностью, установленной на уровне каждой продажи. Иными словами, в этой таблице содержатся строки для каждой продажи, а значит, покупатели в ней будут повторяться. Так, если покупатель приобрел три товара в разные дни, при подсчете среднего значения годовой доход для него будет учтен трижды, что приведет к ошибочным результатам.

Вы могли бы сказать, что таким образом получили средневзвешенную величину годового дохода. Но это не совсем так. Для того чтобы рассчитать средневзвешенное, нам необходимо было бы задать вес для каждой составляющей, а брать в качестве веса количество покупок было бы неправильно. Более логично было бы определить как вес количество купленных товаров, сумму покупки или еще какой-то значимый показатель. Кроме того, в данном примере мы планировали вычислять обычное среднее значение годового дохода покупателей, и созданная мера нам в этом ничуть не помогла.

И хотя это не так просто заметить, здесь мы также столкнулись с проблемой некорректно выбранной гранулярности. Получается, что информация, которая нам нужна, доступна, но не привязана к конкретному покупателю, а вместо этого разбросана по таблице продаж, что значительно затрудняет вычисления. Чтобы получить корректный результат, необходимо изменить гранулярность до уровня покупателя – либо путем повторной загрузки таблицы, либо воспользовавшись сложной формулой на языке DAX.

Если вы решите пойти по пути DAX, можно для вычисления среднего годового дохода воспользоваться следующей формулой, довольно сложной для понимания:

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

В этой не самой простой формуле мы сначала агрегируем продажи на уровне (гранулярности) покупателя, после чего применяем к результирующей таблице, в которой каждый покупатель появляется только один раз, функцию *AVERAGEX*. В примере мы применяем функцию *SUMMARIZE* для предварительной агрегации на уровне покупателя во временной таблице, а затем вычисляем среднее значение по `YearlyIncome`. Как видно по рис. 1.6, итоги правильного расчета среднего годового дохода сильно отличаются от наших прежних расчетов.

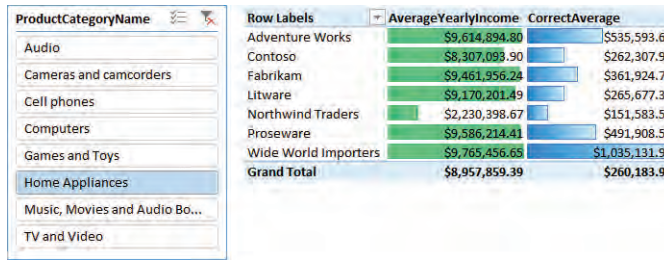


Рис. 1.6. При взгляде на результаты вычислений видно, как далеки мы были от истины

Необходимо хорошо усвоить один простой факт: сумма годового дохода – это величина, обладающая смыслом на уровне гранулярности покупателя. На уровне конкретной продажи этот показатель совершенно неуместен, хоть и показывает верные цифры. Иными словами, мы не можем использовать значение, актуальное на уровне покупателя, с тем же смыслом и на уровне продажи. Таким образом, чтобы получить верный результат, нам пришлось понижать гранулярность исходных данных, пусть и во временной таблице.

Из этого примера можно сделать пару важных выводов:

- правильная формула оказалась куда сложнее простого использования функции AVERAGE. Нам пришлось производить временную агрегацию, чтобы скорректировать гранулярность таблицы, поскольку нужная информация оказалась разбросана по всему набору данных, а не организована должным образом;
- вероятно, вам было бы непросто понять, что произведенные вами расчеты неверны. В нашем примере достаточно одного взгляда на рис. 1.6, чтобы заподозрить наличие ошибки – вряд ли у всех наших покупателей средний годовой доход превышает 2 млн долларов. Однако для более сложных расчетов выявить неточность может быть весьма проблематично, что приведет к появлению ошибок в вашей итоговой отчетности.

Необходимо повышать гранулярность таблицы, чтобы извлекать информацию нужной вам степени детализации, но если зайти в этом слишком далеко, могут возникнуть сложности с вычислением некоторых показателей. Как же выбрать правильный уровень гранулярности? Это непростой вопрос, и ответ на него мы прибережем на потом. Мы надеемся, что сможем научить вас выбирать оптимальный уровень гранулярности таблиц, но не забывайте, что это действительно сложная задача даже для опытных специалистов. А пока достаточно вводных слов о том, что из себя представляет гранулярность и как она важна для каждой таблицы в вашей модели данных.

На самом деле модели данных, которую мы до сих использовали в наших примерах, присуща одна серьезная проблема, отчасти связанная с гранулярностью. Основной ее недостаток состоит в том, что все данные у нас собраны в одной таблице. Если ваша модель, как в наших примерах, состоит из одной таблицы, то вам придется выбирать для нее гранулярность с учетом всех возможных видов отчетов, которые вы захотите формировать в будущем. Как бы вы ни старались, выбранная гранулярность никогда не будет идеально подходить для всех создаваемых вами мер. В следующих разделах мы рассмотрим вариант использования в модели данных сразу нескольких таблиц, что даст вам возможность оперировать более чем одним уровнем гранулярности.

Введение в модель данных

Из предыдущей главы вы узнали, что 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 – это позволило пользователям программы строить очень мощные модели данных.

Что же такое модель данных? *Модель данных* – это просто набор таблиц, объединенных *связями* (relationships). 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.

Создание модели данных вполне естественно при загрузке сразу нескольких таблиц. Более того, обычно информация импортируется из баз данных, обслуживаемых специалистами, которые уже создали модель данных за вас. Это означает, что ваша модель зачастую будет просто имитировать модель из источника данных. В таком случае ваша работа существенно упрощается.

К сожалению – и вы поймете это, читая книгу, – модель данных в источнике очень редко будет отвечать всем вашим требованиям в плане будущего анализа информации. Наша задача – на примерах с возрастающей сложностью научить вас проектировать собственную модель данных, отталкиваясь от источника. А чтобы упростить процесс обучения, мы будем знакомить вас с имеющимися техниками последовательно – от простого к сложному. И начнем с самых основ.

Для знакомства с концепцией модели данных загрузите таблицы Product и Sales из базы данных Contoso в модель Excel. После этого вы увидите диаграмму как на рис. 1.7 – с двумя таблицами и содержащимися в них столбцами.

Примечание. В Power Pivot вы можете получить доступ к диаграмме связей. Для этого выберите вкладку **Power Pivot** на ленте Excel и нажмите **Manage** (Управление). Далее на вкладке **Home** (В начало) окна Power Pivot нажмите **Diagram View** (Представление диаграммы) в группе **View** (Просмотр).

Рис. 1.7. В модель данных вы можете загружать несколько таблиц

Две несвязанные таблицы в представленном примере еще не являются полноценной моделью данных. Пока это просто две таблицы. Чтобы преобразовать их в осмысленную модель, необходимо установить связи между таблицами. В нашем примере обе таблицы содержат общее поле **ProductKey**. В таблице Product этот столбец представляет собой *первичный ключ* (primary key), что предполагает уникальность значений в нем и возможность идентифицировать по ним товары. В таблице Sales этот столбец служит иной цели, а именно для идентификации проданного товара.

Информация. В столбце, являющемся первичным ключом таблицы, содержатся уникальные значения для каждой записи. Таким образом, зная значение поля, вы можете однозначно идентифицировать его положение в таблице, то есть получить строку. При этом столбцов с уникальными значениями может быть несколько, и все они будут являться ключами. В первичном ключе нет ничего загадочного. С технической точки зрения он представляет собой столбец, уникально идентифицирующий строку в таблице. К примеру, в таблице покупателей первичным ключом может быть код покупателя, даже если поле с именем также содержит уникальные значения.

Если у вас есть уникальный идентификатор в одной таблице и поле в другой, ссылающееся на него, вы можете создать между этими двумя таблицами связь. Для правильной установки связи между таблицами оба условия должны выполняться. Если предполагаемое для создаваемой связи ключе-

вое поле хранит неуникальные значения, вам придется предварительно изменить модель данных при помощи определенных техник, описываемых в этой книге. А сейчас давайте на нашем примере поясним некоторые особенности связей:

- **таблица Sales называется *таблицей-источником* (source table).** Связь берет свое начало из таблицы Sales. Это означает, что для того, чтобы получить товар, вы всегда начинаете с продажи. Получив значение ключевого поля товара из таблицы Sales, вы ищете его в таблице Product. Теперь вы знаете, с каким товаром имеете дело, а также получаете доступ ко всем его атрибутам;
- **таблица Product называется *целевой* (target table) для этой связи.** Вы начинаете поиск с таблицы Sales и переходите к Product. Значит, таблица Product и есть цель устанавливаемой связи;
- **связь берет свое начало из таблицы-источника и направляется к целевой таблице.** Иными словами, у связи есть направление. Поэтому на диаграммах связь часто сопровождает стрелка, идущая от источника к цели. Но в разных программных продуктах графическое отображение связи свое;
- **таблица-источник также именуется в связи как *«многие»*.** Этим названием таблица обязана тому, что для каждого товара в таблице продаж может быть много записей, тогда как каждой продаже соответствует лишь один товар. По той же причине целевой таблице в связи отводится название «один». В этой книге мы будем пользоваться именно этой терминологией;
- **столбец ProductKey присутствует в обеих таблицах.** При этом в таблице Product это ключевое поле, а в таблице Sales – нет. По данной причине применительно к таблице Product мы называем поле **ProductKey** первичным ключом, тогда как в таблице Sales оно именуется внешним ключом. Под *внешним ключом* (foreign key) подразумевается столбец, указывающий на первичный ключ в другой таблице.

Все эти термины широко используются в области моделирования данных, и эта книга не станет исключением. Представив терминологию нашим читателям, мы будем использовать ее на протяжении всей книги. Но не волнуйтесь. В первых главах мы будем напоминать вам значение того или иного определения, пока вы к ним не привыкнете.

Используя Excel и Power BI, вы имеете возможность создавать связи путем перетаскивания мышью поля, являющегося внешним ключом (в нашем случае это ProductKey в таблице Sales), к первичному ключу (у нас это ProductKey в таблице Product). 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). На рис. 1.8 представлена соответствующая диаграмма из Power Pivot. Заметьте, что посередине линии все же присутствует стрелка,

но она не определяет направление связи. Вместо этого она служит совсем иным целям, а именно задает направление распространения фильтрации, о чем мы поговорим в следующих главах этой книги.

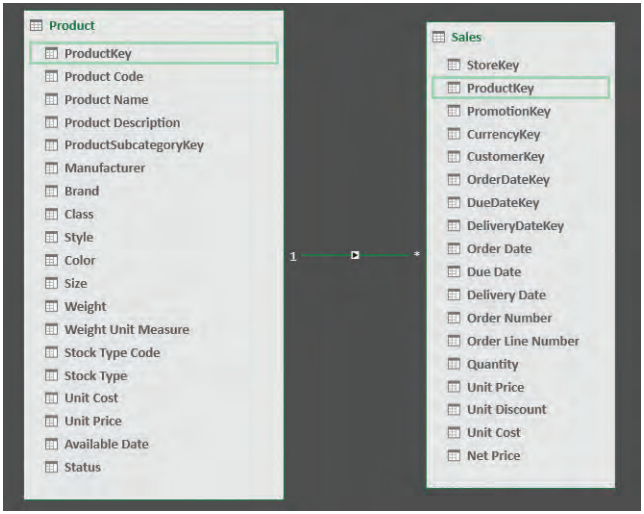


Рис. 1.8. Связь между таблицами представлена линией с индикаторами на концах («1» для одного и «звездочка» для многих)

После связывания таблиц вы можете осуществлять суммирование значений в таблице Sales, делая срезы по столбцам из таблицы Product. К примеру, как показано на рис. 1.9, вы можете использовать цвет товара (столбец Color из таблицы Product, как видно на рис. 1.8) в качестве среза при суммировании по количеству проданных товаров (столбец Quantity в таблице Sales).

Примечание. Если вы не видите вкладку Power Pivot в Excel, вероятно, произошла какая-то ошибка, в результате чего надстройка была отключена. Чтобы вновь активировать ее, нажмите на вкладке **File** (Файл) и выберите пункт **Options** (Параметры) на левой панели. В левой части окна **Excel Options** (Параметры Excel) нажмите на **Add-Ins** (Надстройки). После этого раскройте выпадающий список **Manage** (Управление), выберите пункт **COM Add-Ins** (Надстройки COM) и нажмите **Go** (Перейти). В окне **COM Add-Ins** (Надстройки для модели компонентных объектов (COM)) выберите **Microsoft Power Pivot for Excel**. В том случае, если этот пункт выбран, снимите выделение. После этого нажмите **OK**. Если вы снимали выделение пункта **Microsoft Power Pivot for Excel**, вернитесь в окно **COM Add-Ins** и снова выберите его. Вкладка **Power Pivot** должна появиться на ленте.

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

Это был ваш первый пример модели данных, состоящей из двух таблиц. Как мы уже сказали, модель данных – это просто набор таблиц (в нашем случае Sales и Product), объединенных связями. Перед тем как идти дальше, давайте уделим еще немного времени гранулярности – на этот раз применительно к модели из нескольких таблиц.

В первом разделе этой главы вы уяснили, насколько важно (и сложно) определить правильный уровень гранулярности для конкретной таблицы. При неправильном выборе гранулярности дальнейшие расчеты в этой таблице существенно усложнятся. А что можно сказать о гранулярности в новой модели данных, состоящей из двух таблиц? В этом случае вы столкнетесь с задачей иного характера, решить которую будет в каком-то смысле проще, но понять – сложнее.

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.

Более того, в этом случае совершенно естественно будет установить гранулярность в таблице Sales на уровне продажи, а в таблице Product – на уровне товара. Вспомните первый пример из этой главы. У нас была одна таблица продаж с гранулярностью, установленной на уровне категории и подкатегории товара. Причиной было то, что информация о категории и подкатегории товара хранилась в таблице Sales. Иными словами, *вам необходимо было принимать решение по поводу гранулярности, потому что*

данные располагались не на своем месте. Когда все находится там, где нужно, гранулярность уже не доставляет таких хлопот.

По своей сути категория является атрибутом товара, а не продажи. Да, в определенном смысле категорию можно назвать и атрибутом продажи, но лишь потому, что продажа относится к конкретному товару. 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.

Информация. В хорошо спроектированной модели данных гранулярность каждой таблицы установлена правильно, что делает структуру одновременно более простой и эффективной. Все дело в связях – полноту их мощи вы почувствуете, когда начнете мыслить категориями модели из нескольких таблиц и избавитесь от однотабличного подхода, характерного для работы в Excel.

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. Фактически в базе данных категории и подкатегории товаров разделены на две таблицы. Загрузив в модель данных обе таблицы и правильно построив связи, вы увидите на диаграмме в Power Pivot схему, показанную на рис. 1.10.

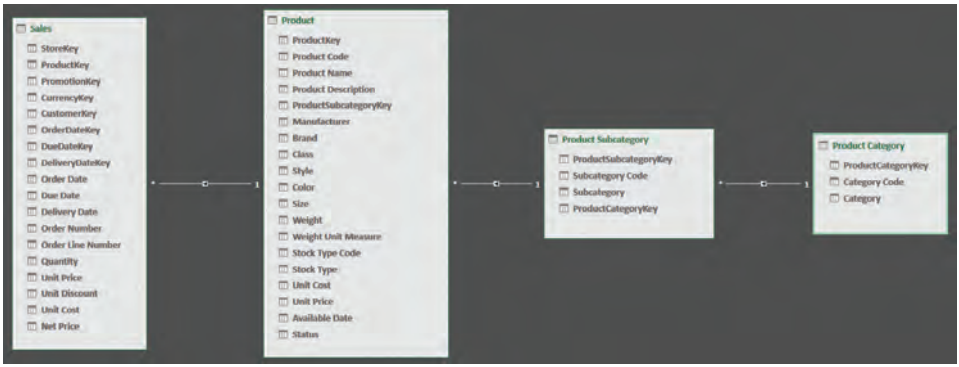


Рис. 1.10. Категории и подкатегории товаров хранятся в разных таблицах, к которым можно обратиться посредством связей

Как видите, информация о товарах разнесена сразу на три таблицы: Product, Product Subcategory и Product Category. Таким образом, образуется це-

лая цепочка связей, начиная с Product, через Product Subcategory и к Product Category.

Что послужило причиной выбора такого подхода к проектированию модели? Поначалу кажется, что это чересчур усложненный способ для хранения довольно простой информации. Однако у этой техники есть целый ряд преимуществ, пусть и не столь очевидных с первого взгляда. Вынос категории товара из таблицы продаж позволяет хранить название категории, к которой могут принадлежать сразу несколько товаров, в единственной строке таблицы Product Category. Это правильный способ хранения информации сразу по двум причинам. Во-первых, это позволяет сохранить место на диске из-за отсутствия необходимости хранить дублирующуюся 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* (normalization). Говорят, что атрибут таблицы (вроде нашей категории товара) нормализован, если он вынесен в отдельную таблицу, а на его место помещен ключ, ссылающийся на эту таблицу. Это широко распространенная техника, которую используют архитекторы баз данных при проектировании моделей. Обратная техника, заключающаяся в хранении атрибутов в таблице, которой они принадлежат, носит название *денормализация* (denormalization). 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.

Вас, должно быть, интересует, почему разработчик базы данных Contoso решил хранить атрибуты категории и подкатегории товаров в отдельных таблицах (то есть в нормализованном виде), а цвет, наименование производителя и бренд – в таблице Product (без применения нормализации). В этом конкретном случае ответ прост: Contoso – это демонстрационная база данных, и на ее примере хотелось показать все возможные техники. На практике вы будете встречаться как с преимущественно нормализованными, так и с денормализованными моделями в зависимости от особенностей использования базы данных. Будьте готовы к тому, что одни атрибуты будут нормализованы, а другие – нет. Это вполне приемлемо для моделирования данных, поскольку здесь есть разные методы и подходы. К тому же вполне возможно, что разработчик базы данных был вынужден принимать то или иное решение по структуре модели уже в процессе работы.

Модели с высокой степенью нормализации обычно используются в *системах обработки транзакций в реальном времени* (online transactional processing systems – OLTP). Такие базы данных спроектированы специально для выполнения ежедневных оперативных действий вроде обслуживания подготовки счетов, размещения заказов, доставки товаров или создания

и удовлетворения заявок. Нормализация здесь используется как способ сокращения занимаемого на диске места (что обычно ведет к увеличению быстродействия базы данных) и повышения эффективности операций вставки и обновления информации, характерных для OLTP-систем. В ежедневной работе компании часто выполняются операции обновления данных (например, о покупателях), и хочется, чтобы обновленная информация мгновенно распространялась на все таблицы, связанные с покупателями. Этого можно добиться путем нормализации соответствующих атрибутов. 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.

OLTP-системы зачастую насчитывают сотни таблиц, поскольку почти каждый атрибут хранится в отдельной таблице. Применительно к товарам, допустим, можно было бы завести таблицы для хранения производителей, брендов, цветов и прочего. В результате хранение простой сущности вроде товаров вылилось бы в 10–20 отдельных таблиц, объединенных связями. Разработчик такой базы данных с гордостью назвал бы свое детище «хорошо спроектированной моделью данных» и, несмотря на некоторые ее странности, был бы прав. Для OLTP-систем нормализация почти всегда будет оптимальным выбором.

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, как на рис. 1.11.

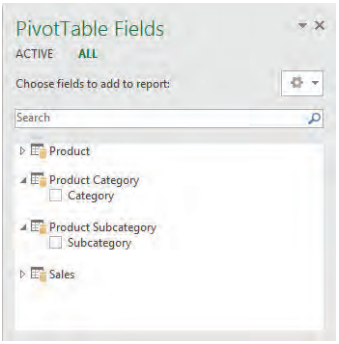


Рис. 1.11. В списке полей сводной таблицы, построенной на основании нормализованной модели данных, слишком много таблиц – легко запутаться

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. Так что хоть нор-

мализация и является оптимальным выбором для OLTP-систем, для нужд аналитики она обычно не подходит. Когда вы формируете отчеты, вам не должны быть интересны технические подробности хранения информации о товарах. Вам будет удобнее, если категория и подкатегория будут представлены как столбцы в таблице Product – это более привычно для анализа данных.

Примечание. В этом примере мы намеренно скрыли некоторые бесполезные столбцы вроде первичных ключей, что является хорошей практикой. В противном случае вы бы видели множество полей, что затруднило бы процесс анализа. Представьте себе, как бы выглядел список полей, если бы информация о товарах хранилась в десяти таблицах. Вам бы пришлось немало потрудиться, чтобы найти нужный столбец для вывода в отчет.

В процессе создания модели данных для нужд аналитики вам необходимо прийти к оптимальному уровню денормализации данных вне зависимости от того, как информация хранится в базе физически. Как вы уже видели, излишняя денормализация может привести к проблемам с определением гранулярности таблиц. Позже вы узнаете, какие еще негативные последствия влечет за собой чрезмерное увлечение денормализацией. Какую же степень денормализации можно считать оптимальной?

Для ответа на этот вопрос нет какого-то единого правила. Вы должны интуитивно дойти до такого уровня денормализации, при котором структура таблицы станет самодостаточной и будет полностью описывать хранящуюся в ней сущность. В нашем примере необходимо перенести столбцы Product Category и Product Subcategory в таблицу Product, поскольку они являются атрибутами товаров и вам не хотелось бы видеть их в отдельных таблицах. При этом не следует денормализовывать информацию о товарах в таблице Sales, поскольку товары и продажи – это разные сущности. Конкретная продажа напрямую связана с товаром, но нельзя сказать, что она составляет с ним единое целое.

На этом этапе вы можете рассматривать модель данных, состоящую из единственной таблицы, как чрезмерно денормализованную. Это так и есть. Вспомните, мы задумывались о том, чтобы установить гранулярность на уровне товара в таблице Sales, что изначально неправильно. В корректно спроектированной модели данных с оптимальной степенью денормализации проблемы с гранулярностью решаются сами собой. Если же модель излишне денормализована, начинаются неприятности с правильным выбором уровня гранулярности.

Введение в схему «звезда»

До сих пор мы имели дело с очень простыми моделями данных, состоящими из товаров и продаж. В реальном мире такие модели практически не встре-

чаются. 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. Например, в определенный день сотрудник, работающий на складе, продал товар конкретному покупателю.

Конечно, каждый бизнес подразумевает свои информационные активы, и события у всех разные. Но если мыслить в общем, то почти в любом виде деятельности будет прослеживаться четкое разделение на активы и события. К примеру, в случае с медицинским учреждением активами могут быть пациенты, заболевания и лекарственные препараты, тогда как к событиям мы причислим постановку диагноза и прием лекарственного средства пациентом. В системе приема заявок к активам могут относиться клиенты, заявки и время, а события генерируются в процессе изменения статуса заявок. Подумайте о виде деятельности, которым занимаетесь вы. Наверняка вам также удастся выделить в своей области активы и события.

Такое разделение делает возможным применение специальной техники моделирования данных, получившей название *схема «звезда»* (star schema). В этой схеме все сущности (таблицы) подразделяются на две категории:

- **измерения.** *Измерение* (dimension) является информационным активом: товар, покупатель, сотрудник или пациент. Измерения содержат *атрибуты* (attribute). К примеру, атрибутами товара являются его цвет, категория, подкатегория, производитель и цена. У пациента это имя, адрес и дата рождения;
- **факты.** *Факт* (fact) – 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* (measures) – числовые показатели, которые можно агрегировать при анализе состояния бизнеса. Это может быть количество или сумма проданного товара, размер скидки и прочее.

После мысленного разделения таблиц на две категории становится ясно, что факты связаны с измерениями. Каждому отдельному товару в таблице продаж соответствует несколько строк. Иными словами, между таблицами Sales и Product есть связь, в которой Product соответствует стороне «один», а Sales – стороне «многие». Если вы расположите на диаграмме в Power Pivot все измерения вокруг единственной таблицы фактов, то получите типичную форму звезды, показанную на рис. 1.12.

Схема «звезда» легка для чтения, понимания и использования. Измерения используются для осуществления срезов данных, тогда как сама агрегация числовых показателей выполняется в таблице фактов. Удобство этой модели еще и в том, что в списке полей сводной таблицы будет не так много сущностей.

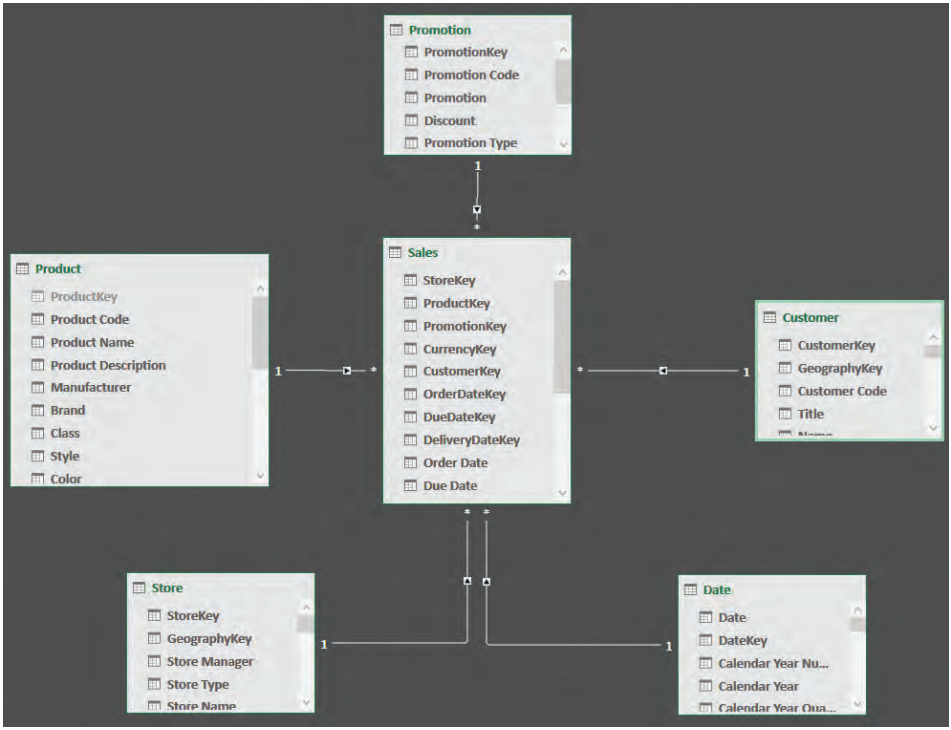


Рис. 1.12. Схема «звезда» приобретает свои очертания после расположения измерений вокруг таблицы фактов

Примечание. Схема «звезда» получила широкое распространение в области хранилищ данных. Сегодня такая модель считается стандартом представления информации для нужд аналитики.

По своей природе таблицы измерений содержат не так много строк – меньше миллиона, а обычно в интервале от нескольких сотен до нескольких тысяч. Таблицы фактов, напротив, чаще всего очень объемные и хранят десятки и сотни миллионов записей. В целом же схема «звезда» получила столь широкую популярность, что большинство систем управления базами данных сегодня оптимизированы в плане производительности именно под ее использование.

Совет. Прежде чем читать дальше, попробуйте представить, как ваша собственная бизнес-модель может быть реализована с использованием схемы «звезда». Не стоит на данном этапе пытаться спроектировать идеальную модель, но размышление над этой задачей поможет вам в будущем лучше оперировать таблицами измерений и фактов.

Важно привыкнуть к схеме «звезда». Посредством нее ваши данные будут представлены в наиболее удобном виде. Кроме того, терминология, применяемая в этой схеме, очень широко используется в сфере бизнес-аналитики (BI), и эта книга – не исключение. Мы часто употребляем термины измерение и таблица фактов, чтобы подчеркнуть разницу между маленькими и большими таблицами. В следующей главе мы будем говорить о главных и подчиненных таблицах, попутно решая задачу установления связей между разными таблицами фактов. И к тому моменту мы будем считать, что вы уже хорошо усвоили разницу между таблицей фактов и измерением.

Стоит отметить несколько важных особенностей устройства схемы «звезда». Одной из них является то, что таблицы фактов могут быть объединены связями с измерениями, тогда как измерения не должны быть связаны между собой. Чтобы проиллюстрировать важность этого правила и показать, что бывает, если ему не следовать, предположим, что мы добавили в модель новое измерение Geography, содержащее географические данные, такие как город, штат и страну/регион рождения. Оба наших измерения Store и Customer могут быть объединены связью с Geography. В итоге у нас могла бы получиться модель, представленная на рис. 1.13 в виде диаграммы Power Pivot.

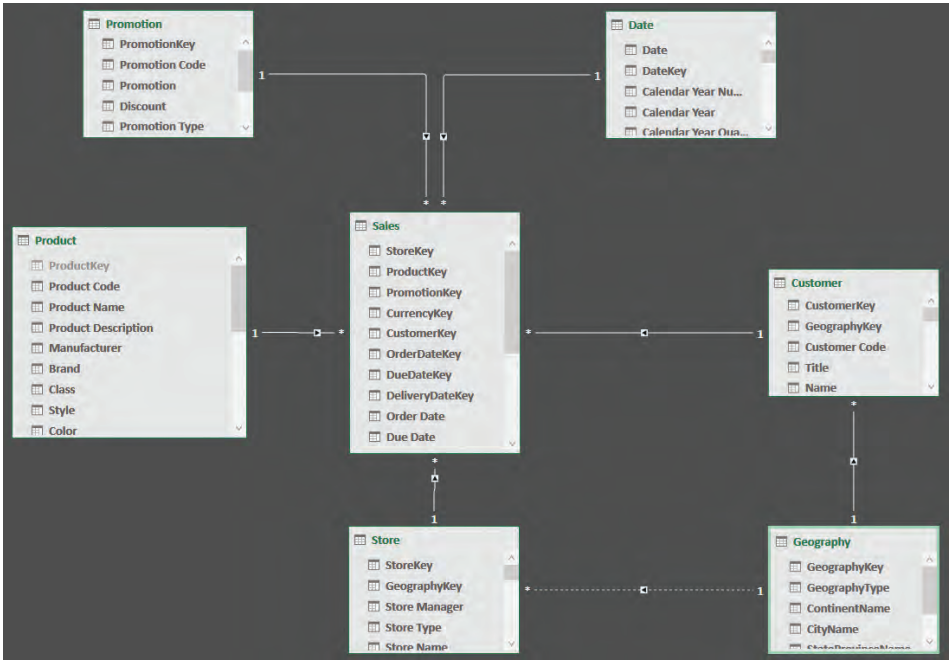


Рис. 1.13. Новое измерение Geography объединено связями с Customer и Store

В этой модели нарушено правило, запрещающее наличие связей между измерениями. По сути, все три таблицы – Customer, Store и Geography – являются измерениями, но при этом они связаны. Что плохого в такой модели? А то, что она вносит *неоднозначность* (ambiguity).

Представьте, что вы делаете срез данных по городу в надежде посчитать количество проданных товаров. В результате запрос может пройти по связи между таблицами Geography и Customer и вернуть количество товаров, проданное покупателям из выбранного города. А если пройти по связи между Geography и Store, то мы получим продажи со склада из этого города. Есть и третий вариант – использовать обе связи и выяснить, какое количество товаров было продано покупателю из выбранного города, со склада, расположенного там же. У нас получилась неоднозначная модель данных, и понять, какие цифры она выдает, крайне проблематично. И это не только техническая проблема, но и логическая. Пользователь, который будет работать с этой моделью, будет сбит с толку и не сможет понять, что значат цифры в отчетах. И именно по причине ее неоднозначности ни Excel, ни Power BI не позволят вам создать подобную модель. В следующих главах мы будем рассматривать вопросы неоднозначности моделей более подробно. Пока же важно знать, что Excel (а именно в нем создавался этот пример) сделал созданную связь между таблицами Store и Geography неактивной, чтобы не допустить неоднозначности в модели данных.

Как разработчик модели вы должны всеми способами стараться избегать неоднозначности. Как избавить рассматриваемую нами модель от неоднозначности? Ответ очень прост. Необходимо провести денормализацию модели – перенести нужные колонки из таблицы Geography в Store и Customer, а само измерение с географией удалить из модели. Также вы могли бы включить в измерения колонку ContinentName с названием континента, и получилась бы модель, представленная на рис. 1.14.

Проведя денормализацию модели, мы избавили ее от неоднозначности. Теперь пользователи смогут осуществлять срезы данных, используя географические признаки из таблицы Customer или Store. В итоге Geography – это то же измерение, но для возможности полноценного использования схемы «звезда» нам пришлось его денормализовать.

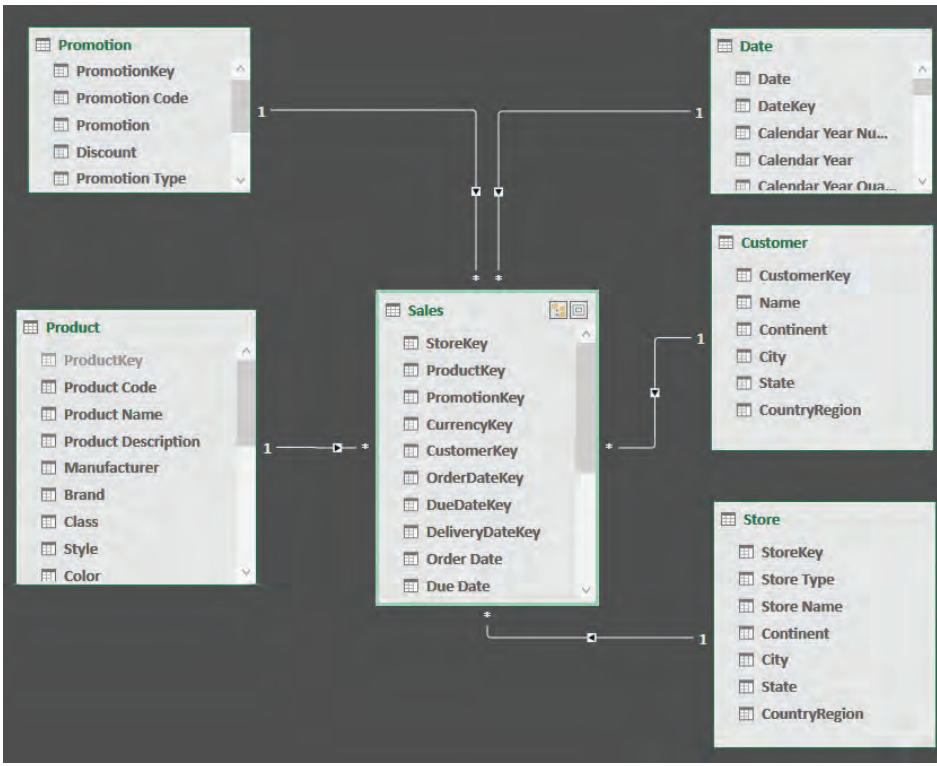


Рис. 1.14. После денормализации колонок из Geography модель вернулась к схеме «звезда»

Напоследок хотелось бы познакомить вас с еще одним термином, который будет часто использоваться в книге, – снежинка. *Схема «снежинка»* (snowflake schema) является разновидностью «звезды» с тем исключением, что некоторые измерения не связаны с таблицей фактов напрямую. Вместо этого они объединены с ней посредством других измерений. Вы уже встречались с такой схемой на страницах этой книги, и мы вновь представим вам ее на рис. 1.15.

Нарушает ли схема «снежинка» правило, запрещающее установку связей между измерениями? В каком-то смысле да, ведь таблицы Product Subcategory и Product представляют собой измерения, и при этом они объединены связью. Отличие этого примера от предыдущего состоит в том, что эта связь является единственной, соединяющей таблицу Product Subcategory с другими измерениями, объединенными с таблицей фактов, или таблицей Product. Так что вы можете рассматривать таблицу Product Subcategory как измерение, объединяющее в группы различные товары, но при этом не группирующее содержимое других измерений или таблицы фактов. То же самое верно и для таблицы Product Category. Таким образом,

хотя схема «снежинка» и нарушает указанное выше правило, она не создает в модели данных неоднозначности, а значит, с ней все в порядке.

Рис. 1.15. Измерения Product Category, Subcategory и Product образуют цепочку связей в виде снежинки

Примечание. Образования схемы «снежинка» можно избежать путем денормализации колонок из дальних таблиц в измерения, непосредственно связанные с таблицей фактов. Но иногда представление данных в виде снежинки бывает оправданным, и если не считать небольших проблем с производительностью, других недостатков у него нет.

Как вы узнаете из этой книги, в большинстве случаев схема «звезда» будет лучшим выбором для вашей модели данных. Да, изредка будут встречаться сценарии, в которых такое представление будет неоптимальным. И все же каждый раз, когда вы будете работать с моделью данных, рассматривайте в качестве приоритетной схему «звезда». Даже если она окажется неидеальной в данной конкретной ситуации, она будет близка к идеалу.

Примечание. В процессе изучения моделирования данных в какой-то момент вам может показаться, что лучше отойти от применения схемы «звезда». Не делайте этого. Есть целый ряд причин, по которым схема «звезда» в подавляющем большинстве случаев будет оптимальным выбором. К сожалению, многие из этих причин становятся очевидными только с приобретением опыта в сфере проектирования моделей данных. Если у вас пока такого опыта нет, доверьтесь десяткам тысяч профессионалов в области бизнес-аналитики по всему миру, которые прекрасно знают, что схема «звезда» будет лучшим выбором почти всегда – какой бы специфики ни касалась модель данных.

Понимание важности именования объектов

При построении модели данных вы обычно загружаете информацию из базы данных SQL Server или других источников данных. Велика вероятность, что разработчик базы данных в процессе именования объектов пользовался определенным соглашением. В наше время существует великое множество соглашений об именовании объектов – мы не сильно ошибемся, если скажем, что свое соглашение есть сегодня буквально у каждого.

Многие разработчики при проектировании модели данных предпочитают использовать префикс Dim для названий измерений и Fact для таблиц фактов. Так что сегодня зачастую можно встретить таблицы с названиями DimCustomer и FactSales. Другие предпочитают делать различия между представлениями и физическими таблицами, используя префиксы Vw и Tbl соответственно. А кто-то считает, что буквенного обозначения недостаточно для полной ясности и добавляет цифры – получается что-то вроде Tbl_190_Sales. Продолжать можно до бесконечности, но суть вы уловили. Стандартов именования масса, и у каждого есть свои плюсы и минусы.

Примечание. Можно поспорить с уместностью применения подобных стандартов при именовании объектов в базах данных, но эта дискуссия выйдет далеко за пределы данной книги. Так что мы ограничимся обсуждением использования соглашений об именовании в моделях данных, которые вы создаете и просматриваете в Power BI и Excel.

Вы не обязаны при именовании объектов следовать каким-либо техническим стандартам – достаточно будет здравого смысла и обеспечения легкости использования в дальнейшем. Например, мало кому доставит удовольствие работа с моделью данных, в которой таблицы носят названия VwDimCstmr или Tbl_190_FactShpmt. Это очень странные и малопонятные наборы символов, но, признаться, мы до сих пор встречаемся с подобными именами объектов в моделях данных. И это мы говорим только о правилах именования таблиц. Когда речь заходит о столбцах, все становится совсем плохо. Единственный наш совет заключается в том, чтобы использовать легко читающиеся названия, ясно описывающие измерение или таблицу фактов.

На протяжении лет мы спроектировали множество аналитических систем и за это время выработали очень простой свод правил по именованию таблиц и столбцов:

- **наименование измерения должно состоять только из названия актива в единственном или множественном числе.** Так, к примеру, таблица со списком покупателей может называться Customer или Customers. Информация о товарах должна храниться в таблице с на-

званием Product или Products. Мы считаем, что единственное число лучше подходит для именования измерений, поскольку оно идеально сочетается с запросами на естественном языке в Power BI;

- **если название актива состоит из нескольких слов, используйте для их разделения прописные буквы.** К примеру, категории товаров могут храниться в таблице с названием ProductCategory, а страна отгрузки может именоваться CountryShip или CountryShipment. Вместо разделения слов прописными буквами допустимо использовать обычные пробелы – например, таблица может называться Product Category. Здесь есть только один минус – код на языке DAX может немного усложниться. Но все это на ваше личное усмотрение;
- **для имени таблицы фактов необходимо использовать название фактической операции и всегда применять множественное число.** Так, факты продаж можно хранить в таблице с названием Sales, а факты закупок, как вы уже догадались, – в таблице Purchases. Если вы будете использовать для фактов исключительно множественное число, то 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 имен объектов.** Названия вроде CountryOfShipmentOfGoodsWhenSoldByReseller могут приводить в замешательство. Никому не интересно будет читать такие длинные имена. Вместо этого лучше подобрать уместную аббревиатуру, попутно исключив лишние слова;
- **избегайте использования слишком коротких имен.** Все любят использовать в своей речи сокращения. И если в повседневном общении это приемлемо и забавно, то в отчетах часто бывает неуместно и вносит неразбериху. К примеру, вы могли бы использовать для обозначения страны отгрузки для торговых посредников (country of shipment for resellers) аббревиатуру CSR, но ее будет очень трудно запомнить тем, кто не работает с вами изо дня в день. Помните о том, что отчеты могут использоваться самыми разными пользователями, многие из которых не имеют понятия о привычных для вас сокращениях;
- **ключевой атрибут в измерении должен содержать название таблицы и окончание Key.** Например, первичный ключ в таблице Customer должен называться CustomerKey. То же самое касается и внешних ключей. Так что в будущем вы сможете легко определять внешние поля по окончанию Key и нахождению в таблице с другим именем. Допустим, поле CustomerKey в таблице Sales является внешним ключом, ссылающимся на таблицу Customer, где оно выступает в качестве первичного ключа.

Как видите, правил немного. Все остальное – на ваше усмотрение. При выборе названий для остальных столбцов полагайтесь на здравый смысл. Хорошо именованной моделью данных легко и просто делиться с другими. Кроме того, при следовании этим простым правилам вам будет легче обнаружить ошибки и неточности в своей модели данных.

Совет. Если сомневаетесь по поводу именования того или иного объекта, спросите себя, поймет ли кто-нибудь выбранное вами имя таблицы или столбца. Не думайте, что вы один будете пользоваться своими отчетами. Рано или поздно вам захочется поделиться ими с человеком, обладающим иными фоновыми знаниями. Если он без труда сможет понять названия объектов в вашей модели, значит, вы на правильном пути. В противном случае вам лучше пересмотреть свои принципы именования.

Заключение

В этой главе вы познакомились с основами моделирования данных, а именно:

- одна таблица – это уже модель данных, пусть и в ее простейшей форме;
- при наличии единственной таблицы вы должны правильно выбрать ее гранулярность. Это облегчит написание формул в будущем;
- разница между моделью с одной таблицей и несколькими состоит в том, что во втором случае таблицы объединены между собой посредством связей;
- любая связь характеризуется стороной с одним элементом и многими – этот показатель говорит о том, сколько строк вы обнаружите, проследовав по связи в этом направлении. Поскольку один товар может присутствовать сразу в нескольких продажах, в соответствующей связи таблица Product будет представлять один элемент, а Sales – многие;
- в целевой для связи таблице обязательно должен присутствовать первичный ключ – колонка с уникальными значениями, однозначно определяющими каждую строку. При отсутствии первичного ключа связь к этой таблице установить невозможно;
- нормализованной моделью данных называется модель, в которой информация хранится в компактном виде, без повторения значений в разных строках. Обычно нормализация модели ведет к образованию большого количества таблиц;
- денормализованная модель данных характеризуется множеством повторений значений в строках (например, слово Red (красный) в такой модели может встречаться многократно – для каждого товара красного цвета), но при этом содержит меньшее количество таблиц;

- нормализованные модели данных обычно используются в OLTP-системах, тогда как денормализация зачастую применяется к моделям, предназначенным для анализа информации;
- в типичной аналитической модели можно провести четкие различия между информационными активами (измерениями) и событиями (фактами). Разделяя сущности на измерения и факты, мы в конечном счете выстраиваем структуру модели в виде звезды. Схема «звезда» является наиболее распространенной архитектурой аналитических моделей данных по одной простой причине – она отлично работает в подавляющем большинстве случаев.