Проектирование баз данных по ERD методологии

Основные компоненты ER диаграммы - сущности, атрибуты, ключи, связи

1. Понятие сущности

Как вы знаете, **сущности** - это понятия, информацию о которых следует сохранять для возможности дальнейшей обработки. В **ERD сущности** являются графическим представлением логической группировки данных. Сущности могут быть вещественными, реальными объектами или неосязаемыми концептуальными абстракциями. Сущности не предназначены для представления единичного объекта, они представляют классы, включающие атрибуты, содержащие информацию о множестве экземпляров.

Ниже будут рассмотрены следующие вопросы, касающиеся сущностей:

- Диаграмма Сущность-Связь (Entity Relationship Diagram)
- Выделение сущностей
- Определение типов сущностей
- Именование и описание сущностей
- Распространенные ошибки при работе с сущностями

1.1. Введение в реляционную диаграмму сущности

Далее для визуального представления сущностей и отношений между ними используются **ERD** (Entity Relationship Diagram - диаграмма сущность-связь), основанная на нотации, используемой ERwin. Хотя существуют и другие методологии моделирования данных, такие как расширенный реляционный анализ (Extended Relational Analysis - **ERA**), объектно-ориентированный подход (Object Oriented - **OO**) и объектно-ролевое моделирование (Object Role Modeling - **ORM**), но, фундаментальные концепции ERD методологии присутствуют и в них.

Методология ER-моделирования разработана П. Ченом в конце 1970-х годов. Для представления сущностей в методологии ER используются прямоугольники. В исходной ER-нотации Чена отношения содержат атрибуты. Равная возможность использования атрибутов в сущностях и отношениях делает различие между сущностями и отношениями достаточно сложным. С течением времени ER-подход изменялся и расширялся, но базовые концепции продолжали обеспечивать надежную основу для грамотного моделирования данных.

1.2. Что такое сущность?

Сущность - это физическое представление логической группировки данных. Сущности могут быть вещественными, реальными объектами, такими как ПЕРСОНА или МОРОЖЕНОЕ, или неосязаемыми концептуальными абстракциями как ЦЕНТР ЗАТРАТ или РЫНОК. Сущности не предназначены для представления единичного объекта, они представляют набор экземпляров, содержащих информацию, представляющую интерес с точки зрения их уникальности. Например, сущность ПЕРСОНА представляет собой экземпляр объектов типа Персона. Иван Петров, Мария Русанова и Савелий Богданов - конкретные примеры экземпляров сущности ПЕРСОНА. Конкретный экземпляр сущности представляется строкой таблицы и идентифицируется первичным ключом.

Сущность имеет следующие признаки:

- Она имеет имя и описание.
- Она представляет класс, а не единичный экземпляр абстракции.
- Ее конкретные представители (экземпляры) могут быть уникально идентифицированы.
- Она содержит логическую группировку атрибутов, представляющих информацию, интересную с точки зрения корпорации.

Формальные определения сущности. Ниже приведен список определений сущности признанных авторитетов в области моделирования данных. Обратите внимание на их сходство:

- Чен (1976): "Вещь, которая может быть однозначно идентифицирована".
- Дейт (1986): "Любой различимый объект, который будет представлен в базе данных".
- Финклештейн (1989): "Информационная сущность представляет некую 'вещь', которая сохраняется для дальнейших ссылок. Термин сущность относится к логическому представлению данных".

1.3. Выделение сущностей

Как приступить к процессу выделения сущностей? Большинство сущностей выявляются в ходе рабочих сессий и интервью. Анализ требований к информации, полученной от экспертов в предметной области и конечных пользователей - вот наилучший источник информации. Другим хорошим источником является корпоративная модель.

Обратите внимание на имена существительные и имена объектов - вполне возможно, что они станут логическими сущностями. Старайтесь не представлять единичные экземпляры в виде сущностей, как это часто бывает, когда сущности моделируются в терминах роли. Моделирование сущностей в терминах роли - достаточно распространенная ошибка. Сущности появляются и в процессе нормализации. Приведение логической модели к третьей нормальной форме вероятнее всего приведет к появлению нескольких дополнительных сущностей.

Существует две основных группы сущностей: **зависимые и независимые**. Независимая сущность не нуждается в информации из другой сущности для идентификации уникального экземпляра. Она представляется в **ERwin** в виде прямоугольника. Первичный ключ независимой сущности не включает в себя первичных ключей других сущностей.

Зависимая сущность должна привлекать информацию из другой сущности для идентификации уникального экземпляра. Она представляется на ER-диаграмме в виде прямоугольника с закругленными углами. Первичный ключ зависимой сущности включает первичные ключи одной или более родительских сущностей.

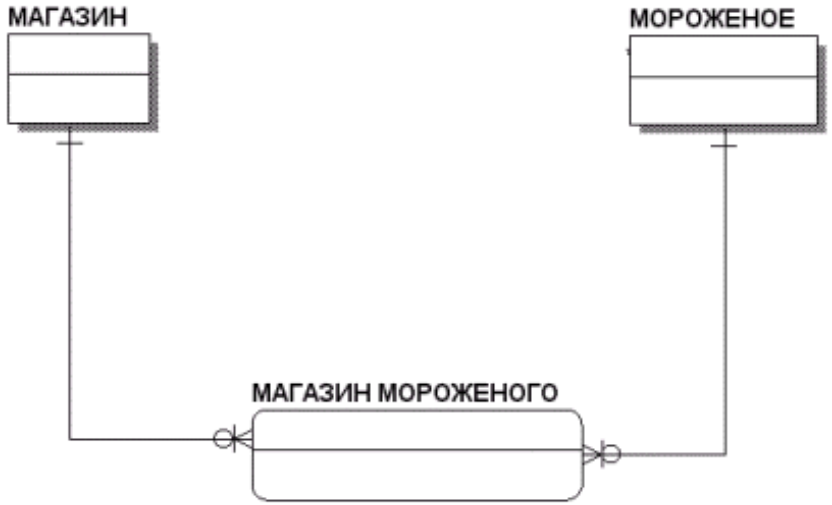

Рис. 2.1. Примеры стержневых сущностей для корпорации, торгующей мороженым.

1.4. Определение типов сущностей

И зависимые, и независимые сущности можно разделить на несколько типов:

- Стержневые сущности - их иногда называют основными или первичными сущностями. Они представляют важные объекты, информацию о которых следует хранить.
- Коды/ссылки/классификаторы - эти сущности содержат строки, определяющие набор значений или область определения для атрибута.
- Ассоциативные сущности - эти сущности используются для разрешения отношений **многие-ко-многим**.
- Характеристические сущности - эти сущности бывают двух типов: исключающие и включающие.

1.4.1. Стержневые сущности

Стержневые сущности представляют наиболее важные корпоративные информационные объекты. Их иногда называют первичными, главными или основными сущностями. Так как эти сущности чрезвычайно важны, то, скорее всего, они используются во многих подразделениях корпорации. Потратьте время на поиск сходных сущностей, поскольку для стержневых сущностей велика вероятность наличия возможности их повторного использования. В рамках корпорации стержневые сущности должны моделироваться единообразно. Хорошие разработчики моделей рассматривают такой подход как исключительно полезный.

Стержневые сущности могут быть как независимыми, так и зависимыми. На рисунке 2.1 представлены примеры стержневых сущностей для корпорации, торгующей мороженым. Сущность МОРОЖЕНОЕ представляет базовый продукт корпорации. Сущность МАГАЗИН является примером канала сбыта или посредника при продаже товара.

Предположим, что дела в корпорации идут хорошо и принимается решение об открытии дополнительного МАГАЗИНА. Для добавления новых экземпляров сущности МАГАЗИН нет необходимости менять модель. То же самое касается и сущности МОРОЖЕНОЕ.

Понимание необходимости моделировать стержневые сущности в виде масштабируемых и расширяемых контейнеров требует от разработчика модели взгляда на сущности как на абстрактные концепции и моделирования информации независимо от текущего способа ее использования. В этом примере модель сущности МОРОЖЕНОЕ полностью вне контекста сущности МАГАЗИН и наоборот. Так что если в корпорации решат продавать МОРОЖЕНОЕ через новый канал сбыта, такой как Интернет или доставка на дом, новый канал сбыта может быть добавлен без изменений в других сущностях.

1.4.2. Кодовые сущности

Кодовые сущности всегда являются независимыми. Их часто называют ссылками, классификаторами или сущностями типов, в зависимости от используемой методологии. Уникальные экземпляры, представляемые кодовыми сущностями, определяют **область определения** для значений атрибутов, принадлежащих другим сущностям. Отношения между кодовыми сущностями и другими сущностями будут рассмотрены далее. У вас может возникнуть искушение использовать единственный атрибут в кодовой таблице. Гораздо лучше включать, по меньшей мере, три атрибута в кодовую сущность: идентификатор, имя (иногда его называют кратким именем) и определение.

На рисунке 2.2 ВЕРХУШКА - независимая сущность (обратите внимание на прямые углы). ВЕРХУШКА является к тому же кодовой сущностью или классификатором. Экземпляры (строки) сущности ВЕРХУШКА определяют список доступных верхушек.

Кодовые сущности обычно содержат ограниченное количество атрибутов. Существуют реализации, где эти сущности имели только один атрибут. Предпочтительно моделировать кодовые сущности с использованием искусственного идентификатора. **Искусственный идентификатор** вместе с именем и определением позволяют добавлять новые виды ВЕРХУШЕК в качестве экземпляров (строк) в сущность. Обратите внимание на три атрибута сущности ВЕРХУШКА.

Специалисты часто ссылаются на кодовые сущности, как на корпоративные бизнес-объекты. Термин корпоративный бизнес-объект указывает, что сущность определена и совместно используется на корпоративном уровне, а не на уровне единичного приложения, системы или подразделения организации. Эти сущности часто совместно используются многими базами данных для обеспечения целостного подхода к формированию сводных отчетов и при проведении анализа тенденций.

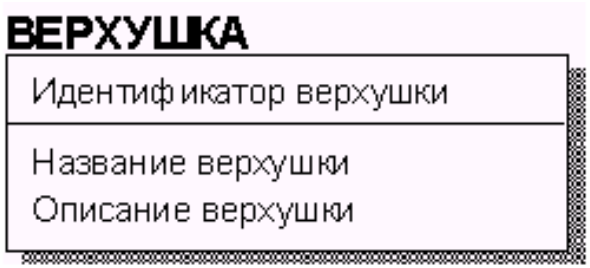

Рис. 2.2. Кодовые сущности позволяют корпорации определять набор значений для централизованного использования в рамках корпорации. Экземпляры кодовой сущности определяют область определения значений для использования в других частях модели.

1.4.3. Ассоциативные сущности

Ассоциативными являются сущности, которые содержат первичные ключи двух или более других сущностей. Ассоциативные сущности всегда зависимы. Они используются для разрешения отношений многие-ко-многим других сущностей. Отношения многие-ко-многим возникают в том случае, когда множество экземпляров одной сущности связаны с множеством экземпляров другой. Ассоциативные сущности позволяют нам моделировать пересечение экземпляров двух сущностей, обеспечивая уникальность каждого экземпляра ассоциации.

> **Примечание**
>
> **Отношения многие-ко-многим не могут быть реализованы в базе данных на физическом уровне. ERwin автоматически создает ассоциативные сущности для разрешения отношений многие-ко-многим при переходе от логической модели к физической.**

На рисунке 2.1 ассоциативная сущность используется для разрешения отношения **многие-ко-многим** между сущностями МАГАЗИН и МОРОЖЕНОЕ. Введение ассоциативной сущности дает возможность использовать одно и то же МОРОЖЕНОЕ для продажи в нескольких экземплярах МАГАЗИНА, без необходимости продажи в каждом из МАГАЗИНОВ одинаковых сортов МОРОЖЕНОГО. Ассоциативная сущность МАГАЗИН МОРОЖЕНОГО учитывает тот факт, что экземпляр МАГАЗИНА продает множество экземпляров МОРОЖЕНОГО, и экземпляр МОРОЖЕНОГО может продаваться многими экземплярами МАГАЗИНА.

1.4.4. Характеристические сущности

Характеристические сущности всегда являются зависимыми. Вы должны использовать характеристические сущности там, где для экземпляров сущностей имеет смысл хранить различные наборы атрибутов. Финклештейн называет характеристические сущности вторичными сущностями. Характеристические сущности всегда имеют одну или более "равноправных" сущностей. Равноправные характеристические сущности связаны с родительской сущностью особым типом отношений, которые могут быть исключающими или включающими.

> **Примечание**
>
> **Равноправные характеристические сущности, которые находятся в исключающем отношении к родительской сущности, указывают на то, что только одна из равноправных сущностей содержит экземпляр для любого из экземпляров родительской сущности. Исключающая характеристическая сущность представляет отношение "является" (is a).**

На рисунке 2.3 представлена сущность КОНТЕЙНЕР и характеристические сущности РОЖОК и СТАКАНЧИК. Магазин мороженого, судя по всему, торгует не на развес, а отдельными порциями. Обратите внимание, что экземпляр КОНТЕЙНЕРА должен быть РОЖКОМ или СТАКАНЧИКОМ. КОНТЕЙНЕР не может быть одновременно и РОЖКОМ и СТАКАНЧИКОМ. Это исключающие характеристические сущности.

Сущность ПЕРСОНА на рисунке 2.3 имеет две характеристические сущности СОТРУДНИК и КЛИЕНТ. Заметьте, что исключающие характеристические сущности не позволят одному экземпляру ПЕРСОНЫ содержать факты, общие для СОТРУДНИКА и КЛИЕНТА. Естественно, это противоречит реальной практике. СОТРУДНИК определенно может быть КЛИЕНТОМ. ПОСТАВЩИК тоже может выступать в качестве КЛИЕНТА. Это пример включающих характеристических сущностей.

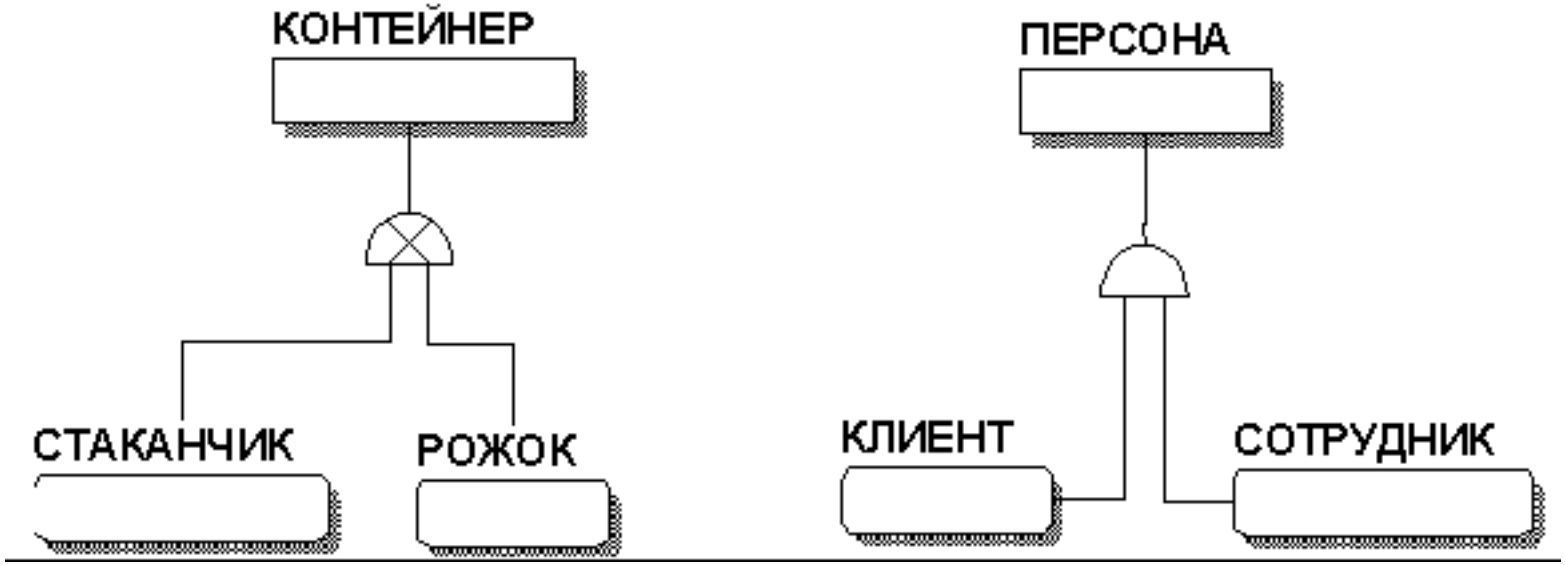

Рис. 2.3. Два примера характеристических сущностей КОНТЕЙНЕР и ПЕРСОНА. Оба примера используют нотацию ERwin IE для представления исключающих и включающих характеристических сущностей. Отсутствие (X) в символе характеристической сущности указывает на включающее отношение.

1.5. Структурная сущность

Иногда экземпляры одной и той же сущности связаны. В своей книге 1992-го года **"Strategic Systems Development"** К. Финклештейн предложил использовать структурные сущности для представления отношений между экземплярами одной и той же сущности. Связи между экземплярами одной и той же сущности называются рекурсивными отношениями. Рекурсивные отношения будут рассмотрены в статье "Понятие отношения". Рекурсивные отношения - это логическая концепция, а концепции не легко воспринимаются пользователями.

На рисунке 2.4 показана дополнительная структурная сущность, описывающая отношение между экземплярами сущности СОТРУДНИК. Диаграмма показывает, что характеристическая сущность СОТРУДНИК сущности ПЕРСОНА имеет две характеристические сущности ИСПОЛНИТЕЛЬ и УПРАВЛЕНЕЦ. Сущность СТРУКТУРА СОТРУДНИКОВ представляет отношение между экземплярами сущности СОТРУДНИК.

Рис. 2.4. Структурная сущность - иллюстрация подхода К. Финклештейна к разрешению рекурсивных отношений.

1.6. Определение первичного ключа

Для идентификации конкретного экземпляра сущности вам необходимо определить первичный ключ. **Первичным ключом** служит атрибут или набор атрибутов, уникально идентифицирующих единственный экземпляр сущности. Другими словами, первичный ключ может быть как одним атрибутом, так и состоять из нескольких. Первичный ключ, состоящий более чем из одного атрибута, называется составным или компонентным ключом. Далее мы будем использовать термин **составной ключ.**

Первичный ключ должен быть статическим (static) и не-разрушаемым (non-volatile). Под статичностью и не-разрушаемостью подразумевается, что первичный ключ не должен подвергаться изменениям. Изменения первичного ключа трудно сопровождать, что часто приводит к весьма дорогостоящим переделкам, поэтому лучшим считается вариант, когда первичный ключ абсолютно не зависит от экземпляров сущности.

> **Примечание**
>
> **Далее будут добавлены первичные ключи в базу данных торговли мороженым и рассмотрены кандидаты в ключи. Концепции ключей только введены в данном разделе, а обсуждаться более подробно будут в статье "Понятие атрибута".**

Для нахождения первичного ключа требуется проанализировать данные, определяющие сущность. Как правило, первичные ключи для стержневых сущностей определяются во время рабочих сессий и обсуждений. Эксперты предметной области и пользователи - хорошие источники информации для выбора потенциальных первичных ключей. Примеры данных тоже обеспечивают ценный вклад при выборе первичного ключа.

Начинайте процесс выявления первичных ключей с определения всех потенциально ключевых атрибутов, называемых кандидатами в ключи. Кандидатом в ключи может быть и один атрибут, и комбинация нескольких атрибутов. Если кандидатов в ключи не существует, или кандидатом является составной ключ, который слишком велик и громоздок, рассмотрите возможность использования искусственного уникального идентификатора. Ключи, заимствованные из родительской сущности, называются внешними ключами. Внешние ключи будут рассматриваться в одной из последующих публикаций на эту тему. Ниже приведено описание различных типов ключей:

- Кандидат в ключи. Кандидатом в ключи является атрибут или набор атрибутов, идентифицирующих единичный экземпляр сущности. Иногда единичный экземпляр сущности идентифицируется несколькими атрибутами или их комбинацией.
- Составной ключ. Ключ, который состоит более чем из одного атрибута, называется составным, сложным или

компонентным. Для составных ключей каждая составляющая ключа должна иметь значение для каждого экземпляра. Ни одна часть ключа не должна быть неопределенной (NULL). Все части ключа являются обязательными и не могут быть опущены.

- Искусственный первичный ключ. Иногда ни единичный атрибут, ни комбинация атрибутов, не определяют экземпляр. В этих случаях вы используете искусственный уникальный идентификатор. Искусственные первичные ключи часто просто нумеруют каждый экземпляр или код.
- Внешние ключи. Когда первичный ключ одной сущности мигрирует в другую таблицу, он называется внешним ключом. Внешние ключи "связывают" сущности, представляя отношения между ними. Внешние ключи будут обсуждаться более подробно в дальнейших статьях по этой тематике.

Приведение модели к третьей нормальной форме включает проверку на отсутствие функциональных зависимостей и выявление первичных или составных ключей. Функциональные зависимости играют важную роль при выявлении первичных ключей и кандидатов в ключи.

1.7. Именование сущностей

Имя, присваиваемое сущности, должно характеризовать экземпляры сущности. Имя должно быть понятным и общепринятым. При выборе имени руководствуйтесь корпоративной точкой зрения и старайтесь использовать имена, отражающие способ использования данных в рамках корпорации, а не в отдельном подразделении. Используйте имена, осмысленные для сообщества пользователей и экспертов предметной области.

1.7.1. Соглашения об именовании сущностей

Соглашения об именовании могут показаться несущественными, если вы работаете в маленькой организации, с небольшим количеством пользователей. Однако, в большой организации с несколькими командами разработчиков и большим количеством пользователей, соглашения об именовании существенно помогают при взаимодействии и совместном использовании данных. В идеале, вы централизованно должны разработать и сопровождать соглашения об именовании, и затем документально оформить их, опубликовав для всей корпорации.

Ниже приведены некоторые положения для формирования начального набора соглашений об именовании, на случай, если в вашей организации пока такой набор не разработан:

- Имя сущности должно быть достаточно описательным. Используйте имена из одного слова, только когда они являются названием широко распространенных концепций. Подумайте об использовании словосочетаний на основе существительных.
- Имя сущности должно быть существительным или словосочетанием на основе существительного в единственном числе. Используйте ПЕРСОНА вместо ПЕРСОНЫ или ЛЮДИ, и КОНТЕЙНЕР - вместо КОНТЕЙНЕРЫ.
- Имя сущности должно быть уникальным. Использование одинаковых имен для сущностей, содержащих различные данные, или разных имен для сущностей, содержащих одинаковые данные, будет без необходимости вводить в заблуждение разработчиков и конечных пользователей.
- Имя сущности должно указывать на данные, которые будут храниться в каждом из экземпляров.
- Имя сущности не должно содержать специальных символов (таких как !, @, #, $, %, &, * и тому подобных) или указывать на принадлежность (МОРОЖЕНОЕ ПЕРСОНЫ).
- Имя сущности не должно содержать акронимов или аббревиатур, если только они не являются частью принятых соглашений об именовании.

Разработчикам моделей рекомендуется использовать хорошие соглашения об именовании, если таковые существуют, или разработать их, следуя приведенным положениям, если таких соглашений нет.

1.7.2. Примеры хороших имен сущностей

Всегда лучше использовать единообразные имена в рамках корпорации. В таблице 2.1 приведены примеры хороших и плохих имен для сущностей.

Совет **Обратите внимание, что для имен сущностей используются заглавные буквы, что соответствует рекомендуемому соглашению об именовании сущностей.**

ТАБЛИЦА 2.1 Примеры имен сущностей с объяснениями

Хорошее имя	**Неудачное имя**	**Пояснение**
МАТЕМАТИЧЕСКАЯ ФОРМУЛА	ФОРМУЛА	ФОРМУЛА - слишком расплывчато, добавление прилагательного МАТЕМАТИЧЕСКАЯ значительно проясняет смысл.
КНИГА	КНИГИ	КНИГА - существительное в единственном числе.
СОФА	СОФА КУШЕТКА	СОФА и КУШЕТКА имеют одинаковый смысл. Выберите что-то одно.
МОРОЖЕНОЕ	КАКОЕ-ТО МОРОЖЕНОЕ	Местоимение КАКОЕ-ТО не привносит дополнительного значения или смысла к термину. Избегайте излишних дополнений.
ФОТОСНИМОК	ИЗОБРАЖЕНИЕ	ФОТОСНИМОК - достаточно определенно. ИЗОБРАЖЕНИЕ - несколько расплывчато.
ОЖИДАЕМОЕ ВРЕМЯ ПРИБЫТИЯ	ОВП	Аббревиатура ОВП может оказаться непонятной для пользователей.
КОМПАНИЯ	КОМПАНИЯ XYZ	XYZ - конкретный экземпляр компании и должен быть строкой в сущности КОМПАНИЯ.

1.8. Описание сущностей

Даже хороших имен, указывающих пользователю, какую информацию стоит ожидать от сущности, обычно недостаточно. Каждая сущность нуждается в ясном, точном и полном описании или определении, чтобы быть однозначно интерпретируемой в рамках корпорации. Описание сущности должно объяснять смысл сущности и ее значение для корпорации.

Хотя описание, определение и назначение часто используются в качестве синонимов, термин описание предпочтительнее, поскольку он побуждает нас описывать сущности в терминах, понятных для пользователя.

1.8.1. Правила формирования хороших описаний

Описание сущности должно объяснять ее смысл, а не то, как будет использоваться информация этой сущности. Вы должны собирать описания сущностей во время идентификации сущностей. Будьте осторожны при включении информации об использовании: подобная информация должна использоваться только в качестве примера или для пояснения. Способ использования информации изменяется более часто, чем информация сама по себе, поэтому информация об использовании непостоянна.

Описание сущности должно быть ясным, точным, полным и непротиворечивым. Оно должно быть сформулировано без привлечения технических терминов, понятно любому, кто хотя бы чуть-чуть знаком с описываемой концепцией. Убедитесь, что описание сформулировано в терминах бизнеса, и включает пояснение значимости сущности.

Далее дан пример хороших описаний. Таблица 2.2 не претендует на полноту, но служит для демонстрации хороших описаний и причин, по которым неудачные описания не отвечают основным положениям.

1.8.2. Примеры хороших описаний

ТАБЛИЦА 2.2. Описания сущностей с пояснениями

Хорошее описание	Неудачное описание	Пояснение
ПЕРСОНА содержит информацию о физических лицах, которые вступают во взаимодействие с корпорацией. Информация о ПЕРСОНЕ помогает корпорации при планировании, разработке продуктов и рекламной деятельности.	Клиент или сотрудник.	Хорошее описание включает определение сущности и ее значение для корпорации.
	Включает имя, дату рождения и т.п. для персоны.	Простое перечисление атрибутов сущности не несет дополнительной информации о том, что собой представляет сущность и почему она важна для корпорации.
	Информация о клиентах и сотрудниках.	Клиент и сотрудник являются примерами ролей, в которых может выступать ПЕРСОНА. Использование одних только примеров не объясняет, что сущность собой представляет и почему она важна для корпорации.
	Сущность содержит символы и числовые данные, извлеченные из POS (Point Of Sale - торговый терминал), хранящиеся с использованием стандартного сжатия и упакованных десятичных чисел.	Данный искусственный пример призван проиллюстрировать, что технические описания и аббревиатуры с трудом понимаются бизнес-пользователями.

1.9. Распространенные ошибки при моделировании сущностей и выборе ключей

Этот раздел, посвященный распространенным ошибкам при моделировании, не претендует на полноту. Его цель - указать на наиболее распространенные ошибки, которые возникают у разработчиков моделей.

> **Совет**
>
> **Иногда при моделировании разработчик модели делает некий выбор, руководствуясь совершенно правильными принципами. Очень важно понимать всю причинно-следственную цепочку принимаемых решений так, чтобы вы ясно понимали полученное заключение.**

1.9.1. Моделирование ролей

Что подразумевается под моделированием ролей? Во время рабочих сессий пользователи могут сказать вам, что им необходимо хранить информацию о сотрудниках. Возникает искушение создать сущность СОТРУДНИК. Более тщательный анализ информации, представляющей интерес для корпорации, например, такой как имя, адрес и номер социального страхования показывает, что эти значения не зависят от сущности СОТРУДНИК. Для конкретного СОТРУДНИКА значение атрибута ИМЯ не зависит от сущности СОТРУДНИК. Это легко понять, если задуматься о том, что ваше имя остается вашим именем вне зависимости от того, являетесь ли вы СОТРУДНИКОМ или нет.

1.9.2. Перегрузка сущностей

Перегруженными являются сущности, содержащие информацию более чем об одном концептуальном объекте. Если некоторые атрибуты сущности описывают одну и ту же концепцию, такие сущности следует проверить. Перегруженные сущности имеют значения не для каждого из атрибутов.

Иногда эксперты из разных предметных областей в корпорации используют имя сущности, которое звучит и пишется одинаково, но имеет разный смысл для разных экспертов. Единственный способ убедится, что одинаковые имена описывают одинаковые объекты, это проверка описаний. Убедитесь, что сущность содержит данные, описывающие единственную концепцию.

Например, сущность ОБОРУДОВАНИЕ может иметь совершенно разное значение для подразделений информационных технологий и для отдела средств массовой информации и коммуникаций.

1.9.3. Избыточные сущности

Избыточными являются сущности, имеющие различные имена, но содержащие информацию о сходных концепциях. Английский язык включает много слов для представления одних и тех же вещей. Один из способов обнаружить такие сущности - это поиск сущностей, содержащих сходные атрибуты. Сравните описания каждой из таких сущностей, чтобы определить, не представляют ли они сходные концепции. Избыточные сущности часто появляются в результате тенденции к моделированию ролей в качестве сущностей.

Например, сущности УПРАВЛЕНЕЦ и СОТРУДНИК могут содержать сходную информацию, поскольку обе являются ролями, которые может играть экземпляр сущности ПЕРСОНА.

1.9.4. Выбор неправильного первичного ключа

Выбор неправильного первичного ключа означает, что вы выбрали первичный ключ, не выдерживающий тестирования. Распространенными ошибками, связанными с первичным ключом, являются:

- Не уникальность: первичный ключ не является уникальным для каждого из экземпляров. Например, разработчик модели может считать, что номер социального страхования является уникальным для каждой ПЕРСОНЫ. Однако номер социального страхования может повторно использоваться в том случае, если первоначальный его владелец скончался.
- Требуемое значение/неопределенность: первичный ключ не имеет значения для некоторых из экземпляров. Например, не каждый экземпляр сущности ПЕРСОНА будет иметь номер социального страхования. Иностранцы и маленькие дети - вот две категории людей, у которых он будет отсутствовать.

1.9.5. Использование неудачных имен сущностей

Непонятные, неоднозначные или неточные имена затрудняют для новых пользователей и команд разработчиков повторное использование или расширение существующей модели.

Не используйте аббревиатуры или акронимы в качестве части имени. Аббревиатуры и акронимы открыты для неправильной интерпретации и даже могут иметь разное значение в разных предметных областях.

Предостережение

Не используйте имена собственные, указывающие на конкретный экземпляр, такие как, например, Компания Интерфейс. Это неудачный выбор имени сущности, который в дальнейшем приведет к серьезным проблемам.

Не включайте месторасположение в качестве части имени. Как правило, вам неизбежно потребуется и другое месторасположение. Имя с указанием расположения является признаком того, что вы моделируете конкретный экземпляр вместо класса сущностей.

1.9.6. Использование неудачных описаний сущностей

Не используйте описаний, заимствованных только из словаря. Описания из словаря не будут включать значимую для бизнеса информацию.

Не пытайтесь перефразировать имя сущности. Не используйте имя сущности в ее описании.

Неясные, расплывчатые или, что еще хуже, неполные описания затрудняют повторное использование и расширение существующей модели. Пользователь не сможет проверить, содержит ли сущность всю необходимую информацию.

При этом значительно повышается риск возникновения перегруженных сущностей и использования их для хранения информации о разных объектах.

Концепции, которые кажутся очевидными для всех участников рабочих сессий, могут перестать быть столь очевидными с течением времени, когда перед новой командой разработчиков будет поставлена задача расширения существующей модели.

1.10. Заключение

Сущности представляют собой объекты, информацию о которых следует накапливать и сопровождать. Они являются "контейнерами" для организации и группировки бизнес-фактов. Наиболее важные сущности обычно выявляются и фиксируются в документах во время рабочих сессий или интервью, а также в результате процесса нормализации.

Сущности делятся на две основные группы: зависимые и независимые. Зависимым сущностям для уникальной идентификации экземпляра требуется информация из других сущностей, независимым - нет. В рамках двух основных групп сущностей выделяются более специализированные типы, с особенностями для поддержки конкретных видов отношений между основными и подчиненными сущностями.

Каждая сущность должна включать один или несколько наборов атрибутов, являющихся кандидатами в ключи. Кандидаты в ключи уникально идентифицируют конкретные экземпляры сущности. Кандидаты в ключи могут состоять из одного атрибута или из группы атрибутов. Если кандидатов в ключи не существует, или их трудно сопровождать, вам может потребоваться создать искусственный первичный ключ. Анализ и исследования играют важную роль в определении первичных ключей, которые будут сохранять уникальность и надежность с течением времени.

Для сущностей необходимы хорошие имена и описания. Стандарты и соглашения об именовании обеспечивают целостный подход к разработке имен и описаний. Характеристики сущности определяются содержащимися в ней атрибутами. Атрибуты сущности представляют факты, касающиеся сущности, которые корпорация заинтересована накапливать и сопровождать.

В следующей статье данной серии будет описан процесс выявления атрибутов и их характеристик, определения ключевых и не ключевых атрибутов, областей определения и необязательных данных, а также сформулированы соглашения для формирования хороших имен и описаний атрибутов.

2. Понятие атрибута

В этой части вы узнаете как:

- Выявлять атрибуты
- Осуществлять нормализацию в процессе анализа атрибутов
- Именовать и описывать атрибуты, а также узнаете о соглашениях, об именовании атрибутов
- Определять типы и характеристики атрибутов, такие как область определения и логические типы данных, и проверять ключи с точки зрения атрибутов
- Избежать распространенных ошибок при работе с атрибутами

На ER-диаграммах сущности и отношения служат для группировки и объединения атрибутов. Именно атрибуты составляют суть модели. Так что давайте приступим к изучению атрибутов - фактов, составляющих информацию логической модели.

2.1. Что такое атрибут?

Атрибут является логическим представлением фактов, данные о которых корпорация заинтересована хранить. Вспомните, что в ERwin сущности служат для визуального представления логической группировки атрибутов. С другой стороны, атрибуты представляют факты, накапливаемые о сущностях в логической модели. Атрибуты представляют собой факты, которые служат для идентификации, характеристики отнесения к категории, числового представления или другого вида описания состояния экземпляра сущности.

Атрибут должен представлять единственную концепцию. Атрибуты формируют логические группы, описывающие каждый экземпляр сущности. Конкретным экземпляром атрибута является **значение** . Например, атрибут с названием Имя определяет область определения для фактов о сущности с названием ПЕРСОНА. Габриэль, Р.Дж., Уилл и Ванесса - примеры конкретных значений Имени для конкретных экземпляров ПЕРСОНЫ. Конкретные значения для каждого из атрибутов сущности представляют единственный экземпляр.

Корректная модель атрибута обладает следующими признаками:

- Значение атрибута представляет интерес для корпорации.
- В логической модели присутствует единственный экземпляр атрибута.
- Атрибут имеет логический тип данных и область определения.
- Значение атрибута определяется как требуемое или необязательное.

• Атрибут имеет имя и описание.
• Для каждого экземпляра сущности может использоваться только одно значение.

> **ПРИМЕЧАНИЕ** На Рисунке 3.1 представлен пример не очень хорошей модели; это прямолинейное отражение требований к информации. В следующих разделах будет предпринята попытка улучшить эту модель для демонстрации процесса размещения атрибутов в соответствующих сущностях.

Корпорация Торговли мороженым Бетти Уилсон хочет заказывать больше наиболее популярных вкусовых добавок и меньше - наименее популярных. Корпорация Бетти делает специальные предложения по продаже мороженого, и заинтересована знать, мороженое с каким вкусом покупатели выбирают для бананового десерта и сливочной помадки во время специальных предложений. Для соответствия бизнес-требованиям необходимо собирать данные о вкусовых добавках к мороженому для бананового десерта и сливочной помадки и дату.

На рисунке 3.1 две сущности БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА. Каждая сущность содержит атрибуты, представляющие компоненты каждого из блюд. Обратите внимание, что для сущности БАНАНОВЫЙ ДЕСЕРТ можно выбрать три вкусовых добавки, три верхушки: банан, взбитые сливки и вишни. Для экземпляра СЛИВОЧНОЙ ПОМАДКИ можно выбрать две вкусовых добавки и банан, взбитые сливки и вишни.

Рис. 3.1. Сущности и атрибуты, представляющие (не очень удачно) две основных концепции: СЛИВОЧНАЯ ПОМАДКА и БАНАНОВЫЙ ДЕСЕРТ

2.2. Выявление атрибутов

С чего начинать процесс выявления атрибутов? Большинство атрибутов выявляются в ходе рабочих сессий и интервью во время определения сущностей. Анализ требований к информации, полученных от экспертов в предметной области и конечных пользователей - наилучший источник информации для идентификации атрибутов.

Корпоративная модель тоже является отличной основой для выделения атрибутов. Сравните сущности и атрибуты корпоративной модели с сущностями и атрибутами новой логической модели. В корпоративной модели присутствуют атрибуты, которые были ранее определены для каждой из сущностей, в особенности для стержневых сущностей. Если атрибут не присутствует в корпоративной модели, дополнительный анализ позволит определить, нужно ли его добавить или он принадлежит другой сущности.

2.2.1. Упорядочивание атрибутов в соответствии с требованиями к информации

Атрибуты логической модели должны строго соответствовать требованиям к информации. Каждый из присутствующих в модели атрибутов должен служить удовлетворению одного или нескольких требований к информации. Модель должна содержать только те атрибуты, которые необходимы для представления фактов, интересующих корпорацию в рамках рассматриваемой предметной области.

> **СОВЕТ** Атрибуты, выходящие за рамки предметной области и не связанные напрямую с одним или более требованиями к информации, должны быть устранены. Атрибуты, не имеющие большого значения для корпорации, тоже должны устраняться в целях сокращения затрат на программирование и сопровождение.

Каждый факт, интересный с точки зрения корпорации, должен быть точно и полно представлен в логической модели. Требования к информации служат мерой необходимости выделения атрибута. Представляется полезным документирование взаимосвязей между атрибутами и требованиями к информации.

2.2.2. Анализ атрибутов

Вам следует проанализировать каждый из атрибутов для определения его взаимосвязей со всеми остальными атрибутами модели. Корректно выполненный анализ гарантирует, что каждый из атрибутов присутствует в модели в единственном экземпляре и размещен в сущности в соответствии с третьей нормальной формой.

Особенно важно проанализировать каждый первичный ключ и каждую часть составного первичного ключа для проверки того, что их значения существуют для каждого экземпляра сущности. Вы должны также убедиться, что

первичный ключ идентифицирует один и только один экземпляр сущности.

С помощью анализа также можно установить, заинтересована ли корпорация накапливать и сопровождать какую-либо информацию собственно об атрибуте. Если атрибут так важен, что требуются дополнительные атрибуты для хранения данных о нем, то следует задуматься о возможности создания новой сущности.

Вы должны проанализировать каждый из атрибутов логической модели, чтобы убедиться, что каждый из атрибутов присутствует в модели в единственном экземпляре и только одно значение атрибута существует для каждого экземпляра сущности. Вы должны поместить атрибут в соответствующей сущности, используя правила нормализации, и определить его характеристики.

2.2.3. Остаться должен только один

Атрибут должен присутствовать в логической модели в единственном экземпляре. "Один факт в одном месте" (Дейт, 1986). Для гарантии того, что каждый факт представлен единственным атрибутом, проверьте атрибуты со сходными именами или описаниями. Кроме того, вы должны определить, являются ли атрибуты реальными экземплярами или конкретными значениями, которые ошибочно представлены в модели разными атрибутами.

Атрибуты со сходными именами и описаниями могут в действительности представлять одну и ту же концепцию и должны быть представлены одним атрибутом. В естественном языке одно и то же слово может представлять несколько концепций. Но что еще хуже, в английском языке для представления одной и той же концепции может существовать несколько разных слов.

Атрибуты, имеющие в составе своего имени слова "индикатор" или "флаг", скорее всего, представляют конкретное значение из области определения атрибута. Конкретное значение является экземпляром атрибута. Использование в модели экземпляров атрибутов - распространенная ошибка. Например, "Индикатор черных волос" имеет значение "да" если присутствуют черные волосы, и значение "нет" если черные волосы отсутствуют. Более предпочтительным будет использование в модели атрибута "Цвет волос", который может иметь конкретное значение "Черный".

Атрибут должен представлять только одну концепцию бизнеса. Он не должен иметь несколько значений для одного экземпляра сущности. На Рисунке 3.1 показаны две сущности, БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА. Обе сущности содержат многозначный атрибут с именем "Дата начала или окончания специального предложения". Имя атрибута показывает, что его значение может представлять дату начала специального предложения или дату окончания специального предложения, и у нас нет возможности их различить! Этот атрибут должен быть разделен на два, каждый из которых будет представлять единственный факт.

ПРИМЕЧАНИЕ Хотя разбиение одного атрибута на два для различения фактов позволяет разрешить проблему с многозначностью атрибута, остается другая проблема: значения атрибутов **Дата начала специального предложения** и **Дата окончания специального предложения** не зависят от идентификаторов сущностей БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА. Эта проблема связана с нормализацией и будет рассмотрена в следующем разделе.

Если мы разрешим атрибуту иметь несколько значений, это может привести к появлению тесно связанных "скрытых" атрибутов. Предыдущий пример достаточно очевиден. Не все многозначные атрибуты могут быть так легко преобразованы. Для вас может оказаться неожиданностью, что в атрибуте, содержащем фрагмент текста, такой как комментарий или примечание, среди текста спрятано множество важных значений атрибута.

2.3. Нормализация: помещение атрибута в соответствующую сущность

Атрибуты определяют количество сущностей, которые будут присутствовать в логической модели, приведенной к третьей нормальной форме. Процесс нормализации заключается в анализе зависимости атрибутов друг от друга и зависимости атрибутов от первичного ключа.

Корректно проведенная нормализация гарантирует, что модель будет масштабируемой и расширяемой за счет помещения атрибутов в соответствующие сущности.

Приведение логической модели к третьей нормальной форме часто приводит к появлению новых сущностей.

> **ПРЕДОСТЕРЕЖЕНИЕ** Осторожно добавляйте новые атрибуты к сущностям нормализованной модели. Новый атрибут должен зависеть от значения ключа, полного ключа, и ни от чего кроме ключа. Рассмотрим случай существования составного первичного ключа: добавление нового атрибута, значение которого зависит от значения части ключа, нарушает требования второй нормальной формы.

Другими преимуществами нормализации являются:

- Устранение или минимизация избыточности. Избыточные данные могут присутствовать в атрибутах, представляющих одну и ту же концепцию, но по-разному поименованных, или в повторяющихся группах. Однократное представление каждого факта в одном месте минимизирует избыточность.
- Устранение или минимизация аномалий при вставке, удалении или обновлении. Ненормализованные структуры данных позволяют одному и тому же факту присутствовать в нескольких местах при неполной или частичной зависимости от первичного ключа. Аномалии при вставке, удалении или обновлении заключаются в противоречивости данных, которая при этих условиях может привести к неожиданностям или неточностям при доступе к данным.
- Устранение или минимизация использования пустых значений для атрибутов. Повторяющиеся группы атрибутов часто содержат пустые значения для многих экземпляров, так как они представляют факт, для которого одни сущности могут иметь несколько значений, а другие - нет. Наличие сущностей, содержащих экземпляры с пустыми значениями, приводит к слабо заполненным (разреженным) структурам данных.

Когда модель приведена к третьей нормальной форме, каждый атрибут принадлежит соответствующей сущности. При приведении модели к третьей нормальной форме часто обнаруживаются новые атрибуты и сущности.

2.3.1. Функциональная зависимость

Функциональная зависимость служит для описания взаимосвязей между атрибутами в модели. Каждый атрибут сущности должен функционально зависеть от первичного ключа сущности (и не зависеть функционально от любого другого атрибута модели). Если это не так, атрибут должен быть перемещен в новую сущность, где это положение будет соблюдаться.

> **ПРИМЕЧАНИЕ** В заданном отношении R атрибут Y функционально зависит от атрибута X. В символьном виде R.X -> R.Y (читается как "R.X функционально определяет R.Y") - в том и только в том случае, если каждое значение X в R ассоциируется строго с одним значением Y в R (в каждый конкретный момент времени). Атрибуты X и Y могут быть составными (Дейт, 1986).

Для определения функциональной зависимости между атрибутами сначала сгруппируйте их в наборы, объединенные общей темой. Тщательно проанализируйте темы с точки зрения их сходства. Проверьте атрибуты в темах, для определения наличия функциональной зависимости атрибутов в рамках темы. Если атрибут, или группа атрибутов, не зависят от первичного ключа сущности, они должны быть перемещены в другую сущность.

Атрибуты, принадлежащие к одной теме, могут оказаться избыточными. Избыточные атрибуты могут быть сгруппированы в единую сущность или могут использовать общую абстракцию более высокого уровня в качестве характеристических сущностей родительской сущности. На рисунке 3.1 присутствует, по меньшей мере, две общих темы: **Вкусовая добавка к мороженому** и **Верхушка**. Эти атрибуты являются хорошими кандидатами на перенос в другие сущности. Рассмотрим их в аспекте функциональной зависимости. Значение атрибута **Вкусовая добавка** к мороженому не зависит от значения первичного ключа - **Ингредиенты бананового десерта**. То же самое касается и ключа **Сливочная помадка**.

Рисунок 3.2 иллюстрирует решение, в котором **Вкусовая добавка к мороженому** и **Верхушка** выделены в сущности, где их значения зависят от первичного ключа. Это решение устраняет некоторые очевидные проблемы, связанные с избыточностью.

2.3.2. Первая нормальная форма

Приведение к первой нормальной форме означает перемещение всех повторяющихся атрибутов в другую сущность. Повторяющиеся атрибуты достаточно легко обнаружить, так как часто они просто пронумерованы как **Верхушка 1** и **Верхушка 2** или **Вкус 1** и **Вкус 2.**

Создайте зависимую сущность, которая будет содержать набор атрибутов для представления повторяющихся атрибутов. Первичный ключ зависимой сущности будет составным первичным ключом, в который войдет первичный ключ родительской сущности и, по меньшей мере, один дополнительный атрибут для гарантии уникальности.

На рисунке 3.2 перенесены повторяющиеся группы **Вкусовая добавка к мороженому** и **Верхушка** в зависимые сущности. Обратите внимание на создание сущности ВКУС.

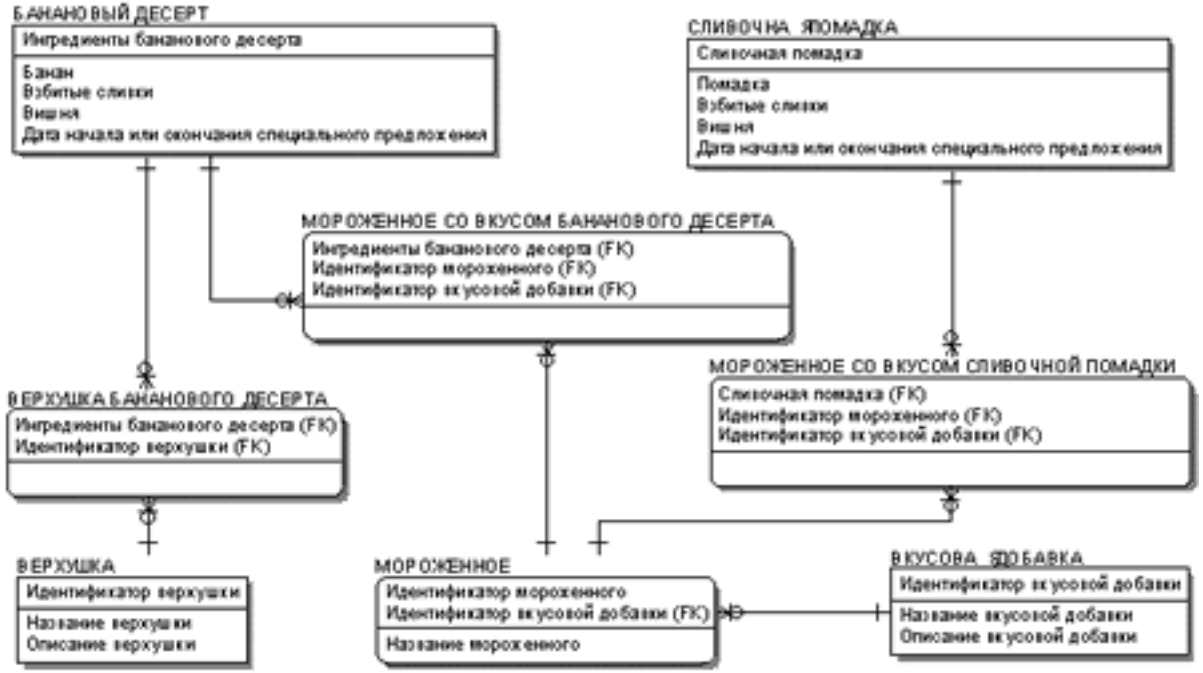

Рис. 3.2. Устранение избыточных атрибутов

2.3.3. Вторая нормальная форма

Приведение ко второй нормальной форме означает удаление избыточных атрибутов. Избыточными атрибутами могут быть:

- Разные атрибуты представляющие одну и ту же концепцию
- Атрибуты различных сущностей, относящиеся к одной и той же теме
- Атрибуты, которые имеют значения не для каждого из экземпляров сущности

> **СОВЕТ** Тщательно проанализируйте сущности со сходными атрибутами. Эти сущности могут оказаться связанными или даже представлять одну и ту же концепцию. Если это так, их нужно объединить.

Атрибуты, представляющие одну и ту же концепцию, должны быть преобразованы к единому атрибуту. Избыточные атрибуты могут иметь значения не для каждого из экземпляров сущности и, таким образом, их существование не будет зависеть от значения первичного ключа. Переместите эти атрибуты в сущность, где они будут иметь значения для каждого из экземпляров.

Создайте сущность с атрибутами для представления избыточных атрибутов. Новая сущность обладает первичным ключом, идентифицирующим единственный экземпляр. Этот первичный ключ станет внешним ключом в исходной сущности. Внешние ключи будут обсуждены позднее.

Рисунок 3.2 демонстрирует решение для некоторых избыточных атрибутов сущностей БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА. Рассмотрим избыточность с точки зрения двух стержневых сущностей. В обеих сущностях присутствуют общие темы: вкусовая добавка к мороженому и верхушка. Это признак того, что стержневые сущности могут быть объединены на более высоком уровне абстракции.

Рисунок 3.3 демонстрирует создание супертипа с именем СМЕСЬ, для которого БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА являются его реализациями. Я добавил классификационный атрибут "Тип смеси" в родительскую сущность СМЕСЬ для идентификации является ли СМЕСЬ экземпляром сущности БАНАНОВЫЙ ДЕСЕРТ или СЛИВОЧНАЯ ПОМАДКА. Экземпляр сущности СМЕСЬ может быть экземпляром сущности БАНАНОВЫЙ ДЕСЕРТ или СЛИВОЧНАЯ ПОМАДКА, но не их обеих одновременно.

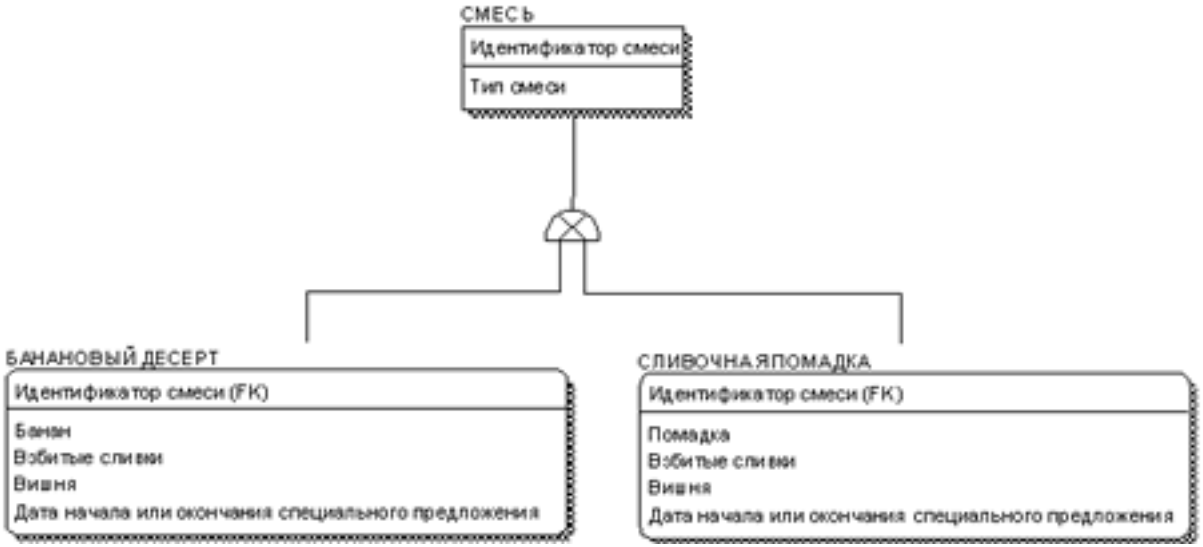

Рис. 3.3. Избыточность стержневых сущностей устранена за счет перемещения общих атрибутов в более общую сущность СМЕСЬ. Обратите внимание, что первичный ключ "Идентификатор смеси" помещен и в сущности БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА.

2.3.4. Третья нормальная форма

Приведение к третьей нормальной форме означает устранение любых атрибутов, которые зависят от значений других атрибутов кроме первичного ключа. Иногда это называют транзитивной зависимостью .

Создайте новую сущность и переместите в нее атрибуты, не зависящие от первичного ключа в исходной сущности. Определите первичный ключ для новой сущности так, чтобы он гарантировал уникальность.

На Рисунке 3.3 атрибуты **Взбитые сливки** и **Вишня** не зависят от первичных ключей сущностей БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА. Фактически вы должны решить, не являются ли атрибуты **Взбитые сливки** и **Вишня** экземплярами сущности ВЕРХУШКА.

> **ПРИМЕЧАНИЕ** Сущность **БАНАНОВЫЙ ДЕСЕРТ** содержит атрибут **Взбитые сливки** и сущность **СЛИВОЧНАЯ ПОМАДКА** тоже содержит атрибут **Взбитые сливки**. Сравнение описаний этих атрибутов показывает, что они описывают одну и ту же концепцию. Взбитые сливки были выбраны как логическое имя для представления общей концепции и перемещены в более общую сущность **СМЕСЬ.**

На Рисунке 3.4 обратите внимание на дополнительный атрибут Дата смеси, который обеспечивает информацию о том, когда был создан экземпляр сущности СМЕСЬ. Я удалил атрибуты Дата начала и Дата окончания из сущностей БАНАНОВЫЙ ДЕСЕРТ и СЛИВОЧНАЯ ПОМАДКА. Новая сущность СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ теперь содержит эти две даты и атрибут Вкусовая добавка к мороженому для указания того, на какой из видов мороженого распространяется предложение.

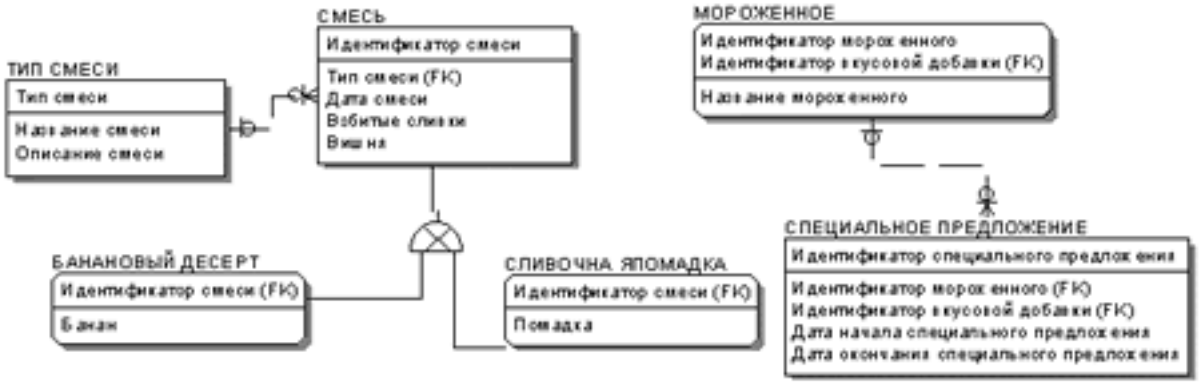

Рис. 3.4. Каждый атрибут зависит от первичного ключа, полного первичного ключа и ни от чего кроме ключа.

2.4. Определение характеристик атрибута

Атрибуты делятся на две группы. Атрибут либо является ключом, либо нет. Рисунок 3.5 показывает ключевые атрибуты для логической модели сущности СМЕСЬ. Заметьте, что, в сущности, атрибуты первичного ключа располагаются над линией внутри сущности, а остальные атрибуты - под линией.

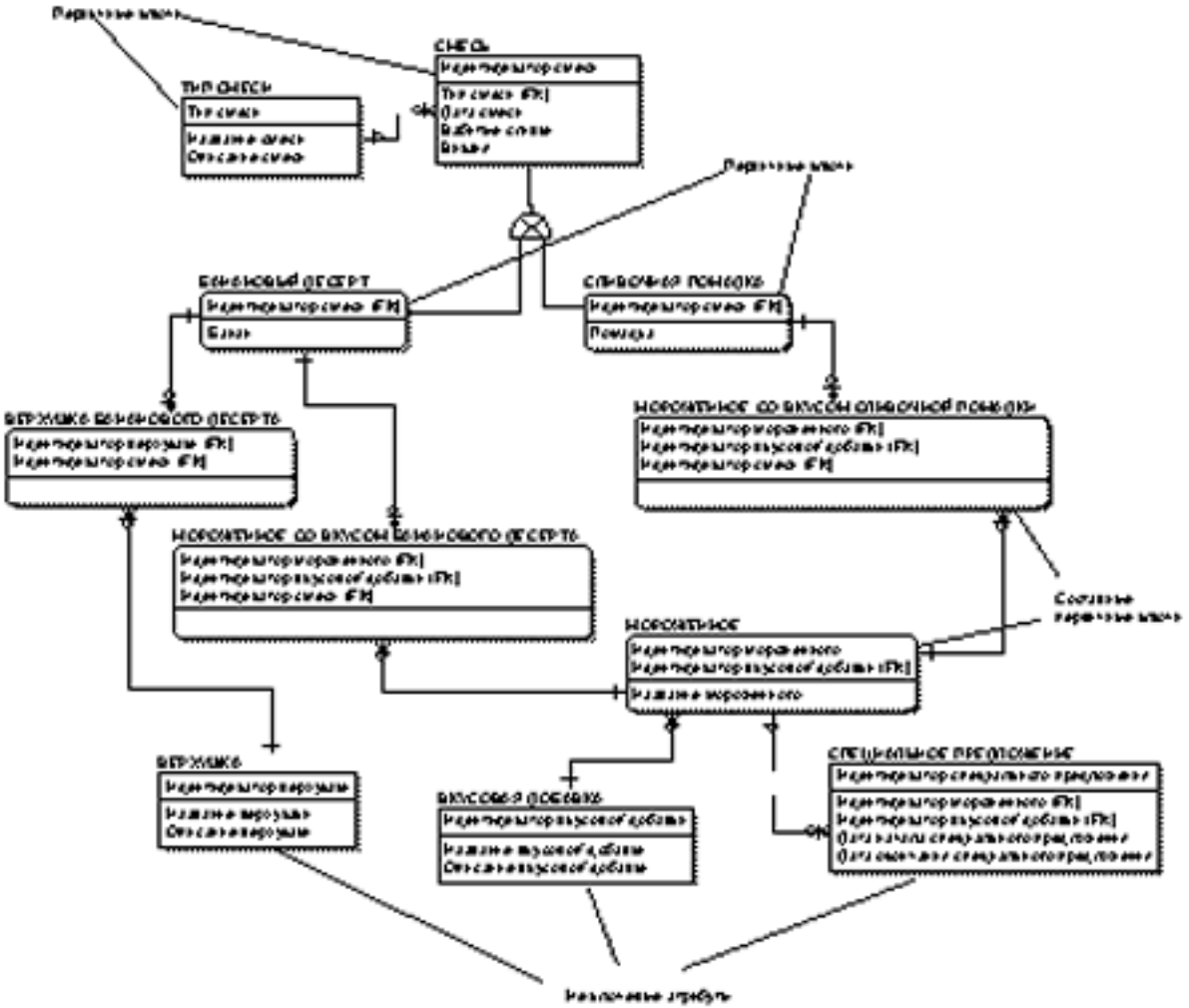

Рис. 3.5. Все атрибуты, не являющиеся частью первичного ключа, располагаются в сущности ниже разделителя. Это могут быть кандидаты в ключи, внешние и альтернативные ключи и простые атрибуты.

2.5. Ключевые атрибуты

Ключевыми являются атрибуты, значения которых определяют значения других атрибутов. Значения ключевых атрибутов не зависят от значений никаких других атрибутов. Ключ может состоять из единственного атрибута или быть составлен из нескольких атрибутов. Эти атрибуты могут быть первичными ключами, составными первичными ключами, кандидатами в ключи, внешними ключами или альтернативными ключами.

2.5.1. Атрибуты первичного ключа

Является ли первичный ключ единственным атрибутом или группой, его значения определяют значения всех других атрибутов.

Хороший первичный ключ будет обладать следующими признаками:

- Значение гарантирует уникальность для каждого из экземпляров
- Значение не имеет скрытого смысла
- Область определения значений будет оставаться постоянной с течением времени
- Значения существуют для каждого из экземпляров сущности

2.5.2. Искусственные первичные ключи

Искусственные первичные ключи - это атрибуты, созданные с единственной целью идентификации конкретных экземпляров сущности. В некоторых случаях не существует атрибута или группы, однозначно идентифицирующих экземпляр сущности. В других случаях, составной первичный ключ слишком громоздок, и его трудно сопровождать. Искусственный первичный ключ иногда называют псевдоключом или ключом, сгенерированным системой . Еще он известен под названием искусственный уникальный идентификатор , которое указывает на его назначение.

Искусственный первичный ключ часто формируется простой последовательной нумерацией каждого из экземпляров сущности. Дополнительным преимуществом таких искусственных ключей является то, что не нужно заботиться о смысле связанных с ними экземпляров сущности, кроме гарантии уникальности. Фактически, искусственные первичные ключи, созданные таким способом, гарантированно обладают особенностями хороших первичных ключей.

Обратите внимание, что большинство первичных ключей на Рисунке 3.5 являются искусственными. По большей части, первичный ключ является просто уникальным номером для каждого из экземпляров.

2.5.3. Кандидаты в ключи

Кандидатом в ключи является атрибут или группа атрибутов, идентифицирующих конкретный экземпляр сущности. Кандидат в ключи представляет механизм определения потенциальных первичных ключей для идентификации конкретных экземпляров сущности.

Кандидат в ключи, который не выбран в качестве первичного ключа, еще называют альтернативным ключом. Альтернативный ключ - это атрибут или группа атрибутов, которые могут быть использованы при индексировании.

2.5.4. Внешние ключи

Внешним ключом является атрибут или группа атрибутов, составляющих первичный ключ другой сущности. Внешний ключ может быть, а может и не быть, ключевым атрибутом в связанной сущности. Обратите внимание на термин связанная сущность. Внешние ключи представляют связи между сущностями, которые более детально будут обсуждаться в следующей статье.

2.5.5. Миграция атрибутов первичного ключа

Внешние ключевые атрибуты являются атрибутами первичного ключа другой сущности, которые мигрировали в данную сущность через связь. Внешние ключи могут быть как идентифицирующими, так и неидентифицирующими. Идентифицирующие внешние ключи становятся частью первичного ключа в той сущности, в которую они мигрировали. Неидентифицирующие внешние ключи становятся не ключевыми атрибутами.

2.6. Неключевые атрибуты

Неключевыми являются атрибуты, значения которых зависят от значений первичного ключа или составного первичного ключа. Эти не ключевые атрибуты должны зависеть от значения ключа, полного ключа, и ни от чего кроме ключа.

В своей книге **Strategic Systems Development** К. Финклештейн определяет несколько типов неключевых атрибутов:

- Селективные атрибуты - атрибуты, используемые для идентификации единственного экземпляра сущности, когда ключ не уникален. Также называются вторичными ключами .
- Групповой атрибут - атрибут, объединяющий группу более детальных атрибутов.
- Атрибуты повторяющейся группы - атрибуты, представляющие несколько включений одного атрибута в рамках сущности.
- Производные атрибуты - атрибуты, чьи значения определяются из значений других атрибутов.
- Атрибуты основных данных - атрибуты, которые не являются селективными, групповыми, атрибутами повторяющейся группы или производными.

Селективные, групповые и атрибуты повторяющейся группы не должны присутствовать в логической модели, приведенной к третьей нормальной форме. Селективные атрибуты должны стать частью первичного ключа, если они нужны для идентификации единственного экземпляра сущности. Групповые атрибуты являются многозначными. По моему мнению, групповые атрибуты лучше всего представлять в модели кодовыми или классификационными сущностями. Как отмечалось выше, повторяющиеся группы должны быть вынесены в подчиненные сущности.

В третьей нормальной форме не ключевые атрибуты должны быть простыми (основными) или производными атрибутами.

2.6.1. Простые атрибуты

Простые атрибуты - это атрибуты, которые в результате декомпозиции были доведены до наивысшей степени детализации и, таким образом, их значения полностью зависят от первичного ключа и определены для каждого из экземпляров сущности. Они не являются критерием выбора и не могут служить для группировки сущностей. Они представляют простые атомарные факты, в которых заинтересована корпорация.

2.6.2. Производные атрибуты

Производными атрибутами являются атрибуты, значения которых выведены или вычислены на основе значений

одного или более других атрибутов. Вопрос допустимости присутствия производных атрибутов в логической модели активно обсуждается. Некоторые эксперты считают, что поскольку значения производных атрибутов зависят от значений исходных атрибутов, производные атрибуты не должны быть представлены в логической модели.

Логическая модель предназначена для полного и точного представления требований к информации. Вы можете принять решение создать производные атрибуты на уровне физической модели в соответствие с требованиями к использованию.

Хотя понятны аргументы в пользу исключения производных атрибутов, тем не менее, логическая модель - наилучшее место для введения имен и описаний всех атрибутов. Поэтому предпочтительнее включать производные атрибуты в логическую модель. Однако, разработчикам моделей стоит отказаться от использования производных атрибутов в качестве первичных ключей или части составных первичных ключей. Также не забудьте включить правило вывода в описание производного атрибута.

2.7. Нахождение области определения атрибута

Область определения атрибута задает список разрешенных значений, которые атрибут может принимать в конкретном экземпляре сущности. Область определения включает, по меньшей мере, область определения универсального типа данных и может включать область определения, заданную пользователем. В завершенной логической модели вы должны найти область определения для каждого из атрибутов.

Можно сказать, что область определения атрибута должна содержать, как минимум, два значения. Атрибут, для которого всегда разрешено только одно значение, вероятно, некорректно отображен в модели. На Рисунке 3.5 присутствует два таких атрибута - Банан и Помадка.

В сущности БАНАНОВЫЙ ДЕСЕРТ присутствует атрибут Банан. Бизнес-правило утверждает, что каждый экземпляр сущности БАНАНОВЫЙ ДЕСЕРТ содержит банан. Поэтому у атрибута Банан может быть только одно значение и, вероятно, этот атрибут не является необходимым. Вместо этого описание сущности БАНАНОВЫЙ ДЕСЕРТ должно быть указание на то, что банан включается в каждый ее экземпляр. То же самое касается атрибута Помадка в сущности СЛИВОЧНАЯ ПОМАДКА.

В Таблице 3.1 приведены области определения логических типов данных сущности СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ логической модели СМЕСЬ.

ТАБЛИЦА 3.1. Примеры логических типов данных

Имя атрибута	Логический тип данных
Идентификатор специального предложения	Number (число)
Идентификатор мороженого	Number (число)
Идентификатор вкуса	Number (число)
Начало специального предложения	Date (дата)
Конец специального предложения	Date (дата)

2.7.1. Области определения простых и расширенных типов данных

Область определения типа данных определяет способ представления значений атрибута. В завершенной логической модели область определения типа данных требуется для каждого атрибута. В следующем списке приведено несколько примеров логических типов данных ERwin:

- Datetime - дата/время
- Number - число
- String - строка

Многие из новых платформ баз данных поддерживают более развитые типы данных. Однако важно помнить, что эти сложные типы данных, за некоторым исключением, зависят от платформы. В любом случае, если пользователю нужен атрибут, он должен быть включен в модель, вне зависимости от его типа данных. Некоторые широко используемые расширенные типы данных приведены ниже:

- Image - изображение
- Sound - звук
- Video - видео

2.7.2. Области определения, вводимые пользователем

Области определения, вводимые пользователем - специализированные области определения, которые уточняют набор значений, допустимых для атрибута. Эти области определения часто специфичны для организации и должны определяться и использоваться единообразно в пределах корпорации. Например, атрибут с областью определения типа данных Number может, кроме того, иметь введенную пользователем область определения, которая ограничивает возможные значения пределами от 1 до 100. Принцип целостности дает корпорации возможность внести изменения в одной сущности расширить область определения для каждого из атрибутов, который ее использует.

2.8. Определение необязательности атрибута

Значение атрибута может быть обязательным или нет. Если значение требуется, или обязательно, значение должно присутствовать в момент создания экземпляра. Если значение необязательно, вы можете создавать экземпляры без него.

В книге **Information Engineering: Strategic Systems Development** К. Финклештейн определяет свойство обязательности атрибута через серию "правил редактирования":

- Добавляется сразу, изменить позже - нельзя.
- Добавляется сразу, изменяется позже.
- Добавляется позже, изменяется позже.
- Добавляется позже, изменить потом - нельзя.

Внимательно следите за необязательными атрибутами. Если атрибут или набор атрибутов имеет значение только для конкретных экземпляров сущности, рассмотрите возможность его переноса в сущность, где значение будет существовать для каждого из экземпляров.

В Таблице 3.2 указано свойство обязательности для атрибутов сущности СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ. Обратите внимание на тот факт, что при создании экземпляра сущности СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ значения требуются для всех атрибутов кроме атрибута Дата окончания специального предложения.

Таблица 3.2. Примеры обязательности атрибутов

Имя атрибута	Обязательность
Идентификатор специального предложения	Требуется
Идентификатор мороженого	Требуется
Идентификатор вкуса	Требуется
Начало специального предложения	Требуется
Конец специального предложения	Необязательно

В таблице 3.2 представлено бизнес-правило, которое говорит, что для экземпляра сущности СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ требуется следующая информация:

- Идентификатор сущности (Идентификатор специального предложения)

• Идентификатор вкусовой добавки специального предложения (Идентификатор мороженого и Идентификатор вкуса)
• Дата начала специального предложения (Начало специального предложения)

Дата окончания для каждого экземпляра сущности СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ необязательна. Бизнес-правило утверждает, что СПЕЦИАЛЬНОЕ ПРЕДЛОЖЕНИЕ должно иметь начало, но не обязательно должно иметь конец.

Атрибуты, значения которых обязательны, не могут иметь пустых значений. Некоторые эксперты считают, что значение должно быть обязательным для каждого экземпляра сущности. Естественно, в предположении, что значение каждого атрибута экземпляра сущности найдено или известно, до того, как экземпляр создается.

Атрибуты, чьи значения необязательны, могут иметь пустые значения. Некоторые эксперты считают, что атрибут не должен присутствовать в сущности, если его значение недоступно для каждого из ее экземпляров. Одной из причин является сложность интерпретации пустых значений. Означает ли пустое значение, что это значение неизвестно для экземпляра, или оно просто не было получено?

ПРИМЕЧАНИЕ

Среди разработчиков моделей дискуссия о недостатках и достоинствах требуемых и необязательных значений все еще продолжается. Одни разработчики уверены, что атрибут не должен иметь пустых значений, и утверждают, что область определения должна содержать такие значения, как неизвестен и неполучен. Другие считают обязательность значений и использование области определения также требует и использования значений по умолчанию, что приводит к ненадежным сомнительным значениям.

Предпочтительнее использовать пустые значения и переложить ответственность за работу с пустыми значениями на прикладную программу или средство формирования запросов. Это наиболее адекватное и гибкое решение, поскольку оно позволяет интерпретировать пустые значения по-разному для удовлетворения разных требований бизнеса.

2.9. Именование атрибутов

Каждый атрибут должен иметь ясное, точное и непротиворечивое имя. Имя атрибута не должно конфликтовать с его описанием. Имя атрибута должно указывать на значения, собираемые для экземпляров атрибута. Имя атрибута должно быть понятным и общепринятым в корпорации.

> **СОВЕТ**
>
> При выборе имени для атрибута, сохраняйте точку зрения корпорации и позаботьтесь о том, чтобы оно отражало смысл атрибута, а не то, как он будет использоваться. Используйте имена, осмысленные для сообщества пользователей и экспертов предметной области.

Вероятно, что у вас в корпорации есть набор соглашений об именовании атрибутов, разработанные в вашей корпорации или при формировании корпоративной модели данных, которыми вы руководствуетесь. Использование соглашений именования атрибутов гарантирует, что имена конструируются единообразно в рамках корпорации, вне зависимости от того, кто конструирует имя.

Соглашения об именовании атрибутов важны, вне зависимости от того, в маленькой или большой организации вы работаете. Однако, в большой организации с несколькими командами разработчиков и большим количеством пользователей, соглашения об именовании существенно помогают при взаимодействии и понимании элементарных данных. В идеале, вы должны разработать и сопровождать соглашения об именовании атрибутов централизованно и затем документально оформить и опубликовать их для всей корпорации.

Ниже представлены некоторые положения для формирования начального набора соглашений об именовании атрибутов, просто на случай, если в вашей организации пока такой набор не разработан:

- Имя атрибута должно быть достаточно описательным. Подумайте об использовании словосочетаний на основе существительных в форме объект/ модификатор/ класс.
- По возможности имя атрибута должно включать имя сущности. Используйте "Имя для персоны" вместо просто "Имя".
- Имя атрибута должно указывать на значения конкретных экземпляров атрибута. Использование одинаковых имен для атрибутов, содержащих различные данные, или разных имен для атрибутов, содержащих одинаковые данные, будет без необходимости вводить в заблуждение разработчиков и конечных пользователей.
- Имя атрибута должно использовать язык бизнеса вместо языка технических описаний.
- Имя атрибута не должно содержать специальных символов (таких как !, @, #, $, %, л, &, * и тому подобных) или указывать на принадлежность (Имя, принадлежащее персоне).

- Имя атрибута не должно содержать акронимов или аббревиатур, если только они не являются частью принятых соглашений именования.

Разработчикам моделей предпочтительно использовать хорошие соглашения именования, если таковые существуют, или разработать их, если таких соглашений нет.

2.9.1. Имена атрибутов в форме Объект/ Модификатор/ Класс

Форма объект/модификатор/класс - широко распространенное в отрасли соглашение об именовании атрибутов. Это соглашение побуждает использовать имена атрибутов, состоящие из трех частей. Часть объект иногда называют субъектом или основным словом. В качестве объекта обычно используется имя сущности.

Модификатор может быть одиночным термом или группой термов. Хотя списка стандартных модификаторов не существует, разработчикам моделей желательно формировать короткие осмысленные модификаторы. Использование модификаторов позволяет вам создавать наглядные осмысленные имена атрибутов. Если имя становится неприемлемо тяжеловесным для пользователей или широкого применения, как требуется в корпорации, вы можете пойти на компромисс, отказавшись от трехсложных имен атрибутов.

Базовой частью имени атрибута является класс, который определяет тип информации, представляемой атрибутом. Некоторые часто используемые классы:

- Идентификатор
- Код
- Дата
- Число
- Величина
- Количество
- Частота

2.9.2. Примеры имен атрибутов

В рамках корпорации всегда лучше использовать единообразные имена атрибутов. В Таблице 3.3 приведены примеры хороших и плохих имен для атрибутов. Обратите внимание, что слова в имени атрибута отделены пробелами, начинаются с заглавных букв и используют строчные символы для остальных.

ТАБЛИЦА 3.3. Имена атрибутов с пояснениями

Хорошее Имя	Неудачное имя	Пояснение
Person First Name (Имя персоны)	Name (Имя)	Name (Имя) - название класса и нуждается в обозначении объекта Person (персона) и в модификаторе First (первое).
Ice Cream Sales Quantity (Объем продаж мороженого)	The Quantity of Sales (Объем продаж)	Quantity (Количество) - название класса и должно быть на последнем месте (в английском варианте имени атрибута). "The" и "of" не привносят дополнительного смысла.
Item Cost Amount (Величина стоимости позиции)	Cost of Item (Стоимость позиции)	"of" не привносит дополнительного смысла. Название класса "Amount" (величина) указывает пользователю, что должно быть в атрибуте.
Product Identifier (Идентификатор продукта)	Product Identifiers (Идентификаторы продуктов)	"Identifiers" (Идентификаторы) - множественное число. Имя атрибута должно быть существительным в единственном числе.
Point of Sale Location Code (Код местоположения	POS Code (Код POS)	"POS" - аббревиатура. Использованное название класса "Code" (код) нуждается в модификаторе.

точки продаж)		
Person Birth Date (Дата рождения персоны)	Birthday (День рождения)	Birthday (День рождения) не содержит названия класса Date (Дата). Включение модификатора и имени объекта проясняет смысл имени атрибута.

2.9.3. Описание атрибутов

Описание атрибута должно быть коротким пояснением смысла атрибута, а не того, как он используется. Описание атрибута не должно противоречить его имени и не должно быть простым повторением имени. Используйте название класса и объекта в утверждении для точного описания данных. Если атрибут выводится или рассчитывается, включайте правила вывода или формулы расчета. Следующие правила касаются описания атрибутов:

- Описание атрибута должно быть ясным, полным и однозначным.
- Описание атрибута должно соответствовать его имени.
- Описание атрибута не должно опираться на описание другого атрибута.
- Описание атрибута должно формулироваться на языке бизнеса, а не на языке технических описаний.
- Имя атрибута должно отражать его смысл, а не то, как он используется.
- В описании атрибута должны быть расшифрованы все аббревиатуры и акронимы, использованные в его имени.

Разработчикам моделей рекомендуется давать хорошие описания для каждого из атрибутов. Хорошие описания атрибутов делают легким использование модели для всех. Те, кто использует модель, созданную хорошим разработчиком, испытывают удовольствие от хорошо сформулированных в модели требований к информации. Сравните примеры из таблицы 3.4.

Таблица 3.4. Имена и описания атрибутов с пояснениями

Имя атрибута	**Хорошее описание**	**Неудачное описание**	**Пояснение**
Person First Name	Имя персоны, которое	Поле с длиной в	Не используется язык

(Имяперсоны)	позволяет корпорации общаться с персоной, используя дружеские обращения.	40 символов.	бизнеса. Применены технические термины.
Ice Cream Sales Quantity (Объем продаж мороженого)	Количество мороженого конкретного сорта, проданного в рамках конкретного мероприятия по продаже.	Объем продаж.	Не добавляет нового смысла, а просто перефразирует имя атрибута в расплывчатых терминах.
Item Cost Amount (Величина стоимости позиции)	Величина стоимости конкретной позиции в конкретный период времени. Представляет суммарную стоимость продажи и доставки.	Шестизначное десятичное число с двумя знаками после запятой.	Слишком техническое описание. Почти ничего не значит для пользователей элемента данных.
Product Identifier (Идентификатор продукта)	Искусственный уникальный числовой идентификатор для конкретного продукта.	Идентификаторы продуктов.	Простая перефразировка имени атрибута.
Point of Sale Location Code (Код местоположения точки продаж)	Уникальный код, идентифицирующий географическое положение точки продаж.	Код POS.	Использованный акроним может быть непонятен пользователям. Кроме того, в описании опущен важный модификатор.
Person Birth Date (Дата рождения персоны)	Дата рождения персоны.	День рождения персоны.	В описании опущено название класса "дата".

2.10. Распространенные ошибки при работе с атрибутами

Этот раздел, посвященный распространенным ошибкам при моделировании атрибутов, не претендует на полноту. Цель его - указать на наиболее распространенные ошибки, которые встречаются у разработчиков моделей.

Иногда, при моделировании чего-либо определенным способом, разработчик модели делает сознательный выбор, руководствуясь вполне правильными принципами. Очень важно понимать всю причинно-следственную цепочку принимаемых решений и результаты, к которым они могут привести.

2.10.1. Моделирование в терминах значений

Что понимается под моделированием в терминах значений? Во время рабочей сессии пользователи могут сказать вам, что им нужен набор атрибутов, указывающих на возрастные категории экземпляра сущности ПЕРСОНА. В этом сценарии возникают, по меньшей мере, три проблемы:

1 Способ определения возрастных категорий в корпорации со временем может измениться.
2 Возраст конкретной персоны определенно меняется с течением времени.
3 Все атрибуты будут представлять значения атрибута **Возраст персоны**. Естественно, **Возраст персоны** с течением времени будет меняться, так что лучшим решением будет использование в модели простого атрибута **Дата рождения персоны.**

> **СОВЕТ**
>
> Если приобретенные или производные данные, такие как возрастная категория, это единственное, что доступно, тогда они, конечно, должны быть включены в модель. Если доступны и возрастная категория и дата рождения, включите в модель и категорию, и дату рождения, и выводите возрастную категорию из даты рождения для экземпляров, для которых дата рождения известна. Как часто придется делать вывод и будет ли возрастная категория иметь физическое представление, зависит от требований к использованию.

2.10.2. Моделирование многозначных атрибутов

Многозначными являются атрибуты, имеющие несколько значений для концепции. Проверьте описания атрибутов, которые указывают на наличие нескольких значений для одной и той же концепции.

Иногда эксперты из различных предметных областей в корпорации используют имя атрибута, которое пишется и

произносится одинаково , но имеет разные значения для разных экспертов. Один из способов убедиться в том, что атрибуты с одинаковыми именами описывают одинаковые объекты , заключается в проверке описаний. Убедитесь, что значения атрибутов описывают единственную концепцию.

Например, вы можете создать искусственные коды путем соединения одного или нескольких кодов для связи прежде не связанных данных. Текстовые фрагменты могут скрывать множество ценных атрибутов и значений.

Неудачное разрешение многозначных атрибутов может привести к тому, что некоторые важные бизнес-правила останутся необнаруженными и недокументированными.

2.10.3. Моделирование избыточных атрибутов

Избыточными являются атрибуты с разными именами, но содержащие информацию о сходных концепциях. Во всех языках существует много слов, представляющих одни и те же вещи. Один из способов найти избыточные атрибуты - просмотреть сущности с похожими атрибутами. Сравните описания всех атрибутов для проверки того, не содержат ли эти сущности данные о сходных концепциях. Избыточные атрибуты часто являются результатом тенденции к моделированию значений в качестве атрибутов. Например, сущности УПРАВЛЕНЕЦ и ИСПОЛНИТЕЛЬ могут содержать атрибуты Имя менеджера и Имя исполнителя. Так как и УПРАВЛЕНЕЦ, и ИСПОЛНИТЕЛЬ являются ролями, в которых может выступать экземпляр сущности ПЕРСОНА, вы можете переместить туда этот атрибут и назвать его Имя ПЕРСОНЫ.

2.10.4. Использование неудачных имен для атрибутов

Неясные, неоднозначные или неточные имена атрибутов усложняют для новых пользователей и команд разработчиков повторное использование или развитие существующей модели.

Не используйте аббревиатур или акронимов в качестве части имени атрибута. Аббревиатуры и акронимы открыты для неправильной интерпретации и даже могут иметь разное значение в разных предметных областях.

Не используйте имена собственные, указывающие на значение для конкретного экземпляра. Имя атрибута, использующее имя собственное - индикатор серьезных проблем при моделировании, заключающихся не только в неудачном выборе имени. Не включайте месторасположение в качестве части имени атрибута. Если значение существует для одного месторасположения, оно определенно существует и для другого месторасположения. Имя атрибута с указанием расположения является признаком того, что вы моделируете конкретный экземпляр вместо класса.

2.10.5. Использование неудачных описаний для атрибутов

Не используйте описаний атрибутов, заимствованных только из словаря. Описания из словаря не будут включать информацию, значимую для бизнеса, которая делает атрибут важным для корпорации. Не используйте простое перефразирование имени атрибута. Не используйте имени атрибута в его описании.

Неясное, неопределенное описание атрибута, или что еще хуже - его отсутствие, затрудняет повторное использование или развитие существующей модели. Пользователи не смогут проверить, что модель содержит все требования к информации. Это так же повышает вероятность использования в модели вместо атрибутов конкретных значений и многозначных атрибутов.

Концепции, которые кажутся очевидными для всех участников рабочих сессий, могут перестать быть столь очевидными с течением времени, когда перед новой командой разработчиков будет поставлена задача развить существующую модель.

2.11. Заключение

Сущности представляют собой факты, информацию о которых корпорация заинтересована накапливать и сопровождать. Они составляют существо модели и в основном выявляются во время рабочих сессий. Полное и точное отражение атрибутов в модели требует тщательного анализа, гарантирующего, что атрибуты точно соответствуют требованиям к информации. Атрибут должен присутствовать в модели в единственном экземпляре и должен представлять единственную концепцию бизнеса. Для помещения атрибутов в соответствующие сущности должны использоваться правила нормализации.

Атрибуты могут быть ключевыми и неключевыми. Ключ может быть единственным атрибутом или группой атрибутов. Первичные ключи выбираются из кандидатов в ключи, которые уникально идентифицируют экземпляр сущности. Атрибуты первичного ключа мигрируют из исходной сущности, чтобы стать внешними ключами вторичных сущностей. Значения неключевых атрибутов должны функционально зависеть от значения первичного ключа.

Область определения задает набор значений атрибута. Логические области определения могут быть простыми типами данных, такими как числа или строки. Они так же могут быть сложными типами данных, определяемыми пользователем, которые приспособлены для удовлетворения специфических требований корпорации. Новые СУБД поддерживают расширенные типы данных, такие как изображения и звук.

Значения атрибута могут быть требуемыми или необязательными. Если значение требуемое, атрибут не может иметь пустых значений. Атрибут должен иметь имя и описание. При именовании атрибутов рекомендуется использовать

стандарт именования в форме объект/ модификатор/ класс. Каждый атрибут должен включать хорошее описание, использующее терминологию бизнеса для определения сущности атрибута, а не того, как он будет использоваться.

Сущности и отношения служат для группировки связанных атрибутов. В последующих статьях будет рассказано о роли отношений в процессе моделирования и о свойствах и типах отношений, там же вы найдете некоторые советы, помогающие избежать распространенных ошибок при моделировании отношений.