Лекция 4

Реляционные объекты данных

Основные понятия и ограничения реляционной модели (от английского relation - отношение) впервые были сформулированы сотрудником компании IBM Е.Ф.Коддом в 1970 г.

Реляционная модель связана с тремя аспектами данных: объектами данных (структурой данных), целостностью данных и обработкой данных [1, 2].

Основной структурной частью (объектом) реляционной модели является отношение. Основные понятия

Рассмотрим наиболее важные термины, используемые в структурной части реляционной

модели.

Декартово произведение Для заданных конечных множеств Di, D2,...,Dn (не обязательно различных) декартовым произведением D1x D2x...x Dn называется множество произведений вида: d1x d2X...x dn , где dieDi, d2eD2,..., dneDn.

Пример: Имеем три домена D1={a,b,c}, D2={m,k}, D3={y,z}.

Декартово произведение этих доменов D = D1 xD2xD3 =(a x m x y, a x m x z, a x k x y, a x k x z,

b x m x y, b x m x z, b x k x y, b x k x z, c x m x y, c x m x z, c x k x y, c x k x z)

Отношением R, определенным на множествах D1? D2,...,Dn (n > 1), необязательно различных,

называется подмножество декартова произведения D1x D2x...x Dn.

Исходные множества D1? D2,...,Dn называются доменами отношения Элементы декартова произведения d1x d2x...x dn называются кортежами Число n определяет степень отношения ( n=1 - унарное, n=2 - бинарное, ..., n-арное) Количество кортежей называется кардинальным числом или мощностью отношения Домен представляет собой именованное множество атомарных значений одного типа. Под атомарным значением понимается “наименьшая семантическая единица данных”, т.е. это значение, не имеющее внутренней структуры при рассмотрении в реляционной модели. Это не значит, что такое значение не имеет внутренней структуры вообще. Например, название должности состоит из букв, но, разложив название по буквам, мы потеряем значение.

Домены являются общими совокупностями значений, из которых берутся конкретные значения атрибутов. Т.е. каждый атрибут должен быть определен на основе одного домена; это значит, что значения атрибута должны браться из этого домена.

Значение доменов заключается в том, что домены ограничивают сравнения. Т.е. если два атрибута определены на одном и том домене, то их можно сравнивать, применяя операции сравнения допустимые для данного домена. Например, атрибуты Дата приема на работу и Дата окончания ВУЗа определены на одном домене Даты; для этого домена допустимы операции сравнения: =, Ф, <, <, >, >. Поэтому данные атрибуты можно сравнивать, используя все указанные операции сравнения.

Отношение удобно представить в виде таблицы, столбцы которой соответствуют вхождениям доменов в отношение, а строки - наборам из n значений, взятых их исходных доменов, и расположенным в соответствии с заголовком отношения (Рис. 2-15). Столбцы отношения называют атрибутами, а строки - кортежами. Однако нельзя сказать, что отношение и таблица полностью идентичны. Различие между отношением и таблицей мы рассмотрим чуть позже, когда будем рассматривать свойства отношений.

Отношение содержит две части: заголовок и тело (заголовок - это строка заголовков столбцов, тело - это множество строк данных).

Заголовок (или схема отношения) содержит фиксированное множество атрибутов или, точнее, пар <имя-атрибута : имя-домена>:

{<A1:D1>, <A2:D2>, <An:Dn>},

причем каждый атрибут Aj соответствует только одному из лежащих в основе доменов Dj (j = 1, 2, ..., n). Все имена атрибутов A1, A2, An разные.

Схемы двух отношений называются эквивалентными, если они имеют одинаковую степень и возможно такое упорядочение имен атрибутов в схемах, что на одинаковых местах будут находиться сравнимые атрибуты, т.е. атрибуты, принимающие значения из одного домена.

Схема БД (в структурном смысле) - это набор именованных схем отношений. Тогда реляционная БД - это набор отношений, имена которых совпадают с именами схем отношений в схеме БД.

Тело содержит множество кортежей.

Каждый кортеж, в свою очередь, содержит множество пар

<имя-атрибута : значение-атрибута>:

{<A1:vi1>, <A2:vi2>, <An:vin>},

(i = 1, 2, ..., m, где m - количество кортежей в этом множестве). В каждом таком кортеже есть одна такая пара <имя-атрибута : значение-атрибута>, т.е. <Aj:vij>, для каждого атрибута Aj в заголовке. Для любой такой пары <Aj:vij> vij является значением из уникального домена Dj, связанного с атрибутом Aj.

Т.е. можно сказать, что отношение - это множество кортежей, соответствующих одной схеме отношения.

Атрибут, значение которого однозначно идентифицирует кортежи, называется ключевым (или просто ключом). Если кортежи идентифицируются только сцеплением значений нескольких атрибутов, то говорят, что отношение имеет составной ключ.

Отношение может содержать несколько ключей. Всегда один из ключей объявляется первичным, его значения не могут обновляться. Все остальные ключи отношения называются возможными (потенциальными или альтернативными) ключами.

Пример: Для иллюстрации введенных терминов рассмотрим отношение Расписание, приведенное на Рис. 1. В этом отношении есть четыре основных домена: домен номеров рейса (№ рейса), домен наименований населенных пунктов (Населенные пункты), домен времени (Время) и домен типов поездов (Тип поезда).

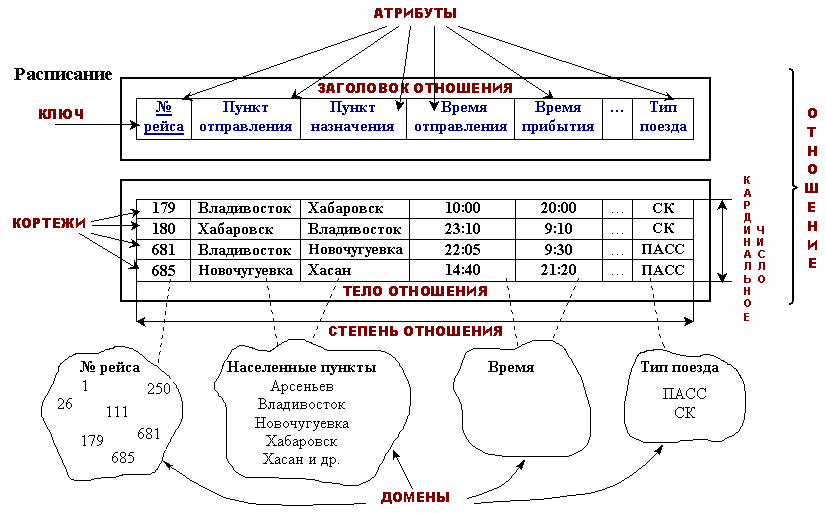


Рисунок 1. Пример отношения ж/д расписание

Обратите внимание, что количество доменов меньше количества атрибутов, т.е. некоторые атрибуты определены на одном и том же домене. Так атрибуты Пункт отправления и Пункт назначения определены на домене Населенные пункты, а атрибуты Время отправления и Время прибытия - на домене Время. Т.е. атрибуты Пункты отправления и Пункт назначения (и соответственно Время отправления и Время прибытия) можно сравнивать. Например, для определения транзитных рейсов, которыми можно добраться из Владивостока в Хасан, необходимо сравнить пункт назначения с пунктом отправления, а также время прибытия одного рейса с временем отправления другого (чтобы узнать есть ли запас времени для пересадки на транзитный рейс).

Схема отношения (заголовок отношения) выглядит как (№ рейса, Пункт отправления, Пункт назначения, Время отправления, Время прибытия, Тип поезда) или по определению схема представляет собой набор упорядоченных пар:

{<№ рейса : № рейса>,

<Пункт отправления : Населенные пункты>,

<Пункт назначения : Населенные пункты>,

<Время отправления : Время>,

<Время прибытия : Время>,

<Тип поезда : Тип поезда>},

где первым компонентом каждой пары является имя атрибута, а вторым компонентом - имя соответствующего домена. На практике чаще всего имена доменов в схеме опускают, и схема отношения представляет собой перечень атрибутов отношения.

Тело отношения представляет собой набор строк (кортежей). Рассмотрим подробнее один из кортежей:

(681, Владивосток, Новочугуевка, 22:05, 9:30, ПАСС)

по определению этот кортеж представляет собой набор упорядоченных пар:

{<№ рейса : 681>,

<Пункт отправления : ‘Владивосток’>,

<Пункт назначения : ‘Новочугуевка’>,

<Время отправления : 22:05>,

<Время прибытия : 9:30>,

<Тип поезда : ‘ПАСС’>},

где первым компонентом каждой пары является имя атрибута, а вторым компонентом - значение соответствующего атрибута.

Часто на практике имена атрибутов опускают, так как известно, что каждое отдельное значение в таблице является значением атрибута, имя которого находится сверху соответствующего столбца; кроме того, значение принадлежит лежащему в основе этого атрибута домену. Например, значение “Владивосток” - это значение атрибута Пункт отправления, и оно взято из домена Населенные пункты.

Ключевым атрибутом отношения Расписание будет атрибут № рейса, т.к. он однозначно идентифицирует кортежи. В самом деле, нет ни одного повторяющегося номера рейса, и по конкретному номеру рейса мы можем найти соответствующий кортеж отношения.

Свойства отношений

Рассмотрим теперь свойства отношений, которые следуют из приведенного выше определения отношения. В любом отношении

* Отсутствуют одинаковые кортежи
* Отсутствует упорядоченность кортежей
* Отсутствует упорядоченность атрибутов
* Все значения атрибутов атомарные Отсутствие одинаковых кортежей

Это свойство следует из определения отношения как множества кортежей, а множества в математике по определению не содержат одинаковых элементов.

Это свойство служит прекрасным примером различия отношения и таблицы, т.к. таблица вполне может содержать одинаковые строки, а отношение не может содержать одинаковые кортежи.

Важным следствием этого свойства является наличие у каждого отношения так называемого первичного ключа - набора атрибутов, значения которых однозначно определяют кортеж отношения. Для каждого отношения, по крайней мере, полный набор его атрибутов обладает этим свойством. Однако при формальном определении первичного ключа требуется обеспечение его "минимальности", т.е. в набор атрибутов первичного ключа не должны входить такие атрибуты, которые можно отбросить без ущерба для основного свойства - однозначно определять кортеж. Понятие первичного ключа является исключительно важным в связи с понятием целостности баз данных.

Отсутствие упорядоченности кортежей

Свойство отсутствия упорядоченности кортежей (сверху вниз) также следует из того, что тело отношения - это математическое множество, а простые множества в математике не упорядочены.

Второе свойство отношений также служит примером различия отношения и таблицы, т.к. строки таблицы упорядочены сверху вниз, а кортежи отношения - нет.

Отсутствие требования к поддержанию порядка на множестве кортежей отношения дает дополнительную гибкость СУБД при хранении баз данных во внешней памяти и при выполнении запросов к базе данных. Это не противоречит тому, что при формулировании запроса к БД, например, на языке SQL можно потребовать сортировки результирующей таблицы в соответствии со значениями некоторых столбцов. Такой результат, вообще говоря, не отношение, а некоторый упорядоченный список кортежей.

Отсутствие упорядоченности атрибутов

Свойство отсутствия упорядоченности атрибутов (слева направо) следует из того факта, что схема отношения также определена как множество пар {имя атрибута, имя домена}. Для ссылки на значение атрибута в кортеже отношения всегда используется имя атрибута.

Это свойство также иллюстрирует отличие таблицы от отношения, поскольку столбцы таблицы упорядочены слева направо, а атрибуты отношения - нет.

Атомарность значений атрибутов

Значения всех атрибутов являются атомарными. Это свойство является следствием того, что все домены, лежащие в основе отношения, содержат только атомарные значения. Иначе можно сказать, что в каждой позиции пересечения столбца и строки таблицы расположено в точности одно значение, а не набор значений. Отношение, удовлетворяющее этому условию, называется нормализованным (представленным в первой нормальной форме). Т.е. с точки зрения реляционной модели все отношения нормализованы, поэтому в реляционных базах данных допускаются только нормализованные отношения или отношения, представленные в первой нормальной форме. Примером ненормализованного отношения является отношение R1 на Рис.2. Чтобы можно было использовать отношение в реляционной БД, его необходимо привести в виду отношения R2 (Рис. 2­16). Процесс получения отношения R2 из R1 называется нормализацией.

Это свойство также иллюстрирует отличие таблицы от отношения. Строго говоря, на Рис. 2, только R2 является отношением, а таблицей можно назвать как R1, так и R2.

R1 - Ненормализованное отношение R2- Нормализованное отношение

|  |  |  |
| --- | --- | --- |
| КодПо с т авщик а | КодПродукта | Продукт |
| Pi | 1 | Сахар |
|  | 2 | Соль |
|  | 13 | Мука |
| Рз | 26 | Рис |
|  | 58 | Гречка |
|  | 130 | Крупа манная |
|  | 162 | Пшено |
| Рз | 474  891 | Молоко  Кефир |

|  |  |  |
| --- | --- | --- |
| КодПо с т авщик а | КодПродукта | Продукт |
| Pi | 1 | Сахар |
| Pi | 2 | Соль |
| Pi | 13 | Мука |
| Рз | 26 | Рис |
| Р2 | 58 | Гречка |
| Р2 | 130 | Крупа манная |
| Рз | 162 | Пшено |
| Рз | 474 | Молоко |
| Рз | 891 | Кефир |

Рисунок 2. Пример нормализации отношения

2.2.2. Ограничения целостности в реляционной модели данных

Вторым аспектом реляционной модели данных является поддержка целостности.

Целостность данных понимается как правильность данных в любой момент времени при манипулировании данными. Поддержание целостности базы данных может рассматриваться как защита данных от неверных изменений или разрушений.

В классическом понимании поддержка целостности включает 3 части:

* Структурная целостность
* Языковая целостность
* Ссылочная целостность

Эти 3 вида целостности определяют допустимую форму представления и обработки информации в реляционных БД.

Для определения некоторых ограничений, связанных с содержанием БД, используется другой вид целостности, а именно:

Семантическая целостность

Структурная целостность

Структурная целостность подразумевает, что реляционная СУБД может работать только с реляционными отношениями. А реляционное отношение, в свою очередь, должно удовлетворять ограничениям, накладываемым на него в классической теории реляционных БД (отсутствие одинаковых кортежей и, следовательно, наличие первичного ключа, отсутствие упорядоченности атрибутов и кортежей).

Требование структурной целостности осуществляется с помощью двух ограничений:

* при добавлении кортежей в отношение проверяется уникальность их первичных ключей
* не допускается, чтобы какой-либо атрибут, участвующий в первичном ключе, принимал

неопределенное значение

Здесь возникает необходимость рассмотреть проблему неопределенных значений (Null- значений) [1, 2]. Неопределенное значение интерпретируется в реляционной модели как значение, неизвестное на данный момент времени. При сравнении неопределенных значений не действуют стандартные правила сравнения: одно Null-значение никогда не считается равным другому Null- значению.

Для выявления равенства значения некоторого атрибута неопределенному применяют стандартные предикаты:

* <Имя атрибута> Is Null
* <Имя атрибута> Is Not Null

Таблица 1 содержит пример проверки атрибута Адрес на неопределенное значение.

Таблица 1

|  |  |  |
| --- | --- | --- |
| Адрес | Адрес Is Null | Адрес Is Not Null |
| Null | True | False |
| ул.Мордовцева, 12 | False | True |

Введение Null-значений привело к модификации классической двузначной логики к трехзначной. Таблица 2 содержит таблицу истинности для трехзначной логики.

Таблица 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Not A | A & B | A v B |
| True | True | False | True | True |
| True | False | False | False | True |
| True | Null | False | Null | True |
| False | True | True | False | True |
| False | False | True | False | False |
| False | Null | True | False | Null |
| Null | True | Null | Null | True |
| Null | False | Null | False | Null |
| Null | Null | Null | Null | Null |

Языковая целостность

Языковая целостность состоит в том, что реляционная СУБД должна обеспечивать языки описания и манипулирования данными не ниже стандарта SQL. Не должны быть доступны иные низкоуровневые средства манипулирования данными, не соответствующие стандарту.

Ссылочная целостность

При установлении связи между отношениями возникает необходимость поддержания целостности по ссылкам. Отношение со стороны «один» будем называть - основным отношением, а отношение со стороны «многие» - подчиненным.

Требование ссылочной целостности состоит в следующем: для каждого значения внешнего ключа, появляющегося в подчиненном отношении, в основном отношении должен существовать кортеж с таким же значением первичного ключа.

У первичного и внешнего ключей, образующих связь, должен быть одинаковый тип данных.

То есть значение внешнего ключа должно либо:

* быть равным значению первичного ключа
* быть полностью неопределенным, т.е. каждое значение поля, участвующего во внешнем

ключе должно быть неопределенным.

Для каждого внешнего ключа в процессе проектирования необходимо решить три вопроса:

1. Может ли данный внешний ключ принимать неопределенные значения
2. Что произойдет при попытке УДАЛЕНИЯ записи из основного отношения, на которую ссылается внешний ключ подчиненного отношения?

Например, удалить поставщика, для которого имеется, по крайней мере, одна поставка.

В общем случае существует три ситуации:

* Каскадирование удаления, при котором удаляются все записи из подчиненного отношения,

соответствующие удаляемому первичному ключу основного отношения (будет удален поставщик и все его поставки)

* Ограничение удаления, при котором удаляется запись из основного отношения только в том

случае, если в подчиненном отношении нет соответствующих значений внешнего ключа, иначе удаление отменяется (удаление поставщика невозможно, пока существует хотя бы одна его поставка)

* Установка неопределенных значений, при которой внешний ключ подчиненного отношения устанавливается в неопределенное значение (Null-значание), а соответствующая запись из основного отношения удаляется (все значения внешнего ключа в поставках принимают Null-значение, а поставщик удаляется)

Данное свойство поддерживается не всеми СУБД. Если необходимо применить эту ситуацию, то в подчиненном отношении сначала нужно удалить все значения внешнего ключа соответствующие первичному, и только после этого удалять запись из основного отношения с соответствующим первичным ключом

1. Что произойдет при попытке ОБНОВЛЕНИЯ первичного ключа основного отношения, на который ссылается некоторый внешний ключ подчиненного отношения? Например, при попытке обновления кода поставщика, для которого имеется хотя бы одна поставка.

Здесь также возможны три ситуации:

* Каскадирование обновления, при котором при обновлении первичного ключа обновляются

все соответствующие внешние ключи (будет обновлен код поставщика в основном отношении и все соответствующие ему внешние ключи в поставках)

* Ограничение обновления, при котором обновляется первичный ключ в основном отношении только в том случае, если в подчиненном отношении нет соответствующих значений внешнего ключа, иначе обновление отменяется (обновление кода поставщика невозможно, пока существует хотя бы одна поставка этого поставщика)
* Установка неопределенных значений, при которой внешний ключ подчиненного отношения устанавливается в неопределенное значение, а соответствующий первичный ключ в основном отношении обновляется (все значения внешнего ключа в поставках принимают Null-значение, а код поставщика в основном отношении обновляется)

Семантическая целостность

Данный вид целостности задается разработчиком в процессе проектирования БД посредством задания ограничений для свойств полей. Обычно задаются ограничения свойств:

* уникальность значений полей. Например, в отношении Студент(№ зачетной книжки, ФИО,

Паспорт, Адрес) свойство уникальности значений должно быть установлено для атрибутов: № зачетной книжки (т.к. это первичный ключ) и Паспорт (т.к. номера всех паспортов уникальны)

* обязательность заполнения полей (допустимость или недопустимость Null-значений).

Например, при вводе данных о поставщиках не вся информация может быть доступна сразу: адрес, телефоны для связи могут быть уточнены позднее. Т.е. для атрибутов Код города, Адрес, Телефон устанавливается допустимость Null-значений

* значение по умолчанию. Задание значения по умолчанию по умолчания означает, что

каждый раз при вводе новой строки в отношение, при отсутствии данных этому атрибуту присваивается значение по умолчанию .Например, если большинство поставщиков находятся во Владивостоке, то для атрибута Код города присваивается значение по умолчанию соответствующее коду Владивостока

* диапазон значений Например, оценки выставляются по пяти бальной шкале от 1 до 5, тогда

условие для этого диапазона (для MS Access) будет выглядеть как: Between 1 And 5

• принадлежность набору значений Например, атрибут РезультатЗачета может принимать значения только «Зачтено» или «Не зачтено», тогда условие на проверку принадлежности набору значений (для MS Access) будет выглядеть как: “Зачтено” Or “Не зачтено”.