Тема 1.4 Реляционная модель данных

Реляционная модель предложена сотрудником компании IBM Е.Ф.Коддом. В настоящее время эта модель является фактическим стандартом, на который ориентируются практически все современные коммерческие СУБД.

Представление данных не зависит от способа их физической организации. Это обеспечивается за счет использования математической теории отношений (само название "реляционная" происходит от английского relation - "отношение").

Реляционная модель состоит из трех частей:
- структурной части;
- целостной части;
- манипуляционной части.

Структурная часть описывает, какие объекты рассматриваются реляционной моделью.
Единственной структурой данных в реляционной модели являются нормализованные n-арные отношения.
Целостная часть описывает ограничения специального вида, которые должны выполняться для любых отношений в любых реляционных базах данных. Это целостность сущностей и целостность внешних ключей.
Манипуляционная часть описывает два эквивалентных способа манипулирования реляционными данными - реляционную алгебру и

Декартово произведение Для заданных конечных множеств **D1, D2,…,Dn** (не обязательно различных) декартовым произведением **D1× D2×…× Dn** называется множество произведений вида: **d1×d2×…× dn** , где **d1∈D1, d2∈D2,…, dn∈Dn**.

Пример: Имеем два домена D1={a,b,c}, D2={m,k}

Декартово произведение этих доменов

D = D1×D2 =(a × m , a × k, b × m, b × k, c × m, c × k)

Отношением **R**, определенным на множествах **D1, D2,…,Dn** (n ≥ 1), необязательно различных, называется подмножество декартова произведения **D1× D2×…× Dn**. Исходные множества **D1, D2,…,Dn** называются **доменами** отношения. Элементы декартова произведения **d1× d2×…× 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

Основными понятиями реляционных баз данных являются: тип данных, домен, атрибут, кортеж, первичный ключ и отношение. Значения атрибута должны браться из домена.

Домен имеет уникальное имя (в пределах базы данных).

Домен определен на некотором простом типе данных или на другом домене.

Домен может иметь некоторое логическое условие, позволяющее описать подмножество данных, допустимых для данного домена.

Домен несет определенную смысловую нагрузку.

Например, домен , имеющий смысл "возраст сотрудника" можно описать как следующее подмножество множества натуральных чисел:

16 < возраст сотрудника < 60

Реляционная модель требует, чтобы типы используемых данных были простыми (не структурированные или ссылочные типы данных.

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 . Столбцы отношения называют атрибутами, а строки – кортежами.

Число атрибутов в отношении называют степенью (или - арностью) отношения.

Мощность множества кортежей отношения называют мощностью отношения.

Реляционной базой данных называется набор отношений.
Схемой реляционной базы данных называется набор заголовков отношений, входящих в базу данных.

Различия между отношениями и таблицами

В отношении нет одинаковых кортежей (ключи у всех разные).
Кортежи не упорядочены. Нельзя сказать, что сотрудник Иванов "предшествует" сотруднику Перову.
Атрибуты не упорядочены (слева направо). Т.к. каждый атрибут имеет уникальное имя в пределах отношения, то порядок атрибутов не имеет значения.
Все значения атрибутов атомарны, среди значений домена не могут содержаться множества значений (отношения).
Для того, чтобы некоторая таблица задавала отношение, необходимо, чтобы таблица имела простую структуру (содержала бы только строки и столбцы, причем, в каждой строке было бы одинаковое количество полей), в таблице не должно быть одинаковых строк, любой столбец таблицы должен содержать данные только одного типа, все используемые типы данных должны быть простыми.

Отношение может содержать несколько ключей. Всегда один из ключей объявляется первичным, его значения не могут обновляться.

Связь между отношениями ОТДЕЛ и СОТРУДНИК создается путем копирования первичного ключа "Номер_отдела" из первого отношения во второе.

Атрибуты, представляющие собой копии ключей других отношений, называются внешними ключами.

Ограничения целостности в реляционной модели данных

Поддержка целостности включает:

- **Структурная целостность**
- **Языковая целостность**
- **Ссылочная целостность**
- **Семантическая целостность**

Структурная целостность подразумевает, что реляционная СУБД может работать только с реляционными отношениями.
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.

Для выявления равенства значения некоторого атрибута неопределенному применяют стандартные предикаты:

• <Имя атрибута> Is Null
• <Имя атрибута> Is Not 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 (поддержание целостности по ссылкам)

Отношение со стороны «один» будем называть – основным отношением, а отношение со стороны «многие» – подчиненным.

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.

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 *Кодгорода, 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 в Ростове, то для атрибута *Код города присваивается значение по* умолчанию соответствующее коду Ростова (=‘863’)

• ***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”.

Специальные правила преобразования ER-диаграмм в реляционные модели

Метод основывается на формировании набора предварительных таблиц из ER-диаграмм.

Для каждой сущности создается таблица. Причем каждому атрибуту сущности соответствует столбец таблицы.

Правила генерации таблиц из ER-диаграмм опираются на два основных фактора – тип связи и принадлежность сущности.

Удобно имена атрибутов в масштабе ER-модели сделать уникальными, тогда при построении реляционной модели их (почти никогда) не придется переименовывать

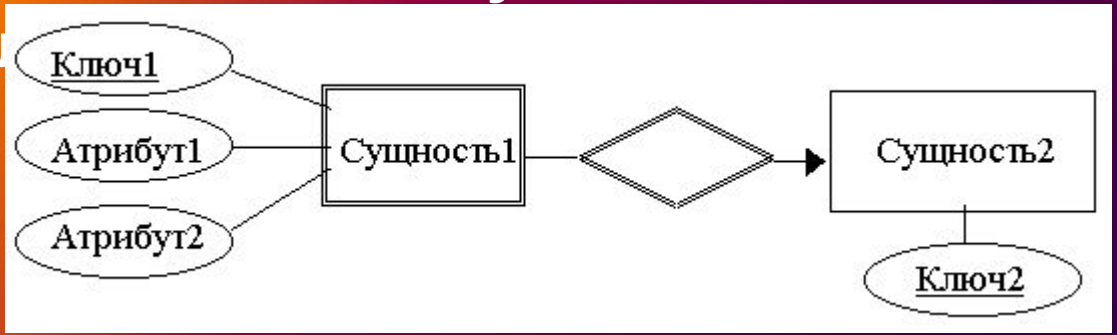
Преобразование обычной сущности

Обычная сущность преобразуется в отдельную таблицу, полями таблицы будут все атрибуты сущности:

Сущность (Ключ, Атрибут1, Атрибут2)

Преобразование слабой сущности

Сущность, которая не идентифицируется с помощью собственных атрибутов называется "слабой сущностью, она определяется через связь с другой сущностью. На схеме слабые сущности обозначаются двойными



Слабая сущность преобразуется в отдельную таблицу, полями таблицы будут все атрибуты сущности плюс ключевые атрибуты всех сущностей, с помощью которых данная слабая сущность идентифицируется. Ключевые поля всех родительских таблиц войдут в ключ дочерней таблицы. Для дочерней таблицы они будут называться внешним ключом.

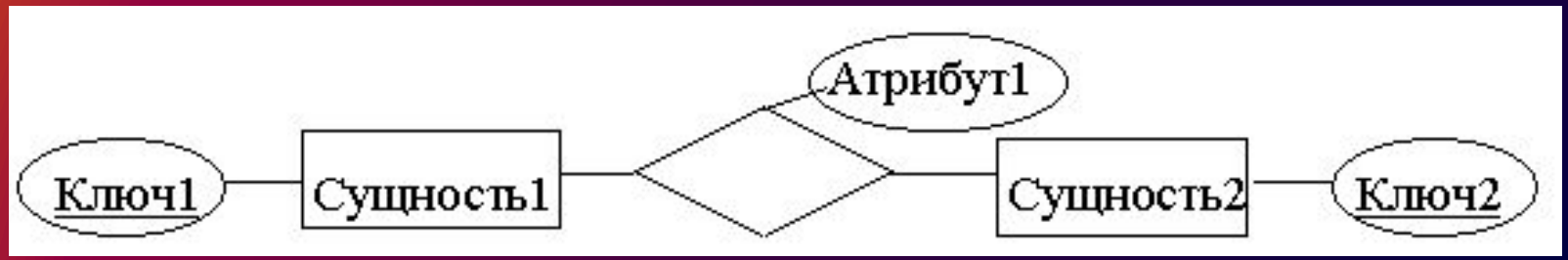
Сущность1 (Ключ1, Ключ2, Атрибут1, Атрибут2)

Преобразование связей

Для одинарных связей ничего делать не нужно, вся информация уже хранится в таблице слабой сущности.

Связь M:M

Создается новая таблица, содержащая ключевые поля каждой сущности, участвующей в связи, и собственные атрибуты связи, если таковые имеются. В названии обычно отражают, какие именно сущности связываются, или называют новую таблицу именем связи.

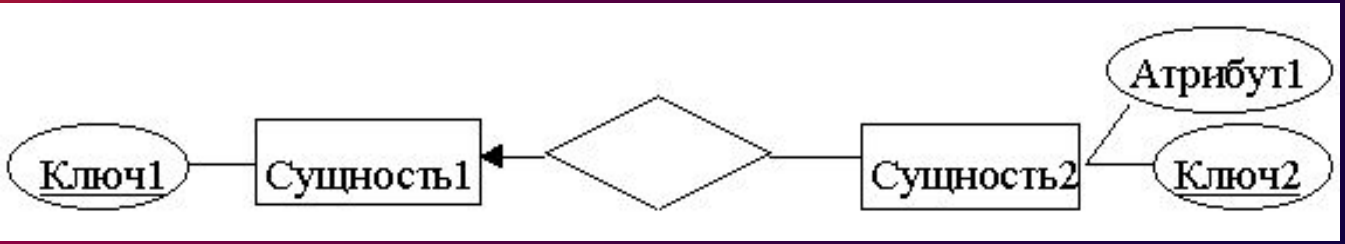


Сущ1Сущ2(Ключ1, Ключ2, Атрибут1).

Преобразование связей

Связь 1:М

1. Создается новая таблица, содержащая ключевые поля каждой сущности, участвующей в связи. В названии обычно отражают, какие именно сущности связываются, или называют новую таблицу именем связи.
Ключом будет ключ второй сущности.
Сущ1Сущ2(Ключ1, Ключ2).

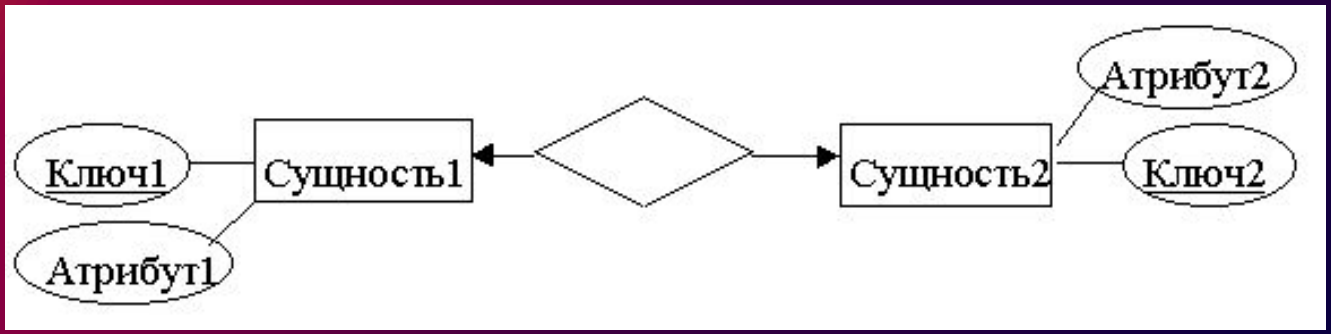


2. В таблицу дочерней сущности добавляют ключевые поля родительской сущности (в ключ дочерней сущности они входить не будут). Ключевые поля родительской сущности представляют собой внешний для дочерней сущности.
Сущность2(Ключ2, Атрибут1, Ключ1).

Преобразование связей

Связь 1:1

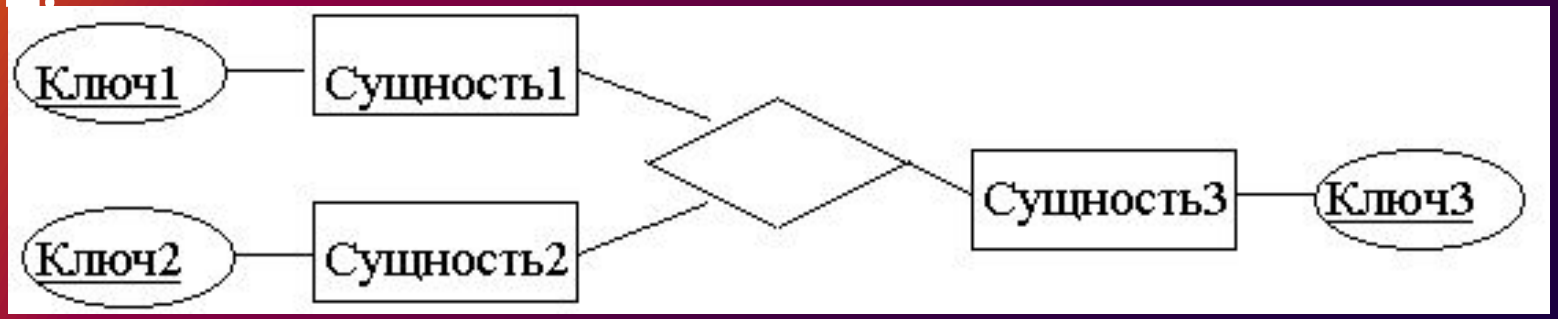
1. Создается новая таблица, содержащая ключевые поля каждой сущности, участвующей в связи. В названии отражают, какие именно сущности связываются, или называют новую таблицу именем связи.
Ключом будет ключ любой сущности.
Сущ1Сущ2(Ключ1, Ключ2) или Сущ1Сущ2(Ключ1, Ключ2)



2. Новая таблица не создается, а в таблицу одной из сущностей (будем считать ее дочерней) добавляют ключевые поля другой сущности (будем считать ее родительской).
Сущность1(Ключ1, Атрибут1, Ключ2),
или Сущность2(Ключ2, Атрибут2, Ключ1)

Преобразование связей

Для связей с арностью более 2 обычно применяется тот же способ, что и для бинарной связи M:M - создается новая таблица, содержащая ключевые поля всех связанных таблиц.



Сущ1Сущ2Сущ3(Ключ1, Ключ2, Ключ3).