

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

КРАСНОЯРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ

*Методические указания для студентов 1-го курса
экономического факультета*

Красноярск 2004

Составитель: Е.Д. Карпова

Реляционная модель данных: Методические указания / Краснояр. гос. ун-т; Сост. Е.Д. Карпова. – Красноярск, 2004. – 45 с.

В методических указаниях даны основные понятия реляционной модели данных, рассмотрены различные подходы к построению таких моделей, описана концептуальная модель предметной области и переход от концептуальной модели к реляционной. Методические указания снабжены многочисленными примерами, содержат лабораторный практикум и контрольные работы.

Печатается по решению редакционно-издательского
совета Красноярского университета

© Красноярский
государственный
университет

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И СУБД

История развития

Развитие вычислительной техники всегда обуславливалось двумя основными направлениями ее использования. Первое направление – применение ЭВМ для выполнения сложных численных расчетов. Прогресс в этом направлении способствовал интенсификации методов численного решения сложных математических и инженерных задач, развитию алгоритмических языков программирования, становлению обратной связи с разработчиками новых архитектур ЭВМ.

Второе направление связано с использованием средств вычислительной техники в автоматических или автоматизированных информационных системах. Миф (или реальность?) о хаосе в системах регистрации документов восходит еще к первым клинописным табличкам и папирусным свиткам. И только в конце XX века появление мощных информационных систем позволило совершить революцию в делопроизводстве и бухгалтерии.

Понятие информационной система (ИС) обычно подразумевает программный комплекс, функции которого состоят в поддержке надежного хранения информации в памяти компьютера, выполнении специфических для данного приложения преобразований информации и/или вычислений, предоставлении пользователям удобного и легко осваиваемого интерфейса. Причем объемы данных в ИС достаточно велики, сами данные имеют крайне сложную структуру, а получение информации зачастую происходит в реальном времени. Классическими примерами ИС являются банковские системы, системы резервирования авиационных или железнодорожных билетов, мест в гостиницах и т.д. Ядром ИС является, как правило, система управления базами данных (СУБД).

Технологическими предпосылками появления СУБД являются, во-первых, развитие элементной базы хранения данных – сравнительно надежной, быстрой и дешевой внешней памяти (ВП), во-вторых, наличие аппаратно-логической поддержки структуризации данных – файловых систем и систем управления файлами (СУФ). Ранние СУБД опирались на СУФ операционной системы (ОС), в результате чего логическая модель данных сильно зависела от физической, существуя как надстройка над СУФ в виде библиотеки программ.

Поскольку СУФ не могла учитывать сложную структуру данных, их внутреннюю согласованность, то неизбежно возникали проблемы избыточности хранения, слабого контроля и недостаточно гибкого управления данными. Кроме того, такие ИС требовали больших затрат труда программиста как при создании ИС, так и при ее администрировании. Все это привело к

тому, что появились специальные программные комплексы, во многом взявшие на себя функции ОС по эффективному управлению данными.

Основные функции современной СУБД

⇒ **Непосредственное управление данными во внешней памяти.** Обеспечение необходимых структур внешней памяти для хранения данных, непосредственно входящих в БД, и данных для служебных целей (индексы и пр.). При этом могут использоваться возможности существующих файловых систем или ОС подменяется вплоть до уровня устройств внешней памяти.

⇒ **Управление буферами оперативной памяти.** Способом увеличения скорости доступа к данным является буферизация данных в оперативной памяти (ОП). Даже если ОС производит общесистемную буферизацию (как в случае, например, ОС UNIX), этого недостаточно для целей СУБД, которая располагает гораздо большей информацией о полезности буферизации БД. Поэтому в развитых СУБД поддерживается собственный набор буферов ОП с собственной дисциплиной замены буферов.

⇒ **Управление транзакциями.** Транзакция – это последовательность операций над БД, рассматриваемая СУБД как единое целое. Либо транзакция успешно выполняется, и СУБД фиксирует (СОММИТ) изменения БД, произведенные этой транзакцией, во внешней памяти, либо ни одно из этих изменений никак не отражается на состоянии БД. Понятие транзакции необходимо для поддержания логической целостности БД. Обеспечение механизма транзакций является важной функцией для однопользовательских СУБД. В многопользовательской среде оно становится жизненно необходимым.

⇒ **Журнализация.** Одним из основных требований к СУБД является надежность хранения данных во внешней памяти, то есть СУБД должна уметь восстанавливать последнее согласованное состояние БД после любого аппаратного или программного сбоя. Понятно, что для восстановления БД нужно располагать некоторой дополнительной (избыточной) информацией. Причем та часть данных, которая используется для восстановления, должна храниться особо надежно. Наиболее распространенным методом поддержания такой избыточной информации является ведение журнала изменений БД. Во всех случаях придерживаются стратегии "упреждающей" записи в журнал (так называемого протокола Write Ahead Log - WAL).

⇒ **Поддержка языков БД.** Для работы с базами данных используются специальные языки. В ранних СУБД существовало несколько специализированных по своим функциям языков. Чаще всего выделялись два языка - *язык определения схемы БД (SDL – Schema Definition Language)* и *язык манипулирования данными (DML – Data Manipulation Language)*. В современных СУБД обычно поддерживается единый интегрированный язык, содержащий все необходимые средства для работы с БД. Стандартным языком наиболее распространенным в настоящее время реляционных СУБД является язык SQL

(Structured Query Language). Язык SQL сочетает средства SDL и DML, т.е. позволяет определять схему реляционной БД и манипулировать данными. Кроме того, основное администрирование и авторизация доступа к объектам БД производится также на основе специального набора операторов SQL.

Модели данных

Основополагающей в концепции реляционных БД является категория **модель данных**.

Под **данными** обычно понимают набор конкретных значений, параметров, которые характеризуют объект, явление, условие, событие и т.д. Данные хранятся, накапливаются и обрабатываются. **Модель данных** – это концептуальный способ структурирования данных. **Модель** – представление реальности, отображающее только избранные детали, это некоторая абстракция, которая, будучи приложима к конкретным *данным*, позволяет трактовать их уже как *информацию*.

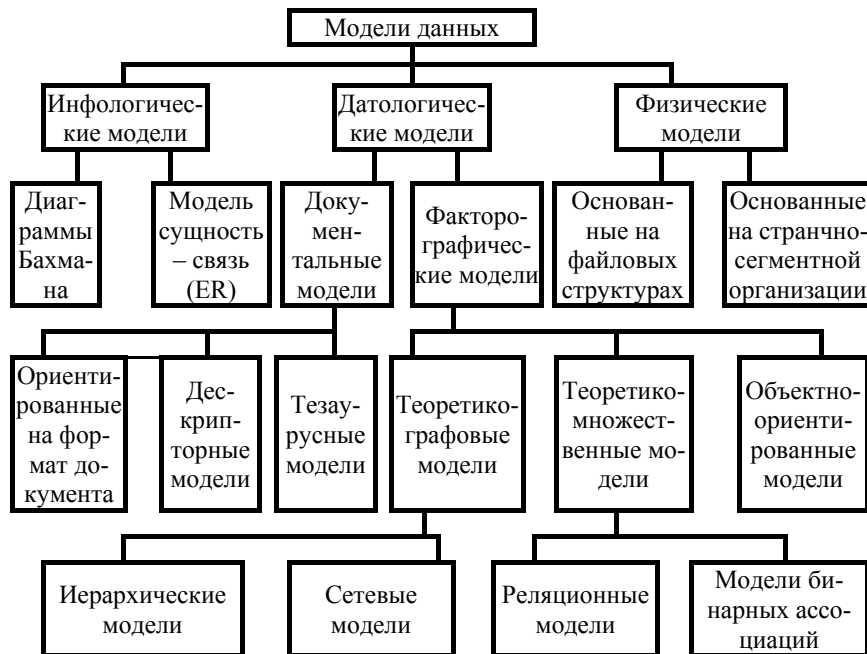


Рис. 1. Иерархия моделей данных

ANSI (American National Standards Institute) предлагает выделять три уровня архитектуры СУБД: **внешняя модель – концептуальная модель – БД (физическая модель)**

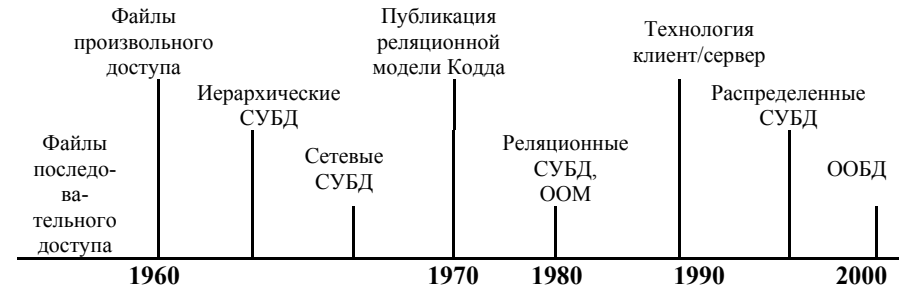


Рис. 2

В соответствии с этой классификацией обычно рассматривают следующие уровни моделей данных – физические модели (*физическое проектирование*); концептуальные (*логическое проектирование*). Внешние модели обычно являются подсхемами концептуального уровня и описываются в тех же терминах и категориях. Однако при проектировании БД рассматривается еще один уровень моделей – *инфологический* или *семантический*.

На рис.1 приведена общая иерархия моделей данных (см. [1]). Рисунок 2 отражает временные рамки развития СУБД.

Ранние подходы к организации БД

Ранними (дореляционными) СУБД принято считать иерархические БД, сетевые и БД, основанные на инвертированных списках. Независимо от методов организации данных, у ранних СУБД можно выделить общие черты.

1. Эти системы активно использовались в течение многих лет, *дольше, чем используется какая-либо из реляционных СУБД*. На самом деле некоторые из ранних систем используются даже в наше время.

2. *Все ранние системы не основывались на каких-либо абстрактных моделях*. Понятие модели данных фактически вошло в обиход специалистов в области БД только вместе с реляционным подходом. Абстрактные представления ранних систем появились позже на основе анализа и выявления общих признаков у разных конкретных систем.

3. *В ранних системах доступ к БД производился на уровне записей*. Пользователи этих систем осуществляли явную навигацию в БД, используя языки программирования, расширенные функциями СУБД. Интерактивный доступ к БД поддерживался только путем создания соответствующих прикладных программ с собственным интерфейсом.

4. Навигационная природа ранних систем и доступ к данным на уровне записей заставляли *пользователя самого* производить *всю оптимизацию доступа к БД*, без какой-либо поддержки системы.

5. После появления реляционных систем *большинство ранних систем было оснащено "реляционными" интерфейсами*. Однако это не сделало их по-настоящему реляционными системами.

Ранние СУБД обладали рядом неоспоримых достоинств:

- развитие средства управления данными во внешней памяти (ВП) на низком уровне;
- возможность построения вручную эффективных прикладных систем;
- возможность экономии памяти за счет разделения подобъектов (в сетевых системах).

Недостатки же этих систем вполне очевидны:

- слишком сложно пользоваться;
- всегда необходимы знания о физической организации;
- прикладные системы зависят от физической организации;
- логика системы перегружена деталями организации доступа к БД.

ПРИНЦИПЫ КОНЦЕПТУАЛЬНОГО ПРОЕКТИРОВАНИЯ

Сейчас считается общепринятым на первом этапе проектирования строить некоторую базовую модель, именуемую *концептуальной (инфологической)*. На нее возлагается ответственность за сохранение семантики предметной области, что не может обеспечить пока никакая модель логического уровня. Ни теоретико-графовые модели, ни реляционная модель не отражают семантику реального мира. Наиболее близко к решению этой задачи подошли объектно-ориентированные модели, что легко просматривается в подходе, который будет изложен в этом параграфе. Однако до коммерческой программной реализации такого подхода напрямую пока далеко.

Проблема представления семантики являлась предметом исследования с середины 1970-х годов. Был предложен ряд моделей, из которых проверку временем выдержала, пожалуй, одна – модель “сущность – связь” (“Entity Relationship” - ER), предложенная Ченом (Chen) в 1976 году. С некоторыми вариациями (например, сущность называют объектным множеством, а связь – отношением) эта модель излагается в большинстве работ, посвященных проектированию БД. Модель обладает двумя неоспоримыми преимуществами. Во-первых, она наглядна и вполне понятна широкому кругу лиц, заинтересованных в разработке ИС, а не только узким специалистам по БД. Во-вторых, существуют простые алгоритмы преобразования ER-модели в основные логические модели (сетевую, иерархическую, реляционную). Существуют даже программные продукты автоматического преобразования ER-модели в реляционную (например, POWER DESIGNER).

Базовые понятия модели “Сущность – связь”

Главными элементами концептуальной модели данных являются *сущности (объекты)* и *связи (отношения)*. Сущности часто представляются в виде существительных, а связи – в виде глаголов.

С помощью сущности моделируется класс однотипных объектов, которые пользователи считают важными в моделируемой части реальности.

DEF. Сущность – множество вещей одного типа, имеющее уникальное имя в пределах моделируемой системы.

DEF. Атрибуты сущности – набор характеристик, определяющих свойства данного представителя класса.

В системе существует множество *экземпляров сущности*, отличающихся друг от друга конкретными значениями атрибутов.

DEF. Ключ сущности – набор атрибутов, однозначно идентифицирующий конкретный экземпляр сущности.

Рисунок 3 иллюстрирует приведенные понятия.

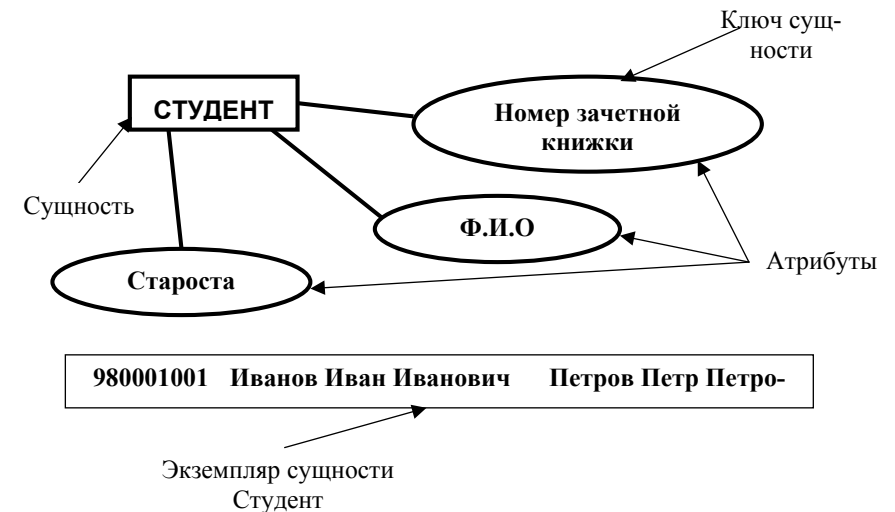


Рис. 3

Некоторые объектные множества содержатся внутри других объектных множеств. *Конкретизация* – это объектное множество, являющееся подмножеством другого объектного множества. *Обобщение* – это объектное множество, являющееся надмножеством другого объектного множества. Проведем аналогию с объектно-ориентированными языками программирования (ООЯП), см. рис. 4.

DEF. Сущность может быть представлена в виде нескольких **подтипов** (конкретизаций), каждый из которых может иметь общие атрибуты, которые определяются однажды на верхнем уровне и наследуются на нижний.

DEF. Сущность, на основе которой строятся подтипы, называется **супертипом** (обобщением). Все подтипы одной сущности рассматриваются как взаимоисключающие (непересекающиеся подмножества).

Отношение связывает два или более объектных множества. Другими словами, между сущностями могут быть установлены связи.

DEF. **Связь** – бинарная ассоциация, показывающая, каким образом сущности соотносятся или взаимодействуют между собой.

Связь может быть установлена между двумя различными сущностями или между сущностью и ей же самой (рекурсивная связь). Между двумя сущностями может быть установлено сколько угодно связей, несущих различную смысловую нагрузку.

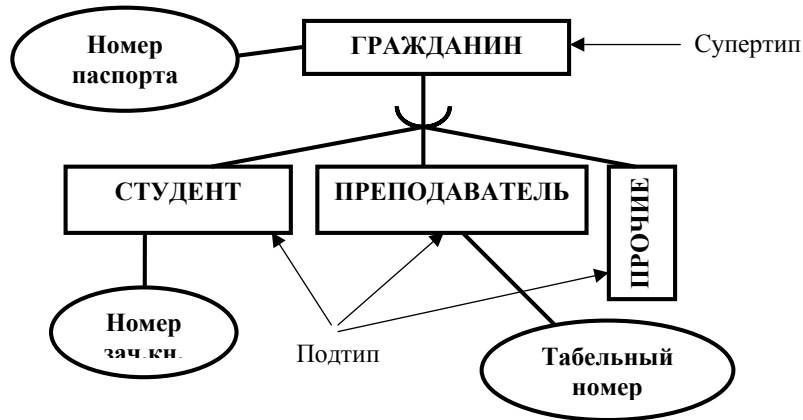


Рис. 4

Связь описывается двумя характеристиками – *степенью связи* (мощностью) и *классом принадлежности*. Обе характеристики определяются отдельно для каждого конца связи (т.е. для конгломерата сущность – связь).

DEF. **Степень связи (мощность)** – *максимальное количество* экземпляров одной сущности, связанных с *одним* экземпляром другой сущности. Всего можно выделить три типа связей: *один-к-одному* (1:1), *один-ко-многим* (1:*), *многие-ко-многим* (*:*)).

DEF. **Класс принадлежности** – показатель *обязательности* участия каждого экземпляра сущности в связи.

Обязательный класс принадлежности связи (на диаграмме перечеркивается линия связи около соответствующей сущности, см. рис. 5) показывает,

что *каждый* экземпляр сущности **ДОЛЖЕН** участвовать в связи. **Необязательный класс принадлежности** (на диаграмме обозначается кружком на линии связи около соответствующей сущности, см. рис. 5) допускает наличие экземпляров сущности, *не участвующих* в связи.

Иногда, кроме мощности связи указывают также нижнюю допустимую границу степени связи, т.е. минимальное количество экземпляров сущности, допустимое к участию в связи. Например, Студент обязан слушать не менее 5 курсов лекций. В этом контексте необязательный класс принадлежности следует понимать как связь с нижней границей равной *нулю*, а обязательный класс принадлежности, если не указано специально, имеет нижнюю границу, равную *единице*.

Следует отметить еще одну особенность связи между сущностями. Дело в том, что связь можно трактовать как *составное* объектное множество (составную сущность). Более того, часто эта составная сущность имеет атрибуты, то есть свойства, не принадлежащие ни одной из сущностей, участвующих в связи, но характеризующих саму связь.

Рисунок 5 иллюстрирует отображение обсуждаемых понятий на ER-диаграммах. Дадим следующие комментарии.

Сущность СТУДЕНТ связана с сущностью ДИПЛОМНЫЙ_ПРОЕКТ с помощью связи 1:1, то есть каждый Студент может писать *не более одного* Дипломного_Проекта, а каждый Дипломный_Проект пишется только *одним* Студентом. При этом Студент *не обязан* выполнять Дипломный_Проект (например, он не является пятикурсником), следовательно, класс принадлежности связи со стороны СТУДЕНТ является *необязательным*, напротив, каждый Дипломный_Проект *обязан* выполняться каким-либо Студентом (класс принадлежности связи со стороны ДИПЛОМНЫЙ_ПРОЕКТ – *обязательный*).

Сущности СТУДЕНТ и ПРЕПОДАВАТЕЛЬ связаны двумя связями, несущими разную смысловую нагрузку. Первая связь, касающаяся лекций, имеет тип *: * и *обязательный* класс принадлежности с обоих концов связи. Таким образом, каждый Преподаватель *обязан* читать *не менее одного* курса лекций, а каждый Студент *обязан* слушать, *по крайней мере, один* курс лекций. Вторая связь, описывающая руководство преддипломной практикой, имеет степень “многие” со стороны СТУДЕНТА (обратите внимание, последнее означает, что каждый Преподаватель может *руководить несколькими* Студентами). Степень связи со стороны Преподаватель – 1, то есть Студент имеет *только одного* научного руководителя. В отличие от предыдущего примера рассматриваются только студенты-дипломники, что приводит к *обязательному* классу принадлежности со стороны СТУДЕНТ, а вот Преподаватель *вовсе не обязан* руководить дипломниками, поэтому класс принадлежности связи со стороны ПРЕПОДАВАТЕЛЬ – *не обязательный*.

Наконец, последняя схема на рис. 5 иллюстрирует ситуацию, в которой существуют атрибуты, являющиеся не свойствами какой-либо отдельной сущности, а суть атрибутами связи. Аудитория, День_Недели, Время не при- сущи ни Преподавателю, ни Студенту, но они описывают процесс чтения лекций (т.е. связь сущностей).

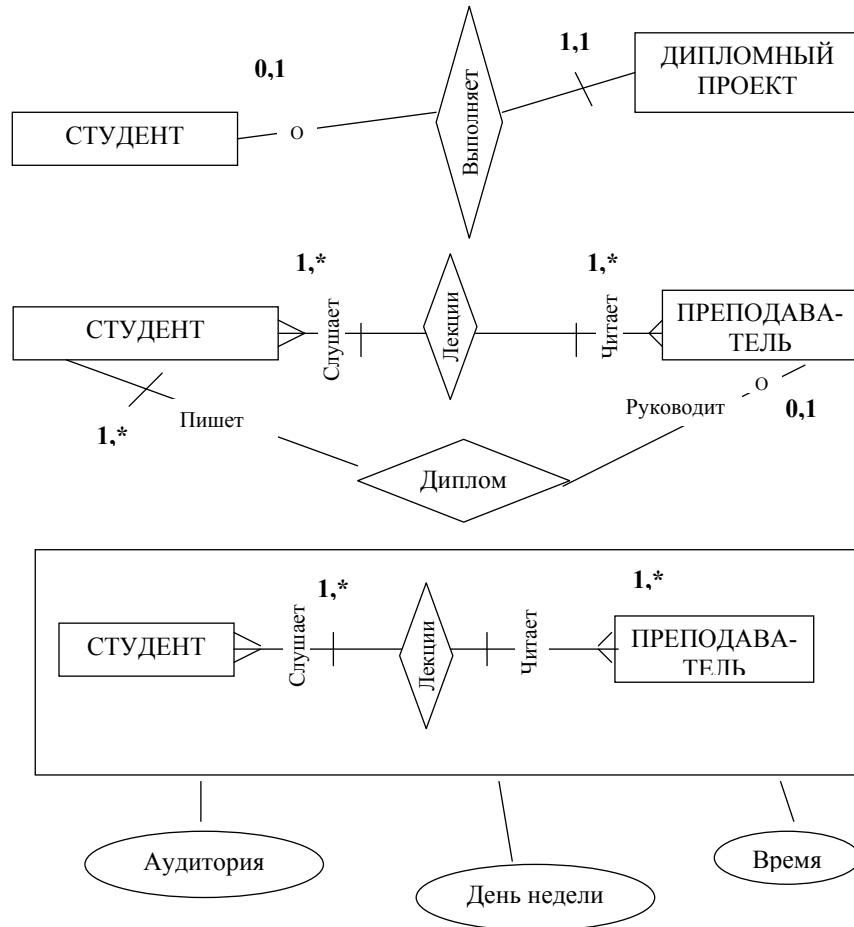


Рис. 5

Во всех связях, которые мы рассматривали до этого момента, участвовало две сущности. Такие связи называются *бинарными*. Однако отношение может связывать три и более сущностей. Эти связи *высокого порядка* называются *n – арными* отношениями, где n обозначает число сущностей. Для уп-

рощения терминологии 3 – арные и 4 – арные отношения называют *трех-сторонними* и *четырёхсторонними*.

РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ

Базовые понятия реляционной модели данных

Основными понятиями реляционных баз данных являются *тип данных*, *домен*, *атрибут*, *кортеж*, *первичный ключ* и *отношение*.

Тип данных

Понятие *тип данных* в реляционной модели данных полностью адекватно понятию типа данных в языках программирования. Обычно в современных реляционных БД допускается хранение символьных, числовых данных, битовых строк, специализированных числовых данных (таких, как “деньги”), а также специальных “темпоральных” данных (дата, время, временной интервал).

Домен

Понятие *домена* более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в языках программирования. В самом общем виде *домен* определяется заданием *базового типа* данных, к которому относятся элементы домена, и *произвольного логического выражения*, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат “истина”, то элемент данных является элементом домена:

Домен = Базовый тип + Правило.

Наиболее правильной интуитивной трактовкой понятия домена является понимание его как допустимого потенциального множества значений данного типа.

Следует отметить также *семантическую* нагрузку понятия домена: данные считаются *сравнимыми (мета-сравнимыми)* только в том случае, когда они относятся к одному домену, а не к одному базовому типу.

Отношение

Атрибут – свойство объекта предметной области. Атрибут характеризуется именем и значением, которое должно принадлежать некоторому домену. Каждый экземпляр объекта в каждый момент времени однозначно характеризуется набором конкретных значений атрибутов.

Схема отношения – это именованное множество пар {**имя атрибута, имя домена**}. **Степень** или “арность” схемы отношения – мощность этого множества, т.е. количество атрибутов.

Схема базы данных – это набор именованных схем отношений.

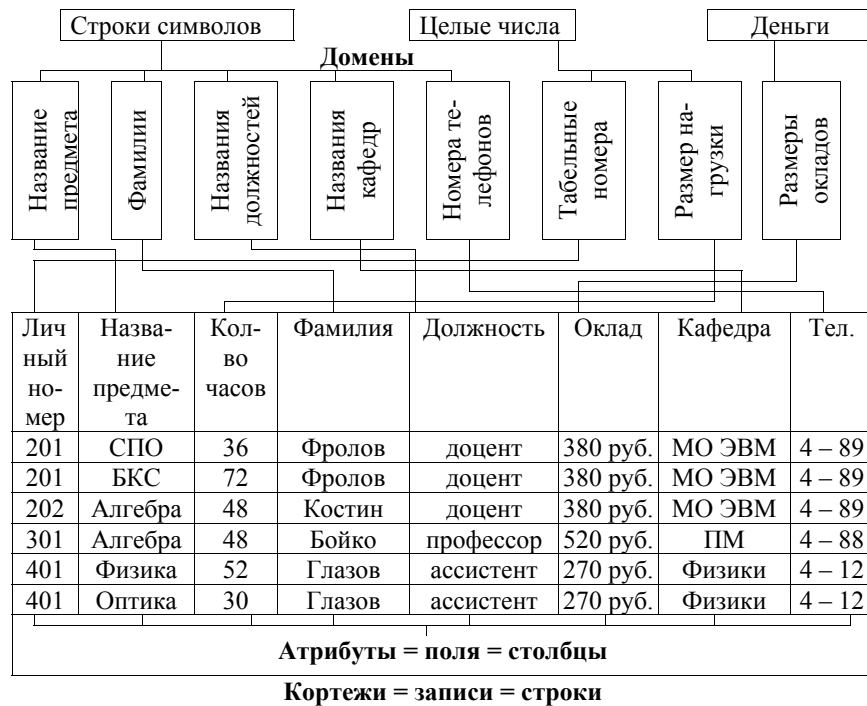
Кортеж, соответствующий данной схеме отношения, – это множество пар {**имя атрибута, значение**}, которое содержит одно вхождение каждого

имени атрибута, принадлежащего схеме отношения. “Значение” является допустимым значением домена данного атрибута.

Отношение – это множество кортежей, соответствующих одной схеме отношения.

На самом деле, понятие схемы отношения ближе всего к понятию структурного типа данных в языках программирования. Было бы вполне логично разрешать отдельно определять схему отношения, а затем одно или несколько отношений с данной схемой. Однако в реляционных базах данных это не принято. *Имя схемы отношения в таких базах данных всегда совпадает с именем соответствующего отношения-экземпляра.*

Типы данных



Отношение ПРЕПОДАВАТЕЛЬ-ДИСЦИПЛИНА = Плоская таблица

Рис. 6

В классических реляционных базах данных после определения схемы базы данных изменяются только отношения-экземпляры. В них могут появляться новые и удаляться или модифицироваться существующие кортежи.

Однако во многих реализациях допускается и изменение схемы базы данных: определение новых и изменение существующих схем отношения. Это принято называть *эволюцией схемы базы данных*.

Схемы двух отношений называют *эквивалентными*, если они имеют одинаковую степень и возможно такое упорядочивание имен атрибутов в схеме, что на одинаковых местах будут находиться тета-сравнимые атрибуты.

Удобным представлением отношения является таблица, заголовком которой является схема отношения, а строками – кортежи отношения-экземпляра; в этом случае имена атрибутов именуют столбцы этой таблицы (см. рис. 6).

Реляционная база данных – это набор отношений, имена которых совпадают с именами схем отношений в схеме БД.

Свойства отношений

Дадим теперь теоретико-множественное описание понятия отношения.

DEF. *N-арным отношением* R называют *подмножество* декартова произведения $D_1 \times D_2 \times \dots \times D_N$ множеств D_1, D_2, \dots, D_N ($N \geq 1$), необязательно различных. Исходные множества D_1, D_2, \dots, D_N называют *доменами*.

Кратко обсудим вытекающие из этого определения фундаментальные свойства отношений.

⇒ Отсутствие кортежей-дубликатов следует из определения отношения как множества кортежей. Из этого свойства вытекает наличие у каждого отношения так называемого **первичного ключа** – *минимального* набора атрибутов, значения которых *однозначно* определяют кортеж отношения.

Для каждого отношения, по крайней мере, полный набор его атрибутов уникален. Однако при формальном определении первичного ключа требуется “минимальность”, 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. Следует отметить, что во многих практических реализациях РСУБД допускается нарушение свойства уникальности кортежей для промежуточных отношений, порождаемых неявно при выполнении запросов. Такие отношения являются не множествами, а мультимножествами.

⇒ Отсутствие упорядоченности кортежей также является следствием определения отношения-экземпляра как множества кортежей. 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.

⇒ **Отсутствие упорядоченности атрибутов.** Для ссылки на значение атрибута в кортеже отношения всегда используется имя атрибута.

⇒ **Атомарность значений атрибутов** означает, что атрибут не может быть отношением. Это свойство следует из определения домена как потенциального множества значений простого типа данных.

DEF. Отношение находится в *первой нормальной форме* (1НФ), если все его атрибуты атомарны.

Отношение, находящееся в 1НФ, также называют *универсальным* или *нормализованным* отношением. Пример ненормализованного отношения приведен в табл. 1.

Таблица 1

Кафедра	Тел.	Личный номер	Фамилия	Должность	Оклад	Название предмета	Кол-во часов
МО ЭВМ	4 – 89	201	Фролов	доцент	380 руб.	СПО БКС	36 72
		202	Костин	доцент	380 руб.	Алгебра	48
ПМ	4 – 88	301	Бойко	профессор	520 руб.	Алгебра	48
Физики	4 – 12	401	Глазов	ассистент	270 руб.	Физика	52
						Оптика	30

Нормализованные отношения составляют основу классического реляционного подхода к организации баз данных. Они обладают некоторыми ограничениями (не любую информацию удобно представлять в виде плоских таблиц), но существенно упрощают манипулирование данными.

Реляционная модель данных: три составляющие

Модель данных описывает некоторый набор родовых понятий и признаков, которыми должны обладать все конкретные СУБД и управляемые ими базы данных, если они основываются на этой модели. Наличие модели данных позволяет сравнивать конкретные реализации, используя один общий язык.

Термин “модель данных” был введен в обиход применительно к реляционным системам и наиболее эффективно используется именно в этом контексте, хотя можно говорить об иерархической, сетевой, и даже концептуальной моделях данных.

Наиболее распространенная трактовка реляционной модели данных, по-видимому, принадлежит Дейту. Согласно Дейту, реляционная модель состоит из трех частей, описывающих разные аспекты реляционного подхода: структурной части, манипуляционной части и целостной части.

⇒ **В структурной части модели** фиксируется, что единственной структурой данных, используемой в реляционных БД, является нормализованное n -арное отношение.

⇒ **В манипуляционной части модели** вводятся два фундаментальных механизма манипулирования реляционными БД - реляционная алгебра и реляционное исчисление. Первый механизм базируется на классической теории множеств, а второй - на классическом логическом аппарате исчисления предикатов первого порядка. Эта часть реляционной модели вводит *меру реляционности* любого конкретного языка РБД: язык называется *реляционным*, если он обладает не меньшей мощностью, чем реляционная алгебра или реляционное исчисление.

⇒ **В целостной части реляционной модели данных** фиксируются два базовых требования целостности, которые должны поддерживаться в любой реляционной СУБД.

Первое – требование **целостности сущностей**: *любое отношение должно обладать первичным ключом.*

Второе – **требование целостности по ссылкам**. В отличие от теоретико-графовых моделей в реляционной модели связи между отношениями поддерживаются неявно. В каждой связи одно отношение выступает как основное, а другое в роли подчиненного, т.е. один кортеж основного отношения может быть связан с несколькими кортежами подчиненного отношения. Таким образом, основными в реляционных БД являются связи типа “один-ко-многим”, связи типа “многие-ко-многим” моделируются с помощью дополнительной таблицы связи (см. тему “Преобразование концептуальной модели в реляционную”). Для поддержки связей в схему подчиненного отношения добавляют первичный ключ основного отношения. Атрибуты такого рода называются *внешним ключом*.

Требование целостности по ссылкам состоит в том, что *для каждого значения внешнего ключа подчиненного отношения в основном отношении должен найтись кортеж с таким же значением первичного ключа, иначе значение внешнего ключа подчиненного отношения должно быть неопределенным* (то есть ни на что не указывать).

Ограничения целостности сущности и по ссылкам должны поддерживаться СУБД.

Для соблюдения целостности сущности достаточно гарантировать отсутствие в любом отношении кортежей с одним и тем же значением первичного ключа.

С целостностью по ссылкам дела обстоят несколько более сложно.

Понятно, что при обновлении ссылающегося отношения (вставке новых кортежей или модификации значения внешнего ключа в существующих кортежах) достаточно следить за тем, чтобы не появлялись некорректные

значения внешнего ключа. Но как быть при удалении кортежа из отношения, на которое ведет ссылка?

Здесь существуют три подхода, каждый из которых поддерживает целостность по ссылкам.

1. Запрещается производить удаление кортежа из основного отношения, если на него существуют ссылки.
2. При удалении кортежа, на который имеются ссылки, во всех ссылающихся кортежах подчиненного отношения значение внешнего ключа автоматически становится неопределенным.
3. Каскадное удаление состоит в том, что при удалении кортежа из основного отношения, автоматически удаляются все ссылающиеся кортежи из подчиненного отношения.

В развитых реляционных СУБД обычно можно выбрать способ поддержания целостности по ссылкам для каждой отдельной ситуации определения внешнего ключа. Конечно, для принятия такого решения необходимо анализировать требования конкретной прикладной области.

ПРОЕКТИРОВАНИЕ РБД С ПОМОЩЬЮ КОНЦЕПЦИИ ФУНКЦИОНАЛЬНЫХ ЗАВИСИМОСТЕЙ

Понятие функциональной зависимости

Отправной точкой рассматриваемого в этом параграфе подхода к проектированию РБД является отношение, находящееся в *первой нормальной форме*.

DEF. Отношение находится в *первой нормальной форме (1НФ)*, если каждый его элемент имеет и всегда будет иметь атомарное значение.

Возможным ключом отношения называется минимальный набор атрибутов, однозначно определяющий кортеж. На первом этапе проектирования из множества возможных ключей следует выбрать один, который будет являться *первичным ключом БД (ПК)*. Отметим, что на практике следует провести нормализацию 1НФ для каждого первичного ключа из множества возможных ключей, а затем выбрать наиболее подходящую из полученных моделей.

DEF. Атрибуты, не входящие ни в один из возможных первичных ключей, называются *неключевыми*.


БД, находящаяся в 1НФ, имеет ряд недостатков, называемых аномалиями хранения, изменения и удаления данных (объясните, каких?). Процесс разбиения отношения с целью уменьшения вероятности возникновения аномалий называется *декомпозицией без потерь*. Ключевой для декомпозиции является концепция *функциональной зависимости (ФЗ)*.

DEF. Говорят, что атрибут В *функционально зависит* от атрибута А, если для каждого значения a атрибута А в любой момент времени существует ровно одно связанное с ним значение b атрибута В.

Атрибуты А и В могут быть составными (то есть состоять из нескольких атрибутов). ФЗ определяются исходя из базовых свойств самих атрибутов и семантики предметной области.

Ниже представлены два возможных способа записи того, что атрибут В ФЗ от А:

$A \rightarrow B$ – математическая форма записи;

 – диаграмма или графическая форма записи.

Рассмотрим в качестве примера (см. табл. 2) отношение ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ, содержащее сведения о преподавателях и читаемых ими курсах.

Если отношение находится в 1НФ, то все неключевые атрибуты функционально зависят от ключа. Но степень зависимости может быть различной.

Таблица 2

Личный номер	Название предмета	Кол-во часов	Фамилия	Должность	Оклад	Кафедра	Телефон
201	СПО	36	Фролов	доцент	380 руб.	МО ЭВМ	4 – 89
201	БКС	72	Фролов	доцент	380 руб.	МО ЭВМ	4 – 89
202	Алгебра	48	Костин	доцент	380 руб.	МО ЭВМ	4 – 89
301	Алгебра	48	Бойко	профессор	520 руб.	ПМ	4 – 88
401	Физика	52	Глазов	ассистент	270 руб.	Физики	4 – 12
401	Оптика	30	Глазов	ассистент	270 руб.	Физики	4 – 12

В нашем примере все имеющиеся ФЗ представлены на рисунке 7:

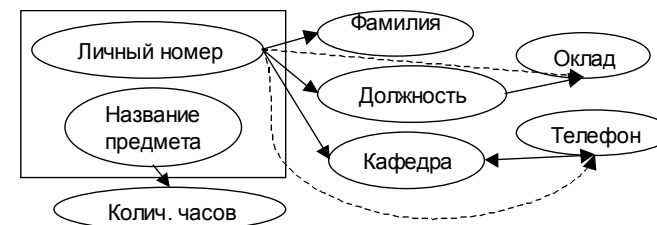


Рис. 7

Название предмета → Кол-во часов

Личный номер → Фамилия

Личный номер → Должность

Личный номер → Кафедра

Личный номер → Оклад

Личный номер → Телефон

Должность → Оклад

Кафедра ↔ Телефон

Отношение ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ имеет только один первичный ключ. <Личный номер, Название предмета>. Этот вывод получен путем нахождения минимального набора значений атрибутов, которые, если они известны, определяют значения всех других атрибутов кортежа.

DEF. Если $A \rightarrow B$ и $B \rightarrow A$, то говорят о *взаимно-однозначной функциональной зависимости*.

Пример: Кафедра ↔ Телефон.

DEF. Если неключевой атрибут зависит только от части ключа, то говорят о *функционально неполной зависимости*.

Пример: Название предмета → Кол-во часов.

Если неключевой атрибут зависит от всего составного ключа, то говорят о его *полной функциональной зависимости*. В нашем примере нет атрибутов, находящихся в полной функциональной зависимости от составного ключа.

DEF. Если для атрибутов A, B, C выполняются условия $A \rightarrow B$ и $B \rightarrow C$, то говорят, что C зависит от A *транзитивно*.

Пример: Личный номер → Должность → Оклад.

DEF. С точки зрения 1НФ, атрибут B *транзитивно зависит* от атрибута A, если A не входит в ПК.

DEF. Если B функционально зависит от A и B не зависит функционально от любого подмножества A, то говорят, что A представляет собой *детерминант* B.

Детерминантами в отношении ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ являются левые части всех ФЗ в отношении, т.е. <Название предмета>, <Должность>, <Кафедра> и <Личный номер>. Взаимно-однозначные зависимости дают два детерминанта.

Вторая и третья НФ. Алгоритм декомпозиции

DEF. Отношение находится во *Второй Нормальной Форме* (2НФ), если оно находится в 1НФ и из него исключены все функционально неполные зависимости.

DEF. БД находится во *Второй Нормальной Форме* (2НФ), если все ее отношения находятся во 2НФ.

DEF. Отношение находится в *Третьей Нормальной Форме* (3НФ), если оно находится во 2НФ и из него исключены все транзитивные функциональные зависимости.

DEF. БД находится в Третьей *Нормальной Форме* (3НФ), если все ее отношения находятся в 3НФ.

Основным приемом нормализации отношения является декомпозиция без потерь. Опишем кратко общий **алгоритм декомпозиции**. Для краткости *аномальной* ФЗ будем называть неполную или транзитивную зависимость.

1. Для каждого детерминанта аномальной функциональной зависимости выписывается свое отношение с этим детерминантом в качестве ПК отношения.
2. Выписывается отношение связи, в которое включаются все атрибуты, не вошедшие в отношения из пункта 1, плюс к ним все ПК отношений из пункта 1 (поля связи).

Следует помнить, что нормализация 1НФ проводится последовательно 1НФ → 2НФ → 3НФ, т.е. сначала проводится декомпозиция 1НФ по всем функционально неполным зависимостям, а затем, полученные отношения разбиваются с целью избавления от транзитивных зависимостей.

Применим этот алгоритм к отношению ПРЕПОДАВАТЕЛЬ_ДИСЦИПЛИНА.

➤ Приведем отношение ко 2НФ, т.е. избавимся от функционально неполных зависимостей:

Название предмета → Кол-во часов

Личный номер → <Фамилия, Должность, Кафедра, Оклад, Телефон>

В результате получим три отношения:

ПРЕПОДАВАТЕЛЬ = < Личный номер, Фамилия, Должность, Оклад, Кафедра, Телефон >;

ПРЕДМЕТ = < Название предмета, Кол-во часов >;

НАГРУЗКА = < Личный номер(FK), Название предмета (FK) >

Знаком FK помечены поля связи, называемые *внешними ключами* (foreign key).

➤ Два последних отношения не содержат транзитивных зависимостей, поэтому находятся в 3НФ. Отношение ПРЕПОДАВАТЕЛЬ содержит три транзитивных зависимости:

Должность → Оклад;

Кафедра → Телефон;

Телефон → Кафедра.

Применим алгоритм декомпозиции для этих ФЗ. Строгое следование алгоритму требует разбиения отношения ПРЕПОДАВАТЕЛЬ на четыре

ТАРИФНАЯ_СЕТКА = < Должность, Оклад >;

ТЕЛ_СПРАВ1 = < Кафедра, Телефон(FK) >;

ТЕЛ_СПРАВ2 = < Телефон, Кафедра(FK) >;

ПРЕПОДАВАТЕЛЬ = < Личный номер, Фамилия, Должность(FK), Кафедра(FK) >.

В последнее отношение вместо атрибута Кафедра можно включить атрибут Телефон. Включение обоих атрибутов в отношение ПРЕПОДАВАТЕЛЬ избыточно. Более того, можно в результате анализа предметной области и/или переговоров с заказчиком БД отказаться от одной стороны взаимно-однозначной зависимости. Тогда соответствующее отношение ТЕЛ_СПРАВ1 или ТЕЛ_СПРАВ2 пропадет. Однако если необходимо сохранить в схеме РБД обе зависимости, то придется оставить оба отношения.

В итоге ЗНФ РБД ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ состоит из шести отношений

ПРЕДМЕТ = < Название предмета, Кол-во часов >;

НАГРУЗКА = < Личный номер(FK), Название предмета(FK) >

ТАРИФНАЯ_СЕТКА = < Должность, Оклад >;

ТЕЛ_СПРАВ1 = < Кафедра, Телефон(FK) >;

ТЕЛ_СПРАВ2 = < Телефон, Кафедра(FK) >;

ПРЕПОДАВАТЕЛЬ = < Личный номер, Фамилия, Должность(FK), Кафедра(FK) >.

Нормальная форма Бойса-Кодда

В области реляционных БД Коддом доказано утверждение о том, что большинство аномалий в БД будет устранено в случае должной декомпозиции каждого отношения в *нормальную форму Бойса – Кодда* (НФБК).

DEF. Отношение находится в **НФБК** тогда и только тогда, когда оно находится в ЗНФ, и каждый его детерминант является так же и возможным первичным ключом.

Отношение ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ не находится в НФБК. Для доказательства сравним возможные ключи и детерминанты (табл. 3).

Однако РБД ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ, состоящая из шести отношений, находится в НФБК, поскольку каждое ее отношение находится в НФБК (см. табл. 4).

Таблица 3

Возможные ключи	Детерминанты
<Название предмета, Личный номер>	<Название предмета>
	<Должность>
	<Кафедра>
	<Личный номер>
	<Телефон>

Таблица 4

Отношение	Возможные ключи	Детерминанты
ПРЕДМЕТ	<Название предмета>	<Название предмета>
ТАРИФНАЯ_СЕТКА	<Должность>	<Должность>
ТЕЛ_СПРАВ1	<Кафедра>	<Кафедра>
ПРЕПОДАВАТЕЛЬ	<Личный номер>	<Личный номер>
ТЕЛ_СПРАВ2	<Телефон>	<Телефон>
НАГРУЗКА	<Название предмета, Личный номер>	<Название предмета, Личный номер>

Может показаться, что любое отношение, находящееся в ЗНФ находится и в НФБК. Однако это не так. Примеры следует искать в отношениях с несколькими возможными первичными ключами.

Рассмотрим процесс сдачи студентом сессии. Структура отношения, отображающего это явление, может иметь вид:

УСПЕВАЕМОСТЬ = < №Зач.Кн., ID_Студента, Дисциплина, Дата, Оценка >.

Под ID_Студента понимается некоторый номер, который присвоен студенту при поступлении, например, в деканате (зачетная книжка может быть утеряна, выдана новая с другим номером, а ID не меняется в течение всего периода учебы).

Структура ФЗ в этом случае изображена на рис. 8.

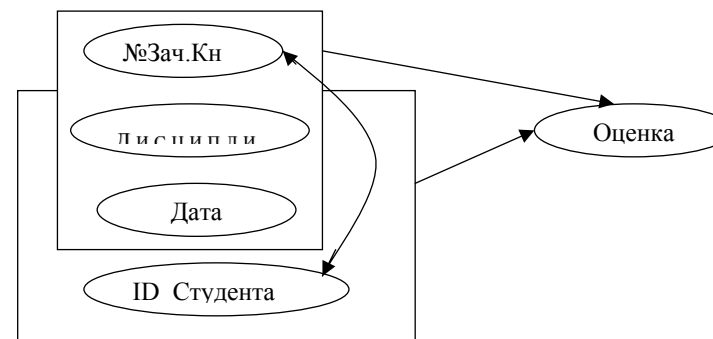


Рис. 8

Множество ВПК: {<№Зач.Кн.>, <ID_Студента>}.

В отношении успеваемость нет ни функционально неполных, ни транзитивных зависимостей. Действительно, взаимно-однозначная ФЗ №Зач.Кн. ↔ ID_Студента не может быть идентифицирована как неполная, поскольку не является зависимостью между неключевыми атрибутами. Таким образом, отношение УСПЕВАЕМОСТЬ находится в ЗНФ. В то же время

оно не находится в НФБК, поскольку множество ВПК и Детерминантов не совпадает (табл. 5).

Таблица 5

Возможные ключи	Детерминанты
<№Зач.Кн., Дисциплина, Дата>	<№Зач.Кн., Дисциплина, Дата>
<ID Студента, Дисциплина, Дата>	<ID Студента, Дисциплина, Дата>
	<ID Студента>
	<№Зач.Кн.>

Чтобы привести отношение УСПЕВАЕМОСТЬ к НФБК следует разделить его на два одним из следующих способов:

СТУДЕНТ1 = <№Зач.Кн., ID_Студента>;

УСПЕВАЕМОСТЬ = <№Зач.Кн., Дисциплина, Дата, Оценка>
или

СТУДЕНТ2= <ID_Студента, №Зач.Кн.>;

УСПЕВАЕМОСТЬ = <ID_Студента, Дисциплина, Дата, Оценка>.

С точки зрения теории нормализации эти схемы равнозначны, поэтому следует делать окончательный выбор из дополнительных рассуждений, например, если при потере зачетных книжек их восстанавливают под другим номером, то предпочтительнее вторая схема.

Отметим, что в любом случае мы отказались от одного из направлений взаимно-однозначного соответствия, иначе нам потребовались бы оба отношения СТУДЕНТ1 и СТУДЕНТ2.

Многозначные зависимости. Четвертая нормальная форма

Хотя в большинстве случаев проектирование РБД заканчивается 3НФ или НФБК, теория рассматривает нормальные формы высших порядков. Их следует иметь в виду, поскольку они связаны с аномалиями, трудно поддающимися диагностике, но, тем не менее, встречающимися на практике. Мы ограничимся только введением 4НФ. Она связана с понятием многозначной зависимости, которая, являясь зависимостью между атрибутами, не является функциональной.

DEF. В отношении $R=\langle A, B, C \rangle$ существует *многозначная зависимость* $R.A \twoheadrightarrow R.B$ в том, и только том случае, если множество значений B , соответствующее паре A и C , зависит только от A и не зависит от C .

Когда мы рассматривали ФЗ, то каждому значению детерминанта соответствовало только одно значение зависимого от него атрибута. При рассмотрении многозначных зависимостей выделяют случаи, когда одному значению некоторого атрибута соответствует устойчивое постоянное множество значений другого атрибута. Причем, если какому-либо значению пары $\langle A, C \rangle$

= $\langle a1, c1 \rangle$ соответствует набор значений атрибута $B = \langle b1, b2, \dots, bN \rangle$, то этот же набор значений соответствует всем возможным парам вида $\langle a1, c2 \rangle$, $\langle a1, c3 \rangle$, ..., $\langle a1, cM \rangle$.

Рассмотрим отношение, моделирующее предстоящую сдачу сессии студентами:

УЧЕБНЫЙ_ПЛАН = <№Зач.Кн., Группа, Дисциплина>.

Перечень дисциплин, которые должен сдавать студент, однозначно определяются не его №Зач.Кн., а номером группы, в которой студент учится, с другой стороны, список студентов, входящих в группу, однозначно определяется ее номером. Таким образом, в отношении УЧЕБНЫЙ_ПЛАН имеется две многозначных зависимости:

Группа \twoheadrightarrow Дисциплина;

Группа \twoheadrightarrow №Зач.Кн.

Если мы будем работать с исходным отношением, то не сможем хранить информацию о новой группе и ее учебном плане, пока туда не зачислен хотя бы один студент. При изменении перечня дисциплин для группы, необходимо внести изменения во все карточки, относящиеся к этой группе. При отчислении студента, мы должны будем удалить все записи, к нему относящиеся и т.д. Таким образом, хотя наше отношение не содержит аномальных ФЗ, аномалии модификации присутствуют, они обусловлены наличием многозначных зависимостей.

В теории РБД доказано, что в отношении $R=\langle A, B, C \rangle$ существует многозначная зависимость $R.A \twoheadrightarrow R.B$ тогда и только тогда, когда существует многозначная зависимость $R.A \twoheadrightarrow R.C$. Избавление от многозначных зависимостей опирается на теорему Фейджина.

Теорема Фейджина. Отношение $R=\langle A, B, C \rangle$ можно спроецировать без потерь в отношения $R1=\langle A, B \rangle$ и $R2=\langle A, C \rangle$ в том, и только том случае, когда существует многозначная зависимость $R.A \twoheadrightarrow R.B$ (что равнозначно существованию двух многозначных зависимостей $R.A \twoheadrightarrow R.B$ и $R.A \twoheadrightarrow R.C$).

DEF. Отношение R находится в 4НФ тогда и только тогда, если в случае существования многозначной зависимости $R.A \twoheadrightarrow R.B$ все остальные атрибуты R функционально зависят от A .

В нашем примере можно произвести декомпозицию отношения УЧЕБНЫЙ_ПЛАН следующим образом:

ГРУППА = <№Зач.Кн., Группа>;

УЧЕБНЫЙ_ПЛАН = <Группа, Дисциплина>.

Избыточные ФЗ. Минимальное покрытие

Рассмотренный алгоритм проектирования БД не свободен от проблемы присутствия избыточных зависимостей.

DEF. Зависимость, не заключающая в себе такой информации, которая не могла бы быть получена на основе других зависимостей из числа использованных при проектировании БД, называется *избыточной ФЗ*.

Поскольку избыточная ФЗ не содержит уникальной информации, она может быть удалена из набора ФЗ без отрицательного воздействия на результаты. Избыточные ФЗ удаляются на начальном этапе проектирования до применения алгоритма декомпозиции.

Транзитивные зависимости. Одним из примеров избыточных ФЗ являются транзитивные. Если $A \rightarrow B$, $B \rightarrow C$ и $A \rightarrow C$ входят в набор ФЗ, следовательно, $A \rightarrow C$ является избыточной и ее использование в процессе проектирования не требуется, ее следует исключить из набора перед началом проектирования.

Рисунок 9 демонстрирует, как может быть упрощен набор ФЗ при помощи исключения транзитивных зависимостей. Используя процедуру декомпозиции, упрощенный набор ФЗ можно привести к нормальной форме Бойса – Кодда (НФБК) $R(\underline{A}, B, C, E)$:

$R1(\underline{C}, E)$; $R2(B, C)$; $R3(\underline{A}, B)$.

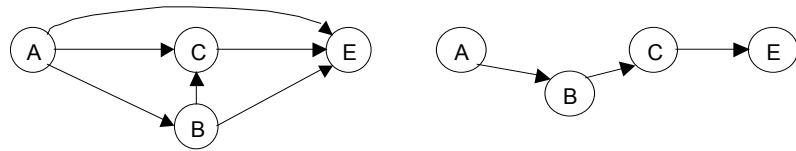


Рис. 9

Добавления. Второй путь возникновения избыточных ФЗ связан с концепцией добавления. Рассмотрим два случая избыточности этого вида (A, B, Z – атрибуты, каждый из которых может быть составным).

⇒ Если $A \rightarrow B$, то $A, Z \rightarrow B$ является *корректной, но избыточной ФЗ*. Иллюстрация дана на рис. 10.

⇒ Если $A \rightarrow B$, то $A, Z \rightarrow B, Z$ является *корректной, но избыточной ФЗ*. Иллюстрация дана на рис. 11.

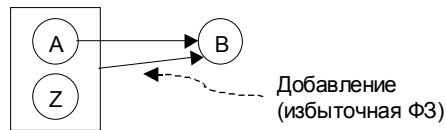


Рис. 10

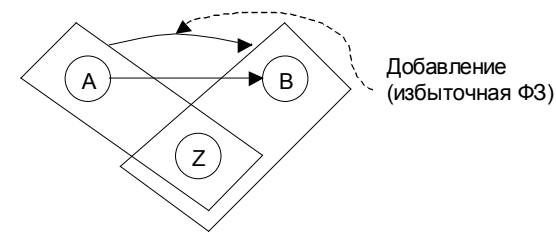


Рис. 11

Правила вывода

Определения транзитивности и концепция добавления касаются трех правил вывода из шести. Эти правила используются для упрощения исходного набора ФЗ. Рассмотрим еще три правила:

1. **Объединение ФЗ:** Если $A \rightarrow B$ и $A \rightarrow C$, то $A \rightarrow B, C$.
2. **Декомпозиция ФЗ:** Если $A \rightarrow B, C$, то $A \rightarrow B$ и $A \rightarrow C$.
3. **Псевдотранзитивность:** Если $A \rightarrow B$ и $B, C \rightarrow Z$, то $A, C \rightarrow Z$.

Приведем пример применения правил вывода. Отметим, что последовательность применения неоднозначна.

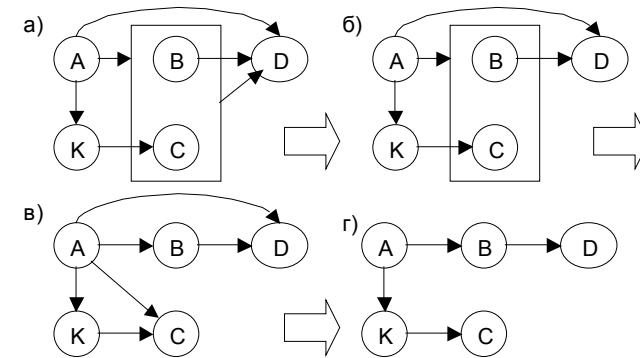


Рис. 12

- а) Исходный набор ФЗ.
- б) $B, C \rightarrow D$ удалена (добавление $B \rightarrow C$)
- в) $A \rightarrow B, C$ замещена $A \rightarrow B$ & $A \rightarrow C$ (декомпозиция)
- г) $A \rightarrow C$ и $A \rightarrow D$ заменяются на $A \rightarrow K$ & $K \rightarrow C$ и $A \rightarrow B$ & $B \rightarrow D$ (транзитивность)

Минимальное покрытие

DEF. Набор избыточных ФЗ, полученных путем удаления всех избыточных ФЗ из исходного набора ФЗ с помощью шести правил вывода, называется *минимальным покрытием*.

Минимальное покрытие не всегда является уникальным, поскольку порядок, в котором осуществляется процедура удаления избыточных ФЗ, может оказать влияние на полученное минимальное покрытие. Избыточные ФЗ следует удалять по одной, каждый раз заново анализируя новый набор ФЗ на предмет присутствия в нем избыточных. Эта процедура завершается, как только не останется ни одной избыточной ФЗ. Оставшийся набор и является минимальным покрытием.

Алгоритм проектирования

1. Построить универсальное отношение для БД.
2. Определить все ФЗ, существующие между атрибутами универсального отношения.
3. Удалить все избыточные ФЗ из исходного набора ФЗ с целью получения минимального покрытия. Эта процедура проводится путем поочередного удаления избыточных ФЗ с последующей проверкой получаемого на каждом шаге набора ФЗ на наличие хотя бы одной избыточной ФЗ.
4. Использовать ФЗ из минимального покрытия для декомпозиции универсального отношения в набор НФБК-отношений. При этом применяется общий алгоритм декомпозиции.
5. Если может быть получено более чем одно минимальное покрытие, осуществляется сравнение результатов, полученных на основе различных минимальных покрытий, с целью определения варианта, лучше других отвечающего требованиям семантики области.
6. После завершения процесса проектирования, полученный набор необходимо проконтролировать на предмет наличия не выявленных проблем:
 - ⇒ одна и та же ФЗ не должна появиться более чем в одном отношении;
 - ⇒ набор ФЗ в полученных отношениях должен в точности совпадать с минимальным покрытием;
 - ⇒ не должно быть избыточных отношений.
 Отношение избыточно если:
 - ⇒ все атрибуты его могут быть найдены в одном другом отношении;
 - ⇒ все атрибуты могут быть найдены в отношении, которое может быть получено из других отношений в результате соединения.
7. Определить будут ли полученные отношения поддерживать те типы запросов и операции обновления, которые предполагается использовать.

ПРЕОБРАЗОВАНИЕ КОНЦЕПТУАЛЬНОЙ МОДЕЛИ В РЕЛЯЦИОННУЮ

Концептуальная модель данных состоит из связанных между собой сущностей. Основными характеристиками сущности является атрибут и ПК сущности. Связь характеризуется степенью и классом принадлежности.

Существует алгоритм, по которому концептуальная модель может быть переведена в реляционную, причем гарантирована по крайней мере 3НФ.

Преобразование сущностей

1. Каждой сущности ставится в соответствие отношение.
2. Каждый атрибут сущности становится атрибутом отношения, которому приписывают тип данных и свойство допустимости/недопустимости для него значения NULL (не определен).
3. Первичный ключ сущности становится ПК отношения (PRIMARY KEY). Атрибуты, входящие в ПК, получают свойство обязательности (NOT NULL) и уникальности (UNIQUE).

Преобразование связей

Преобразование связи производится согласно значениям ее характеристик.

1:1, Класс Принадлежности (КП) обеих сущностей обязательный.

Требуется одно отношение, его первичным ключом может стать ключ любой из сущностей.

1:1, КП одной сущности обязательный, другой - необязательный.

Требуется два отношения, по одному на каждую сущность. Ключ каждой сущности становится ПК соответствующих отношений. Кроме того, ключ сущности, у которой КП необязательный, добавляется в качестве атрибута (FOREIGN KEY) в отношение, выделенное для сущности с обязательным КП. В отношение, выделенное для сущности с обязательным КП, включаются также атрибуты, принадлежащие связи (если они есть).

1:1, КП обеих сущностей необязательный.

Требуется три отношения, по одному на каждую сущность, и отношение связи. Ключ каждой сущности становится ПК соответствующих отношений. Отношение связи содержит в качестве ключевых атрибутов (FOREIGN KEYS) ключи обеих сущностей. В отношение связи включаются также атрибуты, принадлежащие связи (если они есть).

1:*, КП многосвязной сущности является обязательным.

Требуется два отношения, по одному на каждую сущность. Ключ каждой сущности становится ПК соответствующих отношений. Кроме того,

ключ односвязной сущности добавляется в качестве атрибута (FOREING KEY) в отношение, выделенное для многосвязной сущности. Атрибуты связи (если они есть) добавляются в таблицу для многосвязной сущности.

1:*, КП многосвязной сущности является необязательным.

Требуется три отношения, по одному на каждую сущность, и отношение связи. Ключ каждой сущности становится ПК соответствующих отношений. Отношение связи содержит в качестве ключевых атрибутов (FOREING KEYS) ключи обеих сущностей. В отношении связи включаются также атрибуты, принадлежащие связи (если они есть).

***:*, КП обеих сущностей не имеет значения.**

Требуется три отношения, по одному на каждую сущность, и отношение связи. Ключ каждой сущности становится ПК соответствующих отношений. Отношение связи содержит в качестве ключевых атрибутов (FOREING KEYS) ключи обеих сущностей. В отношении связи включаются также атрибуты, принадлежащие связи (если они есть).

ПРЕОБРАЗОВАНИЕ РЕКУРСИВНЫХ СВЯЗЕЙ.

Отношение, имеющее рекурсивную связь, в качестве ПК имеет ключ сущности, а также включает в себя этот же ключ сущности в качестве неключевого атрибута (FOREING KEY).

Преобразование отношений супертип – подтип

Для отражения категоризации возможно несколько вариантов. Перечислим наиболее характерные.

1. Для всех сущностей Супертип–Подтипы заводится одно отношение, содержащее все атрибуты супертипа и все атрибуты всех подтипов. ПК отношения становится ключ супертипа. Значения атрибутов, характеризующих подтипы, допускают значения NULL.
2. Для супертипа и каждого подтипа заводятся разные отношения. Отношение супертипа включает одно поле для связи со всеми отношениями подтипов, которое должно быть ПК в каждом из отношений для подтипов.
3. Для супертипа и каждого подтипа заводятся разные отношения. Отношение супертипа включает по одному полю связи с каждым из отношений для подтипов, которые должны быть ПК в этих отношениях. Поля связи допускают неопределенные значения.

ПРИМЕР ПРОЕКТИРОВАНИЯ РБД НА ОСНОВЕ КОНЦЕПТУАЛЬНОЙ МОДЕЛИ

Выше мы получали РБД ПРЕПОДАВАТЕЛЬ_ПРЕДМЕТ на основе концепции функциональной зависимости. Другой подход к проектированию рассмотрим на этой же модели.

В предметной области можно выделить следующие сущности и соответствующие им атрибуты (ключи сущностей подчеркнуты).

ПРЕПОДАВАТЕЛЬ	КАФЕДРА	ПРЕДМЕТ	ТАРИФНАЯ СЕТКА
<u>Личный №</u> Фамилия	<u>Название</u> Телефон	<u>Название</u> Кол-во часов	<u>Должность</u> Оклад

При этом наблюдаются следующие связи между сущностями.

1. ПРЕПОДАВАТЕЛЬ – ПРЕДМЕТ: тип связи *:*, класс принадлежности обеих сущностей – обязательный. Каждый Преподаватель *должен* вести *хотя бы один* Предмет. Каждый Предмет *должен* вестись *хотя бы одним* Преподавателем. Кроме того, данная связь имеет атрибут связи, характеризующий количество часов, которое ведет преподаватель по предмету. Этот атрибут отличен от общего количества часов, отводимых на изучение конкретного предмета.
2. ПРЕПОДАВАТЕЛЬ – КАФЕДРА: тип связи 1:* (внимательно рассмотрите диаграмму!), класс принадлежности обеих сущностей – обязательный. На каждой Кафедре *должен* работать *хотя бы один* Преподаватель. Каждый Преподаватель *должен* работать *только на одной* Кафедре.
3. ПРЕПОДАВАТЕЛЬ – ТАРИФНАЯ СЕТКА: тип связи 1:*, класс принадлежности сущности ПРЕПОДАВАТЕЛЬ – обязательный, сущности ТАРИФНАЯ СЕТКА – необязательный. Каждый Преподаватель *должен* иметь *только одну* Должность. В каждой Должности *могут* работать *несколько* Преподавателей.

Соответствующая концептуальная модель данных изображена на рисунке 13.

Преобразуем полученную модель в РБД. Согласно правилу преобразования сущностей, каждой сущности заведем по отношению. Ключи сущностей станут первичными ключами своих отношений. Имеем:

ПРЕПОДАВАТЕЛЬ = <Личный №, Фамилия>;
КАФЕДРА = <Название, Телефон>;
ПРЕДМЕТ = <Название, Кол-во_часов>;
ТАРИФНАЯ_СЕТКА = <Должность, Оклад>.

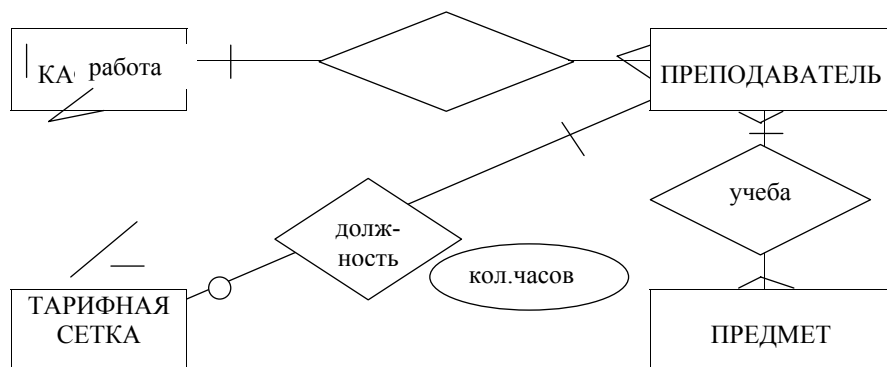


Рис. 13

Эти таблицы принято называть справочниками. Заметим, что поскольку мы не вводили сущность ТЕЛЕФОН, то проблемы с взаимно-однозначной зависимостью не существует. Телефон является атрибутом сущности КАФЕДРА. Такая трактовка предметной области допускает наличие параллельных подключений телефонов.

Согласно правилу преобразования связей один-ко-многим с обязательным классом принадлежности со стороны многосвязной сущности, ключи сущностей КАФЕДРА и ТАРИФНАЯ_СЕТКА должны быть добавлены в отношение ПРЕПОДАВАТЕЛЬ как внешние ключи:

ПРЕПОДАВАТЕЛЬ = <Личный №, Фамилия, Название_кафедры(FK), Должность(FK)>.

Согласно правилу преобразования связей многие-ко-многим в реляционную модель следует добавить отношение связи, состоящие из ключей сущностей ПРЕПОДАВАТЕЛЬ и ПРЕДМЕТ, а так же атрибута связи Количество_часов:

НАГРУЗКА = <Личный №(FK), Название(FK), Нагрузка >.

Полная схема РБД приведена на рис. 14.

Обратите внимание на способ раскрытия не желательной в реляционной модели связи типа «многие-ко-многим» путем создания дополнительной таблицы связи. Это является завуалированным методом перехода от связей типа «многие-ко-многим» к связям «один-ко-многим».

Введение дополнительной таблицы необходимо и при моделировании связи «один-ко-многим» с необязательным классом принадлежности на конце «многие». Однако побуждающие мотивы здесь иные. Поскольку класс принадлежности необязательный, следовательно, у многосвязной сущности могут существовать экземпляры, не участвующие в связи. Поэтому добавление первичного ключа односвязной сущности в отношение для многосвязной

сущности неудобно, это поле часто может оставаться неопределенным, что затрудняет поддержание целостности по ссылкам.

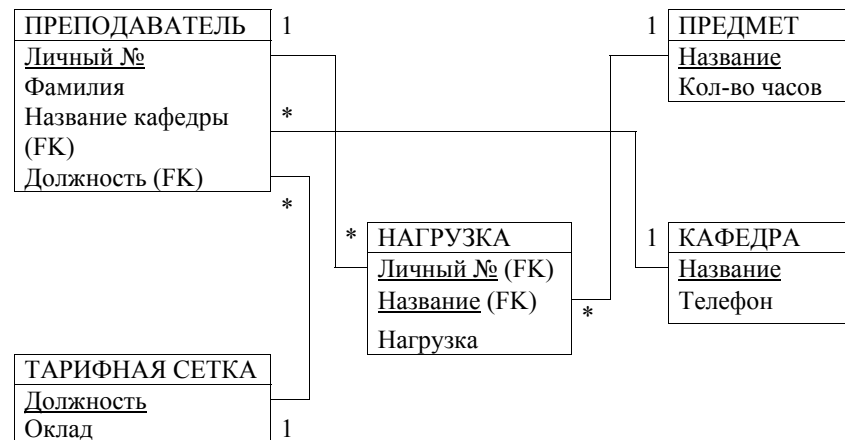


Рис. 14

Однако при моделировании реальной предметной области для предотвращения чрезмерного увеличения количества таблиц отношениями связи в некоторых случаях пренебрегают, предоставляя возможность атрибуту иметь неопределенные значения.

ЛАБОРАТОРНЫЙ ПРАКТИКУМ ПО ПРОЕКТИРОВАНИЮ РБД

Лабораторная работа № 1

Тема: Основные понятия реляционной модели данных

Задание 1.1. Определить первичный и внешние ключи каждого отношения

ОТНОШЕНИЕ КЛИЕНТ

ID-клиент	Клиент-имя	Адрес	Город	Начальный баланс
100	Лагуна	Мира, 15	Красноярск	45551
101	Магазин №1	Ленина, 6	Канск	75314
105	ТЦ	Весны, 111	Уяр	49333
110	Магазин №2	Маркса, 120	Красноярск	27400

ОТНОШЕНИЕ ТОРГОВЫЙ_АГЕНТ

ID-агент	Агент-имя	Офис	%-комиссионных
10	Иванов И.И.	Уяр	10
14	Петров П.П.	Канск	11
23	Сидоров С.С.	Красноярск	13
37	Кузнецов А.А.	Красноярск	10
39	Купцов К.К.	Красноярск	11

ОТНОШЕНИЕ ТОВАР

ID-товар	Товар-название	ID-производ.	Закупочная цена	Цена продажи
1035	Свитер	210	11.25	22.00
2241	Настольная лампа	317	22.25	33.25
2518	Бронзовая скульптура	253	13.60	21.20

ОТНОШЕНИЕ ПРОДАЖА

Дата	ID-клиент	ID-агент	ID-товар	Количество	Сумма
08.02	100	14	2241	200	6650.00
12.02	101	23	2518	300	6360.00
12.02	101	23	1035	150	3300.00
19.02	100	39	2518	200	4240.00
22.02	101	23	1035	200	4400.00
25.02	105	10	2241	100	3325.00
25.02	110	37	2518	150	3180.00

ОТНОШЕНИЕ ПРОИЗВОДИТЕЛЬ

ID-производ.	Производитель-имя	Адрес	Город
210	Одежда “Киви”	Мира, 100	Красноярск
253	Медные изделия	Новая, 15	Томск
317	Лампы Ллана	Старая, 256	Новосибирск

Задание 1.2. Объясните, каким образом неконтролируемый одновременный доступ к базе данных может вызвать проблемы в следующих ситуациях:

- при резервировании мест в системе продажи авиабилетов;
- при обновлении количества товара в системе складского учета;
- при обновлении баланса текущих счетов в банке.

Лабораторная работа № 2

Тема: Концептуальное проектирование

В каждой из следующих задач создайте концептуальную модель данных, которая давала бы ответы на вопросы, подобные данным. Укажите сущности и их атрибуты, связи, их степени и классы принадлежности.

Замечание. Конкретные значения (например, “факультет информатики и вычислительной техники” и “курсы по программированию”) взяты для примера. Ваша модель должна отвечать на любые аналогичные вопросы.

1. Сколько преподавателей работает на факультете информатики и вычислительной техники? Их фамилии? Кто работает на радиотехническом факультете?
2. Какие студенты специализируются в английском? В защите информации?
3. Кто из преподавателей читает курсы по программированию? Какие курсы они читают?
4. Сколько студентов занимаются по программе “Информационные системы, базы и банки данных”? Какие курсы изучает группа ВТ 29-4 в весеннем семестре 2002 года?
5. Сколько королей Пруссии носили имя Фредерик? В какие годы они жили, а в какие – правили? Управляли ли они на протяжении своей жизни какими-либо еще странами?
6. Управлялись ли в XVII веке какие-либо европейские страны женщинами? Если да, то какие?
7. Правил ли дед Марии - Антуанетты какой-либо страной? Какой и когда? Кто была ее мать? Были ли случаи, когда правители двух разных стран женились между собой?
8. Сколько детей Генриха VIII стали королями Англии? Кто были их матери?

9. Репортажи о скольких футбольных матчах ОРТ показала за последний год? Когда она транслировала встречи между “Спартак” и “Динамо”? Матчи какой команды показывались больше всего? Как насчет хоккейных матчей?

10. Был ли показан хоть один теннисный матч с участием Штеффи Граф? Когда и какой канал его транслировал?

11. Какие коммерческие объявления “Афонтово” показала более трех раз в течение одного часа? Когда это было? В течение какого часа, какого числа?

12. Какую плату “Афонтово” назначила за трансляцию коммерческих сообщений? Какова стоимость размещения рекламы? Какого числа, сколько раз и в какое время транслировалась реклама фирмы “Вентокальдо”?

Объедините в общую модель представления данных (или моделей данных), которые вы создали в следующих задачах:

⇒ В задачах 1, 2, 3, 4.

⇒ В задачах 5, 6, 7, 8.

⇒ В задачах 9, 10, 11, 12.

Лабораторная работа № 3

Тема: Проектирование РБД с помощью концепции функциональной зависимости

Задание 3.1. Рассмотрим следующую реляционную таблицу X (заглавные буквы обозначают имена атрибутов, строчные буквы и цифры – значения атрибутов).

X				
A	B	C	D	E
a1	b2	c1	d3	e2
a3	b2	c3	d2	e4
a1	b3	c1	d1	e2
a2	b4	c1	d4	e2

Если считать, что пополнение таблицы происходит не будет, то какие из перечисленных функциональных зависимостей выполнены для X?

A -> C B -> E C -> A E -> B
E -> A C -> B B -> D B -> A

Задание 3.2. Рассмотрим следующую реляционную таблицу Y (заглавные буквы обозначают имена атрибутов, строчные буквы и цифры – значения атрибутов).

Y				
A	B	C	D	E
a1	b2	c1	d3	e2
a2	b2	c3	d3	e4
a1	b3	c2	d1	e2
a2	b4	c5	d1	e5

Если считать, что пополнение таблицы происходит не будет, то какие из перечисленных функциональных зависимостей не выполнены для Y?

A -> C B -> E C -> A E -> B
E -> A C -> B B -> D B -> A

Задание 3.3. Установите соотношение между терминами и объяснениями к ним.

- | | |
|---|---|
| a Нормальная форма Бойса-Кодда | a Все значения должны быть атомарные |
| b Множество возможных первичных ключей | b Минимальный набор атрибутов однозначно определяющих любой кортеж отношения |
| c Первая нормальная форма | c Каждый детерминант отношения является первичным ключом этого отношения. |
| d Первичный ключ | d Никакой неключевой элемент не может зависеть от части первичного ключа |
| e Кортеж | e Неключевой атрибут функционально зависит от части первичного ключа |
| f Транзитивная зависимость | f Множество минимальных наборов атрибутов, каждый из наборов однозначно определяет все не входящие в него атрибуты отношения |
| g Вторая нормальная форма | g Строка реляционной таблицы |
| h Функционально неполная зависимость | h Неключевой атрибут функционально зависит от одного или нескольких неключевых атрибутов |

Задание 3.4. Руководство супермаркета желает разработать БД, предназначенную для хранения информации о счетах покупателей. Информация, касающаяся каждого покупателя, должна включать следующее: номер счета,

фамилию, адрес, номер телефона, кредитоспособность (высокая, средняя, низкая, слабая). Изобразите диаграмму ФЗ, используя соответствующие атрибуты и выделяя сделанные предположения. Получить для БД отношения в НФБК.

Задание 3.5. Создайте реляционную схему, все таблицы которой имеют 4НФ, для следующей информации компании по страхованию жизни:

У компании имеется большое количество полисов. Для каждого полиса мы хотим знать имя держателя полиса, его адрес и дату рождения. Нам также нужно знать номер полиса, годовой процент и сумму выплат в случае смерти. Кроме того, мы хотим знать номер агента, имя и место проживания агента, выписавшего полис. Держатель полиса может иметь несколько полисов, и один агент может выписать много полисов.

Задание 3.6. Для атрибутов отношения $R = \langle A, B, C, D, E \rangle$ определены следующие функциональные зависимости:

1. $A \rightarrow \langle B, C, D, E \rangle$;
2. $C \leftrightarrow D$.

Привести отношение R к НФБК, указать атрибуты, обеспечивающие целостность, получившейся БД, определить степени связей (1:1, 1:*, *:*)).

Задание 3.7. Для атрибутов отношения $R = \langle A, B, C, D, E \rangle$ определены следующие функциональные зависимости:

1. $\langle A, B, C \rangle \rightarrow \langle D, E \rangle$;
2. $\langle A, B \rangle \leftrightarrow E$.

Привести отношение R к НФБК, указать атрибуты, обеспечивающие целостность, получившейся БД, определить степени связей (1:1, 1:*, *:*)).

Задание 3.8. Для атрибутов отношения $R = \langle A, B, C, D, E \rangle$ определены следующие функциональные зависимости:

1. $\langle A, B, C \rangle \rightarrow \langle D, E \rangle$;
2. $\langle B, C \rangle \leftrightarrow D$.

Привести отношение R к НФБК, указать атрибуты, обеспечивающие целостность, получившейся БД, определить степени связей (1:1, 1:*, *:*)).

Задание 3.9. Для атрибутов отношения $R = \langle A, B, C, D, E \rangle$ определены следующие функциональные зависимости:

1. $\langle D, E \rangle \rightarrow \langle A, B, C \rangle$;
2. $\langle B, C \rangle \leftrightarrow A$.

Привести отношение R к НФБК, указать атрибуты, обеспечивающие целостность, получившейся БД, определить степени связей (1:1, 1:*, *:*)).

Задание 3.10. Для атрибутов отношения $R = \langle A, B, C, D, E \rangle$ определены следующие функциональные зависимости:

1. $\langle A, B \rangle \rightarrow \langle C, D, E \rangle$;
2. $A \rightarrow \langle D, E \rangle$;
3. $B \leftrightarrow C$.

Привести отношение R к НФБК, указать атрибуты, обеспечивающие целостность, получившейся БД, определить степени связей (1:1, 1:*, *:*)).

Задание 3.11. В отношении “Гурман” определены атрибуты:

- | | |
|---|--|
| Б1 - наименование блюда; | И1 - наименование ингредиента блюда; |
| Б2 - категория блюда (напитки, мясо, холодные закуски и пр.); | И2 - расход этого ингредиента на приготовление одной порции блюда; |
| Б3 - калорийность; | Р1 - код рецепта приготовления; |
| Б4 - отпускная цена; | Р2 - полное описание рецепта приготовления блюда. |

Определено множество функциональных зависимостей атрибутов отношения “Гурман”:

$B1 \rightarrow \langle B2, B3 \rangle$; $R1 \rightarrow R2$; $I1 \rightarrow I2$; $\langle I1, I2 \rangle \rightarrow \langle R1, R2 \rangle$; $\langle B1, R1, B3 \rangle \rightarrow B4$; $\langle I1, B1 \rangle \rightarrow B4$.

Построить минимальное покрытие. Привести отношение “Гурман” к НФБК. Указать атрибуты, обеспечивающие целостность, полученной БД.

КОНТРОЛЬНЫЕ ЗАДАНИЯ ПО ПРОЕКТИРОВАНИЮ РБД

Вариант 1

Разработать концептуальную модель данных работы цеха. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Цех получает заказы, называемые проектами, для которых определены дата получения и дата выполнения проекта. Для выполнения проекта необходимо заказать у поставщика требующиеся детали. Каждый поставщик может поставлять различные детали. Одна и та же деталь может поставляться для одного проекта разными поставщиками. Деталь характеризуется наименованием, весом, ценой, которая может быть разной у различных поставщиков, но не зависит от проекта. Поставщики характеризуются наименованием, адресом, юридическим адресом.

БД должна уметь отвечать на вопросы, подобные следующим.

Если в детали В обнаружен брак, то следует узнать, кто ее поставил.

К какому сроку должны быть выполнены все проекты, заказавшие деталь В?

Сколько деталей С необходимо поставить к какому-либо сроку?

Кто поставляет детали для всех проектов?

Вариант 2

Разработать концептуальную модель данных о каталоге музыкальных компакт-дисков. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Вся фонотека хранится на компакт-дисках и включает в себя как сборники, так и сольные альбомы. Следует различать музыку по стилям, записи – по исполнителям. Сведения о диске должны содержать информацию о годе выпуска, количестве композиций, общем времени звучания. Исполнитель характеризуется именем или сценическим именем или названием группы, а также страной. Не забудьте о классике!

БД должна уметь отвечать на вопросы, подобные следующим.

Выдать краткое описание стиля, в котором работает исполнитель А?

Сколько в фонотеке сольных CD, а сколько сборников?

Какие композиции коллекции записаны различными исполнителями?

Что нового в коллекции?

Вариант 3

Разработать концептуальную модель БД каталога компьютерных компакт-дисков. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Диски делятся на 2 типа – игровые и прикладные. Игры характеризуются названием стиля (“квест”, “аркада”, “стратегия” и пр.), прикладные программы делятся на трансляторы, редакторы, эл. таблицы, СУБД и т.д.). Игры характеризуются названием, годом выпуска, страной-производителем. Необходимо знать системные требования, предъявляемые как играми, так и ППО.

БД должна уметь отвечать на вопросы, подобные следующим.

Выдать краткое описание стиля игры А?

Сколько в коллекции игр, а сколько наименований ПО?

Какие версии транслятора C++ имеются в коллекции? Каковы требования к самой последней? Самой ранней?

Что нового в коллекции игр?

Вариант 4

Разработать концептуальную модель БД учета садовых посадок. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Владелец имеет несколько участков, именуемых далее Садами. Сад характеризуется наименованием местности, а также кратким описанием почвы. В саду растут фруктовые деревья различных видов (персики, сливы, яблони и т.д.). Каждый вид имеет сорта (груша - бергамот, яблоня – антоновка, и т.д.)

Поскольку дерево можно прививать, то на одном дереве может быть несколько сортов плода данного вида (т.е. дерево относится к одному виду, но может нести несколько сортов). Разумеется, существует множество деревьев каждого вида и сорта. Наконец, каждый сорт относится только к одному виду, тогда как каждый вид имеет несколько сортов. Каждое дерево характеризуется также годом посадки и годом гибели (если дерево живое, то этот атрибут пуст).

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько сортов персиков в саду А?

Сколько деревьев в среднем погибает в год в саду В?

Каков средний возраст яблонь?

На скольких сливах привито по несколько сортов?

Вариант 5

Разработать концептуальную модель БД малого банка. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

У банка есть текущие счета, сберегательные счета и клиенты. При этом один и тот же клиент может иметь несколько сберегательных и один текущий счета. Любой счет, выписывается только на одного клиента. Клиенты делятся на физических лиц (характеризуются именем, адресом, телефонами, полом, датой рождения) и юридических лиц (характеризуются типом организации, юридическим адресом, количеством служащих, данными о представителе, имеющем право подписи). Счет описывается балансом.

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько текущих счетов у банка?

Сколько сберегательных счетов?

Сколько клиентов?

У скольких клиентов несколько текущих счетов?

Каков процент сберегательных счетов, с балансом выше \$1000?

Вариант 6

Разработать концептуальную модель данных работы фирмы-производителя. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Фирма имеет заводы, выпускающие некоторую продукцию, несколько конструкторских бюро (КБ), разрабатывающих новую продукцию, склады. Завод может выпускать несколько видов продукции, и одну и ту же продукцию могут выпускать разные заводы. На складах хранится продукция всех

заводов. Несколько КБ могут разрабатывать один и тот же вид продукции, но в производство запускается только одна из них. Продукция характеризуется наименованием, серийным номером, весом, себестоимостью, которая может быть разной у различных заводов, но не зависит от разработавшего ее КБ. Заводы, КБ и склады характеризуются наименованием, юридическим адресом, именем руководителя.

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько видов продукции А выпускает фирма? Конкретный завод?

Сколько видов продукции разработано в КБ Z за последний год? Сколько разработок не закончено? Сколько закончилось неудачей?

На какую сумму завод В выпустил продукции В за отчетный период?

Получить список ФИО и рабочих телефонов руководителей всех подразделений фирмы?

Вариант 7

Разработать концептуальную модель данных учета книг в библиотеке. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Фонд библиотеки состоит из книг, которые описываются автором(и), названием, издательством, годом выпуска, количеством страниц, тематикой, количеством экземпляров. Экземпляры книги однозначно характеризуются своими инвентарными номерами. Книги могут быть произведением (сборниками произведений) одного писателя, а могут быть тематическими сборниками разных авторов. Пользователи библиотеки – читатели, которые могут брать книги домой. Библиотека поддерживает алфавитный и тематический каталоги.

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько читателей пользуются библиотекой?

Сколько книг находится на руках?

Сколько книг находится на руках у конкретного читателя? Какие это книги?

Какие книги данного автора есть в библиотеке? Сколько из них в соавторстве? Сколько произведений данного автора вошло в сборники?

Сколько книг данной тематики есть в библиотеке?

Вариант 8

Разработать концептуальную модель деканата. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

На факультете читается некоторое количество курсов. Курс предполагает наличие лекций, лабораторных и практических работ. Один курс может читаться только одним преподавателем, но практические и лабораторные работы могут вести несколько преподавателей. Каждый преподаватель может читать несколько курсов, вести только практики или лабораторные. Студенты делятся на группы. Каждая группа слушает одни и те же курсы, и каждый курс может слушать несколько групп.

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько студентов учится на факультете?

Сколько курсов читает преподаватель А? Сколько он ведет практических занятий? Сколько лабораторных? Какие это курсы?

Какие курсы изучает студент В? Группа С?

Сколько часов отводится на преподавание дисциплины Д?

Вариант 9

Разработать концептуальную модель строительной компании. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Строительная компания возводит различные здания. Для работ требуются разнообразные материалы в различных количествах. На разных этапах работы работают различные бригады (например, бригады каменщиков, штукатуров-маляров, кровельщиков, сантехников). Составляя график работ, фирма варьирует состав бригад, т.е. рабочие назначаются в разные бригады в разное время. Рабочий имеет только одну специальность. Для каждой бригады назначается бригадир. Численность бригады варьируется от здания к зданию. Бригадир может быть простым рабочим в другой бригаде. Здание описывается типом, уровнем сложности, адресом. Рабочий характеризуется специальностью, окладом, который не зависит от выполняемых на данный момент работ.

БД должна уметь отвечать на вопросы, подобные следующим.

Кто из рабочих в какую бригаду, на каком здании назначен? Какой у него оклад?

Каков график работ на здании А (кто и когда и какой период времени там должен работать)?

Какие материалы требуются при возведении здания В?

Вариант 10

Разработать концептуальную модель поваренной книги+меню ресторана. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Книга содержит описания рецептов приготовления блюд. Рецепт характеризуется типом блюда (первые/вторые блюда, напитки, сдоба, торт/пирожное и т.д.), набором требуемых продуктов (ингредиентов) с описанием количества на одну порцию, калорийностью, ценой одной порции. Продукты разделяются на типы (мясо птицы, мясо домашних животных, дичь, рыба, овощи и т.д.). Продукты описываются так же названием, единицами измерения (граммы, штуки, тонны, литры, бутылки и т.д.), поставщиками и закупочной ценой одной единицы продукта. Один продукт может поставляться разными поставщиками по разным ценам. Один поставщик может поставлять несколько наименований и даже видов продукции.

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько ингредиентов необходимо для приготовления блюда А? Каких? Как блюдо готовится?

Кто поставляет продукт Б по наиболее выгодной цене?

Какова калорийность заказанного ужина? А какова его цена?

Что можно приготовить из свеклы? А из моркови и свинины вместе?

Вариант 11

Разработать концептуальную модель учета результатов олимпиады по программированию. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Олимпиада проходит в несколько этапов. Каждый этап содержит наперед заданное количество задач. Заранее известна шкала максимально возможного количества баллов за задачи этапа и максимально допустимое общее время этапа. Этап проводится как личное первенство (участник описывается анкетными данными и ВУЗом), и, в то же время, в рамках этапа производится командный зачет (команда характеризуется ВУЗом, Однако один ВУЗ может заявить несколько команд). В каждой команде одинаковое, заранее известное количество участников, один человек может участвовать только в одной команде. В конце каждого этапа выдается протокол, содержащий сведения о баллах, полученных каждым участником за каждое задание и общем времени выполнения заданий этапа. Определение места участника проводится по некоторой формуле с учетом этих показателей. Место команды определяется сложением баллов ее участников.

БД должна уметь отвечать на вопросы, подобные следующим.

Сколько участников в личном зачете? Сколько команд? Кто в какой команде? В каком городе находится ВУЗ победителя?

Сколько баллов набрал победитель? А победила ли его команда?

Команда какого ВУЗа победила в первом туре? В скольких турах победила какая-либо команда из г. Красноярска?

Вариант 12

Разработать концептуальную модель рекламного агентства. По полученной модели построить БД. Показать, что полученная БД находится в форме Бойса-Кодда. Если это не так, выполнить нормализацию.

Описание предметной области:

Агентство размещает рекламу и коммерческие объявления на нескольких каналах. Цена одной минуты рекламного времени зависит от канала и времени показа рекламы (прайм-тайм, утро, новости, погода и т.д.). Эта цена так же зависит от общего временного объема размещаемого заказа, но не зависит ни от фирмы-заказчика, ни от периода времени, в течение которого должен быть показан этот объем. Коммерческое объявление характеризуется типом, из которого можно определить единицы измерения (секунды, слова и т.д.), количество которых однозначно определяют цену одного показа данного коммерческого объявления. Эта цена зависит только от канала. Количество и периодичность показов коммерческих объявлений в сутки строго фиксировано и так же зависит только от канала. Фирма-заказчик пользуется скидкой у рекламного агентства (но не у канала), которая зависит от общего объема заказанного проката рекламной продукции и может быть накопительной.

БД должна уметь отвечать на вопросы, подобные следующим.

На каком канале выгоднее размещать рекламные ролики в прайм-тайм?

Где чаще показывают коммерческие объявления? А где дешевле?

Какая скидка на размещение рекламы у фирмы С?

На какую сумму размещен последний заказ фирмы В? На каком канале?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Глушаков С.В., Ломотько Д.В. Базы данных. Учебный курс. – М.: АСТ, 2000.
2. Грэй П. Логика, алгебра и базы данных. - М.: Машиностроение, 1989.
3. Дейт К. Введение в системы баз данных. – М.: Наука, 1998.
4. Джексон Г. Проектирование реляционных баз данных для использования с микроЭВМ. – М.: Мир, 1991.
5. Карпова Т. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2001.
6. Кузнецов С.Д. Основы современных баз данных. – Информационно-аналитические материалы Центра Информационных технологий. – <http://www.citforum.ru/>

7. Соломон Д. и др. Microsoft SQL Server 6.5. Энциклопедия пользователя: Пер. с англ. – Киев: Издательство “ДиаСофт”, 1998.

8. Тихомиров Ю.В. Microsoft SQL Server 7.0: разработка приложений. – СПб.: БХВ – Санкт-Петербург, 1999.

9. Хансен Г., Хансен Д. Базы данных: разработка и управление: Пер. с англ. – М: ЗАО “Издательство БИНОМ”, 1999.

Чери С. и др. Логическое программирование и базы данных. - М.: Мир, 1992

Содержание

Информационные системы и СУБД	3
История развития	3
Основные функции современной СУБД	4
Модели данных	5
Ранние подходы к организации БД	6
Принципы концептуального проектирования	7
Базовые понятия модели “Сущность – связь”	8
Реляционная модель данных	12
Базовые понятия реляционной модели данных	12
Свойства отношений	14
Реляционная модель данных: три составляющие	15
Проектирование РБД с помощью концепции функциональных зависимостей	17
Понятие функциональной зависимости	17
Вторая и третья НФ. Алгоритм декомпозиции	19
Нормальная форма Бойса-Кодда	21
Многозначные зависимости. Четвертая нормальная форма	23
Избыточные ФЗ. Минимальное покрытие	24
Преобразование концептуальной модели в реляционную	28
Преобразование сущностей	28
Преобразование связей	28
Преобразование отношений супертип – подтип	29
Пример проектирования РБД на основе концептуальной модели	30
Лабораторный практикум по проектированию РБД	33
Лабораторная работа № 1	33
Лабораторная работа № 2	34
Лабораторная работа № 3	35
Контрольные задания по проектированию РБД	38
Библиографический список	44

Реляционная модель данных
Евгения Дмитриевна Карпова

Редактор И.А. Вейсиг
Корректурa автора

Подписано в печать 22.04.2004 г. Уч.-изд. л. 2,9

Тиражируется на электронных носителях

Заказ 305

Дата выхода 23.09.2004

Адрес в Internet: www.lan.krasu.ru/studies/editions.asp

Отдел информационных ресурсов управления информатизации КрасГУ

660041 г. Красноярск, пр. Свободный, 79, ауд. 22-05, e-mail: info@lan.krasu.ru

Издательский центр Красноярского государственного университета

660041 г. Красноярск, пр. Свободный, 79