А.П. Костюкова

БАЗЫ ДАННЫХ:
ОБЩИЙ КУРС
Часть 1

Учебное пособие

Рекомендовано УМО РАЕ
по классическому университетскому
и техническому образованию в качестве
учебного пособия для студентов
высших учебных заведений,
обучающихся по направлениям:
27.03.03 – «Системный анализ
и управление»;
27.03.04 – «Управление в технических
системах»

Москва
2018

УДК 004.14:681.322
ББК 32.973.26 - 018.2
К72

Рецензенты:

Артемов М.А. – доктор физико-математических наук, профессор;
Надеждин Е.Н. – доктор технических наук, профессор.

Костюкова А.П.

К72 Базы данных: общий курс. Часть 1: учебное пособие. – М.: Издательский дом Академии Естествознания, 2018. – 82 с.

ISBN 978-5-91327-538-7

В учебном пособии рассмотрены основные вопросы построения баз данных, как составляющих информационного ресурса общества, классификация, методы и технологические аспекты обработки информации, принципы построения и использования различных видов моделей данных.

Пособие предназначено для студентов специальностей «Управление и информатика в технических системах», «Системный анализ и управление», изучающих дисциплину «Базы данных».

ISBN 978-5-91327-538-7

© Костюкова А.П., 2018
© ИД «Академия Естествознания»
© АНО «Академия Естествознания»

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 5
1. ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ 7
1.1. Предметная область, информация, данные 7
1.2. Банк данных 9
Контрольные вопросы 17
2. МОДЕЛЬ ДАННЫХ 18
2.1. Типы и структуры данных 18
2.2. Модели данных 20
Контрольные вопросы 21
3. КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ДАННЫХ 22
3.1. Назначение концептуальной модели данных 22
3.2. Модель «сущность – связь» 24
3.3. Нотация *IDEF1X* 38
Контрольные вопросы 45
4. ДОРЕЛЯЦИОННЫЕ МОДЕЛИ ДАННЫХ 46
4.1. Иерархическая модель данных 46
4.2. Сетевая модель данных 51
Контрольные вопросы 53
5. РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ 54
5.1. Структура данных 54
5.2. Операции над данными 58
5.3. Ограничения целостности 59
Контрольные вопросы 59
6. ТЕОРИЯ НОРМАЛЬНЫХ ФОРМ 60
6.1. Понятие нормализации 60

6.2. 1*NF* – первая нормальная форма. Понятия простого и сложного атрибута ... 63
6.3. 2*NF* – вторая нормальная форма. Полная функциональная зависимость ... 64
6.4. 3*NF* – третья нормальная форма. Транзитивная зависимость ... 67
6.5. Нормальная форма Бойса – Кодда ... 68
6.6. Многозначные зависимости и приведение отношения к четвертой нормальной форме ... 69
6.7. Зависимости по соединению и пятая нормальная форма (5*NF*) ... 70
Контрольные вопросы ... 71
7. ОГРАНИЧЕНИЯ ЦЕЛОСТНОСТИ В РАСШИРЕННОЙ РЕЛЯЦИОННОЙ МОДЕЛИ ДАННЫХ ... 72
7.1. *NULL*-значения. Трехзначная логика ... 73
7.2. Целостность сущностей (категорная целостность) ... 74
7.3. Целостность ссылок ... 74
7.4. Основные операции, приводящие к нарушению целостности БД. Стратегии поддержания целостности данных ... 76
7.5. Корпоративные ограничения целостности данных (бизнес-правила предметной области) ... 78
Контрольные вопросы ... 79
СПИСОК ЛИТЕРАТУРЫ ... 80

ВВЕДЕНИЕ

Одним из объектов профессиональной деятельности инженеров по специальностям «Управление и информатика в технических системах», «Системный анализ и управление» являются автоматизированные системы обработки информации и управления в различных технических системах. Информационную основу таких систем составляют компьютерные базы (БД) и банки данных (БнД).

Рассмотрим некоторые понятия [1–3].

Информационное обеспечение систем управления (СУ) – совокупность форм документов и внедренных решений по размерам, местам размещения и способам существования информации, которая применяется в действующей СУ.

Информационно-технологическая инфраструктура СУ – это набор информационно-технологических устройств и средств, обеспечивающих реализацию информационных процессов.

Процесс (лат. *processus* – продвижение) – 1) последовательность смены явлений либо состояний в развитии некоторой системы, явления объекта; 2) совокупность последовательности событий для реализации результата (цели); 3) специальным образом упорядоченная совокупность заданий. работ (*activities*) в пространстве и во времени, с определением начального и конечного сроков и точным указанием входных и выходных характеристик/объектов.

Информационная система (ИС) – некое организационно упорядоченное множество документов или массивов документов, а также информационных технологий, быть может, с использованием средств компьютерной техники и телекоммуникационной связи, для реализации информационных процессов.

Информационные процессы – это процессы сбора, обработки, накопления, хранения, актуализации, распространения и поиска информации.

Процедура приема данных заключается в считывании пользователем ИС данных.

Процедура обработки данных заключается в формировании набора данных, требуемых пользователем информационной системы для решения функциональных или управленческих задач из данных, содержащихся в хранилищах собранных данных.

Процедура накопления данных включает последовательность действий по включению собранных данных в систему информационных ресурсов СУ.

Процедура хранения данных заключается в обслуживании, архивов и баз данных, с целью предотвращения потери накопленных данных и уничтожения их злоумышленниками различного типа.

Процедура актуализации данных заключается в обновлении содержания документов библиотек, архивов и файлов баз данных, с целью адекватного отражения в информационной модели СУ текущего состояния внешней и внутренней среды.

Процедура поиска данных заключается в формировании из данных, содержащихся в хранилищах собранных данных, набора данных, соответствующих запросу пользователя информационной системы.

Процедура распространения данных заключается в доставке данных в интервалы времени, установленные регламентами функционирования информационной системы.

Изучению данных вопросов и посвящена дисциплина «Базы данных».

1. ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

План темы

1. Предметная область, информация, данные.
2. Банк данных.

Ключевые слова: *предметная область, информация, данные, база данных, банк данных, система управления базой данных, метаданные, словарь данных, администраторы, языковые средства банка данных.*

1.1. Предметная область, информация, данные

Охарактеризуем некоторые основные понятия теории баз данных [1, 4–6], которыми будем пользоваться в дальнейшем.

Предметная область – это часть реального мира, которая подлежит изучению с целью автоматизации организации управления.

Предметной областью информационной системы является совокупность объектов, свойства которых и отношения между которыми представляют интерес для пользователей ИС.

Любая предметная область может быть разбита на фрагменты. Каждый фрагмент оперирует со своими объектами и с множеством пользователей, которые имеют свои взгляды на предметную область, поэтому выявление предметной области и ее анализ является неотъемлемой частью разработки любой информационной системы.

В компьютерных системах описание некоторого формального объекта определяется посредством установления разнообразных по своим свойствам отношений этого объекта с другими объектами такого же рода.

Информация (Федеральный Закон) – сведения о предметах, лицах, событиях фактах, процессах и явлениях независимо от формы их представления.

или

Информация – это совокупность знаний (фактических данных) об объектах предметной области и зависимостях между ними.

Информация является одним из видов ресурсов, которые используются человеком в быту и профессиональной деятельности. К ресурсам относятся время, сырье, персонал и финансы. Каждый из видов ресурсов обладает такими свойствами как полезность и стоимость.

Полезность информации подтверждается тем, что любое предприятие при управлении другими ресурсами зависит от обеспеченности информацией. Отсутствие своевременной и достоверной информации усложняет деятельность любого предприятия. В условиях конкуренции данные становятся товаром и, следовательно, обладают стоимостью.

Информация бывает трех видов:

1) оперативная;

2) тактическая;

3) стратегическая.

Оперативная информация извлекается из оперативных данных (ведомости, данные инженерных расчетов и т.п.). Тактическая информация представляет собой информацию, полученную после обработки оперативной информации (например, квартальные отчеты). Стратегическая информация – это результаты прогнозирования.

Данные – термин, который связан с представлением информации. Иными словами, сведения, зафиксированные на каком-либо материальном носителе, принято называть данными.

Данные – информация, представленная в виде пригодном для обработки ИС.

Возросший объем данных приводит к необходимости управления данными, т.е. приводит к базам данных.

БД – это системы хранения информации, обращение к которым может осуществляться через средства управления базами данных (СУБД).

Основное предназначение БД – предоставление пользователю необходимой информации в нужном месте и в нужное время. Индустрия СУБД [6, 7] в настоящее время вполне состоялась. Условия на рынке БД в настоящее время определяют такие фирмы как *Oracle*, *IBM* и *Microsoft*.

Необходимо выделить следующие формы деятельности БД.

Во-первых, оперативная обработка информации. Приложения с такими БД используются для поддержания ежедневной активности

предприятия. Такие приложения требуют быстрой реакции и жесточайшего контроля над целостностью и безопасностью данных. Примерами таких приложений являются системы учета отработанного времени работниками предприятия; системы учета на складе; система учета книг в библиотеке и т.п.

Во-вторых, системы поддержки принятия решений. Они используются с целью анализа данных. Пользователям в таких приложениях должна быть предоставлена возможность конструирования запросов к БД различной сложности, осуществления поиска зависимостей, вывода данных в графических формах и передачи данных в другие приложения, например, текстовые редакторы или статистические пакеты.

С другой стороны приложения в зависимости от области применения подразделяются на два типа:

- офисные БД, предназначенные для поддержания деятельности средних и небольших фирм, не связанных с производственной деятельностью;
- БД для промышленной автоматизации.

С точки зрения организации информации производственная автоматизация несколько запаздывает по сравнению с офисной. Это связано с тем, что традиционные БД зачастую неприменимы для систем промышленной автоматизации.

Вот некоторые ограничения:

1. Генерация данных в производственных процессах происходит с большой скоростью и для хранения производственного архива с 7000 переменных в БД каждую секунду необходимо вставлять 7000 строк. Офисные БД не обладают таким быстродействием.

2. Заводской архив с таким количеством характеристик (7000) требует под размещение БД примерно 1 Терабайт дисковой памяти. Сегодняшние БД такими объемами манипулировать не могут.

1.2. Банк данных

Банк данных является современной формой организации хранения и доступа к информации. Существует значительное разнообразие определений понятия «банк данных». В «общеотраслевых руководящих материалах по созданию банков данных» дается следующее определение.

Банк данных – это специальным образом организованная система данных (баз данных), технических, языковых, программных

и организационно-методических средств, которые предназначены обеспечить централизованное накопление и коллективное многоцелевое использование данных.

В таком определении подчеркивается, что БнД является сложной системой, включающей в себя несколько подсистем, необходимых для функционирования любой системы автоматизированной обработки данных.

Использование БнД при правильной его организации должно существенно изменить деятельность той организации, где он внедряется, сократить документооборот, формы документов, перераспределить функциональные обязанности сотрудников.

БнД является сложной человеко-машинной системой, включающей в себя совокупность различных взаимосвязанных и взаимозависимых компонент (рис. 1.1).

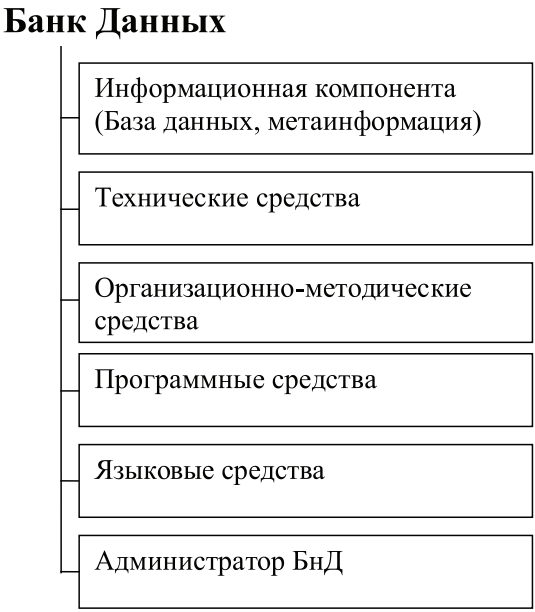


Рис. 1.1. Состав банка данных

Информационная компонента является ядром БнД поскольку включает в себя БД.

Здесь **БД** – совокупность данных, которая организована по определенным правилам и предусматривает общие принципы описания, хранения и манипулирования этими данными, не зависящая от прикладных программ, и находящаяся под управлением СУБД.

В состав БД включаются не только хранимые данные о предметной области, но и собственно описание БД, которое относится к метаинформации, то есть информации характеризующую какую-либо другую информацию.

Метаинформация включает в себя информацию о схеме БД, описание взглядов различных категорий пользователей на предметную область и информацию о проектных решениях. Центральное хранилище метаинформации называется словарем данных. Роль метаинформации возрастает при использовании автоматизированных средств автоматизированного проектирования.

Другой отличительной особенностью БнД является то, что в ней имеются специальные языковые и программные средства, облегчающие выполнение операций, связанных с организацией хранения данных и обеспечения доступа к ним. Комплекс программных средств, которые обслуживают базу данных и позволяют использовать содержащуюся в ней информацию многим пользователям, называют СУБД.

Функции СУБД:

1. Управление данными во внешней памяти (позволяет пользователям выполнять основные операции с данными: сохранять, извлекать и обновлять информации).

2. Управление транзакциями. К транзакциям относится совокупность операций над БД, рассматриваемых СУБД как единое целое. В тоже время, транзакция – это набор действий, которые могут быть выполнены с целью доступа или изменению содержимого БД. В случае, когда необходимо внести в БД сразу несколько изменений, создается сложная транзакция. При использовании транзакций не нарушается целостное состояние БД.

3. Восстановление БД. СУБД должна быть в состоянии восстановить после любого аппаратного или программного сбоя последнее согласованное состояние БД. Это требует избыточности хранения данных. Наиболее распространенным методом поддержания такой избыточности является ведения журнала изменения БД.

4. Поддержка языков БД.

5. Ведение словаря данных. Словарь данных – системный каталог с данными о схемах, пользователях, приложениях и т.д.

6. Управление параллельным доступом. При одновременном доступе к обрабатываемым данным СУБД должна гарантировать, что пользователи не будут мешать друг другу, то есть не произойдут конфликты с нежелательными последствиями для хранимых данных.

7. Управление буферами оперативной памяти. СУБД зачастую функционирует с БД значительного объема. Поэтому при обращении к каждому элементу данных необходимо осуществлять обмен с внешней памятью, что вызывает значительное замедление работы системы, Реальное увеличение скорости возможно за счет буферизации данных в оперативной памяти, но возможности общесистемной буферизации при этом оказывается недостаточно. Поэтому СУБД поддерживает свой собственный набор буферов оперативной памяти с собственной дисциплиной замены буферов.

8. Контроль доступа к данным. СУБД должна поддерживать различный уровень доступа к данным разных категорий пользователей.

9. Поддержка обмена данными. Поддерживает работу в локальных сетях и обладает возможностью объединения с коммуникационным программным обеспечением (ПО) и с различными существующими менеджерами по обмену данными.

10. Поддержка целостности данных, которая предполагает, что СУБД содержит сведения обо всех правилах, которые ни в коем случае нельзя нарушать при работе с данными и обладает инструментами контроля над данными и их изменениями по их соответствиям этим правилам.

11. Поддержка независимости от данных, то есть независимость программ от структур данных.

Основными средствами СУБД являются:

– инструменты структурирования данных (ЯОД – языки описания данных);

– инструменты обработки данных, позволяющие заносить данные в БД, удалять, модифицировать или выбирать существенные данные (языки манипулирования данными – ЯМД);

– инструменты конструирования экранных форм, которые предназначены для ввода данных, просмотра и обработки в диалоговом режиме (генераторы экранных форм);

– инструменты создания запросов для поиска данных при заданных условиях (язык запросов);

– инструменты создания отчетов, вывода на печать результатов обработки в удобном для пользователя виде (генераторы отчетных форм);

– графические языки выборки данных по различным критериям;

– инструменты создания пользовательских приложений, которые позволяют объединять разные операции работы с базой данных в единый технологический процесс (генераторы приложений);

– инструменты, позволяющие специфицировать приложения различных типов (языки спецификаций – документаторы приложений).

Большинство современных СУБД включают в себя несколько языковых средств.

Технические средства БнД включают в себя универсальные, персональные ЭВМ, периферийные средства для ввода/вывода информации. Персональные компьютеры (ПК) делятся на два класса:

1) персональные компьютеры класса *SOHO* (*small office, home office*);

2) рабочие станции.

Под компьютерами *SOHO* подразумевают универсальные машины для использования дома и в офисе. Рабочие же станции предназначены для работы с приложениями большой вычислительной мощности. К рабочим станциям предъявляются более высокие требования, чем к системам *SOHO*.

Организационно-методические инструменты включают в себя методические материалы для пользователей различных категорий.

Администратор базы данных (АБД) – это специалист(ы), который отвечает за управление техническими аспектами системы управления базы данных, разработку программ, определение данных, ввод данных в БД, удаление данных, поддержание целостности и защиты данных. Кроме того, АБД участвует в обучении пользователей различных категорий, планировании и разработке БД. АБД необходимо знать конкретную СУБД и системное окружение.

Классификация БнД. Центральной компонентой БнД является база данных, и большинство квалификационных [8] признаков относится именно к ней. Классификация выполняется по следующим признакам (рис. 1.2):

1. По форме представления информации различают:
– видеосистемы;
– аудиосистемы;
– мультимедийные системы.

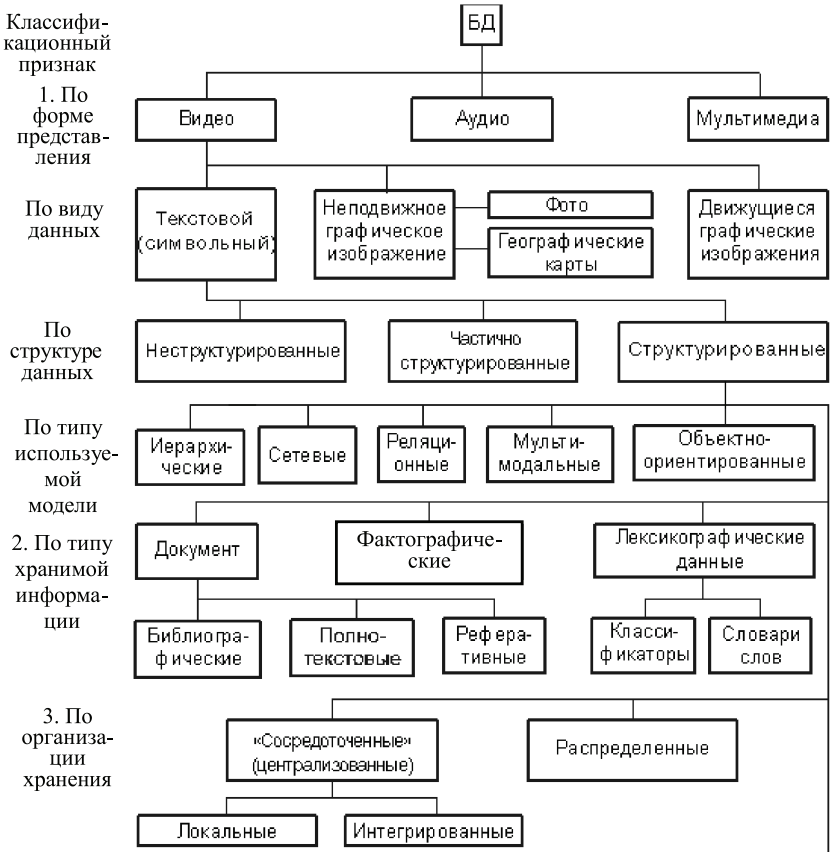


Рис. 1.2. Классификация банков данных

Такая классификация показывает, какая разновидность информации выдается пользователям из БД: изобразительная, звуковая или с использованием комбинации различных форм отображения. Наибольшее практическое использование в настоящее время находят БД, содержащие обычные символьные данные.

Информацию, представленную в символьном виде по **характеру организации можно** разделить на:

– неструктурированные (семантические сети);

– частично структурированные (обычный текст или гипертекст);

– структурированные.

Для структурированных БД требуется предварительное проектирование и описание структуры БД, после чего базы данных этого типа заполняются данными. В свою очередь структурированные БД по **типу используемой модели** подразделяются на:

– иерархические (ИМД);

– сетевые (СМД);

– реляционные (РМД);

– объектно-ориентированные;

– мультимодальные.

ИМД представляются в виде дерева, которое состоит из объектов различных уровней. На верхнем уровне располагается один объект (родитель). На следующем уровне находятся объекты-потомки и т.д. Например, ИМД является файловая система, которая состоит из корневой директории, имеющей иерархию поддиректорий и файлов. В такой модели запрос, который направлен вниз по иерархии, прост в реализации; а вверх направленный запрос, трудно исполним.

СМД подобны иерархическим, исключая того, что в них содержатся указатели, работающие в обоих направлениях, соединяющих родственную информацию. Несмотря на то, что эта модель данных решает часть проблем, связанных с иерархической моделью, но осуществление простых запросов является достаточно непростым процессом.

В основе реляционной базы данных лежит понятие «отношения». Отношение представляется двумерной таблицей, имеющей свое собственное название. В РМД минимальным объектом действий, который сохраняет ее структуру, является строка, состоящая из ячеек

таблицы – полей. С точки зрения логики РМД представляется совокупностью двумерных таблиц различного наполнения.

Объектно-ориентированная база данных (ООБД) – это БД, в которой данные моделируются в виде объектов, их атрибутов, методов и классов. ООБД обычно рекомендованы для применения в случаях, когда необходима высокопроизводительная обработка данных, которые имеют сложную структуру.

Модальность – это присущая человеку форма воздействия на другого человека или компьютер с помощью речи, жестов, прикосновений и т.п. Модальные модели – это формальные модели, используемые для анализа модальностей.

2. По типу хранимой информации:

– документальные (объект хранения – документ);

– фактографические (хранятся сведения об объектах предметной области);

– лексикографические (содержатся многоязычные словари, классификаторы, словари основ слов и т.п.).

Среди документальных баз данных различают полнотекстовые, реферативные и библиографические. БД такого типа могут быть поразному организованы: с хранением и без хранения исходного документа на машинных носителях. К системам второго типа относятся реферативные и библиографические БД, и БД-указатели, отсылающие к источнику информации. Системы, предусматривающие хранение полного текста документа, называются полнотекстовыми. Целью поиска в системах документального типа является не только определенная информация, хранимая в документах, но и сами документы. В качестве признаков поиска могут быть любые «выходные данные» документов, к примеру: «дата принятия документа».

В фактографических БД хранится информация об объектах предметной области в виде «фактов» (например, данные об объеме выпуска продукции производителями и т.п.).

3. По виду организации хранения информации и обращению к ней выделяют:

– локальные или персональные;

– интегрированные (централизованные);

– распределенные.

Персональная база данных предназначена для использования одним пользователем.

Интегрированные и распределенные БД позволяют одновременно обращаться нескольким пользователям к одной и той же информации (режимы доступа – многопользовательский, параллельный). Это вносит специфические проблемы при проектировании и эксплуатации БнД. Кроме того, р**аспределенные** БД (РБД), имеют особенности, которые связаны с тем, что разные части БД физически расположены на разных ЭВМ, а логически, они представляют собой единое целое. Создание РБД было вызвано необходимостью с одной стороны – интеграцией, а с другой – децентрализацией обработки данных.

Интеграция обеспечивает централизованное управление и ведение баз данных. Децентрализация подразумевает хранение данных в местах их возникновения или обработки, при этом увеличивается скорость обработки данных и степень надежности системы, а стоимость – снижается,.

4. По охвату предметной области различают БД:

– территориальные (всемирная, страна, город, регион и т.п.);

– временные (год, месяц и т.п.);

– ведомственные;

– проблемные.

В литературе наблюдаются и другие способы классификации банков данных, но обозначенные являются наиболее значимыми.

Контрольные вопросы

1. Объясните смысл следующих терминов и приведите их примеры: данные, информация, информационная система, предметная область, информационное обеспечение систем управления, СУБД, база данных.

2. Опишите компонентный состав БнД и назначение его составляющих.

3. Охарактеризуйте понятие «метаданные». Назовите основные виды метаданных.

4. Дайте характеристику техническим средствам БнД. Какие известны классы персональных компьютеров?

5. Опишите основные категории языковых средств СУБД.

6. Кого называют администраторами банка данных?

7. Перечислите функции АБД.

8. Перечислите классификационные признаки БнД.

2. МОДЕЛЬ ДАННЫХ

План темы

1. Типы и структуры данных
2. Модели данных

Ключевые слова: *простой и сложный тип данных, структура данных, модель данных.*

2.1. Типы и структуры данных

Понятие «данное» можно определить как триплет, состоящий из имени, типа и значения.

Имя данного выделяет его из общей совокупности данных, придавая ему уникальность. *Тип* показывает принадлежность данного к определенной их разновидности с одинаковыми свойствами. *Значение* определяет его (на числовой или временной оси).

В языках программирования данные, используемые для решения практических задач, разделяются на различные типы, при этом понятие тип данных связано с физическим представлением данных и со способом их обработки.

Структурный аспект данных. Под структурой данных понимается тот или иной способ их объединения.

Определение. **Структура данных – это множество элементов данных, объединенных и упорядоченных одним из принятых способов.**

Данные по структуре могут быть разделены на два типа (рис. 2.1):

- основной (простой) тип, форма представления которого определяется архитектурой ЭВМ. Данные простого типа это – числа, символы и другие элементы, дальнейшее дробление которых не имеет смысл;
- сложный тип, конструируемый пользователем для решения конкретных задач (**сложные данные** – это данные, которые могут быть разделены на более простые типы).

Из элементарных данных формируются сложные типы или структуры данных.

В свою очередь структуры данных делятся на статические и динамические. Статические структуры занимают в памяти ЭВМ постоянный объем. Динамические структуры могут изменять свою длину. Память под них отводится во время выполнения программ, а после их использования – память освобождается.

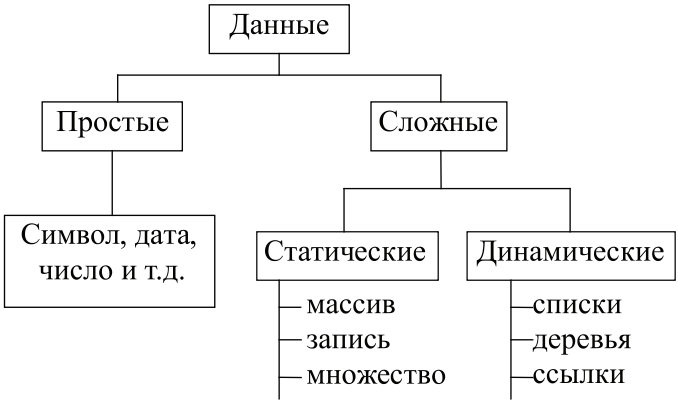


Рис. 2.1. Структура данных

К статическим структурам относятся массив, запись, множество.

Массив – совокупность однотипных элементов, обращение к которым осуществляется по индексу массива.

Запись – совокупность элементов различных типов.

Множество – поименованная совокупность переменных, как правило, одного типа. Порядок расположения элементов в нем не играет роли.

К динамическим структурам относятся дерево, список, ссылка и т.д.

Список – поименованный набор элементов одного типа с одинаковыми или различными размерами элементов. Например, перечень дней недели, перечень фамилий и т.д. Размер списка является величиной переменной, то есть элементы могут добавляться в список и изыматься из него.

Стек – это упорядоченный список, в который элементы добавляются и изымаются с одного конца. Стек – это список типа «последний пришел – первым вышел».

Дерево – это направленный граф, обладающий следующими свойствами: только один узел не имеет дуг, входящих в него; В каждый узел можно попасть за конечное число шагов.

Сложные данные (структуры) в свою очередь могут быть элементами более сложных данных – обобщенных структур (моделей данных – МД). Обобщенные структуры (модели данных) отражают взгляды пользователей на ту или иную предметную область.

2.2. Модели данных

Развитие БД связано с всё большим и большим отстранения ПО от физических структур данных, при работе с которыми необходимо точно знать, где расположены данные, применять для обработки данных довольно сложные, трудно оптимизируемые и модифицируемые алгоритмы.

Модель данных, лежащая в основе БД и отражающая представление пользователя о данных реального мира, содержит три компоненты:

во-первых: ***структура данных*** описывающая с точки зрения пользователя представление данных, т.е. пользователь определяет объекты и связи между ними в предметной области;

во-вторых: ***набор допустимых операций***, которые выполняются на структуре данных, то есть предполагается наличие языков описания и манипулирования данными, который включает в себя операции по извлечению и редактированию данных;

в-третьих: ***ограничения целостности***, т.е. механизма осуществления соответствия на основе формально описанных правил данных предметной области.

В историческом развитии СУБД применялись следующие модели данных:

- иерархическая (структура данных – дерево);
- сетевая (структура данных – граф);
- реляционная (структура данных отношения или таблицы).

Эта классификация весьма условна, каждая СУБД поддерживает оригинальную модель данных.

Выводы (в чем состоит отличие типов данных от моделей данных): в модели данных проектирование структуры данных не обусловлено точным знанием алгоритмов обработки. В языках программирования сначала выбирается метод решения задачи, а только потом определяется, как представить данные для данной задачи. В модели данных используется **принцип независимости данных**, т.е. возможность их использования различными программами.

Контрольные вопросы

1. Охарактеризуйте понятие «данное».
2. Перечислите различные виды данных.
3. В чем заключаются основные различия данных по структуре?
4. Охарактеризуйте следующие свойства данных: наименование, тип, значение.
5. Дайте характеристику каждой разновидности данных в содержательном аспекте.
6. Что собой представляет «модель данных (МД)»?
7. Какие составляющие различаются в МД?
8. Приведите классификацию МД.
9. Что представляет собой независимость данных?

3. КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ДАННЫХ

План темы

1. Назначение концептуальной модели данных.
2. Модель «сущность – связь» (*ER*-модель).
3. Нотации Баркера и *IDEF1X*.

Ключевые слова: *концептуальная и инфологическая модели данных, ER- модель, атрибут, связь, сущность.*

3.1. Назначение концептуальной модели данных

Концептуальная модель (КМ) – это те образы и представления, которые играют роль обобщенных схем поведения.

Концептуальная модель данных (КМД) – **это общая информационная модель предметной области, охватывающая вопросы классификации, структуризации и семантической целостности (достоверности и согласованности данных).**

Концептуальная модель данных разрабатывается **независимо от ограничений,** вытекающих из моделей данных, поддерживаемая той или иной СУБД.

Прежде чем начать проектировать БД [3, 12, 13], необходимо разобраться, каким образом осуществляется функционирование предметной области, для отображения которой создается БД, то есть создается ее предварительное описание. Для описания концептуальной модели может быть использован естественный язык, но такое описание будет громоздким и неоднозначным, поэтому для описания КМ чаще всего используется формализованный язык.

Инфологической моделью (ИЛМ) является описание предметной области, которое выполняется с использованием специальных языковых средств, и не зависит от применяемых программных средств.

К инфологической модели данных предъявляются такие требования как:

1) описание должно быть адекватно предметной области;

2) описание должно быть непротиворечиво, т.е. представлять единое интегрированное описание предметной области,

отображающее взгляды всех пользователей на данную предметную область;

3) описание не должно допускать неоднозначности трактовки модели;

4) инфологическая модель должна быть конечна, т.е. должны быть четко определены границы предметной области, хотя реальный мир, который отображается в ИЛМ, бесконечен;

5) ИЛМ должна обладать свойством расширяемости. Несмотря на конечность модели в нее часто проходится вводить новые объекты. К тому же ИЛМ должна позволять возможности ввода в нее новых объектов без изменения ранее определенных. Тоже же необходимо сказать об удалении объектов.

Язык ИЛМ должен быть применим как при ручном исполнении модели, так и при машинном исполнении [10], т.е. должен обрабатываться ЭВМ, должен иметь средства тестирования ИЛМ.

Составлять инфологическую модель должен специалист, работающий в данной предметной области. Если это невозможно выполнить, то необходимо, чтобы он, хотя бы мог осуществить проверку сделанному описанию, чтобы убедиться, что особенности предметной области восприняты, верно.

К компонентам инфологической модели данных относятся:

1) содержание описания объектов и связей между ними (*модель «Сущность-связь»*);

2) описание информационных потребностей пользователей;

3) описание алгоритмических связей показателей;

4) осуществление ограничения целостности.

Центральным компонентом ИЛМ является модель «Сущность-связь».

Тип запроса, объемно-частотные характеристики, режим использования данных определяют описание информационных потребностей пользователей. А описание алгоритмических связей показателей отражает те показатели, которые служат исходными для вычисления других данных. Обычно для этой цели используются графы взаимосвязи показателей. Ограничения целостности – правила, определяющие особенности функционирования предметной области, нарушение которых недопустимо.

3.2. Модель «сущность – связь»

Перед тем, как приступить к созданию автоматизированной системы обработки информации, необходимо формирование понятий о событиях, фактах и предметах, которыми будет оперировать разрабатываемая система. Для того чтобы привести эти понятия к определенной модели данных, необходимо их заменить соответствующими информационными представлениями. Наиболее удобным инструментом такого представления данных, вне зависимости от реализующего его ПО, является модель «сущность-связь» (*entity – relationship model, ER-model*), основывающаяся на семантической информации о реальном мире и предназначающаяся для логического представления данных.

ER-модель определяет значения одних данных в зависимости от других данных. Важно, что из **модели «сущность – связь»** можно получить все существующие **модели данных** (иерархическую, сетевую, реляционную и объектную).

При этом необходимо отметить, что **модель «сущность – связь» не является моделью данных, так как не определяет операций над данными и ограничивается описанием только их логической структуры.**

Модель «сущность – связь» предложил Питер Пин-Шэн Чен (*Chen*) в 1976 г. В дальнейшем многие авторы предложили свои варианты подобной модели (нотации Мартина, *IDEF1X*, Баркена и др.)..

Далее будет использован некоторый гибрид нотаций Чена (сущности, связи и атрибуты) и Мартина (степени и кардинальности связей). В табл. 3.1 приведен перечень используемых в этом случае обозначений.

Любой фрагмент предметной области может быть представлен как *множество объектов* (*сущностей*), между которыми существует некоторое *множество связей* (*отношений*).

Сущность *entity* (класс объектов) – это совокупность объектов, обладающих одинаковым набором свойств.

Каждая сущность обладает следующими свойствами:

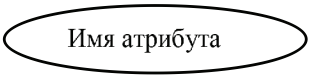
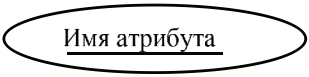
- сущность имеет уникальное имя, к одному имени применяется одна и та же интерпретация, которая не может применяться

к различным именам, кроме случаев, когда они являются псевдонимами/синонимами;

- сущность обладает атрибутами (одним или несколькими), которые или принадлежат сущности, или наследуются через связь;
- сущность обладает атрибутами (одним или несколькими), которые однозначно определяют каждый экземпляр сущности;
- сущность может обладать любым числом связей с другими сущностями модели.

Таблица 3.1

Графические обозначения

Обозначение	Значение
	Независимая сущность
	Зависимая сущность
	Атрибут
	Ключевой атрибут
	Связь

Фактически сущность представляет совокупность **атрибутов**, описывающих свойства каждого члена данного набора сущностей.

Атрибут – это любая, значимая для предметной области характеристика сущности, и предназначенная для идентификации, классификации и количественной характеристики либо выражения состояния сущности.

Пример. Если в качестве предметной области рассматривать вуз, то в ней можно выделить следующие сущности: *обучаемые, преподаватели, сотрудники, аудитории, дисциплины* и т.п. Для объекта класса ЛИЧНОСТЬ В ВУЗЕ таким набором свойств могут быть *год рождения, образование, ученая степень, иностранный язык, адрес и т.п.*

При построении ИМ можно использовать язык *ER*-диаграмм, но чаще применяется более содержательный язык инфологического моделирования (ЯИМ), который представляет сущности предложениями следующего вида:

СУЩНОСТЬ (атрибут 1, атрибут 2, ..., атрибут *n*).

Пример:

ЛИЧНОСТЬ (**табельный номер**, фамилия, имя, отчество, возраст).

Атрибут может быть описательным (дескриптором сущности) или входить в состав первичного ключа (уникального идентификатора).

Ключ сущности (уникальный идентификатор) –атрибут или множество атрибутов и/или связей, который предназначен для однозначной идентификации каждого экземпляра данного типа сущности.

В системах обработки данных используется следующая гипотеза, а именно: каждый объект в наборе объектов различим (т.е. его описание отличается от описания других объектов). В соответствии с этой гипотезой каждая сущность имеет ключ.

В рассмотренном выше примере атрибут *табельный номер* является ключом сущности ЛИЧНОСТЬ В ВУЗЕ (конечно если все табельные номера в вузе уникальны).

Для идентификации каждый экземпляр сущности полностью идентифицируется своим собственным ключевым атрибутом, иначе в его идентификации участвуют и атрибуты другой родительской сущности.

Каждый атрибут описывается уникальным именем, представляемым грамматическим оборотом некоторого существительного, который описывает характеристику представляемую атрибутом.

У одного и того же набора объектов может оказаться несколько ключей. Один из них назначают первичным ключом набора объектов, и служит в дальнейшем ключом соответствующего этому набору файла (его еще называют ключом записи, кортежа). Все остальные ключи набора в этом случае называют возможными ключами.

Кроме таких понятий, как «ключ», «возможный ключ». «первичный ключ» имеется также понятие «составной ключ», которое применяется для ключа, состоящего из нескольких атрибутов.

Диаграмма «Сущность – атрибут». Связь между объектом и его свойством может быть различной. Рассмотрим классификацию связей между объектом и его свойствами (между сущностью и атрибутами):

1) по количеству принимаемых свойством значений связь считается:

– единичной;

– множественной.

Объект может иметь только одно значение какого-то свойства. Например, для объекта класса ЛИЧНОСТЬ таким свойством является *дата рождения*.

Свойство у объекта одновременно может принимать несколько значений. Например, при описании ЛИЧНОСТИ фиксируется свойство *иностранный язык*. Сотрудник (личность) может знать несколько иностранных языков.

2) по оценке постоянства связи, между сущностью и атрибутом во времени связь считается:

– статической (свойство с течением времени не изменяется, например: *дата рождения*);

– динамической (свойство может измениться со временем, например: *ученая степень*).

3) по оценке ассоциаций между объектом и его свойством:

– простые (неделимые);

– условные (свойство присуще не всем экземплярам сущности, например, свойство *ученая степень* для ЛИЧНОСТИ в вузе);

– составные (например, свойство «адрес» состоит из *города*, *улицы*, *дома*, *квартиры*).

Для обозначения связи между объектом и его свойством введены графические обозначения, приведенные в табл. 3.2.

Таблица 3.2

Графические обозначения связи между объектом и его свойством

Обозначение связи	Характер связи
→	единичная связь
→→	множественная связь
→	статическая связь
D →	динамическая связь
– →	условная связь

Пример. Рассмотрим диаграмму «сущность – атрибуты» для предметной области «ВУЗ» (рис. 3.1).

Рис. 3.1. Диаграмма «сущность – атрибуты»

Данная диаграмма дает словесную интерпретацию сущности ЛИЧНОСТЬ через ее свойства. Сущность обозначается именем существительным в единственном числе. Сущность обладает

статическими единичными свойствами: *фамилия, пол*; динамическими множественными свойствами: *образование, иностранный язык*; динамическими единичными свойствами: *ученая степень, адрес*. *В* свою очередь свойство *ученая степень* является условным, а свойство *адрес* – составным.

Связи между объектами предметной области. Связь (от англ. *relationship*) – характер взаимосвязи (ассоциация), установленный между несколькими сущностями.

Набор связей (от англ. *relationship set*) – отношение между n (причем $n => 2$) сущностями, относящимися к некоторому набору сущностей.

В случае, когда $n = 2$ связь называется бинарной, n-арный набор связей ($n > 2$) можно заменить совокупностью бинарных, но n-арные связи отображают точнее семантику предметной области.

Связь обладает *именем*, *степенью* и *классом принадлежности*.

В методике проектирования данных есть правило, согласно которому связи обозначаются глагольными формами.

Пример. Если в качестве предметной области рассматривать предприятие, то в ней можно выделить следующие сущности: СОТРУДНИК, ОТДЕЛ и т.п.

Рассмотрим существующие связи между указанными сущностями:

- так как каждый отдельный сотрудник работает в каком-либо отделе, то между сущностями СОТРУДНИК и ОТДЕЛ имеется связь «работает в» или ОТДЕЛ – РАБОТНИК;
- поскольку один из работников является руководителем отдела, то между упомянутыми сущностями СОТРУДНИК и ОТДЕЛ существует связь «руководит» или ОТДЕЛ – РУКОВОДИТЕЛЬ.

Число ассоциированных сущностей через набор связей с другой сущностью, получило название **степени связи**.

Могут существовать следующие степени бинарных связей:

1. Связь «один к одному» (1:1) представляет тот тип связи, когда один экземпляр объекта, от которого направлена связь, идентифицирует один и только один экземпляр объекта, к которому направлена связь, и наоборот.

Данный тип связи представлен на рис. 3.2, где прямоугольники обозначают сущности, а ромб – связь в соответствии с обозначениями принятыми в табл. 3.1. Поскольку степени связи для каждой сущности равны 1, то они соединены одной линией.

Рис. 3.2. Связь 1:1

Другой характеристикой связи кроме ее степени является **класс принадлежности** сущностей входящих в нее или **кардинальность** связи.

Класс принадлежности характеризует обязательный или необязательный характер связи между экземплярами двух сущностей.

Раз в каждом отделе должен быть руководитель, то сущности «ОТДЕЛ» обязательно должна соответствовать сущность «СОТРУДНИК». Но не каждый сотрудник руководитель отдела, поэтому в такой связи не каждая сущность «СОТРУДНИК» имеет ассоциированную с ней сущность «ОТДЕЛ», таким образом, не каждый экземпляр сущности принимает участие в этой связи.

Говорят, что сущность «СОТРУДНИК» обладает *обязательным классом принадлежности* (такой факт обозначается указанием интервала возможных вхождений сущности в связь, это в данном случае 1,1), а сущность «ОТДЕЛ» обладает *необязательным классом принадлежности* (0,1). Теперь данная связь описывается как 0,1:1,1. В дальнейшем кардинальность бинарных связей степени 1 обозначается, как показано на рис. 3.3.

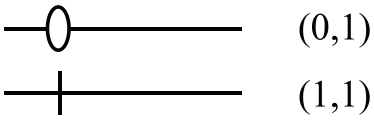


Рис. 3.3. Обозначение принадлежности

2. Связь «один ко многим» (1:M) представляет тот тип связи, когда один экземпляр объекта, от которого направлена связь, идентифицирует некоторое число (0, 1, …) экземпляров объектов данных, к которым направлена связь. А в обратном направлении эта связь идентифицирует только один объект.

Такая связь характерна для связи ОТДЕЛ – СОТРУДНИК, поскольку в отделе может работать любое число сотрудников, но каждый сотрудник может работать только в одном отделе. При этом не каждый отдел (например, вновь образованный) должен иметь хотя бы одного сотрудника.

Степень связи M графически отображается «вороньей лапкой» («древообразной» линией), так это сделано на рис. 3.4. В таком случае необходимо также учитывать класс принадлежности сущностей.

Кардинальность бинарных связей степени M обозначается, как представлено на рис. 3.5.

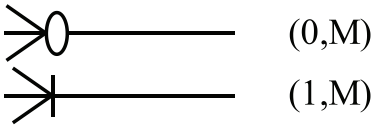


Рис. 3.5. Обозначение кардинальности бинарных связей степени *n*

3. Связь M:1 («много к одному»).

Пример. Предположим, рассматриваемое ранее предприятие действует на основании системы контрактов, которые заключаются с заказчиками. Такой факт изображается в модели «сущность – связь» связью КОНТРАКТ – ЗАКАЗЧИК, который объединяет сущности КОНТРАКТ(НОМЕР, СРОК ИСПОЛНЕНИЯ, СУММА) и ЗАКАЗЧИК(НАИМЕНОВАНИЕ, АДРЕС).

С одним заказчиком может быть заключено любое число контрактов, поэтому связью КОНТРАКТ – ЗАКАЗЧИК будет степень M:1 (рис. 3.6). В таком случае, раз каждый контракт заключается с конкретным заказчиком, а каждый заказчик обладает хотя бы одним контрактом, то каждая сущность характеризуется обязательным классом принадлежности.

Рис. 3.6. Пример связи M:1

4. Связь M:M («много ко многим») представляет тот тип связи, когда один экземпляр объекта, от которого направлена связь, идентифицирует некоторое число объектов, которому направлена связь, и наоборот. То есть связь является неуникальной в обоих направлениях.

Пусть в предыдущем примере для выполнения каждого контракта создана рабочая группа, в которую включены сотрудники из разных отделов. Так как каждый сотрудник может включаться в любое число рабочих групп, и каждая группа должна иметь не меньше, чем одного сотрудника, тогда связь между сущностями СОТРУДНИК и РАБОЧАЯ_ГРУППА характеризуется степенью M:M (рис. 3.7).

Рис. 3.7. Пример связи M:M

При условии зависимости существования сущности *X* от сущности *Y,* сущность *X* называется **зависимой** (при этом сущность *X* называют «слабой», а сущность *Y* – «сильной»). *В* качестве примера рассмотрена связь между описанными ранее сущностями РАБОЧАЯ ГРУППА и КОНТРАКТ. Рабочая группа создается после того, как подписывается контракт с заказчиком, и прекращает свое существование по выполнению контракта. Поэтому, сущность РАБОЧАЯ ГРУППА зависима от сущности КОНТРАКТ. Зависимая сущность обозначается двойным прямоугольником, а ее связь с сильной сущностью – линией со стрелкой (рис. 3.8):

Рис. 3.8. Пример обозначения «слабой» и «сильной» сущностей

Кардинальность связи для сильной сущности всегда будет (1,1). Для зависимой сущности классы принадлежности и степени связи могут быть любыми. Например, рассматриваемое предприятие использует несколько банковских кредитов, которые могут быть представлены набором сущностей КРЕДИТ (НОМЕР ДОГОВОРА, СУММА, СРОК ПОГАШЕНИЯ, БАНК). По каждому кредиту осуществляются платежи в счет его погашения и выплаты процентов. Такой факт характеризуется набором сущностей ПЛАТЕЖ (ДАТА, СУММА) и набором связей «осуществляется по». В случае, если поступление запланированного кредита срывается, информацию о нем необходимо удалить из БД. Соответственно, нужно удалить все сведения о плановых платежах по этому кредиту. Поэтому, сущность ПЛАТЕЖ зависима от сущности КРЕДИТ (рис. 3.9).

Рис. 3.9. Пример обозначения «слабой» и «сильной» сущностей

На языке инфологического моделирования связь (ассоциация) между информационными объектами представляется предложением вида:

АССОЦИАЦИЯ[СУЩНОСТЬ *S*1, СУЩНОСТЬ *S*2](атрибут 1, атрибут 2),

где *S*1, *S*2 – степени связи, а атрибут 1 или 2 – атрибуты, входящие в состав ключа сущности.

Пример представления описания связи между сущностями КРЕДИТ и ПЛАТЕЖ.

Осуществляется по [Кредит 1, Платеж М] (Номер кредита, Номер платежа).

Разновидности объектов. Выше были рассмотрены объекты, не вникая в их сложность. На самом деле различают несколько разновидностей объектов (рис. 3.10).

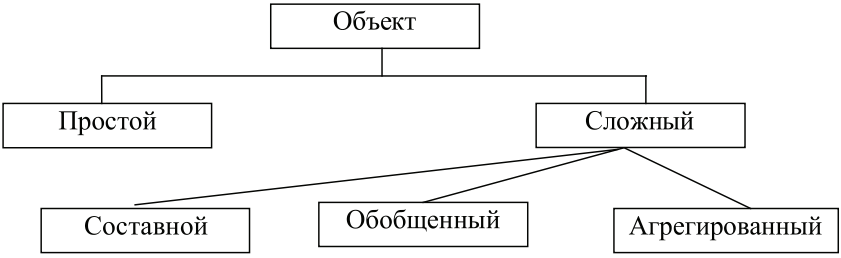


Рис. 3.10. Классификация объектов

Объект называется простым, если он является неделимым.

Сложный объект представляет собой объединение других объектов.

Понятия простой и сложный объекты являются относительными понятиями. Например, *стул* в системе учета материальных ценностях является простым объектом, а в мебельном производстве – сложным объектом.

Выделяют несколько разновидностей сложных объектов: составной, обобщенный, агрегированный.

Составной объект представляет собой связь «целое – часть».

Для отображения составного объекта в инфологической модели не используются никакие специальные обозначения. Характер связи между элементами составного объекта может быть 1:М, М:М. Примеры:

УЗЛЫ – ДЕТАЛИ, ГРУППА – СТУДЕНТ.

Обобщенный объект выражает присутствие связи «род – вид» между некоторыми объектами данной предметной области.

Связь «род – вид» характеризует собой классификацию ряда объектов предметной области по неким признакам. Объекты, которые входят в состав обобщенного объекта являются

его категориями. Такой атрибут «категория» называется дискриминантом.

Пример. Предметная область – ВУЗ. В данной предметной области обобщенным объектом является сущность ЛИЧНОСТЬ (рис. 3.11).

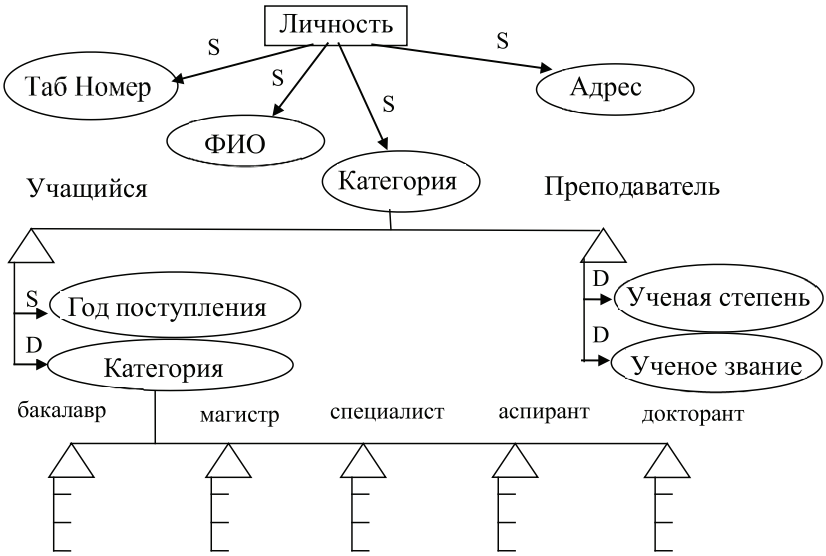


Рис. 3.11. Пример обобщенного объекта ЛИЧНОСТЬ В ВУЗЕ

Классификация внутри обобщенного объекта может быть многоуровневой. Для обозначения подкласса внутри обобщенного объекта используется специальное графическое средство – треугольник.

Родовой и видовые объекты обладают определенным набором свойств.

Агрегированные объекты соответствуют некоторому процессу, в который вовлечены другие объекты.

Агрегированный объект обозначается ромбом и именуется существительным отглагольной формы.

Для примера, агрегированный объект ПОСТАВКА (рис. 3.12) соединяет в себе объекты ПОТРЕБИТЕЛЬ, ПОСТАВЩИК, ТОВАР, ДАТА ПРОДАЖИ. Агрегированный объект может иметь и собственные свойства.

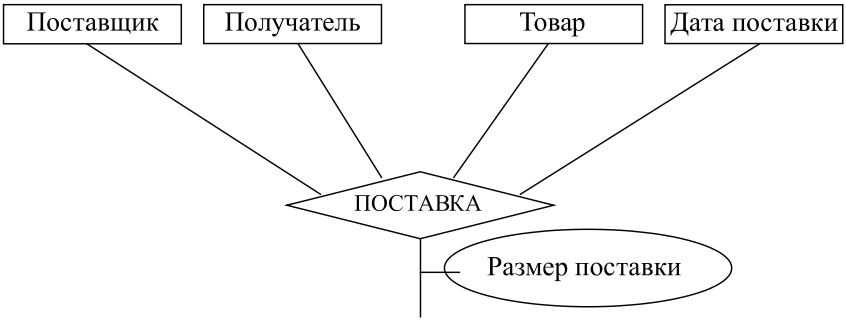


Рис. 3.12. Пример агрегированного объекта

Диаграмма «сущность – связь». Непременным свойством модели «сущность – связь» безусловно, является то, что она может быть выполнена в виде графической схемы. Это значительно упрощает анализ предметной области. Существует несколько разновидностей обозначений элементов диаграммы «сущность – связь», каждая из которых имеет определенные положительные черты. Сущности со связями и атрибуты с сущностями соединяются прямыми линиями. Для указания кардинальностей связей применяются введенные ранее обозначения.

В процессе построения диаграмм выделяются следующие этапы:

1) идентификация сущностей и связей рассматриваемой предметной области;

2) идентификация семантической информации в наборах связей. К примеру, является ли представленный набор связей отображением связи 1:M;

3) определение кардинальностей связей;

4) определение атрибутов и наборов их значений (доменов);

5) организация данных в виде отношений «сущность – связь».

В качестве примера строится диаграмма, отображающая связь данных для подсистемы учета персонала предприятия.

Выделяются сущности и связи. Рассматриваемое предприятие включающие в свой состав отделы с работающими в них сотрудниками. Каждый сотрудник занимает определенную должность (инженер, ведущий инженер, бухгалтер и т.д.). В результате таких рассуждений вводятся следующие наборы сущностей:

ОТДЕЛ (НОМЕР ОТДЕЛА, НАИМЕНОВАНИЕ ОТДЕЛА);

Сотрудник (табельный номер, ФИО сотрудника, адрес РЕГИСТРАЦИИ);

Должность (НАИМЕНОВАНИЕ должности, оклад),

а также набор связей РАБОТАЕТ_В и ЗАНИМАЕТ.

Считаем, что совместительство в организации не разрешено.

Обобщая вышеизложенное, получается диаграмма «сущность – связь», показанная на рис. 3.13.

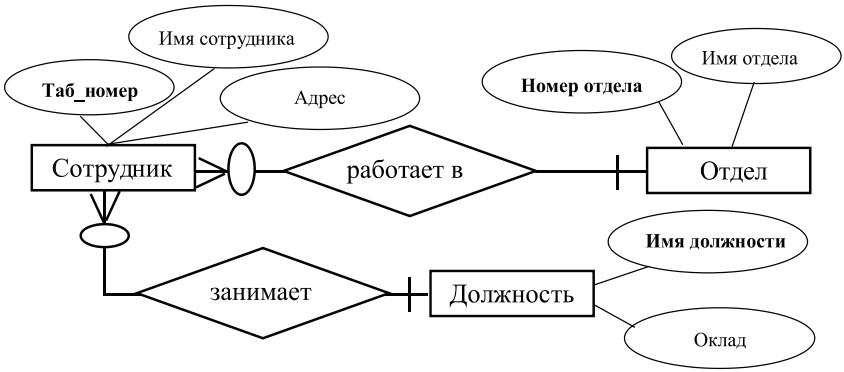


Рис. 3.13. Фрагмент диаграммы учета персонала

ER-модель представляет специфику ограничения целостности данных. В модели предусматриваются следующие ограничения на:

1) допустимые значения – определяют типы данных (численные, символьные и др. значения);

2) допустимые значения для атрибута – диапазон возможных значений (к примеру, возраст сотрудника может быть от 18 до 60 лет);

3) значения данных (например, сумма отчислений не может превышать суммы зарплаты).

3.3. Нотация *IDEF1X*

Метод *IDEF1X,* разработан Т. Рэмей (*T.Ramey*) и основан на подходе П. Чена, позволяет построить модель данных, эквивалентную РМД в ЗНФ. *IDEF1X* реализована с учетом таких требований, как простота изучения и возможность автоматизации. *IDEF1X*-диаграммы используются в ряде распространенных *CASE*-средствах (*ERwin*, *Design/IDEF и* др.).

Сущность в методологии *IDEF1X независима от идентификаторов* или просто независима, при условии, что каждый экземпляр сущности однозначно идентифицируется без определения его отношений с другими сущностями. Сущность является *зависимой от идентификаторов* или просто зависимой, если однозначная идентификация экземпляра сущности зависит от ее отношения к другой сущности (рис. 3.14).

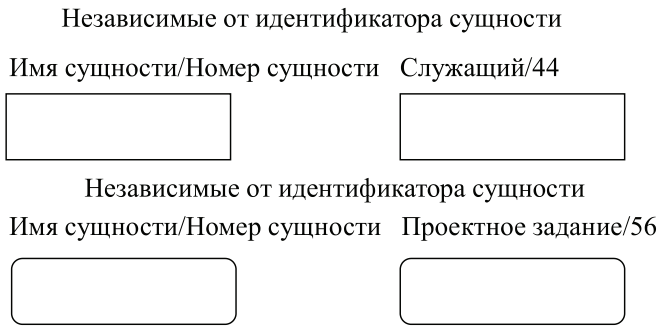


Рис. 3.14. Обозначение сущностей

Каждой сущности присвоены уникальное имя и номер, которые разделены косой чертой «/» и размещены над блоком.

Атрибуты изображаются в виде перечня имен внутри блока сущности. **Атрибуты, однозначно определяющие каждый экземпляр**

сущности, образуют первичный ключ сущности. Атрибуты, которые определяют первичный ключ, размещаются в первой строке перечня и отделяются от других атрибутов горизонтальной чертой (рис. 3.15).

Рис. 3.15. Атрибуты и первичные ключи

Один или несколько атрибутов, значения которых однозначно определяют каждый экземпляр сущности, называются возможным (альтернативным) ключом сущности (рис. 3.16). Каждая сущность должна обладать хотя бы одним возможным ключом.

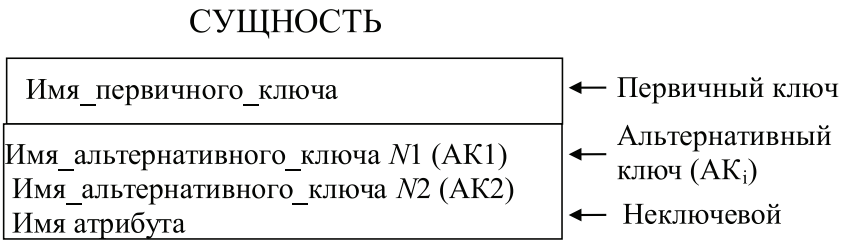


Рис. 3.16. Изображение сущности с альтернативным ключом

Имя атрибута также записывается в единственном числе и должно иметь четкое смысловое значение. Кроме того, оно должно быть уникальным в рамках модели, а не отдельных сущностей.

В дополнении к собственным атрибутам, атрибут может наследоваться сущностью через специфическое отношение. Такие атрибуты называются **внешними ключами (*Foreign Key*)**, которые могут

применяться в качестве части или целого первичного ключа или неключевого атрибута. Внешний ключ помечаются буквами FK в скобках (рис. 3.17).

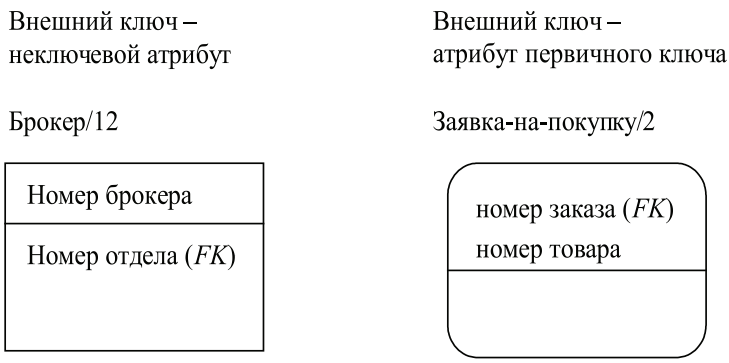


Рис. 3.17. Примеры внешних ключей

Правила атрибутов:

1) атрибут имеет уникальное имя;

2) в сущности может быть любое количество атрибутов;

3) наследуемый атрибут должен являться частью первичного ключа сущности-родителя;

4) любой экземпляр сущности должен содержать конкретное значение для каждого атрибута (правило не обращения в нуль);

5) каждый атрибут сущности может принимать только одно значение (правило неповторяемости).

Типы зависимых сущностей:

1) ассоциативная сущность – сущность, связанная со многими родительскими сущностями;

2) категориальная сущность – дочерняя сущность в иерархии наследования.

Отношение категоризации является отражением того факта, что некоторые реально существующие объекты являются категориями других объектов (классификация «род – вид»). Например, сущности ПОСТОЯННЫЙ СОТРУДНИК и СОВМЕСТИТЕЛЬ – это категории сущности СОТРУДНИК. В *IDEF1X*-модели они связаны между собой через отношение категоризации.

Отношение категоризации – это отношение между двумя и более сущностями, в котором каждый экземпляр общей сущности связан с одним экземпляром одной и только одной сущностью-категорией.

В предыдущем примере СОТРУДНИК – общая сущность, а ПОСТОЯННЫЙ СОТРУДНИК и СОВМЕСТИТЕЛЬ являются сущностями-категориями.

При введении категориальной сущности вводится **дискриминатор**, *атрибут родового предка*. Он определяет характер классификации обобщенного объекта.

Атрибут общей сущности, определяющий, с какими возможными сущностями-категориями он связан, именуется дискриминатором отношения категоризации.

Сущности категории всегда являются взаимоисключающими: экземпляр общей сущности может соответствовать экземпляру только одной сущности-категории.

В примере, показанном на рис. 3.18, дискриминатором является атрибут *тип сотрудника*.

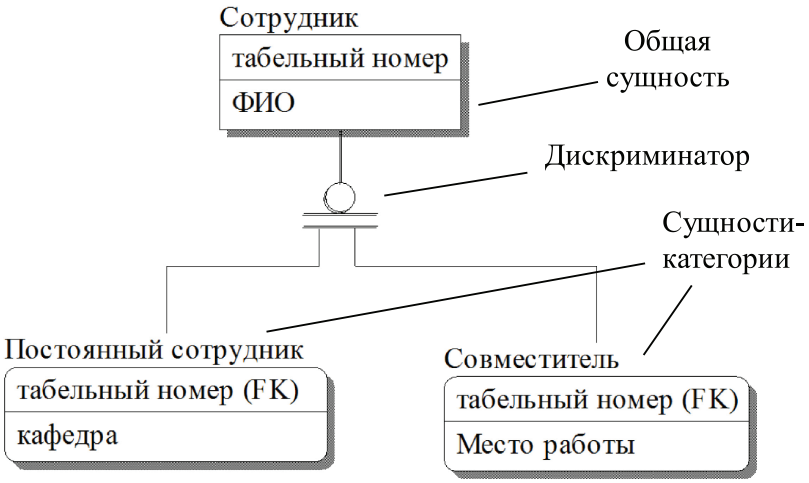


Рис. 3.18. Отношение категоризации

Отношение категоризации не имеет точного названия, но может иметь такую составляющую, как «может быть». Например, СОТРУДНИК может быть СОВМЕСТИТЕЛЕМ.

Правила для отношения категоризации:

1) общая сущность может включать любое количество отношений категоризации.

2) сущность-категория:

- может иметь только одну общую сущность;
- может быть общей сущностью в другом отношении категоризации;

3) все экземпляры сущности-категории имеют одно значение дискриминатора. У разных сущностей-категорий различные значения дискриминатора.

Связь может дополниться указанием степени либо мощности (количество экземпляров сущности-потомка, существующее для экземпляра сущности-родителя). В *IDEF1X* могут быть определены такие **мощности связей**:

- любой экземпляр сущности-родителя может не иметь или иметь один или более связанных с ним экземпляров сущности-потомка;
- любой экземпляр сущности-родителя должен иметь, по крайней мере, одного экземпляра сущности-потомка;
- любой экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

Мощность связи обозначается, как показано на рис. 3.19 (мощность по умолчанию – *N*).

Ноль или один	Один или более	Ноль, один или более
Z	*P*	*N*

Рис. 3.19. Мощность связи

Для экземпляра сущности-потомка однозначно определяемого связью с сущностью-родителем, такая связь именуется *идентифицирующей*, в противном случае – *неидентифицирующей*.

Связь между сущностью-родителем и сущностью-потомком изображается линией с точкой на конце линии у сущности-потомка.

При этом *идентифицирующая связь* между сущностями родителем и потомком выполняется сплошной линией (рис. 3.20). Сущность-потомок в идентифицирующей связи зависима от идентификатора сущностью. А сущность-родитель в идентифицирующей связи может быть как независимой, так и зависимой от идентификатора сущностью (это определяется ее связями с другими сущностями).

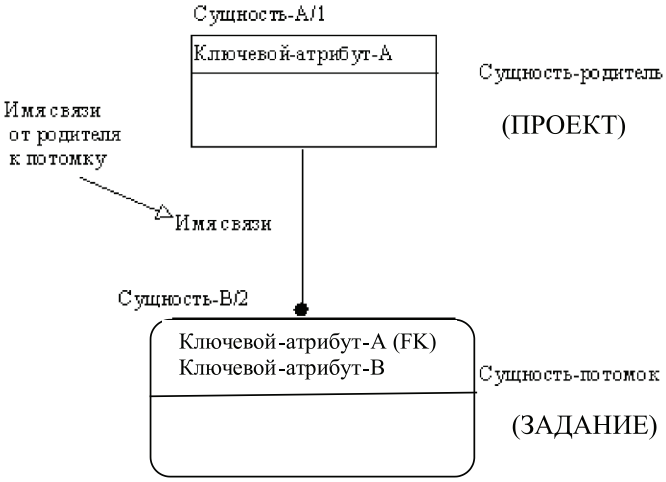


Рис. 3.20. Идентифицирующая связь

Например, при рассмотрении сущностей *ПРОЕКТ и ЗАДАНИЕ* можно отметить следующее. *С проектом связано одно или несколько заданий, которые однозначно идентифицируются в пределах одного проекта.* Следовательно, между сущностями ПРОЕКТ и ЗАДАНИЕ существует идентифицирующее отношение. Ключевым атрибутом *A* является номер проекта, а ключевым атрибутом *B* – номер задания.

Неидентифицирующая связь изображается пунктирной линией (рис. 3.21). Сущность-потомок в неидентифицирующей связи независима от идентификатора, если она не является сущностью-потомком в какой-либо идентифицирующей связи.

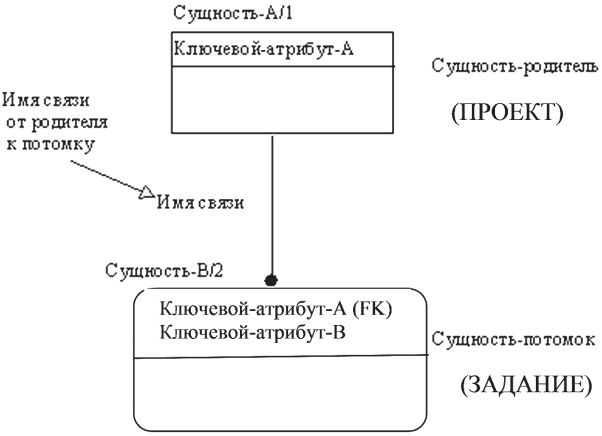


Рис. 3.21. Неидентифицирующая связь

Пример. Для предметной области ПРЕДПРИЯТИЕ при автоматизации учета персонала вводятся сущности: Отдел, Сотрудник, Должность. Модель данных представляется рис. 3.22.

Рис. 3.22. Фрагмент модели данных области ПРЕДПРИЯТИЕ

Выводы. Модель «сущность-связь» относится к семантическим моделям. Семантическое моделирование данных связано со смысловым содержанием данных независимо от их представления в ЭВМ. Семантическая модель данных легко может быть преобразована в сетевую, иерархическую или реляционную модель.

Контрольные вопросы

1. В каких формах может быть представлена предметная область, для которой разрабатывается база данных?
2. Что такое «инфологическая модель (ИЛМ)»?
3. Как в ИЛМ отображаются объекты и их свойства?
4. Как должны задаваться имена объектов в ИЛМ?
5. Какие существуют разновидности связей между объектом и его свойствами?
6. Что представляет собой идентификатор объекта? Каким свойством должен обладать идентификатор объекта?
7. Какие отношения могут быть между связанными информационными объектами?
8. Охарактеризуйте понятие класс принадлежности сущности? Какие бывают классы принадлежности сущностей?
9. Какие объекты называются простыми?
10. Какие объекты называются сложными?
11. Какие различия имеются в изображении ИЛМ в известных вам системах автоматизации проектирования?

4. ДОРЕЛЯЦИОННЫЕ МОДЕЛИ ДАННЫХ

План темы

1. Особенности модели.
2. Структура данных.
3. Операции над данными.
4. Ограничения целостности.

Ключевые слова: *элемент, запись (группа), групповое отношение, дерево, корень, листья, левосписочная структура, операции над данными, цикл, петля, контур.*

4.1. Иерархическая модель данных

Иерархическая модель данных (ИМД) широко применяется на практике. Например, такая структура характерна для каталога папок Windows, с которой легко познакомиться, запустив Проводник.

Ограничимся тремя уровнями (рис. 4.1):

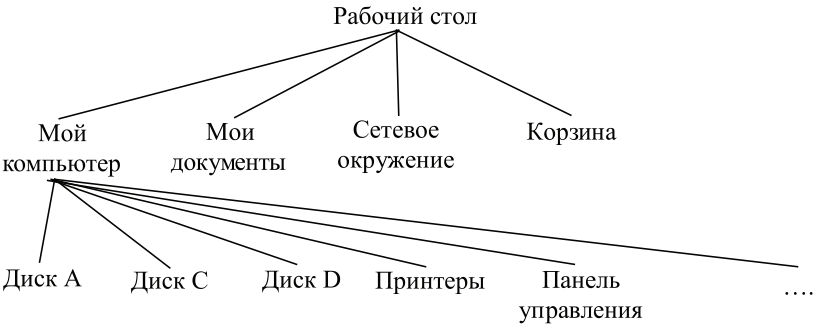


Рис. 4.1. Пример иерархической структуры данных

Примером ИМД является *доменная система имен,* подключенных к Интернету компьютеров.

Структура данных. В ИМД организацию данных принято определять в следующих терминах: *атрибут, запись, групповое отношение, база данных* [9]:

• ***атрибут или элемент данных или поле*** – это наименьшая единица данных. При этом каждому элементу в базах данных дается уникальное имя. По нему обращаются к БД при обработке;

• ***запись*** – это совокупность атрибутов. Применение записей обеспечивает за один запрос к базе данных получение некоторой логически связанной совокупности данных. В процессе функционирования именно записи изменяют, добавляют и удаляют. Состав атрибутов определяет тип записи. Каждый экземпляр записи – это конкретная запись с собственным значением элементов;

• ***групповое отношение*** – иерархическое отношение между отдельными записями двух типов. Запись «Родитель» (владелец группового отношения) называют исходной записью. Записи «Потомки» (члены группового отношения) – подчиненными.

Хранение данных в ИМД имеет вид древовидной структуры.

Дерево может рассматриваться как ориентированный граф, который описывается следующим образом:

а) корень – единственная вершина, в которую не входит ни одно ребро;

б) в остальные вершины входит только одно ребро, а исходит произвольное число ребер (зависимые вершины);

в) в структуре нет циклов.

Зависимые вершины, в которые входит одно ребро и не выходит ни одного, называются листьями.

В дереве (ИМД) вершины представляют собой таблицы описания экземпляров сущностей, дуги (ребра) – описывают связи между сущностями или адресные ссылки.

Пример ИМД. На внутреннем уровне иерархическая БД представляет один файл, который объединяет экземпляры записи одного типа, т.е. дерево в памяти ЭВМ представляется левосписочной структурой, как показано ниже.

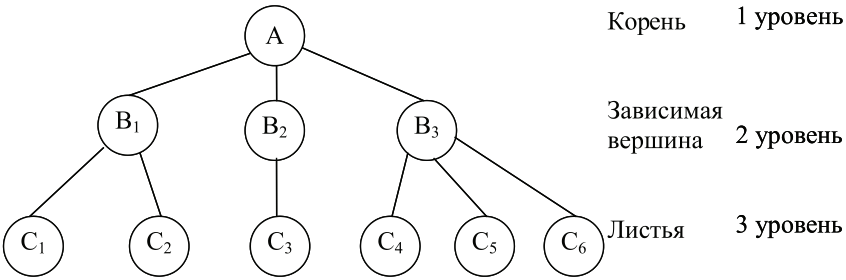


Рис. 4.2. Иерархическая структура данных

файл

$AB_1C_1C_2B_2C_3B_3C_4C_5C_6$	дерево 1 (групповое отношение)
$AB'_1C'_1C'_2B'_2C'_3B'_3C'_4C'_5C'_6$	дерево 2
.....	
$A^nB^n_1C^n_1C^n_2B^n_2C^n_3B^n_3C^n_4C^n_5C^n_6$	дерево n

Корень каждого дерева содержит ключ с уникальным значением в обязательном порядке. Ключи некорневых записей могут иметь уникальное значение только в рамках одного группового отношения. Каждая запись идентифицируется полным сцепленным ключом, под которым понимается совокупность ключей всех записей от корневой записи по иерархическому дереву. Обращение к данным в древовидной структуре происходит через корневую вершину.

Пример. *Предприятие состоит из нескольких отделов, в каждом из которых работают несколько сотрудников, при этом сотрудник не может работать более чем в одном отделе. Для создания информационной системы управления персоналом необходимо иметь групповое отношение (рис. 4.3, а), которое состоит из родительской записи ОТДЕЛ (Наименование отдела, Число работников) и дочерней записи СОТРУДНИК (Фамилия, Должность, Оклад). Для упрощения полагается, что имеются только две дочерние записи.*

Для автоматизации процесса учета контрактов с заказчиками необходимо создание еще одной иерархической структуры: заказчик – контракты – сотрудники, работающие с ними. Такое дерево (рис. 4.3, б) включает следующие записи:

ЗАКАЗЧИК (Наименование заказчика, Адрес),

КОНТРАКТ (Номер, Дата, Сумма),

ИСПОЛНИТЕЛЬ (Фамилия, Должность, Наименование отдела).

Если необходимо показать, что некоторый сотрудник участвует в выполнении нескольких контрактов, то требуется ввод описания нового дерева $\left(\text{« Исполнитель} \xrightarrow{1: M} \text{Контракт»}\right)$ (рис. 4.3, в).

Таким образом, модель данных включает в себя:

а) групповое отношение ОТДЕЛ – СОТРУДНИК;

б) групповое отношение ЗАКАЗЧИК – КОНТРАКТ – ИСПОЛНИТЕЛЬ;

с) групповое отношение ИСПОЛНИТЕЛЬ – КОНТРАКТ.

Из этого примера видны недостатки иерархических БД:

а) избыточность данных (структуры Сотрудник и Исполнитель дублируют друг друга);

б) если требуется организовать связь M:M, то в этом случае необходимо ввести описание нового дерева $\left(\text{« Исполнитель} \xrightarrow{1: M} \text{Контракт»}\right)$, как показано на рис. 4.3, в, что вынуждает дублировать информацию.

в) усложнение операций удаления и включения записи;

г) доступ к любой вершине осуществляется через корневую вершину, что увеличивает время доступа.

В иерархической модели определены следующие операции:

- **добавление** в базу данных новой записи. Для записи «корень» обязательно необходимо сформировать значения ключа;
- **редактирование** значений данных предварительно извлеченной записи. Ключевые данные не должны подвергаться изменениям;
- **удаление** некоторой записи со всеми подчиненными ей записями;
- **извлечение**:

– корневой записи по ключевому значению (допускается также последовательный просмотр корневых записей);

– следующей записи (следующая запись, в этом случае, извлекается в порядке левостороннего обхода дерева).

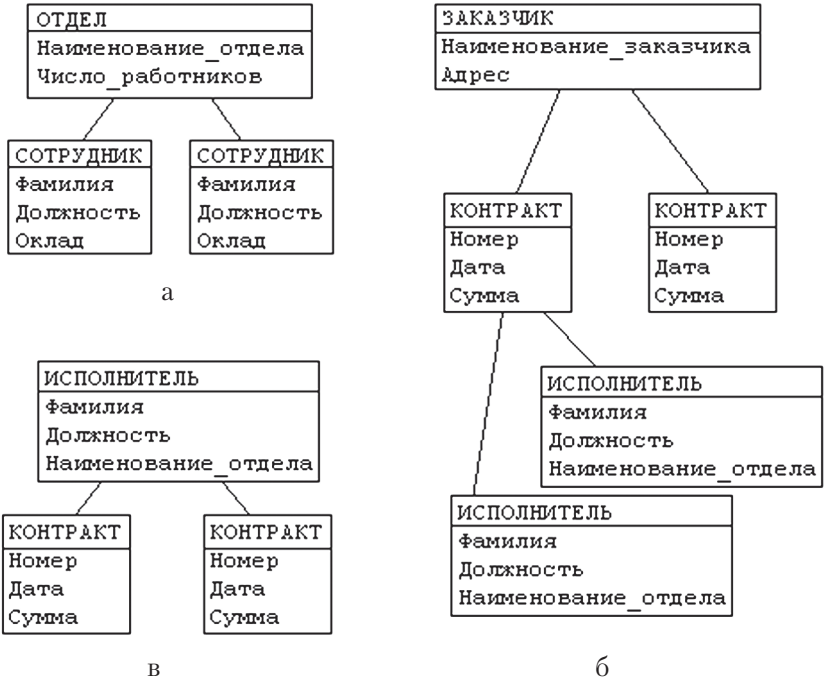


Рис. 4.3. Иерархическая модель данных «Предприятие»

В операции ИЗВЛЕЧЕНИЕ допускается применение условий извлечения (например, выявить сотрудников с определенным окладом).

Таким образом, все операции модификации применяются только к одной «текущей» записи (которая должна быть предварительно извлечена из базы данных). Такой подход к манипулированию данных получил название «навигационного».

Ограничения целостности. В ИМД поддерживается только целостность связей между владельцами и членами группового отношения,

следовательно, никакой потомок не может существовать без родителя. Как уже упоминалось, автоматически не обеспечивается соответствие парных записей, которые входят в различные ИМД.

4.2. Сетевая модель данных

Сетевая модель данных (СМД) определяется в той же терминологии, что и ИМД [9]. Она состоит из некоторой совокупности записей, могущими быть родителями или членами групповых отношений. Связь между записью-родителем и записью-членом также имеет вид 1:*N*.

Основное отличие сравниваемых моделей заключается в том, что в СМД запись может входить ***более чем в одно групповое отношение***. При этом каждое отношение именуется и приводится отличие между его экземпляром и типом. Тип отношения определяется его именем и задает общие свойства для всех экземпляров такого типа. Экземпляр отношения задается записью-родителем и множеством (быть может пустым) записей потомков. При этом выполняется следующее ограничение: ни один экземпляр записи не может являться членом двух экземпляров отношений одного типа (например, сотрудник, может работать только в одном отделе).

СМД имеют более сложные структуры, чем ИМД. Каждая запись СМД может иметь любое число связей и по горизонтали, и по вертикали. Сетевые структуры могут иметь петли и циклы (рис. 4.4).

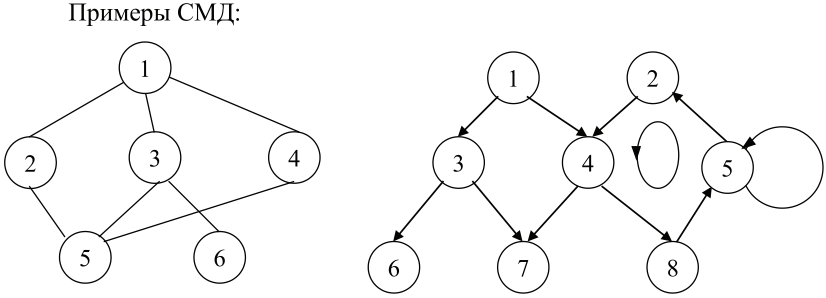


Рис. 4.4. Примеры сетевых структур

Вершинами СМД являются данные (записи); **дугами** – отношения (связи); **циклы** – это замкнутые контура, которые начинаются и заканчиваются у одной и той же записи. **Петли** образуются, когда связи записи указывают на саму запись.

При описании схемы БД для каждого сегмента данных указывается его адрес и семантика отношений с другими сегментами данных. **Доступ к элементам данных в СМД может происходить по любой цепочке.**

Операции над данными:

- **добавление** – в конец файла БД заносится новый сегмент данных с указанием его адреса и с описанием схемы включения в групповое отношение (либо включение ее в отношение, когда она объявляется подчиненной, или не включение);
- **обновление** – изменение значений элементов у предварительно извлеченной из БД записи;
- **извлечение** – нахождение записи по значению ключа и последовательно можно перейти от родителя к записям членам отношений, или от подчиненной записи к родителю;
- **удаление** – исключить запись из БД. Удаляются элементы по их адресу с описанием связи этих элементов с другими элементами данных;
- **исключение из отношения** – разрыв связи между записью-родителем и записью-потомком;
- **включение в отношение** – связь существующей подчиненной записи с записью-родителем;
- **переключение** – связь существующей подчиненной записи с другой записью-родителем в том же отношении.

Ограничения целостности. Обеспечивается сохранение целостности по ссылкам (родитель – член отношения).

Пример. Рассмотрим фрагмент модели данных предприятия (рис. 4.5). При автоматизации учета контрактов вводятся новые сущности: «Заказчик – Контракт», «Сотрудник – Исполнитель».

Примером СБД является Всемирная паутина глобальной компьютерной сети Интернет.

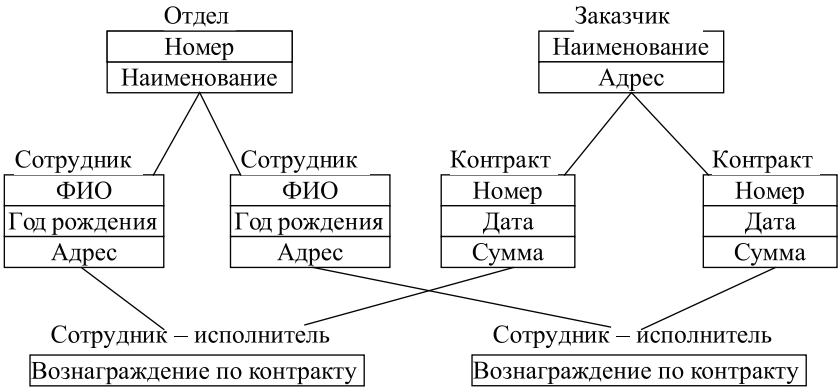


Рис. 4.5. Фрагмент сетевой модели данных

Достоинством СМД является простота описания связи M:M, **недостатком** – сложность описания структуры и выполнения операций.

Контрольные вопросы

1. Каковы структурные элементы иерархической модели данных?
2. Как обеспечивается доступ к ИБД?
3. Какие ограничения целостности хранимых данных должны поддерживаться в иерархической модели данных.
4. Назовите недостатки ИМД.
5. Назовите структурные элементы СМД?
6. Что представляет собой элемент данных, запись в СМД?
7. Как обеспечивается связь между записями в СМД?
8. По каким правилам строится сетевой граф?
9. Какие ограничения целостности хранения данных должны поддерживаться в СМД?

5. РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ

План темы

1. Структура данных.
2. Свойства отношений.
3. Графическая схема отношения.
4. Операции над данными.
5. Механизм поддержания семантики отношений
6. Ограничения целостности.

Ключевые слова: *отношение, кортеж, домен, степень и мощность отношения, ключ, схема отношения, первичные и внешние ключи.*

Реляционную [11] модель данных (РМД) предложил сотрудник компании *IBM* Е.Ф. Кодд в 1970 г. Такая модель данных в настоящее время является практическим стандартом, на который ориентируются фактически все современные коммерческие СУБД.

5.1. Структура данных

В РМД [4, 9, 12, 13] достигается более высокий уровень абстракции данных, по сравнению с ИМД или СМД. Это достигается в результате применения теории отношений (слово «реляционная» происходит от *relation*(английское) – «отношение»).

Особенностью модели Кодда является то, что все данные представляются в виде таблицы. которая по сути дела является последовательным файлом. Строки таблицы являются записями, то есть описывают свойства отдельных экземпляров объекта, а столбцы описывают поля записи.

Таким образом, структура данных представляет собой некоторое отношение между элементами данных и может быть представлена следующей записью (рис. 5.1):

$$R(a, b, c, ..., x),$$

где R – имя отношения, $a, b, c, ..., x$ – атрибуты отношения, a_i, b_i, c_i, ..., x_i – кортеж отношения, c_1, c_2, c_3, ..., c_n – домен отношения (множество значений одного атрибута).

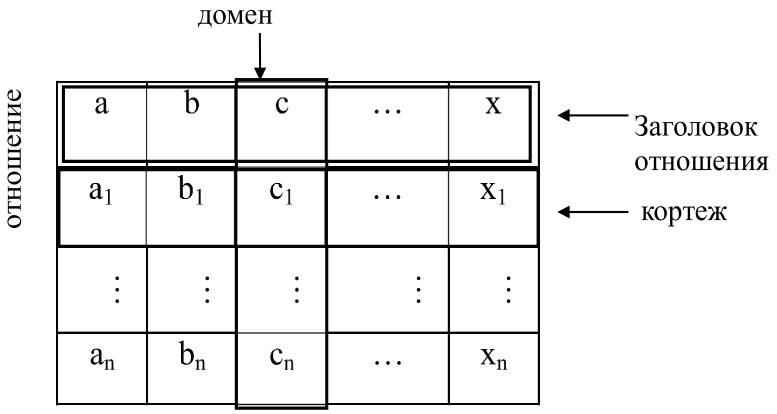


Рис. 5.1. Основные компоненты реляционного отношения

Отношение состоит из двух частей – **заголовка и тела отношения.** Атрибуты – это заголовок отношения. Тело отношения состоит из кортежей.

Строки таблицы или тело отношения представляют собой **кортежи**. **Описание одного объекта реального мира** содержится в строке, **а характеристики объекта – в столбце**.

Каждый атрибут определяется на домене, поэтому **домен может рассматриваться как совокупность допустимых значений или область определений данного атрибута**. Домен имеет имя атрибута и описание типа данных.

Например, для атрибута ВОЗРАСТ (ЧИСЛО_ЛЕТ) домен задается интервалом положительных целых чисел, так как отрицательного возраста не бывает.

Между элементами реляционной модели данных и элементами модели «сущность – связь» можно провести **аналогию.** Таким образом, реляционные отношения это наборы сущностей, кортежи – сами сущности, а столбцы в таблице – **атрибуты**.

Число m столбцов (атрибутов) отношения называется **степенью отношения** или «*арностью*» отношения. Если $m = 1$, то отношение называется унарным, при $m = 2$ – бинарным, ..., при $m = n$ – арным)

Число строк таблицы называется **мощностью отношения.**

Рассмотрим отношение **СОТРУДНИК (Таб_номер, ФИО, Должность, Стаж)**. И на его примере поясним такие понятия как кортеж, домен отношения и т.п. (рис. 5.2).

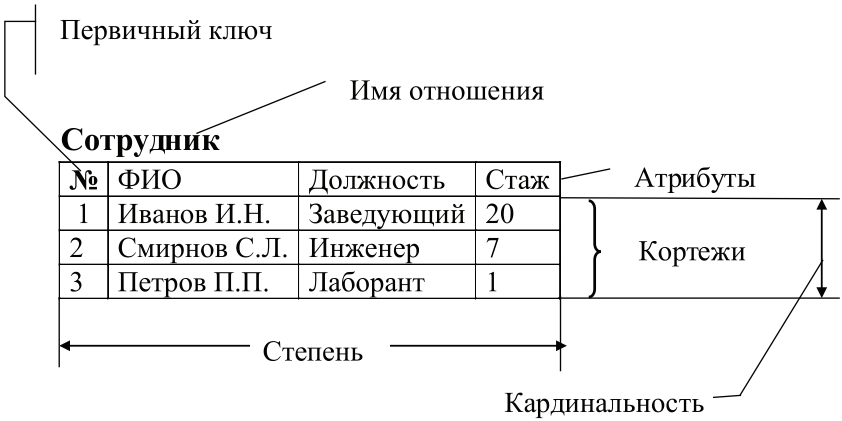


Рис. 5.2. Отношение СОТРУДНИК

Кортеж отношения:

СОТРУДНИК = {1, Иванов И.Н., заведующий, 20}.

Домен отношения:

ФИО = {Иванов И.Н., Смирнов С.Л., Петров П.П.}.

Описание имени, типа данных, размера домена: ФИО, символьный, 15.

Каждый объект, включенный в отношение, должен быть уникальным. **Ключевым** атрибутом (или **ключом**) называется атрибут, однозначно идентифицирующий кортежи. В данном случае атрибут «№» является ключом, так как его значение уникально для каждого сотрудника. В случае, когда кортежи однозначно идентифицируются только конкатенацией (сцеплением) значений совокупностью атрибутов, то говорят, что такое отношение имеет **составной ключ**.

В отношении может быть несколько ключей. При этом один из множества ключей декларируется **первичным** и его значения не

могут обновляться. Остальные ключи отношения называются *возможными или альтернативными ключами*.

Свойства отношений. Отношение (таблица) представляет собой отношение и должна обладать следующими свойствами:

1. В таблице нет одинаковых строк (отсутствие кортежей-дубликатов).

Из такого свойства следует, что у каждого кортежа имеется первичный ключ. Для каждого отношения полный набор его атрибутов является первичным ключом. Однако, при определении первичного ключа должно соблюдаться требование «минимальности», т.е. в него не должны входить те атрибуты, которые можно отбросить без ущерба для основного свойства первичного ключа, – однозначного определения кортежа.

2. Каждый столбец содержит значение одного атрибута и имеет имя, совпадающее с именем атрибута.

3. Порядок строк или столбцов в таблице не существенен.

4. Атрибуты не упорядочены (слева направо).

Графическая **схема отношения** является заголовком отношения (рис. 5.3):

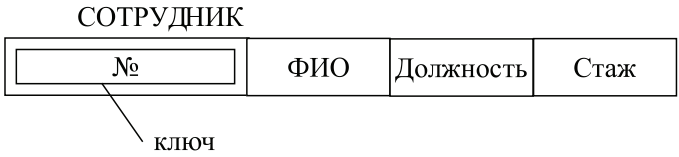


Рис. 5.3. Схема отношения СОТРУДНИК

Ключевой атрибут всегда выделяется в графической схеме.

В отличие от ИМД и СМД в реляционной модели нет такого понятия как групповое отношение. Для отражения ассоциативной связи между кортежами разных отношений используется дублирование их ключей. Приведенный ранее пример БД, которая содержала сведения об отделах предприятия и работающих в них сотрудниках, применительно к РМД будет иметь вид, показанный на рис. 5.4.

Между отношениями ОТДЕЛ и СОТРУДНИК создается связь путем копирования первичного ключа «*Номер_отдела*» из первого отношения во второе. Связь между отношениями осуществляется через дублирование ключей.

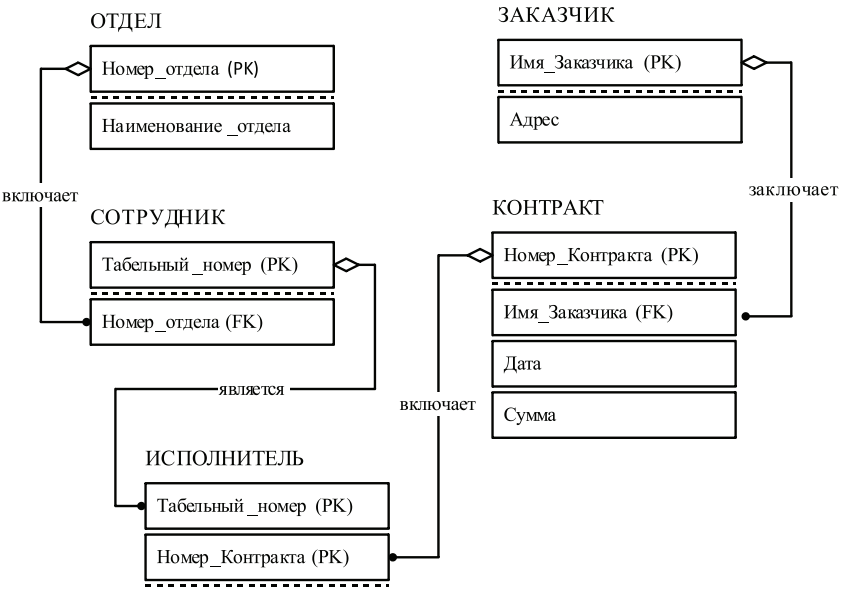


Рис. 5.4. Состав базы данных об отделах и сотрудниках предприятия

Атрибуты, которые представляют собой копии ключей других отношений, именуют **внешними ключами**.

5.2. Операции над данными

Типичными операциями над таблицей являются: **добавление в таблицу новой записи**, **нахождение записи по заданному ключу** с целью доступа и обработки полей записи и **удаление записи** из таблицы.

Для небольшой таблицы для поиска заданной или свободной записи применяется последовательный просмотр таблицы. Для большой таблицы применяют несколько изощренных способов поиска, чтобы не допускать просмотра всей таблицы и ускорить поиск.

Таким образом:

• для того, чтобы найти список работников данного отдела, необходимо:

1) в таблице ОТДЕЛ установить значение атрибута «*Номер отдела*», отвечающее критерию заданного наименования отдела;

2) осуществить выборку из таблицы СОТРУДНИК всех записей с полученным на предыдущем шаге значением атрибута «*Номер отдела*».

● для того, чтобы определить, в каком отделе работает тот или иной сотрудник, необходимо произвести обратную операцию:

1) найти «*Номер отдела*» из таблицы СОТРУДНИК;

2) по определенному на предыдущем шаге значению найти соответствующую запись в таблице ОТДЕЛ.

5.3. Ограничения целостности

Механизма поддержания семантики связи в РМД нет. В РМД каждое отношение существует независимо от другого, поэтому в предметной области «Предприятие»:

1) в отношение СОТРУДНИК можно внести нового сотрудника, не указав номер отдела,

2) из отношения ОТДЕЛ можно удалить некоторую запись, хотя в отношении СОТРУДНИК существует некоторое непустое множество записей, принадлежащих данному отделу.

В модели содержится лишь небольшое число **средств поддержания целостности данных**:

1. Не допускаются кортежи с одинаковым значением первичного ключа.

2. Обеспечивается возможность наложения ограничений на значения доменов (допустимые, разрешенные и существующие).

Допустимые значения – тип данных (дата, символьное значение и т.д.).

Разрешенные значения – диапазон возможных значений.

Существующие значения (например, сумма отчислений не должна превышать суммы зарплаты).

Контрольные вопросы

1. Что представляет собой «отношение»?

2. Дайте развернутое описание структурной части РМД.

3. Дайте определения следующим понятиям: отношение, размерность, ключ.

4. Для какой цели используются ключи отношения?

5. Дайте определения составного родительского и внешнего ключа.

6. Какие ограничения целостности данных поддерживаются РМД Кодда?

6. ТЕОРИЯ НОРМАЛЬНЫХ ФОРМ

План темы

1. Понятие нормализации.
2. 1*NF* – первая нормальная форма. Понятия простого и сложного атрибута.
3. 2*NF* – вторая нормальная форма. Полная функциональная зависимость.
4. 3*NF* – третья нормальная форма. Транзитивная зависимость
5. Нормальная форма Бойса – Кодда.
6. Многозначные зависимости и приведение отношения к четвертой нормальной форме.
7. Зависимости по соединению и пятая нормальная форма (5*NF*)

Ключевые слова: *нормализация, декомпозиция, избыточность данных, простой и составной ключ отношения, функциональная, транзитивная, многозначная зависимости, 1НФ, 2НФ, 3НФ, 4НФ, 5НФ, нормальная форма Бойса-Кодда.*

6.1. Понятие нормализации

Нормализация – это процесс декомпозиции сложной структуры данных на эквивалентное множество плоских файлов.

Декомпозиция – это разбиение объекта обработки (системы, данных, программы, задачи) на структурные единицы.

Рассмотрим преобразование ИМД в РМД. Преобразование выполняется путем размножения узлов (рис. 6.1).

Получено представление файла в виде таблицы. Достаточно присвоить таблице имя и именовать домены.

Рассмотрим преобразование СМД в РМД. Преобразование выполняется в следующей последовательности: СМД → ИМД → РМД. Для его выполнения применяется процесс размножения узлов (рис. 6.2).

Функциональные зависимости. Реляционная БД содержит как структурную, так и смысловую (семантическую) информацию.

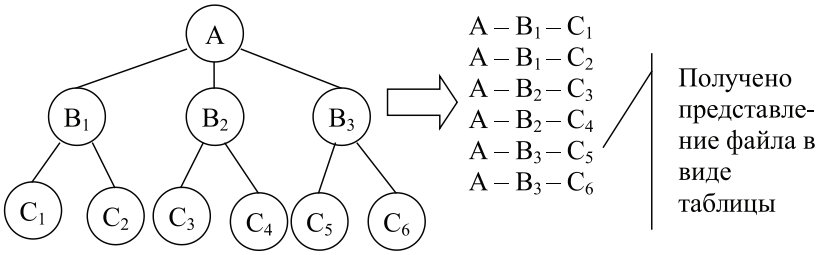


Рис. 6.1. Преобразование ИМД в РМД

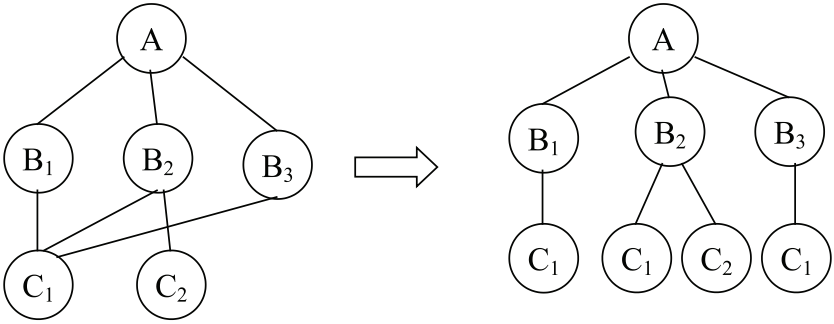


Рис. 6.2. Преобразование СМД в РМД

Структура БД характеризуется видом и числом включенных в нее отношений, а также связями «один ко многим», которые существуют между кортежами этих отношений.

Семантическая часть БД определяет функциональные зависимости, которые существуют между атрибутами отношений.

Дадим определение функциональных зависимостей.

Если заданы два атрибута *X* и *Y* некоторого отношения, то атрибут *Y* функционально зависит от атрибута *X*, если в любой момент времени каждому значению *X* соответствует только одно значение *Y*.

Связь функциональной зависимости обозначается как $X \rightarrow Y$. X и Y могут представлять не только отдельные атрибуты, но и совокупности,

составленные из нескольких атрибутов одного отношения. Некоторые функциональные зависимости могут быть нежелательны.

Определение. **Избыточная функциональная зависимость – это зависимость, которая заключает в себе информацию, могущую быть полученной на основе других зависимостей, имеющихся в БД.**

Корректной считается схема БД, в которой не имеются избыточные функциональные зависимости. Иначе необходимо приступать к процедуре разложения (декомпозиции) имеющегося множества отношений. При этом вновь образуемое множество будет содержать большее число отношений, являющихся проекциями отношений исходного множества.

Обратимый пошаговый процесс замены данного множества отношений другим с удалением избыточных функциональных зависимостей называется **нормализацией.**

Критерий обратимости обязывает, чтобы декомпозиция сохраняла эквивалентность отношений при замене одного отношения на другое, т.е. в полученных отношениях:

- не должны появляться новые ранее отсутствовавшие кортежи;
- на новых отношениях должно выполняться исходное множество функциональных зависимостей.

Вывод: Существует строгая теория, которая использует понятия **нормальных форм отношений (НФ)**. Всего разработано шесть нормальных форм отношений: 1НФ, 2НФ, 3НФ, НФБК (нормальная форма Бойса – Кодда), 4НФ, 5НФ или 1*NF*, 2*NF*, 3*NF,* 4*NF*, 5*NF* [2, 9, 12, 13]. Две последние формы на практике используются очень редко, и разработчик стремится к тому, чтобы обеспечить нормализацию таблиц по крайней мере до уровня 3НФ или НФБК. Предполагается, что каждая нормальная форма анализируемого отношения уже находится в нормальной форме на уровень ниже рассматриваемой. Чем выше уровень нормальной формы, тем более жесткие ограничения накладываются на отношения. В ходе модификаций схема базы данных становится более строгой, а ее отношения всё менее подвержены аномалиям различного рода. Процесс перехода от нормальной формы более низкого уровня к форме высокого уровня называется **нормализацией отношений**.

6.2. 1*NF* – первая нормальная форма. Понятия простого и сложного атрибута

Для обсуждения первой нормальной формы необходимо ввести следующие определения:

Простой атрибут – атрибут, значение которого атомарно (неделимо).

Сложный атрибут – атрибут, который определен на разных доменах (его также называют вектором или агрегатом данных).

Отношение называется нормализованным или приведенным к первой нормальной форме, если все его атрибуты атомарные, то есть являются простыми.

Пример. Рассмотрим ненормализованное отношение в системе учета претендентов на рабочие места. Во время проектирования выясняется, что будущие пользователи системы хотят хранить два справочных телефона для каждого кандидата на рабочее место. Получается следующее отношение (рис. 6.3).

КАНДИДАТ

Идентификатор кандидата	ФИО	Телефон 1	Телефон 2

Рис. 6.3. Пример таблицы со сложным атрибутом

В таблице нарушено условие первой нормальной формы, так как в таблице имеется два однотипных столбца. То есть один атрибут *телефон* определен на двух доменах *телефон 1* и *телефон 2*. Это может привести к следующим последствиям при использовании ИС. Может потребоваться ввод 3–4 телефонов для некоторых кандидатов. В результате необходимо будет менять не только структуру таблицы, но и программный код приложения.

Необходимо удалить сложный атрибут из данного отношения и определить его на одном домене в другом отношении (рис. 6.4).

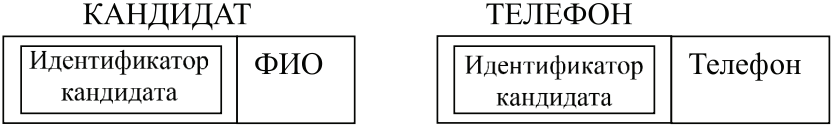


Рис. 6.4. Таблицы, приведенные к первой нормальной форме

Новая структура не содержит повторяющихся доменов.

Таким образом, каждая таблица в **первой нормальной форме (1НФ)** удовлетворяет критерию, в соответствии с которым в каждой ячейке на пересечении строки и столбца таблицы всегда находится одно единственное атомарное значение.

6.3. 2*NF* – вторая нормальная форма. Полная функциональная зависимость

Ключевой атрибут используется для идентификации каждой записи в таблице. Основным требованием к ключу является уникальность его значений. Если нельзя определить уникальность каждой записи в таблице, используя лишь один атрибут, то используют составной ключ, состоящий из нескольких атрибутов.

Первичный ключ отношения, включающий несколько атрибутов, называют, как указывалось выше, **составным**. При этом вводится понятие **полной функциональной зависимости**.

Определение. **Не ключевой атрибут функционально полно зависит от составного ключа, если он функционально зависит от всего ключа в целом, но не находится в функциональной зависимости от какого-либо из входящих в него атрибутов.**

Пример. Пусть имеется некоторое отношение ПОСТАВКА (*№* поставщика, товар, цена). Любой поставщик может доставлять различные товары, а любой товар может доставляться различными поставщиками. В этом случае ключ отношения – «*№ поставщика + товар*». Предположим, что все поставщики доставляют товар по одинаковой цене. Тогда получаем в отношении следующие функциональные зависимости:

- *№_поставщика, товар → цена*;
- *товар → цена*.

Наблюдаемая неполная функциональная зависимость приводит к аномалии: при изменении цены товара необходимо провести полный просмотр отношения для того, чтобы отредактировать все записи о поставщиках этого товара. Такая аномалия является следствием того, что в одной структуре данных объединены два семантических факта. Следующая декомпозиция приводит отношения во 2НФ:

- ПОСТАВКИ (№_поставщика, товар);
- ЦЕНА_ТОВАРА (товар, цена).

Таким образом, если в каком-либо отношении, имеющем составной ключ, наблюдается зависимость неключевых атрибутов от части ключа, необходимо образовать отдельную сущность, первичным ключом которой сделать ту часть составного ключа, от которой зависят данные атрибуты.

Пример: рассмотрим отношение, которое используется для выполнения учета работ, проводимых в МУП ЖРЭУ (табл. 6.1).

Таблица 6.1

Журнал учета проводимых работ

№ п/п	Адрес	Дата	Описание работы	Отзыв
1.	К. Маркса, 13	05.03.04	ремонт отопления	удовл.
2.	К. Маркса, 15	05.03.04	очистка водостока	хорошо
3.	Коммунистическая, 19	05.03.04	ремонт отопления	хорошо

Ключ отношения = № п/п + Дата + Адрес + Описание работы,

где № п/п – порядковый номер на текущую дату.

Вследствие наличия многозначной зависимости *описание работы* также входит в ключ. Атрибут *Описание работы* не полностью зависит от составного ключа. Атрибут *Описание работы* зависит от атрибута *Адрес* и наоборот.

Недостатки данной таблицы: в таблице имеет место дублирование информации при вводе адресов и описании работ, в результате появляется возможность ввода ошибочной информации. Многократный ввод повторяющейся информации приводит к уменьшению скорости при выполнении запросов из-за неоправданного увеличения размера БД.

Для нормализации отношения нужно, чтобы все неключевые атрибуты функционально полно зависели от составного ключа отношения. Для приведения к нормальной форме нужно:

– построить проекцию, исключив атрибуты, которые не находятся в полной функциональной зависимости от составного ключа;

– построить дополнительные проекции на часть составного ключа атрибутов функционально зависящих от этой части ключа.

Для этого исключим описание работы и заменим его идентификатором.

Составной ключ отношения = № п/п + Дата + Адреса +
+ Идентификатор описание работы.

Строим проекции на часть составного ключа и атрибуты, функционально зависящие от этой части ключа (рис. 6.5).

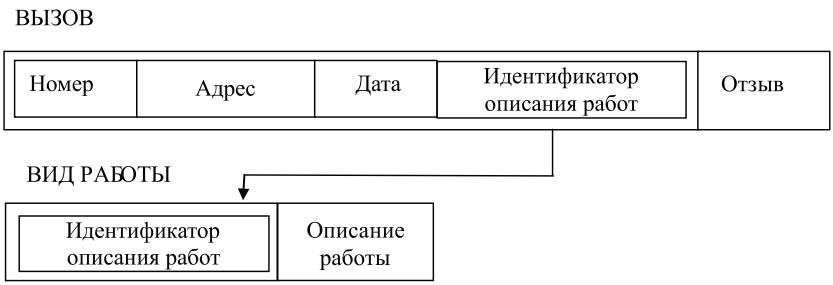


Рис. 6.5. Отношения, приведенные ко второй нормальной форме

Отношение находится во второй нормальной форме, если оно находится в первой нормальной форме и каждый неключевой атрибут функционально полно зависит от составного ключа.

Замечание. Понятие 2НФ применяется только к тем отношениям, которые имеют составной ключ. В случае, когда потенциальный ключ отношения простой, отношение автоматически находится во 2НФ.

6.4. 3*NF* – третья нормальная форма. Транзитивная зависимость

Перед обсуждением третьей нормальной формы вводим понятие **транзитивной функциональной зависимости**.

Определение. **Предположим отношение R включает в себя три атрибута: *X*, *Y*, *Z* (*R* ∈ (*X*, *Y*, *Z*)), при этом между атрибутами *X* и *Y* существует функциональная зависимость *X* → *Y*, такая же зависимость существует между *Y* и *Z* (*Y* → *Z*), но в обратном направлении такой зависимости нет *Z* ↛ *Y* или *Y* ↛ *X*, тогда говорят, что *Z* транзитивно зависит от *X*.**

Пример. Имеется отношение (рис. 6.6), которое содержит информацию о выполненной работе.

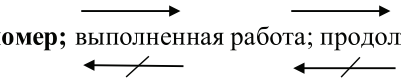


Рис. 6.6. Иллюстрация транзитивной зависимости

Ключ отношения – табельный номер.

Отношение находится в 1*NF* (все атрибуты атомарные).

Отношение находится в 2*NF* (нет составного ключа).

В таблице имеется транзитивная зависимость: одно неключевое поле зависит от другого неключевого поля. А именно:

– табельный номер → выполненная работа;

– выполненная работа → продолжительность работы;

– выполненная работа ←/— продолжительность работы.

В такой таблице наблюдается дублирование информации. Для устранения этой аномалии необходимо декомпозировать исходное отношение на три:

R1 (табельный номер; № выполняемой работы);
R2 (№ выполненной работы; выполняемая работа);
R3 (№ выполняемой работы; продолжительность работы).

Определение третьей нормальной формы.

Отношение находится в 3НФ, если оно находится во 2НФ и каждый неключевой атрибут нетранзитивно зависит от первичного ключа (то есть, нет функциональной связи в отношении между неключевыми атрибутами**).**

6.5. Нормальная форма Бойса – Кодда

Нормальная форма Бойса-Кодда (НФБК) – это фактически несколько улучшенный вариант ЗНФ для ситуаций, в которых отношение имеет несколько потенциальных ключей.

На практике ЗНФ и НФБК часто являются эквивалентными, за исключением тех случаев, когда в наличии имеются следующие условия:

- отношения имеют не менее двух потенциальных ключей;
- два потенциальных ключа являются составными;
- два потенциальных ключа имеют, по крайней мере, один общий атрибут.

Рассмотрим другой пример отношения в НФБК. Отношение РАСПИСАНИЕ содержит только самые необходимые сведения: какой студенческой группе, куда и в какое время приходить. Представлены два варианта с двумя возможными ключами:

РАСПИСАНИЕ (**аудитория, день, время**, группа)

или

РАСПИСАНИЕ (аудитория, **день, время, группа**).

Очевидно, что в этом отношении возможны два составных ключа: **{аудитория, день, время}** и **{день, время, группа}**. В первом случае неключевой атрибут **группа** функционально зависит от всего составного ключа **{аудитория, день, время}**. Во втором случае такая зависимость существует между атрибутом **аудитория** и составным ключом **{день, время, группа}**. Таким образом, оба приведенных варианта отношения РАСПИСАНИЕ находятся в НФБК, т.к. неключевой атрибут полностью зависит от всего ключа.

Определение. Таблица находится в **нормальной форме Бойса-Кодда (НФБК)**, тогда и только тогда, когда **любая** функциональная зависимость между её атрибутами сводится к полной функциональной зависимости от возможного первичного ключа.

6.6. Многозначные зависимости и приведение отношения к четвертой нормальной форме

Многозначная зависимость между атрибутами *X* и *Y* существует, если при заданных значениях атрибута *X* существует множество, состоящее из нуля или более ассоциированных значений атрибута *Y*.

Обозначение: ($X \rightarrow Y$).

Пример. Рассмотрим схему отношения ПРЕПОДАВАТЕЛЬ (табл. 6.2).

Таблица 6.2

Отношение ПРЕПОДАВАТЕЛЬ

Имя *X*	Курс *Y*	Учебное пособие *Z*
N	Теория упругости	Теория упругости
N	Теория упругости	Теория колебаний
N	Теория колебаний	Теория упругости
N	Теория колебаний	Теория колебаний

Данное отношение показывает, что преподаватель *N*, читающий курс «Теория упругости» выпустил два пособия. Кроме того, показывается, что преподаватель *N*, читающий курс «Теория колебаний» выпустил также два пособия.

В данном отношении имеются многозначные зависимости (рис. 6.7).

Рис. 6.7. Иллюстрация наличия многозначной зависимости

Определение. **Отношение находится в четвертой нормальной форме, если в нем отсутствуют многозначные зависимости, которые не являются функциональными.**

Для устранения этих аномалий необходимо декомпозировать исходное отношение на два: КУРС(***N***, *Y*) и УЧЕБНОЕ ПОСОБИЕ(***N***, *Z*).

6.7. Зависимости по соединению и пятая нормальная форма (5*NF*)

Ранее предполагалось, что единственной и достаточной операцией для устранения избыточности в отношении, является его декомпозиция на соответствующие проекции. Однако существуют отношения, для которых нельзя выполнить без потерь декомпозицию на две проекции, но которые можно декомпозировать без потерь на три (или более) проекций. Такой факт называют **зависимостью по соединению**, а такие отношения получили название **3 – декомпозируемые отношения** (ясно, что любое отношение можно назвать «*n*-декомпозируемым», где $n >= 2$) [2]. Зависимость по соединению есть обобщение многозначной зависимости. Отношения, имеющие зависимости по соединению, не являются одновременно ни многозначными, ни функциональными, и характеризуются аномалиями обновления. Поэтому, вводится понятие пятой нормальной формы.

Определение. **Отношение находится в 5НФ тогда и только тогда, когда любая зависимость по соединению в нем определяется только его возможными ключами.**

Другими словами, каждая проекция такого отношения содержит не менее одного возможного ключа и не менее одного неключевого атрибута.

Вывод: Нормализации данных сводит к минимуму их избыточность, Исключаются различные аномалии обработки. Результатом нормализации является модель данных, которую легко поддерживать, и которая не содержит неопределенностей и повторений данных.

Любая нормализация приводит к увеличению числа отношений в схеме БД. Но при этом всегда сохраняется возможность получения исходных отношений путем выполнения операции соединения. С другой стороны появление новых отношений порождает проблему поддержания семантической целостности БД.

Контрольные вопросы

1. Поясните, почему нельзя использовать документы, которые описывают предметную область и имеют табличную форму, напрямую в качестве таблиц проектируемой БД?

2. В чем отличие информационного объекта от документа, который описывает предметную область? В чем отличие информационного объекта от таблицы БД?

3. Объясните цель нормализации.

4. Объясните смысл терминов: избыточность данных, нормализация отношений, декомпозиция, простой и сложный атрибуты, составной ключ, функциональная, транзитивная и многозначная зависимости.

5. Что понимается под атомарностью атрибута?

6. Что такое первичный ключ отношения и какие требования предъявляют к нему?

7. Поясните, что такое транзитивная зависимость? К каким последствиям (негативным) приводит наличие транзитивной зависимости? Как можно избавиться от таких последствий?

8. Объясните отличия третьей нормальной формы отношений и формы Бойса-Кодда.

9. Сформулируйте назначение 1НФ, 2НФ, 3НФ, 4НФ.

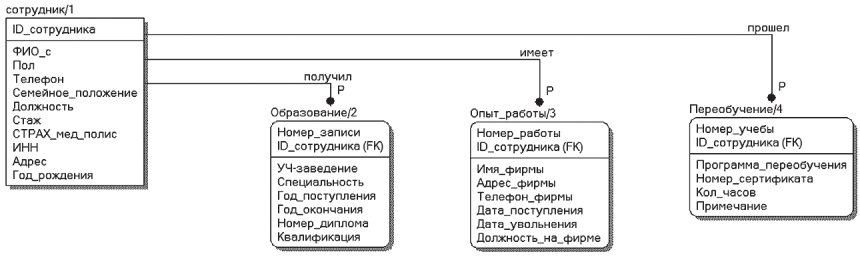
10. В чем преимущества сильно нормализованных отношений?

11. Выполните нормализацию отношения ПРОБА для автоматизированной системы, выполняющей анализ бактерицидных показателей воды: ПРОБА (Номер пробы воды, дата, тип объекта, название объекта, точка отбора, показатели (ОБО, коли-индекс, коли-фаги и т.д.), значение показателя, норматив показателя, результат анализа).

12. Выполните нормализацию отношения

Монтаж оборудования (Заказчик, Адрес_заказчика, ФИО монтажника, Разряд монтажника, Дата начала монтажа, Вид_работ, Категория сложности_работ, стоимость выполняемой работы, Вид_оборудования, Дата_сдачи, Общая стоимость_монтажа).

13. Проверьте находится ли приведенная ниже БД в 1НФ. Присутствует ли в ней дублирование данных? Приведена ли БД к 3НФ? Если БД не приведена к 3НФ, то следует ее привести к 3НФ.



7. ОГРАНИЧЕНИЯ ЦЕЛОСТНОСТИ В РАСШИРЕННОЙ РЕЛЯЦИОННОЙ МОДЕЛИ ДАННЫХ

План темы

1. *NULL*-значения. Трехзначная логика.
2. Целостность сущностей (категорная целостность).
3. Целостность ссылок.
4. Основные операции, приводящие к нарушению целостности БД. Стратегии поддержания целостности данных.
5. Корпоративные ограничения целостности данных (бизнес-правила предметной области).

Ключевые слова: *NULL-значения, трехзначная логика, целостность сущностей (категорная целостность), целостность ссылок, реакции системы (стратегии) на попытку нарушения целостности БД, корпоративные ограничения целостности данных (бизнес-правила предметной области).*

Целостность данных представляет собой механизм поддержания соответствия БД предметной области.

В модели Кодда, кроме **ограничений на значения домена** (допустимые, разрешенные, существующие), **не разрешается изменение первичного ключа**.

Ограничение целостности – это семантическое понятие, т.е. проявляется вследствие некоторых свойств объектов предметной области.

Таким образом, **ограничение целостности – это утверждение, которое может быть ложным или истинным в зависимости от состояния БД.**

Примерами могут служить следующие утверждения:

1) сотрудник должен быть не младше 18 и не старше 65 лет;

2) у сотрудника табельный номер должен быть уникальным.

Считается, что БД находится в согласованном или целостном состояниях, при условии выполнения всех ограничений целостности, определенных для БД.

Расширенная РМД определяет два ограничения, которые должны выполняться в любой реляционной БД [4, 7]:

- целостность сущностей (категорная целостность);
- целостность ссылок.

7.1. *NULL*-значения. Трехзначная логика

В БД могут находиться неопределенные (*Null*) значения. Пример, БД для регистрации информации о преступниках.

***Null*-значения – некие маркеры, показывающие, что данные неизвестны**.

Теоретики БД придерживаются по этому вопросу противоположных точек зрения. Так Кодд считал, что – *null*-значения являются неотъемлемой частью БД. Дейт [2] выступает против использования *null*-значений. Современные СУБД поддерживают *null*-значения.

При выполнении любых математических операций, если одно из данных неизвестно, то и результат неизвестен. При выполнении логических операций при использовании трехзначной логики результат операции может иметь следующие значения: *F* – ложь; *T* – истина; *U* – неизвестно.

Трехзначная логика основана на таблицах истинности (табл. 7.1 и 7.2):

Таблица 7.1 *Таблица 7.2*

Таблицы истинности

x	*y*	*x and y*	*x or y*
F	*F*	*F*	*F*
F	*T*	*F*	*T*
T	*F*	*F*	*T*
T	*T*	*T*	*T*
F	*U*	*F*	*U*
T	*U*	*U*	*T*
U	*U*	*U*	*U*

	Not
F	*T*
T	*F*
U	*U*

Парадоксальные следствия использования трехзначной логики:

1) никогда *Null*-значение не равняется самому себе (*null* ≠ *null*);

2) $a + \overline{a} \neq T$ (не обязательно ИСТИНА, так как $a = U$)

7.2. Целостность сущностей (категорная целостность)

В РМД объект реального мира представляется кортежем некоторого отношения. Требование целостности сущностей – это **любой кортеж каждого отношения отличается от всякого кортежа этого отношения, т.е. каждое отношение должно иметь первичный ключ.**

Если указанное требование не выполняется (т.е. кортежи не уникальны в рамках одного отношения), тогда в БД содержится противоречивая информация об одном и том же объекте. Целостность сущностей поддерживается средствами СУБД. Это обеспечивается с помощью таких ограничений как:

- при добавлении в таблицу записей осуществляется проверка уникальности их первичных ключей;
- не допускается редактирование значений атрибутов, которые входят в первичный ключ.

Правило: атрибуты, которые входят в состав потенциального ключа не могут иметь *null*-значений.

7.3. Целостность ссылок

Объекты предметной области взаимосвязаны. Взаимосвязь объектов в РМД выражается через внешние ключи. Связь может быть типа 1:1 или 1:M.

Поэтому:

1) связи между данными отношениями записываются в терминах функциональных зависимостей;

2) для обеспечения функциональных зависимостей между кортежами разных отношений применяется дублирование первичного ключа родительского отношения в дочернее.

Атрибуты, которые представляют собой копии ключей родительских отношений, именуются **внешними ключами**.

Требование целостности по ссылкам состоит в следующем: **для любого значения внешнего ключа, которое появляется в дочернем отношении, должен найтись кортеж с таким же значением первичного ключа в родительском отношении.** Иными словами внешний ключ дочернего отношения должен быть согласован с первичным ключом родительского отношения.

Пример. Пусть, например, даны отношения ОТДЕЛ (***N*отдела**, Наименование-отдела) и СОТРУДНИК (***N*сотрудника**, *N*отдела, ФИО-сотрудника), хранящие сведения о работниках предприятия и отделах, в которых они работают. В данной паре отношение ОТДЕЛ является родительским, поэтому его первичный ключ *Nomдела* имеется в дочернем отношении СОТРУДНИК. В данном случае требование целостности по ссылкам означает, что в таблице СОТРУДНИК не может быть кортежа со значением атрибута *Nomдела*, не встречающегося в таблице ОТДЕЛ. Если такого значения в отношении ОТДЕЛ не наблюдается, то внешний ключ в отношении СОТРУДНИК считается неопределенным.

Поддержание целостности ссылок, как правило, возлагается на СУБД. Но не все СУБД автоматически поддерживают целостность БД своими средствами. В этом случае пользователь сам должен заботиться об обеспечении целостности БД при удалении и модификации данных. К примеру, она может не разрешить пользователю осуществить добавление записи, содержащей внешний ключ с неопределенным (несуществующим) значением.

Правило: внешние ключи связывают между собой два отношения, поэтому они не должны указывать на несуществующие объекты. Для любого значения внешнего ключа обязательно должно существовать в родительской таблице значение первичного ключа

7.4. Основные операции, приводящие к нарушению целостности БД. Стратегии поддержания целостности данных

Основные операции, приводящие к нарушению целостности БД:

1) удаление записи в родительском и/или дочернем отношении;

2) модификация записи в родительском отношении;

3) модификация записей в дочернем отношении.

Одновременно с понятием целостности БД вводится понятие реакции системы на попытки нарушения целостности БД.

Существует два типа реакций на попытку нарушения целостности (или две основные стратегии):

1. *Restrict* (ограничить – отказ выполнить незаконную операцию).

Данная стратегия сводится к запрету на выполнение действий, приводящих к нарушению целостности БД.

А также не позволяет:

- изменять значения полей первичного ключа в родительском отношении, если в дочернем отношении имеется, хотя бы одна запись, содержащая ссылку на изменяемую запись;
- удалять записи в родительском отношении, если в дочернем отношении имеется, хотя бы одна запись, содержащая ссылку на удаляемую запись. При попытке удаления записи возникает ошибка.

К примеру, если система знает, что в поле *Возраст сотрудника* должны быть целые числа в диапазоне от 18 до 65, то при этом генерируется какое-либо сообщение для пользователя.

2. *Cascade* (каскадировать – выполнение компенсирующего действия).

Данная стратегия обеспечивает автоматическое поддержание целостности БД, поэтому является самой сложной. При редактировании значений полей первичного ключа в родительском отношении автоматически вносится каскадное изменение соответствующих значений в дочернем отношении.

В родительском отношении при удалении записи автоматически производится каскадное удаление записей из дочернего отношения,

связанных с удаляемой записью. Работу системы по проверке ограничений можно представить, как показано на рис. 7.1.

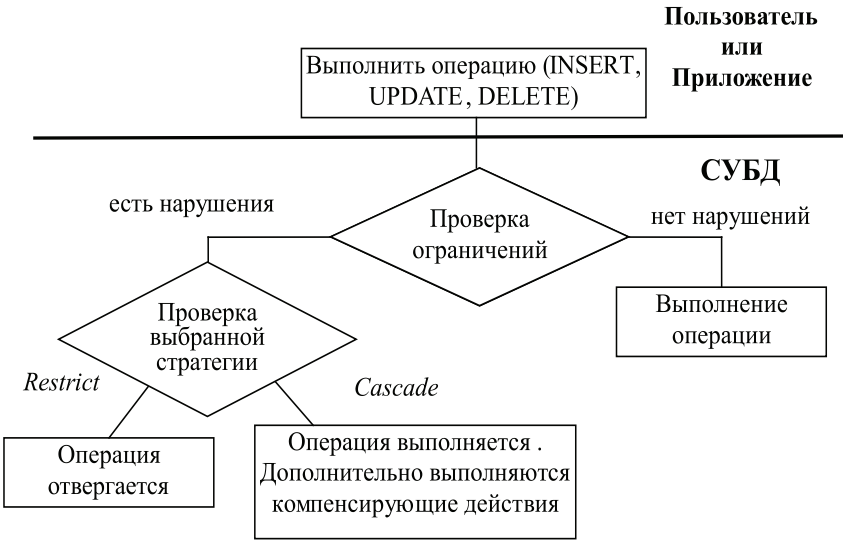


Рис. 7.1. Выполнение проверки ограничений

Дополнительные стратегии:

1. *Ignore* (игнорировать) – при выборе данной стратегии СУБД не обращает внимания на нарушение целостности данных и позволяет:

- изменять значения полей первичного ключа в родительском отношении, в независимости от существования связанных записей в дочернем отношении;
- удалять записи в родительском отношении в независимости от существования связанных записей в дочернем отношении.

2. *Set null* (установить в *NULL*) – разрешает выполнение всех требуемых операций, при этом некорректные значения внешних ключей заменяет на *null*-значения.

3. *Set default* (установить по умолчанию) – разрешает выполнение всех необходимых операций, при этом все некорректные значения внешних ключей заменяет на значения, выбранные по умолчанию.

7.5. Корпоративные ограничения целостности данных (бизнес-правила предметной области)

Еще один вид ограничений на информацию, которая содержится в отношениях, связан с бизнес-правилами, существующими в рассматриваемой предметной области. Это **дополнительные правила поддержки целостности данных, которые определяются пользователем либо администратором данных и называются корпоративными ограничениями целостности или бизнес-правилами предметной области.**

Примеры таких правил.

1. Отдел кадров с основными таблицами:

– Работник (**ТабНомер**, ФИО, дата рождения);

– Должность (**НомерД**, ИмяД)

– Отдел (**НомерО**, Наименование, Количество сотрудников);

– РаботникОтдел (**ТабНомер, НомерО, НомерД**).

Правила: сотруднику разрешается работать одновременно в нескольких отделах на разных должностях; в каждом отделе может быть несколько работников.

2. Библиотечный каталог

Основные таблицы:

– Автор (**КодА**, Фамилия, Имя, Отчество);

– Книга (**КодК**, Наименование, Количество страниц);

– АвторКнига (**КодА, КодК**).

Правила: у любой книги может быть одновременно несколько авторов, а у одного автора может быть несколько книг.

3. Расписание занятий

Основные таблицы:

– Группа (**НомерГ**, Название, Число студентов);

– Аудитория (**Корпус, НомерА**, Число посадочных мест);

– Преподаватель (**ТабелНомер**, ФИО, должность);

– Предмет (**НомерП**, ИмяП)

– Расписание (**ДатаВремя, НомерГ, Корпус, НомерА, ТабелНомер, НомерП**).

Правила: у студентов не может быть плановых занятий длительностью более восьми часов в день; при этом занятия студентов в течении одного дня должны проходить в одном корпусе.

Ограничения целостности, присущие той или иной предметной области, должны выявляться при обследовании и фиксироваться в ИМД. Ограничения целостности в БнД могут определяться либо при описании БД (триггеры, хранимые процедуры), либо в программах обработки данных. Первый подход предпочтителен так, как заданные ограничения будут выявляться при выполнении операций над данными. Проектируя БнД необходимо изучить возможности по контролю целостности данных предоставляемые используемой СУБД. Если СУБД не поддерживает автоматически необходимые ограничения, то их обеспечение становится заботой проектировщика.

Контрольные вопросы

1. Поясните понятие целостности базы данных.
2. Дайте определение двух основных правил целостности базы данных.
3. Какие существуют пути сохранения целостности базы данных при различных операциях изменения данных?
4. Опишите ситуации, приводящие к нарушению целостности данных.

СПИСОК ЛИТЕРАТУРЫ

1, Григорьев, Ю. А. Банки данных: Учеб. для вузов. / Ю. А. Григорьев, Г. И. Ревунков. – М. : Изд-во МГТУ им. Н.Э. Баумана, 2002. – 320 с.

2. Дейт, К. Дж. Введение в системы баз данных / К. Дж. Дейт. – М.: Вильямс, 2001. – 1072 с.

3. Диго, С. М. Базы данных. Проектирование и создание: Учебно-методический комплекс / С. М. Диго. – М.: изд. Центр ЕАОИ. 2008. – 171 с.

4. Малыхина, М. П. Базы данных: основы, проектирование, использование. – СПб. : БХВ-Петербург, 2004. – 512 с.

5. Плю, Р. Освой самостоятельно *SQL* за 24 часа / Р. Плю, Р. К. Стефенс. – М.: Вильямс, 2000. – 352 с.

6. Пушников, А. Ю. Введение в системы управления базами данных. Часть 1. Реляционная модель данных: Учебное пособие. / А. Ю. Пушников. – Уфа: Изд-во Башкирского ун-та, 1999. – 108 с.

7. Пушников, А. Ю. Введение в системы управления базами данных. Часть 2. Нормальные формы отношений и транзакции : Учебное пособие. / А. Ю. Пушников. – Уфа: Изд-во Башкирского ун-та, 1999. – 138 с.

8. Швецов, В. И. Базы данных: Учебное пособие. / В. И. Швецов, А. Н. Визгунов, И. Б. Мееров. – Нижний Новгород: Изд-во ННГУ, 2004. – 217 с.

9. Зеленков, Ю. А. Введение в базы данных [Электронный ресурс] / Ю. А. Зеленков. – Режим доступа: http://www.mstu.edu.ru/study/materials/zelenkov/toc.html. – 1997.

10. Карпова, Т. С. Базы данных [Электронный ресурс] / Т. С. Карпова. – Режим доступа: http://eos.ibi.spb.ru/umk/5_8/index.html, свободный.– Загл. с экрана.

11. Туманов, В. Е. Основы проектирования реляционных баз данных [Электронный ресурс] / В. Е. Туманов. – Сайт: Интуит. Национальный открытый университет – Режим доступа: http://www.intuit.ru/studies/courses/1095/191/lecture/4985?page=2, свободный.– Загл. с экрана.

12. Илюшечкин, В.М. Основы использования и проектирования баз данных. Учебник / В.М. Илюшечкин. – М.: Юрайт, 2015. – 214 с.

13. Фуфаев, Э.В. Базы данных / Э.В. Фуфаев, Д.Э. Фуфаев. – М.: Академия, 2016. – 320 с.

Учебное издание

Костюкова Анастасия Петровна

БАЗЫ ДАННЫХ:
ОБЩИЙ КУРС
Часть 1

Учебное пособие

Технический редактор Кулакова Г.А.
Подписано в печать 06.06.2018
Бумага офсетная.
Гарнитура PetersburgC
Формат 60×84 1/16
Печать трафаретная. Печ. л. 5,13.
Тираж 500 экз. Заказ № 019-18.

Отпечатано в типографии ИД «Академия Естествознания»,
440026, г. Пенза, ул. Лермонтова, 3