Управление данными

Программа курса (ГОС)

1. Основные понятия банков данных и знаний
2. Информация и данные
3. Предметная область банка данных
4. Роль и место банков данных в информационных системах
5. Пользователи банков данных
6. Преимущества централизованного управления данными
7. База данных как информационная модель предметной области
8. Система управления базой данных (СУБД)
9. Администратор базы данных;

Программа курса (ГОС)

10. Архитектура банка данных
11. Инфологическое проектирование базы данных
12. Выбор модели данных
13. Иерархическая, сетевая и реляционная модели данных, их типы структур, основные операции и ограничения
14. Представление структур данных в памяти ЭВМ
15. Современные тенденции построения файловых систем
16. Обзор промышленных СУБД
17. Тенденции развития банков данных

Литература

- *Карпова Т.С.* Базы данных: модели, разработка, реализация : Учебник для вузов. — СПб.: Питер, 2001 . — 304 с.
- *Ю.А. Григорьев, Г.И. Ревунков* Банки данных: Учебник для вузов. — М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. — 320 с.
- *Дейт К. Дж.* Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом «Вильямс», 2005. — 1328 с.
- *Краморенко Н.В.* Базы данных: Учебное пособие. — Владивосток: ТИДОТ ДВГУ, 2004. — 85 с.
- *Кузнецов С.Д.* Введение в реляционные базы данных. — http://www.intuit.ru/department/database/rdbintro/
- *Швецов В.И.* Базы данных. — http://www.intuit.ru/department/database/databases/
- http://citforum.ru/database/
- *Грабер М.* SQL : Пер. с англ. — М.: Изд-во «Лори», 2003. —

Тема 1. Введение в управление данными

1. Информационные системы с точки зрения управления данными
2. Информация и данные
3. Развитие систем и средств управления данными

Информационные системы

Информационная система (ИС) — программно-аппаратный комплекс, предназначенный для хранения и обработки информации какой-либо предметной области.

Предметная область — часть реального мира, рассматриваемая как самостоятельная единица. Она определяет информационные потребности и область применения ИС.

Сферы применения ИС

Классификация ИС по сфере применения:

- экономические
- медицинские
- географические

Информация и данные

Информация — любые сведения о каком-либо событии, сущности, процессе и т.п., являющиеся объектом операций:

- восприятия
- передачи
- преобразования (обработки)
- хранения
- использования

Информация и данные

Данные — информация, фиксированная в определенной форме, пригодной для последующей обработки, хранения и передачи.

- Данные несут в себе информацию о событиях, произошедших в материальном мире
- Чтобы данные стали информацией, необходимо преобразовать их в известные понятия
- Чтобы извлечь из данных информацию необходимо подобрать соответствующий форме данных адекватный метод получения информации

Информация и данные

Информация = данные + смысл

Информация и данные

- Информация не является статичным объектом – она динамически меняется и существует только в момент взаимодействия данных и методов
- Информация существует только в момент протекания информационного процесса
- Все остальное время информация содержится в виде данных

Аспекты проектирования ИС

Два аспекта рассмотрения понятий и проектирования ИС:

- инфологический
- даталогический

Инфологическое проектирование

Рассматривается смысловое содержание данных независимо от способа их представления в памяти.

- О каких объектах или явлениях реального мира требуется накапливать и обрабатывать информацию в системе?
- Какие их основные характеристики и взаимосвязи между собой будут учитываться?
- Какие вводимые в ИС понятия об объектах и явлениях, их характеристиках и взаимосвязях требуют уточнения?

Инфологическое проектирование

- На этапе инфологического проектирования выделяется часть реального мира, определяющая информационные потребности системы, т.е. ее предметная область
- Данные рассматриваются как отдельные факты, характеризующие объекты, процессы и явления в предметной области, а также их свойства

Даталогическое проектирование

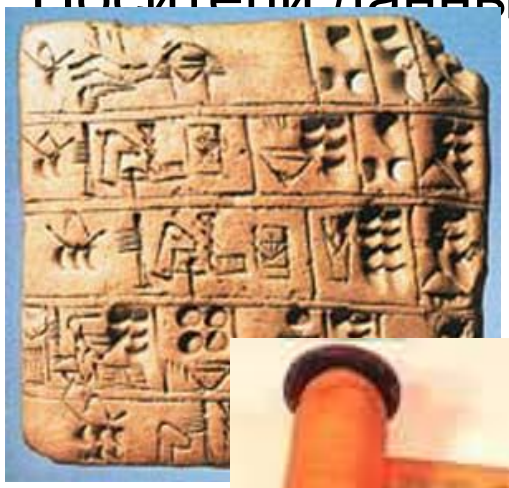
На этапе даталогического проектирования рассматриваются вопросы представления данных в памяти информационной системы:

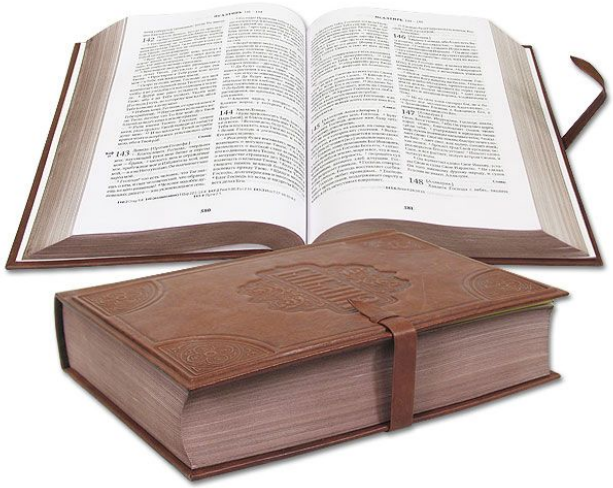
- формы представления информации в системе
- модели и методы представления и преобразования данных
- правила смысловой интерпретации данных

Развитие управления данными

Ручная обработка данных (4000 г. до н. э. – 1900)

Носители данных: глина, папирус, пергамент, бумага

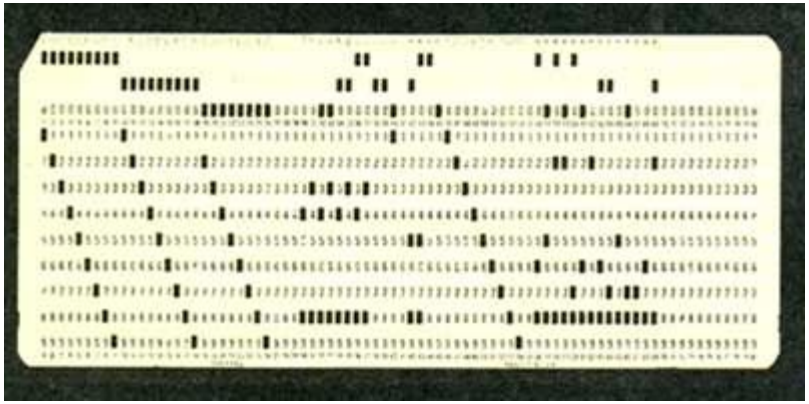




Развитие управления данными

Автоматизированная обработка информации с перфокартами

(1900-1955)



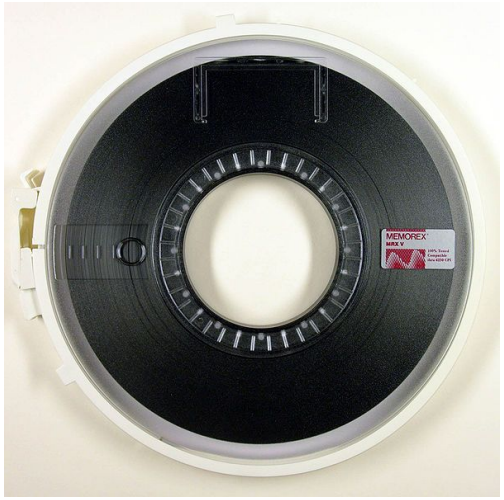
Применение вычислительной техники

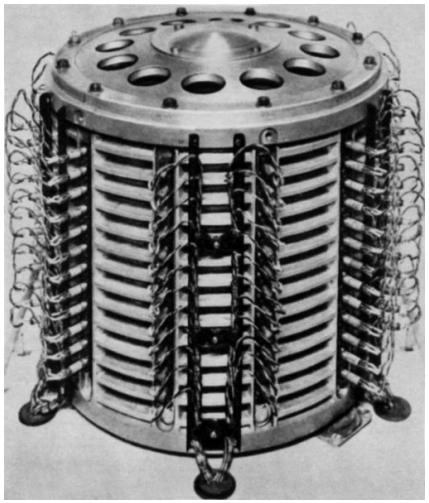
- Первое направление: применение вычислительной техники для выполнения численных расчетов
- Второе направление: использование средств вычислительной техники в автоматических или автоматизированных информационных системах

Развитие управления данными

Программируемое оборудование обработки записей (1955-1970)

Носители данных: магнитные ленты и барабаны





Развитие управления данными

Программируемое оборудование обработки записей

- язык программирования COBOL
- модель обработки записей на основе файлов
- системы пакетной обработки транзакций

Системы управления файлами

Переход к использованию централизованных систем управления файлами

Файл — это именованная область внешней памяти, в которую можно записывать и из которой можно считывать данные

Системы управления файлами

Система управления файлами берет на себя:

- распределение внешней памяти
- отображение имен файлов в соответствующие адреса во внешней памяти
- обеспечение доступа к данным и авторизации

Недостатки:

- зависимость программ от данных (от структуры файлов)
- проблемы при одновременной работе многих пользователей с одним файлом
- отсутствие централизованных методов управления доступом к информации

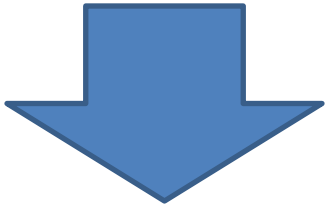
Развитие управления данными

Оперативные сетевые базы данных (1965-1980)

Носители

Развитие управления данными

- Наличие аппаратного обеспечения — устройств внешней памяти: жестких магнитных дисков
- Наличие системного программного обеспечения: операционных систем с системами управления файлами



Новый подход к управлению информацией:

Базы Данных и
Системы Управления Базами Данных

Развитие управления данными

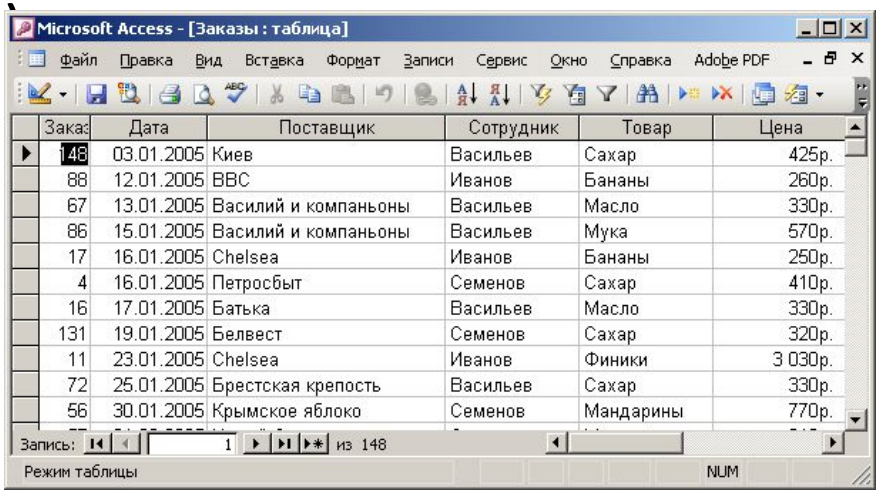
Оперативные сетевые базы данных:

- использование консольных терминалов с интерактивным режимом доступа и центральной ЭВМ
- возможность условно параллельного выполнения всего множества задач
- сетевая модель данных : взаимосвязанные наборы данных с ассоциативным доступом
- поддерживаются языки низкого уровня манипулирования данными
- значительная роль отводится администрированию данных
- транзакции, журналирование, архивация

Развитие управления данными

Реляционные базы данных

- простота и эффективность использования
- простота проектирования и программирования
- наглядность (табличная форма)
- математическое обоснование (реляционная алгебра)



Заказ	Дата	Поставщик	Сотрудник	Товар	Цена
148	03.01.2005	Киев	Васильев	Сахар	425р.
88	12.01.2005	ВВС	Иванов	Бананы	260р.
67	13.01.2005	Василий и компаньоны	Васильев	Масло	330р.
86	15.01.2005	Василий и компаньоны	Васильев	Мука	570р.
17	16.01.2005	Chelsea	Иванов	Бананы	250р.
4	16.01.2005	Петросбыт	Семенов	Сахар	410р.
16	17.01.2005	Батька	Васильев	Масло	330р.
131	19.01.2005	Белвест	Семенов	Сахар	320р.
11	23.01.2005	Chelsea	Иванов	Финики	3 030р.
72	25.01.2005	Брестская крепость	Васильев	Сахар	330р.
56	30.01.2005	Крымское яблоко	Семенов	Мандарины	770р.

Развитие управления данными

Эпоха персональных компьютеров

(1980-1995-…)

- Мощность и доступность персональных компьютеров
- Широкое использование настольных (desktop) СУБД с монопольным доступом к БД
- Развитый и удобный пользовательский интерфейс
- Использование высокоуровневых языков манипулирования данными (SQL)

Распределенные базы данных

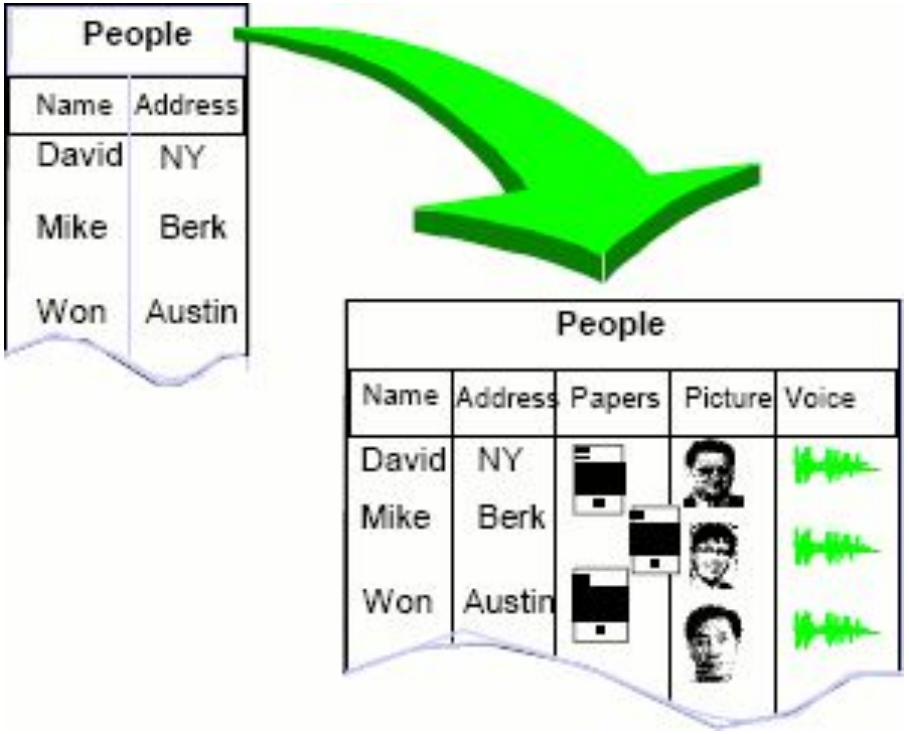
- Проблема согласованности (синхронизации) данных, хранящихся и обрабатывающихся в разных местах
- Развитие сетевых технологий: возможность децентрализованного хранения данных
- Задачи, связанные с параллельной обработкой *транзакций*
- Необходимость поддержки многопользовательской работы с базой данных

Создание распределенных баз данных

Развитие управления данными

Мультимедийные базы данных (1995-...)

Объектно-ориентированный подход



Перспективные направления и задачи

- Определение моделей данных для новых типов (например, пространственных, графических) и их интеграция с традиционными системами баз данных
- Развитие объектно-ориентированных и объектно-реляционных СУБД
- Масштабирование баз данных по размеру (до петабайт), пространственному размещению (распределенные) и многообразию (неоднородные)
- OLAP-технологии – аналитическая обработка в реальном времени

Перспективные направления и задачи

- Автоматическое обнаружение тенденций данных, их структур и аномалий (Data Mining: интеллектуальный анализ данных)
- Интеграция (комбинирование) данных из нескольких источников
- Создание сценариев и управление потоком работ (процессом) и данными в организациях
- Автоматизация проектирования и администрирования баз данных

Тема 2. Основные понятия о базах данных, банках данных и СУБД

1. Основные понятия и определения (БнД, БД, СУБД)
2. Роль и место банков данных в ИС
3. Преимущества использования БД и централизованного подхода к управлению данными
4. Архитектура баз данных. Трехуровневая модель ANSI/SPARC
5. Жизненный цикл банка данных
6. Пользователи банка данных

База данных

База данных (БД) (англ. data base) — именованная совокупность структурированных данных, отражающая состояние объектов и их отношений в рассматриваемой предметной области

- База данных — важнейший компонент любой информационной системы
- База данных является объектом манипулирования со стороны СУБД

Система управления базами данных

Система управления базами данных (СУБД)

(англ. DBMS — Data Base Management System) — программные средства манипулирования данными в целях обеспечения доступа к ним и поддержания БД в актуальном состоянии.

СУБД ≠ БД

Приложение — прикладная программа, использующая БД и написанная в СУБД или на языке программирования.

Банк данных — синоним приложения.

Банк данных

- В узком смысле:

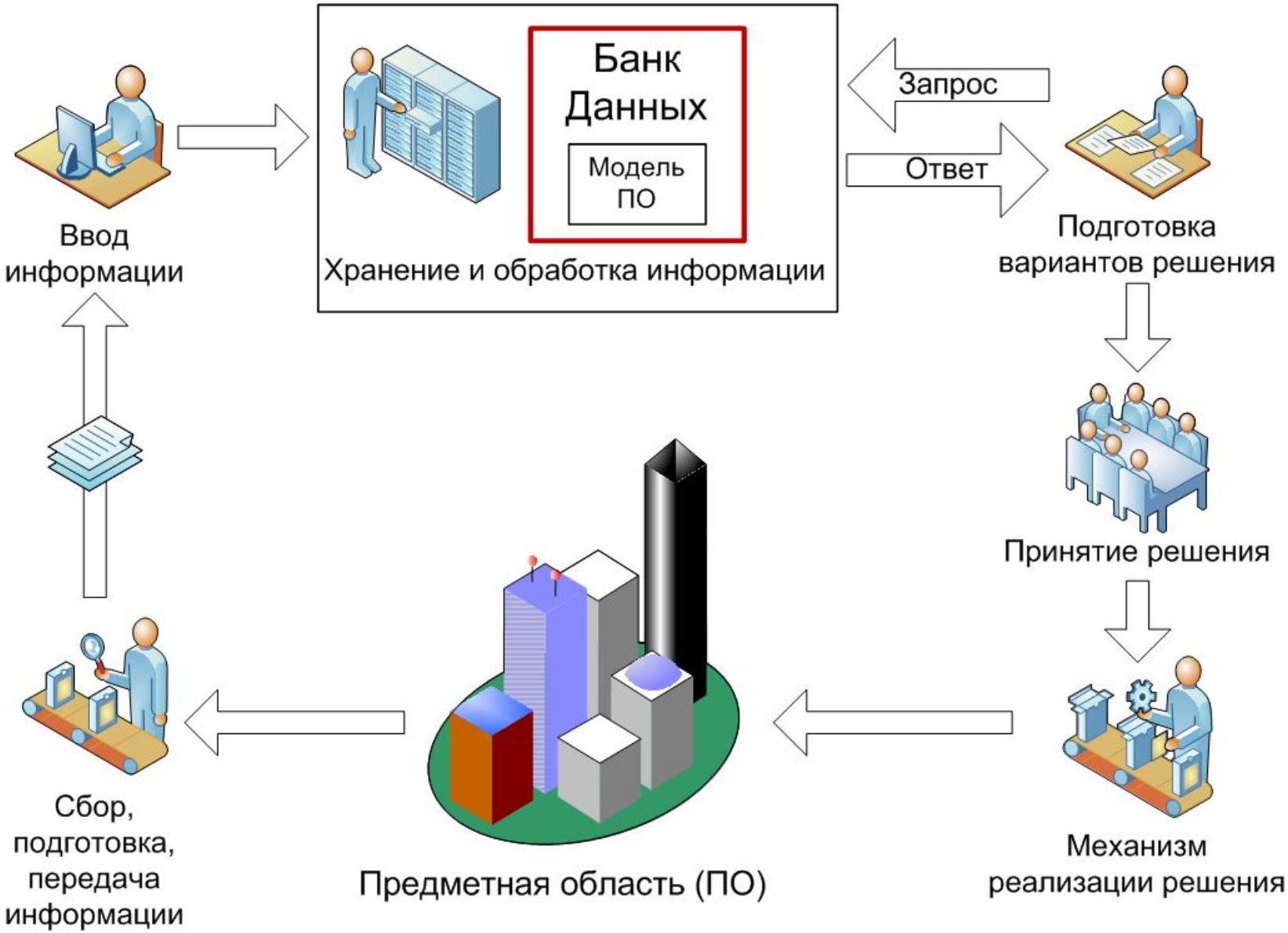
Банк данных = СУБД + БД

- В широком смысле:

Банк данных = АИС

Банк данных (БнД) — это автоматизированная ИС, включающая в свой состав комплекс специальных методов и средств для поддержания динамической информационной модели предметной области с целью обеспечения информационных запросов пользователей.

Роль и место банков данных в ИС



Преимущества использования БД

- Компактность
- Быстродействие
- Низкие трудозатраты
- Актуальность информации
- Защита данных

Преимущества использования БД

Централизованное управление данными:

- возможность совместного доступа к данным
- сокращение избыточности данных
- устранение противоречивости данных
- возможность соблюдения стандартов
- обеспечение целостности данных
- организация защиты данных
- обеспечение независимости данных

Независимость данных

Архитектура баз данных

Архитектура **ANSI/SPARC**

(Трехуровневая модель)

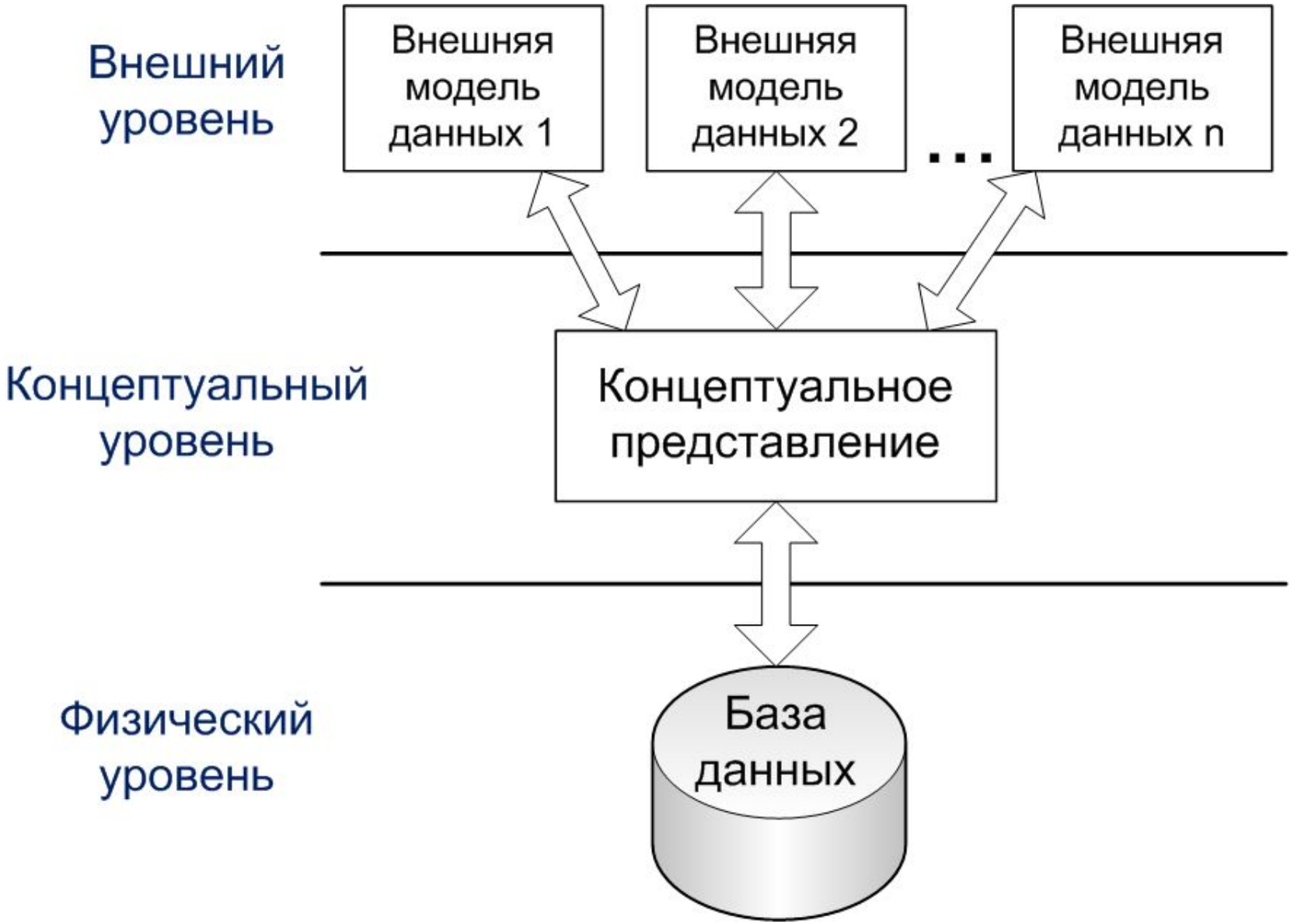
ANSI

American National Standards Institute

SPARC

Study Group on Data Management Systems

Трехуровневая модель ANSI/SPARC



Жизненный цикл банка данных

Пользователи банка данных

- Конечные пользователи (параметристы)
- Разработчики (прикладные программисты) и администраторы приложений
- Администраторы банка данных

Группа администратора БнД

- Системные аналитики
- Проектировщики структур данных
- Проектировщики технологических процессов обработки данных
- Системные и прикладные программисты
- Операторы и специалисты по техническому обслуживанию
- Специалисты по маркетингу (для коммерческих БнД)

Тема 3. Основные модели данных

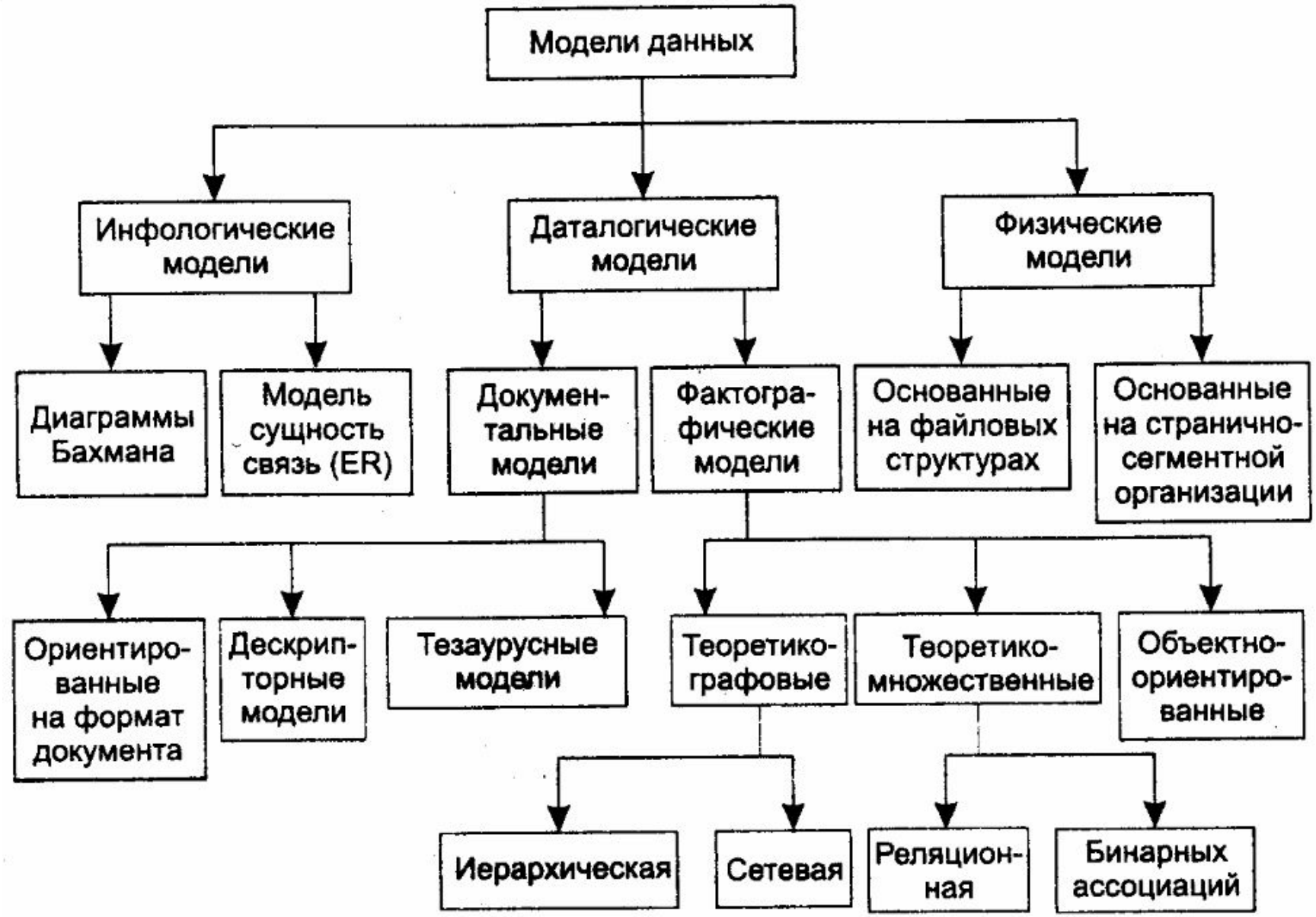
1. Понятие модели данных
2. Классификация моделей данных
3. Иерархическая модель
4. Сетевая модель
5. Реляционная модель

Понятие модели данных

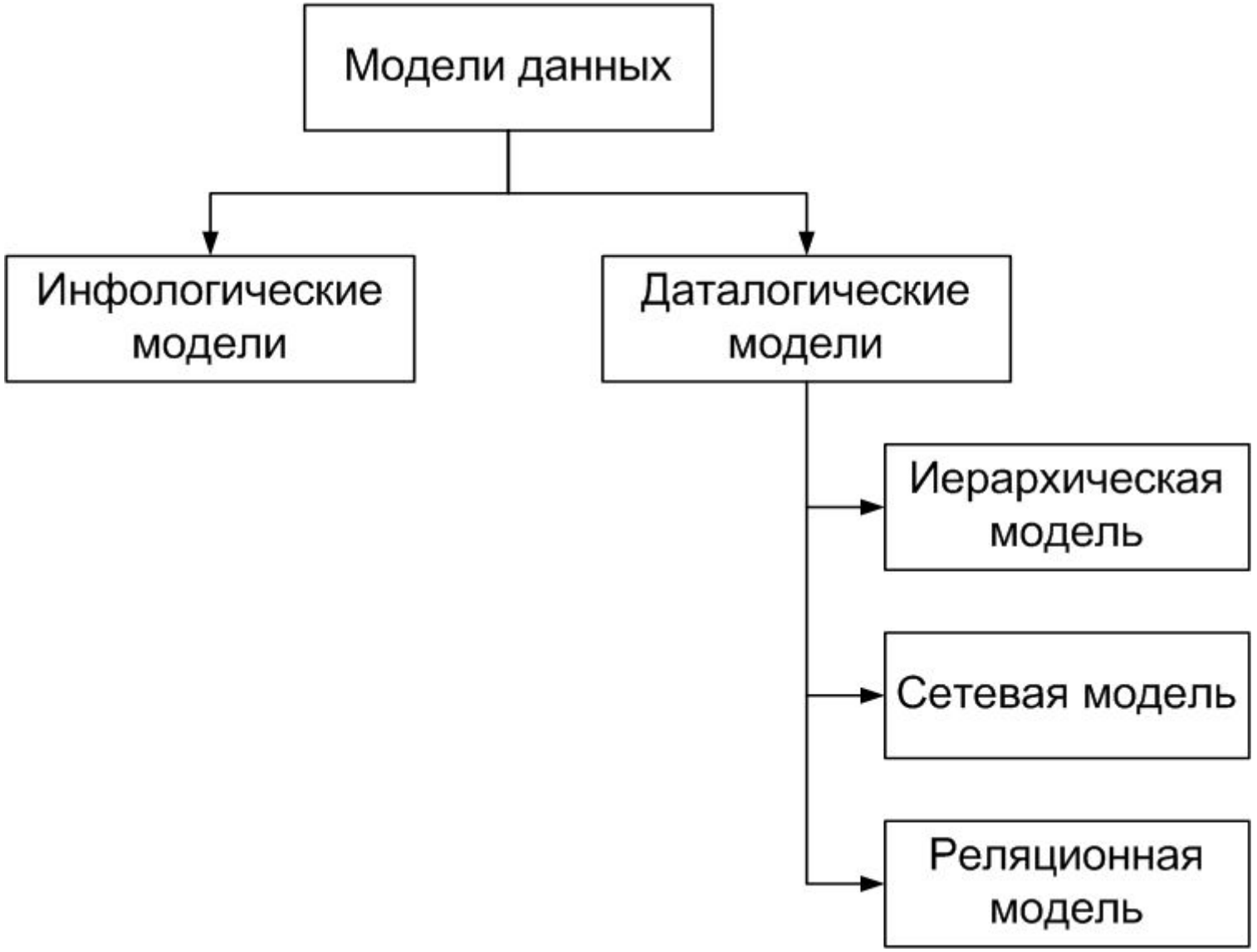
Модель данных — это некоторая абстракция, которая, будучи приложена к **конкретным данным**, позволяет пользователям и разработчикам трактовать их уже как **информацию**

то есть сведения, содержащие не только данные, но и **взаимосвязь** между ними (определенную структуру, смысловое содержание)

Классификация моделей данных

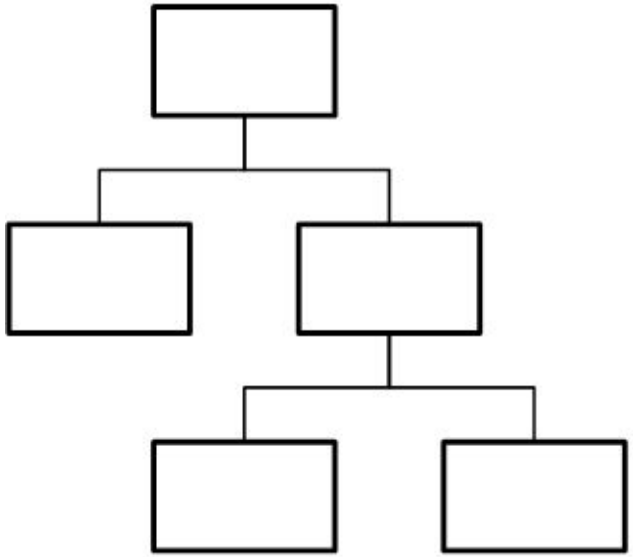


Классификация моделей данных

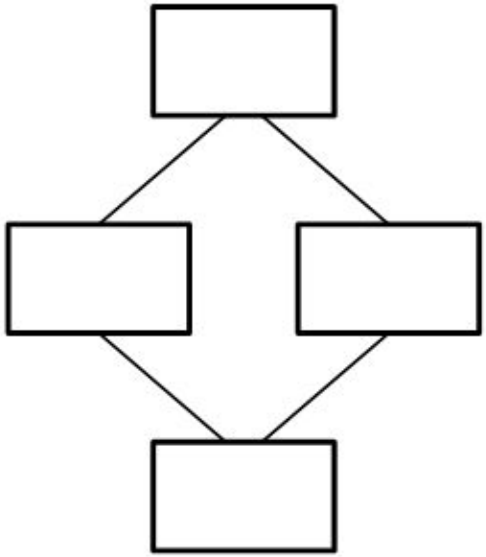


Даталогические модели

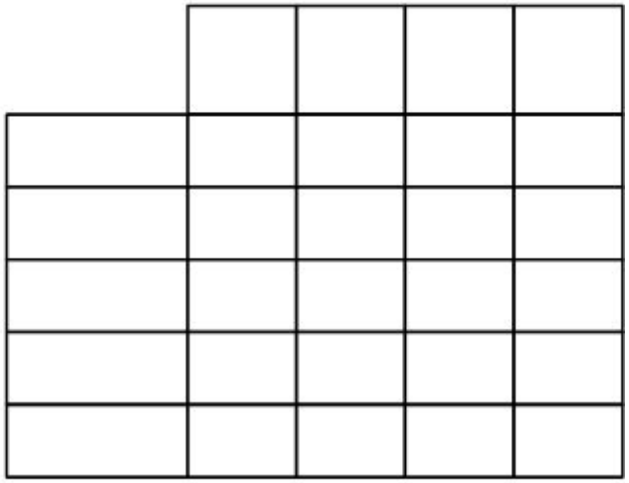
Иерархическая модель



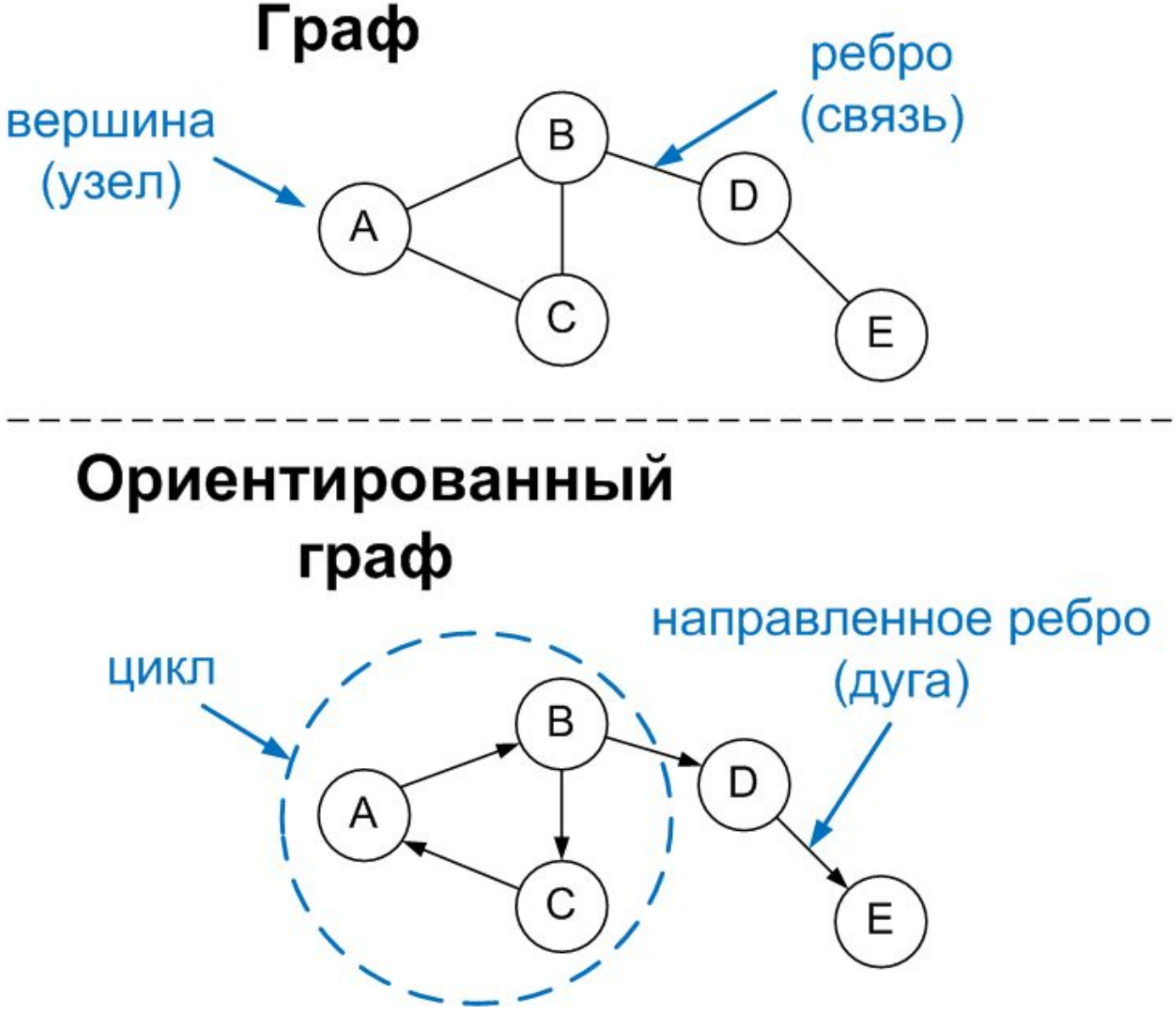
Сетевая модель



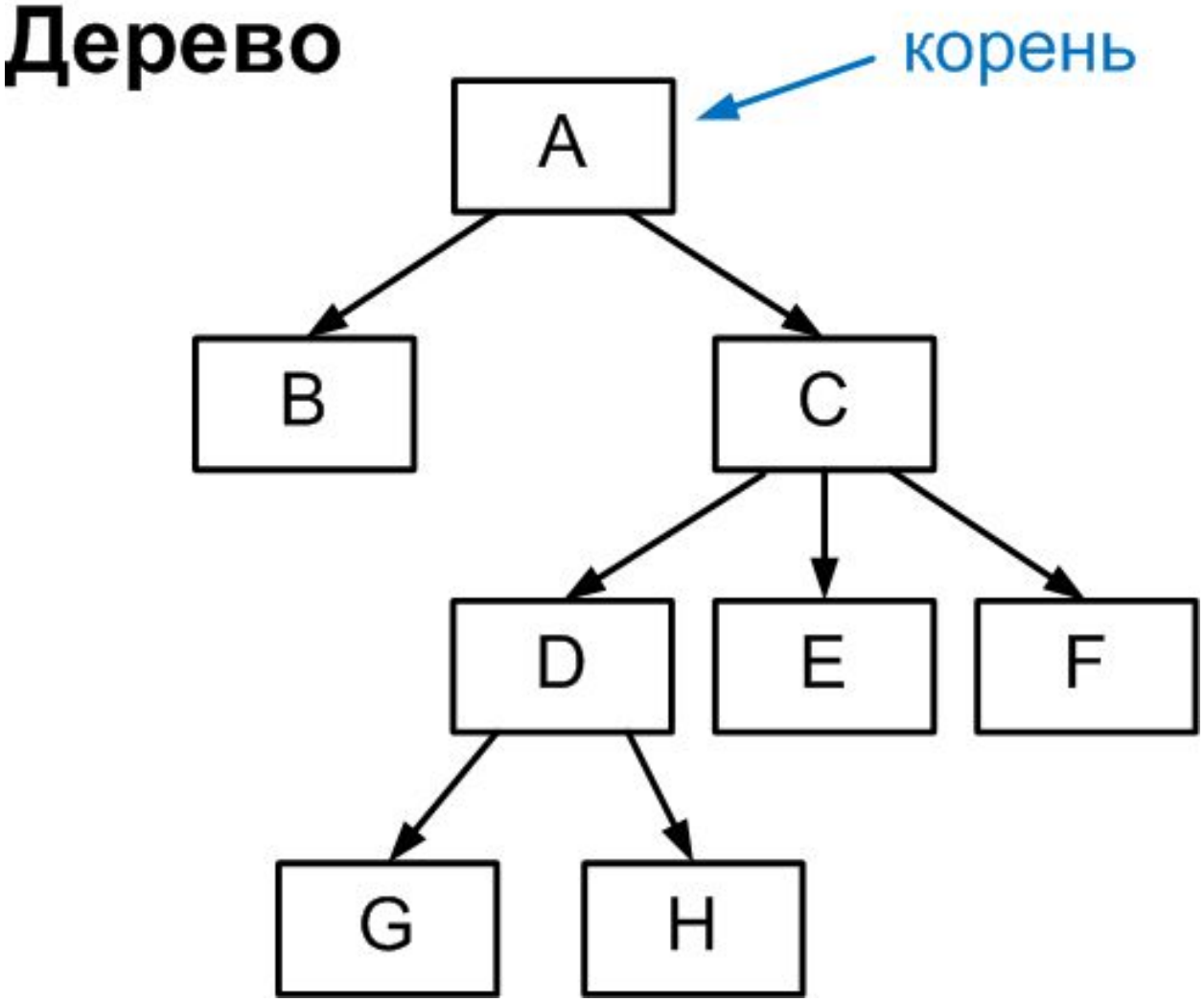
Реляционная модель



Теория графов



Иерархическая модель: дерево



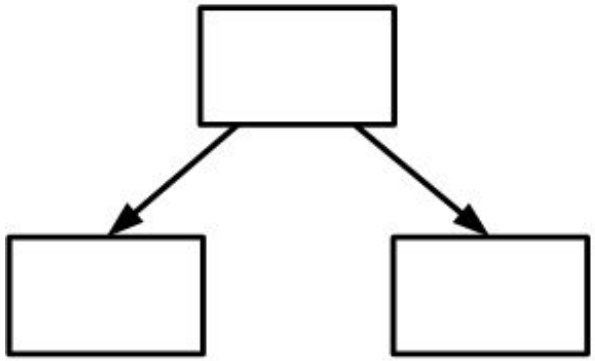
Иерархическая модель: дерево

Корневое дерево как ориентированный граф определяется следующим образом:

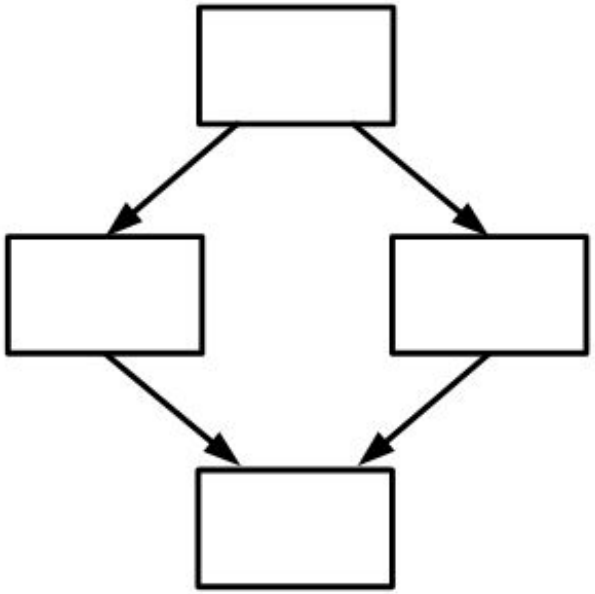
- имеется единственная особая вершина, называемая корнем, в которую не входит ни одно ребро
- во все остальные вершины входит только одно ребро, а исходит произвольное количество ребер
- нет циклов

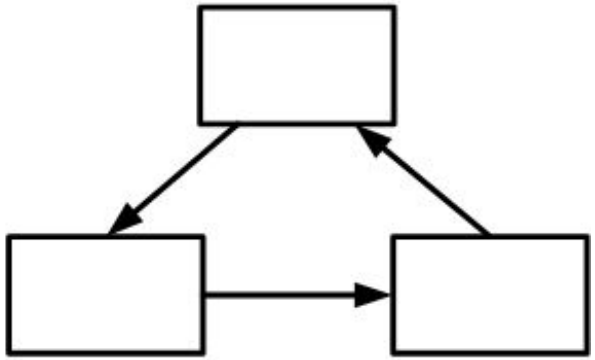
Иерархическая модель: дерево

Можно:



Нельзя:





Иерархическая модель: понятия

Основными информационными единицами в иерархической модели являются:

- **поле**
- **сегмент**
- **база данных (БД)**

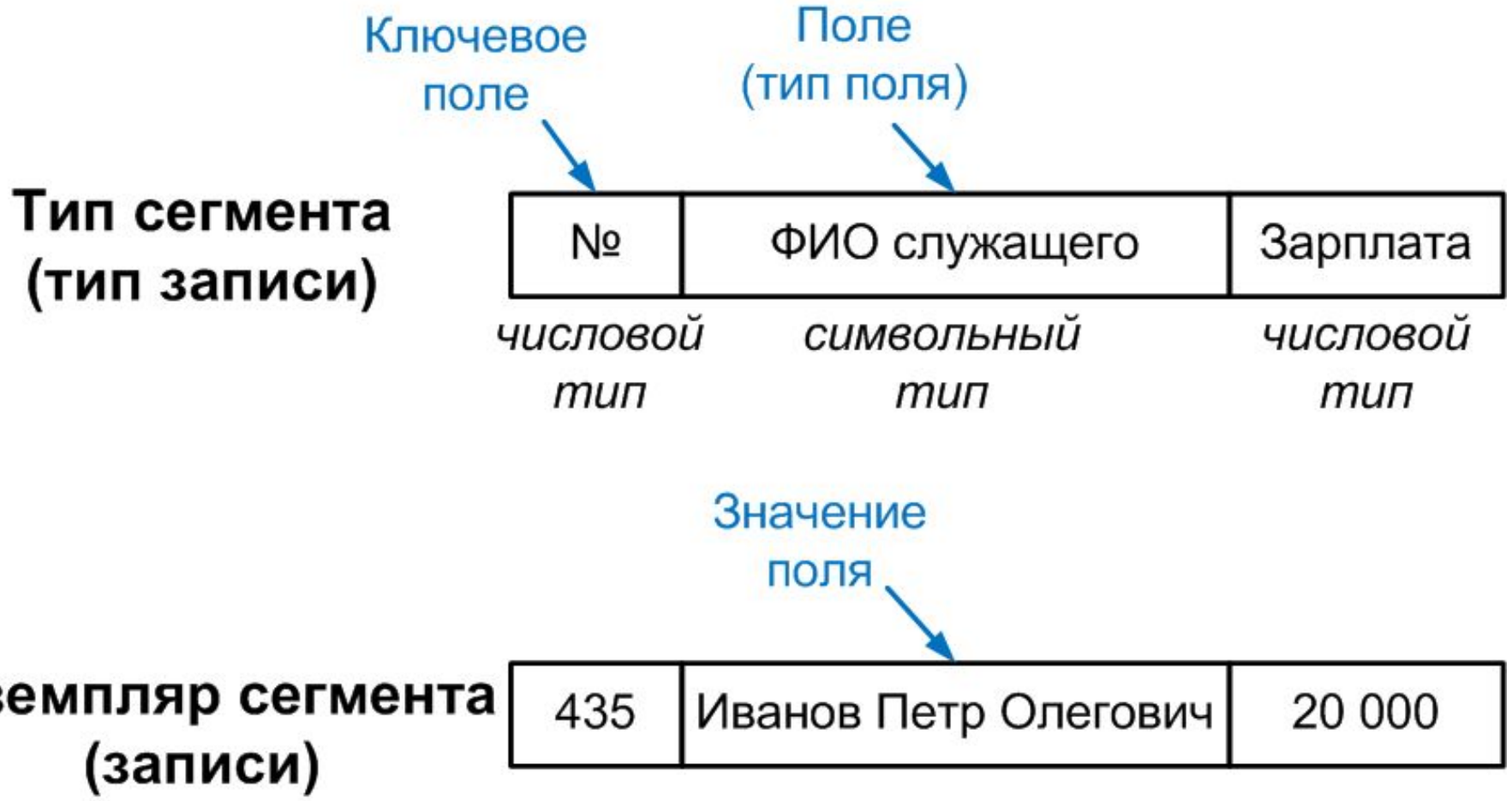
Иерархическая модель: понятия

Поле — минимальная, неделимая единица данных, доступная пользователю с помощью СУБД.

Тип сегмента — это поименованная совокупность типов полей, в него входящих.

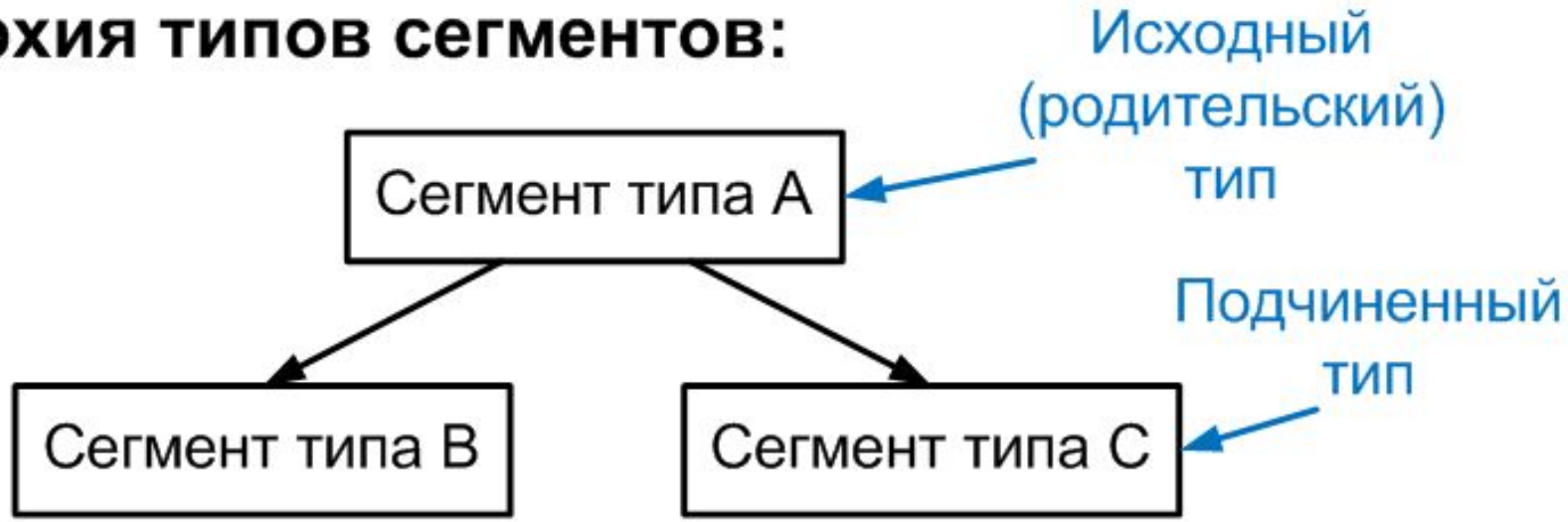
Экземпляр сегмента образуется из конкретных значений полей.

Иерархическая модель: сегменты

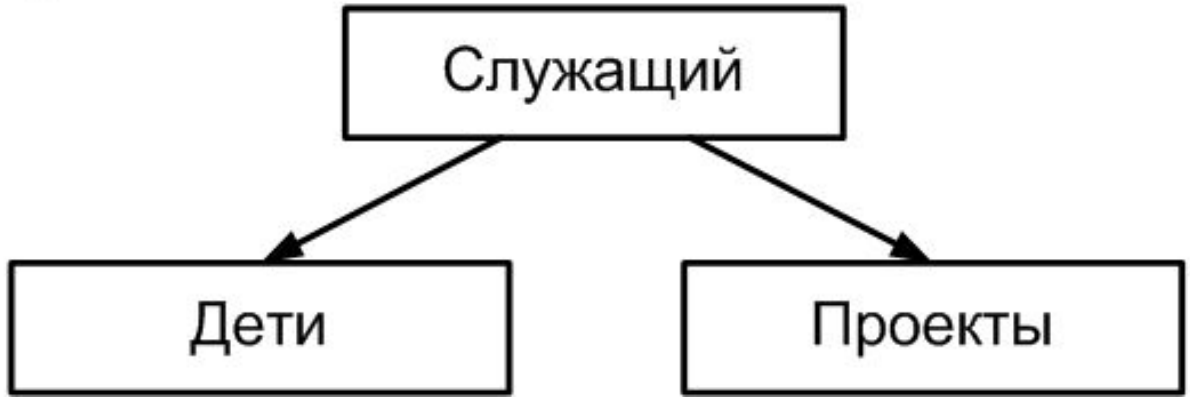


Иерархическая модель: сегменты

Иерархия типов сегментов:



Пример:

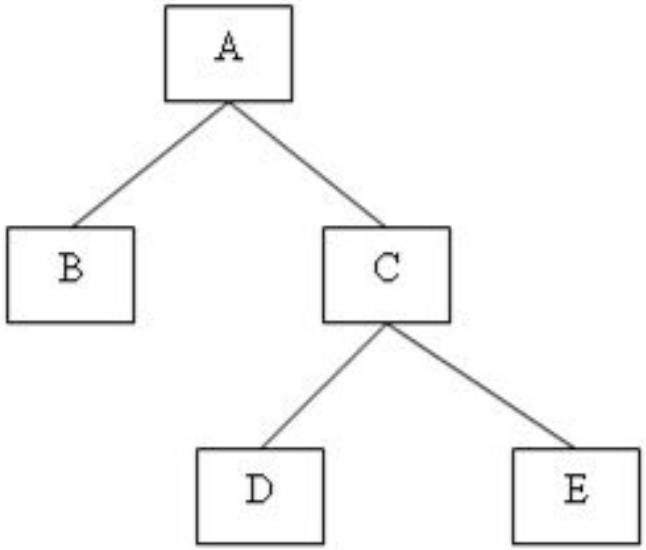


Иерархическая модель: физическая БД

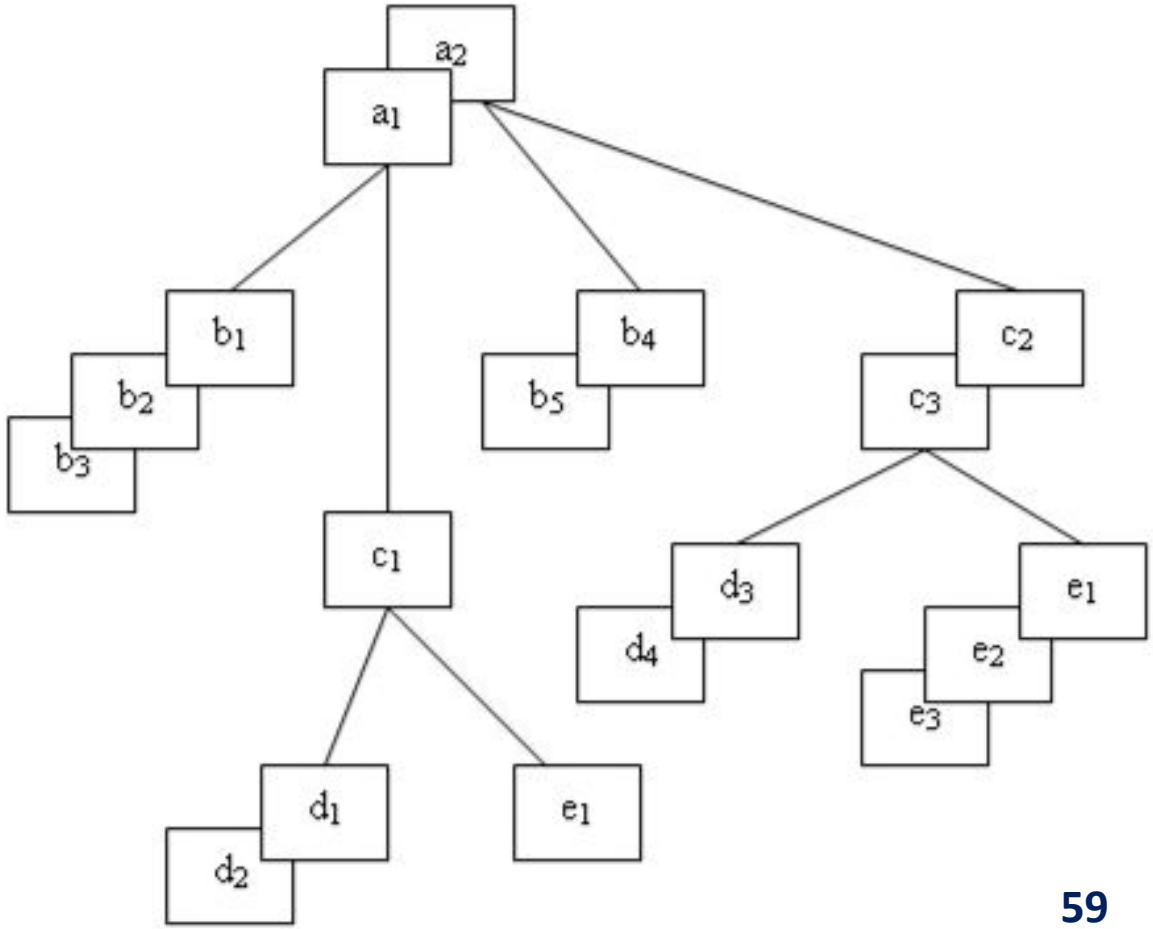
- Схема иерархической БД представляет собой совокупность отдельных деревьев (лес)
- Каждое дерево в рамках модели называется физической базой данных
- Каждая физическая БД удовлетворяет следующим иерархическим ограничениям:
  - существует один корневой сегмент;
  - каждый логически исходный сегмент может быть связан с произвольным числом подчиненных сегментов;
  - каждый логически подчиненный сегмент может быть связан только с одним логически исходным (родительским) сегментом;

Иерархическая модель: примеры

Иерархия типов сегментов:

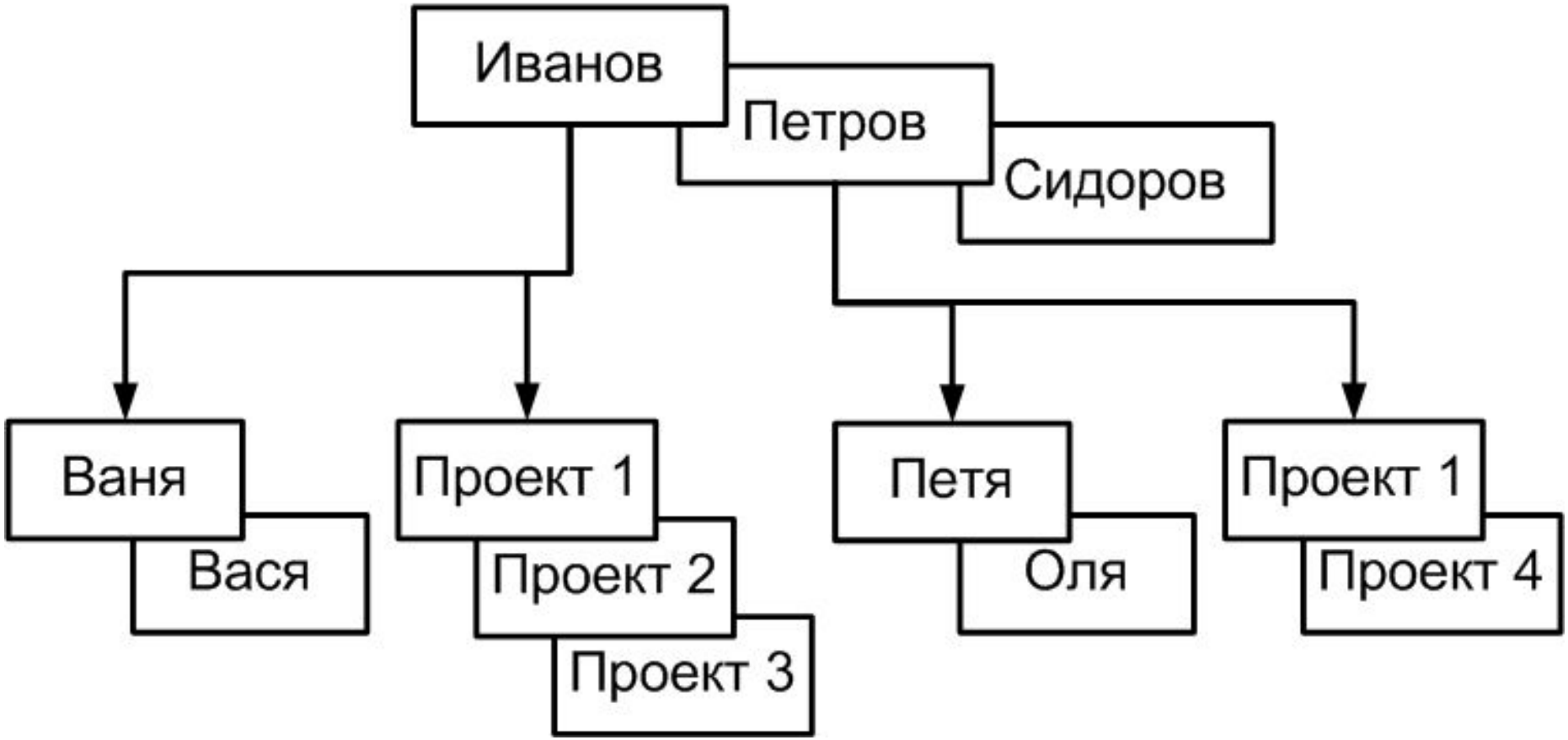


Иерархия экземпляров сегментов:

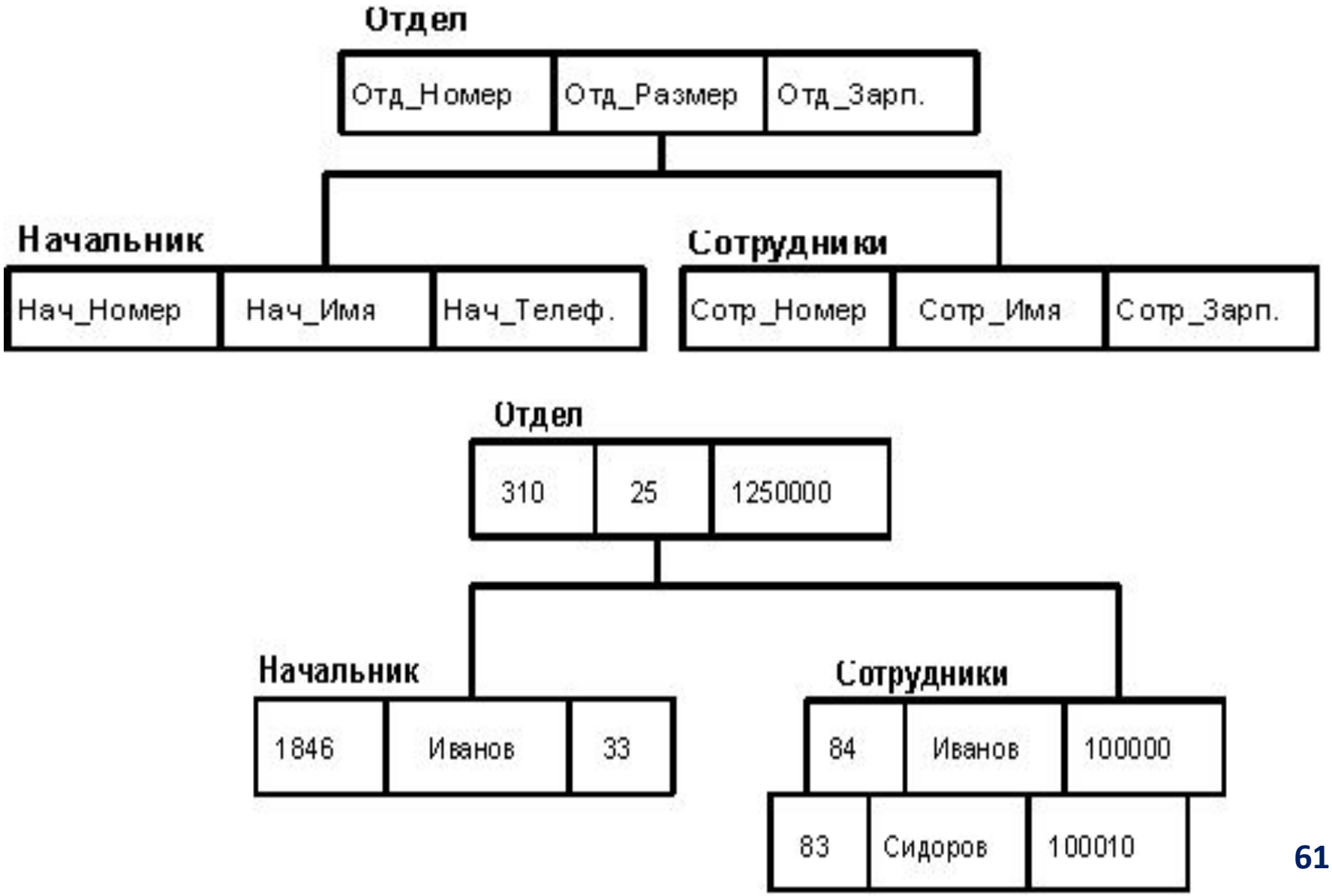


Иерархическая модель: примеры

Иерархия экземпляров сегментов:



Иерархическая модель: примеры



Иерархическая модель: операции

Примеры операция манипулирования данными:

- Найти указанное дерево БД
- Перейти от одного дерева к другому
- Перейти от одной записи к другой внутри дерева
- Перейти от одной записи к другой в порядке обхода иерархии
- Вставить новую запись в указанную позицию
- Удалить текущую запись

Иерархическая модель: операции

- Для поиска нужного сегмента используются навигационные операции — выполняется обход дерева
- При вводе данных — контекстность по вводу
- При удалении данных — контекстность по удалению

Иерархическая модель: выводы

Достоинства:

- легкость реализации
- простота и наглядность представления данных
- простота оценки характеристик БД

Недостатки:

- сложность реализации связи M:N (многие ко многим)
- сложность включения/удаления данных из-за контекстной зависимости данных

Сетевая модель: понятия

CODASYL

(Conference of Data System Languages)

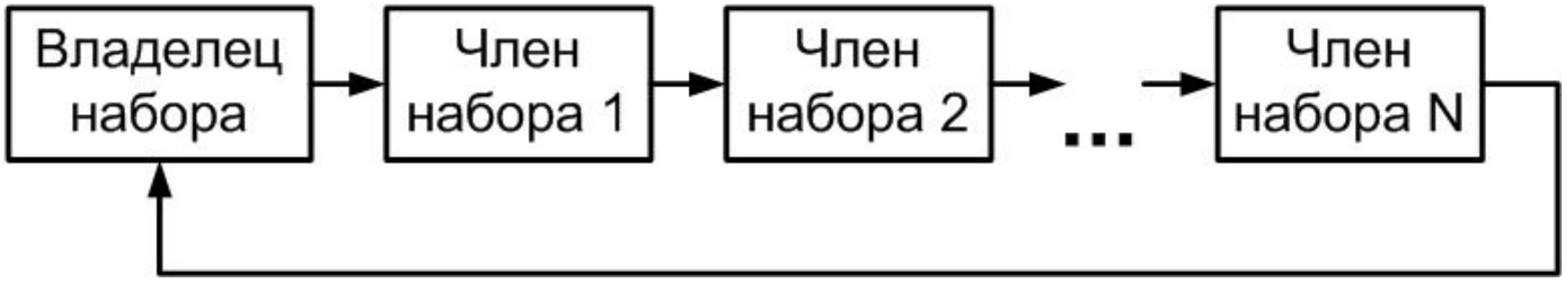
Базовые объекты сетевой модели:

- **элемент данных (= поле)**
- **запись (= сегмент)**
- **набор данных** — двухуровневый граф, связывающий отношением «один-ко-многим» (1:M) два типа записи
- **база данных**

Сетевая модель: набор

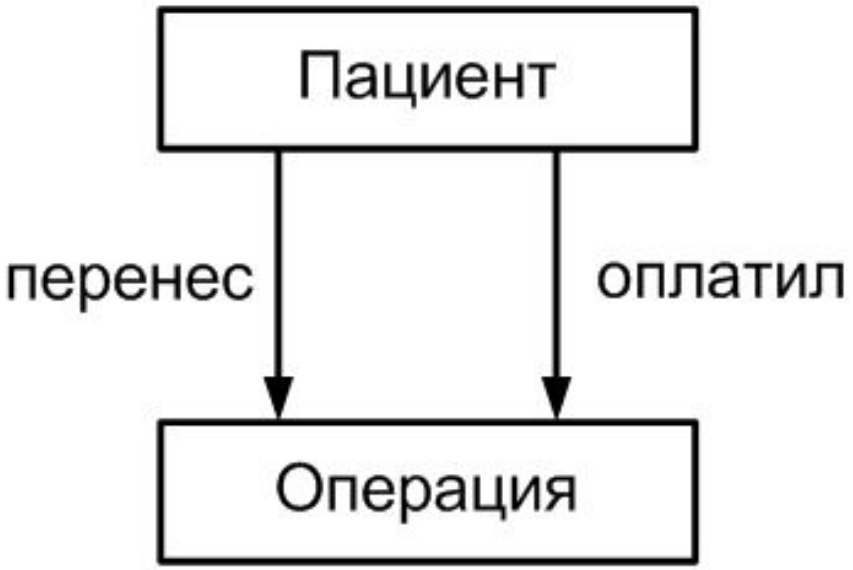
Типы записей:

Экземпляры записей:

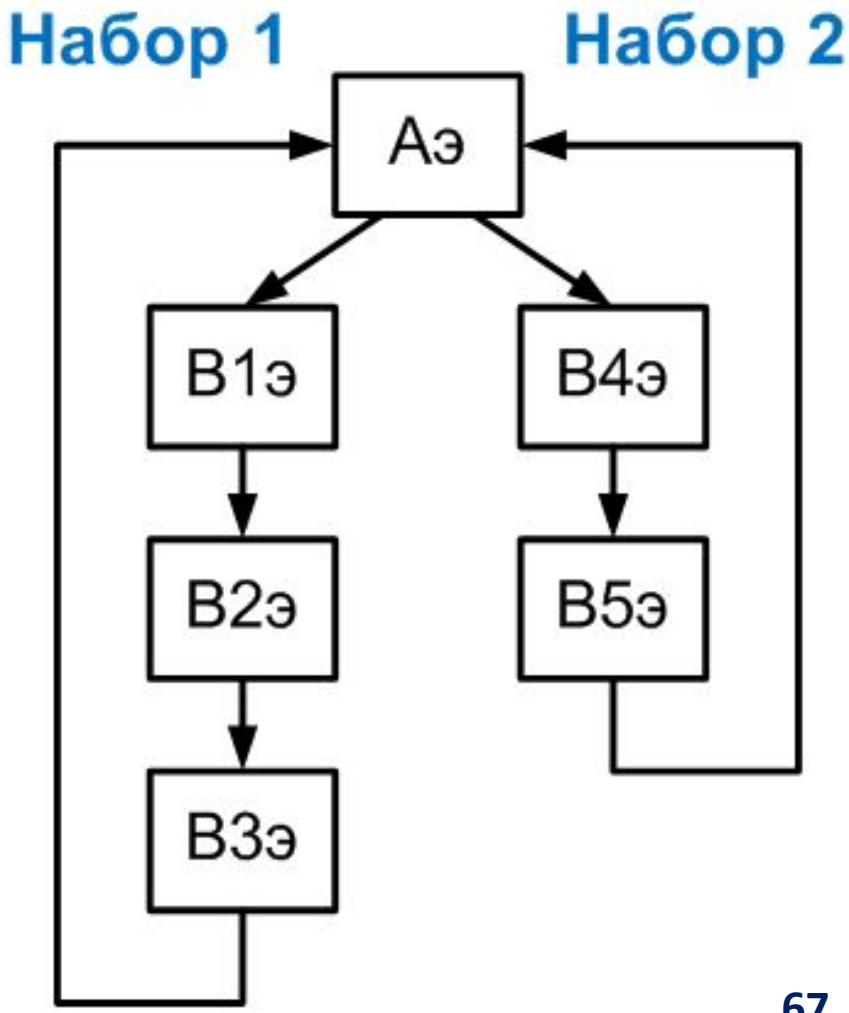


Сетевая модель: примеры

Типы записей:

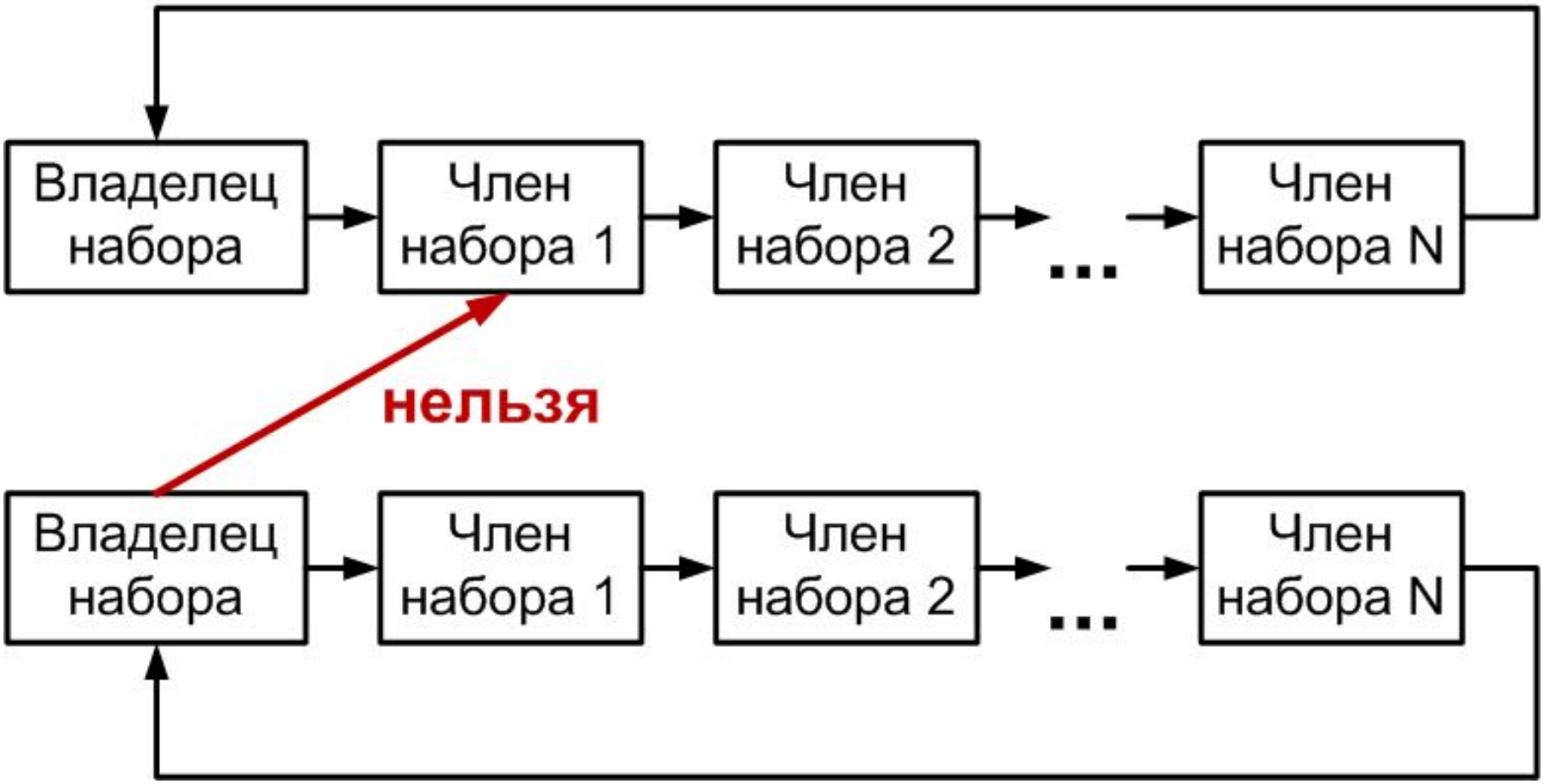


Экземпляры записей:

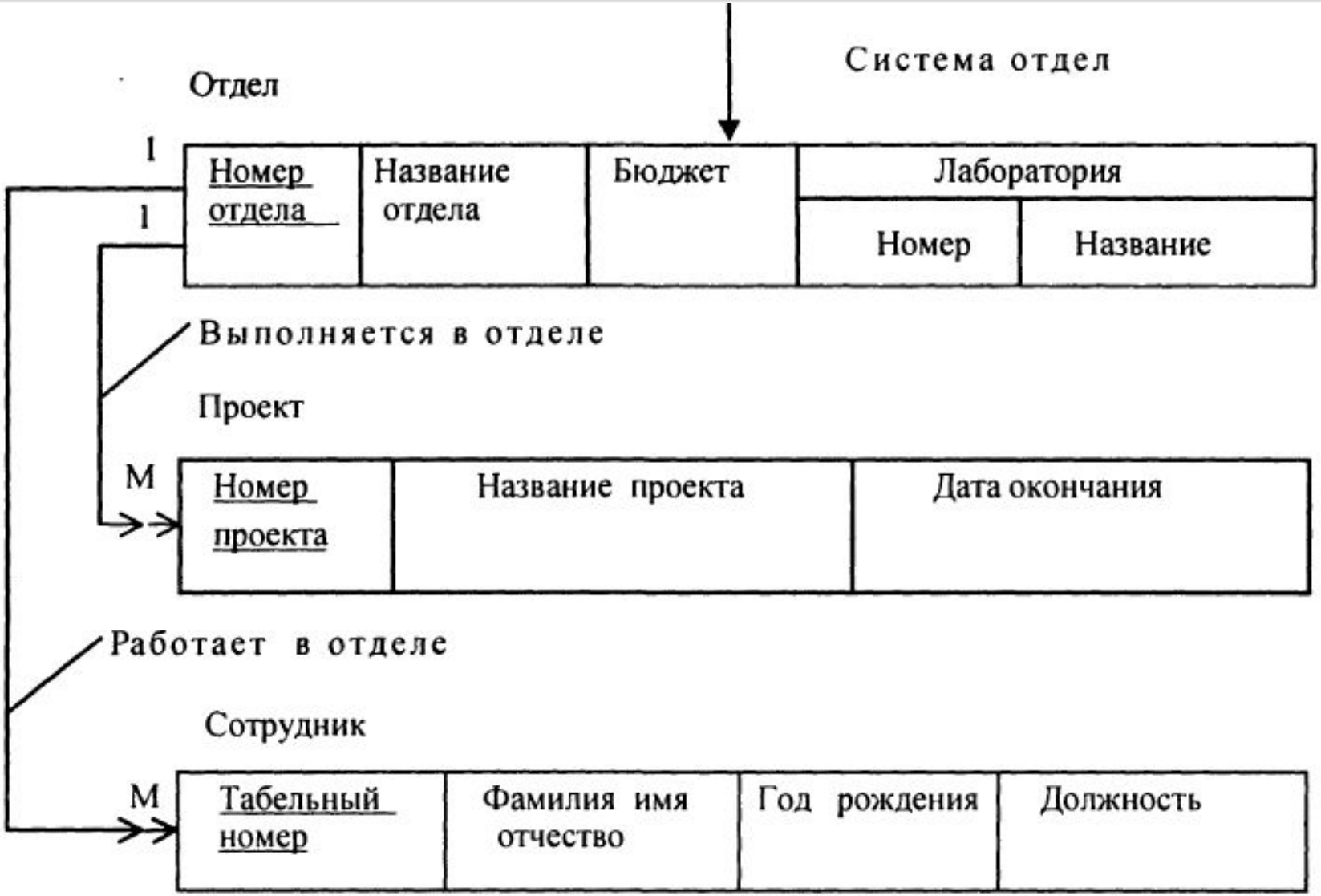


Сетевая модель: наборы

Правило уникальности владения:



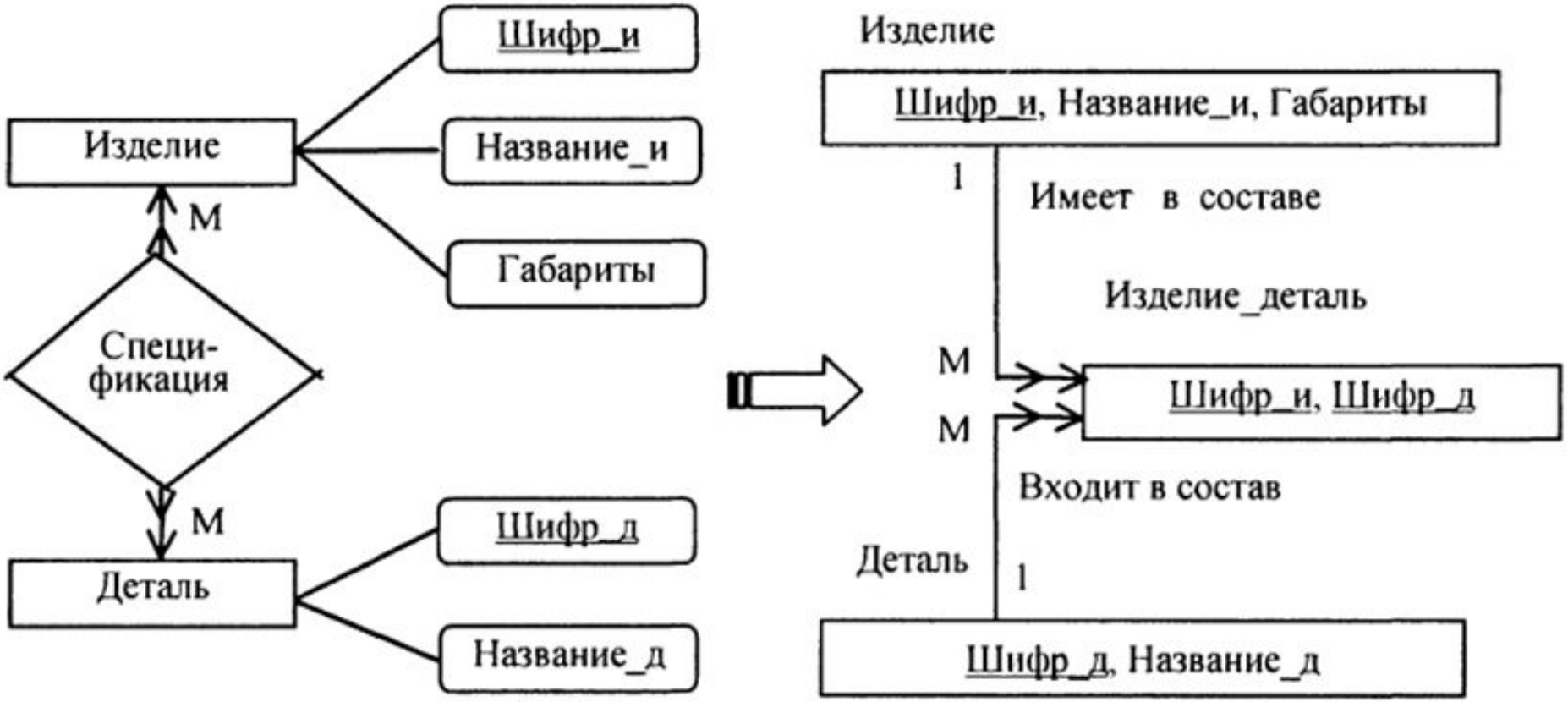
Сетевая модель: примеры



Сетевая модель: примеры

Сетевая модель: связь M:M

Реализация связи «многие ко многим» (M:M)



Сетевая модель: операции

- Сетевая модель также является навигационной
- Предусмотрены операции не только с узлами графа (записями и типами записей), но и с дугами (наборами)

Сетевая модель: операции

- Найти конкретную запись в наборе (по условию)
- Перейти от владельца набора к первому члену набора по закольцованной связи
- Перейти к следующему члену в наборе
- Перейти от члена набора к владельцу
- Создать новую запись
- Удалить запись
- Модифицировать запись
- Включить запись в набор
- Исключить запись из набора
- Переместить запись в другой набор и т.д.

Сетевая модель: выводы

Достоинства:

- простота реализации связей M:M
- поддержка любых структур данных (произвольной сложности)
- экономичность

Недостатки:

- сложность навигации

Реляционная модель

- Американский математик Э. Ф. Кодд в 1970 году впервые сформулировал основные понятия и ограничения реляционной модели
- Простота и наглядность модели и серьезное теоретическое обоснование определили большую популярность этой модели
- Основной структурой данных в модели является отношение, именно поэтому модель получила название ***реляционной*** (от английского *relation* — отношение)

Реляционная модель: аспекты

Три аспекта данных реляционной модели:

- объекты данных (структура данных)
- целостность данных
- обработка данных (реляционная алгебра)

Реляционная модель: понятия

Основные понятия реляционных БД:

- тип данных
- домен
- атрибут
- кортеж
- первичный ключ
- отношение
- схема отношения
- база данных и схема БД

Реляционная модель: домен

- Домен представляет собой именованное множество атомарных значений одного типа
- Домены являются общими совокупностями значений, из которых берутся конкретные значения **атрибутов**
- Атрибут — подмножество значений доменов, имеющие смысл для данной предметной области
- Домены ограничивают сравнения: если два атрибута определены на одном и том домене, то их можно сравнивать

Реляционная модель: отношение

- Отношение удобно представить в виде таблицы, столбцы которой соответствуют вхождениям доменов в отношение, а строки – наборам из n значений, взятых из исходных доменов, и расположенным в соответствии с заголовком отношения.
- Столбцы отношения называют атрибутами, а строки — кортежами.

Реляционная модель: отношение

Отношение содержит две части: **заголовок** и **тело**:

Заголовок — это строка заголовков столбцов.

Тело отношения — это множество строк данных.

Заголовок (или **схема отношения**) содержит фиксированное множество атрибутов или, точнее, пар **<имя-атрибута : имя-домена>**:
{<A1:D1>, <A2:D2>, …, <An:Dn>},
где n – **степень отношения**

Реляционная модель: схемы

- Схемы двух отношений называются эквивалентными, если они имеют одинаковую степень и возможно такое упорядочение имен атрибутов в схемах, что на одинаковых местах будут находиться сравнимые атрибуты (одного домена).
- Схема БД — это набор именованных схем отношений.
- Реляционная БД — это **набор отношений**, имена которых совпадают с именами схем отношений в схеме БД.

Реляционная модель: отношение

- Тело отношения содержит множество кортежей
- Каждый **кортеж** содержит множество пар **<имя-атрибута : значение-атрибута>**
- Отношение — это множество кортежей, соответствующих одной схеме отношения
- Количество кортежей называется кардинальным числом или мощностью отношения

Реляционная модель: ключи

Ключ — атрибут, значение которого однозначно идентифицирует кортежи.

- Если кортежи идентифицируются только сцеплением значений нескольких атрибутов, то отношение имеет составной ключ
- Всегда один из ключей объявляется первичным (PRIMARY KEY), его значения не могут обновляться

Реляционная модель: ключи

Основные свойства ключей:

- Уникальность
- Наличие значений (NOT NULL)

Дополнительные свойства:

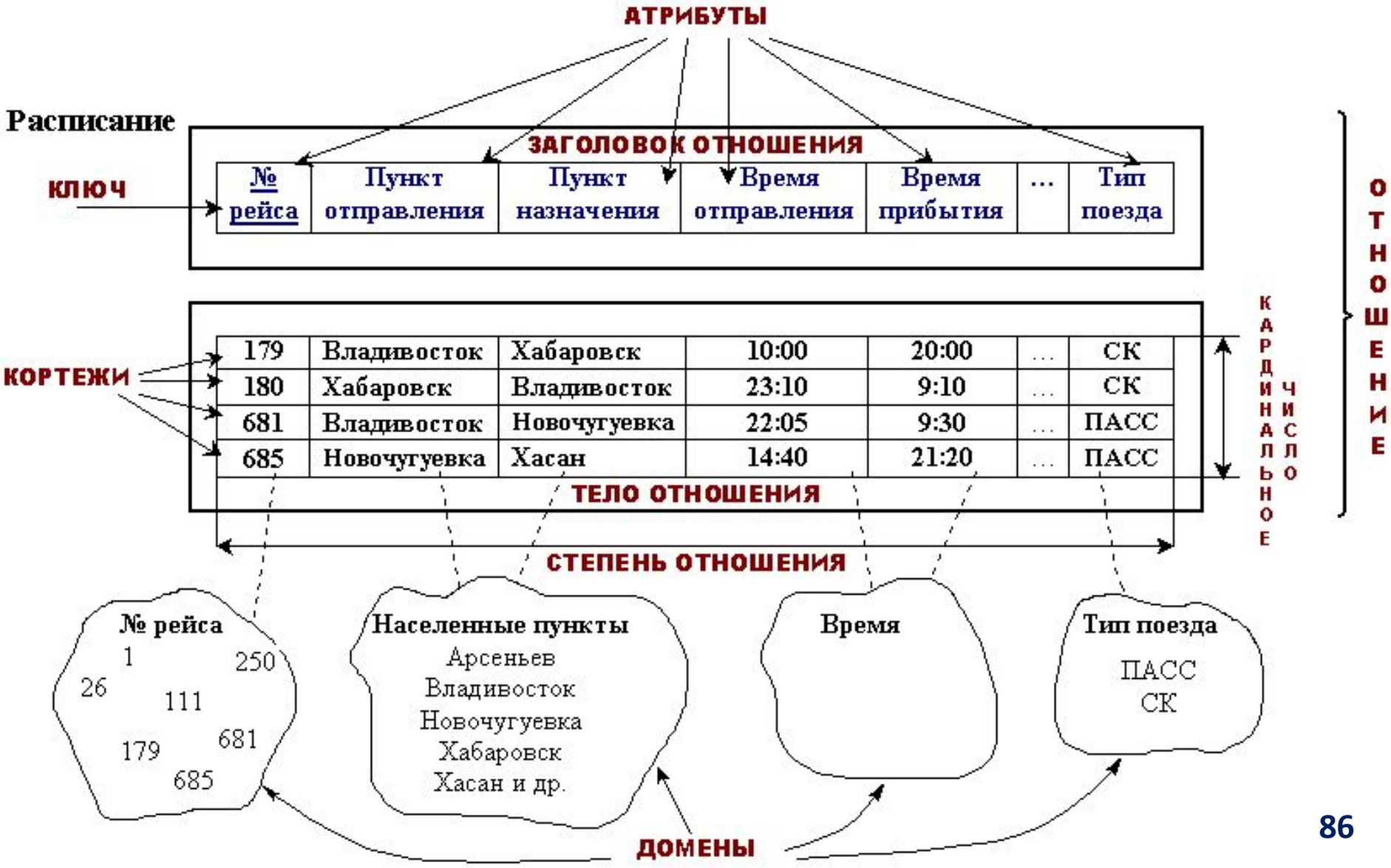
- Компактность
- Стабильность

Реляционная модель: ключи

Виды ключей:

- Естественный ключ — один или несколько атрибутов отношения, удовлетворяющие основным свойствам ключей
- Суррогатный ключ — атрибут отношения, искусственно добавляемый разработчиком для обеспечения уникальности кортежей

Реляционная модель: пример



Реляционная модель: пример

Схема отношения (заголовок):

{<№ рейса : № рейса>,

<Пункт отправления : Населенные пункты>,

<Пункт назначения : Населенные пункты>,

<Время отправления : Время>,

<Время прибытия : Время>,

<Тип поезда : Тип поезда>}

Реляционная модель: пример

Тело отношения (один из кортежей):

{<№ рейса : 681>,

<Пункт отправления : ‘Владивосток’>,

<Пункт назначения : ‘Новочугуевка’>,

<Время отправления : 22:05>,

<Время прибытия : 9:30>,

<Тип поезда : ‘ПАСС’>}

Реляционная модель: свойства

Свойства реляционных таблиц (отношений):

- Каждый элемент таблицы (атрибут) содержит один элемент данных
- Каждый столбец таблицы однороден, т. е. все элементы столбца имеют одинаковую природу (один тип данных)
- Столбцам однозначно присвоены имена
- В таблице нет двух одинаковых строк
- Строки и столбцы можно просматривать в любом порядке

Реляционная модель: свойства

Отличие обычной таблицы от отношения:

атомарность значений атрибутов

R1 – Ненормализованное отношение

КодПоставщика	КодПродукта	Продукт
P_1	1	Сахар
	2	Соль
	13	Мука
P_2	26	Рис
	58	Гречка
	130	Крупа манная
	162	Пшено
P_3	474	Молоко
	891	Кефир

R2 – Нормализованное отношение

КодПоставщика	КодПродукта	Продукт
P_1	1	Сахар
P_1	2	Соль
P_1	13	Мука
P_2	26	Рис
P_2	58	Гречка
P_2	130	Крупа манная
P_2	162	Пшено
P_3	474	Молоко
P_3	891	Кефир

Реляционная модель: пример

Пример использования суррогатных

Таблица Clients Name Address

Name	Address	City	State	Country
Gerry Farish	127 Caesar Chavez St.	San Francisco	CA	USA
Celia Brock	246 3rd St.	Sonoma	CA	USA
Julio Abregon	778 Avenida Tarragon	Patzcauro	MICH	MEX

Таблица Clients с первичным ключом

Id_Num	Name	Address	City	State	Country
1007	Gerry Farish	127 Caesar Chavez St	San Francisco	CA	USA
1008	Celia Brock	246 3rd St.	Sonoma	CA	USA
1010	Julio Abregon	778 Avenida Tarragon	Patzcauro	MICH	MEX

Реляционная модель: связи (задача)

Задача: требуется добавить к таблице Clients столбец с номерами телефонов. Большинство людей имеют несколько телефонных номеров (домашний, сотовый, рабочий).

Противоречие свойству атомарности атрибутов
либо избыточность данных

Необходимость второй таблицы

Реляционная модель: связи (примеры)

Первый вариант добавления информации о телефонах:

Таблица Clients

Id_Num	Name	Address	City	State	Country	Phone	Type
1007	Gerry Farish	127 Caesar Chavez St	San Francisco	CA	USA	4156479772	home
1008	Celia Brock	246 3rd St.	Sonoma	CA	USA	7074568232	work
						7079402092	home
1010	Julio Abregon	778 Avenida Tarragon	Patzcauro	MICH	MEX		

Нарушение атомарности атрибутов

Второй вариант добавления информации о телефонах:

Таблица Clients

Id_Num	Name	Address	City	State	Country	Phone	Type
1007	Gerry Farish	127 Caesar Chavez St	San Francisco	CA	USA	4156479772	home
1008	Celia Brock	246 3rd St.	Sonoma	CA	USA	7074568232	work
1009	Celia Brock	246 3rd St.	Sonoma	CA	USA	7079402092	home
1010	Julio Abregon	778 Avenida Tarragon	Patzcauro	MICH	MEX		

Избыточность данных

Реляционная модель: связи (пример)

Основное отношение:

Родительский ключ

Таблица Clients с первичным ключом

Id_Num	Name	Address	City	State	Country
1007	Gerry Farish	127 Caesar Chavez St	San Francisco	CA	USA
1008	Celia Brock	246 3rd St.	Sonoma	CA	USA
1010	Julio Abregon	778 Avenida Tarragon	Patzcauro	MICH	MEX

Подчиненное отношение:

Внешний ключ

Таблица Client_Phone

Id_Num	Phone	Type
1007	4156479772	home
1008	7074568232	work
1008	7079402092	home

Реляционная модель: связи (понятия)

- Реляционная модель представляет базу данных в виде множества взаимосвязанных отношений
- В каждой связи одно отношение может выступать как основное (родительское), а другое отношение выступает в роли подчиненного
- Один кортеж основного отношения может быть связан с несколькими кортежами подчиненного отношения

Реляционная модель: связи (понятия)

- В основном отношении для связи используется родительский ключ (parent) отношения
- В качестве родительского ключа обычно выступает первичный ключ (PRIMARY KEY) основного отношения
- В подчиненном отношении используется набор атрибутов, соответствующий первичному ключу основного отношения — внешний ключ (FOREIGN KEY)

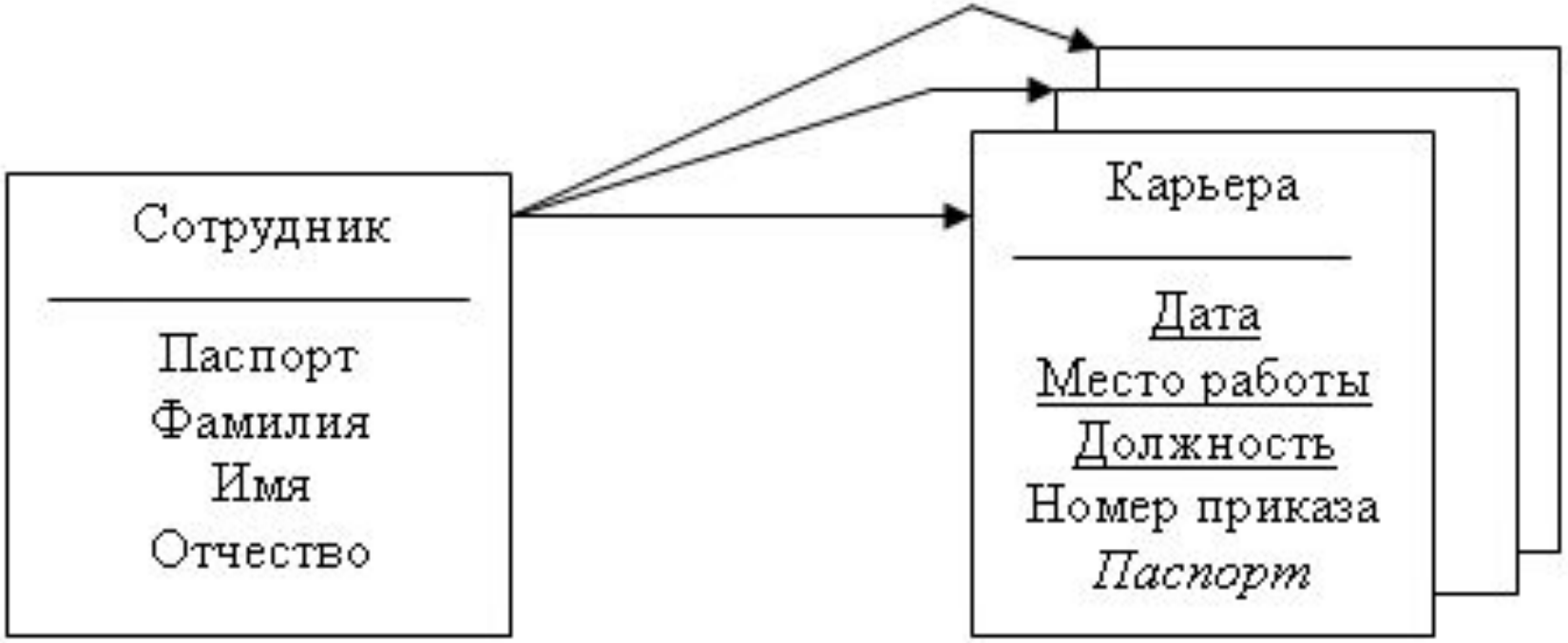
Реляционная модель: связи (типы)

Типы связей:

- **«один ко многим» (1:M)**
- «один к одному» (1:1)
- «многие ко многим» (M:N)

Реляционная модель: связи (пример)

PRIMARY KEY отношения «Сотрудник» атрибут ***Паспорт*** является FOREIGN KEY для отношения «Карьера»



Реляционная модель: целостность

Целостность данных — правильность данных в любой момент времени при манипулировании данными:

- структурная целостность
- языковая целостность
- ссылочная целостность
- семантическая целостность

Реляционная модель: целостность

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

Требования структурной целостности:

- при добавлении кортежей в отношение проверяется уникальность их первичных ключей
- не допускается, чтобы какой-либо атрибут, участвующий в первичном ключе, принимал неопределенное значение (NOT NULL)

Реляционная модель: целостность

Языковая целостность состоит в том, что реляционная СУБД должна обеспечивать языки описания и манипулирования данными не ниже стандарта SQL

Требование языковой целостности:

не должны быть доступны иные низкоуровневые средства манипулирования данными, не соответствующие стандарту

Реляционная модель: целостность

Ссылочная целостность обеспечивает поддержание целостности по ссылкам при установлении связи между отношениями

Требование ссылочной целостности:

для каждого значения внешнего ключа, появляющегося в подчиненном отношении, в основном отношении должен существовать кортеж с таким же значением родительского ключа

Реляционная модель: целостность

Требование ссылочной целостности:
то есть значение внешнего ключа должно либо:

- быть равным значению родительского ключа
- быть полностью неопределенным (NULL)

Реляционная модель: целостность

Для каждого внешнего ключа нужно решить:

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

Реляционная модель: целостность

При удалении возможно три варианта:

- Каскадирование удаления
- Ограничение удаления
- Установка неопределенных значений для внешнего ключа при удалении

Реляционная модель: целостность

3. Что произойдет при попытке ОБНОВЛЕНИЯ родительского ключа основного отношения, на который ссылается некоторый внешний ключ подчиненного отношения?

При обновлении также возможно три варианта:

- Каскадирование обновления
- Ограничение обновления
- Установка неопределенных значений для

Реляционная модель: целостность

Семантическая целостность задается разработчиком посредством задания ограничений для свойств атрибутов

Виды ограничений:

- уникальность значений
- обязательность заполнения
- значение по умолчанию
- вхождение в диапазон значений
- принадлежность набору значений

Реляционная модель: операции

Реляционная алгебра: совместимость по типу

Два отношения **совместимы по типу**, если у них эквивалентные схемы:

- если каждое из них имеет одно и то же множество атрибутов
- если возможно такое упорядочение атрибутов в схемах, что на одинаковых местах будут находиться сравнимые атрибуты

Реляционная алгебра: совместимость по типу

Продукты1

КодПродукта	Продукт	КодПоставщика
1	Сахар	P_1
2	Соль	P_1
13	Мука	P_1
26	Рис	P_2
58	Гречка	P_2
130	Крупа манная	P_2
162	Пшено	P_2
474	Молоко	P_3
891	Кефир	P_3

Продукты2

КодПродукта	Продукт	КодПоставщика
26	Рис	P_2
35	Перловка	P_2
58	Гречка	P_2
130	Крупа манная	P_2
162	Пшено	P_2
200	Крупа ячневая	P_2

Поставщики

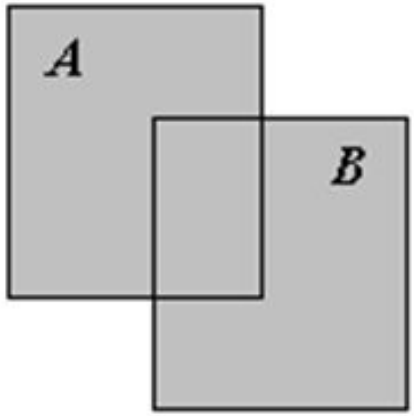
КодП	Наименование	Город
P_1	ООО «Восток»	Владивосток
P_2	ОАО «Приморье»	Уссурийск
P_3	ПБОЮЛ Сидоров А.С.	Находка
P_4	ОАО «Владхлеб»	Владивосток

ВидПродукта

КодВида	Вид
1	Молочная
2	Мясная
3	Хлебопродукция

Реляционная алгебра: объединение

Объединение: $A \cup B$



Объединением двух совместимых по типу отношений A и B называется отношение, содержащее все кортежи, принадлежащие или одному из двух определенных отношений, или обоим.

Реляционная алгебра: объединение

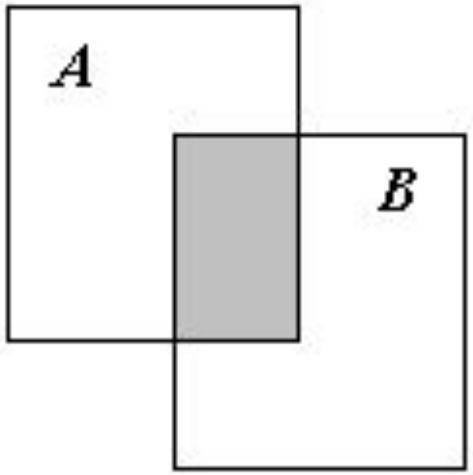
Пример $R1 = Продукты1 \cup Продукты2$

R1

КодПродукта	Продукт	КодПоставщика
1	Сахар	P_1
2	Соль	P_1
13	Мука	P_1
26	Рис	P_2
58	Гречка	P_2
130	Крупа манная	P_2
162	Пшено	P_2
474	Молоко	P_3
891	Кефир	P_3
35	Перловка	P_2
200	Крупа ячневая	P_2

Реляционная алгебра: пересечение

Пересечение $A \cap B$



Пересечением двух совместимых по типу отношений A и B называется отношение, содержащее все кортежи, принадлежащие одновременно двум определенным отношениям.

Реляционная алгебра: пересечение

Пример:

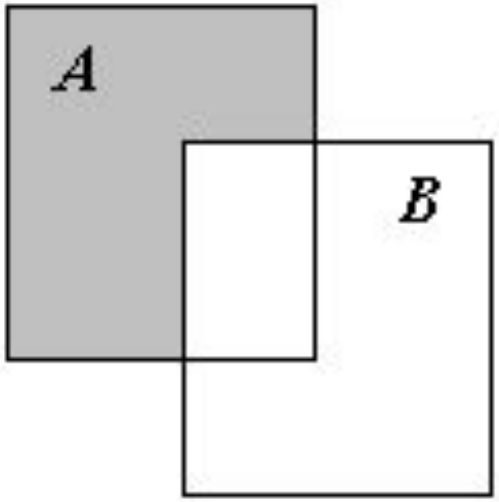
$R2 = Продукты1 \cap Продукты2$

R2

КодПродукта	Продукт	КодПоставщика
26	Рис	P_2
58	Гречка	P_2
130	Крупа манная	P_2
162	Пшено	P_2

Реляционная алгебра: вычитание

Вычитание $A \setminus B$



Вычитанием двух совместимых по типу отношений A и B называется отношение, содержащее все кортежи, которые принадлежат первому из двух определенных отношений и не принадлежат второму.

Реляционная алгебра: вычитание

Примеры: R3 = Продукты1 \ Продукты2

R3

КодПродукта	Продукт	КодПоставщика
1	Сахар	P_1
2	Соль	P_1
13	Мука	P_1
474	Молоко	P_3
891	Кефир	P_3

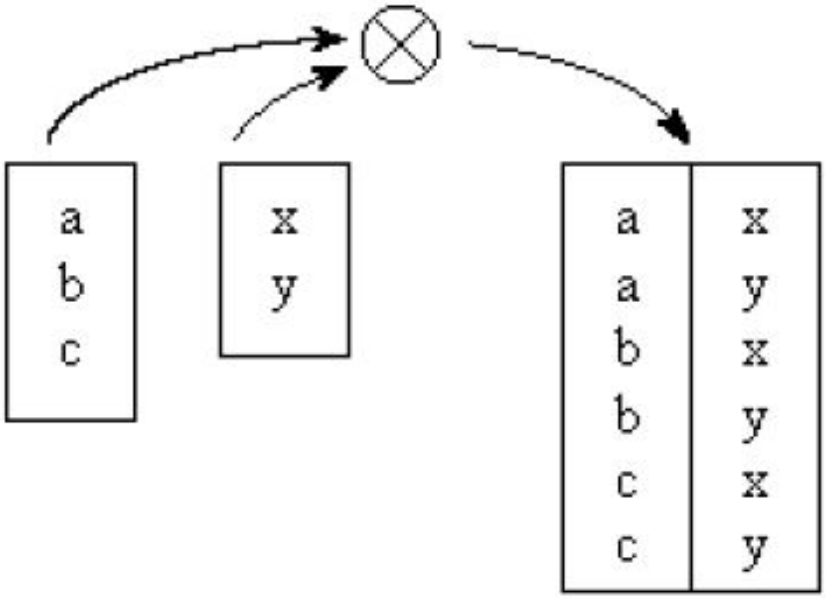
R4 = Продукты2 \ Продукты1

R4

КодПродукта	Продукт	КодПоставщика
35	Перловка	P_2
200	Крупа ячневая	P_2

Реляционная алгебра: декартово пр-е

Декартово произведение: $A \otimes B$



Декартовым произведением двух отношений A и B называется отношение, содержащее всевозможные кортежи, являющиеся сочетанием двух кортежей, принадлежащих соответственно двум определенным отношениям.

Реляционная алгебра: декартово пр-е

Пример:

R5 = Поставщики ⊗ ВидПродукта

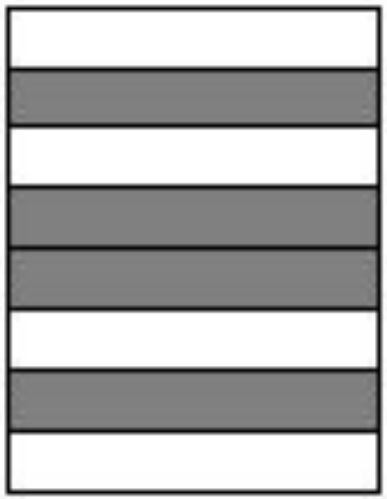
R5

КодП	Наименование	Город	КодВида	Вид
P_1	ООО «Восток»	Владивосток	1	Молочная
P_1	ООО «Восток»	Владивосток	2	Мясная
P_1	ООО «Восток»	Владивосток	3	Хлебопродукция
P_2	ОАО «Приморье»	Уссурийск	1	Молочная
P_2	ОАО «Приморье»	Уссурийск	2	Мясная
P_2	ОАО «Приморье»	Уссурийск	3	Хлебопродукция
P_3	ПБОЮЛ Сидоров А.С.	Находка	1	Молочная
P_3	ПБОЮЛ Сидоров А.С.	Находка	2	Мясная
P_3	ПБОЮЛ Сидоров А.С.	Находка	3	Хлебопродукция
P_4	ОАО «Владхлеб»	Владивосток	1	Молочная
P_4	ОАО «Владхлеб»	Владивосток	2	Мясная
P_4	ОАО «Владхлеб»	Владивосток	3	Хлебопродукция

Реляционная алгебра: ограничение

Ограничение $A[\alpha]$

(выборка, фильтрация)



Ограничением, заданным на отношении A в виде условного выражения α, называется отношение, содержащее все кортежи из определенного отношения, удовлетворяющие определенным условиям.

Реляционная алгебра: ограничение

Примеры:

a) R6 = Продукты1 [КодПоставщика = "Р3"]

R6

КодПродукта	Продукт	КодПоставщика
474	Молоко	Р3
891	Кефир	Р3

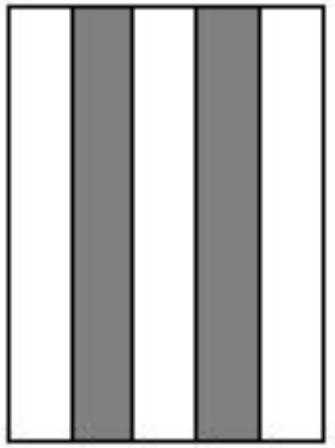
b) R7 = Поставщики [Город = "Владивосток"]

R7

КодП	Наименование	Город
P_1	ООО «Восток»	Владивосток
P_4	ОАО «Владхлеб»	Владивосток

Реляционная алгебра: проекция

Проекция $A[X, Y, \ldots, Z]$



Проекцией отношения A по атрибутам X, Y, …, Z, где каждый из атрибутов принадлежит отношению A, называется отношение, содержащее все кортежи определенного отношения после исключения из него некоторых атрибутов.

Реляционная алгебра: проекция

Примеры:

a) R8 = Продукты1 [КодПоставщика]

R8

КодПоставщика
P_1
P_2
P_3

b) R9 = Поставщики [Город]

R9

Город
Владивосток
Уссурийск
Находка

c) R7 = Поставщики [Город = “Владивосток”]
R10 = R7 [Наименование]
или R10 = (Поставщики [Город = “Владивосток”]) [Наименование]

R7

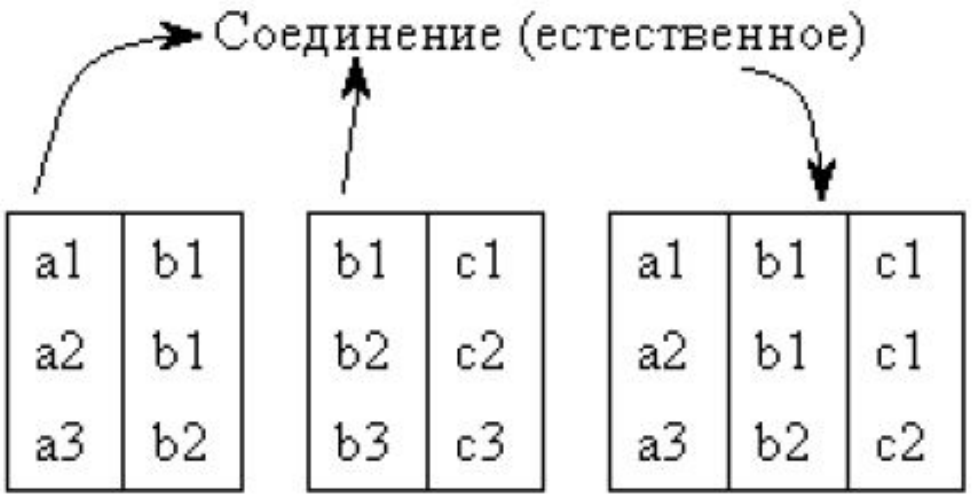
КодП	Наименование	Город
P_1	ООО «Восток»	Владивосток
P_4	ОАО «Владхлеб»	Владивосток

R10

Наименование
ООО «Восток»
ОАО «Владхлеб»

Реляционная алгебра: соединение

Естественное соединение:



Естественным соединением отношений A и B, имеющим один или несколько общих атрибутов, называется отношение, кортежи которого – это сочетание двух кортежей (принадлежащих соответственно двум определенным отношениям), имеющих общее значение для одного или нескольких атрибутов этих двух отношений.

Реляционная алгебра: соединение

Пример естественного соединения:

R11 = Продукты1 [Продукты1.КодПоставщика = Поставщики.КодПоставщика] Поставщики

R11

КодПродукта	Продукт	КодПоставщика	Наименование	Город
1	Сахар	P_1	ООО «Восток»	Владивосток
2	Соль	P_1	ООО «Восток»	Владивосток
13	Мука	P_1	ООО «Восток»	Владивосток
26	Рис	P_2	ОАО «Приморье»	Уссурийск
58	Гречка	P_2	ОАО «Приморье»	Уссурийск
130	Крупа манная	P_2	ОАО «Приморье»	Уссурийск
162	Пшено	P_2	ОАО «Приморье»	Уссурийск
474	Молоко	P_3	ПБОЮЛ Сидоров А.С.	Находка
891	Кефир	P_3	ПБОЮЛ Сидоров А.С.	Находка

Реляционная алгебра: соединение

Пример условного соединения:

R12 = Продукты1 [(Продукты1.КодПоставщика = Поставщики.КодП) ∧
Поставщики.Город = “Владивосток”] Поставщики
R13 = R12 [Продукт]
или
R13 = (Продукты1 [(Продукты1.КодПоставщика = Поставщики.КодП) ∧
Поставщики.Город = “Владивосток”] Поставщики) [Продукт]

R12

КодПродукта	Продукт	КодПоставщика	КодП	Наименование	Город
1	Сахар	P_1	P_1	ООО «Восток»	Владивосток
2	Соль	P_1	P_1	ООО «Восток»	Владивосток
13	Мука	P_1	P_1	ООО «Восток»	Владивосток

R13

Продукт
Сахар
Соль
Мука

Реляционная модель: замкнутость

Свойство замкнутости операций реляционной алгебры:

Результат каждой операции над отношением также является **отношением**.

Вывод: поскольку результат любой операции имеет тот же тип, что и исходные объекты (отношения), то результат одной операции может использоваться в качестве исходных данных для другой

Реляционная модель: выводы

Достоинства:

- простота и наглядность представления
- простота проектирования и программирования
- гибкость
- теоретическое обоснование
- защищенность данных (независимость таблиц)

Недостатки:

- реализация неполного набора операций
- необходимость использования оптимизаторов запросов
- ограниченность возможностей для представления сложных структур данных

Тема 4. Проектирование баз данных

1. Жизненный цикл БД
2. Этапы проектирования БД
3. Системный анализ предметной области
4. Инфологическое моделирование предметной области. Модель «сущность-связь»
5. Даталогическое проектирование. Переход от модели «сущность-связь» к реляционной модели. Принципы нормализации

Жизненный цикл баз данных

Этапы проектирования БД

Системный анализ предметной области

Цель: провести подробное словесное описание объектов предметной области и реальных связей между объектами

- **Функциональный подход** — реализует принцип движения «от задач» , когда заранее известны необходимые функции
- **Предметный подход** — когда информационные потребности будущих пользователей БД жестко не фиксируются

Системный анализ предметной области

Системный анализ должен включать:

- подробное описание информации об **объектах предметной области**
- формулировку **конкретных задач** с кратким описанием алгоритмов их решения
- **описание выходных документов**, которые должны генерироваться в системе
- **описание входных документов**, которые служат основанием для заполнения данными БД

Пример описания предметной области

Задача: требуется разработать ИС для автоматизации учета получения и выдачи книг в библиотеке

Основные объекты:

- книги и экземпляры книг
- читатели
- выдачи книг на руки

Пример описания предметной области

Параметры, характеризующие каждую книгу:

- уникальный шифр
- название
- фамилии авторов (могут отсутствовать)
- место издания (город)
- издательство
- год издания
- количество страниц
- стоимость книги
- область знаний
- количество экземпляров книги в библиотеке

Пример описания предметной области

На каждого читателя в картотеку заносятся следующие сведения:

- уникальный номер читательского билета
- фамилия, имя, отчество
- домашний адрес
- телефон
- дата рождения

Пример описания предметной области

Каждый экземпляр книги имеет:

- уникальный инвентарный номер
- шифр книги, который совпадает с уникальным шифром из описания книг
- место размещения в библиотеке

При выдаче экземпляра книги читателю заносятся следующие сведения:

- номер билета читателя, который взял книгу
- дата выдачи книги
- дата возврата

Пример описания предметной области

Предусмотреть следующие **ограничения** :

- Книга может не иметь ни одного автора
- В библиотеке должны быть записаны читатели не моложе 17 лет
- В библиотеке присутствуют книги, изданные начиная с 1960 по текущий год
- Каждый читатель может держать на руках не более 5 книг
- Каждый читатель при регистрации в библиотеке должен дать телефон для связи
- Каждая область знаний может содержать ссылки на множество книг, но каждая книга может относиться к различным областям знаний

Пример описания предметной области

С данной ИС должны работать следующие группы пользователей:

- библиотекари
- читатели
- администрация библиотеки

Затем необходимо определить, какие задачи будет решать каждый пользователь (или группа пользователей)

Инфологическое моделирование

- Инфологическое проектирование связано с представлением семантики предметной области в модели базы данных
- Инфологическое описание не должно быть привязано к конкретной СУБД
- Инфологическая (семантическая) модель представляет собой емкое формализованное описание предметной области

Модель «сущность-связь»

Модель «сущность-связь»

(Entity-Relationship model, ER-модель)

- ER-модель является концептуальной моделью, т.е. не учитывает особенности конкретной СУБД
- Из модели могут быть получены все основные фактографические модели данных
- Процесс создания модели является итерационным (уточняющим)

Модель «сущность-связь»: понятия

В основе ER-модели лежат следующие базовые понятия:

- **Сущности**
- **Атрибуты**
- **Связи**

Модель «сущность-связь»: сущность

Сущность — это реальный или представляемый объект, информация о котором должна сохраняться в проектируемой системе

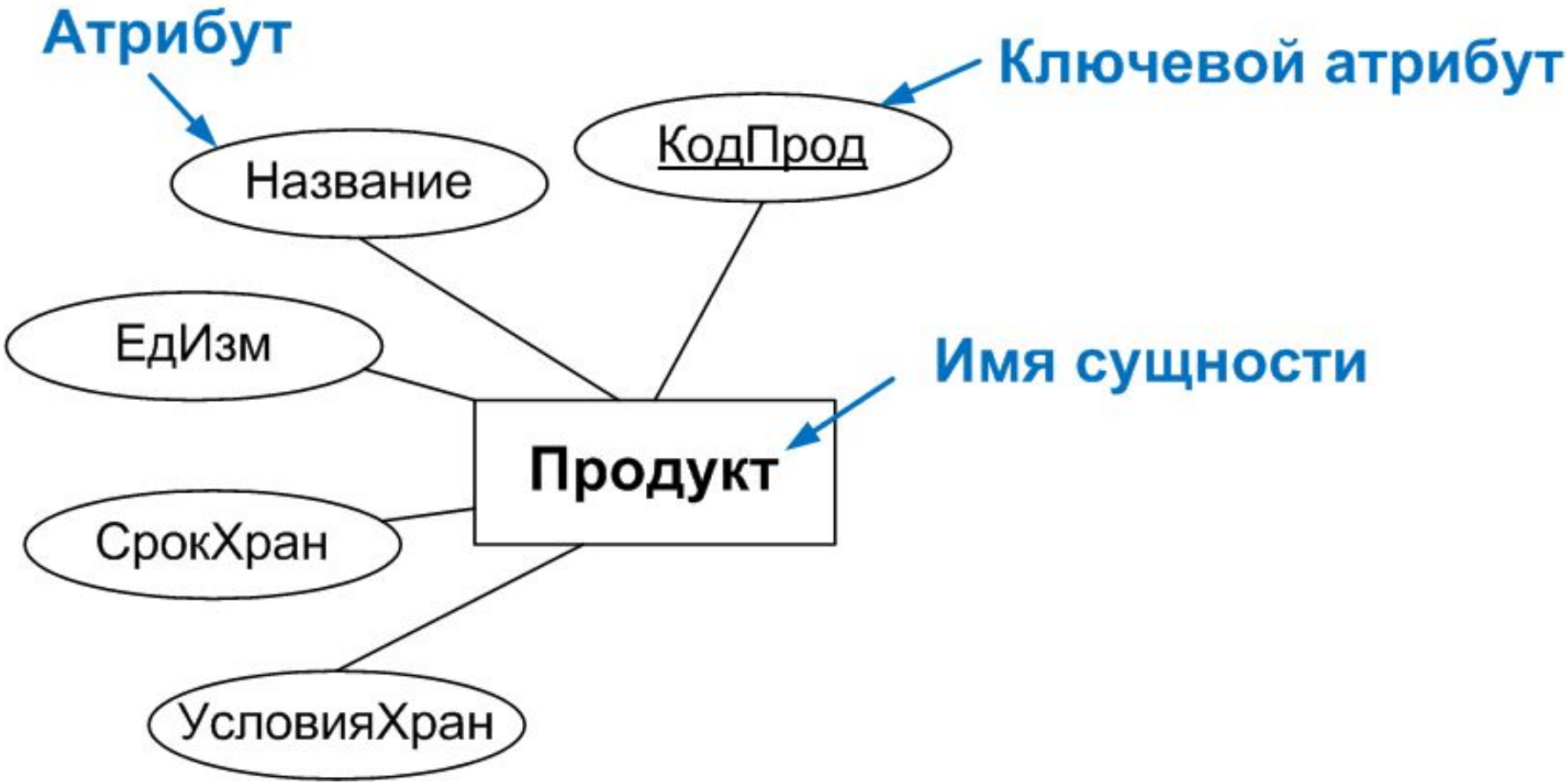
- Сущность имеет имя, **уникальное** в пределах системы
- Сущность соответствует некоторому классу однотипных объектов (существует множество экземпляров данной сущности)

Модель «сущность-связь»: атрибуты

- Объект имеет свой набор атрибутов — характеристик, определяющих свойства данного объекта
- Атрибут должен иметь имя, **уникальное** в пределах данной сущности
- Ключ сущности — это минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности

Модель «сущность-связь»: сущность

Модель «сущность-связь»: сущность



Модель «сущность-связь»: связь

Связь — это ассоциация, установленная между несколькими сущностями и показывающая, как взаимодействуют сущности между собой

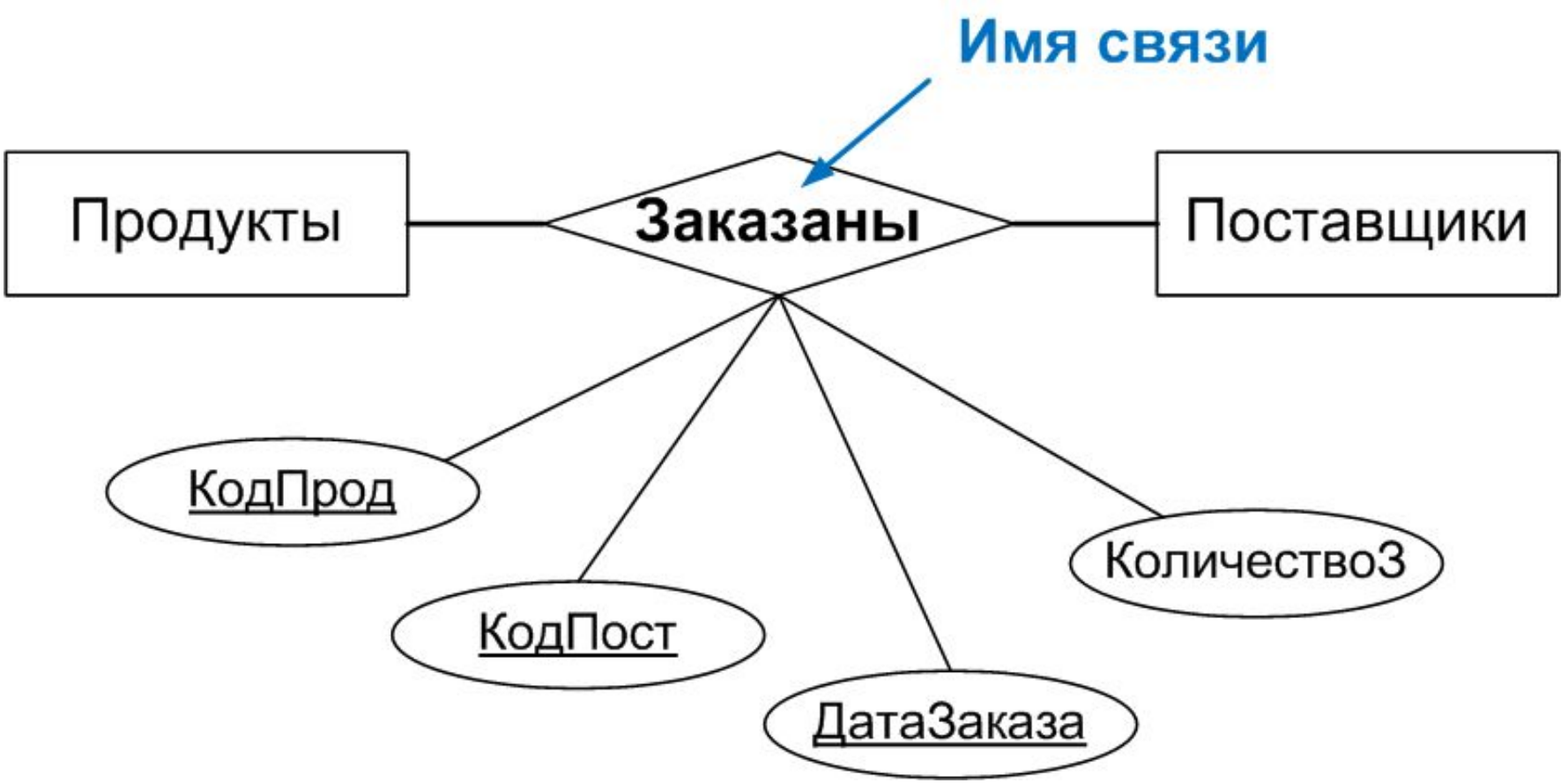
- Связь определяет взаимосвязь между **экземплярами сущностей**
- Связь также может иметь **атрибуты**
- Между сущностями может быть задано сколько угодно связей с разными смысловыми нагрузками

Модель «сущность-связь»: связь

Связь может существовать:

- между двумя разными сущностями (**бинарная связь**)
- между n сущностями (**n-арная связь**)
- между сущностью и ей же самой (**рекурсивная связь**)

Модель «сущность-связь»: связь



Модель «сущность-связь»: связь

Степень связи — число экземпляров сущностей, которое может быть ассоциировано через связь с экземплярами другой сущности

Модель «сущность-связь»: связь

Степени бинарных связей:

- один-к-одному (1:1)

- один-ко-многим (1:M)

- многие-ко-многим (M:N)

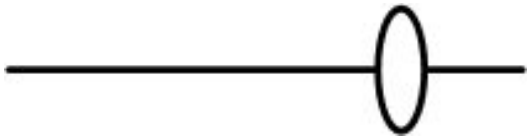
Модель «сущность-связь»: связь

Класс принадлежности входящих в связь сущностей:

- Связь любого из типов может быть обязательной, если в данной связи должен участвовать каждый экземпляр сущности
- Связь любого из типов может быть необязательной, если не каждый экземпляр сущности должен участвовать в данной связи

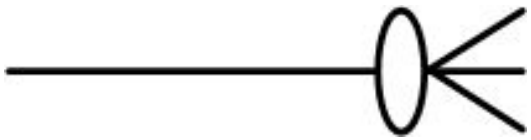
Модель «сущность-связь»: связь

- Связь степени 1, необязательный класс

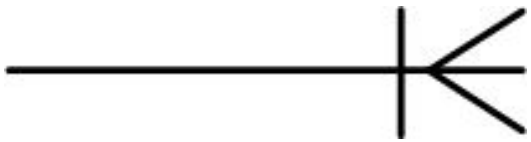


- Связь степени 1, обязательный класс

- Связь степени N, необязательный класс

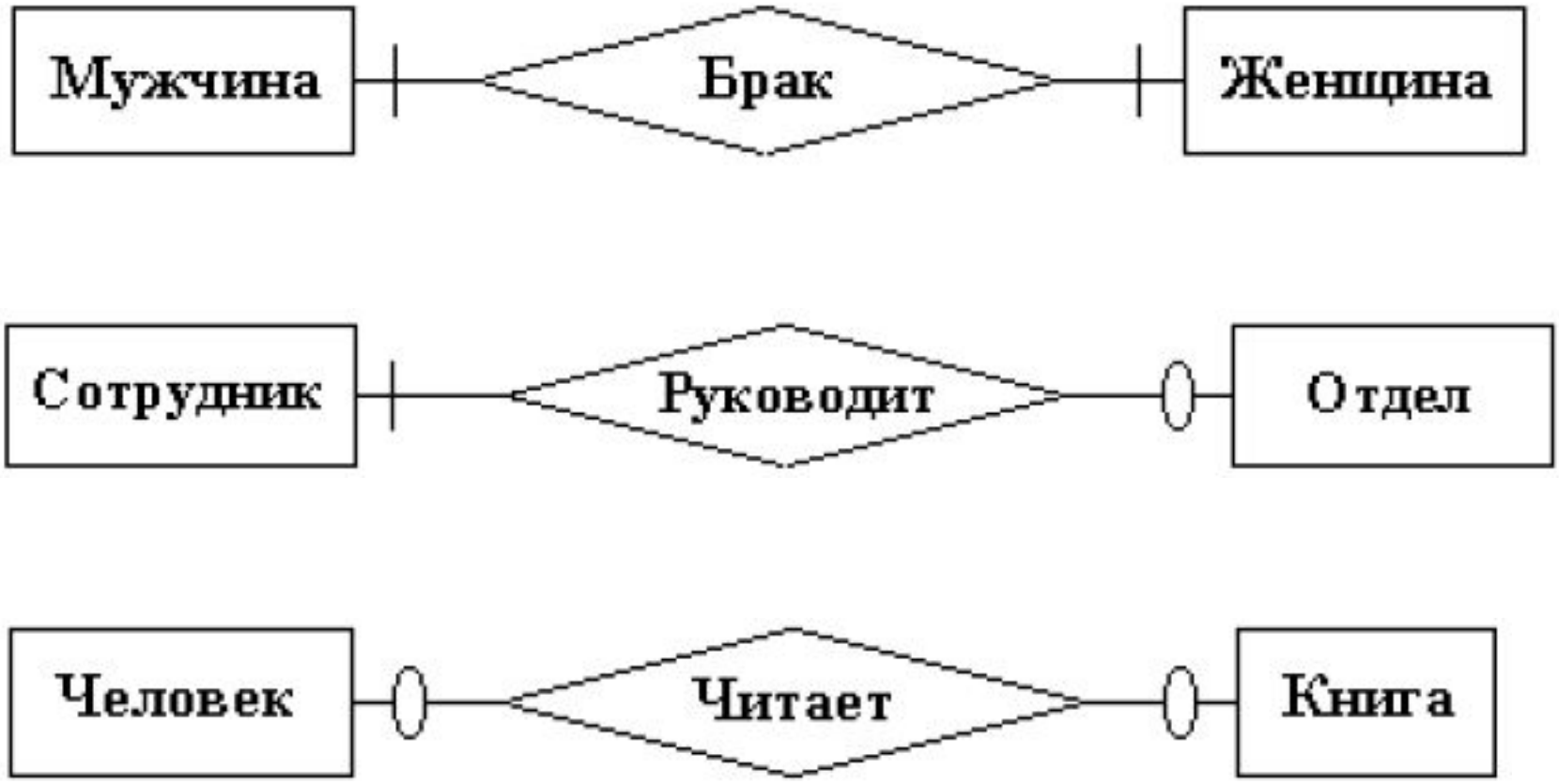


- Связь степени N, обязательный класс



Модель «сущность-связь»: примеры

Примеры связей один-к-одному:



Модель «сущность-связь»: примеры

Примеры связей один-ко-многим:

Модель «сущность-связь»: связь

Если существование сущности **x** зависит от существования сущности **y**, то **x** называется [зависимой](#) сущностью

Модель «сущность-связь»: примеры

Примеры связей многие-ко-многим:

Между одними и теми же сущностями могут существовать **несколько** связей:

Модель «сущность-связь»: построение

Этапы построения диаграммы «сущность-связь»:

1. Определение списка сущностей выбранной предметной области
2. Определение списка атрибутов сущностей
3. Описание связей между сущностями (степени, классы принадлежности связей, а также атрибуты связей, если они необходимы)
4. Организация данных в виде диаграммы "сущность-связь"

Модель «сущность-связь»: пример

Задача: построить диаграмму, отображающую связь данных для информационной системы учета продажи продуктов в магазине.

БД должна хранить информацию:

- о **продуктах**, поставляемых в магазин
- об ежедневной **продаже** продуктов
- о **заказах на поставку** продуктов
- о **поставщиках** продуктов

Модель «сущность-связь»: пример

Составим список сущностей с их атрибутами:

1. Сущность «**Продукты**»
   - *Код продукта* – уникальный идентификатор, ключевой атрибут
   - *Продукт* – название продукта
   - *Единица измерения* – литры, килограммы, штуки и т.п.
   - *Срок хранения в днях* – для определения даты окончания срока годности продукта
   - *Условия хранения* – температура, влажность и т.п.

Модель «сущность-связь»: пример

2. Сущность «**Поставщики**»
   - *Код поставщика* – уникальный идентификатор, ключевой атрибут
   - *Поставщик* – название организации или ФИО физического лица
   - *Код города* – город, где находится поставщик (для поиска)
   - *Адрес* – улица и дом (а также квартира – для физического лица)
   - *ФИО директора*
   - *Телефон*
   - *Факс*

Модель «сущность-связь»: пример

3. Сущность «**Продажи**»
   - *Дата продажи*
   - *Код продукта* – какой именно продукт был продан
   - *Количество* – сколько продано этого продукта в тех единицах измерения, которые указаны для этого продукта в сущности Продукт
   - *Цена продажи* – цена при продаже за единицу продукта

Модель «сущность-связь»: пример

4. Сущность «**Города**»
   - *Код города* – уникальный идентификатор, ключевой атрибут
   - *Город* – название города

Модель «сущность-связь»: пример

Рассмотрим связи, существующие между сущностями:

1. Связь M:N «**Поставляют**» между сущностями Продукты и Поставщики

Модель «сущность-связь»: пример

Связь «**Поставляют**» имеет следующие атрибуты:

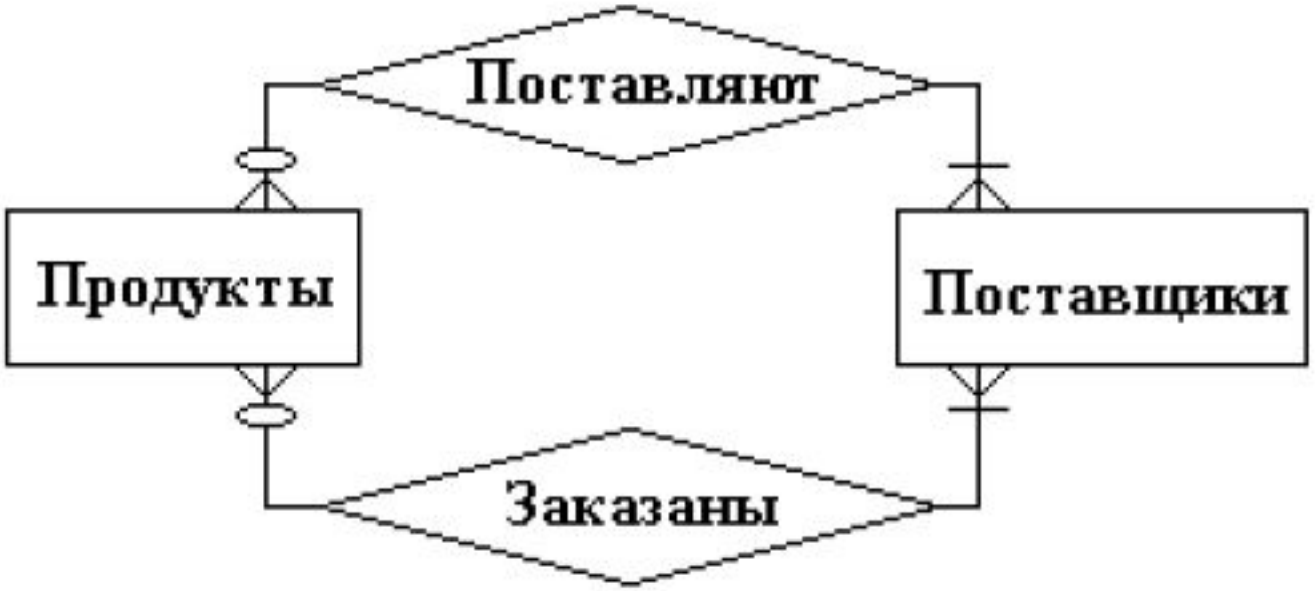
- *Дата поставки*
- *Код поставщика* – какой поставщик поставил этот продукт
- *Код продукта* – какой именно продукт был поставлен
- *КоличествоП* – сколько поставлено этого продукта
- *Цена поставки* – цена при поставке за единицу продукта
- *Дата изготовления* – дата изготовления продукта

Модель «сущность-связь»: пример

2. Связь M:N «**Заказаны**» между сущностями Продукты и Поставщики
   - *Дата заказа*
   - *Код поставщика* – какому поставщику заказан этот продукт
   - *Код продукта* – какой именно продукт был заказан
   - *КоличествоЗ* – сколько поставлено этого продукта

Модель «сущность-связь»: пример

Связи между сущностями Продукты и Поставщики:

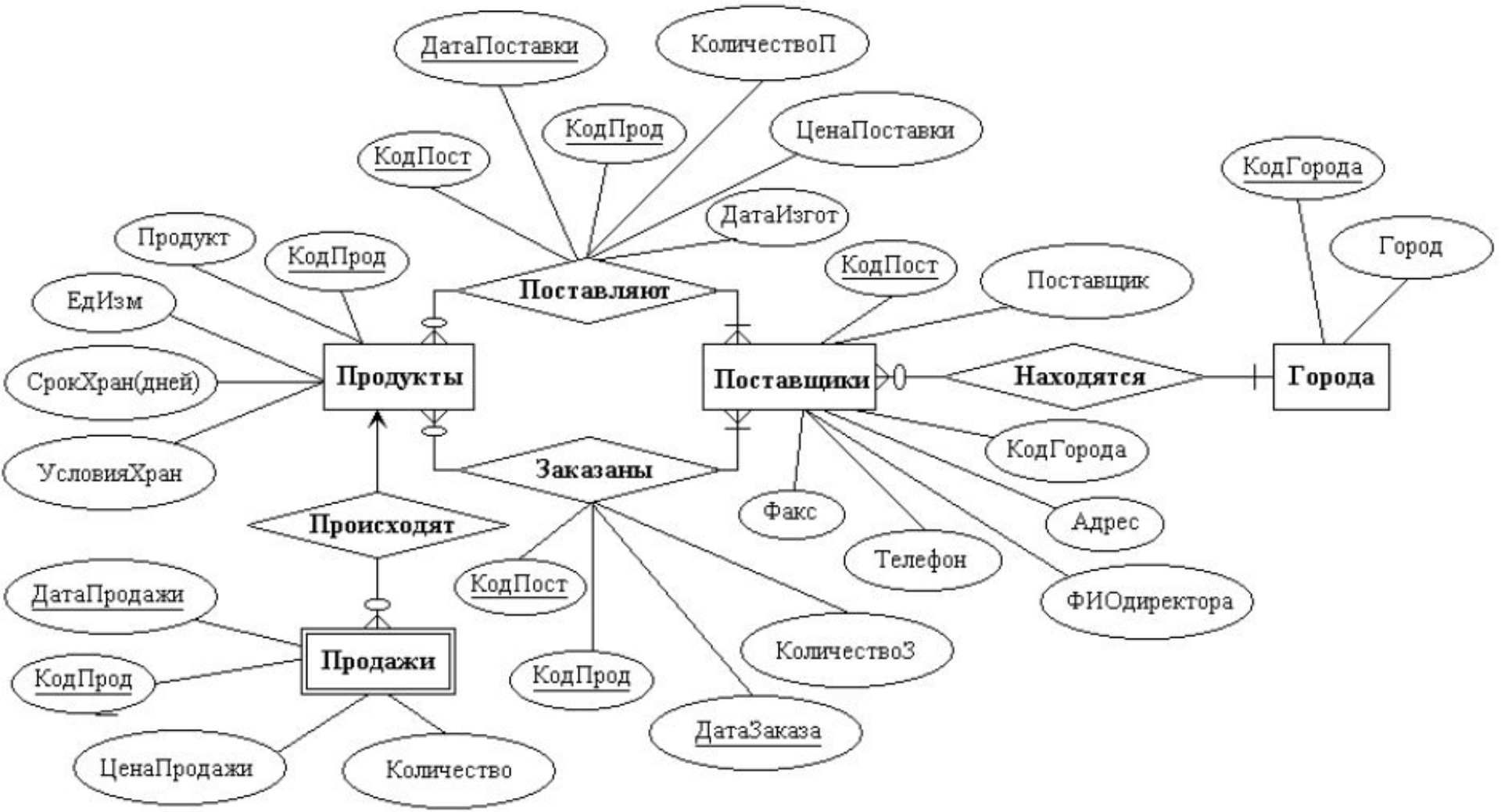


Модель «сущность-связь»: пример

3. Связь N:1 «**Происходят»** между сущностями Продажи и Продукты

4. Связь N:1 «**Находятся**» между сущностями Поставщики и Города

Модель «сущность-связь»: пример



Инфологическое моделирование: CASE

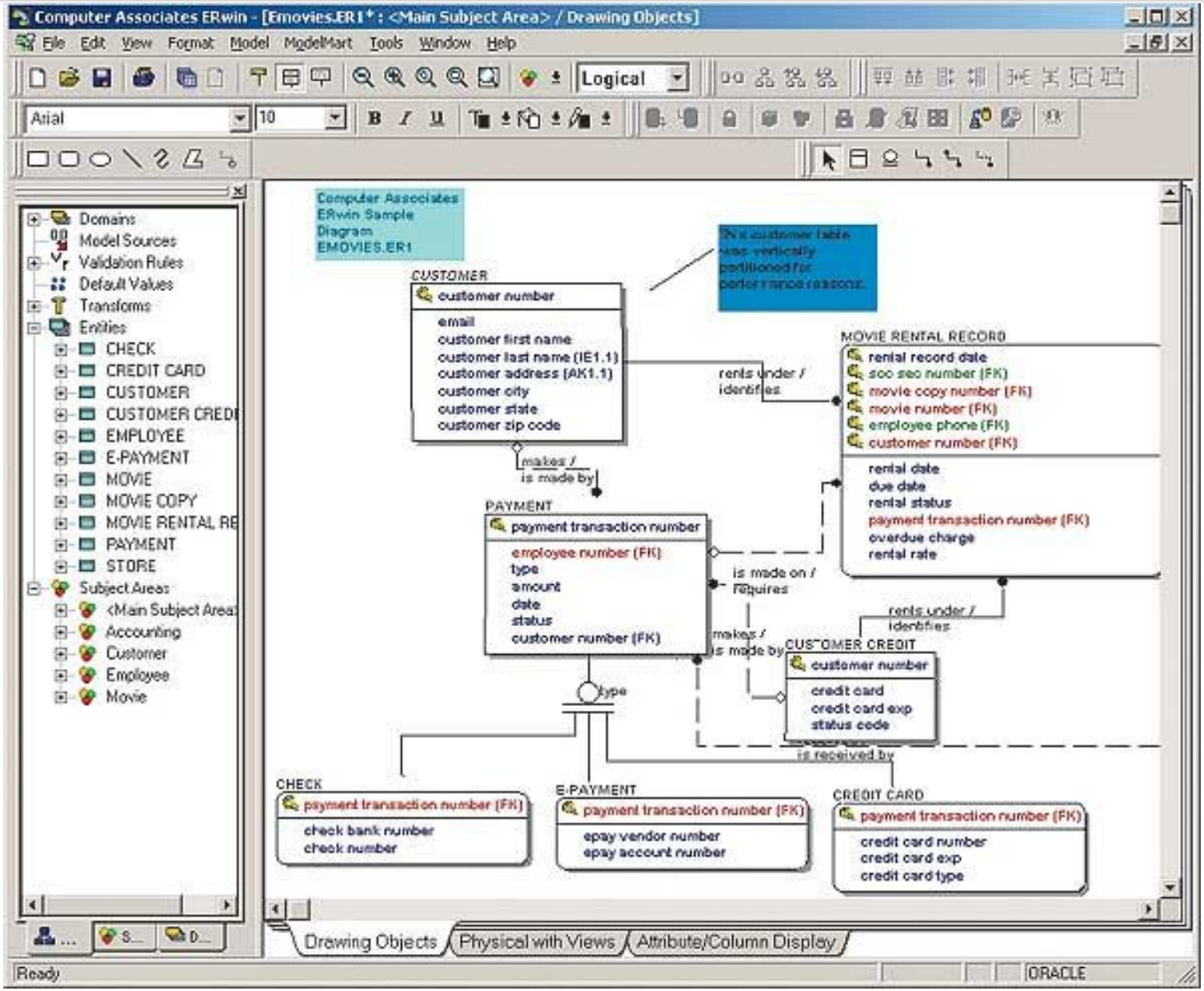
CASE-средства

Computer-Aided System (Software) Engineering

CASE-средства обеспечивают поддержку технологий автоматизированного проектирования, разработки и сопровождения программных систем

Пример: AllFusion ERwin Data Modeler (ERwin)

Инфологическое моделирование: CASE

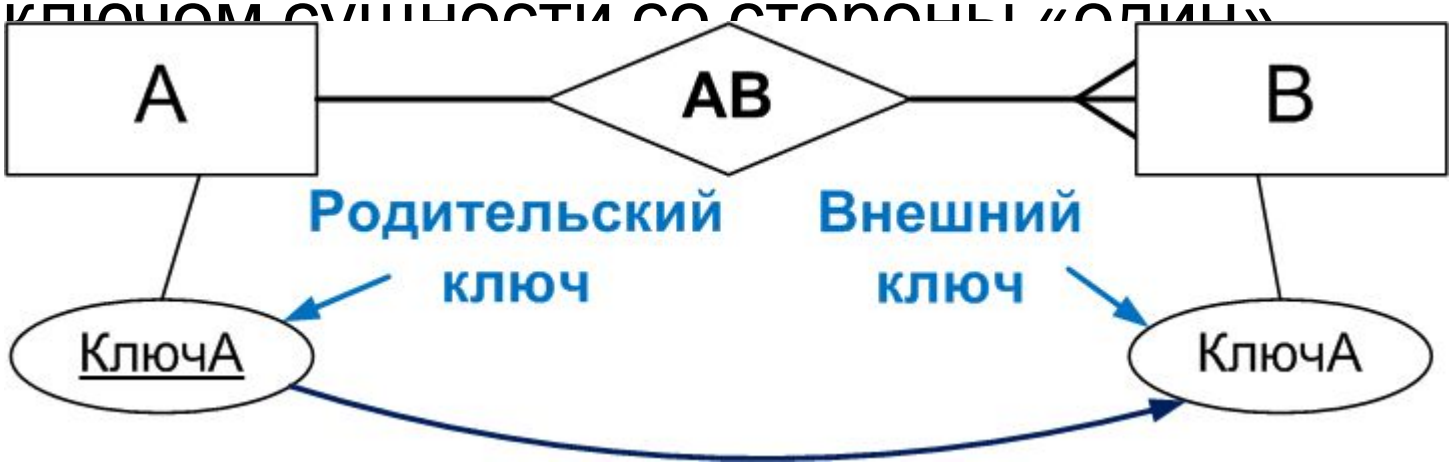


Алгоритм перехода к реляционной модели

1. Каждой сущности модели «сущность-связь» ставится в соответствие отношение реляционной модели
2. Каждый атрибут сущности становится атрибутом соответствующего отношения:
   - задается конкретный допустимый в СУБД тип данных
   - обязательность или необязательность данного атрибута (допустимость или недопустимость NULL-значений)

Алгоритм перехода к реляционной модели

3. Первичный ключ сущности становится первичным ключом соответствующего отношения
4. В каждое отношение, соответствующее сущности со стороны «многие» (связь **1:M**), добавляется набор атрибутов сущности со стороны «один», являющихся первичным ключом сущности со стороны «один»

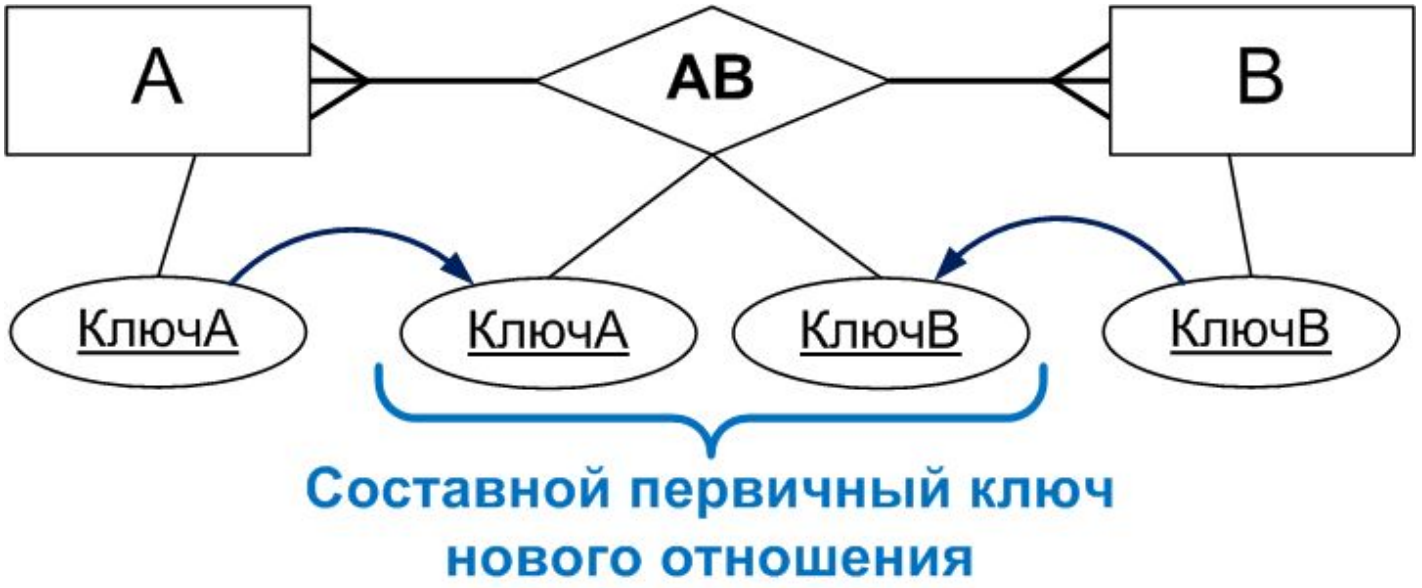


Алгоритм перехода к реляционной модели

5. Для моделирования необязательного и обязательного класса принадлежности:
   - у атрибутов сущности необязательного класса принадлежности, соответствующих внешнему ключу, устанавливается свойство **допустимости неопределенных значений**
   - при обязательном классе принадлежности атрибуты получают свойство **отсутствия неопределенных значений**

Алгоритм перехода к реляционной модели

6. Разрешение связей типа **M:N**:
   - Связи становится в соответствие новое отношение, имеющее атрибуты, которые в сущностях являются первичными ключами, а в новом отношении будут внешними ключами
   - Первичным ключом нового отношения будет **совокупность внешних ключей**



Пример перехода к реляционной модели

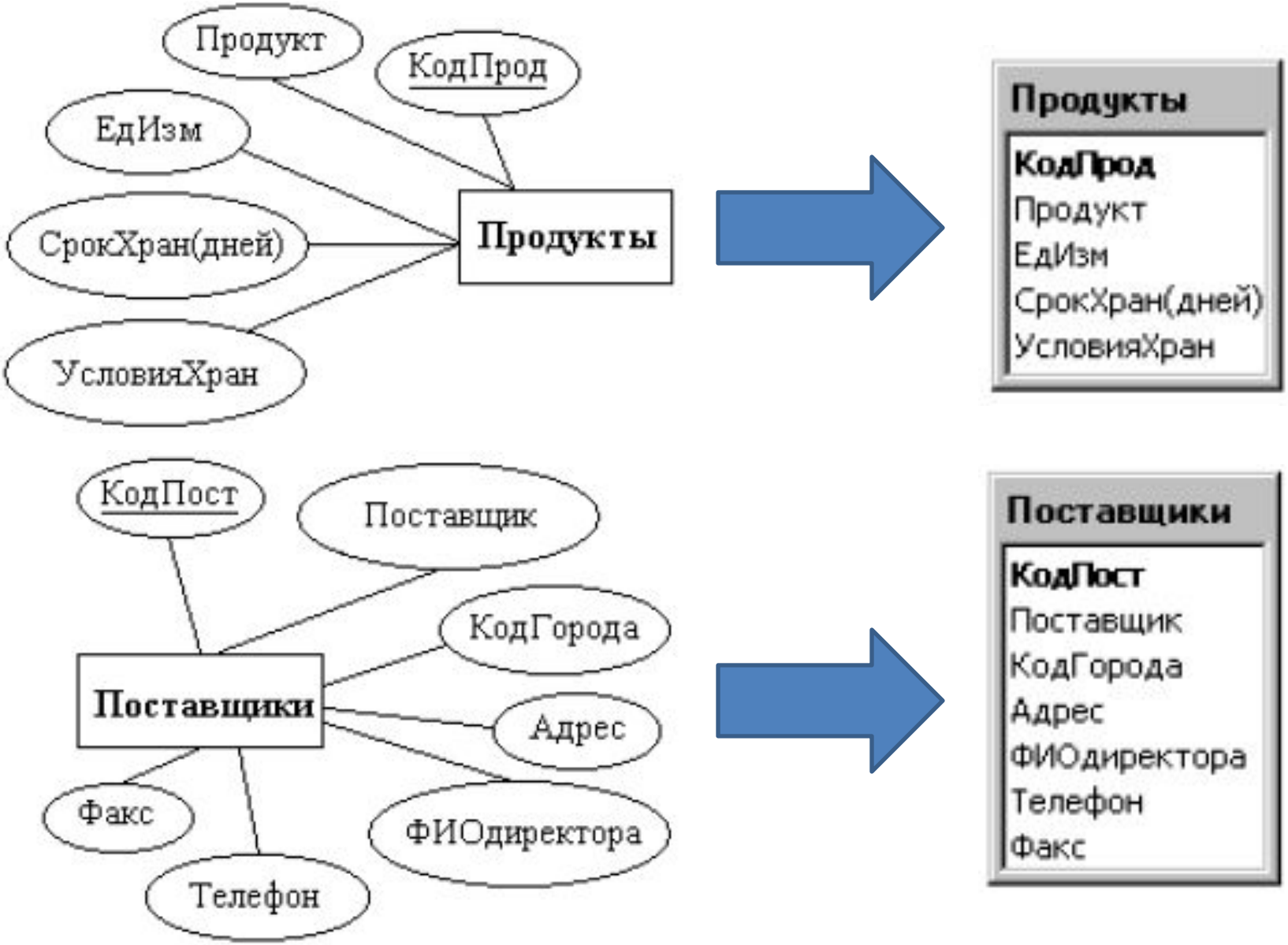
Пример преобразования модели «сущность-связь» к реляционной модели:

В указанной модели мы имеем дело со следующими сущностями:

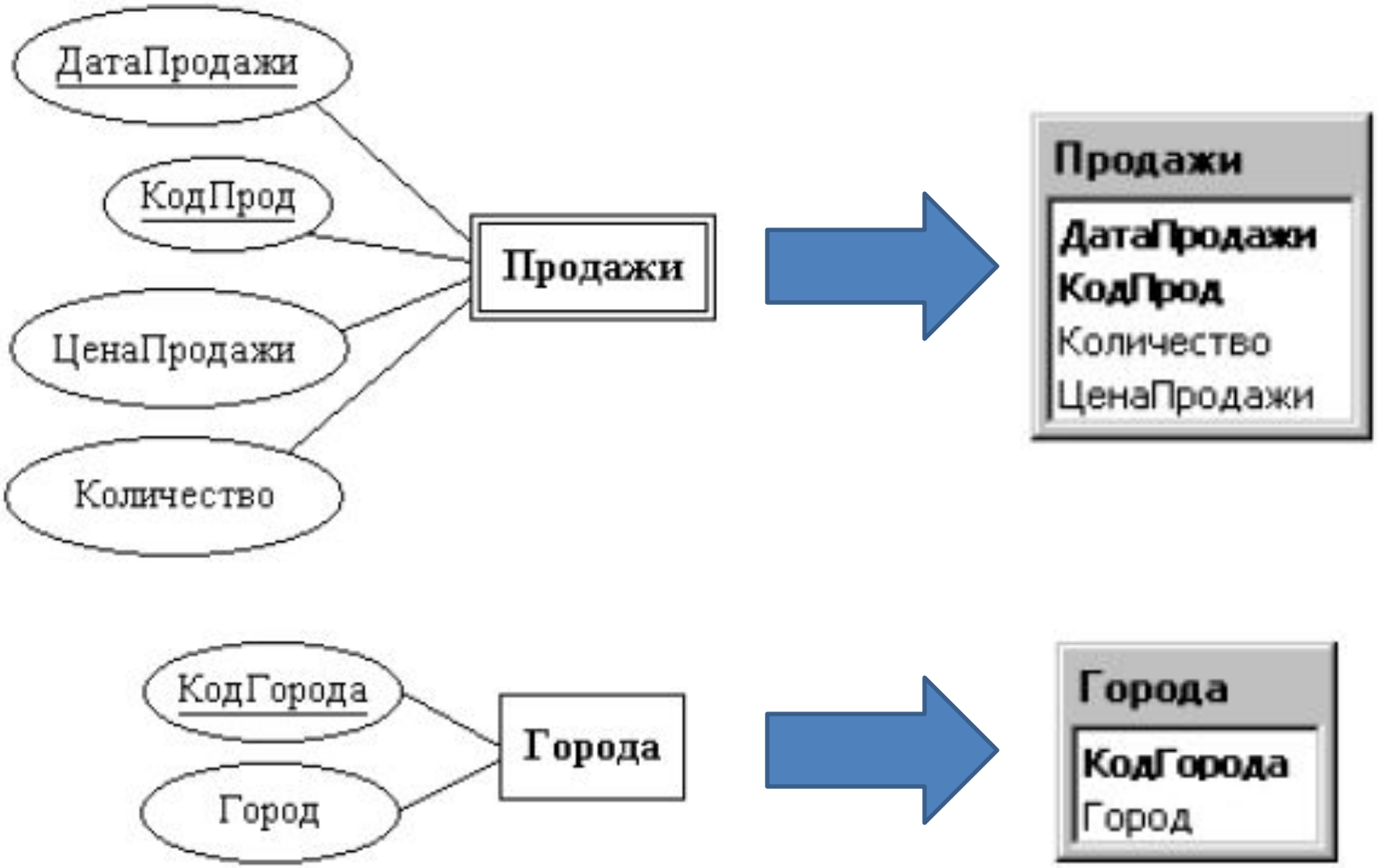
- Продукты
- Поставщики
- Города
- Продажи

Следовательно, и в реляционной модели будут участвовать четыре отношения с такими же именами.

Пример перехода к реляционной модели



Пример перехода к реляционной модели



Пример перехода к реляционной модели

Схема отношения «**Продукты**»

Атрибут	Тип данных (СУБД Access)	Обязательный атрибут	Первичный ключ	Внешний ключ
КодПрод	Целое	Да	+	
Продукт	Текстовый (30)	Да		
ЕдИзм	Текстовый (5)	Нет		
СрокХран	Целое	Нет		
УсловияХран	Текстовый (200)	Нет		

Пример перехода к реляционной модели

Схема отношения «**Поставщики**»

Атрибут	Тип данных (СУБД Access)	Обязательный атрибут	Первичный ключ	Внешний ключ
КодПост	Целое	Да	+	
Поставщик	Текстовый (50)	Да		
КодГорода	Целое	Да		+
Адрес	Текстовый (100)	Нет		
ФИОдиректора	Текстовый (50)	Нет		
Телефон	Текстовый (15)	Нет		
Факс	Текстовый (15)	Нет		

Пример перехода к реляционной модели

Схема отношения «**Продажи**»

Атрибут	Тип данных (СУБД Access)	Обязательный атрибут	Первичный ключ	Внешний ключ
ДатаПродажи	Дата/время	Да	+	
КодПрод	Целое	Да		+
Количество	Одинарное с плавающей точкой	Нет		
ЦенаПродажи	Денежный	Нет		

Пример перехода к реляционной модели

Схема отношения «**Города**»

Атрибут	Тип данных (СУБД Access)	Обязательный атрибут	Первичный ключ	Внешний ключ
КодГорода	Целое	Да	+	
Город	Текстовый (30)	Да		

Пример перехода к реляционной модели

В примере две связи имеют степень M:N. Это связи **Поставляют** и **Заказаны**.

Следовательно, дополнительно появляются еще два отношения:

- **Поставки**
- **Заказы**

Пример перехода к реляционной модели

Схема отношения «**Поставки**»

Атрибут	Тип данных (СУБД Access)	Обязательный атрибут	Первичный ключ	Внешний ключ
ДатаПоставки	Дата/Время	Да		
КодПост	Целое	Да	+	+
КодПрод	Целое	Да		+
КоличествоП	Одинарное с плавающей точкой	Нет		
ЦенаПоставки	Денежный	Нет		
ДатаИзгот	Дата время	Нет		

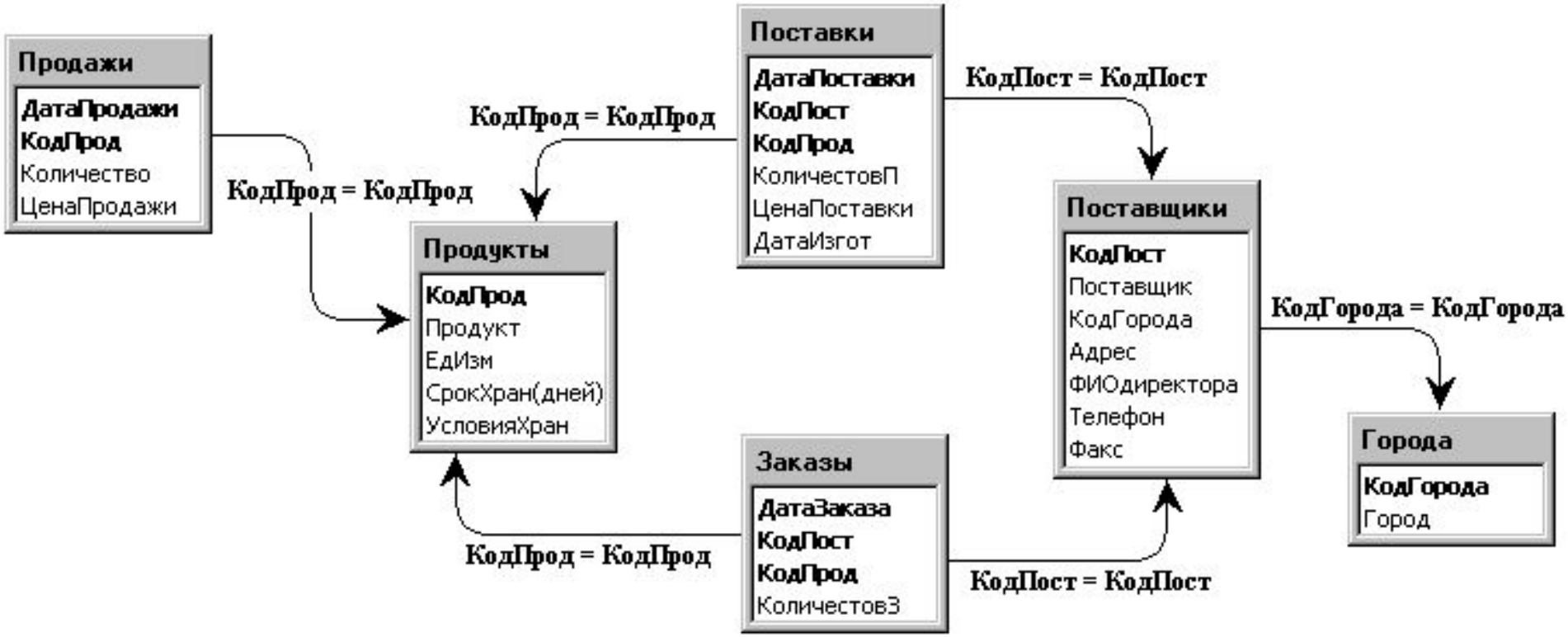
Пример перехода к реляционной модели

Схема отношения «**Заказы**»

Атрибут	Тип данных (СУБД Access)	Обязательный атрибут	Первичный ключ	Внешний ключ
ДатаЗаказа	Дата/Время	Да		
КодПост	Целое	Да	+	+
КодПрод	Целое	Да		+
КоличествоЗ	Одинарное с плавающей точкой	Нет		

Пример перехода к реляционной модели

Окончательный вариант реляционной модели (Схемы БД)



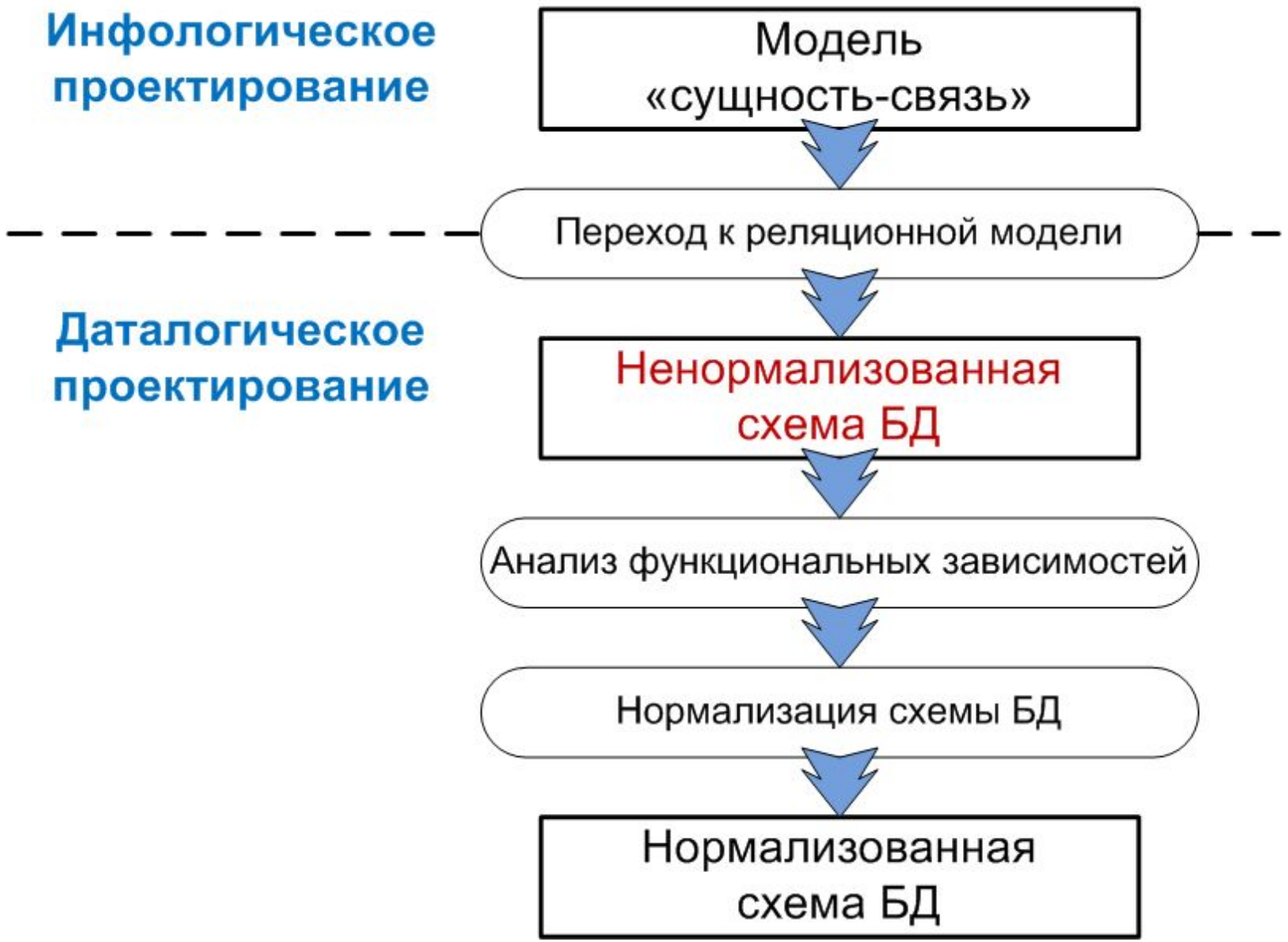
Даталогическое проектирование

Цель даталогического проектирования:

разработка корректной **схемы БД** в терминах выбранной СУБД

Основой анализа корректности схемы являются анализ **функциональных зависимостей** между атрибутами отношений БД

Даталогическое проектирование



Даталогическое проектирование

После нормализации схемы БД и окончательного выбора СУБД выполняется:

- Описание концептуальной схемы БД в терминах выбранной СУБД
- Описание внешних моделей в терминах выбранной СУБД
- Описание правил поддержки целостности базы данных
- Разработка процедур поддержки семантической целостности базы данных

Проектирование схемы БД

Проектирование схемы БД может быть выполнено двумя путями:

- **путем декомпозиции (разбиения):**
  путем последовательной **нормализации** схем отношений
- путем синтеза

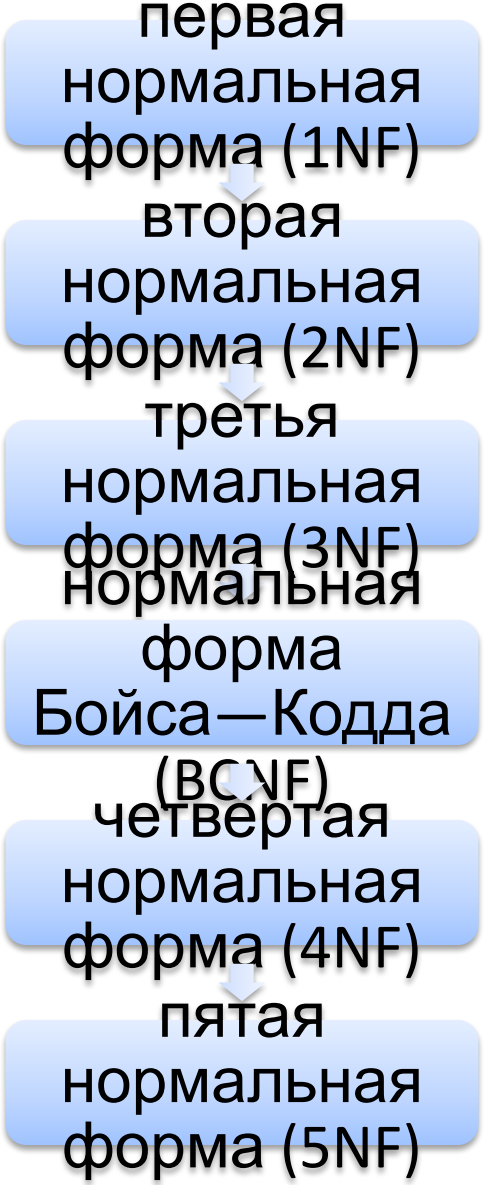
Универсальное отношение — это таблица, в которую включены все интересующие атрибуты, то есть та таблица, которая требует нормализации

Нормализация базы данных

Нормализация — это процесс преобразования отношения в состояние, обеспечивающее лучшие условия выборки, добавления, изменения и удаления данных.

Главная цель нормализации: устранение избыточности и дублирования информации в базе данных

Нормальные формы



Свойства нормальных форм

Каждой нормальной форме соответствует определенный набор ограничений

Основные свойства нормальных форм:

- каждая следующая нормальная форма улучшает свойства предыдущей
- при переходе к следующей нормальной форме свойства предыдущих нормальных форм **сохраняются**

Первая нормальная форма

Отношение находится в первой нормальной форме, если значения всех его атрибутов **атомарны**.

R1 – Ненормализованное отношение

КодПоставщика	КодПродукта	Продукт
P_1	1	Сахар
	2	Соль
	13	Мука
P_2	26	Рис
	58	Гречка
	130	Крупа манная
	162	Пшено
P_3	474	Молоко
	891	Кефир

R2 – Нормализованное отношение

КодПоставщика	КодПродукта	Продукт
P_1	1	Сахар
P_1	2	Соль
P_1	13	Мука
P_2	26	Рис
P_2	58	Гречка
P_2	130	Крупа манная
P_2	162	Пшено
P_3	474	Молоко
P_3	891	Кефир

Первая нормальная форма: пример

Преподаватель	День недели	Номер пары	Название дисциплины	Тип занятий	Группа
Петров В. И.	Понедельник	1	Теор. выч. проц.	Лекция	4906
	Вторник	1	Комп. графика	Лаб. раб.	4907
	Вторник	2	Комп. графика	Лаб. раб.	4906
Киров В. А.	Понедельник	2	Теория информ.	Лекция	4906
	Вторник	3	Пр-е на C++	Лаб. раб.	4907
	Вторник	4	Пр-е на C++	Лаб. раб.	4906
Серов А. А.	Понедельник	3	Защита инф.	Лекция	4944
	Среда	3	Базы данных	Лаб. раб.	4942
	Четверг	4	Базы данных	Лаб. раб.	4922

Первая нормальная форма: пример

Преподаватель	День недели	Номер пары	Название дисциплины	Тип занятий	Группа
Петров В. И.	Понедельник	1	Теор. выч. проц.	Лекция	4906
Петров В. И.	Вторник	1	Комп. графика	Лаб. раб.	4907
Петров В. И.	Вторник	2	Комп. графика	Лаб. раб.	4906
Киров В. А.	Понедельник	2	Теория информ.	Лекция	4906
Киров В. А.	Вторник	3	Пр-е на C++	Лаб. раб.	4907
Киров В. А.	Вторник	4	Пр-е на C++	Лаб. раб.	4906
Серов А. А.	Понедельник	3	Защита инф.	Лекция	4944
Серов А. А.	Среда	3	Базы данных	Лаб. раб.	4942
Серов А. А.	Четверг	4	Базы данных	Лаб. раб.	4922

Недостатки первой нормальной формы

- **избыточность** — многократное повторение информации в столбцах данных
- аномалии модификации (обновления) данных
- аномалии добавления данных
- аномалии удаления данных

Пример:

Экзамены (ФИО, Номер зач.кн., Группа, Дисциплина, Дата экзамена, Оценка)

Избыточность данных: пример

ФИО	Номер ЗачКн	Группа	Название дисциплины	Дата	Оценка
Пупкин В. И.	323556	ММ-117	Управление данными	17/01/10	2
Пупкин В. И.	323556	ММ-117	Управление данными	25/01/10	3
Петров В. А.	156900	ММ-117	Управление данными	25/01/10	5
Сидоров А. А.	278001	ММ-119	Мат. анализ	21/01/10	5
Киров В. У.	777890	ММ-119	Мат. анализ	21/01/10	4
Хренова Г. П.	123456	ММ-334	Инф. менеджмент	21/01/10	3
Бобриков С. С.	998769	ММ-334	Инф. менеджмент	21/01/10	5
Хренова Г. П.	123456	ММ-334	Базы данных	24/01/10	2
Бобриков С. С.	998769	ММ-334	Базы данных	24/01/10	4

Функциональная зависимость

Атрибут Y некоторого отношения функционально зависит от X (атрибуты могут быть составными), если в любой момент времени каждому значению X соответствует одно значение Y.

Функциональная зависимость обозначается: $X \rightarrow Y$

Пример: Номер зач.кн. $\rightarrow$ ФИО

Полная функциональная зависимость

Неключевой 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:

Номер зач.кн., Дисциплина, Дата → Оценка

Вторая нормальная форма

Отношение (таблица) находится во 2НФ, если оно находится в 1НФ, и **каждый неключевой** атрибут функционально полно зависит от всего ключа.

Приводить ко 2 НФ необходимо только отношения с составным ключом

Вторая нормальная форма

Если какой-либо атрибут зависит от **части составного первичного ключа**, то необходимо:

- создать новое отношение, атрибутами которого будут:
- часть составного ключа (первичный ключ нового отношения)
- атрибут, зависящий от нового ключа
- из исходного отношения исключить атрибут, включенный в новое отношение

Вторая нормальная форма

Полная ФЗ: $\underline{k1, k2} \rightarrow a1$

Неполная ФЗ: $\underline{k1} \rightarrow a2$

1 НФ

R (<u>k1, k2</u>, a1, a2)

Проекция

2 НФ

R1(<u>k1</u>, a2) — 1 — ∞ — R2(<u>k1, k2</u>, a1)

Вторая нормальная форма: пример

Определение неполных ФЗ

Составление таблицы-опросника:

КЛ – ключевые атрибуты, НК – неключевые

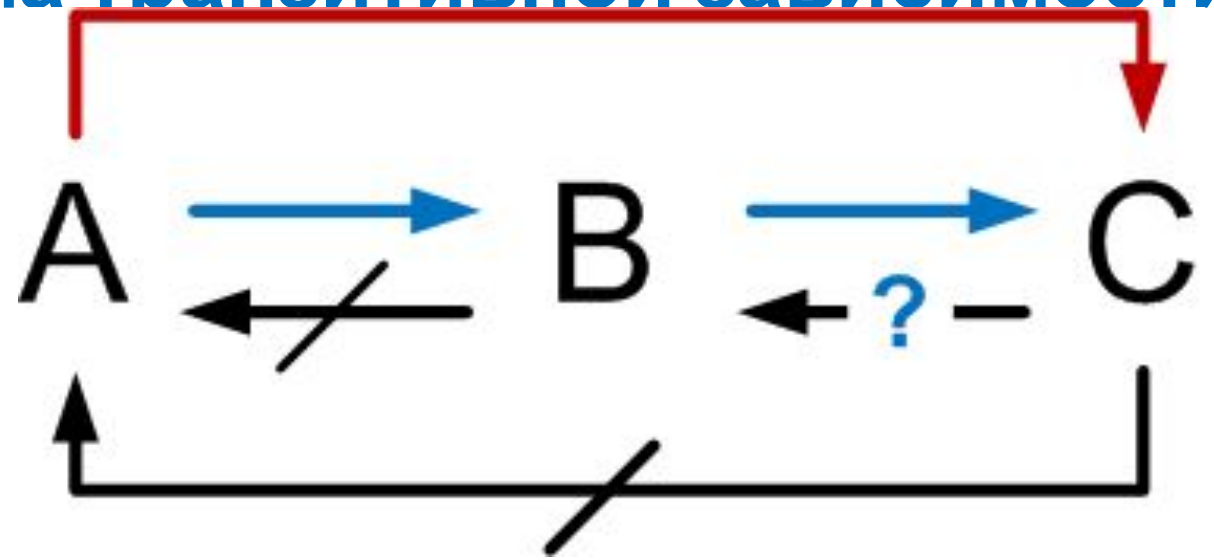
НК \ КЛ	КЛ1	КЛ2	…	КЛn
НК1	+	+	+	+
НК2		+		+
…	+	+	+	+
НКn	+			

Транзитивная зависимость

Транзитивная функциональная зависимость:

Пусть A ,B, C – три атрибута некоторого отношения R.

Схема транзитивной зависимости:



Третья нормальная форма

Отношение находится в 3НФ, если оно находится во 2НФ и **каждый неключевой атрибут** нетранзитивно зависит от первичного ключа.

Наличие транзитивной зависимости влечет за собой появление **аномалий обновления**.

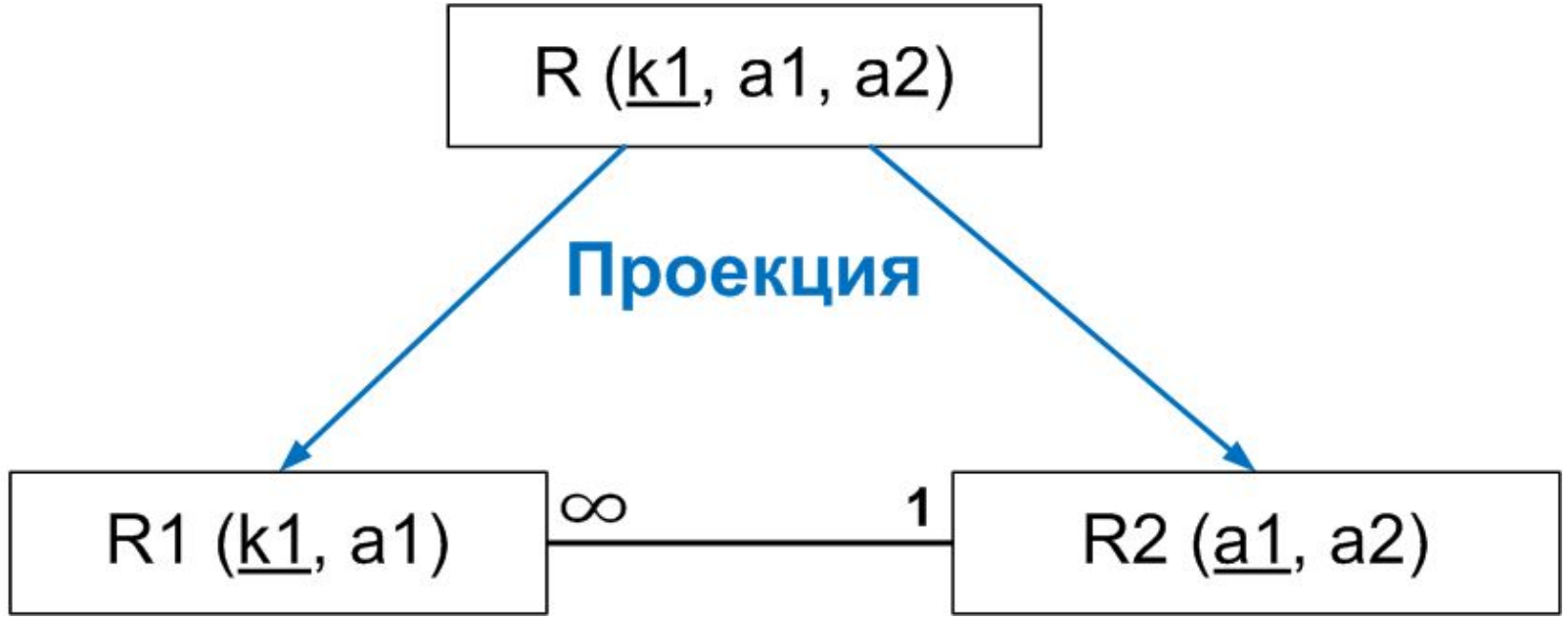
Третья нормальная форма

Транзитивная зависимость:

$k1 \rightarrow a1 \quad (a1 \not\rightarrow k1)$

$a1 \rightarrow a2 \quad (a2 \not\rightarrow a1)$

2 НФ



3 НФ

Третья нормальная форма: пример

Транзитивная зависимость:

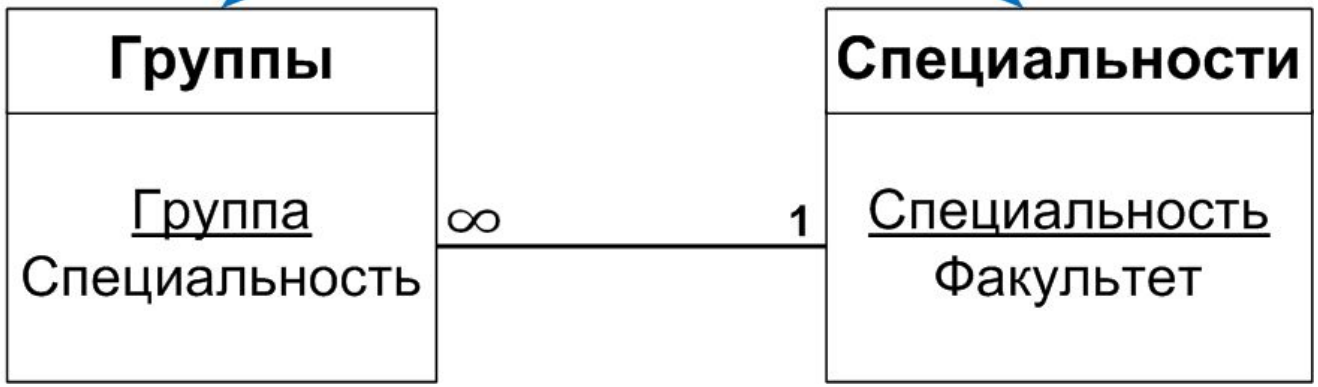
Группа → Специальность

Специальность → Факультет

2 НФ

Проекция

3 НФ



Определение транзитивных ФЗ

Составление таблицы-опросника:

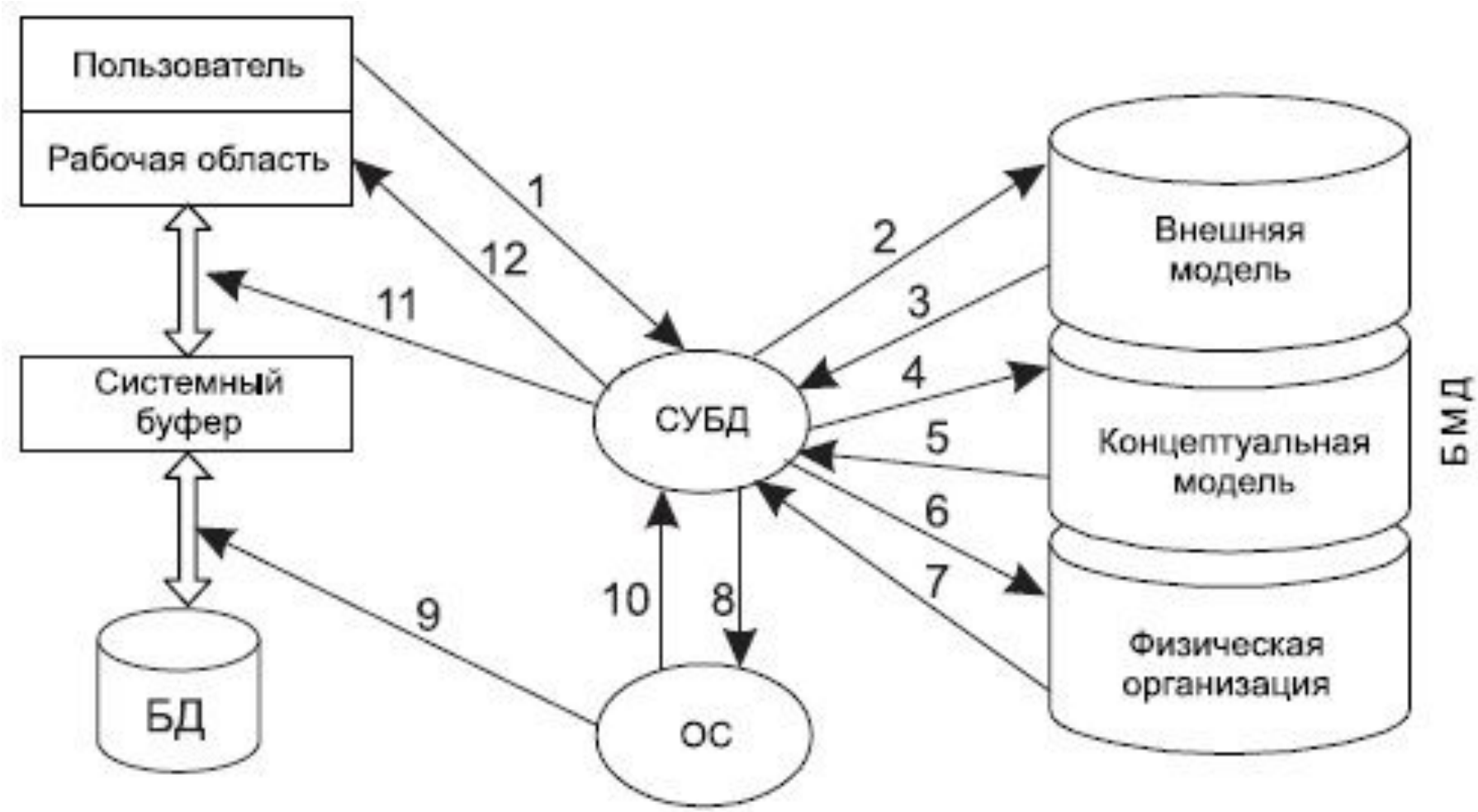
НК – неключевые атрибуты

НК / НК	НК1	НК2	…	НКn
НК1		+		
НК2	+			
…				
НКn		+		

Тема 6. Приложения и системы управления базами данных

1. Прохождение запроса к БД
2. Основные функции СУБД
3. Режимы работы с БД
4. Распределенная обработка данных
5. Уровни приложения. Архитектуры приложений
6. Транзакции
7. Индексы

Схема прохождения запроса к БД



Основные функции СУБД

- Определение схемы данных
- Манипулирование данными
- Оптимизация и выполнение запросов
- Обеспечение независимости данных
- Защита и поддержка целостности данных
- Поддержка многопользовательской работы
- Управление ресурсами среды хранения
- Поддержка функций

Режимы работы с БД

Архитектуры приложений

Централизованная архитектура (монолитное приложение)

Двухзвенная архитектура («файл-сервер» или «клиент-сервер»)

Трехзвенная архитектура

Централизованная архитектура

Автономная работа
(все размещено на одном компьютере)

Главный недостаток: невозможна параллельная работа нескольких пользователей

Централизованная архитектура

Примеры СУБД с централизованной архитектурой (70-80-е года):

- Первые версии Oracle
- Первые версии DB2
- Первые версии Ingres

Распределенная обработка данных

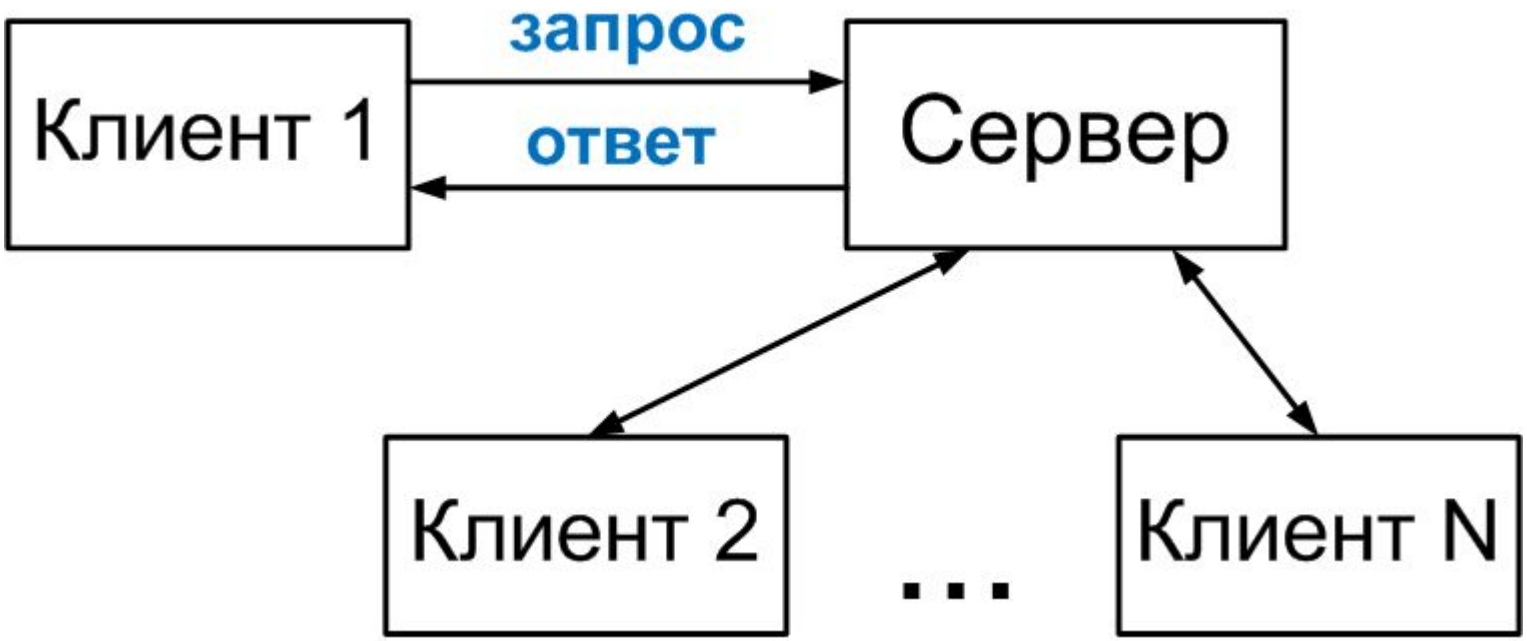
Система распределенной обработки данных — система, обеспечивающая параллельный доступ пользователей компьютерной сети к **централизованной БД**

Распределенная база данных — совокупность логически взаимосвязанных баз данных, распределенных в компьютерной сети

Двухзвенная архитектура

Сервер — логический процесс, обеспечивающий обслуживание других процессов

Клиент — логический процесс, посылающий серверу запрос на обслуживание



Уровни приложения

Presentation Logic

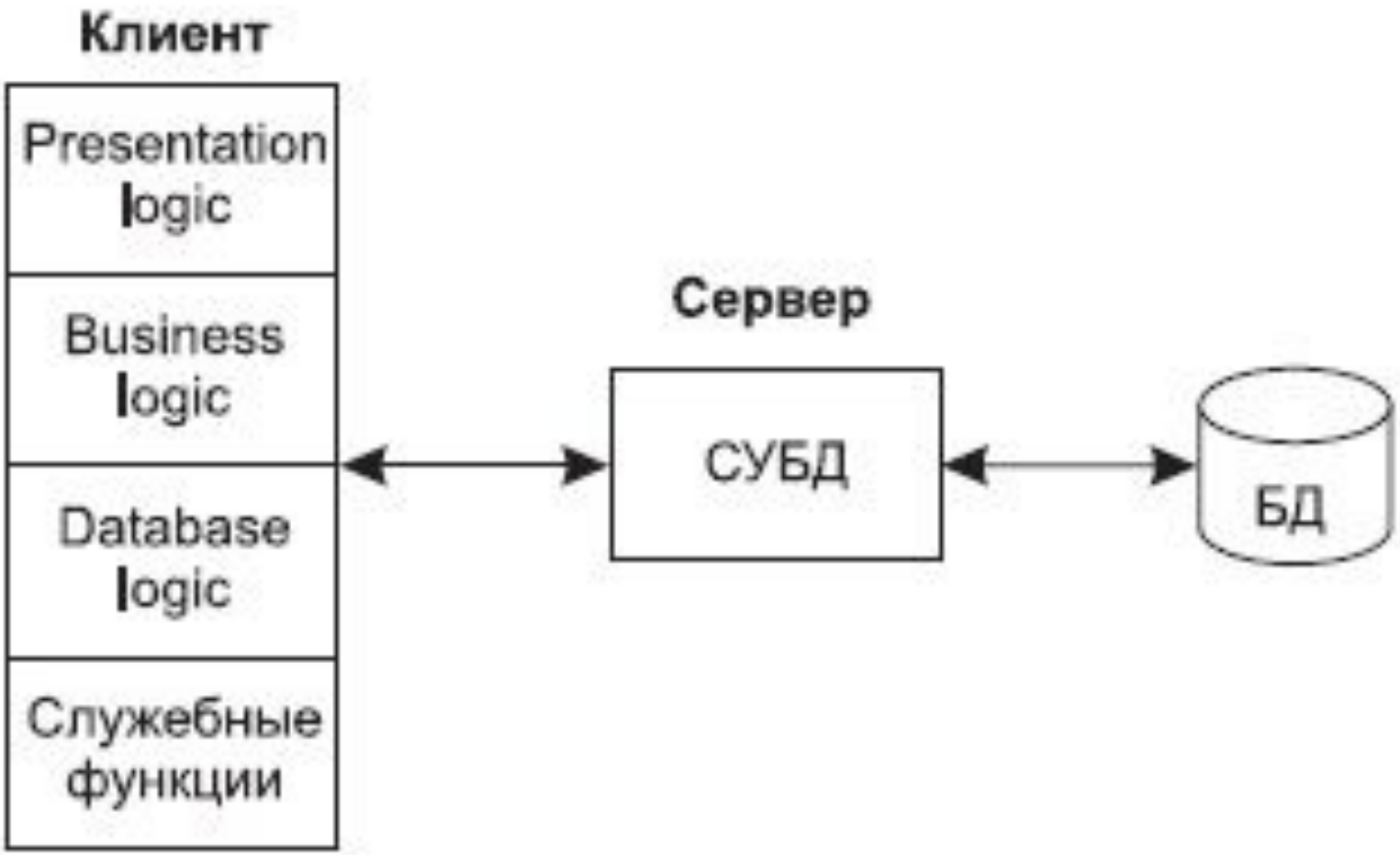
Business Logic

Database Logic

Database Manager System Processing

Служебные функции

Уровни приложения



Модель «File Server» (FS)

Модель файлового сервера

Модель «File Server»

Основные свойства:

- Выделяется файл-сервер для реализации услуг по обработке файлов
- Сервер передает СУБД, размещенной на компьютере-клиенте, требуемый блок данных
- Протокол обмена — набор низкоуровневых вызовов файловых команд
- Вся обработка осуществляется на компьютере-клиенте

Модель «File Server»

Преимущества:

- разделение монолитного приложения на два взаимодействующих процесса (клиент и сервер)
- простота архитектуры, использование штатных средств ОС

Недостатки:

- высокий сетевой трафик
- загруженность клиентского компьютера
- низкая производительность при многопользовательской работе
- узкий спектр операций манипулирования с данными
- защита данных и администрирование только на уровне файловой системы
- недостаточно развитый аппарат транзакций

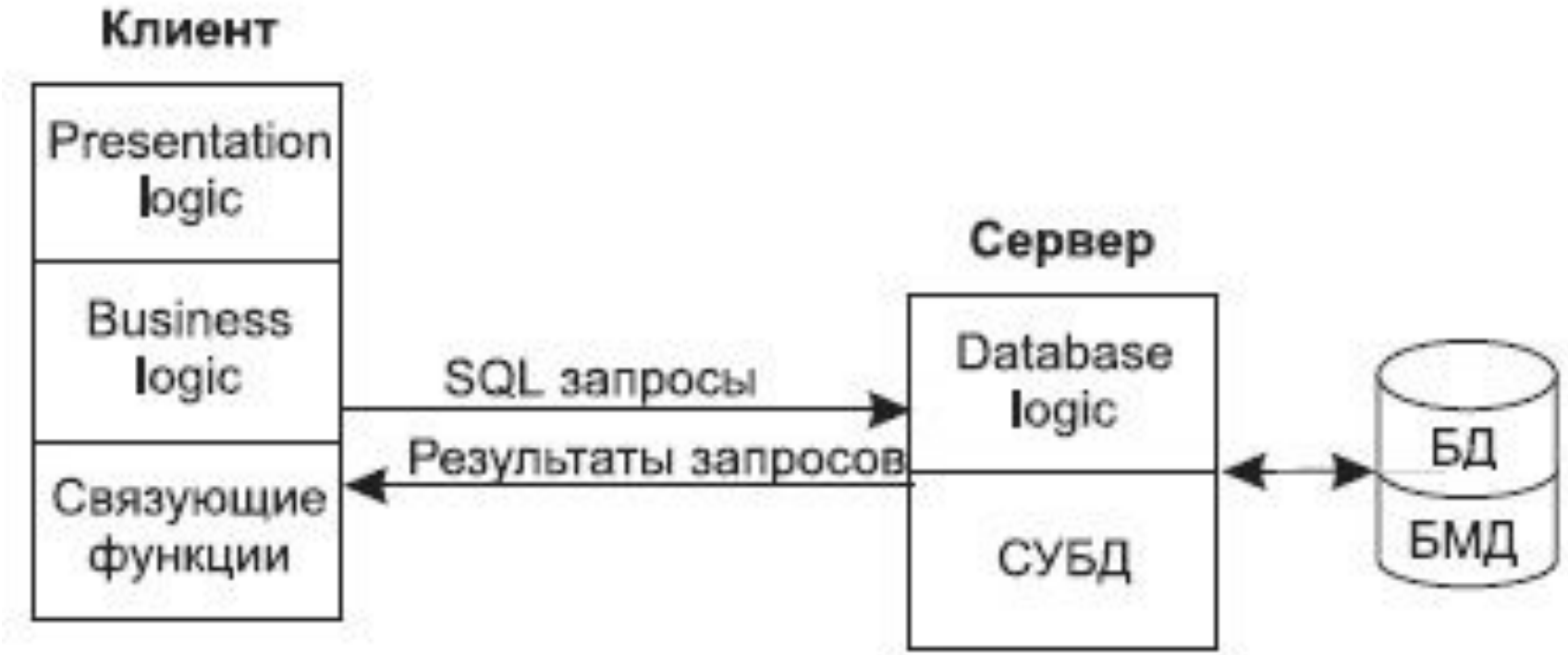
Модель «File Server»

Примеры файл-серверных СУБД:

- dBase
- Microsoft Access
- FoxPro и Visual FoxPro
- Paradox
- Clipper

Модель «Remote Data Access» (RDA)

Модель удаленного доступа к данным



Сервер БД — логический процесс, отвечающий за обработку запросов к БД

Модель «Remote Data Access»

Основные свойства:

- Коды компонента представления и прикладного компонента совмещены и выполняются на компьютере-клиенте
- Доступ к информационным ресурсам обеспечивается операторами языка SQL
- Инициатор манипуляций с данными — программы на компьютере-клиенте
- Ядро СУБД выполняет пассивную роль (выполняет SQL-команды от клиента)

Модель «Remote Data Access»

Преимущества:

- процессор сервера загружается операциями обработки данных
- уменьшается загрузка сети (передача только SQL-запросов)

Недостатки:

- сервер играет пассивную роль
- затрудненность администрирования и контроля приложения из-за совмещения на клиенте различных функций

Модель «Database Server» (DBS)

Модель сервера баз данных

Модель «Database Server»

Основные свойства:

- Использования механизма **хранимых процедур** и **триггеров**, как средство программирования SQL-сервера
- Компонент представления выполняется на компьютере-клиенте
- Прикладной компонент и ядро СУБД — на компьютере-сервере базы данных

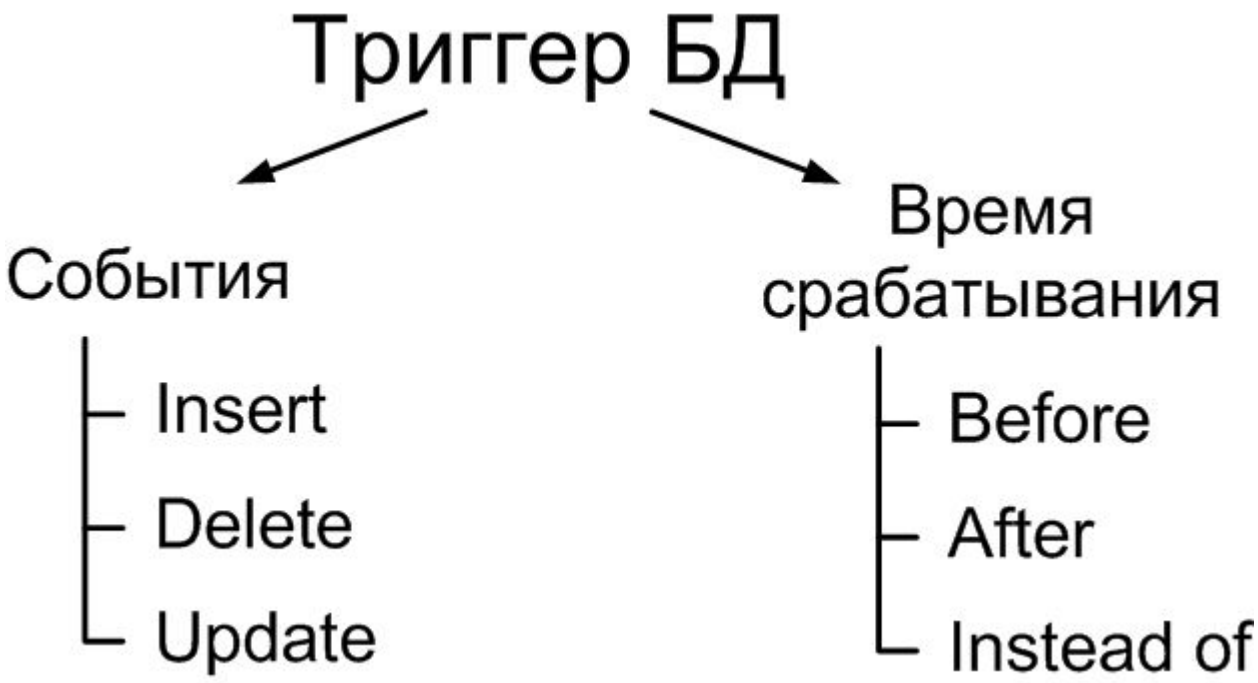
Хранимые процедуры

Хранимая процедура — фрагмент программного кода, который хранится на сервере БД и выполняется по запросу клиента

- представляет собой набор SQL-инструкций
- компилируется один раз и хранится на сервере
- в коде могут использоваться инструкции управления процессом исполнения (ветвления, циклы)

Триггеры

Триггер базы данных — это хранимая процедура особого типа, которая вызывается при наступлении определенного события (действия)



Модель «Database Server»

Преимущества:

- низкие требования к клиенту («тонкий» клиент)
- возможность централизованного администрирования
- централизованное управление и настройка бизнес-логики

Недостатки:

- возможна большая загрузка сервера
- недостаточно возможностей для отладки и типизирования хранимых процедур
- ограниченность средств для написания хранимых процедур

Примеры RDA- и DBS-СУБД

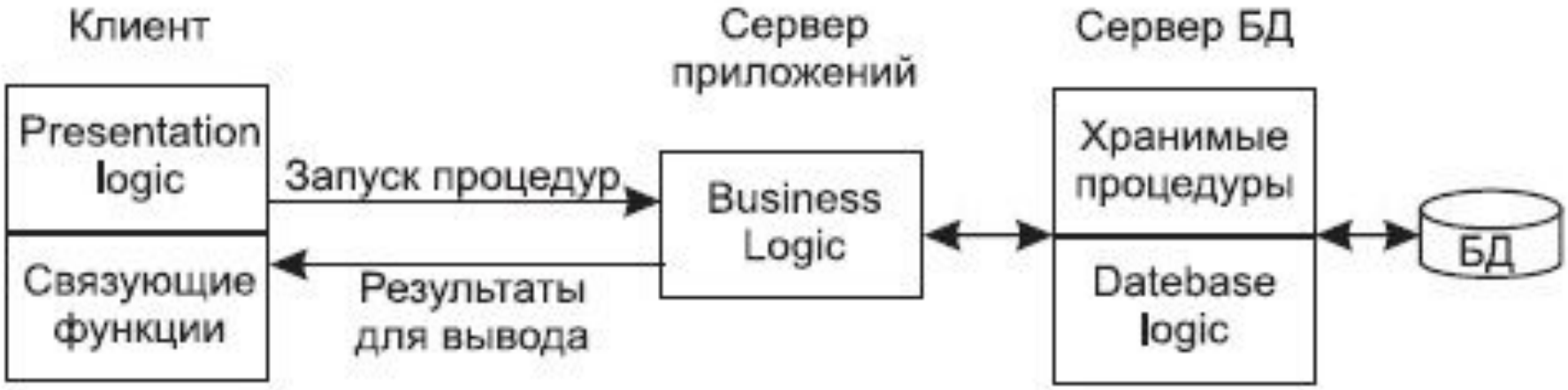
Примеры СУБД, реализующих синтез RDA- и DBS-моделей:

- Oracle
- MS SQL Server
- DB2
- Sybase
- Ingres
- Informix
- PostgreSQL
- MySQL

Трехзвенная архитектура

Модель «Application Server» (AS)

(модель сервера приложений)



Трехзвенная архитектура

Основные свойства:

- Клиент отвечает только за интерфейс пользователя
- Прикладные функции (бизнес-логика) выделены как **важнейший изолированный элемент** и выполняются на сервере приложений (AS)
- Все операции над БД выполняются соответствующим сервером БД

Трехзвенная архитектура

Преимущества:

- «Тонкий» клиент (чаще всего web-клиент)
- Централизованное управление приложениями (настройка, обновление)

Недостатки:

- сложное программное обеспечение

Модель «Application Server»

Примеры серверов приложений:

- **Java application servers**
  - Apache Geronimo
  - Glassfish Application Server (Sun)
  - WebSphere Application Server (IBM)
  - JBoss (Red Hat)
  - Jetty (Eclipse Foundation)
  - WebLogic Server (Oracle)
- **Microsoft .NET Framework**

Понятие транзакции

Транзакция — законченная совокупность действий, которая может быть:

— либо **полностью выполнена**
COMMIT (фиксация изменений)

— либо **полностью отвергнута**
ROLLBACK (откат изменений)

- Транзакция — это **логическая единица** работы с базой данных
- Транзакция обычно включает в себя несколько операций над базой данных

Пример транзакции

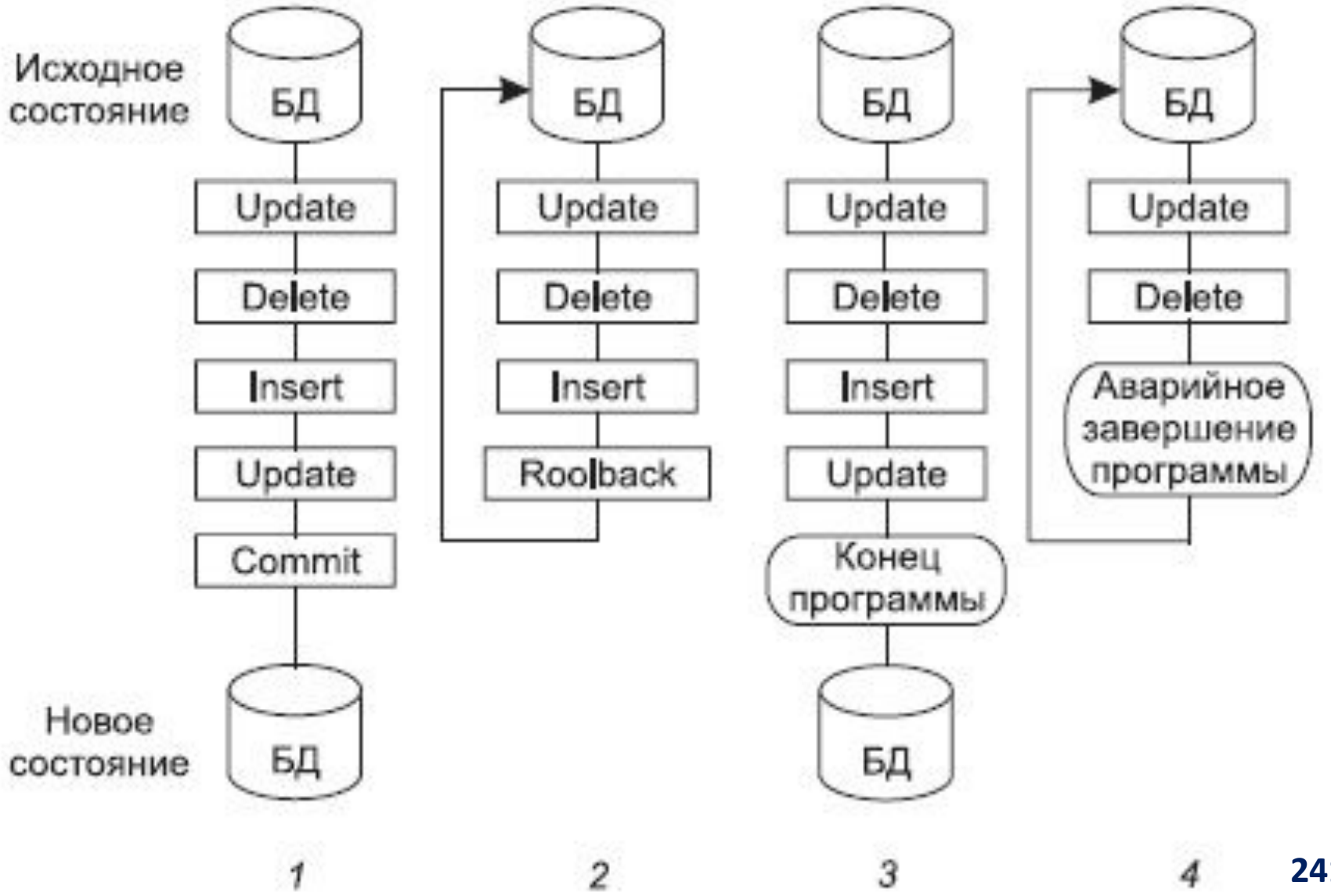
```
/*Перевод денег со счета A на счет B*/
BEGIN TRANSACTION;
UPDATE account A;  /*Списание денег со счета A */
UPDATE account B;  /*Зачисление денег на счет B */
IF <все выполнено успешно>
   THEN COMMIT;  /* Нормальное завершение */
   ELSE ROLLBACK;  /* Аварийное завершение */
END IF;
```

Свойства транзакций

Требования ACID:

- Атомарность (**A**tomicity)
- Согласованность (**C**onsistency)
- Изолированность (**I**solation)
- Долговечность (**D**urability)

Модель транзакций ANSI/ISO



Журнализация транзакций

Журнал транзакций — системная структура, хранящая информацию об изменениях базы данных

Цель журнализации: обеспечение возможности восстановления согласованного состояния базы данных после любого сбоя

Восстанавливается последнее по времени согласованное состояние базы данных

Восстановление базы данных

- Индивидуальный откат транзакции
- Восстановление после внезапной потери содержимого оперативной памяти
  (**мягкий сбой**)
- Восстановление после поломки основного внешнего носителя базы данных
  (**жесткий сбой**)

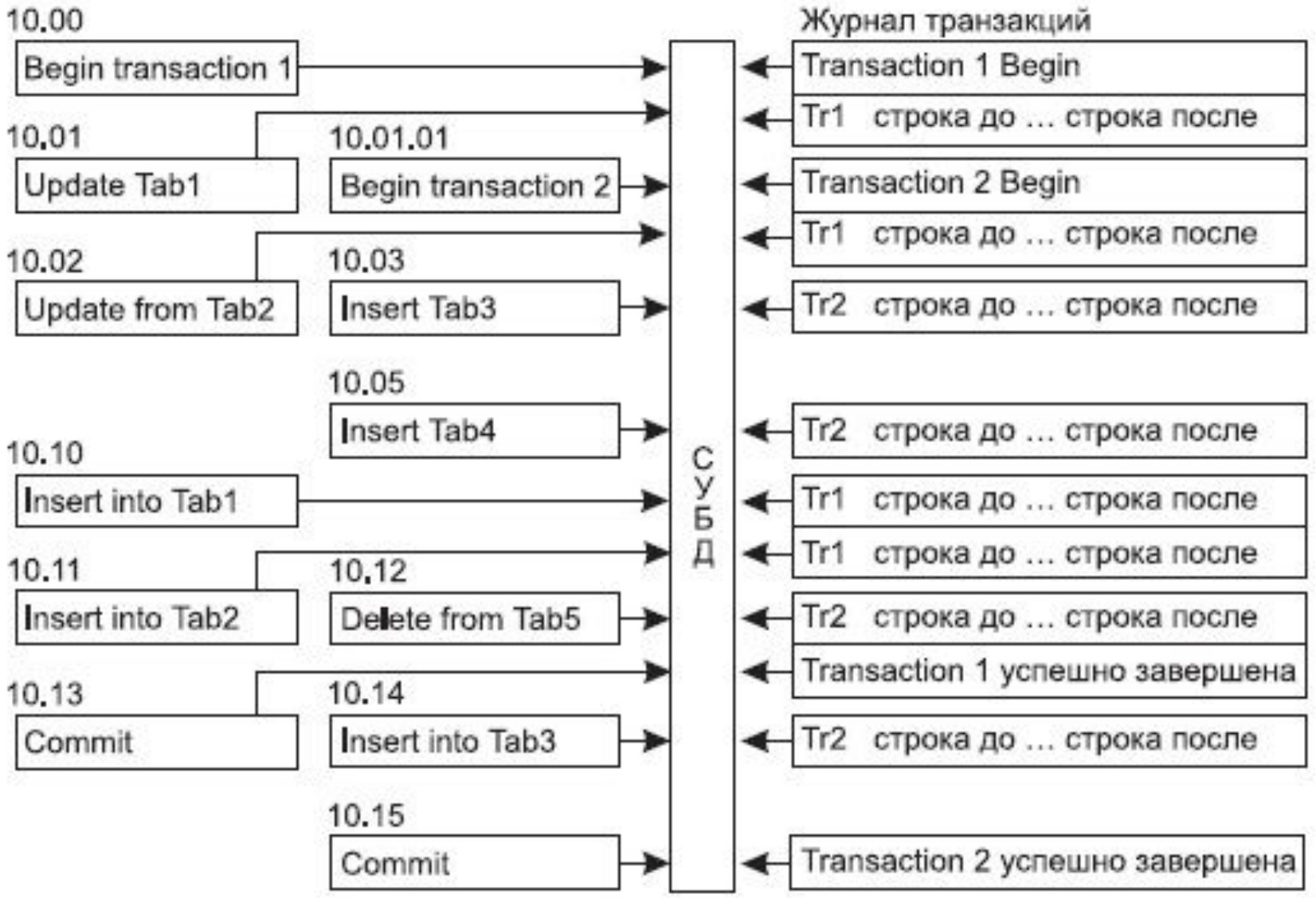
Журналы транзакций

Варианты ведения журналов транзакций:

- Для каждой транзакции поддерживается отдельный **локальный журнал** изменений БД и поддерживается **общий журнал** изменений базы данных
- Поддержание только **общего журнала** изменений базы данных
  (чаще всего используется этот вариант)

Общая структура журнала — последовательный файл, в котором фиксируется каждое изменение БД, которое происходит в ходе выполнения транзакции

Пример журнала транзакций



Тема 7. Знания, интеллектуальные банки и базы знаний (на самостоятельное изучение)

1. Направления развития искусственного интеллекта, представление знаний
2. Данные, информация и знания
3. Особенности знаний, классификация знаний
4. Иерархическая структура обработки информации
5. Банки и базы знаний
6. Экспертные системы

Направления развития искусственного интеллекта

- **Представление знаний и разработка систем, основанных на знаниях**
- Разработка естественно-языковых интерфейсов и машинный перевод
- Распознавание образов
- Новые архитектуры компьютеров
- Интеллектуальные роботы
- Специальное программное обеспечение
- Обучение и самообучение
- Игры и творчество

Данные

Данные — это отдельные факты, характеризующие объекты, процессы и явления в предметной области, а также их свойства

При обработке на ЭВМ данные трансформируются, условно проходя следующие этапы:

1) данные как результат измерений и наблюдений
2) данные на материальных носителях информации (таблицы, протоколы, справочники)
3) модели (структуры) данных в виде диаграмм, графиков, функций
4) данные в компьютере на языке описания

Знания

Знания — это выявленные закономерности предметной области (принципы, связи, законы), позволяющие решать задачи в конкретной предметной области.

При обработке знания трансформируются аналогично данным:

1) знания в памяти человека как результат мышления
2) материальные носители знаний (учебники, методические пособия)
3) поле знаний — условное описание основных объектов предметной области, их атрибутов и закономерностей, их связывающих
4) знания, описанные на языках представления знаний (продукционные языки, семантические сети, фреймы)

Данные, информация, знания

информация = данные + смысл

знание = информация + цель

знания = данные + смысл + цель

Знания — это **система** информации, обеспечивающая увеличение вероятности достижения какой-либо **цели**

Знания — это технологии обработки информации

Особенности знаний

Для знаний (в отличие от данных) характерно:

1. Интерпретируемость
2. Наличие классифицируемых отношений
3. Наличие ситуативных связей

Классификация знаний

Знания бывают:

- **поверхностными** — знания о видимых взаимосвязях между отдельными событиями и фактами в предметной области
- **глубинными** — абстракции, аналогии, схемы, отображающие структуру и природу процессов предметной области

Знания также можно разделить на:

- **процедурные** — знания, «растворенные» в алгоритмах (на языках программирования)
- **декларативные** — знания, записанные на языках представления знаний (близкие к естественным языкам)

Иерархическая структура обработки информации

УПРАВЛЕНИЕ
использования знаний
для достижения определенных целей

ПРОГНОЗИРОВАНИЕ
использование смысла причинно-следственных зависимостей в предметной области для предсказания поведения объекта управления в условиях действия определенных факторов

АНАЛИЗ
выявление смысла в данных, т.е. выявление в них причинно-следственных взаимосвязей

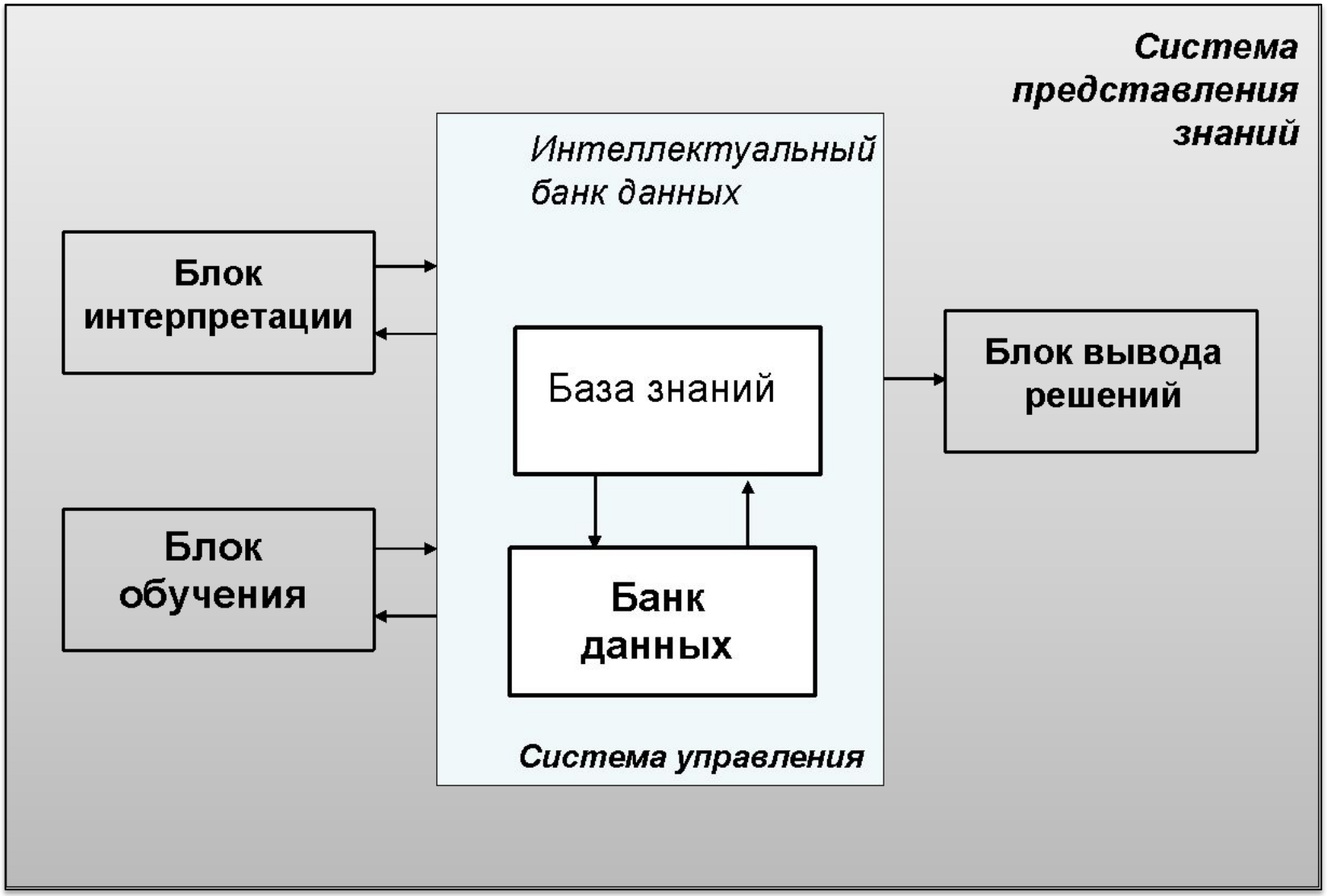
МОНИТОРИНГ
накопление данных (эмпирических фактов) по ряду показателей об объекте управления с привязкой ко времени

Банки и базы знаний

Банк знаний (БнЗ) — это информационная система представления знаний, ядром которой является **интеллектуальный банк данных** (банк данных + база знаний)

База знаний (БЗ) — это **структурированная совокупность знаний** предметной области, записанная на машинный носитель в форме, понятной эксперту и пользователю (обычно на некотором **языке представления знаний**, приближенном к естественному языку)

Информационная модель системы представления знаний



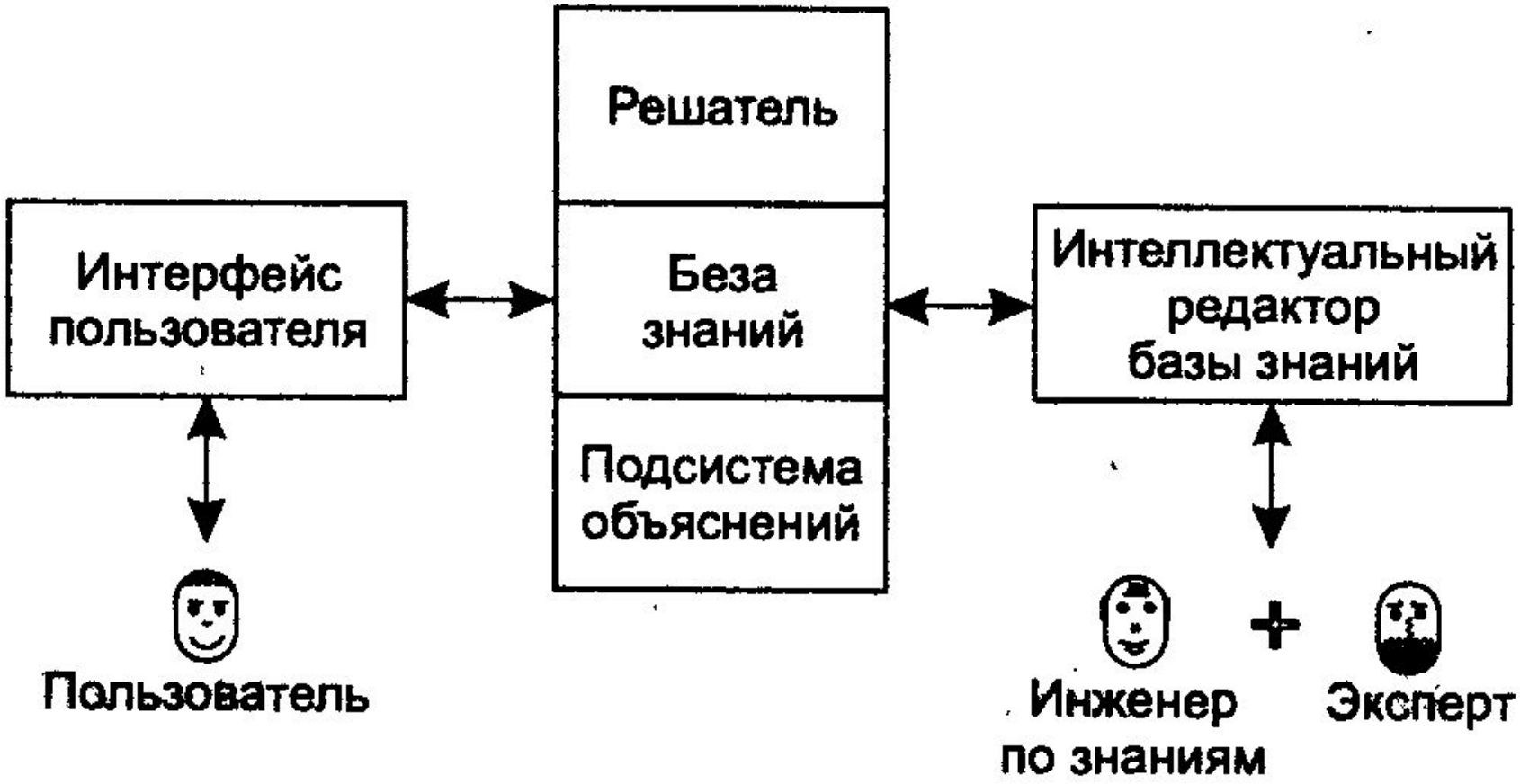
Экспертные системы

Наибольшее применение на практике находит такая разновидность БнЗ, как **экспертные системы (ЭС)**

Экспертные системы (ЭС) — это сложные программные комплексы, которые:

- аккумулируют знания специалистов (экспертов) в конкретных предметных областях
- предоставляют эти знания для консультаций менее квалифицированных пользователей

Структура экспертной системы



Задачи, решаемые экспертными системами

- Интерпретация данных
- Диагностика
- Мониторинг
- Проектирование
- Прогнозирование
- Планирование
- Обучение
- Управление
- Поддержка принятия решений

Литература к теме 7

1. Т. А. Гаврипова. В. Ф. Хорошевский
   Базы знаний интеллектуальных систем
   — СПб: Питер. 2000. — 384 с.
   - **Глава 1.** Введение интеллектуальные системы
     пункты 1.1, 1.2, 1.3
   - **Глава 2.** Разработка систем, основанных на знаниях
     2.1. Введение в экспертные системы
     2.2. Классификация систем, основанных на знаниях

2. Ю.А. Григорьев, Г.И. Ревунков
   Банки данных: Учебник для вузов.
   — М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. — 320 с.
   - **Пункт 1.1.** Информация, данные, знания
   - **Пункт 1.6.** Архитектура банков знаний