МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
ФГБОУ ВО «УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информатики и робототехники

КАФЕДРА
Автоматизированных систем управления

Дисциплина

«БАЗЫ ДАННЫХ»

Старший преподаватель кафедры АСУ

Демченко Марина Сергеевна

IT-инфраструктура

Инфраструктура данных

- Базы данных
- Хранилища данных
- Хранилища документов

- СУБД
- Системы управления хранилищами данных

- Регламентирующие документы, доступ

Техническая инфраструктура

- Аппаратные средства вычислительной техники (серверы, рабочие станции, накопители……)

- Вычислительные сети
- Каналы связи, аппаратура
- Коммуникационные устройства
- Протоколы, сервисы

- Регламентирующие документы, аварийные планы

Программная инфраструктура

Общесистемное программное обеспечение:

- Операционные системы
- Утилиты и программы
- Интерфейсы взаимодействия
- Методы и средства разработки приложений

Прикладное программное обеспечение:

- Приложения и прикладные системы для пользователей для исполнения бизнес-функций и бизнес-процессов

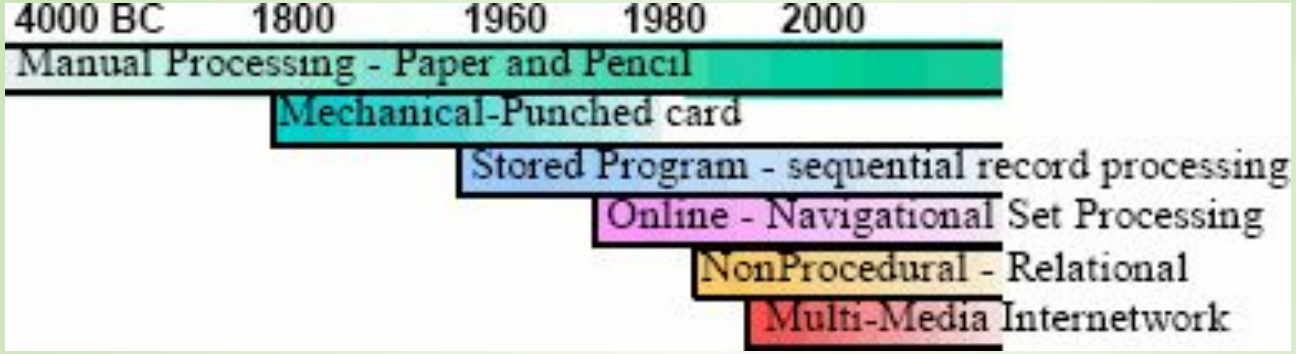
Информационные системы. История

- Вычисления
- Системы управления файлами
- Информационные системы

Предпосылки появления баз данных:

- необходимость хранить и обрабатывать большой объем данных
- совместное использование информации
- внешние носители

Управление данными



0 : менеджеры записей (4000 г. до н. э. – 1900)
1 : менеджеры записей (1900-1955) перфокарты
2 : программируемое оборудование обработки записей (1955-1970)
3 : оперативные сетевые базы данных (1965-1980)
4 : реляционные базы данных и архитектура клиент-сервер (1980-1995)
5: мультимедийные базы данных (1995-...)
6: NoSQL (2000-…)

Базы данных: термины

Информация – сведения о каком-либо событии, объекте или процессе.

Данные – информация, представленная в определенном виде, позволяющем автоматизировать ее сбор, хранение и дальнейшую обработку. Для компьютерных технологий данные — в виде, удобном для хранения, обработки на ЭВМ, а также для передачи по каналам связи.

Управление данными – совокупность функций для требуемого представления данных, их накопления и хранения, обновления, удаления, поиска по заданному критерию и выдачи данных. [ГОСТ 20886-85]

База данных (БД) – поименованная совокупность взаимосвязанных структурированных данных, которая отражает состояние объектов и их отношений в рассматриваемой предметной области, причем они организованы так, чтобы обеспечить независимость данных от программ обработки.

Предметная область – информация о части реального мира, подлежащая изучению с целью организации управления и автоматизации.

Система управления базами данных (СУБД) – это совокупность программ и языковых средств, предназначенных для управления данными в базе данных, ведения базы данных и обеспечения взаимодействия её с прикладными программами [ГОСТ 20886-85].

План лекций

Раздел 1.
Основы теории проектирования баз данных

Раздел 2.
Технологии создания и преобразования информационных объектов

Раздел 3.
Организация баз данных

Раздел 4.
Управление базами данных в СУБД

База данных: общее понятие

База данных:

- хранилище информации
- отражает объект реального мира
- имитирует деятельность объекта реального мира

База данных – определение:

База данных – совокупность структурированных данных, хранящихся с минимальной (структурированной) избыточностью и используемых несколькими приложениями под управлением единого метода доступа.

Модели баз данных: историческое развитие

- Двумерный файл
- Иерархическая модель
- Сетевая модель
- Реляционная модель
- Объектно-реляционная модель

База данных: пример

База данных школы №1:

- Преподаватели: Иванов, Кузнецов
- Ученики: Петров, Сидоров, Федоров, Семенов, Алексеев
- Обучение:

Петров, Сидоров, Федоров - преподаватель Иванов

Петров, Семенов, Алексеев - преподаватель Кузнецов

Сетевая модель

- База данных - файл
- Записи логически организованы в виде сети
- Произвольное отношение подчиненности: ветвь может иметь более одного корня

Сетевая модель: пример

Реляционная модель

- **Структура:** данные хранятся в виде совокупности двумерных таблиц (отношений)
- **Целостность:** Существуют ограничения - структуры для обеспечения непротиворечивости и целостности базы данных
- **Манипулирование:** Существует набор операторов для воздействия на отношения (обновление содержимого отношений, создание новых отношений)

Реляционная модель: структура

- База данных - файл; набор файлов; сервер
- Таблица - основная структура хранения данных, состоит из строк и столбцов (количество столбцов - более одного, количество строк - не ограничено)
- В столбце содержатся данные одного типа
- Строки должны быть уникальными
- Поле находится на пересечении строки и

Реляционная модель: структура

Школы

№
1

Преподаватели

№	Фамилия	№ школы
1	Иванов	1
2	Кузнецов	1

Ученики

№	Фамилия	№ школы
1	Петров	1
2	Сидоров	1
3	Федоров	1
4	Семенов	1
5	Алексеев	1

Обучение

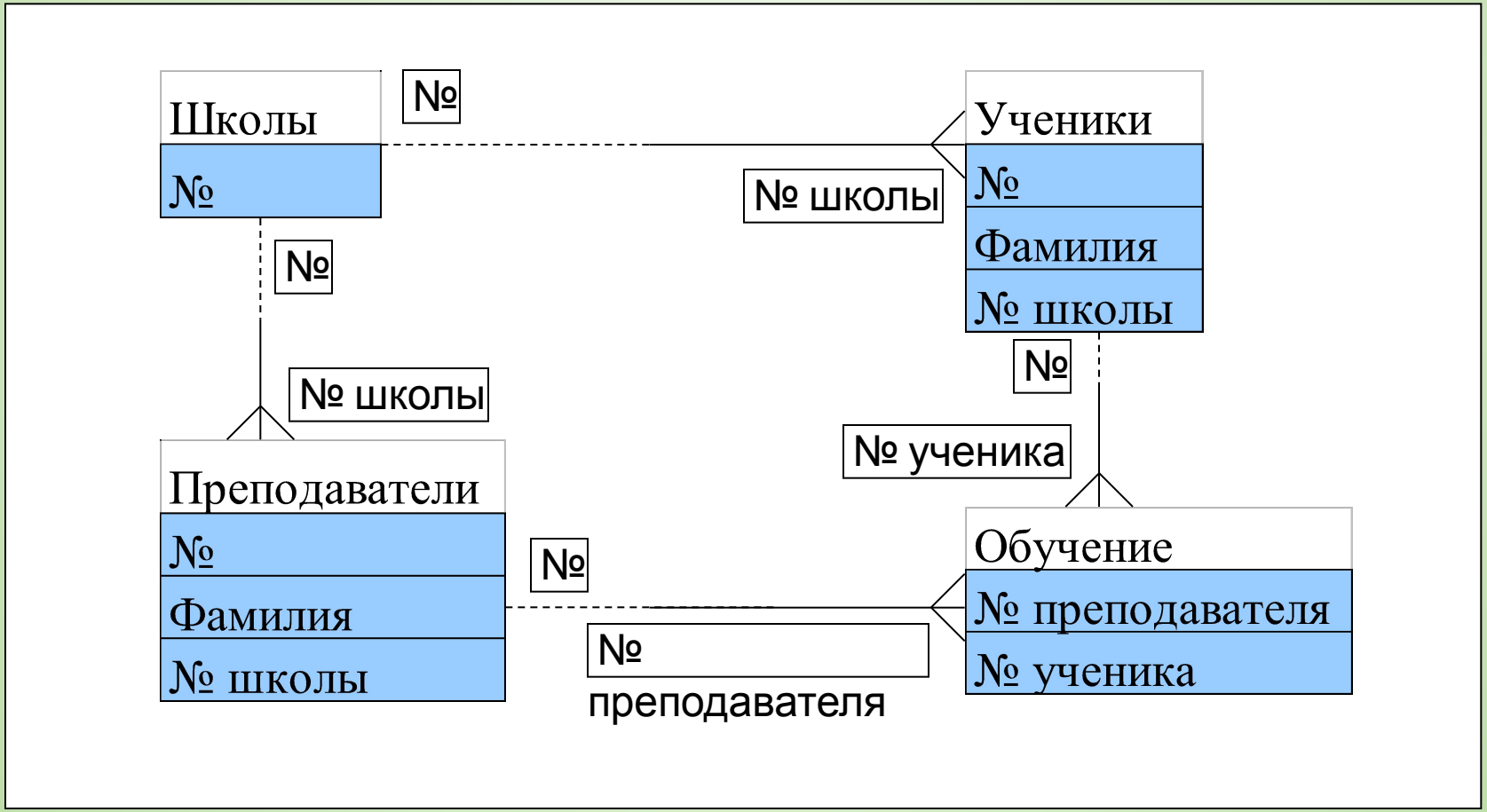
№ преподавателя	№ ученика
1	1
1	2
1	3
2	1
2	4
2	5

Реляционная модель: целостность

Ограничения, направленные на обеспечение целостности:

- **Первичный ключ (Primary key)** - уникальный идентификатор каждой строки в таблице, предотвращает избыточность данных
- **Внешний ключ (Foreign key)** - ссылка на первичный ключ в той же самой или другой таблице, обеспечивает непротиворечивость и целостность данных

Реляционная модель: целостность



Аппаратное и программное обеспечение баз данных

Компоненты *организационного обеспечения БД:*

Прикладные программисты

Конечный пользователь

Метаданные

Администратор

Аналитик данных

Аппаратное обеспечение БД:

- - тома вторичной (внешней) памяти (обычно это магнитные диски), используемые для хранения информации, а также соответствующие устройства ввода-вывода (дисководы и т.д.), контроллеры устройств, каналы ввода-вывода и т.д.
- - аппаратный процессор (или процессоры) вместе с основной (первичной) памятью, предназначенные для поддержки работы ПО системы БД.

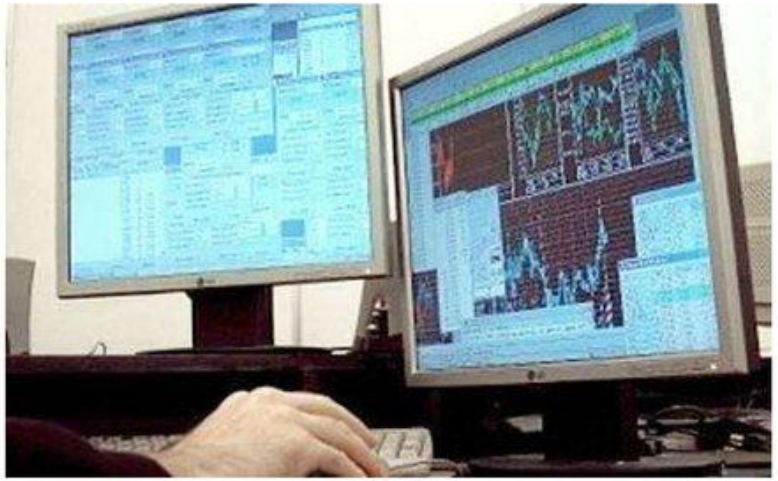
Программное обеспечение БД

- Между собственно физической БД (т.е. данными, которые реально хранятся) и пользователями системы располагается уровень ПО, который носит название СУБД (система управления базами данных) (database management system – DBMS). Основная задача СУБД – предоставить пользователю БД возможность работать с ней, не вникая в детали на уровни аппаратного обеспечения.

История появления и развития БД

Появление мощных рабочих станций и сетей ЭВМ повлияло также и на развитие технологии баз данных.

Можно выделить **четыре этапа** в развитии данного направления в обработке данных.



История появления и развития БД

Первый этап развития БД связан с организацией баз данных на больших ЭВМ типа IBM 360/370, ЕС-ЭВМ и мини-ЭВМ типа PDP11 (фирмы Digital Equipment Corporation — DEC), разных моделях HP (фирмы Hewlett Packard).

Характеристика БД первого этапа:

- Базы данных хранились во внешней памяти центральной ЭВМ.
- Программы доступа к БД писались на различных языках.
- Доступ к БД поддерживался от многих пользователей-задач (распределенный доступ).

История появления и развития БД

Второй этап - появление персональных компьютеров

Появились первые программы - **системы управления базами данных,** которые позволяли хранить значительные объемы информации, они имели удобный интерфейс для заполнения данных, встроенные средства для генерации различных отчетов. Появились настольные СУБД. Эти программы позволяли автоматизировать многие **учетные** функции, которые раньше велись вручную.

Характеристика БД второго этапа:

- Все СУБД были рассчитаны на создание БД в основном с монопольным доступом.
- В настольных СУБД поддерживались низкоуровневые языки манипулирования данными на уровне отдельных строк таблиц.
- Сравнительно скромные требования к аппаратному обеспечению со стороны настольных СУБД.

История появления и развития БД

Третий этап - распределенные базы данных

Появившиеся *распределенные базы данных,* сохраняют все преимущества настольных СУБД и в тоже время позволяют организовать параллельную обработку информации и поддержку целостности БД

Характеристика БД третьего этапа

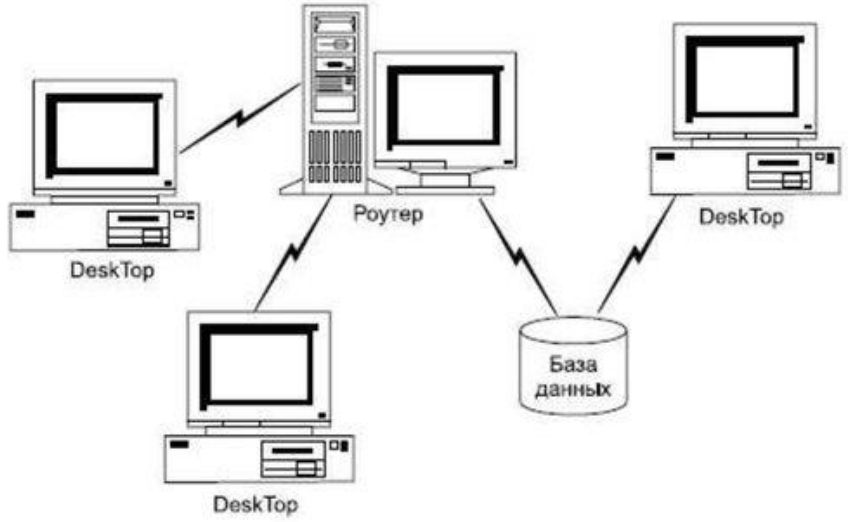
- Практически все современные СУБД обеспечивают поддержку полной реляционной модели.
- Большинство современных СУБД рассчитаны на многоплатформенную архитектуру.
- Необходимость поддержки многопользовательской работы с базой данных и возможность децентрализованного хранения данных потребовали развития средств администрирования БД с реализацией общей концепции средств защиты данных.

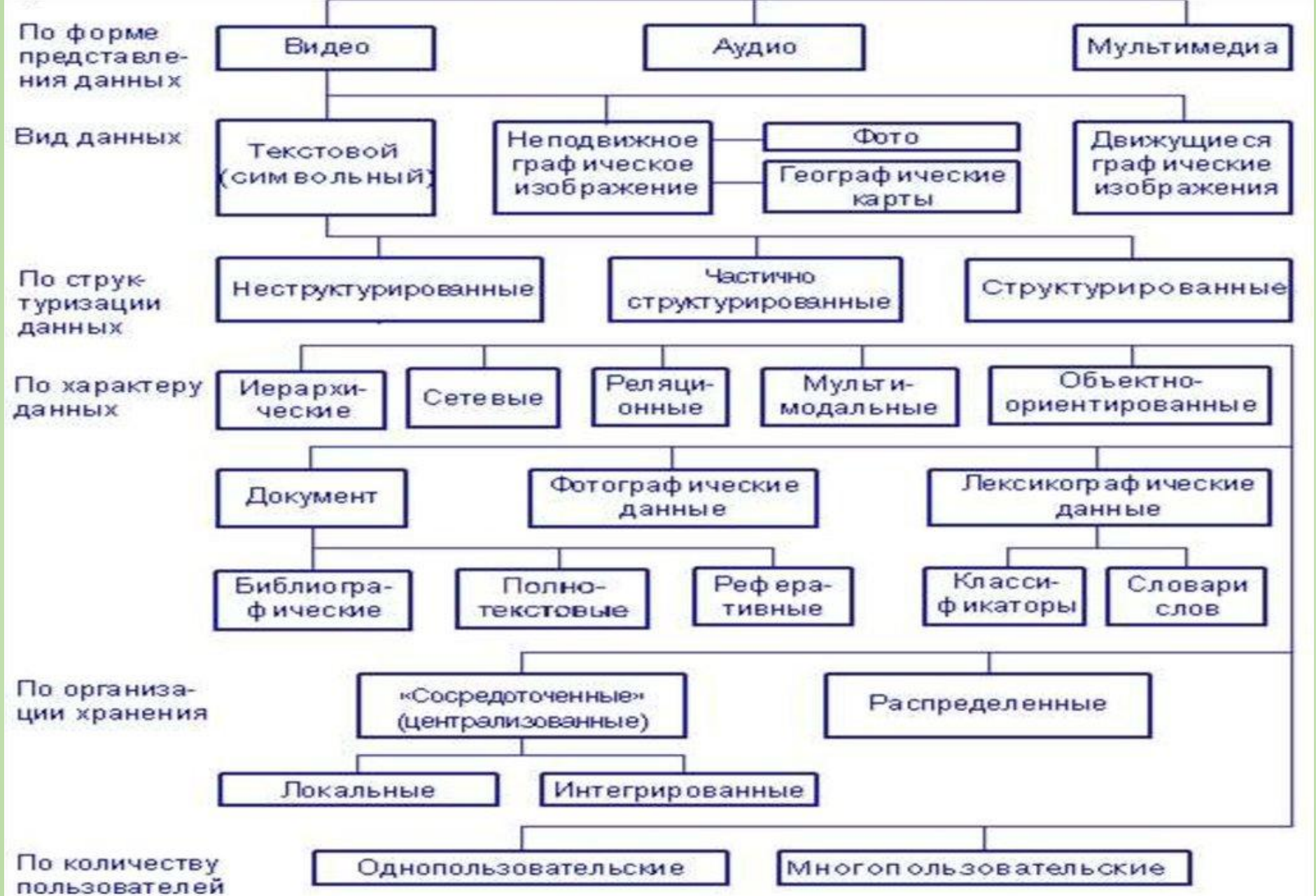
История появления и развития БД

Четвертый этап – сетевой доступ к системе управления базами данных

Этот **этап** характеризуется появлением новой технологии доступа к данным — ***интранет***. Основное отличие этого подхода от технологии клиент-сервер состоит в том, что отпадает необходимость использования специализированного клиентского программного обеспечения.

Для работы с удаленной базой данных используется стандартное программное приложение - браузер Интернета и для конечного пользователя процесс обращения к данным происходит аналогично работе с ресурсами в сети Интернет.





Классификация *БД по* основным признакам

Понятие систем управления базами данных (СУБД)

СУБД представляет собой ПО, которое управляет всем доступом к БД.

Концептуально это происходит так:

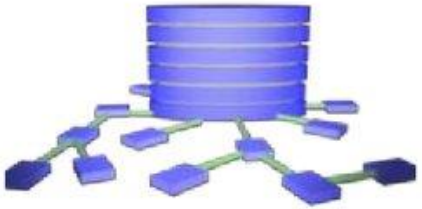
- 1. Пользователь выдает запрос на доступ к данным, предъявляя определенный подъязык данных (обычно это SQL).
- 2. СУБД перехватывает этот запрос и анализирует его.
- 3. СУБД выполняет необходимые операции в хранимой БД с учетом прав пользователя.

Систе́ма управле́ния ба́зами да́нных (**СУБД**) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

История появления и развития БД: поколения СУБД

К **СУБД первого поколения** относят СУБД на основе *сетевой модели данных* (их иногда называют CODASYL-системы) и системы на основе иерархических подходов.

CODASYL (англ. COnference on DAta SYstems Language — Конференция по языкам систем обработки данных) — организация (название произносится «кодасил»), принимавшая активное участие в эволюции информационных технологий в 60-80-е годы XX века.



СУБД второго поколения – *реляционные.*

СУБД третьего поколения – *объектно-реляционные и объектно-ориентированные.*

СУБД включает в себя:

- - процессор ЯОД (языка обработки данных), предназначенный для задания внутренней и внешней структуры БД.
- - процессор ЯМД (языка манипулирования данными) для того, чтобы обрабатывать запросы пользователей на выбору, изменение или удаление данных, уже имеющихся в БД, или на добавление в нее новых данных.

В целом, назначением СУБД является предоставление пользовательского интерфейса с базой данных.

Пользовательский интерфейс может быть определен как существующий в системе ограничительный уровень, ниже которого для пользователя все остается невидимым.

Классификация СУБД

В силу многогранности баз данных и СУБД (комплекса технических и программных средств, для хранения, поиска, защиты и использования данных) имеется множество классификационных признаков:

По степени универсальности (сфере применения) :

- СУБД общего назначения (СУБД ОН) и специализированные СУБД (СпСУБД).

По используемой модели данных

- иерархические, сетевые, реляционные; объектно-ориентированные СУБД.

По методам организации хранения и обработки данных :

- централизованные (локальные, файл – серверные, клиент-серверные) и распределённые СУБД.

По сфере применения

- справочные системы и системы обработки данных.

Классификация по масштабу систем:

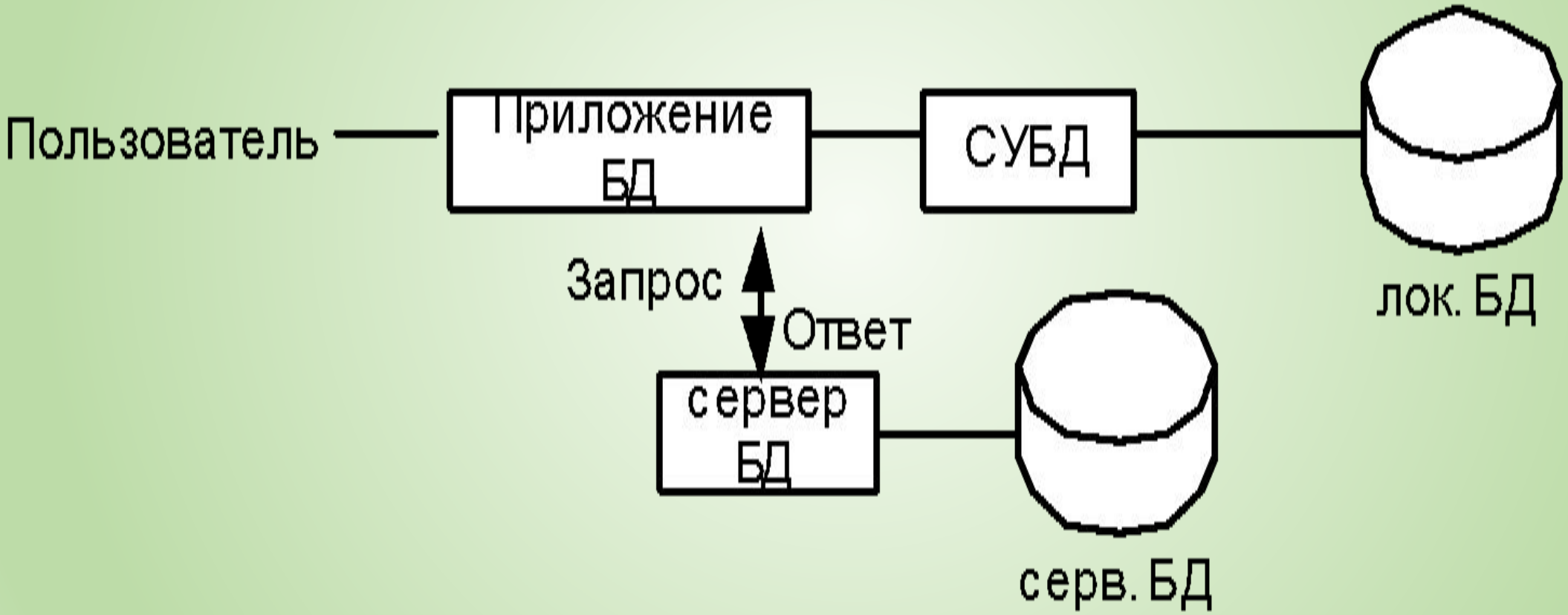
- персональные; уровня группы, отдела, предприятия; корпоративные; географически распределенные.

Архитектура взаимодействия приложений с базой данных.

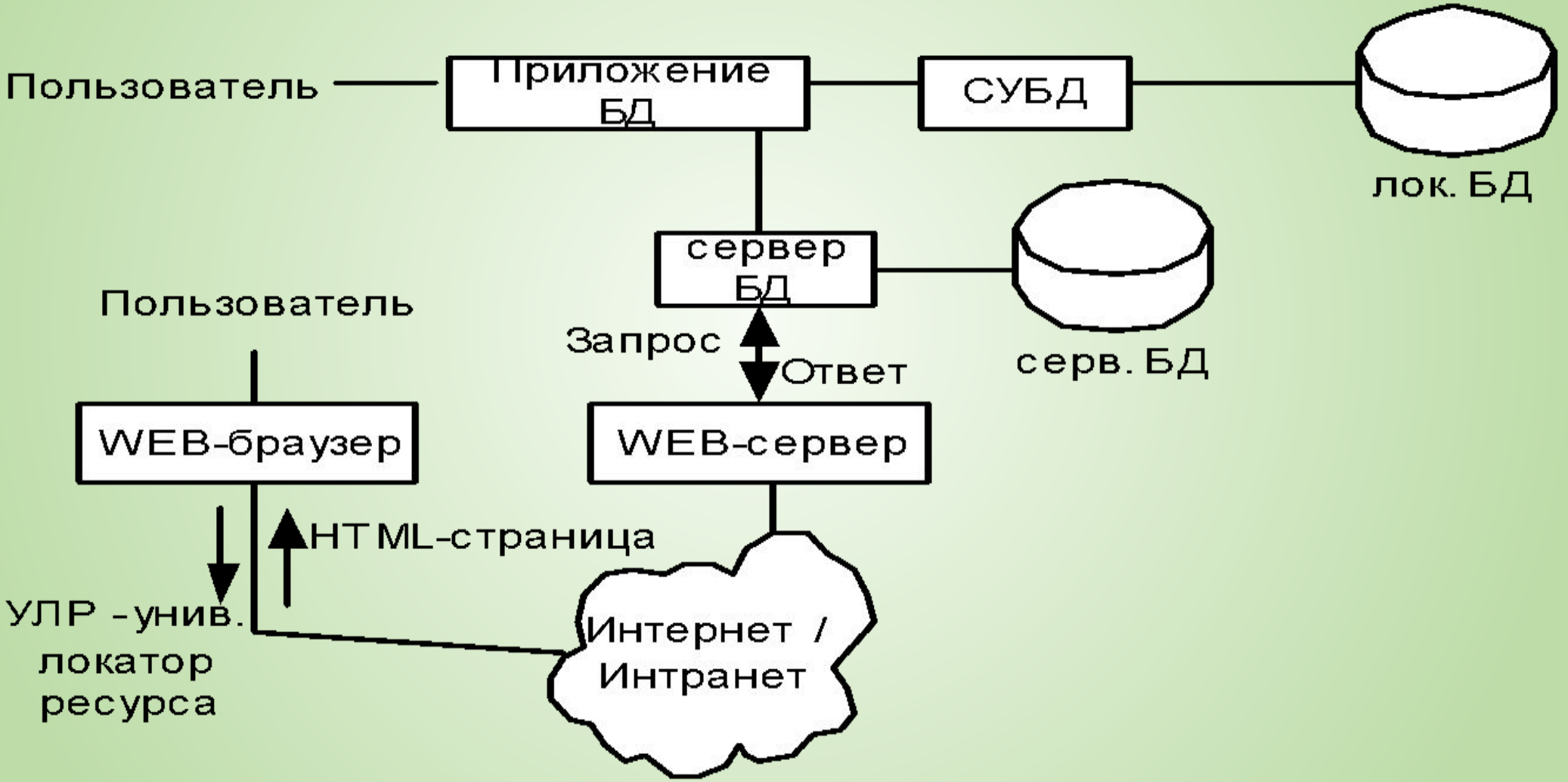
3 типа архитектур:
файл-сервер, клиент-сервер, трехуровневая архитектура.

Файл-сервер

Клиент-сервер (2-уровневый)



Трехуровневая архитектура



Большинство систем будут выполнять следующие три основные задачи, соответствующие трем уровням n-уровневой модели:

- 1. Интерфейс пользователя и перемещение. Этот уровень заключает все возможности работы пользователя. Этот уровень не только обеспечивает графический интерфейс пользователя, позволяющий взаимодействовать с приложением, вводить данные и просматривать результаты запросов, но и управляет манипулированием и форматированием данных, полученных клиентом. В веб-приложениях задачи этого уровня выполняются веб-обозревателем.
- 2. Бизнес-логика. Уровень 2, между интерфейсом и уровнями служб данных — это «владения» разработчика распределенного приложения. Бизнес-логика, которая собирает правила, управляющие обработкой приложения, соединяет пользователя, представляющего одну сторону, с данными — другой стороной. Функции, которыми управляют правила, имитируют ежедневные деловые задачи и могут быть простой задачей или последовательностью задач.
- 3. Службы данных. Службы данных предоставляются структурированными (база данных SQL, Oracle) и неструктурированными хранилищами данных (Microsoft Exchange, Microsoft Message Queuing), которые управляют доступом и обеспечивают доступ к данным приложения. Одно приложение может привлекать службы одного или нескольких хранилищ данных.

Спасибо за внимание!

Модели данных, поддерживаемые СУБД

Иерархическая, сетевая и реляционная модели данных. Отличительные особенности этих моделей данных

Прежде всего модели данных можно разделить на реляционные и нереляционные.

Нереляционные делятся, в свою очередь, на иерархические (hierarchic) и сетевые (network).

СУБД, базирующиеся на реляционной модели данных, стали сегодня преобладающими.

- Иерархическая модель
- Сетевая модель
- Реляционная модель

Иерархическая модель данных

Получая доступ к информации, содержащейся в БД, программа могла:

- - найти конкретную деталь по ее номеру
- - перейти вниз к первому потомку
- - перейти вверх к предку
- - перейти в сторону к другому потомку

- Пример иерархических СУБД – Information Management System (IMS) (1968). Достоинства:
- - простота модели
- - использование отношений предок/потомок
- - быстродействие

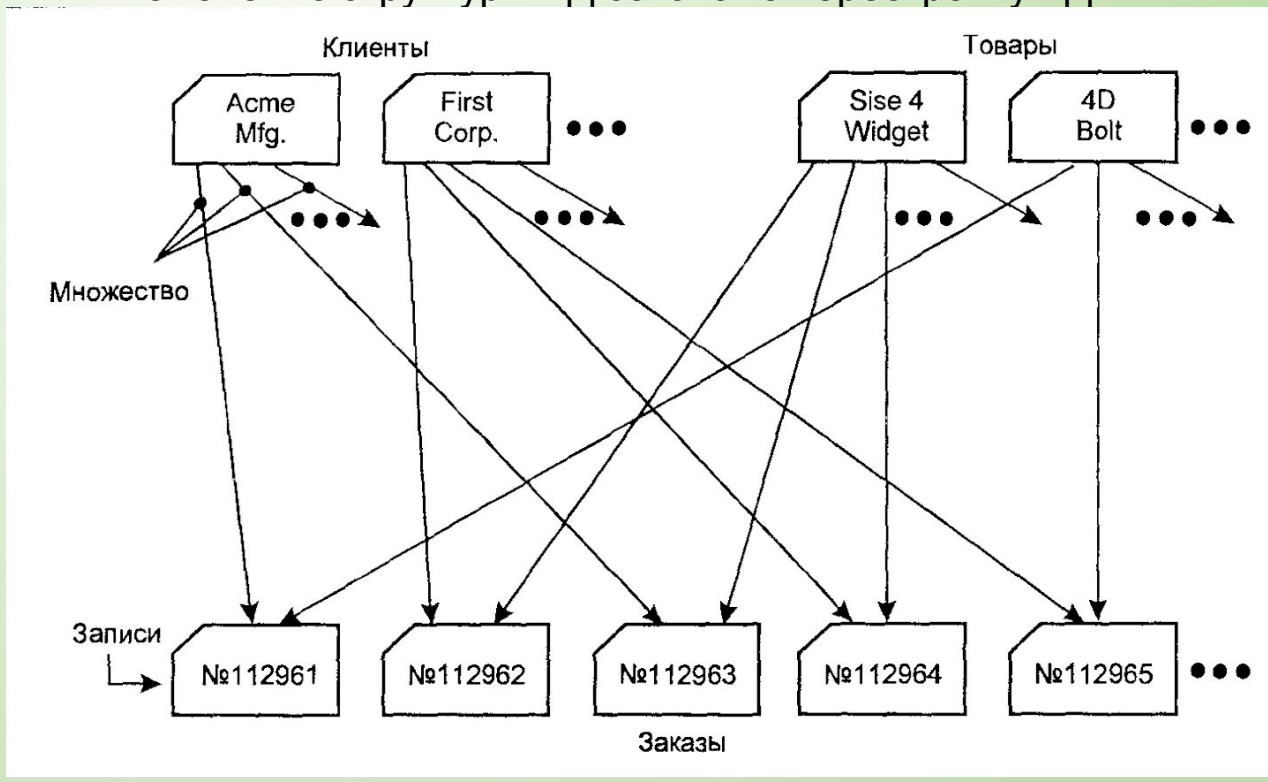
Сетевая модель данных

Стандарт сетевых баз данных – модель CODASYL. Пример СУБД –IDMS.

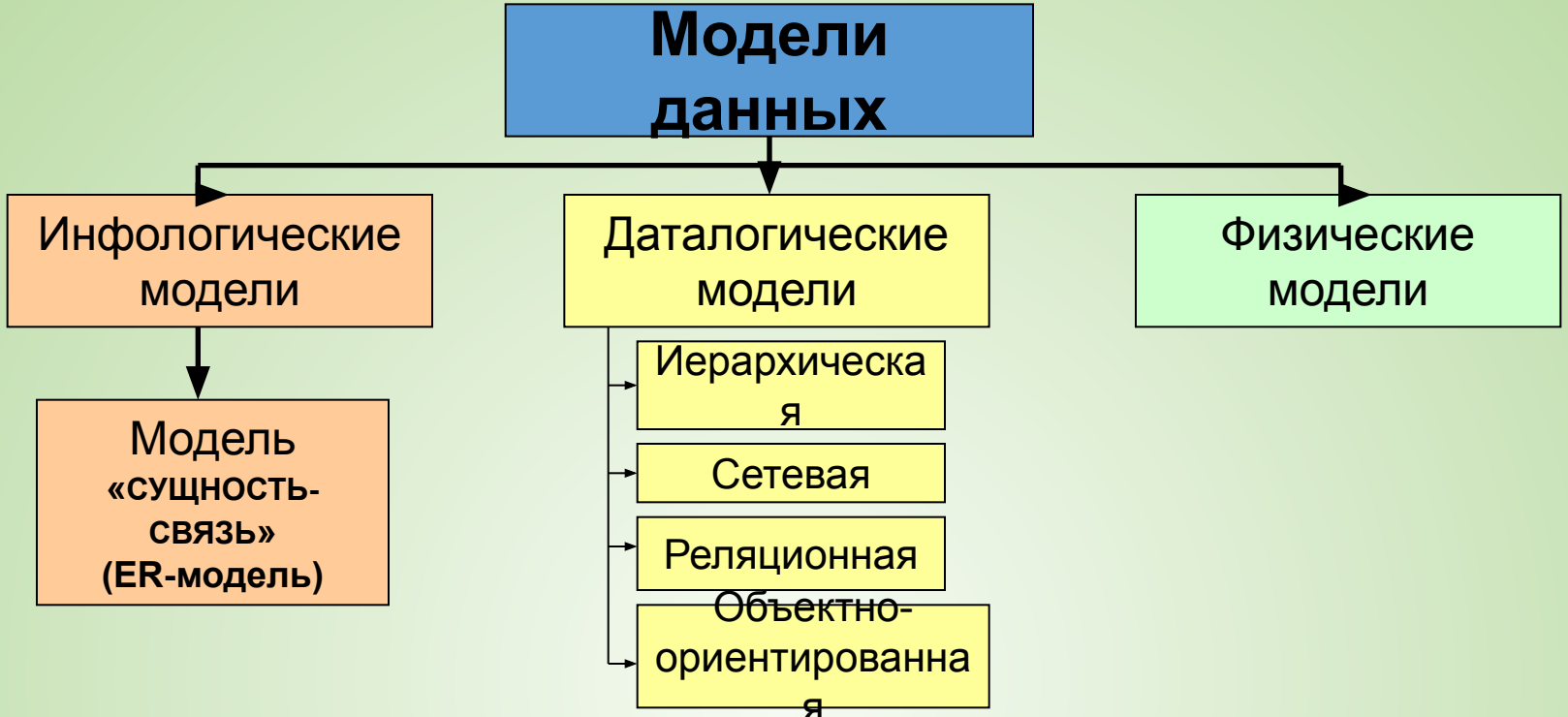
Прикладная программа могла:

- - найти конкретную запись предка по ключу (например, номер клиента)
- - перейти к первому потомку в конкретном множестве (первый заказ, размещенный клиентом)
- - перейти в сторону от одного потомка к другому в конкретном множестве (следующий заказ, сделанный клиентом)
- - перейти вверх от потомка к его предку в другом множестве (служащий, принявший заказ).

- Сетевые БД обладали рядом преимуществ:
- - гибкость
- - стандартизация
- - быстродействие
- Недостатки – те же, что и у иерархических моделей. Они были «жесткими», т.е. наборы отношений и структуру записей приходилось задавать наперед. Изменение структуры БД означало перестройку БД.




Модели данных
Инфологические модели
Даталогические модели
Физические модели
Диаграммы Бахмана
Модель «сущность-связь» (ER-модель)
Основанные на файловых структурах
Основанные на странично-сегментной организации
Документальные
Фактографические
Ориентированные на формат документа
Дескрипторные
Тезаурусные
Теоретико-графовые
Теоретико-множественные
Объектно-ориентированные
Иерархическая
Сетевая
Реляционная
Бинарных ассоциаций



Инфологическое моделирование связано со 2-м этапом проектирования БД: созданием формализованного описания предметной области

Логическое (или даталогическое) моделирование осуществляется после этапа выбора СУБД. Этот тип модели полностью зависит от типа модели, поддерживаемой выбранной системой.

Физическое моделирование заключается в выборе эффективного размещения БД на внешних носителях для обеспечения наиболее эффективной работы.

Предметная область

(часть реального мира отображаемая в базе данных)

ИНФОЛОГИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Формализованное, обобщенное, не привязанное к каким-либо СУБД описание предметной области

Модели описания, используемые СУБД

ДАТАЛОГИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Описание на языке конкретной СУБД

ФИЗИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Описание хранимых данных на уровне операционной системы

БАЗА ДАННЫХ

СУБД, базирующиеся на реляционной модели данных, стали сегодня преобладающими. Что мы подразумеваем под реляционной системой?

Реляционная система – это система, основанная на следующих принципах:

- - данные передаются пользователю в виде таблиц (и не как иначе)
- - пользователю предоставляются операторы (например, для выборки данных), позволяющие генерировать новые таблицы на основе уже существующих.

Требования к реляционным СУБД

Э.Ф. Кодд предложил схему представления данных в виде таблиц, называемых *отношениями* (relations), и охарактеризовал требования к реляционным СУБД (Dr. E. F. Codd's 12 rules for defining a fully relational database, 1985 г.):

1. реляционная СУБД должна быть способна полностью управлять базой данных через ее **реляционные возможности**;
2. **информационное правило** — вся информация должна определяться строго как значения в таблицах;
3. **гарантированный доступ**
4. **поддержка пустых значений** (null value);
5. **доступ к словарю данных** в терминах реляционной модели
6. **единственный язык запросов**, который позволяет выполнять все операции работы к данным

Требования к реляционным СУБД (по Кодду)

7. поддержка **обновляемых представлений** (View Updating Rule).
8. Операции вставки, модификации и удаления данных должны поддерживаться по отношению к любому множеству строк.
9. **физическая независимость данных**.
10. **логическая независимость данных**.
11. **независимость контроля целостности**. СУБД должна выполнять проверку заданных **ограничений целостности** и автоматически поддерживать целостность данных.
12. **независимость от распределенности**.
13. **согласование языковых уровней**. Если используется низкоуровневый язык доступа к данным, он не должен игнорировать правила безопасности и целостности, которые поддерживаются языком более высокого уровня.

Концепция реляционных баз данных
Общая характеристика реляционной модели данных

Для реляционной модели данных характерны 3 признака:

1. Данные в базе воспринимаются пользователем как таблицы (и не как иначе)

2. Эти таблицы удовлетворяют определенным условиям целостности

3. В распоряжении пользователя имеются операторы манипулирования данными (например, выборки информации), которые генерируют новые таблицы на основании уже имеющихся и среди которых есть, по крайней мере, операторы выборки, проекции и объединения.

поставщик

код	имя	статус	город
s1	Смит	20	Лондон
s2	Джонс	10	Париж
s3	Блейк	30	Париж
s4	Кларк	20	Лондон
s5	Адамс	30	Афины

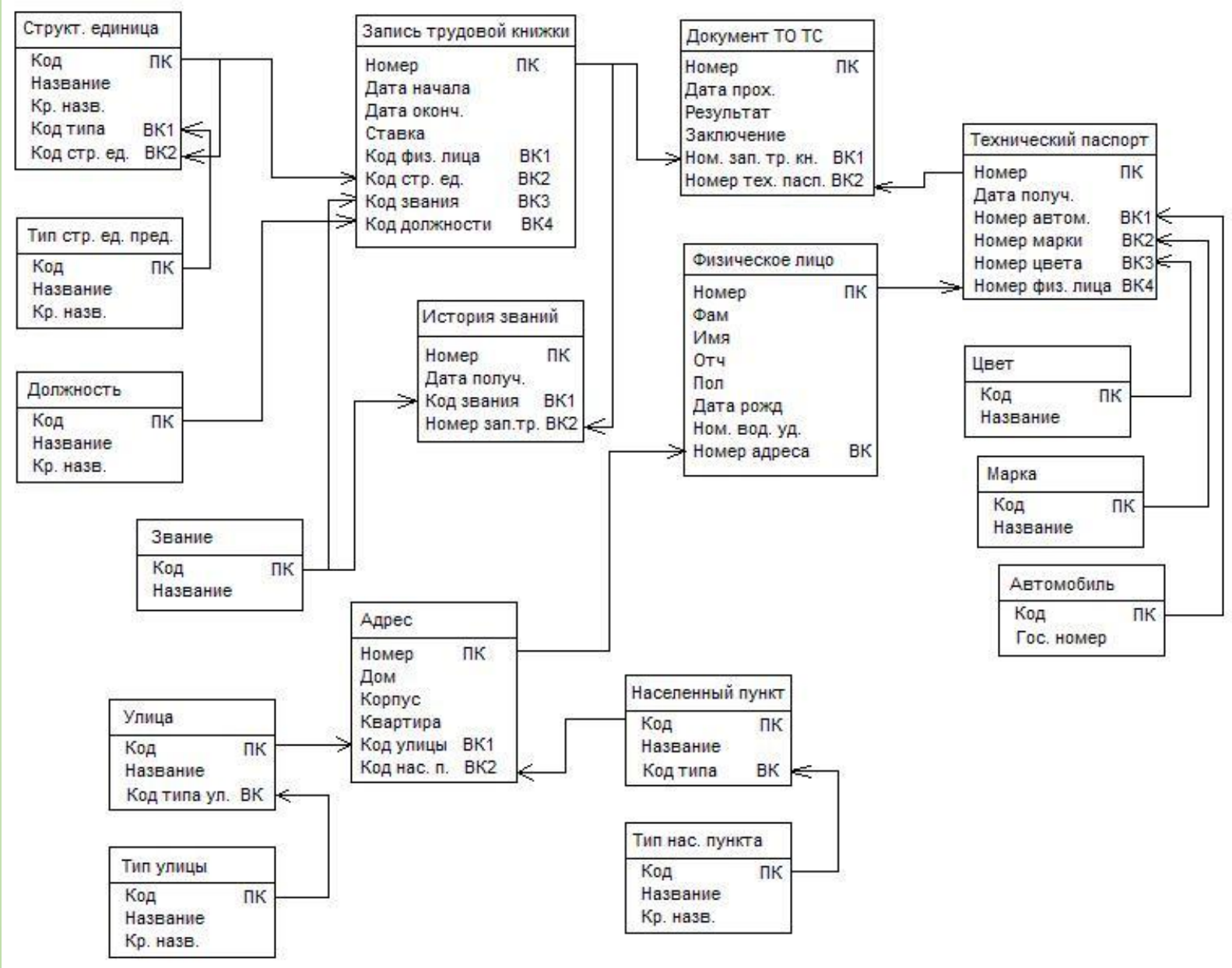
товар

код	назв	цвет	вес	город
p1	Гайка	красный	12	Лондон
p2	Болт	белый	17	Париж
p3	Шайба	синий	17	Рим
p4	Шайба	красный	14	Лондон
p5	Винт	синий	12	Париж
p6		красный	19	Лондон

поставка

код-п	код-п	кол
s1	p1	300
s1	p2	200
s1	p3	400
s1	p4	200
s1	p5	100
s1	p6	100
s2	p1	300
s2	p2	400
s3	p2	200
s4	p2	200
s4	p4	300
s4	p5	400

Реляционная модель данных



Основные термины реляционной базы данных:

Отношение – информация об объектах одного типа. Отношение обычно хранится в виде *таблицы*. *Свойства таблиц.*

Атрибут – определенная часть информации о некотором объекте. Атрибут обычно хранится в виде *столбца* или *поля таблицы*.

Связь – способ, которым связана информация в одной таблице с информацией в другой таблице. Типы связей

Объединение – процесс объединения таблиц или запросов на основе совпадающих значений определенных атрибутов.

Свойства реляционных таблиц

- каждый элемент таблицы - один элемент данных;

- все записи в столбцах таблицы однородные, т.е. все элементы в столбце имеют одинаковый тип (числовой, символьный и т.д.) и длину;

- каждый столбец имеет уникальное имя;

- одинаковые строки в таблице отсутствуют;

- порядок следования строк и столбцов может быть произвольным.

Реляционная база данных

Все объекты в реляционной базе данных связаны между собой.

Различают связи нескольких типов:

Один к одному (1:1)

любая запись в первой таблице связана только с одной записью во второй таблице и наоборот *(создается между ключевыми полями таблиц)*

Один ко многим (1:М)

любая запись в первой таблице может быть связана с несколькими записями во второй, но в то же время любая запись второй таблицы связана только с одной записью первой

Многие ко многим (М:М)

многие записи одной таблицы соответствуют многим записям из другой таблицы

Понятие ключевого поля

Поле, каждое значение которого однозначно определяет соответствующую запись, называется **простым ключом (ключевым полем).**

Если записи однозначно определяются значениями нескольких полей, то такая таблица БД имеет **составной ключ.**

Чтобы связать две реляционные таблицы, необходимо **ключ первой таблицы ввести в состав ключа второй таблицы** (связь 1:1).

Система управления базами данных

Систе́ма управле́ния ба́зами да́нных (**СУБД**) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных[1].

Основные функции СУБД:

- управление данными во внешней памяти (на дисках);
- управление данными в оперативной памяти с использованием дискового кэша. (**Кэш** или **кеш** — промежуточный буфер с быстрым доступом, содержащий информацию, которая может быть запрошена с наибольшей вероятностью. Доступ к данным в кэше осуществляется быстрее, чем выборка исходных данных из более медленной памяти или удаленного источника);
- журнализация изменений(**Журнализация изменений** — функция СУБД, которая сохраняет информацию, необходимую для восстановления базы данных в предыдущее согласованное состояние в случае логических или физических отказов), резервное копирование и восстановление базы данных после сбоев;
- поддержка языков БД (язык определения данных, язык манипулирования данными).

Data Definition Language

- **Data Definition Language** (**DDL**) (язык описания данных) — это семейство компьютерных языков, используемых в компьютерных программах для описания структуры баз данных.
- На текущий момент наиболее популярным языком DDL является SQL, используемый для получения и манипулирования данными в РСУБД, и сочетающий в себе элементы DDL, DML и DCL.
- Функции языков DDL определяются первым словом в предложении (часто называемом **запросом**), которое почти всегда является глаголом. В случае с SQL это глаголы — «create» («создать»), «alter» («изменить»), «drop» («удалить»). Эти запросы или команды часто смешиваются с другими командами SQL, в связи с чем DDL не является отдельным компьютерным языком.
- Запрос «create» используется для создания базы данных, таблицы, индекса, представления или хранимой процедуры. Запрос «alter» используется для изменения существующего объекта базы данных (таблицы, индекса, представления или хранимой процедуры) или самой базы данных. Запрос «drop» используется для удаления существующего объекта базы данных (таблицы, индекса, представления или хранимой процедуры) или самой базы данных. И наконец, в DDL существуют понятия первичного и внешнего ключа, которые осуществляют соблюдение целостности данных. Команды "первичный ключ" primary key, "внешний ключ" foreign key включаются в запросы «create table", «alter table».[1][1]
- Языки DDL могут существенно различаться у различных производителей СУБД. Существует ряд стандартов SQL, установленный ISO/IEC (SQL-89,SQL-92, SQL:1999, SQL:2003, SQL:2008), но производители СУБД часто предлагают свои собственные «расширения» языка и,

Data Manipulation Language

- **Data Manipulation Language** (**DML**) (язык управления (манипулирования) данными) — это семейство компьютерных языков, используемых в компьютерных программах или пользователями баз данных для получения, вставки, удаления или изменения данных в базах данных.
- На текущий момент наиболее популярным языком DML является SQL, используемый для получения и манипулирования данными в РСУБД. Другие формы DML использованы в IMS/DL1, базах данных CODASYL (таких как IDMS), и других.
- Языки DML изначально использовались только компьютерными программами, но с появлением SQL стали также использоваться и людьми.
- Функции языков DML определяются первым словом в предложении (часто называемом **запросом**), которое почти всегда является глаголом. В случае с SQL эти глаголы — «select» («выбрать»), «insert» («вставить»), «update» («обновить»), и «delete» («удалить»). Это превращает природу языка в ряд обязательных утверждений (команд) к базе данных.
- Языки DML могут существенно различаться у различных производителей СУБД. Существует стандарт SQL, установленный ANSI, но производители СУБД часто предлагают свои собственные «расширения» языка.
- Языки DML разделяются в основном на два типа:
- Procedural DMLs — описывают действия над данными.
- Declarative DMLs — описывают сами данные.

Состав СУБД

Обычно современная СУБД содержит следующие компоненты:

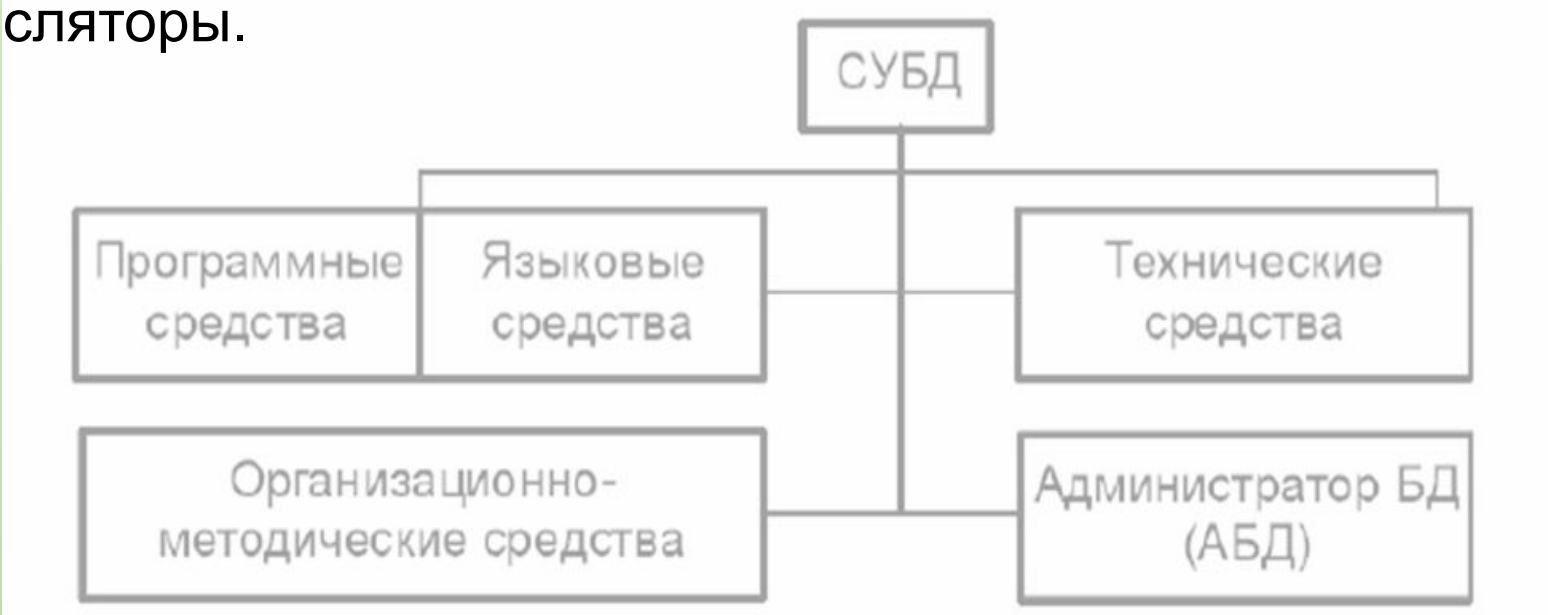
- **ядро**, которое отвечает за управление данными во внешней и оперативной памяти и журнализацию,
- **процессор языка базы данных**, обеспечивающий оптимизацию запросов (1) функция СУБД, осуществляющая поиск оптимального плана выполнения запросов из всех возможных для заданного запроса, 2) процесс изменения запроса и/или структуры БД с целью уменьшения использования вычислительных ресурсов при выполнении запроса. Один и тот же результат может быть получен СУБД различными способами (планами выполнения запросов), которые могут существенно отличаться как по затратам ресурсов, так и по времени выполнения. Задача оптимизации заключается в нахождении оптимального способа) на извлечение и изменение данных и создание, как правило, машинно-независимого исполняемого внутреннего кода,
- **подсистему поддержки времени исполнения**, которая интерпретирует программы манипуляции данными, создающие пользовательский интерфейс с СУБД
- **сервисные программы** (внешние утилиты — вспомогательная компьютерная программа в составе общего программного обеспечения для выполнения специализированных типовых задач, связанных с работой оборудования и операционной системы, утилиты предоставляют доступ к возможностям (параметрам, настройкам, установкам), недоступным без их применения, либо делают процесс изменения некоторых параметров проще (автоматизируют его), утилиты могут входить в состав операционных систем, идти в комплекте со специализированным оборудованием или распространяться отдельно)), обеспечивающие ряд дополнительных возможностей по обслуживанию информационной системы

Состав СУБД

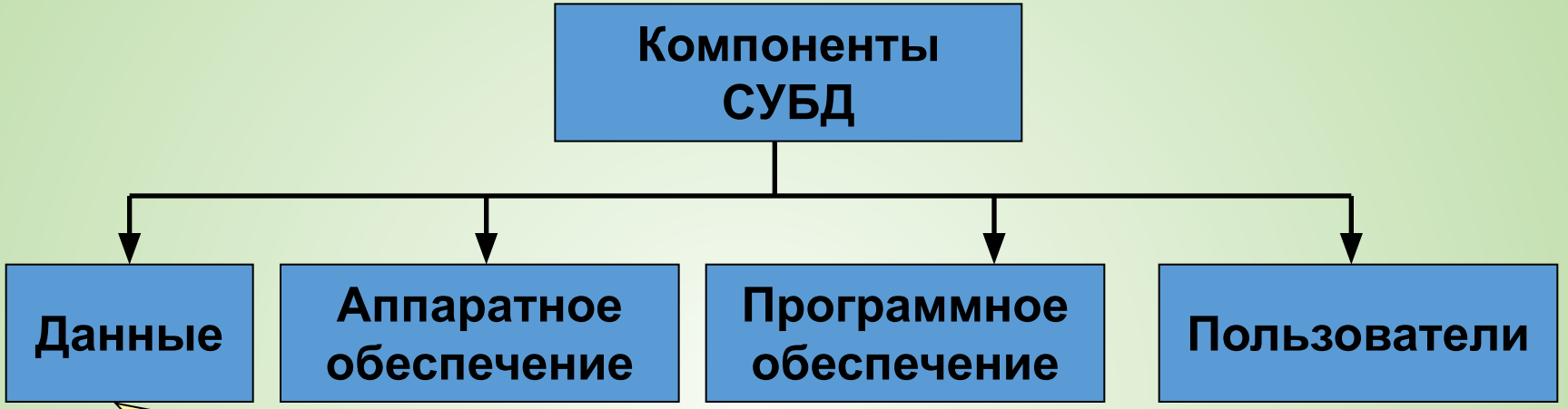
СУБД представляет собой оболочку, с помощью которой при организации структуры таблиц и заполнения их данными получается та или иная база данных.

В связи с этим в составе СУБД различают систему *программно-технических, организационных и "человеческих" составляющих*.

Программные средства включают систему управления, обеспечивающую ввод-вывод, обработку и хранение информации, создание, модификацию и тестирование БД, трансляторы.



Основные компоненты СУБД и их состав

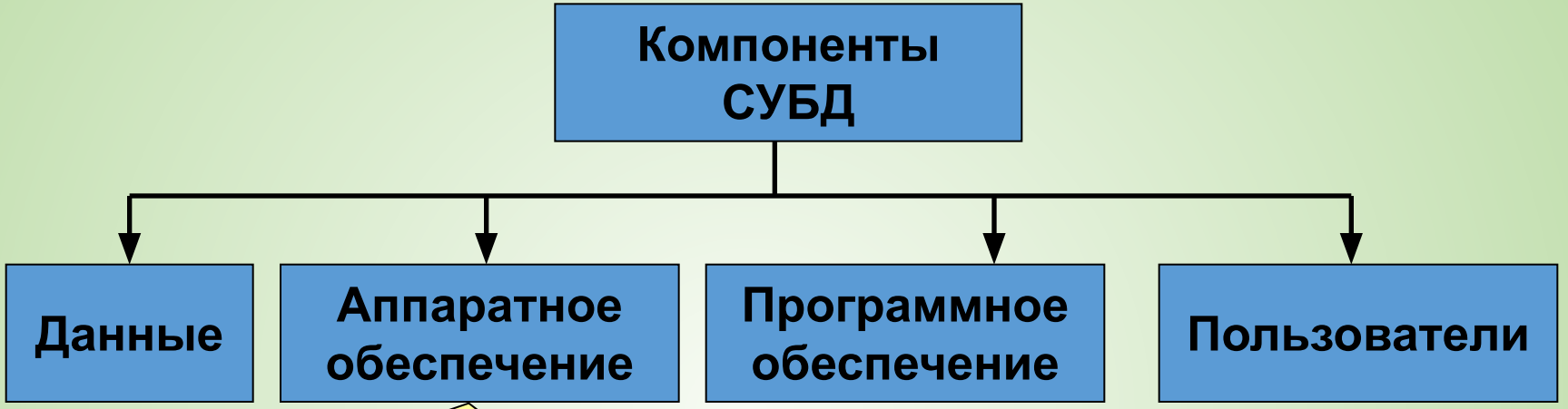


Данные должны быть интегрированными и общими.

Интегрирование – возможность представлять базу данных как объединение нескольких отдельных файлов данных полностью или частично не перекрывающихся.

Общие – возможность использования отдельных областей данных в БД несколькими различными пользователями, причем даже в одно и тоже время (одновременный доступ).

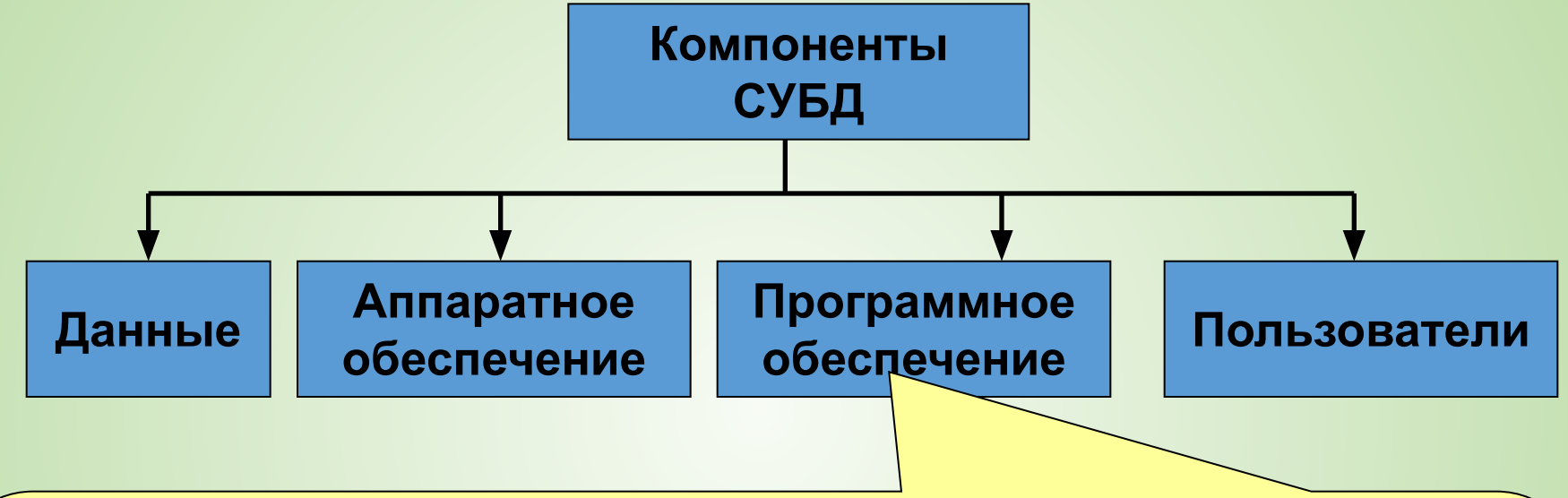
Основные компоненты СУБД и их состав



Накопители для хранения информации (обычно диски с перемещаемыми головками) вместе с подсоединенными устройствами ввода-вывода, контроллерами устройств, каналами ввода-вывода и т.д.

Процессор или процессоры вместе с основной памятью, которая используется для поддержки работы программного обеспечения системы

Основные компоненты СУБД и их состав

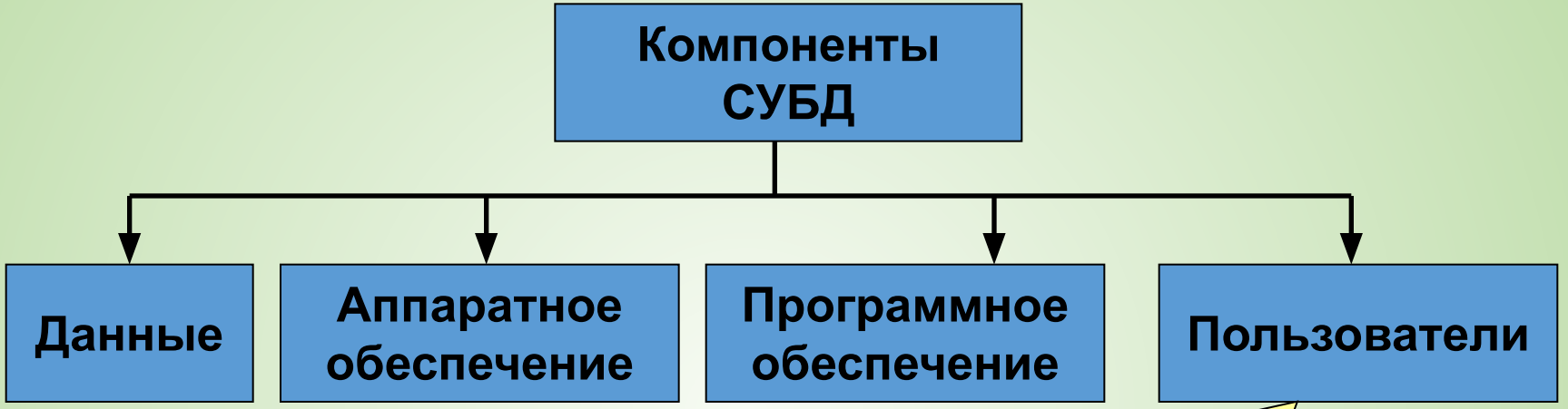


Диспетчер базы данных (database manager),или система управления базами данных СУБД (database management system(DBMS)).

СУБД предоставляет пользователю возможность рассматривать БД как объект более высокого уровня по сравнению с аппаратным обеспечением, а также поддерживает выражаемые в терминах высокого уровня пользовательские запросы(SQL).

Кроме СУБД, в программном обеспечении – утилиты, средства разработки приложений, средства проектирования, генераторы отчетов и другие.

Основные компоненты СУБД и их состав



Работающие с базами данных пользователи обладают различными знаниями, навыками и сталкиваются с решением различных задач:

- конечные пользователи;

- разработчики баз данных;

- разработчики приложений;

- администраторы баз данных.

Основные компоненты СУБД и их состав

Базовыми внутренними языками программирования являются языки четвертого поколения. В качестве базовых языков могут использоваться **C, C++, Pascal, Object Pascal.**

Исторически *для системы управления базой данных сложились* ***три языка***:

1. **Язык описания данных** (ЯОД), называемый также языком описания схем, - для построения структуры ("шапки") таблиц БД;
2. **Язык манипулирования данными** (ЯМД) - для заполнения БД данными и операций обновления (*запись, удаление, модификация*);
3. **Язык запросов** - *язык поиска наборов величин* в файле в соответствии с заданной совокупностью критериев поиска и *выдачи затребованных данных* без изменения содержимого файлов и БД (язык преобразования критериев в систему команд).

В настоящее время функции всех трех языков выполняет **язык SQL**, относящийся к классу языков, базирующихся на исчислении кортежей (кортеж - единица информации), языки СУБД FoxPro, Visual Basic for Application (СУБД Access) и т.д.

Структурные элементы БД

Поле – *элементарная единица логической организации данных*, которая соответствует неделимой единице информации – реквизиту. Для описания поля используются следующие ***характеристики***:

Имя, например, Фамилия, Имя, Отчество, Дата рождения;

Тип, например, символьный, числовой, календарный;

Длина, например, 15 байт, определяется максимально возможным количеством символов;

Точность для числовых данных, например два десятичных знака для отображения дробной части числа.

Запись – совокупность логически связанных полей. Экземпляр записи – отдельная реализация записи, содержащая конкретные значения ее полей.

Таблица (файл) – совокупность экземпляров записей одной структуры.

Классификация СУБД

В силу многогранности баз данных и СУБД (комплекса технических и программных средств, для хранения, поиска, защиты и использования данных) имеется множество классификационных признаков:

По степени универсальности (сфере применения) :

- СУБД общего назначения (СУБД ОН) и специализированные СУБД (СпСУБД).

По используемой модели данных

- иерархические, сетевые, реляционные; объектно-ориентированные СУБД.

По методам организации хранения и обработки данных :

- централизованные (локальные, файл – серверные, клиент-серверные) и распределённые СУБД.

По сфере применения

- справочные системы и системы обработки данных.

Классификация по масштабу систем:

- персональные; уровня группы, отдела, предприятия; корпоративные; географически распределенные.

Классификация СУБД

По степени универсальности различают два класса СУБД:

- системы общего назначения (СУБД ОН) ;

- специализированные системы (СпСУБД).

СУБД общего назначения не ориентированы на какую-либо предметную область или на информационные потребности какой-либо группы пользователей.

Каждая система такого рода реализуется как программный продукт, способный функционировать на некоторой модели ЭВМ в определенной операционной системе и поставляется многим пользователям как коммерческое изделие.

Такие СУБД обладают средствами настройки на работу с конкретной базой данных. Им присущи развитые функциональные возможности, функциональная избыточность.

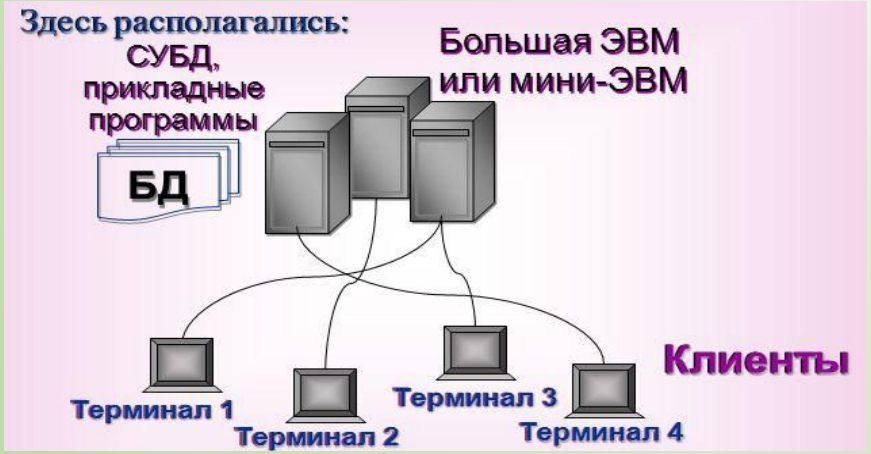
Специализированные СУБД создаются в редких случаях при невозможности или нецелесообразности использования СУБД общего назначения.

Классификация СУБД

По технологии обработки

Централизованные

(хранятся в памяти одной вычислительной системы, возможен распределенный доступ к данным)



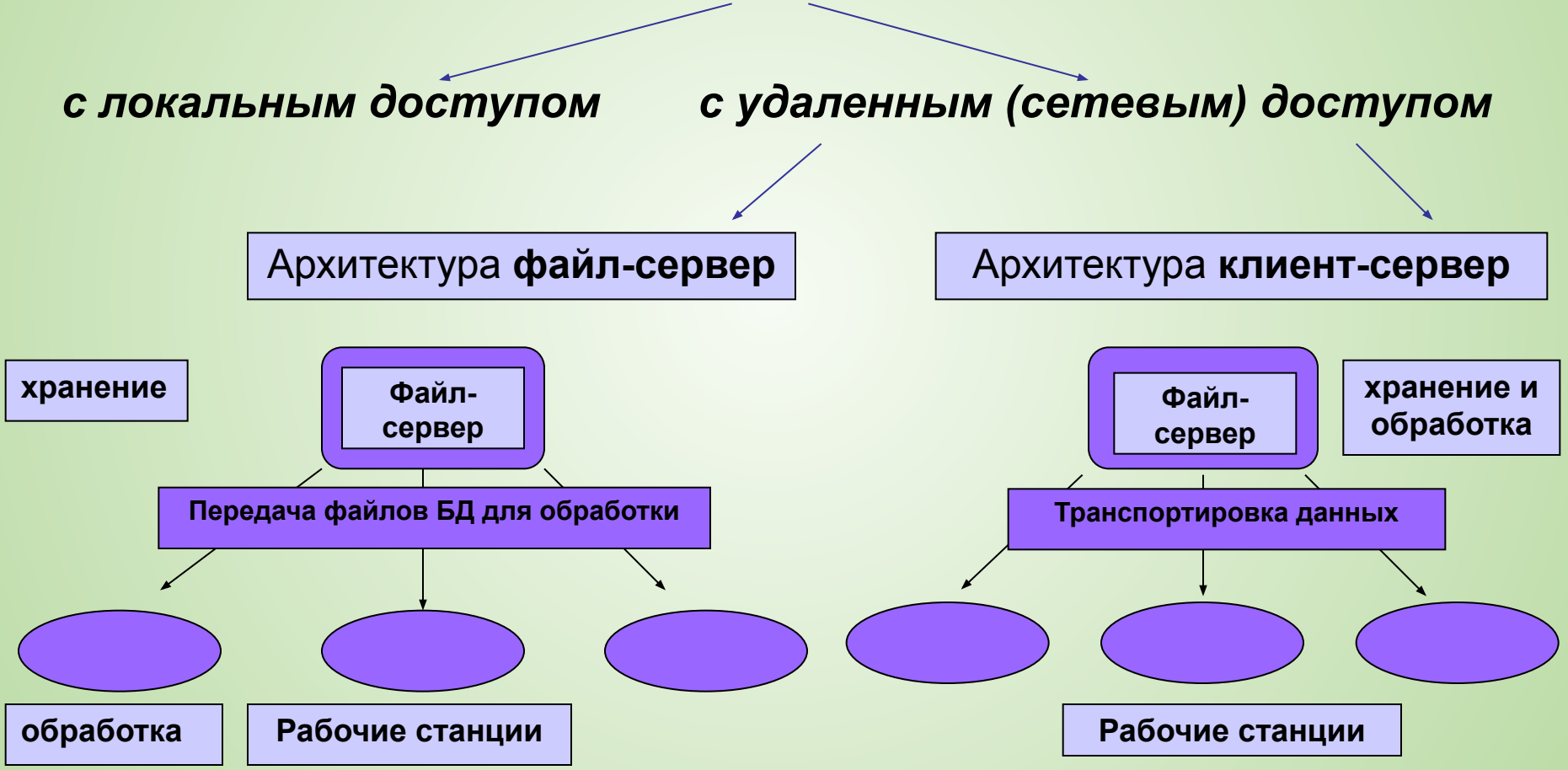
Распределенные

(состоит из нескольких, возможно пересекающихся или даже дублирующих друг друга частей, хранимых в различных ЭВМ вычислительной сети)

Классификация СУБД

По способу доступа к данным

с локальным доступом ***с удаленным (сетевым) доступом***



Классификация СУБД

Архитектура систем БД с сетевым доступом (**файл-сервер**) предполагает выделение одной из машин сети в качестве центральной (сервер файлов).

На ней хранится совместно используемая централизованная БД. Все другие машины сети являются рабочими станциями.

Файлы БД в соответствии с пользовательскими запросами передаются на рабочие станции, где и производится обработка.

При большой интенсивности доступа к одним и тем же данным производительность системы падает.

БД с сетевым доступом
(Файл-сервер)

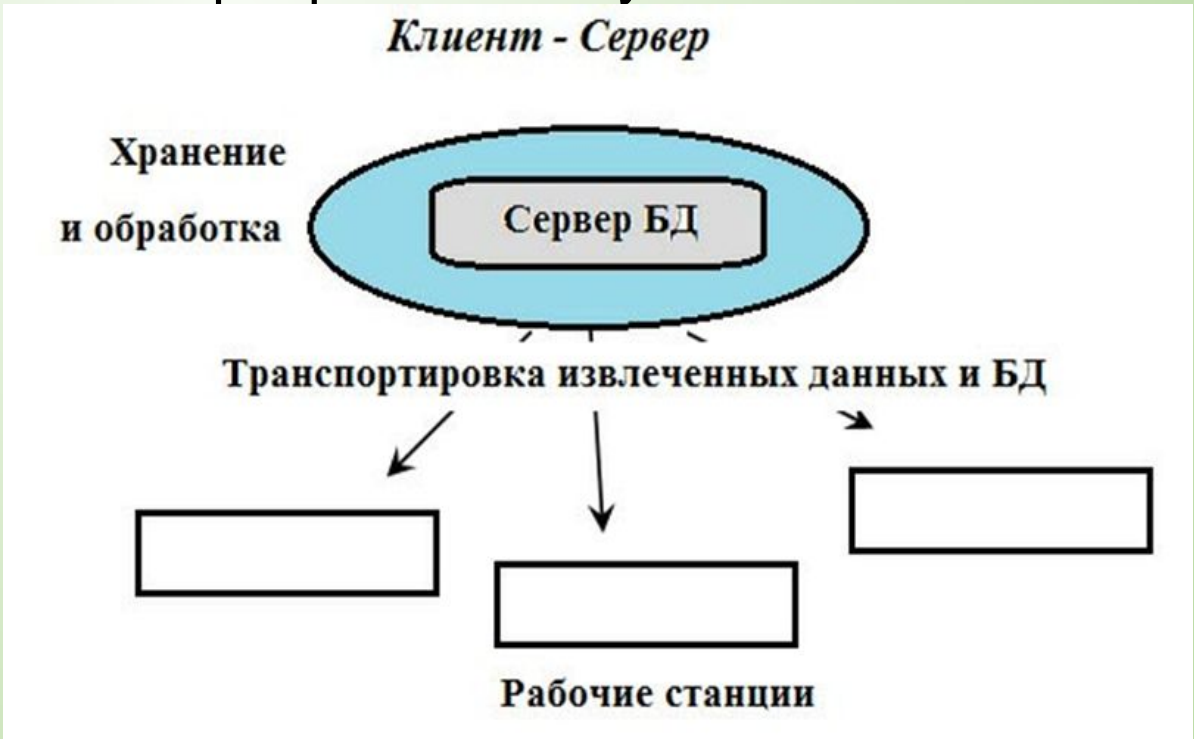
Классификация СУБД

В архитектуре ***Клиент-сервер*** - помимо хранения централизованной БД центральная машина (**сервер базы данных**) должна обеспечивать выполнение основного объёма обработки данных.

Запрос на данные клиента, порождает поиск и извлечение данных на сервере. Извлечённые данные (но не файлы) транспортируются по сети от сервера к клиенту.

БД с сетевым доступом

Клиент - сервер



Microsoft Office Access или просто **Microsoft Access** — реляционная система управления базами данных (СУБД) корпорации Microsoft. Входит в состав пакета Microsoft Office. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных. Благодаря встроенному языку VBA, в самом Access можно писать приложения, работающие с базами данных.

В СУБД Microsoft Access используется стандартный для операционных систем многооконный интерфейс, но в отличие от других приложений, не многодокументный. Единовременно может быть открыта только одна база данных, содержащая обязательное окно базы данных и окна для работы с объектами базы данных. В каждый момент времени одно из окон является активным и в нем курсором отмечается активный объект.

Окно базы данных — один из главных элементов интерфейса СУБД. Здесь систематизированы все объекты базы данных: таблицы, запросы, формы, отчеты.

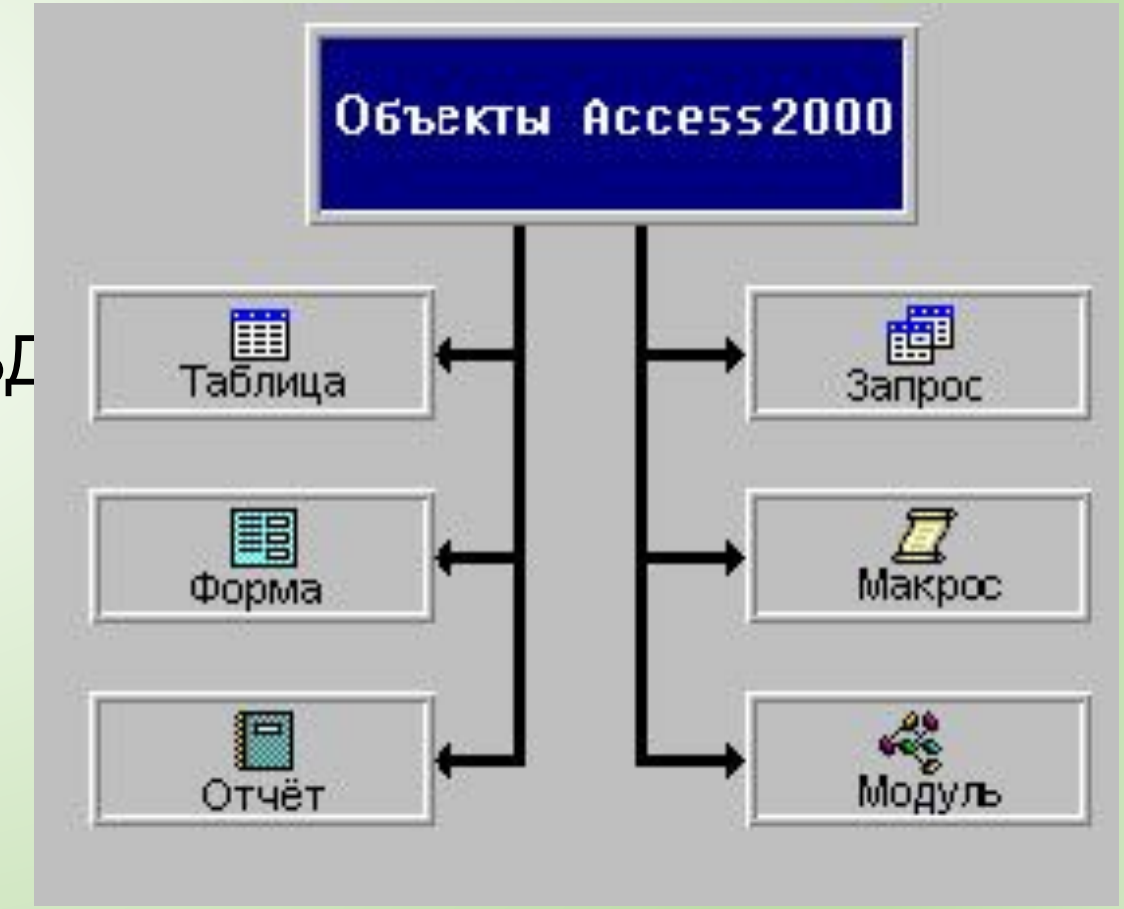
Объект СУБД-это БД

Основные типы объектов СУБД Access.

- Access предоставляет большой выбор способов хранения, отображения и предоставления данных. Компоненты, которые используются для хранения и представления данных, называются **ОБЪЕКТАМИ**.

В системе Access существуют следующие объекты:

1. Таблицы, запросы, схемы данных, непосредственно имеющие отношение к БД
2. Формы, отчеты, макросы и модули, называемые объектами приложения



Таблицы. Основным структурным компонентом базы данных является таблица.

Таблица – это объект, предназначенный для хранения данных в виде записей и полей.

- Каждая таблица включает информацию об объекте реального мира. Таблица состоит из заголовка и тела.
- Заголовок включает имена атрибутов объекта и их свойства.
- Тело содержит кортежи, каждый из которых представляет множество значений столбцов, в которых хранятся данные о конкретном экземпляре объекта.
- Для каждой таблицы можно определить первичный ключ, обеспечивающий уникальность каждой строки.
- При разработке структуры таблицы, прежде всего, необходимо определить названия полей, из которых она должна состоять, типы полей и их размеры.
- Каждому полю таблицы присваивается уникальное имя, которое не может содержать более 64 символов. Имя желательно делать таким, чтобы функция поля узнавалась по его имени.
- Далее надо решить, данные какого типа будут содержаться в каждом поле.
- В Access можно выбирать любые из основных типов данных. Один из этих типов данных должен быть присвоен каждому полю. Значение типа поля может быть задано только в режиме конструктора.

Типы данных Access и их описание.

Тип данных	Описание
Текстовый (Значение по умолчанию)	Текст или числа, не требующие проведения расчетов (до 255 знаков)
Числовой	Числовые данные различных форматов, используемые для проведения расчетов
Дата/время	Для хранения информации о дате и времени с 100 по 9999 год включительно
Денежный	Денежные значения и числовые данные, используемые в математических расчетах, проводящихся с точностью до 15 знаков в целой и до 4 знаков в дробной части
Поле MEMO	Для хранения комментариев (до 65535 символов)
Счетчик	Специальное числовое поле, в котором Access автоматически присваивает уникальный порядковый номер каждой записи. Значения полей типа счетчика обновлять нельзя
Логический	Может иметь только одно из двух возможных значений (True/False, Да/Нет)
Поле объекта OLE	Объект (например, электронная таблица Microsoft Excel, документ Microsoft Word, рисунок, звукозапись или другие данные в двоичном формате), связанный или внедренный в таблицу Access
Гиперссылка	Строка, состоящая из букв и цифр и представляющая адрес гиперссылки. Адрес гиперссылки может состоять максимум из трех частей: текст, выводимый в поле или в элементе управления, путь к файлу (в формате пути UNC) или к странице (адрес URL). Чтобы вставить адрес гиперссылки в поле или в элемент управления, выполните команду Вставка, Гиперссылка
Мастер подстановок	Создает поле, в котором предлагается выбор значений из списка или из поля со списком, содержащего набор постоянных значений или значений из другой таблицы. Это в действительности не тип поля, а способ хранения поля

Запросы.

Запросы являются мощным средством обработки данных, хранимых в таблицах Access. С помощью запросов можно просматривать, анализировать и изменять данные из нескольких таблиц. Они также используются в качестве источника данных для форм и отчетов. Запросы позволяют вычислять итоговые значения и выводить их в компактном формате, подобном формату электронной таблицы, а также выполнять вычисления над группами записей.

Запросы можно создавать самостоятельно и с помощью мастеров. Мастера запросов автоматически выполняют основные действия в зависимости от ответов пользователя на поставленные вопросы. Самостоятельно разработать запросы можно в режиме конструктора.

В Access можно создавать следующие типы запросов:

- запрос на выборку;
- запрос с параметрами;
- перекрестный запрос;
- запрос на изменение (запрос на удаление, обновление и добавление записей на создание таблицы);
- запросы SQL (запросы на объединение, запросы к серверу, управляющие запросы, подчиненные запросы)

Формы.

- Формы обеспечивают наиболее гибкий способ ввода, редактирования, просмотра и удаления данных и фактически являются шаблонами, управляющими отображением информации. Форма позволяет отображать одновременно все поля одной или нескольких записей. Оптимально построенная форма может вмещать несколько десятков полей на одном экране, а если полей намного больше, то для каждой записи можно создать многостраничную форму. Можно создать форму-меню для вызова других форм, таблиц, запросов или отчетов. В форме каждое поле можно разместить в точно заданном месте, выбрать для него цвет или заливку и добавить элементы управления текстом для эффективного ввода данных.

Отчеты.

- Отчет – это гибкое и эффективное средство для организации просмотра и распечатки итоговой информации. В отчете можно получить результаты сложных расчетов, статистических сравнений, а также поместить в него рисунки и диаграммы.
- Пользователь имеет возможность разработать отчет самостоятельно или создать отчет с помощью мастера. Мастер по разработке отчетов выполняет всю рутинную работу и позволяет быстро разработать отчет. После вызова Мастера выводятся диалоговые окна с приглашением ввести необходимые данные, и отчет создается на основании ответов пользователя. После этого можно переключиться в режим конструктора и внести изменения в стандартный макет.

Макросы и модули.

Макрос – это объект, представляющий собой структурированное описание одного или нескольких действий, которые могут выполняться в ответ на определенное событие. Например, можно определить макрос, который в ответ на выбор некоторого элемента в основной форме открывает другую форму. Можно из одного макроса запустить другой макрос и функцию модуля.

Модуль – это объект, содержащий программы на Microsoft Access Visual Basic, которые могут разрабатываться пользователем для реализации нестандартных процедур при создании приложения.

Реляционная алгебра.
Теоретико-множественные операторы

Язык SQL представляет собой смесь операторов реляционной алгебры и выражений реляционного исчисления, использующий синтаксис, близкий к фразам английского языка и расширенный дополнительными возможностями, отсутствующими в реляционной алгебре и реляционном исчислении.

Вообще, язык доступа к данным называется ***реляционно-полным***, если он по выразительной силе не уступает реляционной алгебре, т.е. любой оператор реляционной алгебры может быть выражен средствами этого языка. Именно таким и является язык SQL.

Традиционно определяют восемь реляционных операторов, объединенных в две группы.

Теоретико-множественные операторы:

- ***Объединение***
- ***Пересечение***
- ***Вычитание***
- ***Декартово произведение***

Специальные реляционные операторы:

- ***Выборка***
- ***Проекция***
- ***Соединение***
- ***Деление***

Не все они являются независимыми, т.е. некоторые из этих операторов могут быть выражены через другие реляционные операторы.

Отношения, совместимые по типу

Определение. Будем называть отношения ***совместимыми по типу***, если они имеют идентичные заголовки, а именно:

- отношения имеют *одно и то же множество имен атрибутов*, т.е. для любого атрибута в одном отношении найдется атрибут с таким же наименованием в другом отношении;
- атрибуты с одинаковыми именами *определены на одних и тех же доменах* (или типах, если домены не поддерживаются).

Примечание: Некоторые отношения не являются совместимыми по типу, но после переименования атрибутов могут ими стать, для этого можно использовать вспомогательный *оператор переименования атрибутов.*

Теоретико-множественные операторы

Объединением двух совместимых по типу отношений *A* и *B* называется отношение с тем же заголовком, что и у отношений *A* и *B*, и телом, состоящим из совокупности кортежей обоих отношений.

Синтаксис операции объединения: A UNION B (A ∪ *B)*

Замечание. Объединение, как и любое отношение, не может содержать одинаковых кортежей. Поэтому, если некоторый кортеж входит и в отношение *A*, и отношение *B*, то в объединение он входит один раз.

Пусть даны два отношения *A* (таблица 6.1) и *B* (таблица 6.2) с информацией о сотрудниках:

Таблица 6.1 - Отношение A

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Петров	2000
3	Сидоров	3000

Таблица 6.2. Отношение B

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Пушников	2500
4	Сидоров	3000

В результате операции объединения *(A B)* будет получено отношение *C* с тем же заголовком что и у отношений *A* и *B* (таблица 6.3):
Отношение *C*, не наследует первичного ключа. Поэтому, в объединении отношений *A* и *B* атрибут «Табельный номер» может содержать дубликаты значений. Наследование ключей противоречило бы понятию объединения как 26

«объединению множеств».
Конечно, объединение отношений *A* и *B* имеет, как и любое отношение, потенциальный ключ, например, состоящий из всех атрибутов.

Таблица 6.3 - Результирующее отношение $(A \cup B)$

Табельный номер	Фамилия	Зарплата
1	Иванов	1000
2	Петров	2000
3	Сидоров	3000
2	Пушников	2500
4	Сидоров	3000

Никакие реляционные операторы не передают результирующему отношению никаких данных о потенциальных ключах. Причина заключается в том, что потенциальный ключ - семантическое понятие, отражающее различимость объектов предметной области.

Наличие потенциальных ключей *не выводится* из структуры отношения, а явно задается для каждого отношения, исходя из его смысла.

Реляционные же операторы являются *формальными* операциями над отношениями и выполняются одинаково, независимо от смысла данных, содержащихся в отношениях.

Поэтому, реляционные операторы ничего не могут «знать» о смысле данных. Трактовка результата реляционных операций - дело пользователя.

Пересечением двух совместимых по типу отношений *A* и *B* называется отношение с тем же заголовком, что и у отношения *A* и *B*, и телом, состоящим из кортежей, принадлежащих одновременно обоим отношениям.

Синтаксис операции пересечения: A INTERSECT B (A □ B)

Для исходных отношений (таблицы 6.1 и 6.2) пересечение примет вид (таблица 6.4):

Таблица 6.4 - Результирующее отношение *(A ∩ B)*

Табельный номер	Фамилия	Зарплата
1	Иванов	1000

Вычитанием двух совместимых по типу отношений *A* и *B* называется отношение с тем же заголовком, что и у отношений *A* и *B*, и телом, состоящим из кортежей, принадлежащих отношению *A* и не принадлежащих отношению *B*.

Синтаксис операции вычитания: A MINUS B (A \ B)

Для исходных отношений (таблицы 6.1 и 6.2) результат вычитания примет вид (таблица 6.5):

Таблица 6.5 - Результирующее отношение *(A \ B)*

Табельный номер	Фамилия	Зарплата
2	Петров	2000
3	Сидоров	3000

Декартовым произведением двух отношений *A* и *B* называется отношение *C* полученное сцеплением их заголовков и кортежей соответствующих отношений, причем каждому кортежу отношения *A* должны быть противопоставлены все кортежи отношения *B*

Синтаксис операции декартового произведения: A TIMES B (A □ B)

Пусть даны два отношения с информацией о поставщиках *A* и деталях *B*.

Пример. Пусть даны два отношения *A* и *B* с информацией о поставщиках и деталях (таблицы 6.6 и 6.7). Тогда декартово произведение отношений *A* и *B* примет вид указанный в таблице 6.8.

Таблица 6.6 - Отношение *A* (Поставщики)

Номер поставщика	**Наименование поставщика**
1	Иванов
2	Петров
3	Сидоров

Таблица 6.7 - Отношение *B* (Детали)

Номер детали	**Наименование детали**
1	Болт
2	Гайка
3	Винт

Таблица 6.8 - Результирующие отношение *(A ⊗ B)*

Номер поставщика	**Наименование поставщика**	**Номер детали**	**Наименование детали**
1	Иванов	1	Болт
1	Иванов	2	Гайка
1	Иванов	3	Винт
2	Петров	1	Болт
2	Петров	2	Гайка
2	Петров	3	Винт
3	Сидоров	1	Болт
3	Сидоров	2	Гайка
3	Сидоров	3	Винт

Замечания:

1. Мощность произведения $A \times B$ равна произведению мощностей отношений A и B, т.к. каждый кортеж отношения A соединяется с каждым кортежем отношения B.
2. Если в отношениях A и B имеются атрибуты с одинаковыми именами, то перед выполнением операции такие атрибуты необходимо переименовать.
3. Перемножать можно любые два отношения, совместимость по типу при этом не требуется.
4. Декартово произведение не дает никакой новой информации, по сравнению с предыдущими операциями, однако она важна для выполнения специальных реляционных операций.

Задания для самостоятельной работы

Задание 1. Даны два отношения А (таблица 6.9) и В (таблица 6.10) содержащие данные о товарах, необходимо выполнить операции объединения, пересечения и вычитания. Попытайтесь определить смысл результирующих отношений.

Таблица 6.9 – Отношение А

Код	Наименование	Единица	Цена единицы
4640	Плитка «Неаполь»	шт	25.50
4778	Плитка «Экстра»	шт	14.00
4788	Клей «Монолит»	шт	100.00
4796	Гипс	кг	10.00
4899	Шпатель 201	шт	21.50

Таблица 6.10 – Отношение В

Код	Наименование	Единица	Цена единицы
4640	Плитка «Неаполь»	шт	25.50
4779	Плитка «Лазурь»	шт	14.50
4780	Клей «ПВА»	кг	45.00
4788	Клей «Монолит»	шт	100.00
4903	Шпатель 201	шт	21.50

Задание 2. Выполните декартово произведение двух отношений приведенных на рисунке 5.2

Реляционная алгебра. Специальные реляционные операторы

С практической точки зрения, специальные реляционные операции имеют большее практическое значение по сравнению с теоретико-множественными.

Выборкой (ограничением, селекцией или фильтрацией) на отношении *A,* с условием *C* называется отношение с тем же заголовком, что и у отношения *A*, и телом, состоящем из кортежей, значения атрибутов которых при подстановке в условие *C* дают значение ИСТИНА. *C* - логическое выражение, в которое могут входить атрибуты отношения *A* и (или) скалярные выражения.

В простейшем случае условие С имеет вид *X*□*Y*, где □ - один из операторов сравнения (□, □, □, □, □, □ и т.д.), а *X* и *Y* – атрибуты отношения А или скалярные значения. Такие выборки называются □ - *выборки (тэта-выборки)* или □ - *селекция*, □ - *ограничения*.

Синтаксис операции выборки: *A WHERE C*, где *C* – условие выборки, или *X*□*Y.*

Пусть дано отношение *A* с информацией о сотрудниках (таблица 6.1), необходимо выбрать всех сотрудников с зарплатой менее 3000, в этом случае выполняем выборку *A WHERE Зарплата* < 3000, результат выборки в таблице 7.1:

Таблица 7.1 - Результат операции *A WHERE Зарплата*<3000

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Петров	2000

Смысл операции выборки очевиден - выбрать кортежи отношения, удовлетворяющие некоторому условию. Таким образом, операция выборки дает «*горизонтальный срез*» отношения по некоторому условию.

Проекцией отношения *A* по атрибутам *(X,Y,…,Z)*, где каждый из атрибутов принадлежит отношению *A*, называется отношение с заголовком *(X,Y,…,Z)* и телом, содержащим кортежи соответствующих атрибутов

Синтаксис проекции: A[X,Y,…,Z]

Для отношения *A* (таблица 6.1) результатом проекции *A* [Фамилия, Зарплата] таблица 7.2:

Таблица 7.2 - Результат операции *A [Фамилия, Зарплата]*

Фамилия	**Зарплата**
Иванов	1000
Петров	2000
Сидоров	3000

Видно, что операция проекции выполняет «*вертикальный срез*» отношения, в котором будут удалены все возникшие при таком срезе дубликаты кортежей.

Соединение. Операция соединения отношений, наряду с операциями выборки и проекции, является одной из наиболее важных реляционных операций.

Обычно рассматривается несколько разновидностей операции соединения:

- общая операция соединения;
- -соединение (тэта-соединение);
- экви-соединение;
- естественное соединение.

Наиболее важным из этих частных случаев является операция естественного соединения. Все разновидности соединения являются частными случаями общей операции соединения.

Соединением отношений *A* и *B* по условию *C* называется отношение образованное последовательностью операций декартова произведения и выборки:

(A B) WHERE C,

где *C* представляет собой логическое выражение, в которое могут входить атрибуты отношений *A* и *B* и (или) скалярные выражения.

Если в отношениях *A* и *B* имеются атрибуты с одинаковыми наименованиями, то перед выполнением соединения такие атрибуты необходимо переименовать.

Тэта – соединение. Пусть отношение *A* содержит атрибут *X*, отношение *B* содержит атрибут *Y*, а □ - один из операторов сравнения (□, □, □, □, □, □ и т.д.). Тогда □-***соединением*** отношения *A* по атрибуту *X* с отношением *B* по атрибуту *Y* называют отношение *(A □ B) WHERE X□Y*

Это частный случай операции общего соединения. Иногда, для операции соединения применяют более короткий синтаксис *A[X□Y]B*.

Экви-соединение является наиболее важным частным случаем тэта-соединения, когда тэта является просто равенством и имеет следующий синтаксис: *A[X=Y]B* или *(A □ B) WHERE X=Y.*

Пусть даны два отношения *A* и *B*. Отношение *A* (таблица 7.3) - данные о товарах, отношение *B* (таблица 7.4) - данные о продаже товаров. Необходимо определить, когда и в каком количестве отпускались товары со склада.

Таблица 7.3 - Отношение А, «Товары»

Код товара	Товар	Единица	Цена единицы
1	Сахар	кг	16р.
2	Макароны	кг	14р.

Таблица 7.4 - Отношение B, «Отпуск товаров»

Код тов.	Дата продажи	Количество
1	12.07.02	10
2	12.07.02	15
2	12.07.02	3

Таблица 7.5 - Соединение *(A ⊗ B) WHERE A.Код товара = B.Код тов.*

Код товара	Товар	Единица	Цена единицы	Код тов.	Дата продажи	Количест-во
1	Сахар	кг	16р.	***1***	12.07.02	10
1	Сахар	кг	16р.	***2***	12.07.02	15
1	Сахар	кг	16р.	***2***	12.07.02	3
2	Макароны	кг	16р.	***1***	12.07.02	10
2	Макароны	кг	16р.	***2***	12.07.02	15
2	Макароны	кг	16р.	***2***	12.07.02	3

Таблица 7.5 представляет собой декартово произведение двух отношений, в котором темным выделены кортежи, для которых не выполнится условие выборки *А.Код товара = В.Код тов.,* следовательно, они будут вычеркнуты из окончательного результата (таблица 7.6).

Таблица 7.6 – Окончательный результат соединения *(А □ В) WHERE А.Код товара = В.Код тов.*

Таблица 7.6 – Окончательный результат соединения *(А ⊗ В) WHERE товара = В.Код тов.*

Код товара	**Товар**	**Единица**	**Цена единицы**	**Код тов.**	**Дата продажи**	**Количест-во**
1	Сахар	кг	16р.	***1***	12.07.02	10
2	Макароны	кг	16р.	***2***	12.07.02	15
2	Макароны	кг	16р.	***2***	12.07.02	3

Естественное соединение

Пусть даны отношения $A(A_1, A_2, \ldots, A_n, X_1, X_2, \ldots, X_p)$ и $B(X_1, X_2, \ldots, X_p, B_1, B_2, \ldots, B_m)$, имеющие одинаковые атрибуты $X_1, X_2, \ldots, X_p$ (т.е. атрибуты с одинаковыми именами и определенные на одинаковых доменах).

Тогда ***естественным соединением*** отношений A и B называется отношение с заголовком $(A_1, A_2, \ldots, A_n, X_1, X_2, \ldots, X_p, B_1, B_2, \ldots, B_m)$, и телом, содержащим множество соответствующих кортежей.

Естественное соединение настолько важно, что для него используют специальный синтаксис: A JOIN B.

Замечания:

В синтаксисе естественного соединения не указываются, по каким атрибутам производится соединение. Естественное соединение производится *по всем* одинаковым атрибутам.

Естественное соединение эквивалентно следующей последовательности реляционных операций:

1. Переименовать одинаковые атрибуты в отношениях
2. Выполнить декартово произведение отношений
3. Выполнить выборку по совпадающим значениям атрибутов, имевших одинаковые имена
4. Выполнить проекцию, удалив повторяющиеся атрибуты
5. Переименовать атрибуты, вернув им первоначальные имена

Можно выполнять последовательное естественное соединение нескольких отношений. Естественное соединение (как и соединение общего вида) обладает свойством *ассоциативности*, т.е. *(A JOIN B) JOIN C = A JOIN (B JOIN C),* поэтому его можно записать, опуская скобки *A JOIN B JOIN C.*

Применяя естественное соединение, результат, полученный в таблице 7.6, можно было получить операцией *A JOIN B*, но с одним условием, атрибут отношения B используемый для связи с отношением *A* должен иметь имя совпадающее с атрибутом связи отношения *A* (т.е. Код товара).

Деление. Пусть даны отношения $A(X_1,X_2,\ldots,X_n,Y_1,Y_2,\ldots,Y_m)$ и $B(Y_1,Y_2,\ldots,Y_m)$, причем атрибуты $(Y_1,Y_2,\ldots,Y_m)$ - общие для двух отношений. ***Делением отношений*** *A* на *B* называется отношение с заголовком $(X_1,X_2,\ldots,X_n)$ и телом, содержащим множество кортежей $(x_1,x_2,\ldots,x_n)$, только таких, для которых найдутся *все* кортежи $(y_1,y_2,\ldots,y_m)\square B$, в отношении *A*.

Синтаксис операции деления: A DEVIDBY B (A : B)

Замечание. Типичные запросы, реализуемые с помощью операции деления, обычно в своей формулировке имеют слово «все» - «какие поставщики поставляют *все* детали?».

Таблицы 6.6 и 6.7 нуждаются в логическом дополнении, т.е. нужна таблица, связывающая поставляемые товары и поставщиков (по их кодам). Введем такую таблицу и на ее примере рассмотрим операцию деления.

Таблица 7.7 - Отношение X «Поставщики-Детали»

Номер поставщика	Номер детали
1	1
1	2
1	3
2	1
2	2
3	1

Требуется узнать, какой поставщик поставляет *все* детали. Отношение X возьмем в качестве делимого, а проекцию таблицы 7 «детали» - Y=B[Номер детали] (таблица 7.8):

Таблица 7.8 - Отношение Y = B[Номер детали]

Номер детали
1
2
3

Деление *X DEVIDBY Y* дает список номеров поставщиков, поставляющих *все* детали (таблица 7.9):

Таблица 7.9 - Результирующее отношение *(A : B)*

Номер поставщика
1

Задания для самостоятельной работы

Даны отношения, моделирующие работу банка (таблица 7.10) и его филиалов (таблица 7.11).
Клиент может иметь несколько счетов, при этом они могут быть размещены как в одном, так и в разных филиалах банка. В отношении R1 (таблица 7.10) содержится информация обо всех клиентах и их счетах в филиалах нашего банка. Каждый клиент, в соответствии со своим счетом, может рассчитывать на некоторый кредит от нашего банка, сумма допустимого кредита также зафиксирована.

Таблица 7.10 – Отношение R1

ФИО клиента	№ филиала	№ счета	Остаток	Кредит

Таблица 7.11 – Отношение R2

№ филиала	Район

С использованием языка реляционной алгебры составить запросы, позволяющие выбрать:

1. Филиалы, клиенты которых имеют счета с остатком, превышающим $1000.
2. Клиентов, которые имеют счета во всех филиалах данного банка.
3. Клиентов, которые имеют только по одному счету в разных филиалах банка. То есть, в общем, у этих клиентов может быть несколько счетов, но в одном филиале не более одного счета.
4. Клиенты, которые имеют счета в нескольких филиалах банка расположенных только в одном районе.
5. Филиалы, которые не имеют ни одного клиента.
6. Филиалы, которые имеют клиентов с остатком на счету 0 (ноль).
7. Филиалы, у которых есть клиенты с кредитом, превышающим остаток на счету в 2 раза.

Понятие избыточных данных. Нормализация отношений. I, II, III нормальные формы Понятие физической и логической целостности. Приведение отношений к соответствующим нормальным формам. Нормализация баз данных

Спасибо за внимание!