

# Базы данных

- 1. Основные понятия**
- 2. Классификация баз данных**
- 3. Структурные элементы базы данных**
- 4. Виды моделей данных**

# Основные понятия

Создавая базу данных, пользователь стремится упорядочить информацию по различным признакам и быстро извлекать выборку с произвольным сочетанием признаков. Сделать это возможно, только если данные структурированы.

**Структурирование** - это введение соглашений о способах представления данных.



- Неструктурированными называют данные, записанные, например, в текстовом файле.
- **Пример 1.** На рис. 1 пример неструктурированных данных, содержащих сведения о студентах (номер личного дела, фамилию, имя, отчество и год рождения) Легко убедиться, что сложно организовать поиск необходимых данных, хранящихся в неструктурированном виде, а упорядочить подобную информацию практически не представляется реальным.

Личное дело № 16493, Сергеев Петр Михайлович, дата рождения 1 января 1987г.; Л/д № 16593, Петрова Анна Владимировна, дата рожд. 15 марта 1985г.; № личн. дела 16693, д. р. 14.04.86, Анохин Андрей Борисович.

Рис.1. Пример неструктурированных данных

- Чтобы автоматизировать поиск и систематизировать эти данные, необходимо выработать определенные соглашения о способах представления данных, т.е. дату рождения нужно записывать одинаково для каждого студента, она должна иметь одинаковую длину и определенное место среди остальной информации Эти же замечания справедливы и для остальных данных (номер личною дела, фамилия, имя, отчество).

Пример.2. После проведения несложной структуризации с информацией, указанной в примере (рис.1), она будет выглядеть так, как это показано на рис.2.

№ Личного дела	Фамилия	Имя	Отчество	Дата рождения
16493	Сергеев	Петр	Михайлович	01.01.87
16593	Петрова	Анна	Владимировна	15.03.85
16693	Анохин	Андрей	Борисович	14.04.86

Рис.2. Пример структурированных данных

В современной технологии баз данных предполагается, что создание базы данных, ее поддержка и обеспечение доступа пользователей к ней осуществляются централизованно с помощью специального программного инструментария — *системы управления базами данных*.

**База данных (БД)** — это поименованная совокупность структурированных данных, относящихся к определенной предметной области. **Система управления базами данных (СУБД)** — это комплекс программных и языковых средств, необходимых для создания баз данных, поддержания их в актуальном состоянии и организации поиска в них необходимой информации.

# Классификация баз данных

По *технологии обработки* данных базы данных подразделяются на централизованные и распределенные.

*Централизованная база* данных хранится в памяти одной вычислительной системы. Если эта вычислительная система является компонентом сети ЭВМ, возможен распределенный доступ к такой базе. Такой способ использования баз данных часто применяют в локальных сетях ПК.



- *Распределенная база* данных состоит из нескольких, возможно пересекающихся или даже дублирующих друг друга частей, хранимых в различных ЭВМ вычислительной сети. Работа с такой базой осуществляется с помощью системы управления распределенной базой данных (СУРБД).
- По *способу доступа* к данным базы данных разделяются на базы данных с *локальным доступом* и базы данных с *удаленным (сетевым) доступом*.

Системы централизованных баз данных с сетевым доступом предполагают различные архитектуры подобных систем:

- файл-сервер;
- клиент-сервер.

**Файл-сервер.** Архитектура систем БД с сетевым доступом предполагает выделение одной из машин сети в качестве центральной (сервер файлов). На такой машине хранится совместно используемая централизованная БД. Все другие машины сети выполняют функции рабочих станций. Файлы базы данных в соответствии с пользовательскими запросами передаются на рабочие станции, где и производится обработка.

**Клиент-сервер.** В этой концепции подразумевается, что помимо хранения централизованной базы данных центральная машина (сервер базы данных) должна обеспечивать выполнение основного объема обработки данных. Запрос на данные, выдаваемый клиентом (рабочей станцией), порождает поиск и извлечение данных на сервере. Извлеченные данные (но не файлы) транспортируются по сети от сервера к клиенту. Спецификой архитектуры клиент-сервер является использование языка запросов SQL.

Рис.3 Схема обработки информации в БД по принципу файл-сервер



# Схема обработки информации в БД по принципу клиент-сервер

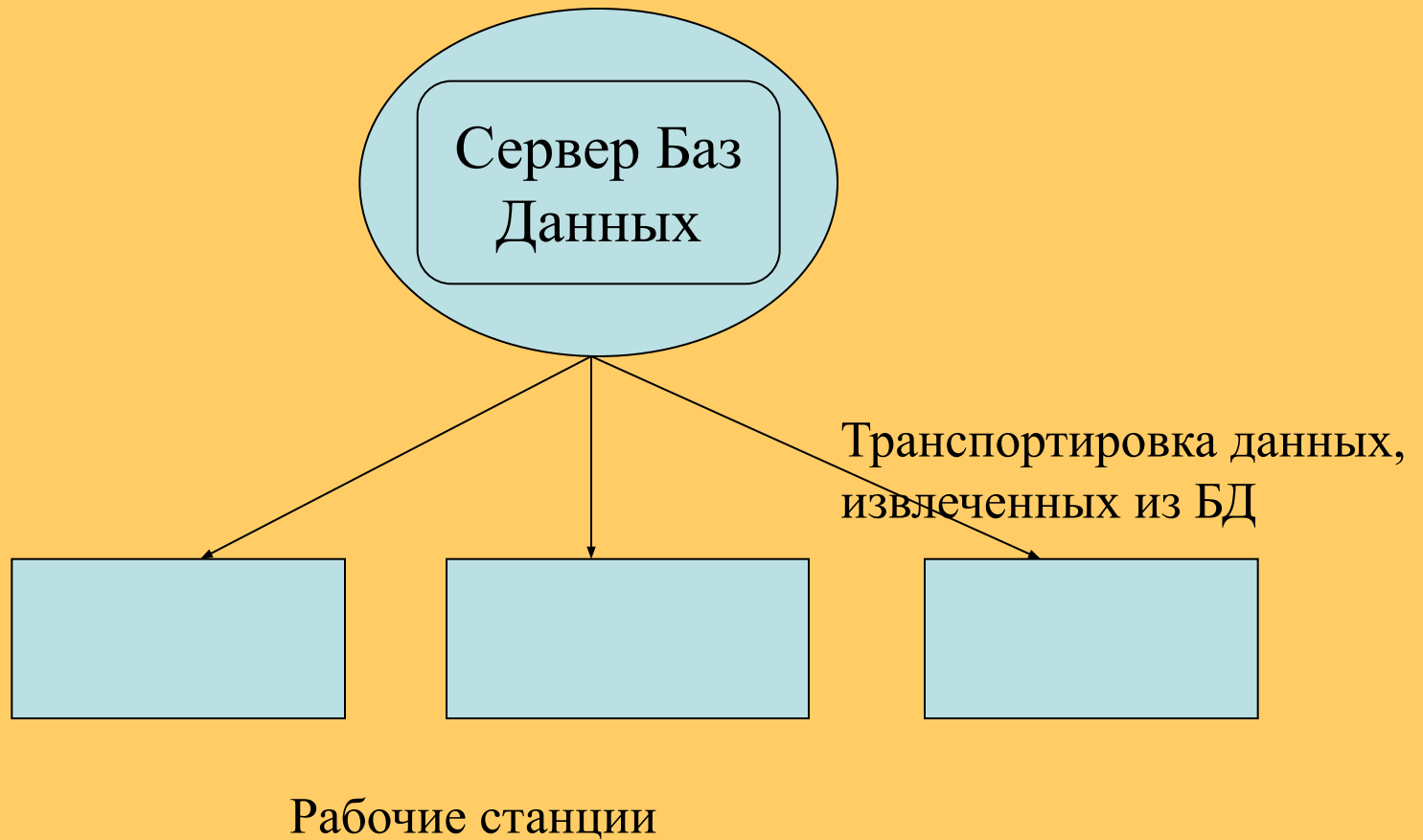


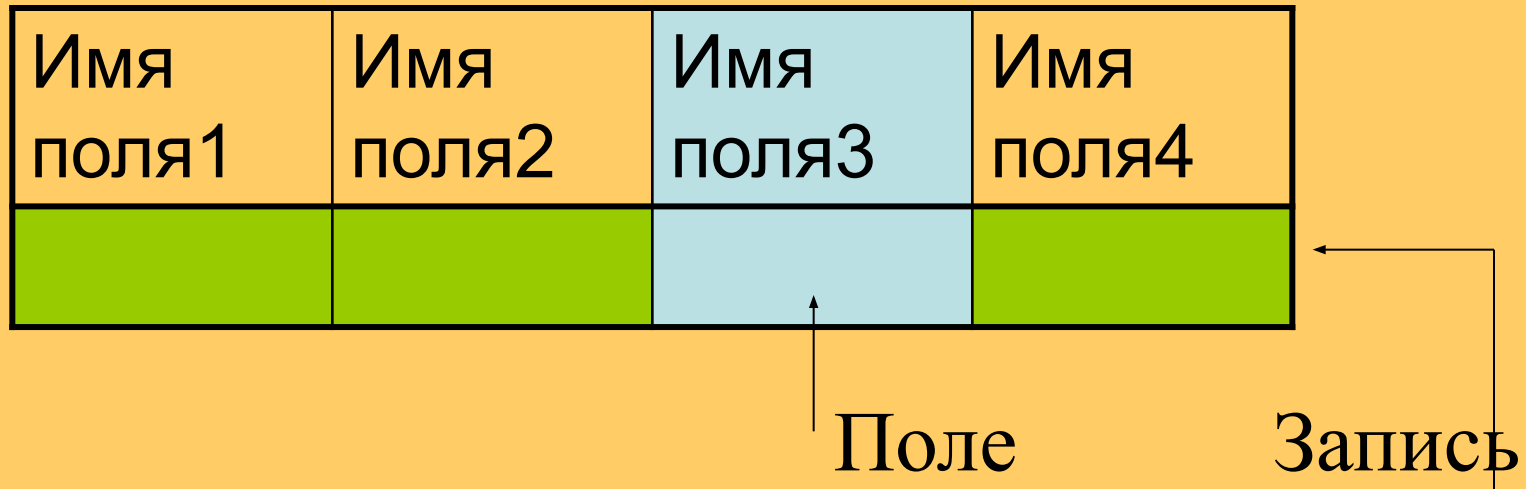
Рис.4

# Структурные элементы базы данных

**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, календар-ный;
- **длина**, например, 15 байт, причем будет определяться максимально возможным количеством символов;
- **точность** для числовых данных, например два десятичных знака для отображения дробной части числа.

- Рис.5. Основные структурные элементы БД



**Запись** — совокупность логически связанных полей. Экземпляр записи - отдельная реализация записи, содержащая конкретные значения ее полей.

**Файл (таблица)** — совокупность экземпляров записей одной структуры.

Имя файла					
Поле		Признак ключа	Формат поля		
Имя (обозначение)	Полное наименование		Тип	Длина	Точность (для чисел)
Имя1					
....					
имя n					

Рис. 6. Описание логической структуры записи файла



- В структуре записи файла указываются поля, значения которых являются **ключами**:
- **первичных (ПК)**, которые идентифицируют экземпляр записи, и **вторичными (ВК)**, которые выполняют роль поисковых или группировочных признаков (по значению вторичного ключа можно найти несколько записей).
- **Пример 3.** На рис. 7 приведен пример описания логической структуры записи файла (таблицы) **СТУДЕНТ**, содержимое которого приводится на рис.2. Структура записи файла **СТУДЕНТ** линейная она содержит записи фиксированной длины. Повторяющиеся группы значений полей в записи отсутствуют. Обращение к значению поля производится по его номеру.

## Имя файла СТУДЕНТ

Поле		Признак ключа	Формат поля		
Обозначение	Наименование		Тип	Длина	Точность
Помер	№ личного дела	*	Симв	5	
Фамилия	Фамилия студента		Сими	15	
Имя	Имя студента		Симв	10	
Отчество	Отчество студента		Симв	15	
Дата	Дата рождения		Дата	8	

- Рис.7 Описание логической структуры записи файла СТУДЕНТ

# • ВИДЫ МОДЕЛЕЙ ДАННЫХ

Ядром любой базы данных является модель данных. Модель данных представляет собой множество структур данных, ограничений целостности и операций манипулирования данными. С помощью модели данных могут быть представлены объекты предметной области и взаимосвязи между ними.

**Модель данных** — совокупность структур данных и операций их обработки.

СУБД основывается на использовании иерархической, сетевой или реляционной модели, на комбинации этих моделей или на некотором их подмножестве.

Существует три основных типа моделей данных:

**Типы моделей данных:**

**1. иерархическая,**

**2. сетевая**

**3. реляционная**

- Иерархическая структура представляет совокупность элементов, связанных между собой по определенным правилам. Объекты, связанные иерархическими отношениями, образуют ориентированный граф (перевернутое дерево).
- В сетевой структуре при тех же основных понятиях (уровень, узел, связь) каждый элемент может быть связан с любым другим элементом.

# Реляционная модель данных

Понятие **реляционный** (англ. *relation* — отношение) связано с разработками известного американского специалиста в области систем баз данных Е. Кодда.

Эти модели характеризуются простотой структуры данных, удобным для пользователя табличным представлением и возможностью использования формального аппарата алгебры отношений и реляционного исчисления для обработки данных.

Реляционная модель ориентирована на организацию данных в виде двумерных таблиц. Каждая **реляционная таблица** представляет собой двумерный массив и обладает следующими свойствами:

- каждый элемент таблицы — один элемент данных;
- все столбцы в таблице 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.

**Пример.** Реляционной таблицей можно представить информацию о студентах, обучающихся в вузе.

№ личного дела	Фамилия	Имя	Отчество	Дата рождения	Группа
16493	Сергеев	Петр	Михайлови ч	01.01.76	111
16593	Петрова	Анна	Владимиро вна	15.03.75	112
16693	Анохин	Андрей	Борисович	14.04.76	111



Отношения представлены в виде таблиц, строки которых соответствуют кортежам или записям, а столбцы — атрибутам отношений, доменам, полям.

Поле, каждое значение которого однозначно определяет соответствующую запись, называется простым ключом (ключевым полем). Если записи однозначно определяются значениями нескольких полей, то такая таблица базы данных имеет составной ключ. В примере, показанном на рис. 4, ключевым полем таблицы является "№ личного дела".

- Чтобы связать две реляционные таблицы, необходимо ключ первой таблицы ввести в состав ключа второй таблицы (возможно совпадение ключей); в противном случае нужно ввести в структуру первой таблицы **внешний ключ** — ключ второй таблицы.

**Рис. 8.** Пример реляционной модели

