МИНОБРНАУКИ РОССИИ

Передовые инженерные школы

СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Лекция 1

ЛЕКТОР

Иванова Софья Сергеевна

МФТИ (2001 г.), К.Э.Н.

20+ лет в банковском секторе

В МФТИ с 2023г.

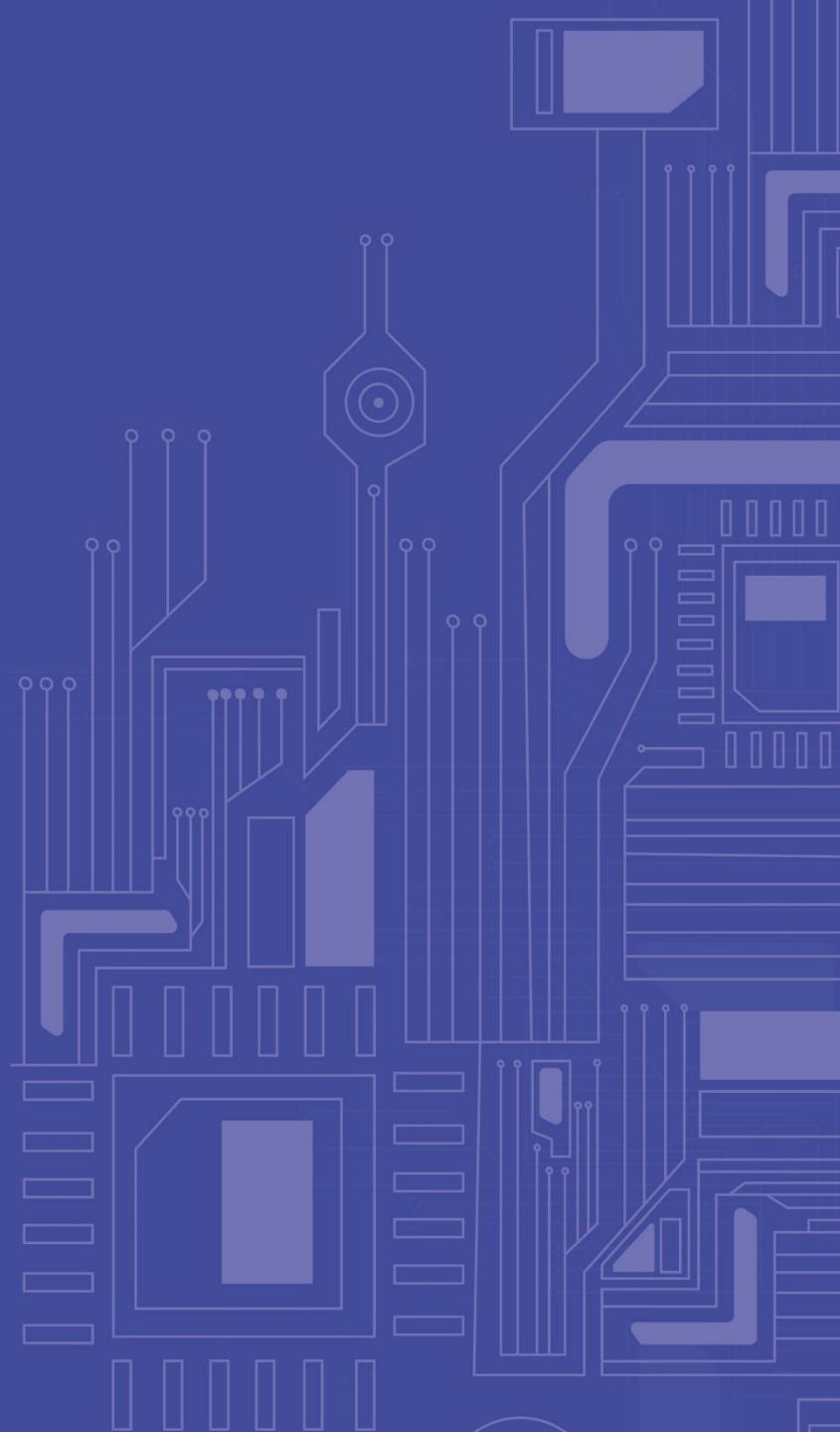
ivasonik@mail.ru

@Ivasonik

ПЛАН ЛЕКЦИИ

- Обзор курса
  - Устройства внешней памяти
    - Файловые системы
      - Структуризация данных
        - Целостность данных
          - Языки запросов
          - СУБД
        - Разные модели данных

ОБЗОР КУРСА



РАЗДЕЛЫ КУРСА И ЗАДАНИЯ

Введение 1 лекция, 1 семинар	• Контрольные вопросы • Практическое задание
Математические основы 2 лекции, 2 семинара	• Контрольные вопросы • Теоретические задачи
Проектирование 2 лекции, 2 семинара	• Контрольные вопросы • Практические задания • Теоретические задачи
Физическое представление и управление базами данных 2 лекции, 2 семинара	• Контрольные вопросы • Сдача предыдущих заданий
SQL 6 лекций, 6 семинаров	• Контрольные вопросы • Практические задания • Теоретические задачи

ТЕХНОЛОГИИ

- Ноутбук
  - Windows(/WSL/Linux)
    - GitHub
      - Sqlite3/Sqlitebrowser
        - Draw.io
          - SQLite Studio
        - Telegram

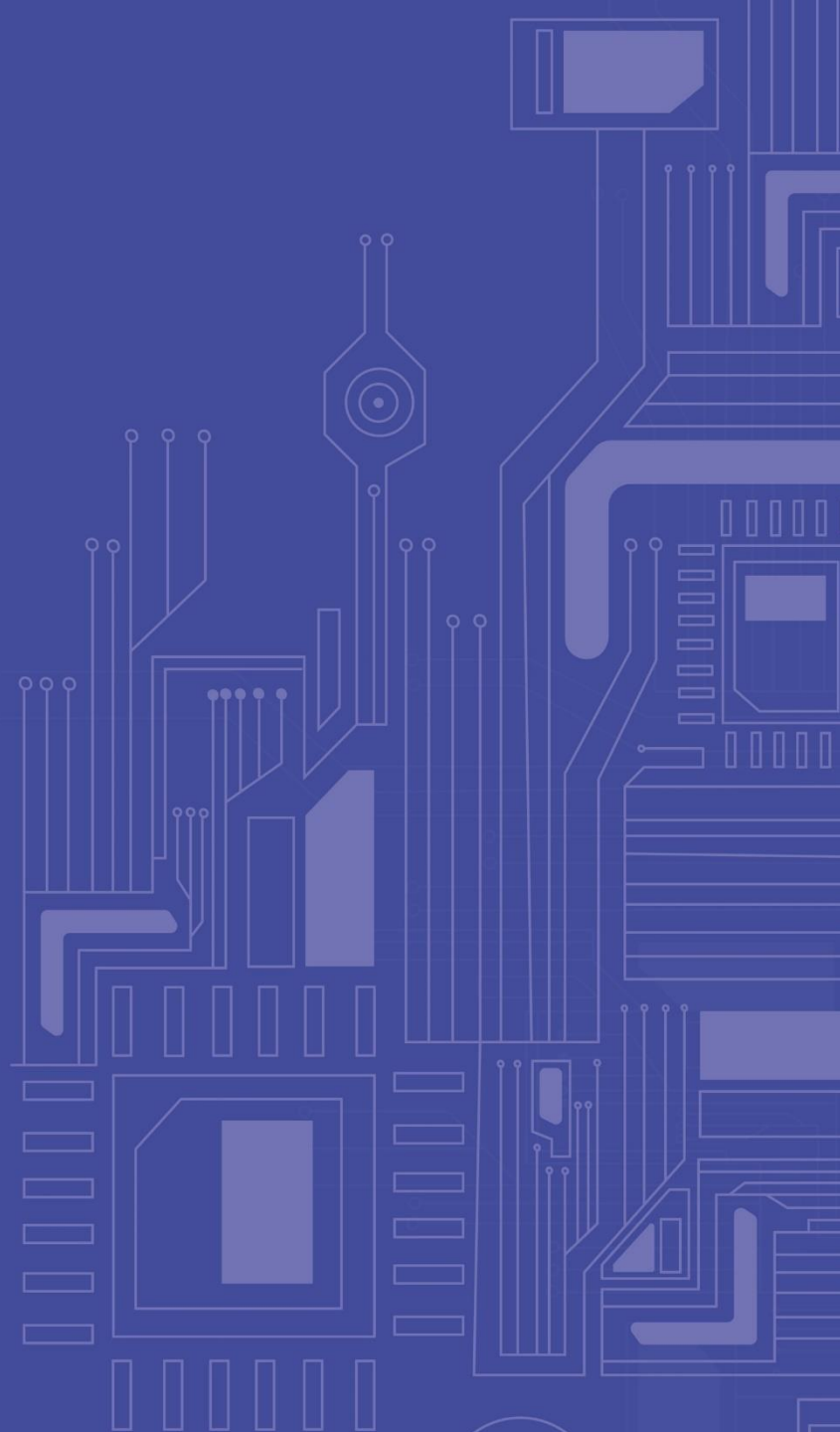
ЭКЗАМЕНЫ

Государственный экзамен

Семестровый дифференциальный зачет

- По итогам курса у каждого студента сформируется скоринг. 40% - пятерки (8-10 баллов), 40% - четверки (6-7 баллов), 20% - тройки (3-5 баллов).

УСТРОЙСТВА ВНЕШНЕЙ ПАМЯТИ



ЗАЧЕМ ЭТО ИЗУЧАТЬ?

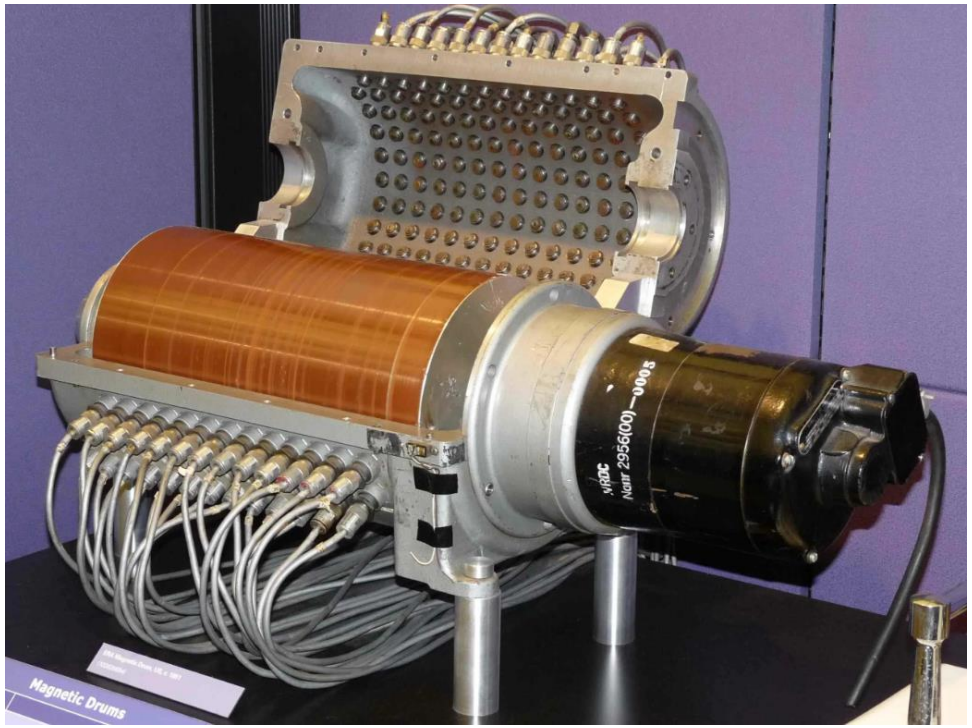
Полезно понимать как развивалась мысль при создании современной архитектуры устройств и приложений

История учит нас уважительно относиться к тому, что мы имеем

Полезно представлять физические процессы, которые обеспечивают работу современных информационных систем

УСТРОЙСТВА ВНЕШНЕЙ ПАМЯТИ (1/3)

Емкость магнитной ленты пропорциональна ее длине. Чтобы получить доступ к требуемой порции данных, нужно в среднем перемотать половину ее длины. Но чисто механическую операцию перемотки нельзя выполнить очень быстро.



Магнитный барабан

- **Магнитный барабан** — ранняя разновидность компьютерной памяти, широко использовавшаяся в 1950—1960-х годах. Изобретена Густавом Таушеком (en) в 1932 году в Австрии. Для многих вычислительных машин барабан являлся основной памятью, в которой располагались программы и данные, записываемые или считываемые с барабана при помощи таких носителей информации, как перфолента или перфокарты. Барабаны применялись настолько широко, что содержащие их вычислительные машины часто называли «барабанными компьютерами».

УСТРОЙСТВА ВНЕШНЕЙ ПАМЯТИ (2/3)

Принципы записи данных на магнитный диск

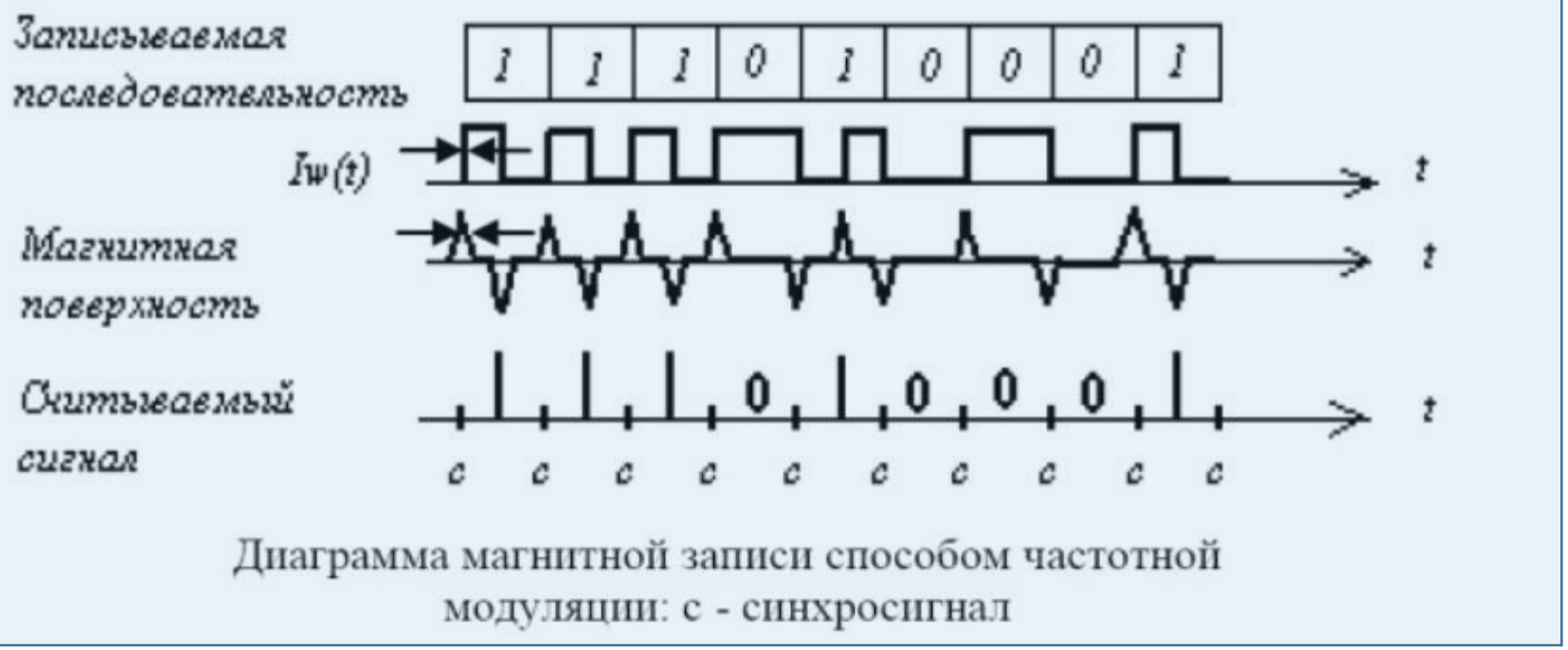
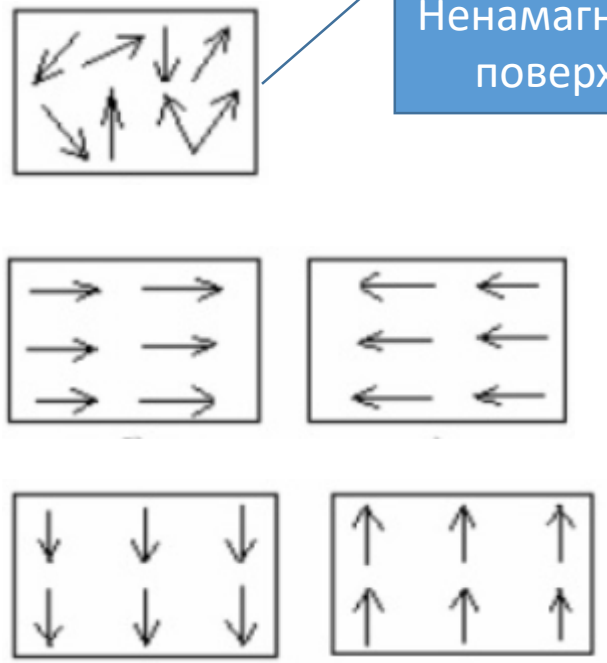
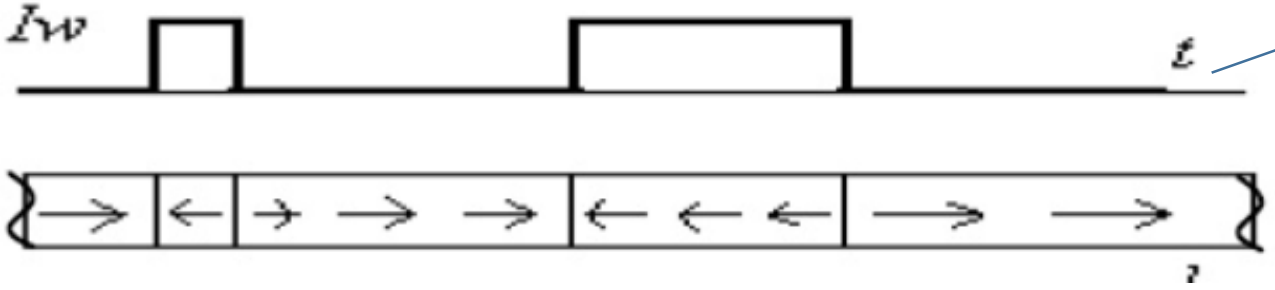


Диаграмма магнитной записи способом частотной модуляции: с - синхросигнал



Ненамагниченная поверхность

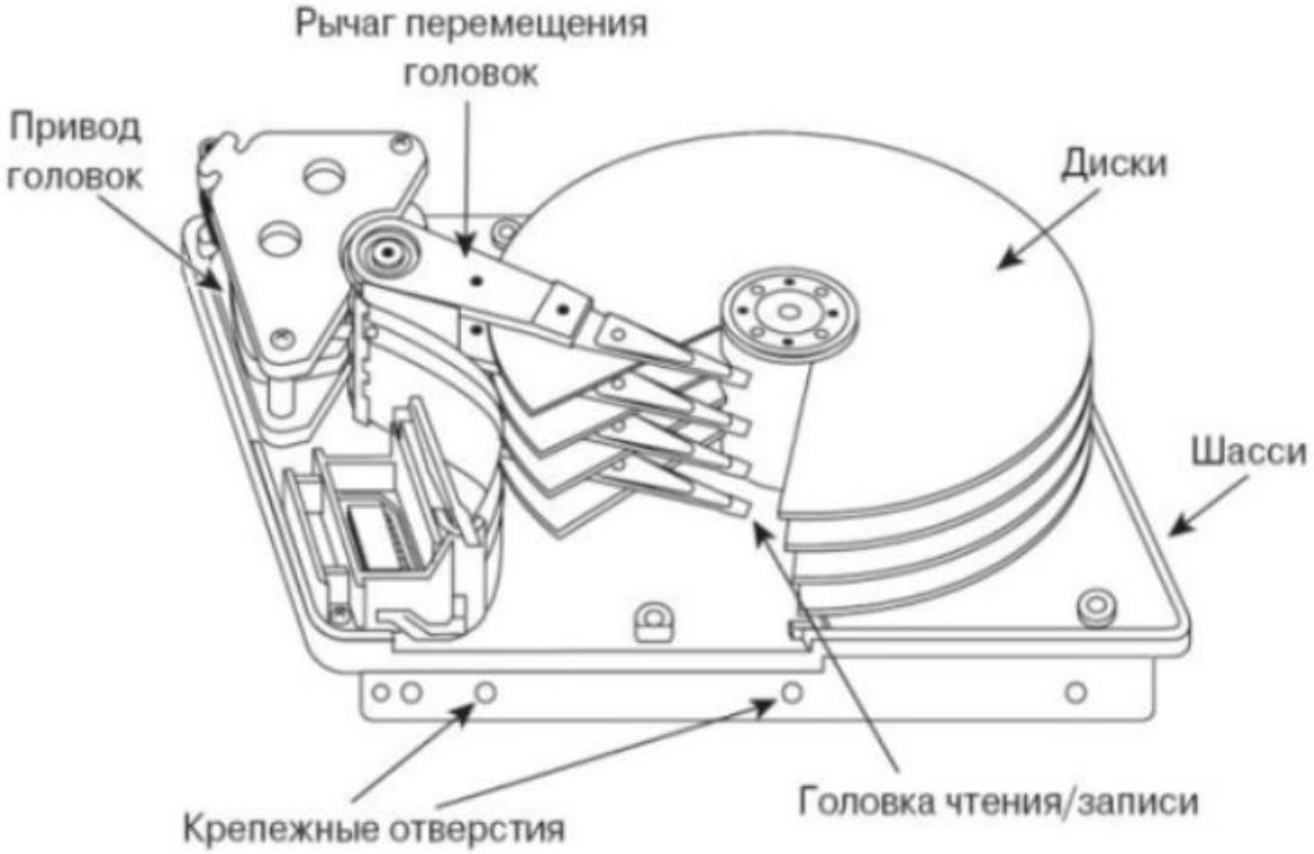


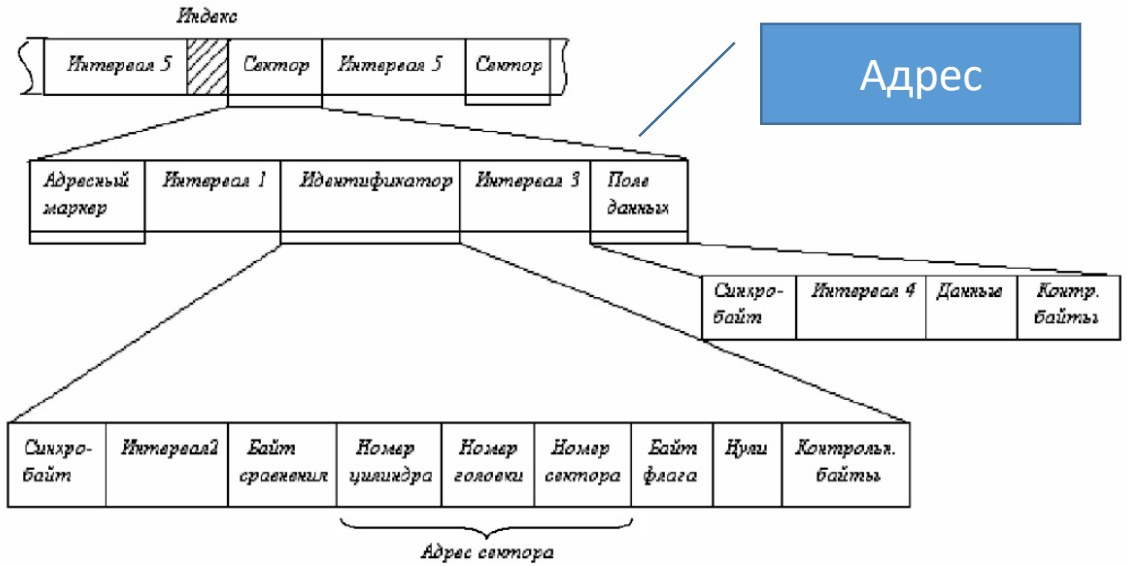
Воздействие тока на различные участки носителя при его движении (лента)

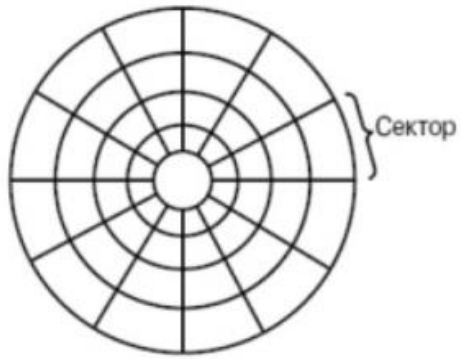
Время ожидания данных ~ ½ длины ленты (долго)

УСТРОЙСТВА ВНЕШНЕЙ ПАМЯТИ (3/3)

Накопитель на жестких магнитных дисках (НЖМД)







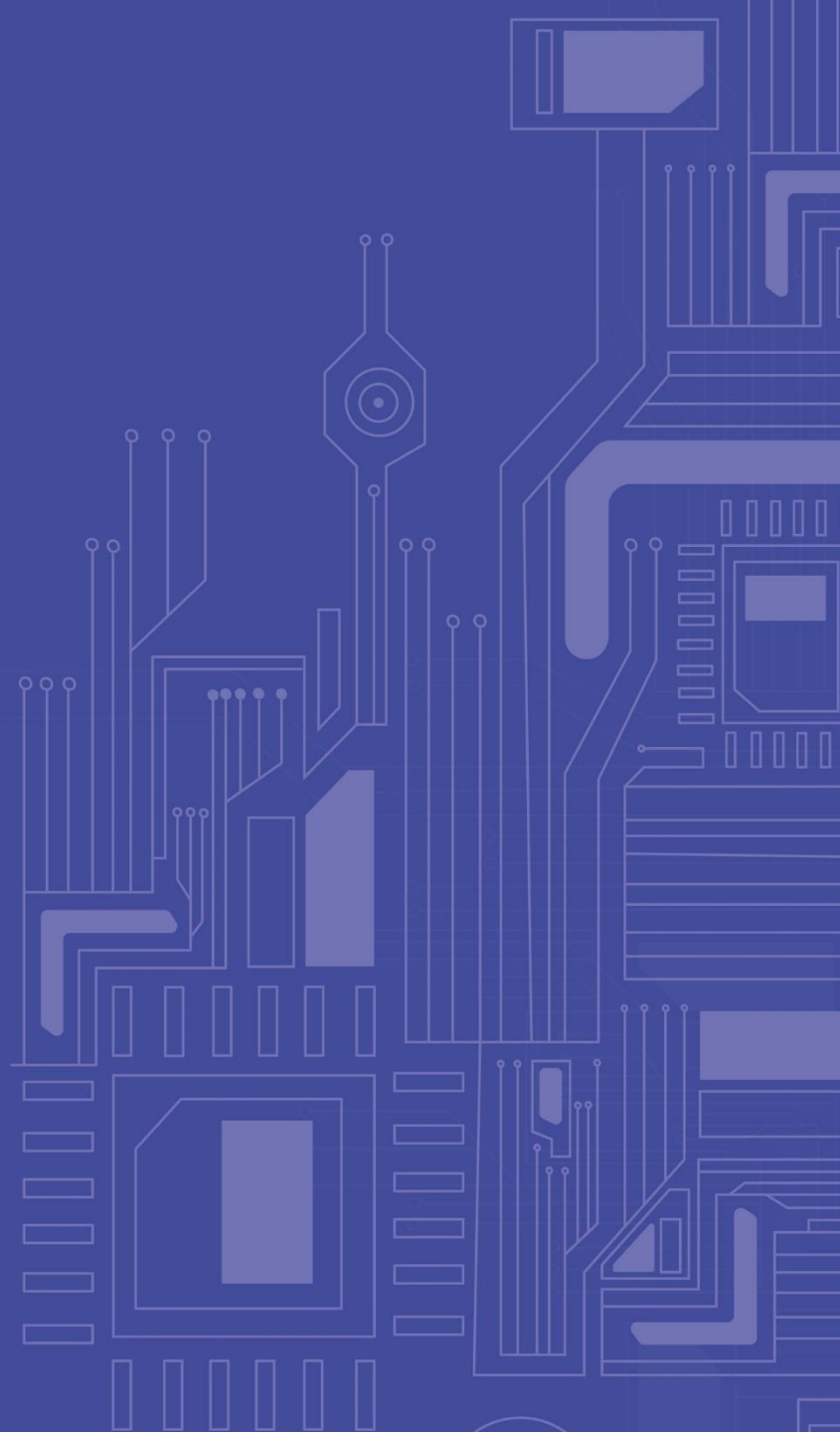
tn - среднее время позиционирования;
F - скорость вращения диска;
tобм - время обмена

Значительно быстрее ленты

Среднее время доступа к информации на НЖМД составляет

$$tср = tn + 0,5/F + tобм$$

ФАЙЛОВЫЕ СИСТЕМЫ



ЗАЧЕМ ЭТО ИЗУЧАТЬ?

При проектировании базы данных необходимо представлять образ размещения ее в памяти устройства

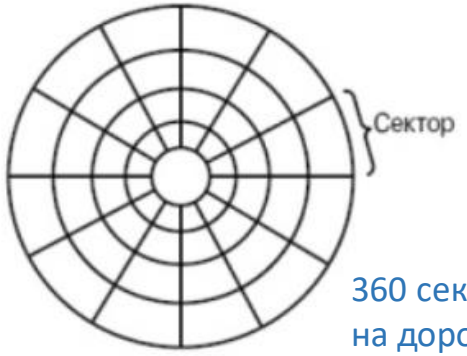
При планировании запроса к данным необходимо представлять навигацию по базе данных с учетом размещения ее в памяти

ФАЙЛОВЫЕ СИСТЕМЫ (1/3)

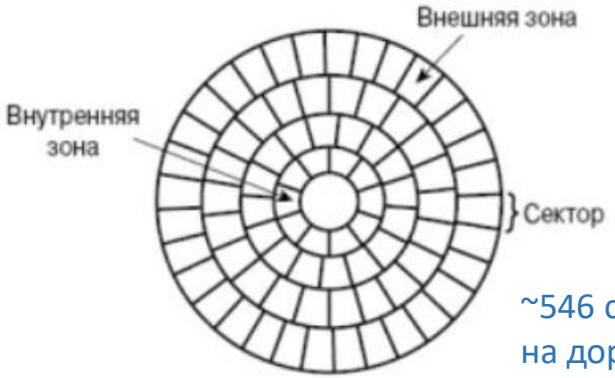
Первая развитая файловая система была разработана специалистами IBM в середине 60-х гг. для выпускавшейся компанией серии компьютеров «360». Файловая система OS/360 обеспечила будущих разработчиков уникальным опытом использования дисковых устройств с подвижными головками, который отражается во всех современных файловых системах.

ФОРМАТИРОВАНИЕ

→ Низкого уровня



360 секторов на дорожку



~546 секторов на дорожку

Существуют различия в скорости передачи данных для каждой зоны. Поскольку частота вращения шпинделя 7 200 об/мин, один оборот совершается за 1/120 секунды или же 8,33 миллисекунды. Дорожки во внешней зоне (нулевой) имеют скорость передачи данных 44,24 Мбайт/с, а во внутренней зоне (15) - всего 22,12 Мбайт/с. Средняя скорость передачи данных составляет 33,52 Мбайт/с.

→ Организация разделов на диске

Разделы, создаваемые на жестком диске, обеспечивают поддержку различных файловых систем, каждая из которых располагается на определенном разделе диска.

→ Высокого уровня

При форматировании высокого уровня операционная система создает структуры для работы с файлами и данными. В сущности, форматирование высокого уровня - это не столько форматирование, сколько создание оглавления диска и таблицы размещения файлов.

ФАЙЛОВЫЕ СИСТЕМЫ (2/3)

Возможность обмениваться с магнитными дисками порциями данных, размеры которых меньше полного объема блока, в настоящее время в файловых системах не используется во избежание внешней фрагментации памяти.

В памяти образуется большое число маленьких свободных фрагментов. Их совокупный размер может быть больше размера любого требуемого буфера, но его можно выделить, только если произвести сжатие памяти, т. е. подвижку всех занятых фрагментов таким образом, чтобы они располагались вплотную один к другому. Во время выполнения операции сжатия памяти нужно приостановить выполнение обменов, а сама эта операция занимает много времени.

Форматирование высокого уровня

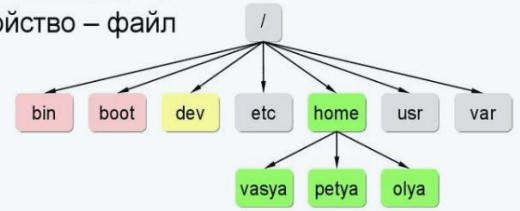
Логическая структура файловых систем и именование файлов

➡ Два подхода:

1

Файловые системы в *Linux*

- один корневой каталог «/»
- любое устройство – файл



Путь к файлу: `/home/petya/qq.txt`

`bin` – команды операционной системы;
`boot` – ядро ОС и данные для загрузки;
`dev` – файлы устройств
`etc` – файлы с настройками ОС и некоторых программ
`home` – домашние каталоги пользователей
`usr` – установленные пакеты программ
`var` – часто меняющиеся данные, например, журналы ОС

2

Во многих системах(WINDOWS) управления файлами требуется, чтобы каждый архив файлов (полное дерево каталогов) целиком располагался на одном дисковом пакете или логическом диске – разделе физического дискового пакета, логически представляемом в виде отдельного диска с помощью средств операционной системы. В этом случае полное имя файла начинается с имени дискового устройства, на котором установлен соответствующий диск.

ФАЙЛОВЫЕ СИСТЕМЫ (3/3)

Авторизация доступа к файлу

- **Мандатная**

  У каждого пользователя индивидуальный мандат на работу с файлом

- **Дискреционная**

  У каждого файла есть создатель, группа создателя и другие – у каждой из трех категорий есть/нет права на чтение/запись/запуск.

9 бит к каждому файлу

Синхронизация доступов к файлу

Режим работы:

- Чтение
- Изменение

Чтение

Много пользователей одновременно допустимо

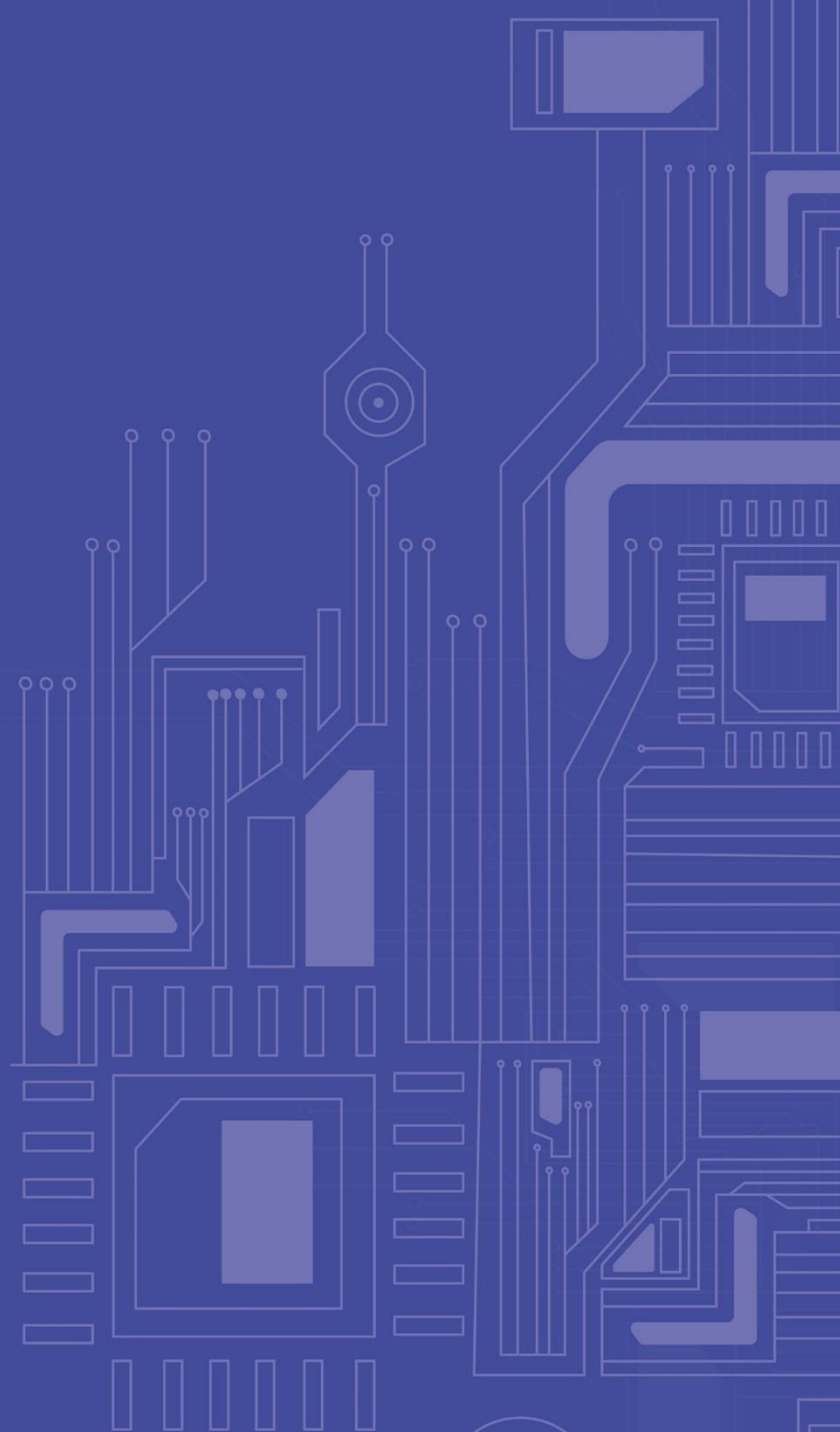
Изменение

Один меняет

Другие:

- Отмена + уведомление
- Блокировка (ожидание)

ПОТРЕБНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ



ЗАЧЕМ ЭТО ИЗУЧАТЬ?

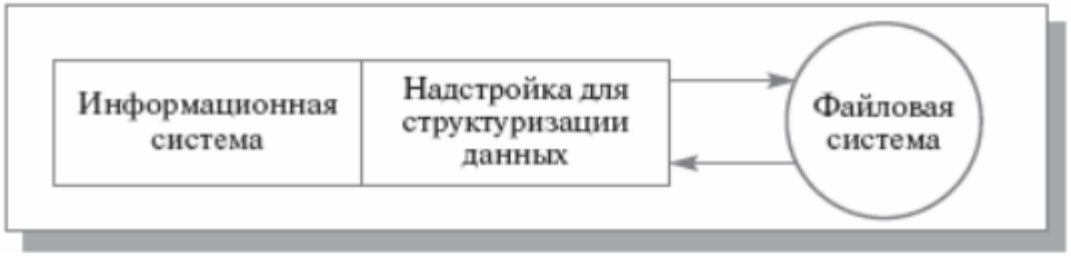
При проектировании информационной системы необходимо учитывать максимум аспектов ее работы

Здорово, когда кто-то до нас систематизировал чек-лист, описав основные проблемы

Как вообще люди дошли до идеи СУБД?

СТРУКТУРИЗАЦИЯ ДАННЫХ

В начале развития структур данных информационные системы работали с индивидуальными надстройками для структуризации данных, хранимых в файлах



Со временем похожие функции структуризации стали объединять в общие библиотеки/надстройки, что позволило экономить память, а также послужило стимулом к дальнейшему развитию в направлении СУБД

t

ПРИМЕР ИНФОРМАЦИОННОЙ СИСТЕМЫ (1/2)

Система кадрового учета: описание

ЗАДАЧИ

- выдавать списки служащих по отделам;
- поддерживать возможность перевода служащего из одного отдела в другой;
- обеспечивать средства поддержки приема на работу новых служащих и увольнения работающих служащих.

ЗАПРОСЫ

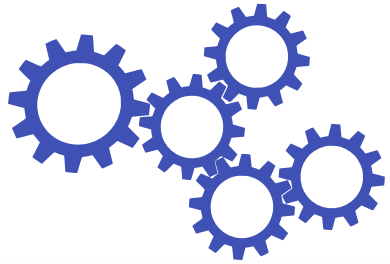
- имя руководителя отдела;
- общая численность отдела;
- общая сумма зарплаты служащих отдела, средний размер зарплаты и т. д.;
- номер удостоверения по полному имени служащего (для простоты допустим, что имена всех служащих различны);
- полное имя по номеру удостоверения;
- информация о соответствии служащего занимаемой должности и о размере его зарплаты.

ПРИМЕР ИНФОРМАЦИОННОЙ СИСТЕМЫ(2/2)

Система кадрового учета: данные

- полное имя служащего (СЛУ_ИМЯ);
- номер его удостоверения (СЛУ_НОМЕР);
- данные о соответствии служащего занимаемой должности (СЛУ_СТАТ; для простоты «да» или «нет»);
- размер зарплаты (СЛУ_ЗАРП);
- номер отдела (СЛУ_ОТД_НОМЕР);
- имя руководителя отдела (СЛУ_ОТД_РУК).

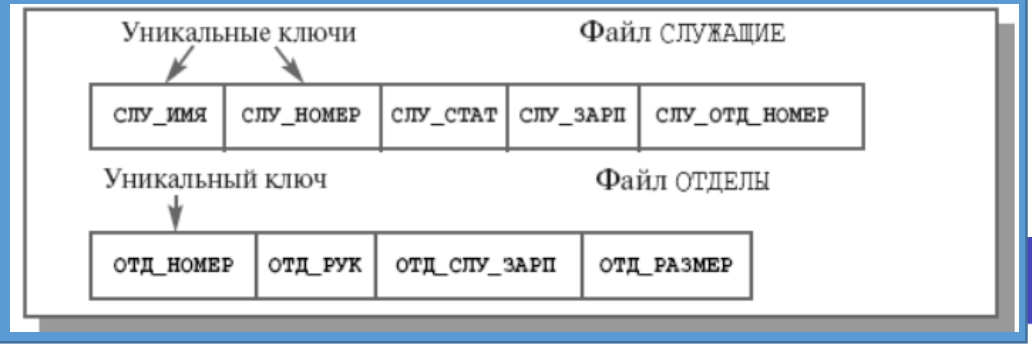
1 файл



ПРОБЛЕМЫ:

- требуется создание достаточно сложной надстройки для многоключевого доступа к файлам;
- возникает существенная избыточность данных (для каждого служащего повторяется имя руководителя его отдела);
- требуется выполнение массовой выборки и вычислений для получения суммарной информации об отделах.

Улучшение: 2 файла



ЦЕЛОСТНОСТЬ ДАННЫХ

Согласованность данных = СУБД

Пример с системой кадрового учета с двумя файлами

Ссылочная целостность	если в файле СЛУЖАЩИЕ содержится запись со значением поля СЛУ_ОТД_НОМЕР, равным n, то и в файле ОТДЕЛЫ должна содержаться запись со значением поля ОТД_НОМЕР, также равным n
	если в файле ОТДЕЛЫ содержится запись со значением поля ОТД_РУК, равным m, то и в файле СЛУЖАЩИЕ должна содержаться запись со значением поля СЛУ_НОМЕР, также равным m
Общие ограничения целостности БД	при любом корректном состоянии информационной системы значение поля ОТД_СЛУ_ЗАРП любой записи отд_k файла ОТДЕЛЫ должно быть равно сумме значений поля СЛУ_ЗАРП всех тех записей файла СЛУЖАЩИЕ, в которых значение поля СЛУ_ОТД_НОМЕР совпадает со значением поля ОТД_НОМЕР записи отд_k
	при любом корректном состоянии информационной системы значение поля ОТД_РАЗМЕР любой записи отд_k файла ОТДЕЛЫ должно быть равно числу всех тех записей файла СЛУЖАЩИЕ, в которых значение поля СЛУ_ОТД_НОМЕР совпадает со значением поля ОТД_НОМЕР записи отд_k

Нужен слой метаданных, общей библиотекой надстроек не решить

ЯЗЫКИ ЗАПРОСОВ

Задача: найти общую численность отдела, в котором работает Петр Иванович Сидоров.

Решение без инструмента: сначала ищем в файле сотрудников Сидорова, потом в файле отделов численность.

Запрос с присоединением

```
SELECT ОТД_РАЗМЕР
  FROM СЛУЖАЩИЕ, ОТДЕЛЫ
  WHERE СЛУ_ИМЯ = 'ПЕТР ИВАНОВИЧ СИДОРОВ' AND
    СЛУ_ОТД_НОМЕР = ОТД_НОМЕР;
```

Запрос с вложением

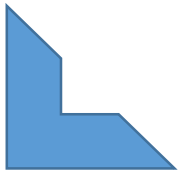
```
SELECT ОТД_РАЗМЕР
  FROM ОТДЕЛЫ
  WHERE ОТД_НОМЕР =
    (SELECT СЛУ_ОТД_НОМЕР
      FROM СЛУЖАЩИЕ
      WHERE СЛУ_ИМЯ = 'ПЕТР ИВАНОВИЧ СИДОРОВ');
```

СУБД сама за счет информации слоя метаданных поймет соответствие сущностей ОТДЕЛ и СЛУЖАЩИЙ по ключам, не надо думать, как будет исполняться запрос, можно писать на SQL в бизнес логике.

СОГЛАСОВАННОСТЬ ДАННЫХ

Транзакции, журнализация и многопользовательский режим

Аварийное отключение электричества в процессе модификации одного из файлов: рассогласованность.
СУБД осуществляет проверку и поддержание согласованности данных.



Приложение все же должно проверять допустимость операций с данными.
Для проверки согласованности также используются бизнес-кейсы.

Многопользовательский режим. СУБД осуществляет блокировку на запись, уведомление и постановку в очередь. Современные СУБД обеспечивают сложные настройки доступов с минимизацией времени ожидания.

СУБД – СИСТЕМНЫЙ КОМПОНЕНТ (1/2)

AS OF NOW

В информационной системе кадрового учета создали СУБД, управляющую двумя файлами, отвечающую за согласованность данных.

Внезапно нужна еще информационная система бухгалтерского учета

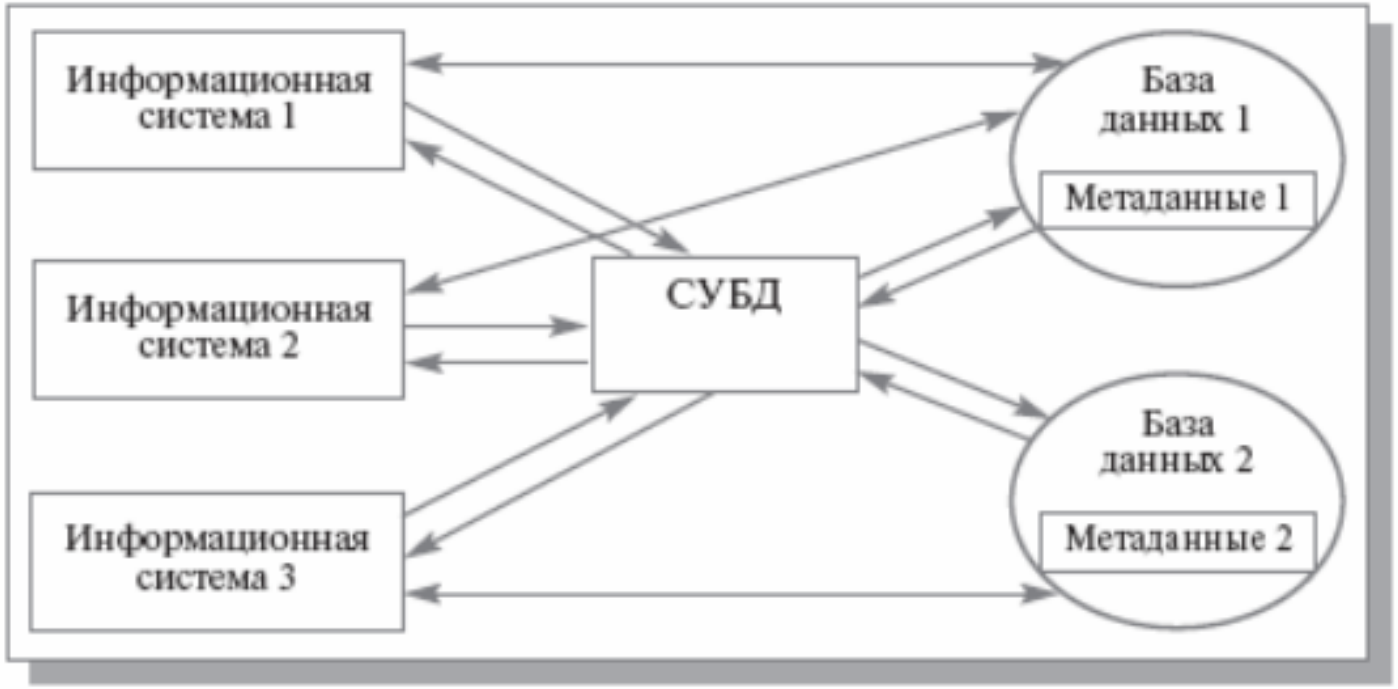
Добавить бухгалтерию в кадровый учет?

Бухгалтерские системы часто поставляются как отдельный продукт. Большие затраты на интеграцию. Функциональность бухгалтерской системы значительнее существенно.

Добавить метаданные кадрового учета в бухгалтерию?

Как синхронизировать метаданные двух информационных систем? Что делать при модификации структур метаданных?

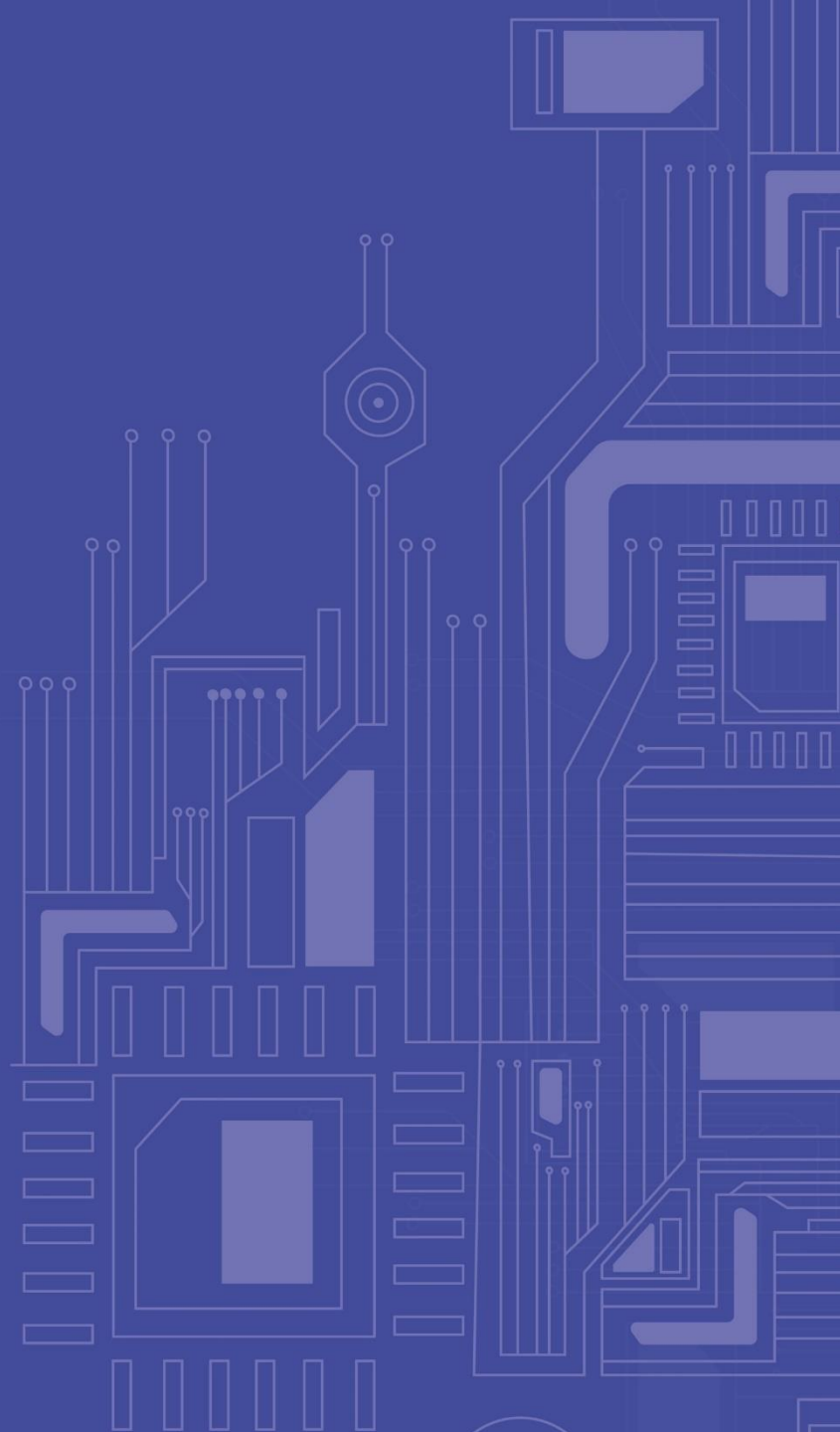
СУБД – СИСТЕМНЫЙ КОМПОНЕНТ (2/2)



Похоже на архитектуру клиент - сервер.
СУБД служит «сервером» для информационных систем.

Здесь мы видим три информационные системы, которые через одну СУБД работают с двумя разными базами данных, причем первая и вторая системы работают с общей базой данных. Это возможно, поскольку метаданные каждой базы данных содержатся в самих базах данных, и достаточно лишь указать СУБД, с какой базой данных желает работать данное приложение. Поскольку СУБД функционирует отдельно от приложений, и ее работа с базами данных регулируется метаданными, совместное использование одной базы данных двумя информационными системами не вызовет потери согласованности данных, и доступ к данным будет должным образом синхронизироваться.

МОДЕЛИ ДАННЫХ



ЗАЧЕМ ЭТО ИЗУЧАТЬ?

Оказывается, реляционная модель данных не единственно возможная

Под разные задачи подходят разные модели данных.

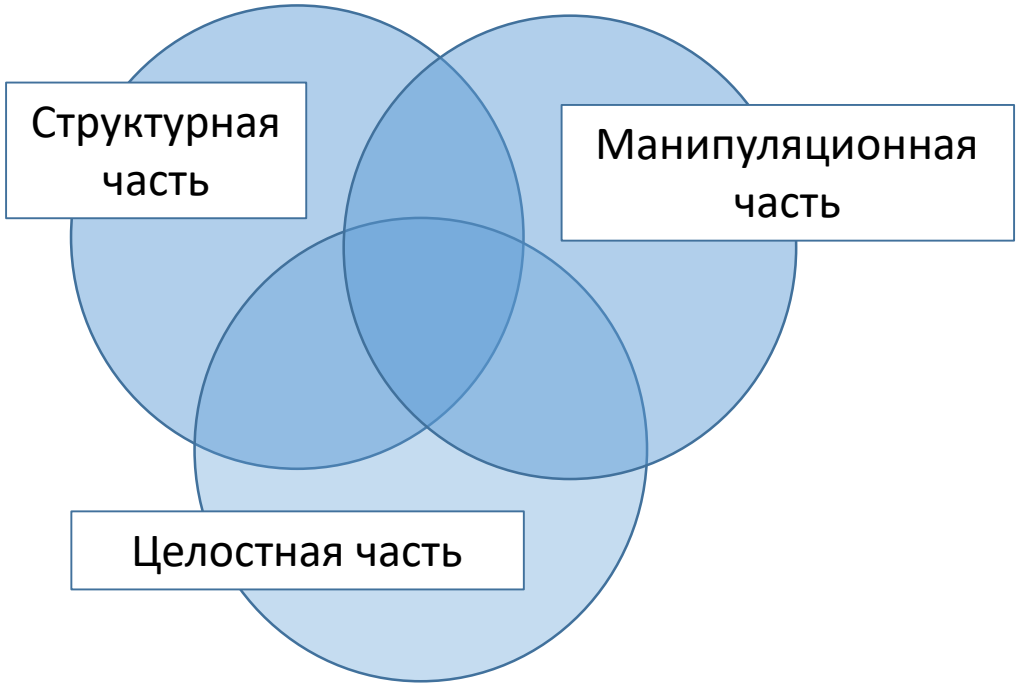
Понятие модели данных не равняется понятию базы данных или СУБД.

МОДЕЛИ ДАННЫХ

В 1969 г. Кодд ввел понятие модели данных.

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

Дейт развил это понятие для реляционной модели,
выделив три составляющие:



Структурная часть описывает из чего состоит БД (таблицы/объекты/..).

Манипуляционная часть модели данных содержит спецификацию одного или нескольких языков, предназначенных для написания запросов к БД.

Целостная часть модели данных (не во всех известных моделях есть) описывает механизмы ограничений целостности, которые обязательно должны поддерживаться во всех реализациях СУБД, соответствующих данной модели (z.B. поддержка ограничения первичного ключа в любой переменной отношения в РМД).

РАННИЕ МОДЕЛИ ДАННЫХ

Особенности:

- Ранние модели данных были до появления РМД (реляционной модели данных). До сих пор есть системы, где они используются.
- Все ранние системы не основывались на каких-либо абстрактных моделях. Впервые абстрактные представления появились у РМД, потом их применили и к другим моделям.
- Доступ к данным БД осуществлялся через доступ к записям непосредственно. Навигация по БД писалась пользователем на языке программирования. Оптимизацию доступа никакая система не поддерживала.

Примеры:

- Модель данных инвертированных таблиц
- Иерархическая модель данных
- Сетевая модель данных

МОДЕЛЬ ДАННЫХ ИНВЕРТИРОВАННЫХ ТАБЛИЦ

- СУБД Datacom/DB, выведенная на рынок в конце 1960-х гг. компанией Applied Data Research, Inc. (ADR)
- СУБД Adabas (ADAptable DAtabase System), которая была разработана компанией Software AG в 1971 г.

Структурная часть

Физический доступ к записям разрешен пользователю. Для каждой таблицы можно определить произвольное число ключей поиска, для которых строятся индексы. Эти индексы автоматически поддерживаются системой, но явно видны пользователям.

Манипуляционная часть

- Операции, устанавливающие адрес записи: прямые поисковые операторы (например, установить адрес первой записи таблицы по некоторому пути доступа) и операторы, устанавливающие адрес записи при указании относительной позиции от предыдущей записи по некоторому пути доступа.
- Операции над адресуемыми записями.

z.B. LOCATE FIRST, LOCATE NEXT, RETRIEVE, UPDATE, DELETE..

Целостная часть

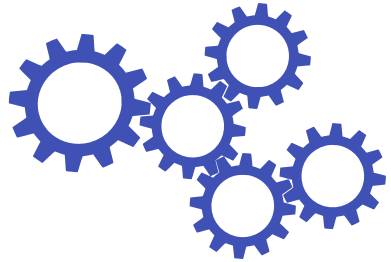
Целостность данных СУБД не поддерживает, это задача пользователя.

ИЕРАРХИЧЕСКАЯ МОДЕЛЬ ДАННЫХ (1/2)

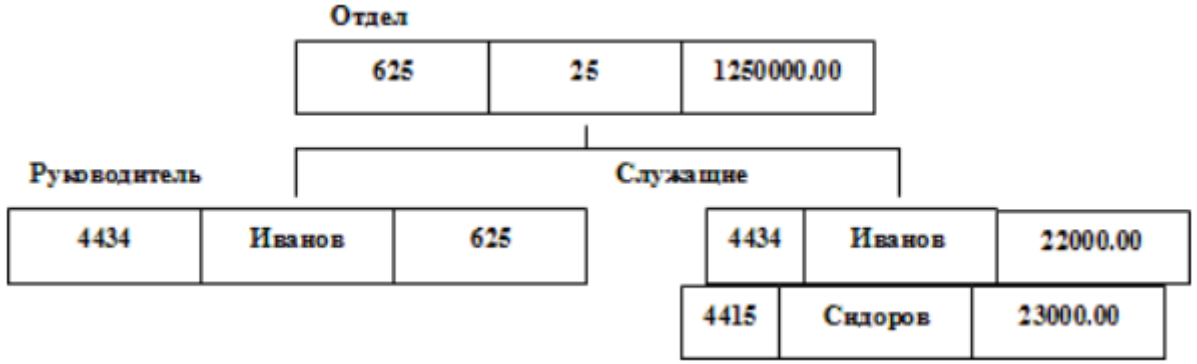
СУБД IMS (Information Management System) компании IBM. Первая версия системы появилась в 1968 г.

Структурная часть

Иерархическая БД состоит из упорядоченного набора деревьев; более точно, из упорядоченного набора нескольких экземпляров одного типа дерева.



Экземпляр дерева в базе данных



ИЕРАРХИЧЕСКАЯ МОДЕЛЬ ДАННЫХ (2/2)

Манипуляционная часть

Примерами типичных операций манипулирования иерархически организованными данными могут быть следующие:

- найти указанный экземпляр типа дерева БД (например, отдел 310);
- перейти от одного экземпляра типа дерева к другому;
- перейти от экземпляра одного типа записи к экземпляру другого типа записи внутри дерева (например, перейти от отдела к первому сотруднику);
- перейти от одной записи к другой в порядке обхода иерархии;
- вставить новую запись в указанную позицию;
- удалить текущую запись.

Целостная часть

В иерархической модели данных автоматически поддерживается целостность ссылок между предками и потомками. Основное правило: *никакой потомок не может существовать без своего родителя.*

СЕТЕВАЯ МОДЕЛЬ ДАННЫХ (1/2)

СУБД IDMS (Integrated Database Management System), разработанная компанией Cullinet Software, Inc. , основана на предложениях Data Base Task Group (DBTG) организации CODASYL (COnference on DAta SYstems Languages) от 1971 г.

Структурная часть

Сетевая БД состоит из набора записей и набора связей между этими записями. Тип связи определяется для двух типов записи: предка и потомка.

- каждый экземпляр типа записи P является предком только в одном экземпляре типа связи L;
- каждый экземпляр типа записи C является потомком не более чем в одном экземпляре типа связи L.

Свойства:

- тип записи потомка в одном типе связи L1 может быть типом записи предка в другом типе связи L2 (как в иерархии);
- данный тип записи P может быть типом записи предка в любом числе типов связи;
- данный тип записи P может быть типом записи потомка в любом числе типов связи;
- типы записи X и Y могут быть предком и потомком в одной связи и потомком и предком – в другой.

СЕТЕВАЯ МОДЕЛЬ ДАННЫХ (2/2)

Манипуляционная часть

Примерами типичных операций манипулирования сетевыми данными могут быть следующие:

- найти конкретную запись в наборе однотипных записей ;
- перейти от предка к первому потомку по некоторой связи;
- перейти к следующему потомку в некоторой связи;
- перейти от потомка к предку по некоторой связи;
- создать новую запись;
- уничтожить запись;
- модифицировать запись;
- включить в связь;
- исключить из связи;
- переставить в другую связь и т.д.

Целостная часть

Имеется (необязательная) возможность потребовать для конкретного типа связи отсутствие потомков, не участвующих ни в одном экземпляре этого типа связи (как в иерархической модели).


ВСЁ!