# 1. ОСНОВЫ БАЗ ДАННЫХ

# 1.1 ЧИСЛОВАЯ И НЕЧИСЛОВАЯ ОБРАБОТКА ДАННЫХ



**Числовая обработка данных.**

- **Компьютеры** были созданы для удовлетворения нужд крупных учреждений при реализации большого объема вычислений – *числовой обработки данных*;
- **Точность и время выполнения** – параметры существенные для числовой обработки данных
- **Примеры** числовой обработки данных: решение линейных и дифференциальных уравнений, преобразование Лапласа, операции с матрицами, векторами и т. д.).
- **Данные на уровне объектов** – *переменные; матрицы; многомерные массивы; константы и т.д.*
- **Характерная особенность**

  *Небольшой объём входных данных, сложные и длительные вычисления.*

# Нечисловая обработка данных



**Нечисловая обработка данных**

- **Возникновение** других областей применения компьютеров, отличных от вычислений:
- **Примеры нечисловой обработки**: автоматизация управления производством **о**бработка экономической информации, автоматизация конторского труда;
- **Данные на уровне объектов** – *файлы; записи; поля; иерархии; сети; отношения и т.д.*
- **Характерная особенность**

  *Большой объём входных данных, несложные вычисления.*

---

***Любые данные обладают некоторым содержанием***.

***При числовой обработке*** *– содержание не имеет большого значения*

***При нечисловой обработке*** *– содержание имеет большого значения*

# 1.2. ТРАДИЦИОННЫЕ ФАЙЛОВЫЕ СИСТЕМЫ



- **ПОДХОД, ИСПОЛЬЗУЕМЫЙ В ФАЙЛОВЫХ СИСТЕМАХ**
- **ОГРАНИЧЕНИЯ, ПРИСУЩИЕ ФАЙЛОВЫМ СИСТЕМАМ**

# 1.2.1. ПОДХОД, ИСПОЛЬЗУЕМЫЙ В ФАЙЛОВЫХ СИСТЕМАХ



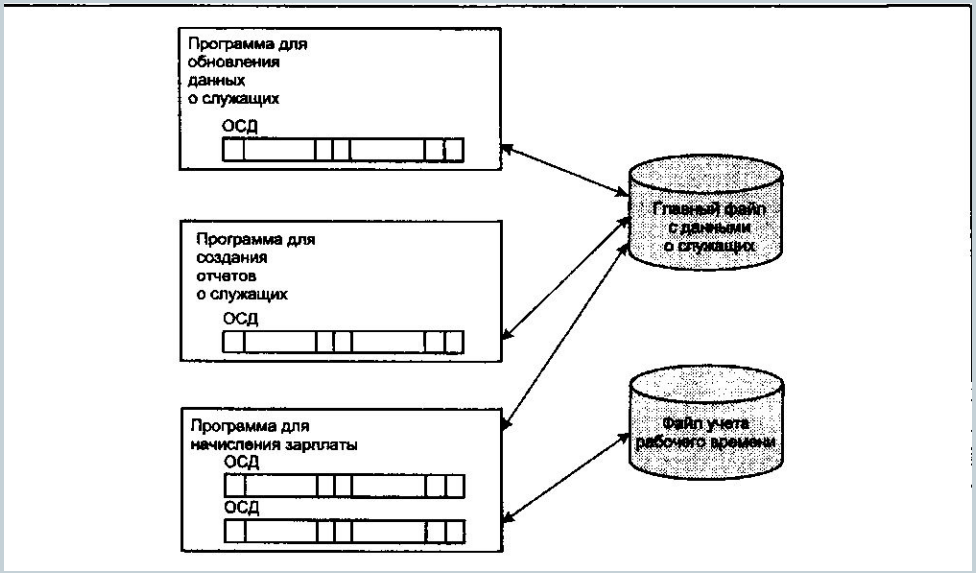
**Файловые системы были первой попыткой компьютеризировать нечисловую обработку данных.**

- ***Система управления файлами*** - является частью операционной системы, следит за именами файлов и их расположением.
- ***Файловые системы в нечисловой обработки данных***. Набор прикладных программ, которые выполняют для пользователей некоторые операции , например создание отчетов. Каждая программа хранит свои собственные данные и управляет ими.
- В системах управления файлами модели данных, как правило, не использовались, эти системы ничего не знали о внутреннем содержимом файлов.

# ПОДХОД, ИСПОЛЬЗУЕМЫЙ В ФАЙЛОВЫХ СИСТЕМАХ



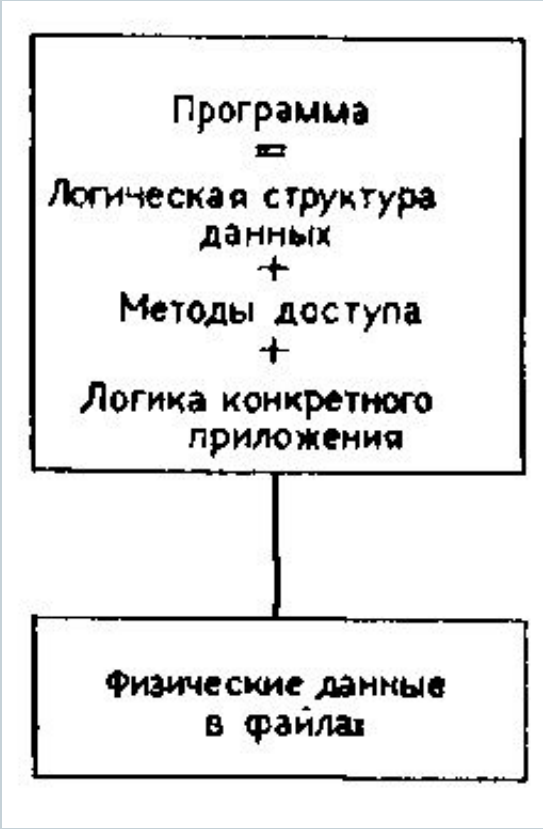
**Приложение для начисления зарплаты, использующее систему управления файлами**



# ПОДХОД, ИСПОЛЬЗУЕМЫЙ В ФАЙЛОВЫХ СИСТЕМАХ



**До применения СУБД**
**(файловая система)**



# 1.2.2. ОГРАНИЧЕНИЯ, ПРИСУЩИЕ ФАЙЛОВЫМ СИСТЕМАМ



**ОГРАНИЧЕНИЯ**

- РАЗДЕЛЕНИЕ И ИЗОЛЯЦИЯ ДАННЫХ
- ДУБЛИРОВАНИЕ ДАННЫХ
- ЗАВИСИМОСТЬ ОТ ДАННЫХ
- НЕСОВМЕСТИМОСТЬ ФАЙЛОВ
- ФИКСИРОВАННЫЕ ЗАПРОСЫ/БЫСТРОЕ УВЕЛИЧЕНИЕ КОЛИЧЕСТВА ПРИЛОЖЕНИЙ

# РАЗДЕЛЕНИЕ И ИЗОЛЯЦИЯ ДАННЫХ



- Когда данные изолированы в отдельных файлах, доступ к ним затруднён.
- Для извлечения соответствующей поставленным условиям информации программист должен организовать синхронную обработку двух файлов.
- Трудности существенно возрастают, когда необходимо извлечь данные более чем из двух файлов.

# ДУБЛИРОВАНИЕ ДАННЫХ



- Из-за децентрализованной работы с данными в файловой системе фактически допускается бесконтрольное дублирование данных.
- Бесконтрольное дублирование данных нежелательно по следующим причинам:
  - неэкономичное расходование ресурсов;
  - нарушение целостности данных.
  - необходимость дополнительного места во внешней памяти, что связано с дополнительными накладными расходами;

# ЗАВИСИМОСТЬ ОТ ДАННЫХ



- Физическая структура и способ хранения записей файлов данных жестко зафиксированы в коде приложений. Это значит, что изменить существующую структуру данных достаточно сложно.

# НЕСОВМЕСТИМОСТЬ ФОРМАТОВ ФАЙЛОВ



- Структура файлов определяется кодом приложений, она также зависит от языка программирования этого приложения.
- Прямая несовместимость файлов затрудняет процесс их совместной обработки.

# ФИКСИРОВАННЫЕ ЗАПРОСЫ/БЫСТРОЕ УВЕЛИЧЕНИЕ КОЛИЧЕСТВА ПРИЛОЖЕНИЙ



- **Файловые системы** требуют больших затрат *труда программистов*, поскольку все необходимые запросы и отчёты создаются именно им.
- В результате события реализовались по 2-м сценариям:

  **Во-первых**, во многих организациях типы применяемых запросов и отчетов имели фиксированную форму, и не было никаких инструментов создания незапланированных или произвольных *запросов как к самим данным, так и к сведениям о том, какие* типы данных доступны.
- **Во-вторых**, в других организациях наблюдалось быстрое увеличение количества файлов и приложений.

# 1.3. СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ БАЗ ДАННЫХ



**Все перечисленные ограничения файловых систем являются следствием двух факторов.**

**1. Определение данных содержится внутри приложений, а не хранится отдельно и независимо от них.**

**2. Помимо приложений не предусмотрено никаких других инструментов доступа к данным и их обработки.**

**Для повышения эффективности работы необходимо использовать новый подход, а именно *базу данных (database) и систему управления базами данных, или СУБД* (Database Management System — DBMS).**

# 1.3.1 БАЗА ДАННЫХ



- **База данных.** Совместно используемый набор логически связанных данных (и описание этих данных), предназначенный для удовлетворения информационных потребностей организации.
- **База данных** — это единое, большое хранилище данных, которое однократно определяется, а затем используется одновременно многими пользователями.
- **База данных** хранит не только рабочие данные, но и их описания. По этой причине базу данных еще называют *набором интегрированных записей с самоописанием.*
- *В совокупности описание данных называется* ***системным каталогом*** *(system catalog), или словарем данных (data dictionary), а сами элемен*ты описания принято называть *метаданными (meta-data), т.е. "данными о данных".*
- Наличие самоописания данных в базе данных обеспечивает в ней ***независимость программ от данных*** *(program-data independence).*

**БД – это логически связанный набор данных**

- При анализе информационных потребностей организации выделяют: ***сущности, атрибуты и связи***.

# БАЗА ДАННЫХ

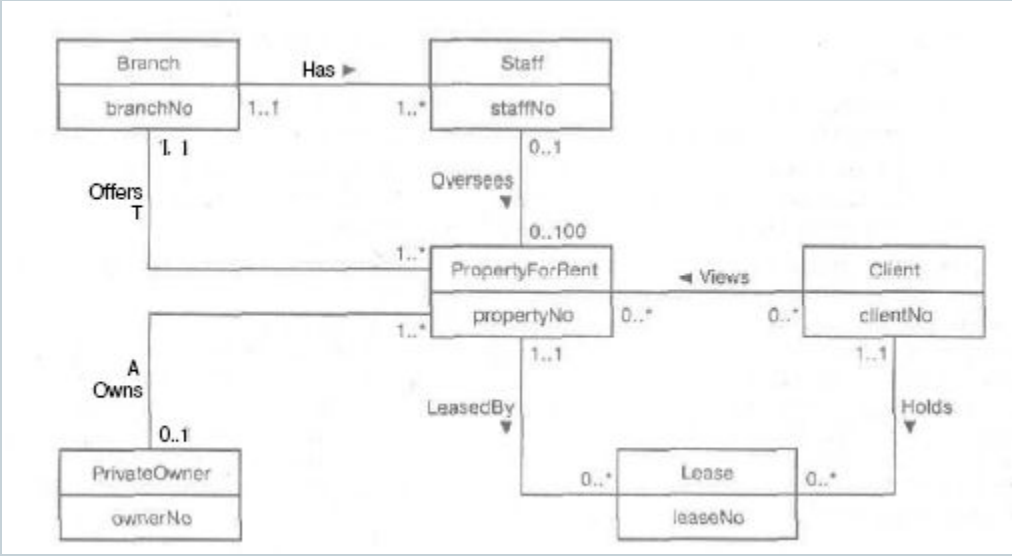


- ***Сущность(entity) -*** называется отдельный тип объекта (человек, место или вещь, понятие или событие), который нужно представить в базе данных.
- ***Атрибут***(attribute) - свойство, которое описывает некоторую характеристику рассматриваемого объекта.
- ***Связь*** *(relationship) — это то, что объединяет не*сколько сущностей.

**Возможно графическое представление БД с использованием этих понятий**

**Пример диаграммы "сущность-связь"**

- **Шесть сущностей (которые обозначены прямоугольниками):** Branch(Отделение), Staff (Работник), PropertyForRent (Сдаваемый в аренду объект), Client (Клиент), PrivateOwner (Владелец объекта недвижимости) и Lease (Договор аренды);
- **Семь связей** (которые обозначены стрелками): Has (Имеет), *Offers* (Предлагает), Oversees (Управляет), Views (Осматривает), Owns (Владеет),LeasedBy (Сдается в аренду) и Holds (Арендует);
- **Шесть атрибутов, которые соответствуют каждой сущности:** branchNo (Номер отделения), staffNo (Табельный номер работника), propertyNo (Номер сдаваемого в аренду объекта), clientNo (Номер клиента), ownerKo (Номер владельца) и leaseNo (Номер договора аренды).



# 1.3.2. СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ — СУБД



- **СУБД.** Программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также осуществлять к ней контролируемый доступ.
- **СУБД** — это программное обеспечение, которое взаимодействует с прикладными программами пользователя и базой данных и обладает перечисленными ниже возможностями:

1. Позволяет создать базу данных, что обычно осуществляется с помощью языка определения данных (DDL — Data Definition Language). Язык DDL предоставляет пользователям средства указания типа данных и их структуры, а также средства задания ограничений для информации, хранимой в базе данных.
2. Позволяет вставлять, обновлять, удалять и извлекать информацию из базы данных, что обычно осуществляется с помощью языка манипулирования данными (DML — Data Manipulation Language).

# СУБД



3. Предоставляет контролируемый доступ к базе данных с помощью перечисленных ниже средств:
   - системы обеспечения защиты, предотвращающей несанкционированный доступ к базе данных со стороны пользователей;
   - системы поддержки целостности данных, обеспечивающей непротиворечивое состояние хранимых данных;
   - системы управления параллельной работой приложений, контролирующей процессы их совместного доступа к базе данных;
   - системы восстановления, позволяющей восстановить базу данных до предыдущего непротиворечивого состояния, нарушенного в результате сбоя аппаратного или программного обеспечения;
   - доступного пользователям каталога, содержащего описание хранимой в базе данных информации.

# СУБД



**Пример реализации подхода с применением базы данных**


- Отдел реализации и отдел контрактов используют собственные приложения для доступа к общей базе данных, организованной с помощью СУБД.
- Набор приложений каждого отдела обеспечивает ввод и корректировку данных, а также генерацию необходимых отчетов.
- В отличие от варианта с файловой системой физическая структура и способ хранения данных контролируются с помощью СУБД.

# Представления

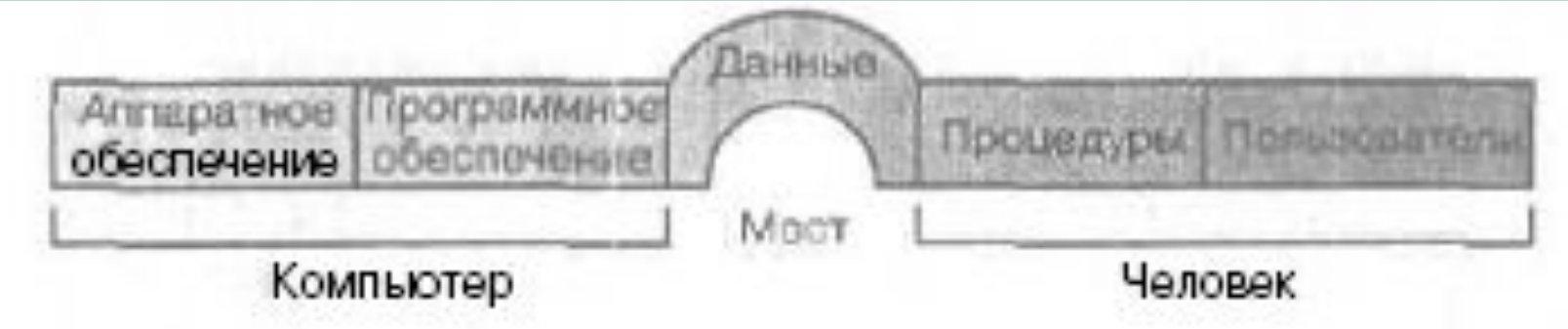


- Для решения проблемы "устранения" излишних данных в СУБД предусмотрен механизм создания **представлений (view),** который позволяет любому пользователю иметь свой собственный "образ" базы данных (представление можно рассматривать как некоторое подмножество базы данных).
- Помимо упрощения работы за счет предоставления пользователям только действительно нужных им данных, представления обладают несколькими другими достоинствами.
  - Обеспечивают дополнительный уровень безопасности.
  - Предоставляют механизм настройки внешнего интерфейса базы данных.
  - Позволяют сохранять внешний интерфейс базы данных непротиворечивым и неизменным даже при внесении изменений в ее структуру.

# 1.3.3. КОМПОНЕНТЫ СРЕДЫ СУБД



- В среде СУБД можно выделить следующие пять основных компонентов:
  - аппаратное обеспечение;
  - программное обеспечение;
  - данные;
  - процедуры ;
  - пользователи.



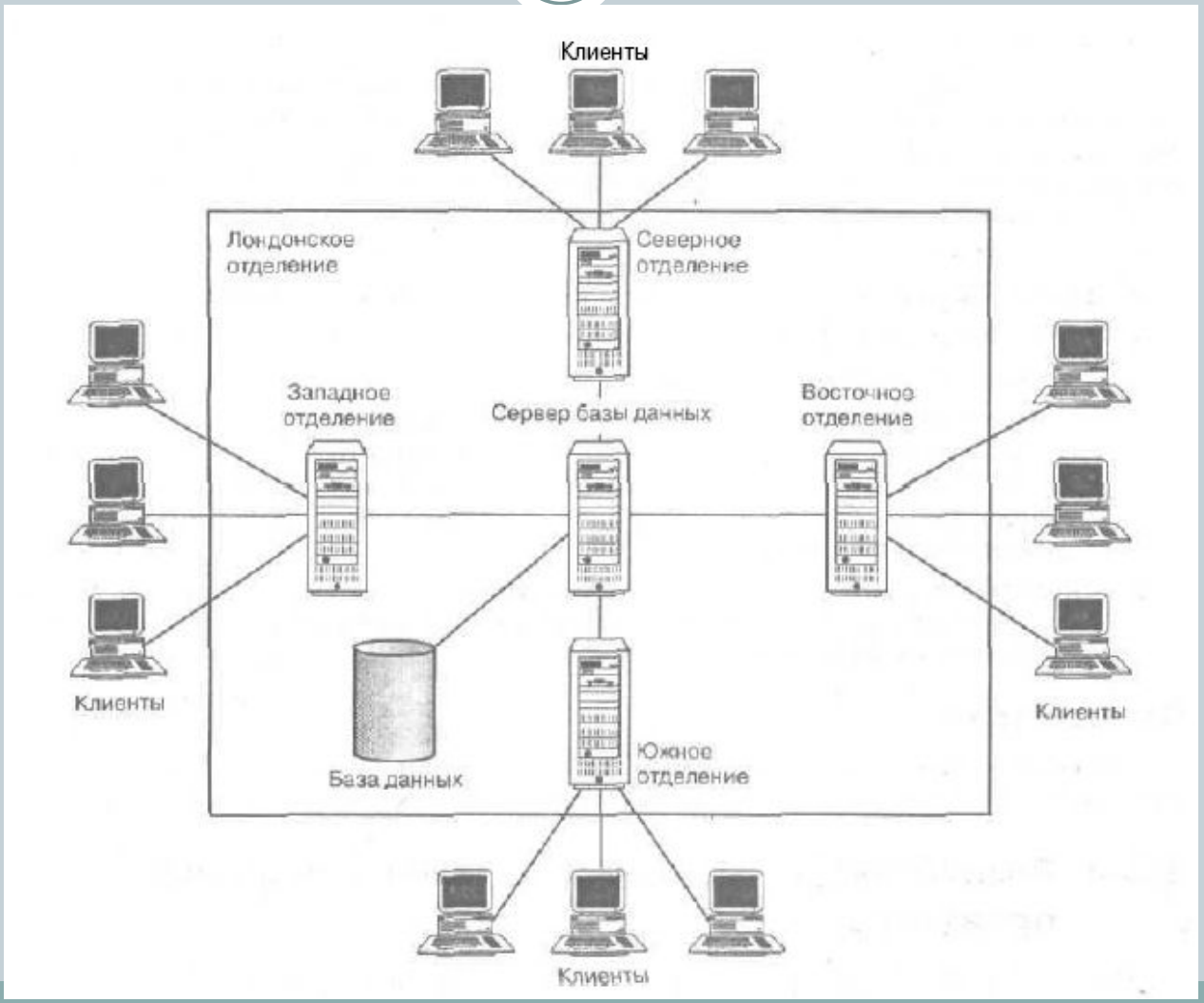
# Аппаратное обеспечение



- Для работы СУБД и приложений необходимо некоторое аппаратное обеспечение. Оно может варьировать в очень широких пределах — от единственного персональ - ного компьютера или одного мэйнфрейма до сети из многих компьютеров.
- Используемое аппаратное обеспечение зависит от требований данной организации и типа СУБД.
- Одни СУБД предназначены для работы только с конкретными типами операцион- ных систем или оборудования, другие могут работать с широким кругом аппаратно - го обеспечения и различными операционными системами.

# Конфигурация аппаратного обеспечения





# Программное обеспечение



- ПО охватывает:
  - программное обеспечение самой СУБД ;
  - прикладные программы;
  - операционные системы;
  - сетевое программное обеспечение (если СУБД используется в сети).

- Приложения создаются на языках
  - третьего поколения ( C, C++, Java, Visual Basic, COBOL, Fortran, Ada или Pascal)
  - четвертого поколения, таких как SQL, операторы которых внедряются в программы на языках третьего поколения.

- СУБД может иметь свои собственные инструменты четвертого поколения, предназначенные для быстрой разработки приложений с использованием встроенных непроцедурных языков запросов, генераторов отчетов, форм, графических изображений и даже полномасштабных приложений.

- Использование инструментов четвертого поколения позволяет существенно повысить производительность системы и способствует созданию более удобных для обслуживания программ.

# Данные



- Самым важным компонентом среды СУБД (с точки зрения конечных пользователей) являются данные.
- Данные играют роль моста между компьютером и человеком.
- База данных содержит как рабочие данные, так и метаданные, т.е. "данные о данных".
- Структура базы данных называется схемой (schema).

# Процедуры



- К процедурам относятся инструкции и правила, которые должны учитываться при проектировании и использовании базы данных.
- Пользователям и обслуживающему персоналу базы данных необходимо предоставить документацию, содержащую подробное описание процедур использования и сопровождения данной системы, включая инструкции о правилах выполнения следующих действий.
  - Регистрация в СУБД.
  - Использование отдельного инструмента СУБД или приложения.
  - Запуск и останов СУБД.
  - Создание резервных копий СУБД.
- Обработка сбоев аппаратного и программного обеспечения, а также восстановления базы данных после устранения неисправности.
  - Изменение структуры таблицы, реорганизация базы данных,
- Способы улучшения производительности и методы архивирования данных на вторичных устройствах хранения.

# Пользователи



- *Администраторы данных и администраторы баз данных*
  - Администратор данных, или АД {Data Administrator — DA), отвечает за управление данными, включая планирование базы данных, разработку и сопровождение стандартов, прикладных алгоритмов и деловых процедур, а также за концептуальное и логическое проектирование базы данных.
  - Администратор базы данных, или АБД (Database Administrator — DBA), отвечает за физическую реализацию базы данных, включая физическое проектирование и воплощение проекта; за обеспечение безопасности и целостности данных; за сопровождение операционной системы, а также за обеспечение максимальной производительности приложений и пользователей.
  - По сравнению с АД обязанности АБД носят более технический характер, и для него необходимо знание конкретной СУБД и системного окружения.
- *Разработчики баз данных.* В проектировании больших баз данных участвуют разработчики двух разных типов: разработчики логической базы данных и разработчики физической базы данных

# Пользователи



- Разработчик логической базы данных занимается идентификацией данных {т.е. сущностей и их атрибутов), связей между данными, и устанавливает ограничения, накладываемые на хранимые данные. Разработчик логической базы данных должен обладать всесторонним и полным пониманием структуры данных организации и ее делового регламента.
- Разработчик физической базы данных получает готовую логическую модель данных и занимается ее физической реализацией, в том числе:
  • преобразованием логической модели данных в набор таблиц и ограничений целостности данных;
  • выбором конкретных структур хранения и методов доступа к данным, обеспечивающих необходимый уровень производительности при работе с базой данных;
  • проектированием любых требуемых мер защиты данных.

# Пользователи



- *Прикладные программисты*
  - После создания базы данных следует разработка приложений, предоставляющих пользователям необходимые им функциональные возможности.
  - Эту работу и выполняют прикладные программисты.
  - Программы могут создаваться на различных языках программирования третьего или четвертого поколения.

- *Конечные пользователи*
  - Конечные пользователи являются клиентами базы данных — она проектируется, создается и поддерживается для того, чтобы обслуживать их информационные потребности.

# 1.3.4. РАЗРАБОТКА БАЗЫ ДАННЫХ — СМЕНА ПРИНЦИПОВ ПРОЕКТИРОВАНИЯ



- Структура базы данных определяется во время ее проектирования.
- Для создания системы, которая удовлетворяла бы информационным потребностям некоторой организации, необходимо использовать подход, совершенно отличающийся от методов разработки обычных файловых систем, в которых вся работа заключается в разработке приложений, удовлетворяющих нуждам отдельных подразделений.
- Для успешной реализации системы на основе базы данных необходимо подумать прежде всего о данных и лишь потом о приложениях.
- Такая смена подхода вполне может расцениваться как смена принципов проектирования.
- Плохо спроектированная база данных будет порождать ошибки, способные привести к принятию неправильных решений, которые повлекут за собой самые серьезные последствия для данной организации.
- С другой стороны, хорошо спроектированная база данных позволит создать систему, поставляющую корректную информацию, которая может успешно использоваться для принятия правильных и эффективных решений.

# 1.5. ПРЕИМУЩЕСТВА И НЕДОСТАТКИ СУБД



**ПРЕИМУЩЕСТВА**

- Контроль за избыточностью данных
- Непротиворечивость данных
- Больше полезной информации при том же объеме хранимых данных
- Совместное использование данных
- Поддержка целостности данных
- Повышенная безопасность
- Применение стандартов
- Повышение эффективности с ростом масштабов системы
- Возможность нахождения компромисса при противоречивых требованиях
- Повышение доступности данных и их готовности к работе
- Улучшение показателей производительности
- Упрощение сопровождения системы за счет независимости отданных
- Улучшенное управление параллельной работой
- Развитые службы резервного копирования и восстановления

# Контроль за избыточностью данных



- Традиционные файловые системы неэкономно расходуют внешнюю память, сохраняя одни и те же данные в нескольких файлах.
- При использовании базы данных предпринимается попытка исключить избыточность данных за счет интеграции файлов, что позволяет исключить необходимость хранения нескольких копий одного и того же элемента информации.
- Избыточность информации в базах данных не исключается, а лишь ограничивается ее степень.
- В одних случаях ключевые элементы данных необходимо дублировать для моделирования связей, а в других случаях некоторые данные требуется дублировать из соображений повышения производительности системы.

# Непротиворечивость данных



- Устранение избыточности данных или контроль над ней позволяет уменьшить риск возникновения противоречивых состояний. Если элемент данных хранится в базе только в одном экземпляре, то для изменения его значения потребуется выполнить только одну операцию обновления, причем новое значение станет доступным сразу всем пользователям базы данных.
- Если элемент данных с ведома системы хранится в базе данных в нескольких экземплярах, то такая система сможет следить за тем, чтобы копии не противоречили друг другу.
- Во многих современных СУБД такой способ обеспечения непротиворечивости данных не поддерживается автоматически.

# Больше полезной информации при том же объеме хранимых данных



- Благодаря интеграции рабочих данных организации на основе тех же данных можно получать дополнительную информацию.

# Совместное использование данных



- Файлы обычно принадлежат отдельным лицам или целым отделам, которые используют их в своей работе.
- База данных принадлежит всей организации в целом и может совместно использоваться всеми зарегистрированными пользователями.

# Поддержка целостности данных



- Целостность базы данных означает корректность и непротиворечивость хранимых в ней данных.
- Целостность обычно описывается с помощью ограничений, т.е. правил поддержки непротиворечивости, которые не должны нарушаться в базе данных.
- Ограничения можно применять к элементам данных внутри одной записи или к связям между записями.
- Интеграция данных позволяет АБД задавать требования по поддержке целостности данных, а СУБД применять их.

# Повышенная безопасность



- Безопасность базы данных заключается в защите базы данных от несанкци - онированного доступа со стороны пользователей;
- Без привлечения соответствующих мер безопасности интегрированные данные становятся более уязвимыми, чем данные в файловой системе.
- Интеграция позволяет АБД определить требуемую систему безопасности базы данных, а СУБД привести ее в действие.
- Система обеспечения безопасности может быть выражена в форме имен и паролей для идентификации пользователей, которые зарегистрированы в этой базе данных.

# Применение стандартов



- Интеграция позволяет АБД определять и применять необходимые стандарты:
  - отдела ;
  - организации;
  - государственные ;
  - международные;
- Стандарты могут регламентировать:
  - формат данных при обмене ими между системами;
  - соглашения об именах;
  - форму представления документации;
  - процедуры обновления;
  - правила доступа.

# Улучшенное управление параллельной работой



- В файловых системах при одновременном доступе к одному и тому же файлу двух пользователей может возникнуть конфликт двух запросов, результатом которого будет потеря информации или утрата ее целостности.

- В СУБД предусмотрена возможность параллельного доступа к базе данных и гарантируется отсутствие подобных проблем.

# Развитые службы резервного копирования и восстановления



- Ответственность за обеспечение защиты данных от сбоев аппаратного и программного обеспечения в файловых системах возлагается на пользователя.
- В современных СУБД предусмотрены средства снижения вероятности потерь информации при возникновении различных сбоев.

# Недостатки СУБД



- Сложность
- Размер
- Стоимость СУБД
- Дополнительные затраты на аппаратное обеспечение
- Затраты на преобразование
- Производительность
- Более серьезные последствия при выходе системы из строя

# Сложность и размер



- Обеспечение функциональности, которой должна обладать каждая хорошая СУБД, сопровождается значительным усложнением программного обеспечения СУБД.
- Чтобы воспользоваться всеми преимуществами СУБД, пользователи должны хорошо понимать функциональные возможности СУБД.
- Сложность и широта функциональных возможностей приводит к тому, что СУБД становится чрезвычайно сложным программным продуктом, который может потребовать много места на диске и нуждаться в большом объеме оперативной памяти для эффективной работы.

# Стоимость СУБД



- В зависимости от имеющейся вычислительной среды и требуемых функциональных возможностей стоимость СУБД может изменяться в очень широких пределах ($100-$1 000 000)
- Ежегодные расходы на сопровождение системы, составляют некоторый процент от ее общей стоимости.

# Производительность



- Обычно файловая система создается для некоторых специализированных приложений (например для оформления счетов), а потому ее производительность может быть весьма высока.
- СУБД предназначены для решения более общих задач и обслуживания сразу нескольких приложений, а не какого-то одного из них. В результате многие приложения в новой среде будут работать не так быстро, как прежде.

# Более серьезные последствия при выходе системы из строя



- Централизация ресурсов повышает уязвимость системы.
- Работа всех пользователей и приложений зависит от готовности к работе СУБД,
- Выход из строя одного из  компонентов СУБД может привести к полному прекращению всей работы организации.

# Резюме