Тема 0. Базы данных. Введение

1. Концепция баз данных

1.1. Файловые системы

Настоящая история информационных систем начинается с появлением магнитных дисков.

Это позволило осуществить переход к использованию систем управления файлами.

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

1.2. Файловые системы обработки данных (ФСОД)

ФСОД представляли собой совокупности программ, написанных на языках высокого уровня. Каждая такая программа решала одну или несколько задач обработки данных.

Например, начисляла зарплату сотрудникам и печатала платёжные ведомости и расчётные листки или генерировала и печатала складской отчёт, или рассчитывала и печатала смету и т.п.

Основные недостатки таких систем

1. ***Зависимость программ от данных.*** ***В файловой системе определения файлов и способы доступа зафиксированы в теле прикладной программы. Если нужно изменить структуру какого-то файла, то программу придётся переписывать полностью.***

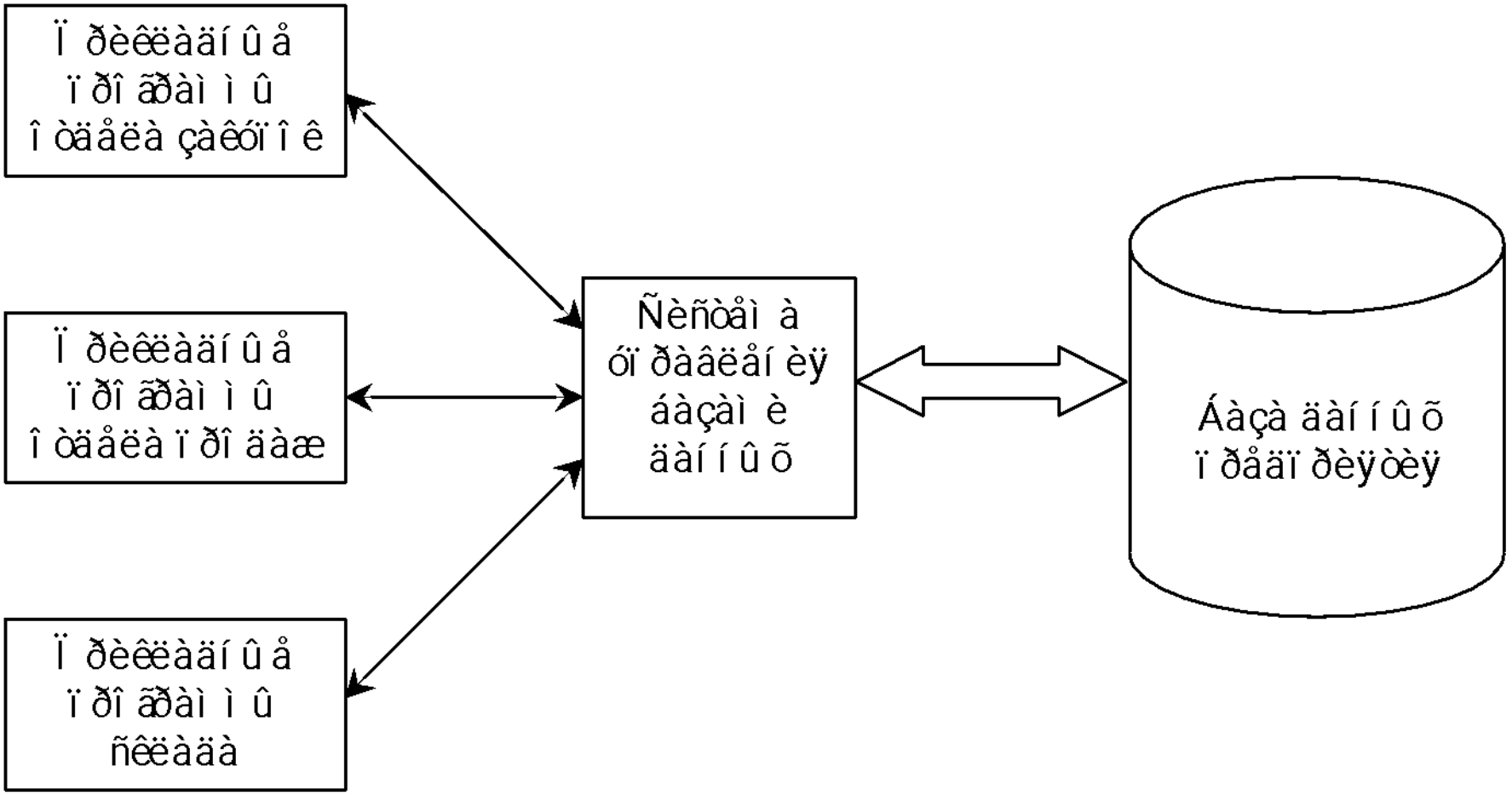
Было бы хорошо, если бы определения данных физического уровня не нужно было включать в тело прикладной программы.

2. ***Большое количество прикладных программ в системе.*** ***Каждая прикладная программа ФСОД создаётся для выполнения небольшого числа предопределённых запросов конечного пользователя.***

Было бы хорошо, если бы система могла обрабатывать произвольные запросы пользователей без посредства прикладных программистов.

1.3. Базы данных (БД)

Схема обработки данных предприятия в технологии БД



Основная особенность СУБД – это наличие процедур для ввода и хранения не только самих данных, но и описаний их структуры. Файлы, снабженные описанием хранимых в них данных и находящиеся под управлением СУБД, стали называть банки данных, а затем "*Базы данных" (БД)*.

Пример. Требуется хранить расписание движения самолетов и ряд других данных, связанных с организацией работы аэропорта (БД "Аэропорт").

Номер рейса	Дни недели	Пункт отправления	Время вылета	Пункт назначения	Время прибытия	Тип самолета	Цена билета
138	2_4_7	Баку	21.12	Москва	0.52	ИЛ-86	115.00
57	3_6	Ереван	7.20	Киев	9.25	ТУ-154	92.00
1234	2_6	Казань	22.40	Баку	23.50	ТУ-134	73.50
242	1 по 7	Киев	14.10	Москва	16.15	ТУ-154	57.00
86	2_3_5	Минск	10.50	Сочи	13.06	ИЛ-86	78.50
137	1_3_6	Москва	15.17	Баку	18.44	ИЛ-86	115.00
241	1 по 7	Москва	9.05	Киев	11.05	ТУ-154	57.00

```
СОЗДАТЬ ТАБЛИЦУ Расписание
   (Номер_рейса            Целое,
    Дни_недели             Множество,
    Пункт_отправления      Текст (24),
    Время_вылета           Время,
    Пункт_назначения       Текст (24),
    Время_прибытия         Время,
    Тип_самолета           Текст (8),
    Стоимость_билета       Валюта);
```

Сформировав запрос

```
ВЫБРАТЬ  Номер_рейса, Дни_недели,
Время_вылета
ИЗ ТАБЛИЦЫ  Расписание
ГДЕ  Пункт_отправления = 'Москва'
  И  Пункт_назначения  = 'Киев'
  И  Время_вылета > 17;
```

получим расписание "Москва-Киев" на вечернее время

```
ВЫБРАТЬ  КОЛИЧЕСТВО(Номер_рейса)
ИЗ ТАБЛИЦЫ  Расписание
ГДЕ  Пункт_отправления = 'Москва'
  И  Пункт_назначения  = 'Минск';
```

Получим количество рейсов "Москва-Минск".

Эти запросы не потеряют актуальности и при расширении таблицы:

```
ДОБАВИТЬ В ТАБЛИЦУ Расписание
    Длительность_полета  Целое;
```

2. Классификация СУБД.

2.1. По хранимой информации.

В настоящее время принято выделять два крупных класса таких систем по характеру хранимой информации – документальные и фактографические.

1. Документальные системы используются для работы с текстами на естественном языке. Они обеспечивают их смысловой анализ и поиск по различным критериям. Мы здесь об этих системах говорить не будем.

2. Фактографические системы работают с фактами, относящимися к конкретной деятельности, представленными в виде наборов взаимосвязанных записей. Они могут выполнять не только справочные функции, но и обработку записей, т.е. ввод, редактирование, хранение, сортировку, группирование записей однородной структуры, и представление результатов обработки в виде отчётов табличной формы.

Предмет нашего курса – фактографические системы.

Фактографические системы также можно разделить на два класса по *способу использования.*

1) Системы оперативной обработки данных ориентированы на быстрое выполнение большого количества относительно простых запросов многих пользователей.

Это основа каждодневного функционирования предприятия: принятие заказов клиентов, учет сырья, складской учет, учет оплаты продукции, т.е. главным образом учетные функции.

Пользователи таких систем выполняют ***в оперативном режиме обновление хранимых данных и используют данные для принятия оперативных решений. Область использования таких систем:***

- регистрация платежей,
- резервирование мест в гостиницах, на поездах, самолётах,
- регистрация отпуска материалов со склада, принятие заказов клиентов, учет сырья, складской учет, учет оплаты продукции, т.е. главным образом учетные функции и т.п.

Логическая единица работы с данными в такой системе – *транзакция.*

Транзакция – это некоторое логически завершённое действие над базой данных.

Например, перевод денег с одного банковского счёта на другой предполагает

- **извлечение записи счёта-источника,**
- **уменьшение его остатка на переводимую сумму,**
- **извлечение записи счёта-приёмника,**
- **увеличение его остатка на ту же сумму и**
- **сохранение обновлённых записей счетов в базе данных.**

Эта совокупность действий и есть транзакция.

Системы операционной обработки данных поддерживают транзакции как ***неделимые единицы работы над данными. Поэтому они называются системами оперативной обработки транзакций или OLTP-системами (On-Line Transaction Processing).***

2) **Системы оперативного анализа данных** или OLAP-системы (On-Line Analysis Processing).

Предназначены для выполнения более сложных запросов, требующих анализа больших совокупностей данных, накопленных за некоторый промежуток времени.

Они обеспечивают выделение содержательной информации из накопленных данных – *извлечение знаний о каких-то закономерностях*. Эти системы не поддерживают транзакции. Хранимые данные обновляются редко.

Пользователями таких систем являются обычно менеджеры среднего звена. Они используют аналитические возможности системы для принятия управленческих решений

2.2. По моделям представления данных

базы данных делят на:

1)иерархические;

2)сетевые;

3)реляционные;

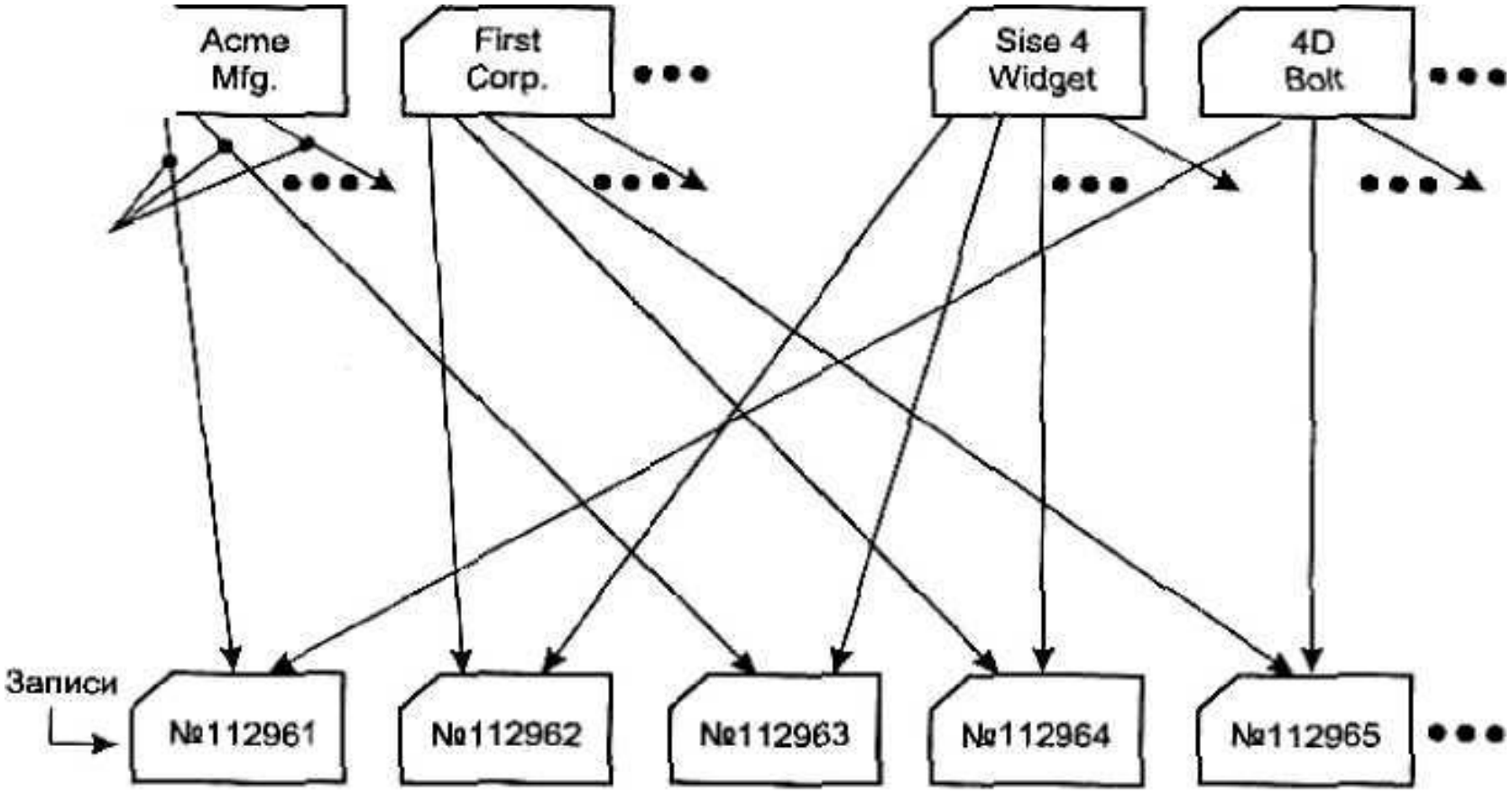
4)объектно-реляционные.

1) Иерархические базы данных - это самая первая модель представления данных, в которой все записи базы данных представлены в виде дерева, с отношениями предок-потомок.

Физически данные отношения реализуются в виде указателей на предков и потомков, содержащихся в самой записи. **Предок всегда один.**

Иерархическая модель не является оптимальной. Допустим, что один и тот же тип болтов используется в автомобиле 300 раз в различных узлах. При использовании иерархической модели, данный тип болтов будет фигурировать в базе данных не 1 раз, а 300 раз (в каждом узле - отдельно). Налицо дублирование информации. Чтобы устранить этот недостаток была введена сетевая модель представления данных.

2) Сетевые база данных — это база данных, в которой одна запись может участвовать в нескольких отношениях предок-потомок. Т.е. фактически, база данных представляет собой не дерево, а граф.



3) Реляционные базы данных. Вся информация представляется в виде таблиц и любые операции над данными — это операции над таблицами.

Таблицы состоят из строк и столбцов. Строки — это записи, а столбцы представляют структуру записи (каждый столбец имеет определенный тип данных и длину данных).

Строки в таблице не упорядочены — не существует первой или десятой строки. Однако поскольку на строки надо как-то ссылаться, то вводится понятие "*первичный ключ*".

Первичный ключ — это столбец, значения которого во всех строках разные. Используя первичный ключ можно однозначно сослаться на какую-либо строку таблицы. Первичный ключ может состоять и из нескольких столбцов (*составной первичный ключ*).

Некоторые СУБД требуют в явном виде указать первичный ключ таблицы, а некоторые позволяют пользователю не задавать для таблицы первичный ключ — в таком случае СУБД сама добавляет в таблицу столбец — первичный ключ, не отображаемый на экране.

Отношения предок-потомок в реляционных БД реализуются при помощи внешних ключей.

Внешний ключ — это столбец таблицы, значения которого совпадают со значениями первичного ключа некоторой другой таблицы.

Пример. Столбец "Ответственный" таблицы "Мероприятия" является внешним ключом для таблицы "Сотрудники" (первичный ключ — столбец "Табельный номер").

Сотрудники

Табельный_номер	Фамилия
…	
47	Иванов
52	Петров
53	Сидоров
…	

Мероприятия

Мероприятие	**Табельный_номер**
8 Марта	47
23 февраля	52
…	…

4) Объектно-реляционные базы данных появились в последнее время у значительного числа производителей СУБД (Oracle, Informix, PostgreSQL) и сочетают в себе реляционную модель данных с концепциями объектно-ориентированного программирования (полиморфизм, инкапсуляция, наследование).

2.3. По архитектуре.

По способу организации взаимодействия с базой данных через сеть, СУБД делят на:

1)СУБД с централизованной архитектурой.

2)СУБД с архитектурой **файл-сервер.**

3)СУБД с архитектурой **клиент-сервер.**

4)СУБД с трехуровневой архитектурой: **"тонкий клиент" — сервер приложений — сервер базы данных.**

1) СУБД с централизованной архитектурой.

СУБД и сама база данных размещается и функционирует на центральном компьютере, а пользователи получают доступ к базе данных при помощи обычных терминалов

(устройств ввода и отображения информации: на центральный компьютер передаются нажатия клавиш, в обратном направлении передаются данные, отображаемые непосредственно на мониторе пользователя).

2) СУБД с архитектурой файл-сервер.

База данных хранится на сервере, а копии СУБД устанавливаются на компьютерах пользователей. Файл базы данных, находящийся на сервере, совместно используется всеми пользователями одновременно.

Ярким примером такой архитектуры является СУБД **MS Access**: копии СУБД установлены на компьютере каждого пользователя, а сам файл базы данных находится на сервере в сетевой папке.

3) СУБД с архитектурой клиент-сервер.

База данных хранится на сервере, а СУБД подразделяется на две части: клиентскую и серверную.

Клиентская часть СУБД выполняется на стороне клиента и обеспечивает интерактивное взаимодействие с пользователем и формирование запросов к базе данных (на языке SQL).

Серверная часть работает на сервере и взаимодействует с базой данных, обеспечивая выполнение запросов клиентской части.

Большинство современных СУБД реализованы по архитектуре клиент-сервер.

4) СУБД с трехуровневой архитектурой ("тонкий клиент" — сервер приложений — сервер базы данных).

В функции **клиентской части** ("тонкий клиент") входит только интерактивное взаимодействие с пользователем, а вся деловая логика вынесена на сервер приложений, который собственно и обеспечивает формирование запросов к базе данных, передаваемых на выполнение серверу базы данных.

"Тонкий клиент" находится на компьютере пользователя и чаще всего представляет из себя Web-броузер (например, Internet Explorer).

Сервер приложений находится на сервере и может являться специализированной программой или обычным Web-сервером, вызывающим для обработки HTTP-запроса программу.

3. Функции СУБД.

3.1. Непосредственное управление данными во внешней памяти

В некоторых реализациях СУБД активно используются возможности существующих файловых систем, в других работа производится вплоть до уровня устройств внешней памяти.

Но в развитых СУБД пользователи в любом случае не обязаны знать, использует ли СУБД файловую систему, и если использует, то как организованы файлы. В частности, СУБД поддерживает собственную систему именования объектов БД.

3.2. Управление буферами оперативной памяти

СУБД обычно работают с БД значительного размера; по крайней мере этот размер обычно существенно больше доступного объема оперативной памяти.

Понятно, что если при обращении к любому элементу данных будет производиться обмен с внешней памятью, то вся система будет работать со скоростью устройства внешней памяти.

Практически единственным способом реального увеличения этой скорости является буферизация данных в оперативной памяти. В развитых СУБД поддерживается собственный набор буферов оперативной памяти с собственной дисциплиной замены буферов.

3.3. Управление транзакциями

Транзакция - это последовательность операций над БД, рассматриваемых СУБД как ***единое целое***. Либо транзакция успешно выполняется, и СУБД фиксирует изменения БД, произведенные этой транзакцией, во внешней памяти, либо ни одно из этих изменений никак не отражается на состоянии БД.

Иногда транзакцию определяют, как ***перевод БД из одного безопасного (целотностного) состояния в другое*.**

При выполнении параллельно выполняющихся транзакций каждый из пользователей может в принципе ощущать себя единственным пользователем СУБД.

3.4. Журнализация

Одним из основных требований к СУБД является надежность хранения данных во внешней памяти. Под надежностью хранения понимается то, что СУБД должна быть в состоянии восстановить последнее согласованное состояние БД после любого аппаратного или программного сбоя.

Наиболее распространенным методом реализации данной цели является ведение журнала изменений БД.

Журнал - это особая часть БД, недоступная пользователям СУБД и поддерживаемая с особой тщательностью (иногда поддерживаются две копии журнала, располагаемые на разных физических дисках), в которую поступают записи обо всех изменениях основной части БД.

3.5. Поддержка языков БД

Для работы с базами данных используются специальные языки, чаще всего в реляционных базах данных используется язык SQL.

4. Реляционные СУБД. Основные концепции и термины.

Реляционная база данных – это совокупность отношений, содержащих всю информацию, которая должна храниться в БД.

Однако пользователи могут воспринимать такую базу данных просто как совокупность таблиц. При этом для массового пользователя реляционных СУБД можно с успехом использовать неформальные эквиваленты этих понятий:

Отношение	**–**	**Таблица (иногда Файл),**
Кортеж	**–**	**Строка (иногда Запись),**
Атрибут	**–**	**Столбец, Поле.**