

Лекция. Системы управления базами данных.

Учебные вопросы:

- 1. Основные положения.**
- 2. Структурные элементы базы данных.**
- 3. Виды моделей данных.**
- 4. Основные характеристики СУБД.**
- 5. Обобщенная технология работы СУБД**

Современные информационные системы, основанные на концепции интеграции данных, характеризуются огромными объемами хранимых данных, сложной организацией, необходимостью удовлетворять разнообразные требования многочисленных пользователей.

1. Основные положения.

Общие положения

Цель любой информационной системы – обработка данных об объектах реального мира. В широком смысле слова база данных – это совокупность сведений о конкретных объектах реального мира в какой-либо предметной области. Под *предметной областью* принято понимать часть реального мира, подлежащего изучению для организации управления и в конечном счете автоматизации, например, предприятие, вуз и т.д.

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 1976 г.; Л/д № 16593, Петрова Анна Владимировна, дата рожд. 15 марта 1975 г.; № личн. дела 16693, д.р. 14.04.76, 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 личного дела, фамилия, имя, отчество).

Пример 2. После проведения несложной структуризации с информацией, указанной в примере (рис. 9.1), она будет выглядеть так, как это показано на рис. 9.2.

№ личного дела	Фамилия	Имя	Отчество	Дата рождения
16493	Сергеев	Петр	Михайлович	01.01.76
16593	Петрова	Анна	Владимировна	15.03.75
16693	Анохин	Андрей	Борисович	14.04.76

Рис. 2. Пример структурированных данных

Пользователями базы данных могут быть различные прикладные программы, программные комплексы, а также специалисты предметной области, выступающие в роли потребителей или источников данных, называемые *конечными пользователями*.

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.

Централизованный характер управления данными в базе данных предполагает необходимость существования некоторого лица (группы лиц), на которое возлагаются функции администрирования данными, хранимыми в базе.

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:

1. файл-сервер;
2. 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, с помощью которых поддерживается доступ пользовательской системы к централизованной базе данных. Файлы базы данных в соответствии с пользовательскими запросами передаются на рабочие станции, где в основном и производится обработка. При большой интенсивности доступа к одним и тем же данным производительность информационной системы падает. Пользователи могут создавать также на рабочих станциях локальные БД, которые используются ими монополично. Концепция файл-сервер условно отображена на рис. 3.

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. Концепция клиент-сервер условно изображена на рис. 9.4.



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



Рис. 4. Схема обработки информации в БД по принципу *клиент-сервер*

2. Структурные элементы базы данных

Понятие базы данных тесно связано с такими понятиями структурных элементов, как поле, запись, файл (таблица) (рис. 5).

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, причем будет определяться максимально возможным количеством символов;

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

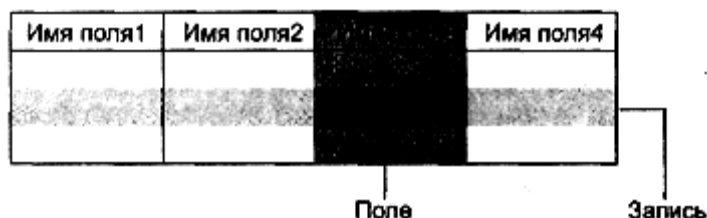


Рис. 5. Основные структурные элементы БД

Запись – совокупность логически связанных полей. Экземпляр записи – отдельная реализация записи, содержащая конкретные значения ее полей. **Файл (таблица)** – совокупность экземпляров записей одной структуры.

Описание логической структуры записи файла содержит последовательность расположения полей записи и их основные характеристики, как это показано на рис. 6.

Имя файла					
Поле		Признак ключа	Формат поля		
Имя (обозначение)	Полное наименование		Тип	Длина	Точность (для чисел)
имя 1					
...					
имя n					

Рис..6. Описание логической структуры записи файла

В структуре записи файла указываются поля, значения которых являются *ключами*: *первичными* (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
Номер	№ личного дела	*	Симв	5	
Фамилия	Фамилия студента		Симв	15	
Имя	Имя студента		Симв	10	
Отчество	Отчество студента		Симв	15	
Дата	Дата рождения		Дата	8	

Рис. 7. Описание логической структуры записи файла СТУДЕНТ

3. Виды моделей данных.

Общие положения

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 [1].

Рассмотрим три основных типа моделей данных: иерархическую, сетевую и реляционную.

Иерархическая модель данных

Иерархическая структура представляет совокупность элементов, связанных между собой по определенным правилам. Объекты, связанные иерархическими отношениями, образуют ориентированный граф (перевернутое дерево).

К основным понятиям иерархической структуры относятся: уровень, элемент (узел), связь. *Узел* – это совокупность атрибутов данных, описывающих некоторый объект. На схеме иерархического дерева узлы представляются вершинами графа. Каждый узел на более низком уровне связан только с одним узлом, находящимся на более высоком уровне. Иерархическое дерево имеет только одну вершину (корень дерева), не подчиненную никакой другой вершине и находящуюся на самом верхнем (первом) уровне. Зависимые (подчиненные) узлы находятся на втором, третьем и т.д. уровнях. Количество деревьев в базе данных определяется числом корневых записей.



Рис. 8. Пример иерархической структуры БД

Сетевая модель данных

В сетевой структуре при тех же основных понятиях (уровень, узел, связь) каждый элемент может быть связан с любым другим элементом.

Пример 4. Примером сложной сетевой структуры может служить структура базы данных, содержащей сведения о студентах, участвующих в научно-исследовательских работах (НИРС). Возможно участие одного студента в нескольких НИРС, а также участие нескольких студентов в разработке одной НИРС. Графическое изображение описанной в примере сетевой структуры, состоящей только из двух типов записей, показано на рис. 9. Единственное отношение представляет собой сложную связь между записями в обоих направлениях.



Рис. 9. Пример сетевой структуры БД

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.

3. Реляционная модель ориентирована на организацию данных в виде двумерных таблиц. Каждая *реляционная таблица* представляет собой двумерный массив.

Пример 9.4. Реляционной таблицей можно представить информацию о студентах, обучающихся в вузе (рис. 10).

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

Рис.10. Пример реляционной таблицы

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. 10, 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.

Пример 5. На рис.11 показан пример реляционной модели, построенной на основе отношений: СТУДЕНТ, СЕССИЯ, СТИПЕНДИЯ.

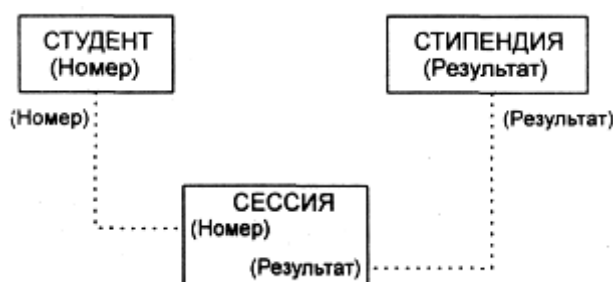


Рис. 11. Пример реляционной модели

СТУДЕНТ (*Номер*, Фамилия, Имя, Отчество, Пол, Дата рождения, Группа); СЕССИЯ (*Номер*, Оценка1, Оценка2, Оценка3, Оценка4, Результат); СТИПЕНДИЯ (*Результат*, Процент).

Таблицы СТУДЕНТ И СЕССИЯ имеют совпадающие ключи (Номер), что дает возможность легко организовать связь между ними. Таблица СЕССИЯ имеет

первичный ключ Номер и содержит внешний ключ Результат, который обеспечивает ее связь с таблицей СТИПЕНДИЯ.

4. Основные характеристики СУБД.

Производительность СУБД

Производительность СУБД оценивается:

временем выполнения запросов;

скоростью поиска информации в неиндексированных полях;

временем выполнения операций импортирования базы данных из других форматов;

скоростью создания индексов и выполнения таких массовых операций, как обновление, вставка, удаление данных;

максимальным числом параллельных обращений к данным в многопользовательском режиме;

временем генерации отчета.

На производительность СУБД оказывают влияние два фактора:

- СУБД, которые следят за соблюдением целостности данных, несут дополнительную нагрузку, которую не испытывают другие программы;

- производительность собственных прикладных программ сильно зависит от правильного проектирования и построения базы данных.

Самые быстрые программные изделия отнюдь не обладают самыми развитыми функциональными возможностями на уровне процессора СУБД.

Можно заключить, что самой быстрой СУБД является FoxPro 2.6, однако она не обладает средствами соблюдения целостности данных в отличие от более медленной СУБД Access 2.0.

Обеспечение целостности данных на уровне базы данных

Эта характеристика подразумевает наличие средств, позволяющих удостовериться, что информация в базе данных всегда остается корректной и полной. Должны быть установлены правила целостности, и они должны храниться вместе с базой данных и соблюдаться на глобальном уровне. *Целостность данных* должна обеспечиваться независимо от того, каким образом данные заносятся в память (в интерактивном режиме, посредством импорта или с помощью специальной программы).

К средствам обеспечения целостности данных на уровне СУБД относятся:

встроенные средства для назначения первичного ключа, в том числе средства для работы с типом полей с автоматическим приращением, когда СУБД самостоятельно присваивает новое уникальное значение;

средства поддержания ссылочной целостности, которые обеспечивают запись информации о связях таблиц и автоматически пресекают любую операцию, приводящую к нарушению ссылочной целостности.

Некоторые СУБД имеют хорошо разработанный процессор СУБД для реализации таких возможностей, как уникальность первичных ключей, ограничение (пресечение) операций и даже каскадное обновление и удаление информации. В таких системах проверка корректности, назначаемая полю или таблице, будет проводиться всегда после изменения данных, а не только во время ввода информации с помощью

экранной формы. Это свойство можно настраивать для каждого поля и для записи в целом, что позволяет контролировать не только значения отдельных полей, но и взаимосвязи между несколькими полями данной записи.

Access и Paradox for Windows гораздо ближе других СУБД соответствуют реляционной модели по надежности сохранения целостности данных на уровне базы данных; правила хранятся вместе с базой данных и автоматически соблюдаются.

СУБД dBASE IV и FoxPro 2.6 (DOS и WINDOWS) совсем не обладают средствами этого рода, и ввод в программу процедур, обеспечивающих выполнение правил целостности, возлагается на программиста.

Обеспечение безопасности

Некоторые СУБД предусматривают средства обеспечения *безопасности данных*. Такие средства обеспечивают выполнение следующих операций:

шифрование прикладных программ;

шифрование данных;

защиту паролем;

ограничение уровня доступа (к базе данных, к таблице, к словарю, для пользователя).

Самый высокий уровень безопасности данных реализован в СУБД dBASE IV. Администратор может назначать системе различные права доступа на уровне файла, поля, а также организовать автоматическое шифрование данных.

Хорошими характеристиками обеспечения безопасности отличается Access 2.0. Он предусматривает назначение паролей для индивидуальных пользователей или групп пользователей и присвоение различных прав доступа отдельно таблицам, запросам, отчетам, макрокомандам или новым объектам на уровне пользователя или группы.

Работа в многопользовательских средах

Практически все рассматриваемые СУБД предназначены для работы в *многопользовательских средах*, но обладают для этого различными возможностями.

Обработка данных в многопользовательских средах предполагает выполнение программным продуктом следующих функций:

- блокировку базы данных, файла, записи, поля;
- идентификацию станции, установившей блокировку;
- обновление информации после модификации;
- контроль за временем и повторение обращения;
- обработку транзакций (транзакция – последовательность операций пользователя над базой данных, которая сохраняет ее логическую целостность);
- работу с сетевыми системами (LAN Manager, NetWare, Unix).

Лучшими возможностями для работы в многопользовательских средах обладают СУБД Paradox for DOS 4.5, Access 2.0 и dBASE IV.

Импорт-экспорт

Эта характеристика отражает:

возможность обработки СУБД информации, подготовленной другими

программными средствами;

возможность использования другими программами данных, сформированных средствами рассматриваемой СУБД.

Особый интерес представляют следующие форматы файлов: ASCII-файлы, .DBF, .WK*, .XLS.

Все рассматриваемые здесь СУБД обладают хорошими возможностями импорта-экспорта данных.

5. Обобщенная технология работы СУБД

Общее представление об этапах технологии

Каждая конкретная СУБД имеет свои особенности, которые необходимо учитывать.

Однако имея представление о функциональных возможностях любой СУБД, можно представить обобщенную технологию работы пользователя в этой среде.

В качестве основных этапов обобщенной технологии работы с СУБД, которая схематично представлена на рис. 12, можно выделить следующие:

- создание структуры таблиц базы данных;
- ввод и редактирование данных в таблицах;
- обработка данных, содержащихся в таблицах;
- вывод информации из базы данных.

Рассмотрим выделенные этапы более подробно.

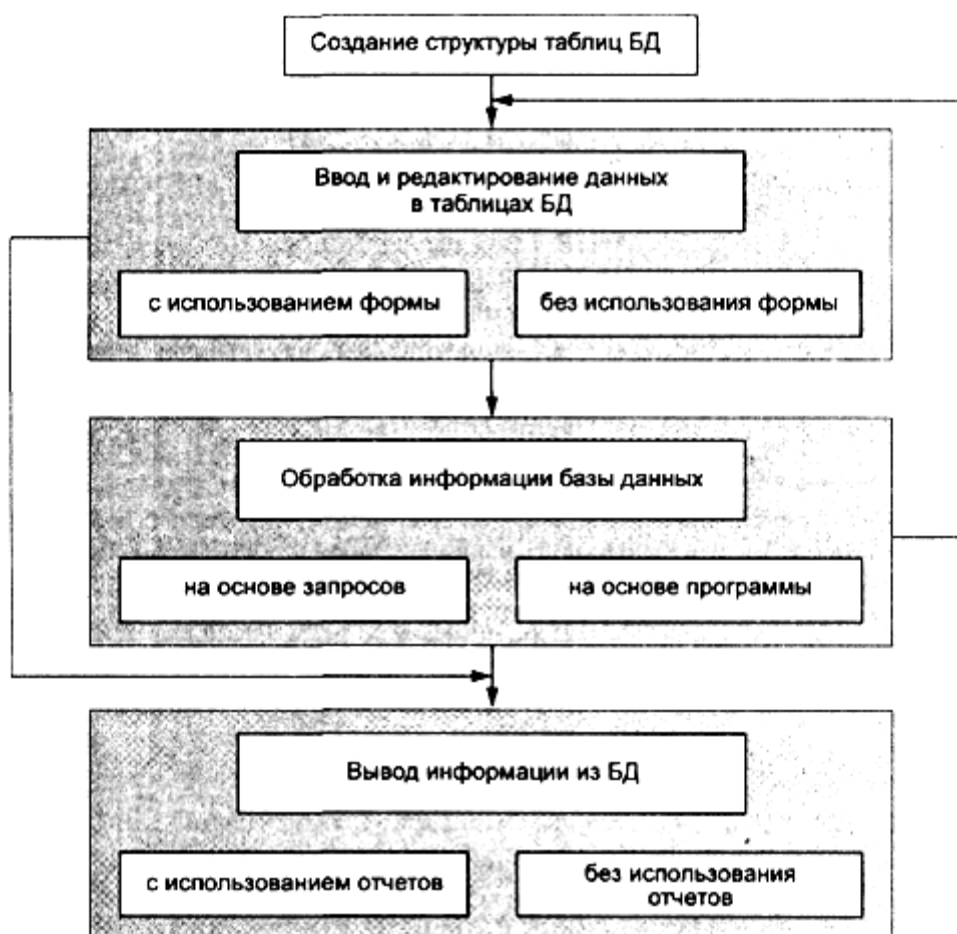


Рис. 12. Схема обобщенной технологии работы в СУБД

Создание структуры таблиц базы данных

При формировании новой таблицы базы данных работа с СУБД начинается с создания структуры таблицы. Этот процесс включает определение перечня полей, из которых состоит каждая запись таблицы, а также типов и размеров полей.

Практически все используемые СУБД хранят данные следующих типов: текстовый (символьный), числовой, календарный, логический, примечание. Некоторые СУБД формируют поля специального типа, содержащие уникальные номера записей и используемые для определения ключа.

СУБД, предназначенные для работы в Windows, могут формировать поля типа объекта OLE, которые используются для хранения рисунков, графиков, таблиц.

Если обрабатываемая база данных включает несколько взаимосвязанных таблиц, то необходимо определение ключевого поля в каждой таблице, а также полей, с помощью которых будет организована связь между таблицами.

Создание структуры таблицы не связано с заполнением таблиц данными, поэтому эти две операции можно разнести во времени.

Ввод и редактирование данных

Заполнение таблиц данными возможно как непосредственным вводом данных, так и в результате выполнения программ и запросов.

Практически все СУБД позволяют вводить и корректировать данные в таблицах двумя способами:

- с помощью предоставляемой по умолчанию стандартной формы в виде таблицы;
- с помощью *экранных форм*, специально созданных для этого пользователем.

Обработка данных, содержащихся в таблицах

Обрабатывать информацию, содержащуюся в таблицах базы данных, можно путем использования запросов или в процессе выполнения специально разработанной программы.

Конечный пользователь получает при работе с СУБД такое удобное средство обработки информации, как запросы. *Запрос* представляет собой инструкцию на отбор записей.

Большинство СУБД разрешают использовать запросы следующих типов:

- запрос-выборка, предназначенный для отбора данных, хранящихся в таблицах, и не изменяющий эти данные;
- запрос-изменение, предназначенный для изменения или перемещения данных; к этому типу запросов относятся: запрос на добавление записей, запрос на удаление записей, запрос на создание таблицы, запрос на обновление;
- запрос с параметром, позволяющий определить одно или несколько условий отбора во время выполнения запроса.

Самым распространенным типом запроса является запрос на выборку.

Результатом выполнения запроса является таблица с временным набором данных (динамический набор). Записи динамического набора могут включать поля из одной или нескольких таблиц базы данных. На основе запроса можно построить отчет или форму.

Вывод информации из базы данных

Практически любая СУБД позволяет вывести на экран и принтер информацию, содержащуюся в базе данных, из режимов таблицы или формы. Такой порядок вывода данных может использоваться только как черновой вариант, так как позволяет выводить данные только точно в таком же виде, в каком они содержатся в таблице или форме.

Каждый пользователь, работающий с СУБД, имеет возможность использования специальных средств построения *отчетов* для вывода данных. Используя специальные средства создания отчетов, пользователь получает следующие дополнительные возможности вывода данных:

- включать в отчет выборочную информацию из таблиц базы данных;
- добавлять информацию, не содержащуюся в базе данных;
- при необходимости выводить итоговые данные на основе информации базы данных;
- располагать выводимую в отчете информацию в любом, удобном для пользователя виде (вертикальное или горизонтальное расположение полей);
- включать в отчет информацию из разных связанных таблиц базы данных.