ГОСУДАРСТВЕННЫЙ КОМИТЕТ СВЯЗИ,
ИНФОРМАТИЗАЦИИ И ТЕЛЕКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ
РЕСПУБЛИКИ УЗБЕКИСТАН

ФЕРГАНСКИЙ ФИЛИАЛ
ТАШКЕНТСКОГО УНИВЕРСИТЕТА ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ

Факультет «Компьютер инжиниринг»

Кафедра «Прикладная информатика»

«Системы управления базами данных»

Конспект лекций для бакалавров направлений образования:

5811100– “Сервис предприятий”

Фергана 2017

Лекция 1.

Введение в предмет «Системы управления базами данных». База данных, модель базы данных. Основы работы с СУБД.

План:

1. Понятие база данных. СУБД- системы управления базами данных.
2. Функции СУБД. Непосредственное управление данными во внешней памяти.
3. Управление буферами оперативной памяти.
4. Управление транзакциями.
5. Журнализация
6. Поддержка языков БД

Предметом курса являются системы управления базами данных (СУБД). Это очень важная тема, без основательного знакомства с которой в наше время невозможно быть не только квалифицированным программистом, но даже и грамотным пользователем компьютеров.

База данных (БД) — организованная совокупность данных, предназначенная для длительного хранения во внешней памяти ЭВМ, постоянного обновления и использования.
База данных может состоять из одной таблицы — однотабличная БД, или из множества взаимосвязанных таблиц — многотабличная БД.

Структурными составляющими таблицы являются записи и поля.

	Поле 1 Фамилия	Поле 2 Имя	Поле 3 Отчество	Поле 4 Дата рождения
Запись 1	Алиев	Карим	Эргашевич	01.01.1987
Запись 2	Закиров	Анвар	Рашидович	19.07.1989
Запись 3	….	…	….	…
Запись 4	…	…	…	…

Системы управления базами данных - одна из фундаментальных составляющих компьютерного обеспечения информационных процессов, являющаяся основой для построения большинства современных информационных систем. Главной функцией СУБД является эффективное хранение и предоставление данных в интересах конкретных прикладных задач.

В настоящее время существует несколько видов СУБД. Наиболее известными и популярными СУБД являются Access, FoxPro и Paradox, Oracle, SQL, Cashe. Каждая из этих систем обладает своими достоинствами и недостатками. Более точно, к числу функций СУБД принято относить следующие:

Непосредственное управление данными во внешней памяти

Эта функция включает обеспечение необходимых структур внешней памяти как для хранения данных, непосредственно входящих в БД, так и для служебных целей, например, для убыстрения доступа к данным в некоторых случаях (обычно для этого используются индексы). В некоторых реализациях СУБД активно используются возможности существующих файловых систем, в других работа производится вплоть до уровня устройств внешней памяти.

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. При этом, даже если операционная система производит общесистемную буферизацию (как в случае ОС UNIX), этого недостаточно для целей СУБД, которая располагает гораздо большей информацией о полезности буферизации той или иной части БД. Поэтому в развитых СУБД поддерживается собственный набор буферов оперативной памяти с собственной дисциплиной замены буферов.Заметим, что существует отдельное направление СУБД, которое ориентировано на постоянное присутствие в оперативной памяти всей БД. Это направление основывается на предположении, что в будущем объем оперативной памяти компьютеров будет настолько велик, что позволит не беспокоиться о буферизации. Пока эти работы находятся в стадии исследований.

Управление транзакциями

Транзакция - это последовательность операций над БД, рассматриваемых СУБД как единое целое. Либо транзакция успешно выполняется, и СУБД фиксирует (COMMIT) изменения БД, произведенные этой транзакцией, во внешней памяти, либо ни одно из этих изменений никак не отражается на состоянии БД. Понятие транзакции необходимо для поддержания логической целостности БД. То свойство, что каждая транзакция начинается при целостном состоянии БД и оставляет это состояние целостным после своего завершения, делает очень удобным использование понятия транзакции как единицы активности пользователя по отношению к БД. При соответствующем управлении параллельно выполняющимися транзакциями со стороны СУБД каждый из пользователей может в принципе ощущать себя единственным пользователем СУБД (на самом деле, это несколько идеализированное представление, поскольку в некоторых случаях пользователи многопользовательских СУБД могут ощутить присутствие своих коллег).

Журнализация

Одним из основных требований к СУБД является надежность хранения данных во внешней памяти. Под надежностью хранения понимается то, что СУБД должна уметь восстановить последнее согласованное состояние БД после любого аппаратного или программного сбоя. Обычно рассматриваются два возможных вида аппаратных сбоев: так называемые мягкие сбои, которые можно трактовать как внезапную остановку работы компьютера (например, аварийное выключение питания), и жесткие сбои, характеризуемые потерей информации на носителях внешней памяти. Примерами программных сбоев могут быть: аварийное завершение работы СУБД (по причине ошибки в программе или в результате некоторого аппаратного сбоя) или аварийное завершение пользовательской программы, в результате чего некоторая транзакция остается незавершенной. Первую ситуацию можно рассматривать как особый вид мягкого аппаратного сбоя; при возникновении последней требуется ликвидировать последствия только одной транзакции.Понятно, что в любом случае для восстановления БД нужно располагать некоторой дополнительной информацией. Другими словами, поддержание надежности хранения данных в БД требует избыточности хранения данных, причем та часть данных, которая используется для восстановления, должна храниться особо надежно. Наиболее распространенным методом поддержания такой избыточной информации является ведение журнала измененийБД.

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. В разных СУБД

изменения БД журнализуются на разных уровнях: иногда запись в журнале соответствует некоторой логической операции изменения БД (например, операции удаления строки из таблицы реляционной БД), иногда - минимальной внутренней операции модификации страницы внешней памяти; в некоторых системах одновременно используются оба подхода.

Во всех случаях придерживаются стратегии "упреждающей" записи в журнал (так называемого протокола Write Ahead Log - WAL). Грубо говоря, эта стратегия заключается в том, что запись об изменении любого объекта БД должна попасть во внешнюю память журнала раньше, чем измененный объект попадет во внешнюю память основной части БД. Известно, что если в СУБД корректно соблюдается протокол WAL, то с помощью журнала можно решить все проблемы восстановления БД после любого сбоя. Самая простая ситуация восстановления - индивидуальный откат транзакции.

При мягком сбое во внешней памяти основной части БД могут находиться объекты, модифицированные транзакциями, не закончившимися к моменту сбоя, и могут отсутствовать объекты, модифицированные транзакциями, которые к моменту сбоя успешно завершились (по причине использования буферов оперативной памяти, содержимое которых при мягком сбое пропадает). При соблюдении протокола WAL во внешней памяти журнала должны гарантированно находиться записи, относящиеся к операциям модификации обоих видов объектов.

Для восстановления БД после жесткого сбоя используют журнал и архивную копию БД. Грубо говоря, архивная копия - это полная копия БД к моменту начала заполнения журнала (имеется много вариантов более гибкой трактовки смысла архивной копии). Конечно, для нормального восстановления БД после жесткого сбоя необходимо, чтобы журнал не пропал. Как уже отмечалось, к сохранности журнала во внешней памяти в СУБД предъявляются особо повышенные требования. Тогда восстановление БД состоит в том, что исходя из архивной копии по журналу воспроизводится работа всех транзакций, которые закончились к моменту сбоя. В принципе, можно даже воспроизвести работу незавершенных транзакций и продолжить их работу после завершения восстановления. Однако в реальных системах это обычно не делается, поскольку процесс восстановления после жесткого сбоя является достаточно длительным.

Поддержка языков БД

Для работы с базами данных используются специальные языки, в целом называемые языками баз данных. В ранних СУБД поддерживалось несколько специализированных по своим функциям языков. Чаще всего выделялись два языка - язык определения схемы БД (SDL - Schema Definition Language) и язык манипулирования данными (DML - Data Manipulation Language). SDL

служил главным образом для определения логической структуры БД, т.е. той структуры БД, какой она представляется пользователям. DML содержал набор операторов манипулирования данными, т.е. операторов, позволяющих заносить данные в БД, удалять, модифицировать или выбирать существующие данные. В современных СУБД обычно поддерживается единый интегрированный язык, содержащий все необходимые средства для работы с БД, начиная от ее создания, и обеспечивающий базовый пользовательский интерфейс с базами данных. Стандартным языком наиболее распространенных в настоящее время реляционных СУБД является язык SQL (Structured Query Language). В нескольких лекциях этого курса язык SQL будет рассматриваться достаточно подробно, а пока мы перечислим основные функции реляционной СУБД, поддерживаемые на "языковом" уровне (т.е. функции, поддерживаемые при реализации интерфейса SQL). Прежде всего, язык SQL сочетает средства SDL и DML, т.е. позволяет определять схему реляционной БД и манипулировать данными. При этом именование объектов БД (для реляционной БД - именование таблиц и их столбцов) поддерживается на языковом уровне в том смысле, что компилятор языка SQL производит преобразование имен объектов в их внутренние идентификаторы на основании специально поддерживаемых служебных таблиц-каталогов. Внутренняя часть СУБД (ядро) вообще не работает с именами таблиц и их столбцов.

Язык SQL содержит специальные средства определения ограничений целостности БД. Опять же, ограничения целостности хранятся в специальных таблицах-каталогах, и обеспечение контроля целостности БД производится на языковом уровне, т.е. при компиляции операторов модификации БД компилятор SQL на основании имеющихся в БД ограничений целостности генерирует соответствующий программный код. Специальные операторы языка SQL позволяют определять так называемые представления БД, фактически являющиеся хранимыми в БД запросами (результатом любого запроса к реляционной БД является таблица) с именованными столбцами. Для пользователя представление является такой же таблицей, как любая базовая таблица, хранимая в БД, но с помощью представлений можно ограничить или наоборот расширить видимость БД для конкретного пользователя. Поддержание представлений производится также на языковом уровне.

Контрольные вопросы:

1. Что такое база данных?
2. Объясните понятие СУБД.
3. Назовите основные функции СУБД.
4. Что означает понятие журнализация?
5. Что такое транзакция?
6. Перечислите несколько видов современных СУБД.

Лекция 2

Общие сведения о MY SQL СУБД ORACLE. Сведения о языке PL/SQL.

План:

СУБД Oracle. Архитектура ORACLE.

Я зык структурированных запросов SQL.

3. Сведения о языке PL/SQL. Функциональность и особенности PL/SQL.

СУБД Oracle

На сегодняшний день СУБД Oracle – это интегрированное программное обеспечение для сети распределенных вычислений Grid. Для реализации сложных комплексных решений имеется широкий набор программных продуктов, который можно разбить на несколько разделов по их функциональному назначению.

База данных – это набор данных. Oracle позволяет сохранять данные и получать к ним доступ в соответствии с моделью, называемой реляционной. В связи с этим Oracle называют системой управления реляционными базами данных (РСУБД). Чаще всего под базой данных подразумевают не только физические данные, но также и комбинацию физических объектов, объектов памяти и процессов.

Архитектура ORACLE

Архитектура Oracle включает в себя три основных компонента:

• Файлы. Будут рассмотрены пять видов файлов, образующих базу данных и поддерживающих экземпляр. Это файлы параметров, сообщений, данных, временных данных и журналов повторного выполнения.

• Структуры памяти, в частности системная глобальная область (System Global Area – SGA).

• Физические процессы или потоки.

На рисунке 1 схематично представлена общая архитектура СУБД Oracle:

1. Клиентское приложение взаимодействует с серверным процессом в режиме запрос-ответ.

2. Серверный процесс взаимодействует с базой данных и выполняет запрошенные клиентом операции. При обработке запроса данные кешируются в специальных областях памяти экземпляра.

3. Фоновые процессы экземпляра обеспечивают функционирование базы данных (на рисунке не указаны, т. к. являются составной частью экземпляра).

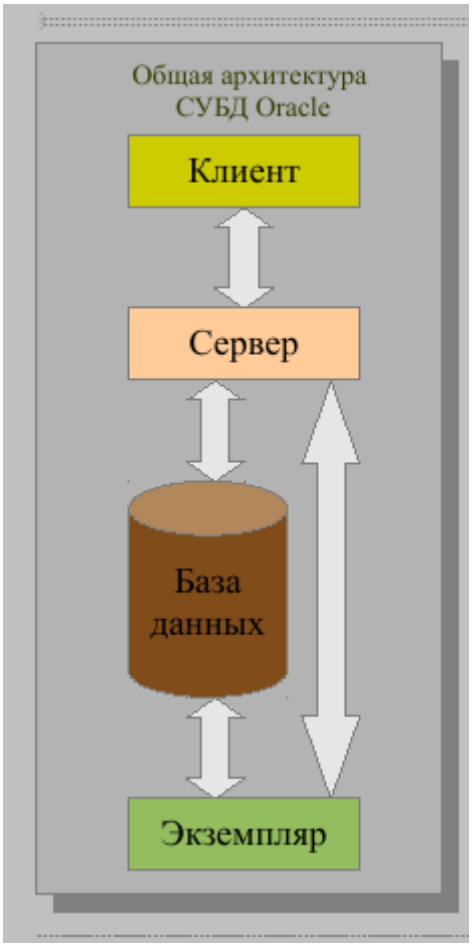


Рис.1. Общая архитектура СУБД Oracle

Язык SQL

SQL (англ. Structured Query Language – «язык структурированных запросов») – универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Язык SQL был разработан фирмой IBM в конце 1970-х годов и был принят Американским национальным институтом стандартов (ANSI) в качестве национального стандарта США в 1992 году .

Язык SQL ориентирован на текст. Он был разработан задолго до появления графических интерфейсов пользователя, так что для работы с ним требуется лишь текстовый редактор. Разумеется, в таких СУБД как Oracle имеются графические средства для выполнения многих из тех задач, которые ранее могли быть выполнены только с помощью SQL. Но не все из того, что позволяет делать SQL, можно осуществить с помощью графических средств; более того, в ряде случаев, например для динамической генерации операторов SQL в программном коде, SQL использовать необходимо.

С помощью SQL можно определять структуры базы данных, а также запрашивать и обновлять информацию в базе данных. Совокупность команд, служащих для определения данных, называют иногда языком определения данных (data definition language, DDL), а совокупность команд для обновления и запроса данных — языком манипулирования данными (data manipulation language, DML).

По типу работы с данными операторы SQL делятся на 4 основные группы: DDL, DML, DCL, TCL.

Data Definition Language (DDL)

● Команды, которые работают со словарем базы данных.

Основные команды:

CREATE, ALTER, DROP, GRANT, REVOKE.

Data Manipulation Language (DML)

● Команды, манипулирующие с данными в таблицах.

Основные команды:

SELECT, INSERT, DELETE, UPDATE.

Transaction Control Language (TCL)

● Команды управления транзакциями.

Основные команды:

COMMIT, ROLLBACK, SAVEPOINT.

Data Control Language (DCL)

● Команды, определяющие доступ к данным.

Основные команды:

GRANT, REVOKE, DENY.

Сведения о языке PL/SQL.

PL/SQL (англ. Procedural Language/SQL) — язык программирования, процедурное расширение языка SQL, разработанное корпорацией Oracle. PL/SQL разработан на основе структурно-модульного языка программирования Ада.

Функциональность и особенности PL/SQL:

● Поддерживает переменные, операторы, массивы, курсоры.

● Начиная с версии 8 доступна объектно-ориентированная модель.

● Имеет строгие правила области видимости переменных.

● Поддерживает пакеты, триггеры, параметризованные вызовы процедур и функций.

● Хранимые процедуры могут быть написаны на языке Java.

● Реализован механизм обработки исключений (Exception Handler).

● Платформо-независимый.

● Поддерживает интерфейс для работы с языками высокого уровня C/C++. Разработка внешних процедур на C/C++.

Структура программы на PL/SQL

Модельным прототипом для создания языка PL/SQL послужил язык программирования ADA, поэтому он обладает набором средств, характерных для любого современного языка программирования. Всякая программа на PL/SQL состоит из трех блоков: блока описаний, блока исполняемого кода и блока обработки исключительных ситуаций. Блок исполняемого кода может быть структурирован с помощью операторных скобок BEGIN … END.

Синтаксически программа на PL/SQL оформляется следующим образом:

```
DECLALE
  Описание переменных и констант
BEGIN
```

Операторы
EXCEPTION
Операторы
END;

Перед блоком DECLALE могут располагаться команды установки переменных окружения. В блоке DECLALE описываются константы, переменные и определенные пользователем типы данных. Первый оператор BEGIN отмечает начало тела основной программы. В тело программы могут быть включены другие блоки, ограниченные операторными скобками. Блок EXCEPTION определяет фрагменты программного кода для обработки исключительных ситуаций в программе. Последний оператор END указывает конец тела программы. В любые части программы могут быть включены комментарии, т.е. текст, который начинается с символов -- и продолжается до конца текущей строки. Строка, начинающаяся с ключевого слова REM, также рассматривается как комментарий.

Константы, переменные и типы в PL/SQL

Переменная или константа определяется своим именем, которое является допустимым именем в СУБД Oracle. Любая константа или переменная должна иметь один из допустимых в СУБД Oracle типов. Константа идентифицируется ключевым словом CONSTANT и отличается от переменной тем, что попытка изменить ее значение в программе приведет к ошибке. Для определения констант и переменных используется следующий синтаксис:

Имя [CONSTANT] тип данных [:= значение];

Пример. Рассмотрим пример простой программы, которая вычисляет значение синуса двух углов, кратных p.

```
-- переменные
окружения
set serveroutput on;
set echo on;
DECLARE
Pi CONSTANT real :=3.14;
x real :=1;
BEGIN
DBMS_OUTPUT.PUT_LINE ('y ='|| sin(Pi*x));
x:=x+1;
DBMS_OUTPUT.PUT_LINE ('y ='|| sin(Pi*x));
END;
```

/Установка переменных окружения определяет режим вывода на терминал пользователя. Системная процедура DBMS_OUTPUT.PUT_LINE обеспечивает вывод данных на терминал. Символ / указывает на завершение текста программы и является командой к выполнению программы.

Контрольные вопросы:

1.Что представляет собой СУБД Oracle ?
2.Как построена архитектура Oracle ?
3.Для чего предназначен язык SQL ?
4.На какие группы операторов делится SQL по типу работы с данными ?
5.Общие сведения о языке PL/SQL.

Лекция 3

База данных. База данных для предметной области.

План:

1. Понятие предметная область. Модель предметной области.
2. Логическая и физическая модель данных.
3. Адекватность базы данных предметной области.
4. Легкость разработки и сопровождения базы данных.

При разработке базы данных обычно выделяется несколько уровней моделирования, при помощи которых происходит переход от предметной области к конкретной реализации базы данных средствами конкретной СУБД. Можно выделить следующие уровни:

- Сама предметная область
- Модель предметной области
- Логическая модель данных
- Физическая модель данных
- Собственно база данных и приложения

Предметная область - это часть реального мира, данные о которой мы хотим отразить в базе данных. Например, в качестве предметной области можно выбрать бухгалтерию какого-либо предприятия, отдел кадров, банк, магазин и т.д. Предметная область бесконечна и содержит как существенно важные понятия и данные, так и малозначащие или вообще не значащие данные. Так, если в качестве предметной области выбрать учет товаров на складе, то понятия "накладная" и "счет-фактура" являются существенно важными понятиями, а то, что сотрудница, принимающая накладные, имеет двоих детей - это для учета товаров неважно. Однако, с точки зрения отдела кадров данные о наличии детей являются существенно важными. Таким образом, важность данных зависит от выбора предметной области.

Модель предметной области. Модель предметной области - это наши знания о предметной области. Знания могут быть как в виде неформальных знаний в мозгу эксперта, так и выражены формально при помощи каких-либо средств. В качестве таких средств могут выступать текстовые описания

предметной области, наборы должностных инструкций, правила ведения дел в компании и т.п. Опыт показывает, что текстовый способ представления модели предметной области крайне неэффективен. Гораздо более информативными и полезными при разработке баз данных являются описания предметной области, выполненные при помощи специализированных графических нотаций. Имеется большое количество методик описания предметной области. Из наиболее известных можно назвать методику структурного анализа SADT и основанную на нем IDEF0, диаграммы потоков данных Гейна-Сарсона, методику объектно-ориентированного анализа UML, и др. Модель предметной области описывает скорее процессы, происходящие в предметной области и данные, используемые этими процессами. От того, насколько правильно смоделирована предметная область, зависит успех дальнейшей разработки приложений.

Логическая модель данных. На следующем, более низком уровне находится логическая модель данных предметной области. Логическая модель описывает понятия предметной области, их взаимосвязь, а также ограничения на данные, налагаемые предметной областью. Примеры понятий - "сотрудник", "отдел", "проект", "зарплата". Примеры взаимосвязей между понятиями - "сотрудник числится ровно в одном отделе", "сотрудник может выполнять несколько проектов", "над одним проектом может работать несколько сотрудников". Примеры ограничений - "возраст сотрудника не менее 16 и не более 60 лет".

Логическая модель данных является начальным прототипом будущей базы данных. Логическая модель строится в терминах информационных единиц, но без привязки к конкретной СУБД. Более того, логическая модель данных необязательно должна быть выражена средствами именно реляционной модели данных. Основным средством разработки логической модели данных в настоящий момент являются различные варианты ER-диаграмм (Entity-Relationship, диаграммы сущность-связь). Одну и ту же ER-модель можно преобразовать как в реляционную модель данных, так и в модель данных для иерархических и сетевых СУБД, или в постреляционную модель данных. Однако, т.к. мы рассматриваем именно реляционные СУБД, то можно считать, что логическая модель данных для нас формулируется в терминах реляционной модели данных.

Решения, принятые на предыдущем уровне, при разработке модели предметной области, определяют некоторые границы, в пределах которых можно развивать логическую модель данных, в пределах же этих границ можно принимать различные решения. Например, модель предметной области складского учета содержит понятия "склад", "накладная", "товар". При разработке соответствующей реляционной модели эти термины обязательно должны быть использованы, но различных способов реализации

тут много - можно создать одно отношение, в котором будут присутствовать в качестве атрибутов "склад", "накладная", "товар", а можно создать три отдельных отношения, по одному на каждое понятие.

При разработке логической модели данных возникают вопросы: хорошо ли спроектированы отношения? Правильно ли они отражают модель предметной области, а следовательно и саму предметную область?

Физическая модель данных. На еще более низком уровне находится физическая модель данных. Физическая модель данных описывает данные средствами конкретной СУБД. Мы будем считать, что физическая модель данных реализована средствами именно реляционной СУБД, хотя, как уже сказано выше, это необязательно. Отношения, разработанные на стадии формирования логической модели данных, преобразуются в таблицы, атрибуты становятся столбцами таблиц, для ключевых атрибутов создаются уникальные индексы, домены преображаются в типы данных, принятые в конкретной СУБД.

Ограничения, имеющиеся в логической модели данных, реализуются различными средствами СУБД, например, при помощи индексов, декларативных ограничений целостности, триггеров, хранимых процедур. При этом опять-таки решения, принятые на уровне логического моделирования определяют некоторые границы, в пределах которых можно развивать физическую модель данных. Точно также, в пределах этих границ можно принимать различные решения. Например, отношения, содержащиеся в логической модели данных, должны быть преобразованы в таблицы, но для каждой таблицы можно дополнительно объявить различные индексы, повышающие скорость обращения к данным. Многое тут зависит от конкретной СУБД.

При разработке физической модели данных возникают вопросы: хорошо ли спроектированы таблицы? Правильно ли выбраны индексы? Насколько много программного кода в виде триггеров и хранимых процедур необходимо разработать для поддержания целостности данных?

Собственно база данных и приложения. И, наконец, как результат предыдущих этапов появляется собственно сама база данных. База данных реализована на конкретной программно-аппаратной основе, и выбор этой основы позволяет существенно повысить скорость работы с базой данных. Например, можно выбирать различные типы компьютеров, менять количество процессоров, объем оперативной памяти, дисковые подсистемы и т.п. Очень большое значение имеет также настройка СУБД в пределах выбранной программно-аппаратной платформы.

Но опять решения, принятые на предыдущем уровне - уровне физического проектирования, определяют границы, в пределах которых

можно принимать решения по выбору программно-аппаратной платформы и настройки СУБД.

Таким образом ясно, что решения, принятые на каждом этапе моделирования и разработки базы данных, будут сказываться на дальнейших этапах. Поэтому особую роль играет принятие правильных решений на ранних этапах моделирования.

Для того чтобы оценить качество принимаемых решений на уровне логической модели данных, необходимо сформулировать некоторые критерии качества в терминах физической модели и конкретной реализации и посмотреть, как различные решения, принятые в процессе логического моделирования, влияют на качество физической модели и на скорость работы базы данных.

Критерии, которые являются безусловно важными с точки зрения получения качественной базы данных:

- Адекватность базы данных предметной области
- Легкость разработки и сопровождения базы данных

Адекватность базы данных предметной области

База данных должна адекватно отражать предметную область. Это означает, что должны выполняться следующие условия:

1. Состояние базы данных в каждый момент времени должно соответствовать состоянию предметной области.
2. Изменение состояния предметной области должно приводить к соответствующему изменению состояния базы данных
3. Ограничения предметной области, отраженные в модели предметной области, должны некоторым образом отражаться и учитываться базе данных.

Легкость разработки и сопровождения базы данных

Практически любая база данных, за исключением совершенно элементарных, содержит некоторое количество программного кода в виде триггеров и хранимых процедур.

Хранимые процедуры - это процедуры и функции, хранящиеся непосредственно в базе данных в откомпилированном виде и которые могут запускаться пользователями или приложениями, работающими с базой данных. Хранимые процедуры обычно пишутся либо на специальном процедурном расширении языка SQL (например, PL/SQL для ORACLE или

Transact-SQL для MS SQL Server), или на некотором универсальном языке программирования, например, C++, с включением в код операторов SQL в соответствии со специальными правилами такого включения. Основное назначение хранимых процедур - реализация бизнес-процессов предметной области.

Триггеры - это хранимые процедуры, связанные с некоторыми событиями, происходящими во время работы базы данных. В качестве таких событий выступают операции вставки, обновления и удаления строк таблиц. Если в базе данных определен некоторый триггер, то он запускается автоматически всегда при возникновении события, с которым этот триггер связан. Очень важным является то, что пользователь не может обойти триггер. Триггер срабатывает независимо от того, кто из пользователей и каким способом инициировал событие, вызвавшее запуск триггера. Таким образом, основное назначение триггеров - автоматическая поддержка целостности базы данных. Триггеры могут быть как достаточно простыми, например, поддерживающими ссылочную целостность, так и довольно сложными, реализующими какие-либо сложные ограничения предметной области или сложные действия, которые должны произойти при наступлении некоторых событий. Например, с операцией вставки нового товара в накладную может быть связан триггер, который выполняет следующие действия - проверяет, есть ли необходимое количество товара, при наличии товара добавляет его в накладную и уменьшает данные о наличии товара на складе, при отсутствии товара формирует заказ на поставку недостающего товара и тут же посылает заказ по электронной почте поставщику.

Очевидно, что чем больше программного кода в виде триггеров и хранимых процедур содержит база данных, тем сложнее ее разработка и дальнейшее сопровождение.

Контрольные вопросы:

1. Что представляет собой понятие предметная область?
2. Что означает модель предметной области?
3. Что такое логическая модель данных и каким образом она организуется?
4. Что такое физическая модель данных и каким образом она организуется?
5. Объясните адекватность базы данных предметной области.
6. Каковы функции хранимых процедур и триггеров?

Лекция 4

Автоматизированные информационные системы. Система управления базой данных. Классификация баз данных.

План:

1. Автоматизированная информационная система (АИС).
2. Структура АИС.
3. Общие характеристики ранних СУБД.
4. Классификация баз данных по структуре хранимой информации.

Под системой понимают любой объект, который одновременно рассматривается и как единое целое, и как объединенная в интересах достижения поставленных целей совокупность разнородных элементов. Системы значительно отличаются между собой как по составу, так и по главным целям. Понятие «система» широко распространено и имеет множество смысловых значений. Чаще всего оно используется применительно к набору технических средств и программ. Системой может называться аппаратная часть компьютера. Системой может также считаться множество программ для решения конкретных прикладных задач, дополненных процедурами ведения документации и управления расчетами.Добавление к понятию «система» слова «информационная» отражает цель ее создания и функционирования. Информационные системы обеспечивают сбор, хранение, обработку, поиск, выдачу информации, необходимой в процессе принятия решений задач из любой области. Они помогают анализировать проблемы и создавать новые продукты.

Информационная система - взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели1.

Современное понимание информационной системы предполагает использование в качестве основного технического средства переработки информации персонального компьютера. В крупных организациях наряду с персональным компьютером в состав технической базы информационной системы может входить суперЭВМ. Кроме того, техническое воплощение информационной системы само по себе ничего не будет значить, если не учтена роль человека, для которого предназначена производимая информация и без которого невозможно ее получение и представление, поэтому

Автоматизированная информационная система (АИС) - это человеко-машинная система, обеспечивающая автоматизированную подготовку, поиск

и обработку информации в рамках интегрированных сетевых, компьютерных и коммуникационных технологий для оптимизации экономической и другой деятельности в различных сферах управления2.

На этой основе создаются различные автоматические и автоматизированные системы управления технологическими процессами. Типичным примером таких систем может служить в связи - автоматическая коммутационная станция. В этой системе управление осуществляется с помощью технических устройств типа процессоров или других более простых приборов. Человек-оператор не входит в контур управления, замыкающий связи объекта и органа управления, а лишь следит за ходом технологического процесса и по мере необходимости (например, в случае сбоя) вмешивается. Иначе обстоит дело с автоматизированной системой управления производственным процессом. В АС производственными процессами и объект и орган управления представляет собой единую человеко-машинную систему, человек обязательно входит в контур управления. По определению АС - это человеко-машинная система, предназначенная для сбора и обработки информации, необходимой для управления производственным процессом, то есть управления коллективами людей. Иначе говоря, успех функционирования таких систем во многом зависит от свойств и особенностей жизнедеятельности человеческого фактора. Без человека система АС производством самостоятельно не может работать, так как человек формирует задачи, разрабатывает все виды обеспечивающих подсистем, выбирает из выданных ЭВМ вариантов решений наиболее рациональный. И, разумеется, человек, что очень важно, в конечном счете юридически отвечает за результаты реализации принятых им решений. Как видим, роль человека огромна и не заменима. Человек организует программу подготовительных мероприятий перед созданием АС, следовательно, требуется помимо всего прочего специальное организационное и правовое обеспечение.

Структуру АИС составляет совокупность отдельных ее частей, называемых подсистемами. Подсистема - это часть системы, выделенная по какому-либо признаку.

АС состоит из двух подсистем: функциональной и обеспечивающей. Функциональная часть АС включает в себя ряд подсистем, охватывающих решение конкретных задач планирования, контроля, учета, анализа и регулирования деятельности управляемых объектов. В ходе аналитического обследования могут быть выделены различные подсистемы, набор которых зависит от вида предприятия, его специфики, уровня управления и других факторов. Для нормальной деятельности функциональной части АС в ее

состав входят подсистемы обеспечивающей части АС (так называемые обеспечивающие подсистемы).

Общие характеристики ранних СУБД

Эти системы (системы, основанные на инвертированных списках, иерархические и сетевые системы) использовались в течение многих лет, дольше, чем используется какая-либо из реляционных СУБД. На самом деле некоторые из ранних систем используются даже в наше время, накоплены громадные базы данных, и одной из актуальных проблем информационных систем является использование этих систем совместно с современными системами. .

Все ранние системы не основывались на каких-либо абстрактных моделях. Понятие модели данных фактически вошло в обиход специалистов в области БД только вместе с реляционным подходом. .

В ранних системах доступ к БД производился на уровне записей. Интерактивный доступ к БД поддерживался только путем создания соответствующих прикладных программ с собственным интерфейсом. Это заставляло пользователя самого производить всю оптимизацию доступа к БД без какой-либо поддержки системы. .

После появления реляционных систем большинство ранних систем было оснащено "реляционными" интерфейсами. Однако в большинстве случаев это не сделало их по-настоящему реляционными системами, поскольку оставалась возможность манипулировать данными в естественном для них режиме.

По структуре хранимой информации различают:

- Документальные БД - для работы с документами на естественном языке – это БД, предназначенные для накопления и подбора документов, удовлетворяющих различным критериям (тексты законодательных актов, публикации, подборка монографий и т.д.);

- Фактографические БД – фактические сведения, представленные в формализованном виде, используемые для решения задач обработки данных.

Пользователь БД – лицо, работающее с БД. Пользователи применяют БД в различных областях, например:

- организация хранилищ данных;
- анализ данных и принятие решения;
- мобильные, персональные БД, распределенные ИС;

- географические БД, мультимедиа БД;
- базы данных для всемирной сети World Wide Web.

Контрольные вопросы:

1. Понятие автоматизированной информационной системы.
2. Структура АИС.
3. Опишите ранние СУБД.
4. Классификация баз данных по структуре хранимой информации.
5. Пользователь БД и его функции.

Лекция 5

Система управления базами данных и её составляющие. Архитектура банка данных.

План:

1. Понятие база и банк данных
2. Система управления базами данных. Администратор базы данных.
3. Основная задача СУБД. Ядро СУБД
4. Основные типы данных СУБД

Банк данных является разновидностью информационных систем, в которой хранится и накапливается информация. Он состоит из следующих компонентов:

- одной или несколько БД
- системы управления базами данных
- системного каталога
- администратора
- вычислительной системы
- обслуживающего персонала.

База данных представляет собой совокупность данных, хранимых в памяти вычислительной системы и отображающих состояние объектов и их взаимосвязей.

БД бывают централизованными (хранятся на одном компьютере) и распределенными (хранятся на нескольких компьютерах некоторой сети).

Система управления базами данных – это комплекс языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями.

Словарь данных - представляет собой подсистему БД, предназначенную для хранения информации о структурах данных, взаимосвязях файлов БД друг с другом, типах данных и форматах их представления, Словарь данных называют системным каталогом. Он хранит служебную информацию о данных в базе.

Администратор базы данных – лицо или группа лиц, отвечающая за создание, эффективное использование и сопровождение БД. В процессе эксплуатации администратор обычно следит за функционированием информационной системы.

Вычислительная система – представляет собой несколько взаимосвязанных и согласованно действующих компьютеров или процессоров и других устройств.

Обслуживающий персонал – выполняет функции поддержания технических и программных средств в работоспособном состоянии.

\Система управления базами данных – это комплекс языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями. СУБД обеспечивает определение физической и логической структуры БД.
Основной задачей СУБД является выполнение операций над информацией, и включает в себя:

- ввод в БД новых записей и файлов;
- обновление содержания БД;
- удаление из БД устаревшей информации;
- быстрый поиск требуемой информации;
- выдача необходимой информации пользователям;
- объединение и разъединение файлов;
- копирование и восстановление файлов и др.

Ядро СУБД отвечает за управление данными во внешней памяти, управление буферами оперативной памяти, управление транзакциями и журнализацию.

Можно выделить такие компоненты ядра:

- менеджер данных;
- менеджер буферов;

- менеджер транзакций;
- менеджер журнала.

Функции этих компонентов взаимосвязаны, и взаимодействуют по тщательно продуманным и проверенным протоколам.

Ядро СУБД обладает собственным интерфейсом, не доступным пользователям напрямую.

Основной функцией компилятора языка БД является компиляция операторов языка БД в некоторую выполняемую программу.

Реальное выполнение оператора производится с привлечением подсистемы поддержки времени выполнения, представляющей собой интерпретатор.

В отдельные утилиты БД обычно выделяют такие процедуры, как загрузка и выгрузка БД, сбор статистики, глобальная проверка целостности БД и т.д.

Основные типы данных СУБД

Первоначально СУБД применялись преимущественно для решения финансово-экономических задач. При этом независимо от модели использовались следующие основные типы данных:

числовые – используются целочисленные, денежные (финансовые),. вещественные типы данных: 0,43; 328.

символьные – алфавитно-цифровые типы данных: «студент»; «лекция».

логические – принимающие значения «истина» (true) и «ложь» (false);

типа «Дата» - 1.12.97; 2/23/2005.

С расширением области применения персональных компьютеров стали появляться специализированные системы обработки данных, например, обработка видеоизображений и т.д. В них стали вводиться новые типы данных. К числу новых типов данных относятся следующие:

временные и дата-временные – предназначенные для хранения . информации о времени и/или дате: 31.01.85(дата); 9:10:03(время); . . 1.03.1994 12:00(дата и время).

символьные переменной длины – для хранения текстовой информации . большой длины, например, документа.

двоичные (мультимедиа-данные) - для хранения графических объектов, .аудио и видеоинформации, . . пространственной, хронологической, . текстовой с эффектами . отображения на экране (анимацией) и . другой специальной информации.

гиперссылки (hyperlinks) - предназначенные для хранения ссылок /на различные ресурсы (узлы, файлы, / документы), находящиеся вне БД, например, в сети Internet: http:\\www.chat.ru, . ftp:\\chance4u.teens.com.

данные в XML формате -- стандарт форматирования данных в / Internete (набор типов, включаемых в / документ для определения его / структуры).

Контрольные вопросы:

1. Что представляет собой банк данных ?
2. Какие виды баз данных вы знаете?
3. Что такое система управления базами данных?
4. Что входит в ядро СУБД?
5. Основные типы данных СУБД.

Лекция 6.

Модель базы данных. Иерархическая и сетевая модель данных.

План:

1.Модели данных. Разновидности моделей данных.
2. Иерархическая модель данных.Достоинства и недостатки.
3. Манипулирование данными и ограничения целостности.
4. Сетевая модель данных.Достоинства и недостатки.

Большинство объектов физического мира неимоверно сложны по своей организации. Когда мы пытаемся описать какой-либо из таких объектов мы на самом деле придумываем модель, соответствующую ему в нашем понимании. Если объекты можно поделить на некоторые группы, удовлетворяющие одинаковым моделям, то мы получаем ситуацию, когда внутри базы данных хранятся две группы сущностей:

- описания моделей объектов;
- записи, удовлетворяющие какой-либо из модели и соответствующие различным представителям объектов.

Но бывают ситуации, когда объекты настолько различны, что их нельзя классифицировать. Тогда база данных представляет из себя набор из одних лишь моделей.

Разновидности моделей данных

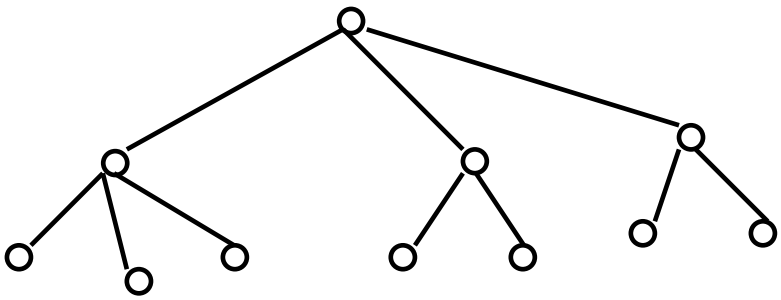
Когда мы говорим о моделях данных мы должны понимать, что нужно уметь хранить не только сами объекты, но и взаимосвязи между ними. За долгую историю развития баз данных было разработано много вариантов организации информации. Вот основные из них:

- иерархическая модель;
- сетевая модель;
- реляционная модель;
- объектная модель.

Сейчас актуальны в основном реляционная и объектная модели.

Иерархическая модель данных

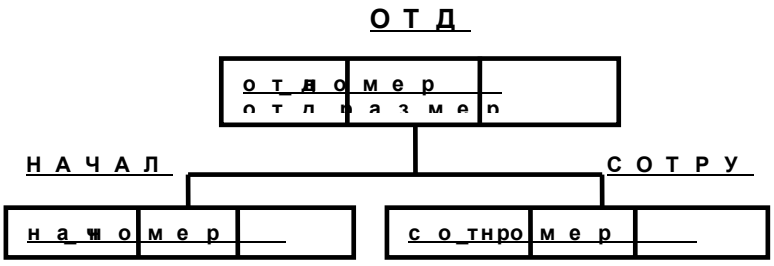
В иерархической модели связи между данными можно описать с помощью упорядоченного графа (или дерева). Представление связей между данными в иерархической модели:



Иерархическая модель позволяет строить БД с древовидной структурой, где каждый узел содержит свой тип данных (сущность). На верхнем уровне дерева имеется один узел – корень, на следующем уровне располагаются узлы, связанные с этим корнем, затем узлы, связанные с узлами предыдущего уровня и т.д.Иерархическая БД состоит из упорядоченного набора деревьев; более точно, из упорядоченного набора нескольких экземпляров одного типа дерева. Тип дерева является составным. Он включает в себя подтипы («поддеревья»), каждый из которых в свою очередь является типом «дерево» Тип дерева состоит из одного "корневого" типа записи и упорядоченного набора из нуля или более типов поддеревьев. Тип дерева в целом представляет собой иерархически организованный набор типов записи.

Каждый из элементарных типов, включенных в тип «дерево», является простым или составным типом «запись». Простая «запись» состоит из одного типа, например числового, а составная «запись» объединяет некоторую совокупность типов, например целое, строку символов и указатель (ссылку).

Пример типа дерева (схемы иерархической БД):



Здесь ОТДЕЛ является предком для НАЧАЛЬНИК и СОТРУДНИКИ.

НАЧАЛЬНИК и СОТРУДНИКИ - потомки ОТДЕЛ. Между типами записи поддерживаются связи.

Корневым называется тип, который имеет подчиненные типы, а сам не является подтипом. Подчиненный тип (подтип) является потомком по отношению к типу, который выступает для него в роли предка (родителя).

Такая БД представляет собой упорядоченную совокупность экземпляров данного типа «дерево» (деревьев), содержащих экземпляры типа «запись» (записи). Часто отношения родства между типами переносят на отношения между самими записями. Поля записей хранят собственно числовые или символьные значения, составляющие основное содержание БД. База данных с такой схемой могла бы выглядеть следующим образом (один экземпляр дерева):

Все экземпляры данного типа потомка с общим экземпляром типа предка называются близнецами. Для БД определен полный порядок обхода - сверху-вниз, слева-направо. Поиск данных в иерархической системе всегда начинается с корня. Затем производится спуск с одного уровня дерева на другой, пока не будет достигнут искомый уровень. Перемещение по системе от одной записи к другой осуществляется с помощью ссылок.

Манипулирование данными и ограничения целостности

Примерами типичных операторов манипулирования иерархически организованными данными могут быть следующие:

- Найти указанное дерево БД (например, отдел 310);
- Перейти от одного дерева к другому;
- Перейти от одной записи к другой внутри дерева (например, от отдела - к первому сотруднику - Давлатов);
- Перейти от одной записи к другой в порядке обхода иерархии;
- Вставить новую запись в указанную позицию;
- Удалить текущую запись.

Автоматически поддерживается целостность ссылок между предками и потомками.

Основное правило: никакой потомок не может существовать без

своего родителя,

каждый узел может иметь только одного предка.

Заметим, что аналогичное поддержание целостности по ссылкам между записями, не входящими в одну иерархию, не поддерживается.

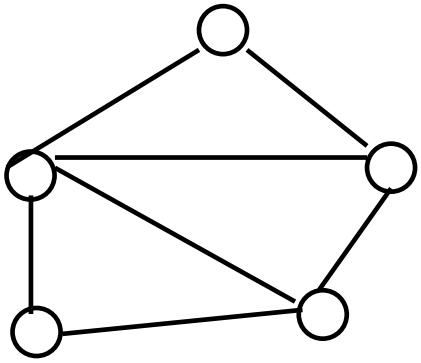
Достоинства иерархической модели - простота описания иерархических структур реального мира, быстрота выполнения запросов, эффективное использование памяти.

Основные недостатки:

● Неэффективна.
● Медленный доступ к сегментам данных нижних уровней иерархии.
● Четкая ориентация на определенные типы запросов
● Громоздкая для обработки информации со сложными логическими связями (параллельные структуры–деревьев). Вспомните библиотеку: предметный указатель и алфавитный указатель.
● Сложная в понимания для обычного пользователя.

Сетевая модель данных

Представителем является Integrated Database Management System (IDMS) компании Cullinet Software, Inc., предназначенная для использования на машинах основного класса фирмы IBM под управлением большинства операционных систем. Сетевая модель данных позволяет отображать разнообразные взаимосвязи элементов данных в виде произвольного графа, обобщая тем самым иерархическую модель данных.



Для описания схемы сетевой БД используется две группы типов: запись и связь. Тип связь определяется для двух типов запись: предка и потомка. Переменные типа связь являются экземплярами связей.

Сетевой подход к организации данных является расширением иерархического. В иерархических структурах запись - потомок должна иметь в точности одного предка; в сетевой структуре данных потомок может иметь любое число предков (сводных родителей).

Сетевая БД состоит из набора записей и набора связей между этими записями, а если говорить более точно, из набора экземпляров каждого типа из заданного в схеме БД набора типов записи и набора экземпляров каждого типа из заданного набора типов связи.

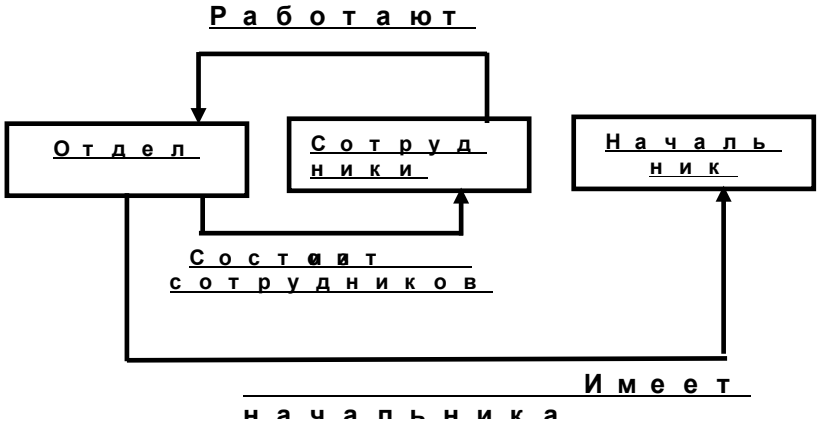
Для данного типа связи L с типом записи предка P и типом записи потомка C должны выполняться следующие два условия:

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

На формирование типов связи не накладываются особые ограничения. Например, потомок в типе связи L1 может быть предком в типе связи L2 и т.п. Типы связей здесь обозначены надписями на соединяющих типы записей линиях.

В различных СУБД сетевого типа для обозначения одинаковых по сути понятий зачастую используют различные термины. Например, такие как элементы и агрегаты данных, записи, наборы, области и т.д.

Пример сетевой схемы БД:

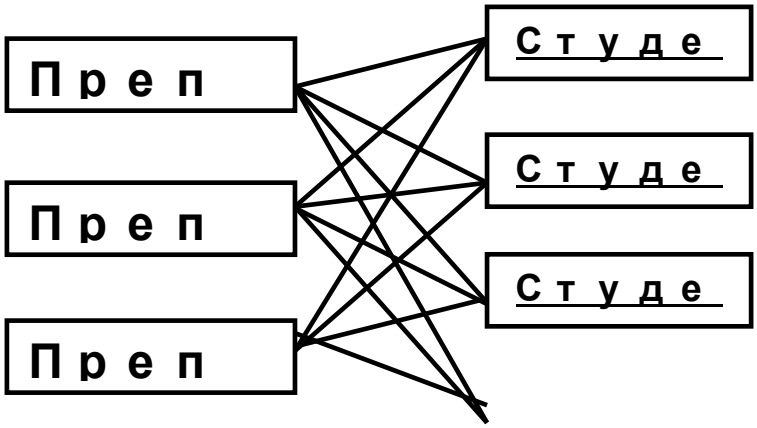


Использование и иерархической и сетевой моделей ускоряет доступ к информации в БД. Однако, поскольку каждый элемент данных должен содержать ссылки на некоторые другие элементы, требуются значительные ресурсы как дисковой, так и основной памяти. Недостаточность основной памяти, конечно, снижает скорость обработки данных.

Достоинством сетевой модели данных является возможность эффективной реализации по показателям затрат памяти и оперативности. В сравнении с иерархической моделью сетевая модель представляет большие возможности в смысле допустимости образования произвольных связей.

Недостатком сетевой модели данных является высокая сложность и жесткость схемы БД, построенной на ее основе, а также сложность для понимания и выполнения обработки информации в БД. Также здесь ослаблен контроль целостности связей вследствие допустимости установления произвольных связей между записями.

Пример, каждый преподаватель может обучать многих (теоретически всех) студентов и каждый студент может обучаться у многих (теоретически у всех) преподавателей



С т у д е

Контрольные вопросы:

1. Ранние и современные модели БД.
2. Иерархическая модель.
3. Понятия «корень», «предок», «потомок», «близнец».
4. Сетевая модель данных .
5. Достоинства и недостатки сетевых и иерархических моделей.