Базы данных. СУБД. Модели представления Данных.

Традиционные файловые системы.

Файловая система - это набор программ, которые выполняют для пользователей некоторые операции, причем каждая программа определяет свои собственные данные и управляет ими.

В этом случае каждая функция автоматизированной организации может быть реализована одной или несколькими программами. Для доступа к каждому файлу надо создавать свои собственные программы.

Недостатки файловых систем:

- Дублирование данных из-за децентрализованной работы.
- а) сопровождается неэкономным расходованием ресурсов, т. к. ввод избыточных данных требует дополнительных временных, дисковых и др. финансовых ресурсов;
- б) может привести к нарушению целостности системы (целостность – набор правил (ограничений) в предметной области)
- Зависимость от данных – физическая структура и способ хранения информации жестко зафиксирован в коде программ, это сильно затрудняет изменение структуры информации.
- Несовместимость форматов файлов – поскольку структура файлов определяется кодом приложения, то она также зависит от языка программирования этого приложения.
- Фиксированные запросы – файловые системы зависят от программиста и поэтому выполняют те запросы, которые он туда заложил.

Все перечисленные недостатки файловых систем являются следствием 2-х факторов:

1) определение данных содержится внутри приложений, а не хранится отдельно;

2)помимо приложений не существует никаких инструментов доступа к данным.

База данных – это совместно используемый набор логически связанных данных и описание этих данных.

База данных представляет собой единое хранилище информации организации. При этом база данных должна быть снабжена описанием структуры. Такое описание называется словарем данных (системным каталогом), элементы этого каталога называются метаданными (данные о данных).

Хранение данных, отдельно от их описания позволяет использовать абстрагирование данных. Оно заключается в том, что можно изменить внутреннее представление объекта, не изменяя внешнего, то есть изменение объекта не приводит к изменению способов его обработки. Логическая связь данных заключается в том, что информация описывается некоторой логической моделью (напр., модель «сущность-связь»).

СИТСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ (СУБД) – это программное обеспечение, с помощью которого пользователи могут определять, создавать БД, а также осуществлять к ним контролируемый доступ.

Основные функции СУБД

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

СУБД предоставляет пользователям возможность сохранять, извлекать и изменять данные. 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). Подчеркнем, что в развитых СУБД пользователи в любом случае не обязаны знать то, как организованы файлы.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

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 СУБД поддерживается собственный набор буферов оперативной памяти с собственной дисциплиной замены буферов.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

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 (ROLLBACK). Понятие транзакции необходимо для поддержания логической целостности БД.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

Поддержание механизма транзакций является обязательным условием даже однопользовательских СУБД. Но понятие транзакции гораздо более важно в многопользовательских СУБД.

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 (в некоторых случаях пользователи многопользовательских СУБД могут ощутить присутствие своих коллег).

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

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.

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.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

Типичная запись журнала

Время начала операции	Имя пользователя	Текст запроса	Результат запроса	Разрешения пользователя	Сетевой адрес	…

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

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 (создание таблиц, внешних ключей и т.п.).

DML содержал набор операторов манипулирования данными, т.е. операторов, позволяющих заносить данные в БД, удалять, модифицировать или выбирать существующие данные.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

Стандартным языком наиболее распространенных в настоящее время реляционных СУБД является язык SQL (Structured Query Language). Прежде всего, язык SQL сочетает средства SDL и DML, т.е. позволяет определять схему реляционной БД и манипулировать данными.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

Язык SQL содержит специальные средства определения ограничений целостности БД. 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.

- Непосредственное управление данными во внешней памяти
- Управление буферами оперативной памяти
- Управление транзакциями
- Журнализация
- Поддержка языков БД
- Поддержка целостности и контроль доступа к данным

Специальные операторы языка SQL позволяют определять так называемые представления БД, фактически являющиеся хранимыми в БД запросами 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.

Наконец, авторизация доступа к объектам БД производится также на основе специального набора операторов SQL. Идея состоит в том, что для выполнения операторов SQL разного вида пользователь должен обладать различными полномочиями.

Уровни представления данных

Описание, выполненное с использованием естественного языка, математических формул, таблиц, графиков и других средств, понятных всем людям, работающих над проектированием базы данных, называют *инфологической моделью данных.*
Такая человеко-ориентированная модель полностью независима от физических параметров среды хранения данных.

Нужные данные отыскиваются СУБД на внешних запоминающих устройствах по *физической модели данных*.

Так как указанный доступ осуществляется с помощью конкретной СУБД, то модели должны быть описаны на *языке описания данных* этой СУБД. Такое описание, создаваемое АБД по инфологической модели данных, называют *даталогической (логической) моделью данных*.

Трехуровневая архитектура (инфологический, даталогический и физический уровни) позволяет обеспечить *независимость хранимых данных* от использующих их программ.

Администратор БД может при необходимости переписать хранимые данные на другие носители информации и (или) реорганизовать их физическую структуру, изменив лишь физическую модель данных.

Администратор БД может подключить к системе любое число новых пользователей (новых приложений), дополнив, если надо, логическую модель.
Указанные изменения физической и логической моделей не будут замечены существующими пользователями системы (окажутся "прозрачными" для них), так же как не будут замечены и новые пользователи.

Независимость данных обеспечивает возможность развития системы баз данных без разрушения существующих приложений.

Иерархическая модель данных

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

Тип дерева состоит из одного "корневого" типа записи и упорядоченного набора из нуля или более типов поддеревьев (каждое из которых является некоторым типом дерева). Тип дерева в целом представляет собой иерархически организованный набор типов записи.

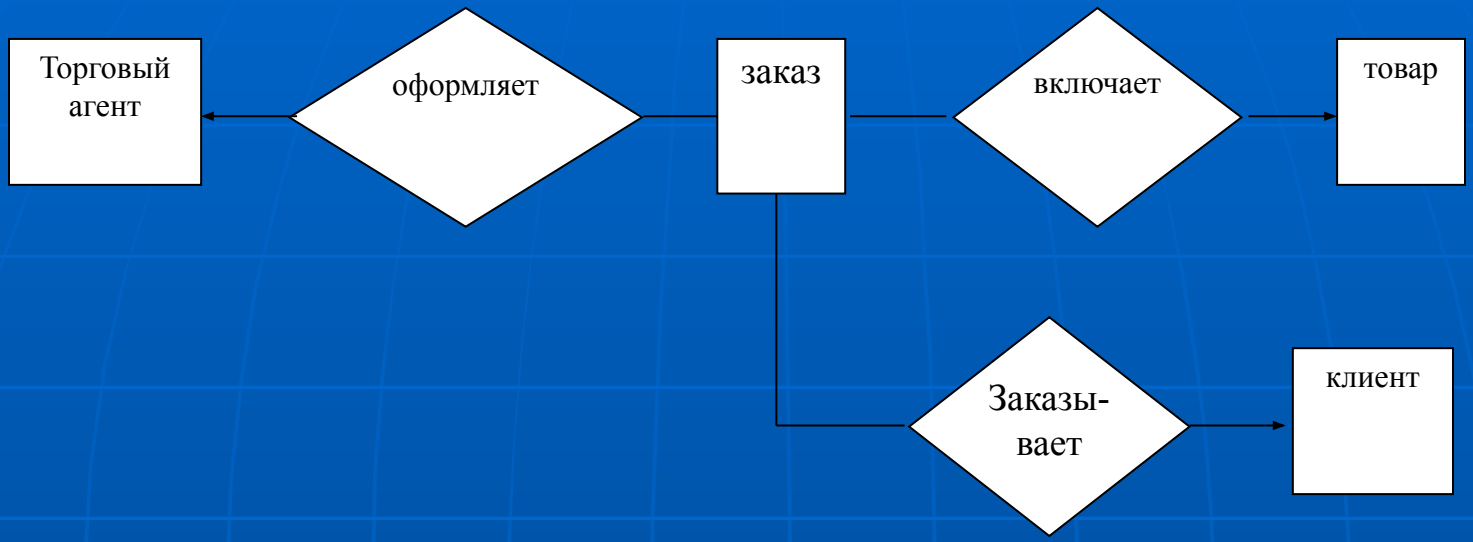
Достоинства:
простота понимания принципа иерархии.

Недостатки:
сложность отображения связи МНОГИЕ КО МНОГИМ;
иерархия в значительной степени усложняет включение информации о новых объектах в БД и удаление устаревшей.

Сетевая модель данных

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.

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.



Достоинства:

- Поддержание правил целостности на семантическом уровне;
- Реализация связи МНОГИЕ КО МНОГИМ.

Недостатки:

- Слишком сложно пользоваться;
- Фактически необходимы знания о физической организации;
- Прикладные системы зависят от этой организации.

Реляционная модель данных

Основными понятиями реляционных баз данных являются тип данных, домен, атрибут, кортеж, первичный ключ и отношение.

Понятие *тип данных* в реляционной модели данных полностью адекватно понятию типа данных в языках программирования. Обычно в современных реляционных БД допускается хранение символьных, числовых данных, битовых строк и т.п.

Понятие *домена* более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в некоторых языках программирования. В самом общем виде домен определяется заданием некоторого базового типа данных, к которому относятся элементы домена, и произвольного логического выражения, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат "истина", то элемент данных является элементом домена. Наиболее правильной интуитивной трактовкой понятия домена является понимание домена как допустимого потенциального множества значений данного типа.

Атрибут – неотъемлемое свойство объекта. Те атрибуты, которые однозначно определяют сущность объекта – называются **ключевыми** (ключом).

Схема отношения - это именованное множество пар {имя атрибута : имя домена}. Степень или "арность" схемы отношения - мощность этого множества. Схема БД (в структурном смысле) - это набор именованных схем отношений.

Кортеж, соответствующий данной схеме отношения, - это множество пар {имя атрибута, значение}, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. "Значение" является допустимым значением домена данного атрибута (или типа данных, если понятие домена не поддерживается). Тем самым, степень или "арность" кортежа, т.е. число элементов в нем, совпадает с "арностью" соответствующей схемы отношения. Попросту говоря, кортеж - это набор именованных значений заданного типа.

Отношение - это множество кортежей, соответствующих одной схеме отношения.


Целые числа
Строковые переменные
Вещественные числа
Типы данных
Табельные номера
Имена
Валюта
Номера отделов
Домены
Сотр_номер
Сотр_фамилия
Сотр_зарплата
Сотр_отдел
2901
Иванов
3000
100
2903
Петров
3500
100
2906
Сидоров
5200
100
Отношение
Первичный ключ

Свойства отношений

Свойства отношений непосредственно следуют из приведенного выше определения отношения.

1. В отношении нет одинаковых кортежей. Действительно, тело отношения есть множество кортежей и, как всякое множество, не может содержать неразличимые элементы.

2. Кортежи не упорядочены (сверху вниз).

3. Атрибуты не упорядочены (слева направо).

4. Все значения атрибутов атомарны. Это следует из того, что лежащие в их основе атрибуты имеют атомарные значения.

Из свойств отношения следует, что не каждая таблица может задавать отношение.

Реляционный термин	Соответствующий "табличный" термин
База данных	Набор таблиц
Схема базы данных	Набор заголовков таблиц
Отношение	Таблица
Заголовок отношения	Заголовок таблицы
Тело отношения	Тело таблицы
Атрибут отношения	Наименование столбца таблицы
Кортеж отношения	Строка таблицы
Степень (-арность) отношения	Количество столбцов таблицы
Мощность отношения	Количество строк таблицы
Домены и типы данных	Типы данные в ячейках таблицы

Для того, чтобы некоторая таблица задавала отношение, необходимо, чтобы таблица имела простую структуру (содержала бы только строки и столбцы, причем, в каждой строке было бы одинаковое количество полей), в таблице не должно быть одинаковых строк, любой столбец таблицы должен содержать данные только одного типа, все используемые типы данных должны быть простыми.

Наиболее распространенная трактовка реляционной модели данных принадлежит Дейту, который воспроизводит ее практически во всех своих книгах. Согласно Дейту реляционная модель состоит из трех частей, описывающих разные аспекты реляционного подхода:

- структурной части,
- манипуляционной части,
- целостной части.

Структурная часть описывает, какие объекты рассматриваются реляционной моделью. Постулируется, что единственной структурой данных, используемой в реляционной модели, являются нормализованные n-арные отношения.

Целостная часть описывает ограничения специального вида, которые должны выполняться для любых отношений в любых реляционных базах данных. Это целостность сущностей и целостность внешних ключей.

Манипуляционная часть описывает два эквивалентных способа манипулирования реляционными данными - реляционную алгебру и реляционное исчисление.

Современные СУБД

- MS SQL Server
- Interbase/Firebird
- Oracle
- IBM DB2
- MySQL/PostgreSQL