Автоматизированные информационные системы

- С самого начала развития вычислительной техники образовались два основных направления ее использования. Первое направление - применение вычислительной техники для выполнения численных расчетов, которые слишком долго или вообще невозможно производить вручную.
- Второе направление, это использование средств вычислительной техники в автоматических или автоматизированных информационных системах.

- 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.
- Информационная система взаимосвязанная совокупность средств, методов и персонала представляющая собой программный комплекс, функции которого состоят в поддержке надежного хранения, в выполнении специфических для данного приложения преобразований информации и/или вычислений и выдачи информации в интересах достижения поставленной цели, в предоставлении пользователям удобного и легко осваиваемого интерфейса.

- Обычно объемы информации, с которыми приходится иметь дело таким системам, достаточно велики, а сама информация имеет достаточно сложную структуру. Классическими примерами информационных систем являются банковские системы, системы резервирования авиационных или железнодорожных билетов, мест в гостиницах и т.д.
- Информационные технологии – это методы и способы, использующие компьютерные программно-технические средства, информационные процессы и операции, предназначенные для достижения поставленных целей.

- Автоматизированная информационная система (АИС) – комплекс программных, технических, информационных, лингвистических, организационно-технологических средств и персонала, предназначенный для решения задач справочно-информационного обслуживания и (или) информационного обеспечения пользователей.

- На рынке автоматизированных систем для крупных корпораций и финансово-промышленных групп на сегодня можно выделить два основных субъекта: это ранок автоматизированных банковских систем (АБС) и рынок корпоративных информационных систем промышленных предприятий.

Современные крупные проекты ИС характеризуются следующими особенностями:

- сложность описания;
- наличие совокупности тесно взаимодействующих компонентов, имеющих свои локальные задачи и цели функционирования;
- отсутствие прямых аналогов, ограничивающее возможность их использования;
- необходимость интеграции существующих и вновь разрабатываемых приложений;
- функционирование в неоднородной среде на нескольких аппаратных платформах.

Типовая организация современной СУБД

- управление данными во внешней памяти;
- управление буферами оперативной памяти;
- управление транзакциями;
- журнализация и восстановление БД после сбоев;
- поддержание языков БД.

Управление данными во внешней памяти

- Эта функция включает обеспечение необходимых структур внешней памяти как для хранения непосредственных данных, 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 серверов баз данных активно используются возможности существующих файловых систем, в других работа производится вплоть до уровня устройств внешней памяти.

управление буферами оперативной памяти

- Серверы баз данных работают с БД значительного размера; этот размер больше доступного объема оперативной памяти. При обращении к любому элементу данных будет производиться обмен с внешней памятью, поэтому вся система будет работать со скоростью устройства внешней памяти. Единственным способом реального увеличения этой скорости является буферизация данных в оперативной памяти.

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 в состоянии БД

Транзакции необходимо для поддержания логической целостности БД. 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.

Журнализация и восстановление БД после сбоев

Одним из основных требований к СУБД является надежное хранение 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.

- Существуют два возможных вида аппаратных сбоев:
  - *мягкие сбои,* которые можно трактовать как внезапную остановку работы компьютера (например, аварийное выключение питания);
  - *жесткие сбои*, характеризуемые потерей информации на носителях внешней памяти..

- ***Журнал*** - это особая часть БД, недоступная пользователям СУБД и поддерживаемая с особой тщательностью, в которую поступают записи обо всех изменениях основной части БД..
- 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, если в СУБД корректно соблюдается протокол WAL, то с помощью журнала можно решить все проблемы восстановления БД после любого сбоя.

Поддержание 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 или Standard Query Language)..

- **Базы данных** позволяют хранить, структурировать информацию и извлекать оптимальным для пользователя образом.

В узком смысле слова, база данных

- это некоторый набор данных, необходимых для работы (актуальные данные).

Данные - это отражение объектов реального мира.

В традиционной терминологии объекты реального мира, сведения о которых хранятся в базе данных, называются сущностями - entities,

а их актуальные признаки - атрибутами (attributes).

Каждый признак конкретного объекта есть значение атрибута.

В широком смысле слова *база данных*

- - это совокупность описаний объектов реального мира и связей между ними, актуальных для конкретной прикладной области.

ОПРЕДЕЛЕНИЯ

- БАЗА ДАННЫХ (БД) — однозначно идентифицируемый массив данных заданной структуры, размещаемый на машиночитаемых носителях.
- БАЗА ДАННЫХ — совокупность организованных взаимосвязанных данных на машиночитаемых носителях.
- СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ (СУБД) — совокупность программ и языковых средств, предназначенных для управления данными в базе данных и обеспечения взаимодействия ее с прикладными программами.

- **БАЗА ЗНАНИЙ** — совокупность фактов (или утверждений), относящихся к некоторой предметной области, и правил, которые инициируются путем сопоставления с соответствующими признаками, характеризующими данную задачу, и которые могут добавляться, модифицироваться или изыматься пользователем.
- **БАНК ДАННЫХ** — совокупность баз данных, а также программные, языковые и другие средства, предназначенные для централизованного накопления данных и их использования с помощью электронных вычислительных машин.

База данных (БД, database) - поименованная совокупность структурированных данных, относящихся к определенной предметной области.

Предметная область - некоторая часть реально существующей системы, функционирующая как самостоятельная единица. Полная предметная область может представлять собой экономику страны или группы союзных государств, однако на практике для информационных систем наибольшее значение имеет предметная область масштаба отдельного предприятия или корпорации.

Система управления базами данных (СУБД) - комплекс программных и языковых средств, необходимых для создания и модификации базы данных, добавления, модификации, удаления, поиска и отбора информации, представления информации на экране и в печатном виде, разграничения прав доступа к информации, выполнения других операций с базой.

- **Реляционная БД** - основной тип современных баз данных. Состоит из таблиц, между которыми могут существовать связи по ключевым значениям.

- **Таблица** базы данных (table) - регулярная структура, которая состоит из однотипных строк (записей, records), разбитых на столбцы (поля, fields).

- В теории реляционных баз данных синоним таблицы - **отношение** (relation), в котором строка называется **кортежем**, а столбец называется **атрибутом**.

- В концептуальной модели реляционной БД аналогом таблицы является сущность (entity), с определенным набором свойств - **атрибутов**, способных принимать определенные значения (набор допустимых значений - **домен**).

- **Ключевой элемент таблицы** (ключ, regular key) - такое ее поле (простой ключ) или строковое выражение, образованное из значений нескольких полей (составной ключ), по которому можно определить значения других полей для одной или нескольких записей таблицы. На практике для использования ключей создаются индексы - служебная информация, содержащая упорядоченные сведения о ключевых значениях. В реляционной теории и концептуальной модели понятие "ключ" применяется для атрибутов отношения или сущности.
- **Первичный ключ** (primary key) - главный ключевой элемент, однозначно идентифицирующий строку в таблице. Могут также существовать альтернативный (candidate key) и уникальный (unique key) ключи, служащие также для идентификации строк в таблице.
- В реляционной теории **первичный ключ** - минимальный набор атрибутов, однозначно идентифицирующий кортеж в отношении.
- В концептуальной модели **первичный ключ** - минимальный набор атрибутов сущности, однозначно идентифицирующий экземпляр сущности.

- **Связь** (relation) - функциональная зависимость между объектами. В реляционных базах данных между таблицами устанавливаются связи по ключам, один из которых в главной (parent, родительской) таблице - первичный, второй - внешний ключ - во внешней (child, дочерней) таблице, как правило, первичным не является и образует связь "один ко многим" (1:N). В случае первичного внешнего ключа связь между таблицами имеет тип "один к одному" (1:1). Информация о связях сохраняется в базе данных.
- **Внешний ключ** (foreign key) - ключевой элемент подчиненной (внешней, дочерней) таблицы, значение которого совпадает со значением первичного ключа главной (родительской) таблицы.
- **Ссылочная целостность данных** (referential integrity) - набор правил, обеспечивающих соответствие ключевых значений в связанных таблицах.

- **Хранимые процедуры** (stored procedures) - программные модули, сохраняемые в базе данных для выполнения определенных операций с информацией базы.
- **Триггеры** (triggers) - хранимые процедуры, обеспечивающие соблюдение условий ссылочной целостности данных в операциях изменения первичных ключей (возможно каскадное изменение данных), удалении записей в главной таблице (каскадное удаление в дочерних таблицах) и добавлении записей или изменении данных в дочерних таблицах.
- **Объект** (object) - элемент информационной системы, обладающий определенными свойствами (properties) и определенным образом реагирующий на внешние события (events).
- **Система** - совокупность взаимодействующих между собой и с внешним окружением объектов.

- **Репликация базы данных** - создание копий базы данных (реплик), которые могут обмениваться обновляемыми данными или реплицированными формами, отчетами или другими объектами в результате выполнения процесса синхронизации.

- **Транзакция** - изменение информации в базе в результате выполнения одной операции или их последовательности, которое должно быть выполнено полностью или не выполнено вообще. В СУБД существуют специальные механизмы обеспечения транзакций.

- **Язык SQL** (Structured Query Language) - универсальный язык работы с базами данных, включающий возможности ее создания, модификации структуры, отбора данных по запросам, модификации информации в базе и прочие операции манипулирования базой данных.

- **Null** - значение поля таблицы, показывающее, что информация в данном поле отсутствует. Разрешение на возможность существования значения Null может задаваться для отдельных полей таблицы.

- **Концептуальная модель** - отображает информационные объекты, их свойства и связи между ними без указания способов физического хранения информации (модель предметной области, иногда ее также называют информационно-логической или инфологической моделью). Информационными объектами обычно являются **сущности** - обособленные объекты или события, информацию о которых необходимо сохранять, имеющие определенные наборы свойств - **атрибутов**.

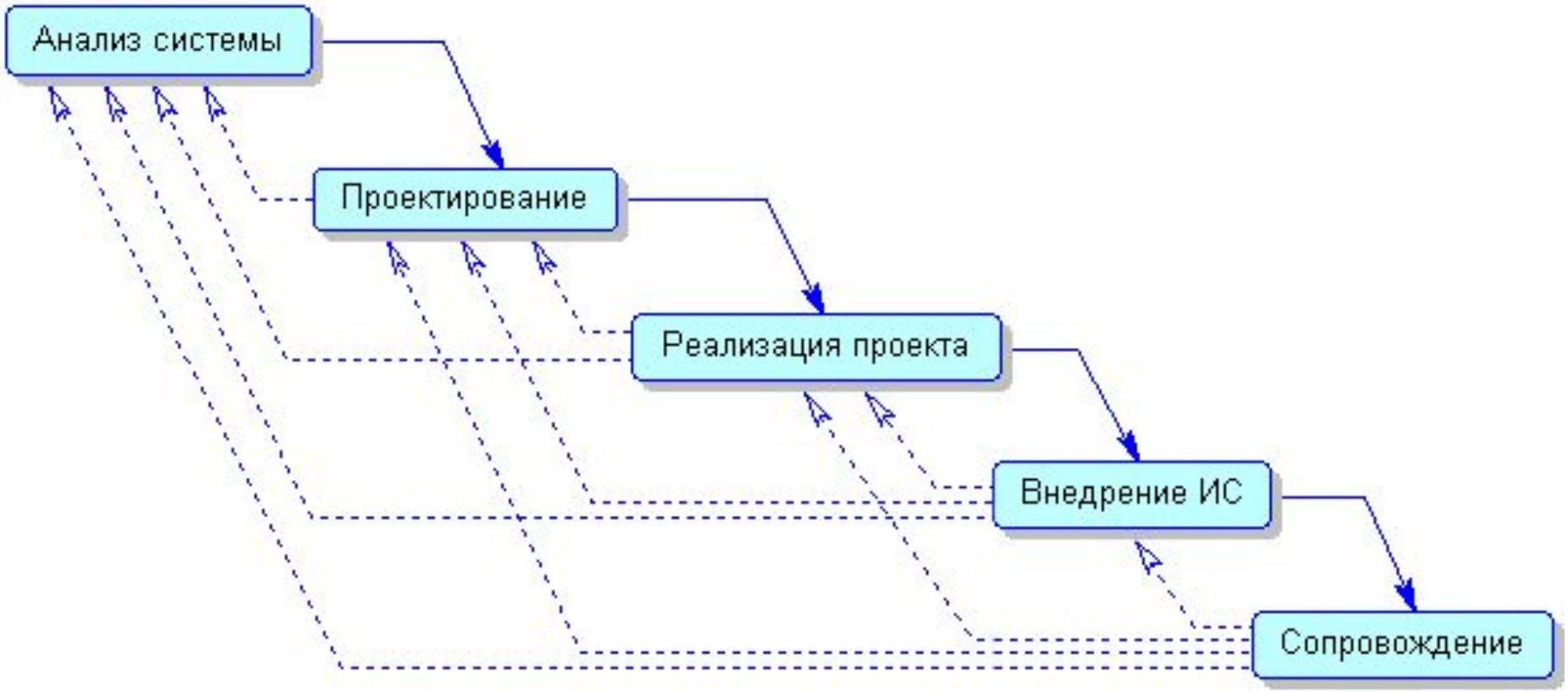
- **Физическая модель** - отражает все свойства (атрибуты) информационных объектов базы и связи между ними с учетом способа их хранения - используемой СУБД.

- **Внутренняя модель** - база данных, соответствующая определенной физической модели.

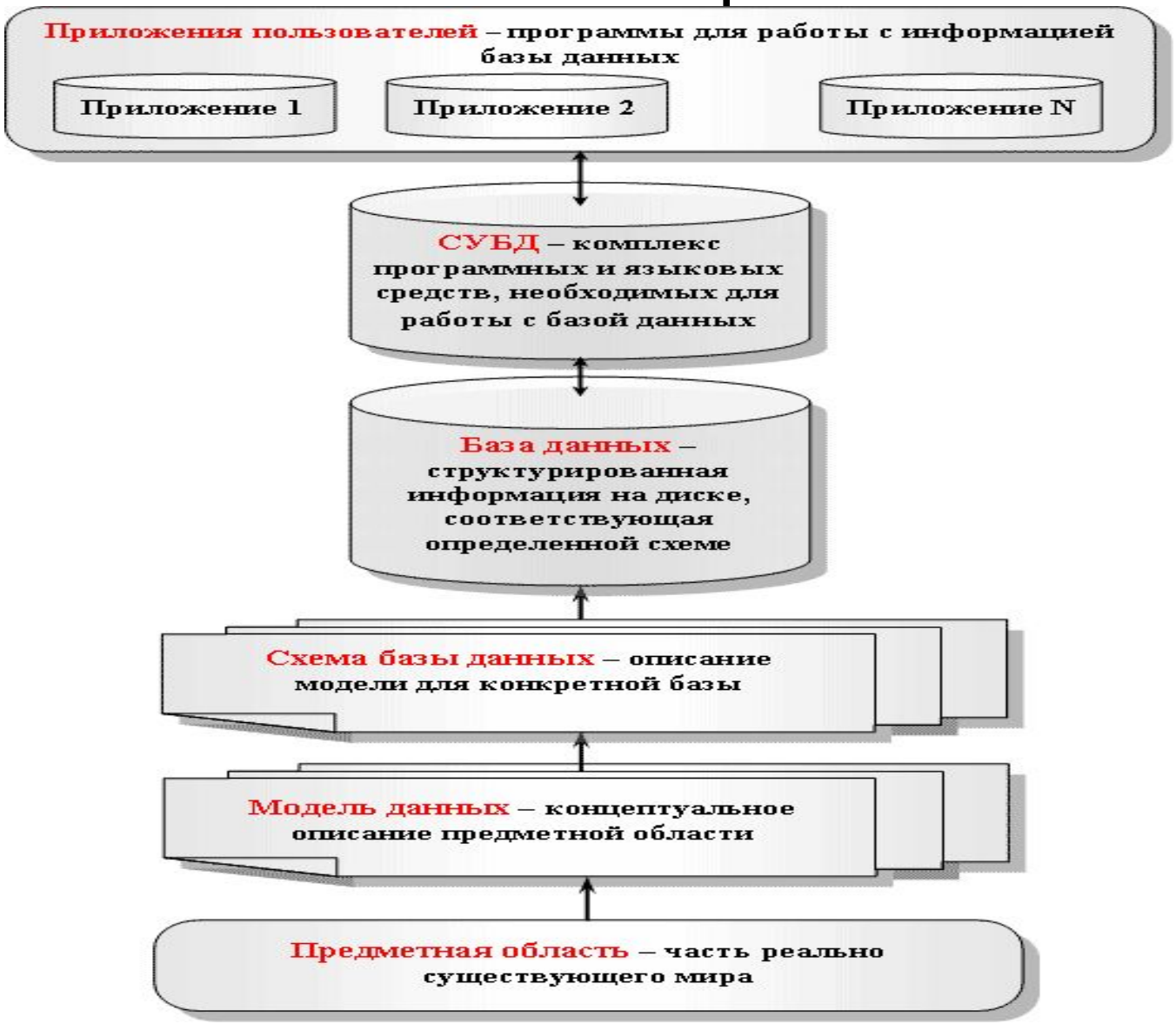
- **Внешняя модель** - комплекс программных и аппаратных средств для работы с базой данных, обеспечивающий процессы создания, хранения, редактирования, удаления и поиска информации, а также решающий задачи выполнения необходимых расчетов и создания выходных печатных форм.

Схема формирования информационной модели

Жизненный цикл разработки



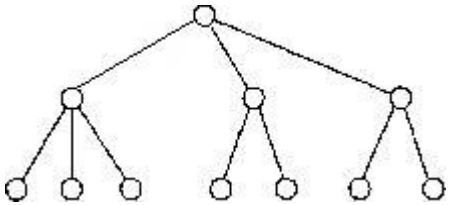
Взаимосвязь терминов



Классификация (по модели)

- Иерархическая
- Сетевая
- Реляционная
- Постреляционная
- Многомерная
- Объектно-ориентированная

Иерархическая модель



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

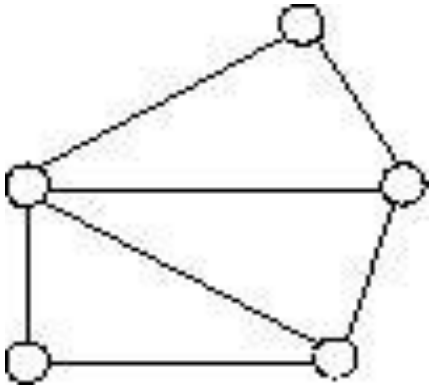
Основные операции манипулирования данными:
поиск указанного экземпляра БД;
переход от одного дерева к другому;
переход от одной записи к другой внутри дерева;
вставка новой записи в указанную позицию;
удаление текущей записи и т. д.

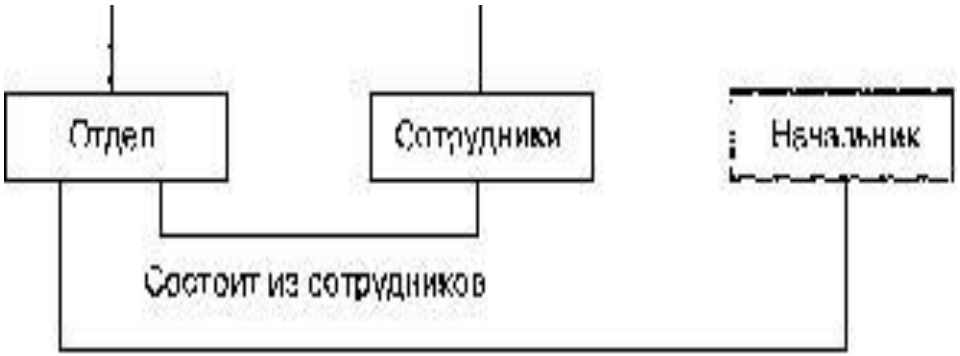
К достоинствам иерархической модели данных относятся эффективное использование памяти ЭВМ и неплохие показатели времени выполнения основных операций над данными. Иерархическая модель данных удобна для работы с иерархически упорядоченной информацией.

Недостатком иерархической модели является ее громоздкость для обработки информации с достаточно сложными логическими связями, а также сложность понимания для обычного пользователя.

На иерархической модели данных основано сравнительно ограниченное количество СУБД, в числе которых можно назвать зарубежные системы IMS, PC/Focus, Team-Up и Data Edge, а также отечественные системы Ока, ИНЭС и МИРИС.

Сетевая модель





Сетевая модель данных позволяет отображать разнообразные взаимосвязи элементов данных в виде произвольного графа, обобщая тем самым иерархическую модель данных. Наиболее полно концепция сетевых БД впервые была изложена в Предложениях группы КОДАСИЛ (KODASYL).

Сетевая модель

Основные операции манипулирования данными:
поиск записи в БД;
переход от предка к первому потомку;
переход от потомка к предку;
создание новой записи;
удаление текущей записи;
обновление текущей записи;
включение записи в связь;
исключение записи из связи;
изменение связей и т. д.

Системы на основе сетевой модели не получили широкого распространения на практике. Наиболее известными сетевыми СУБД являются следующие: IDMS, db VistaIII, СЕТЬ, СЕТОР и КОМПАС.

Достоинством сетевой модели данных является возможность эффективной реализации по показателям затрат памяти и оперативности. В сравнении с иерархической моделью сетевая модель предоставляет 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 обычным пользователем. Кроме того, в сетевой модели данных ослаблен контроль целостности связей вследствие допустимости установления произвольных связей между записями.

Реляционная модель

Реляционная модель данных предложена сотрудником фирмы IBM Эдгаром Коддом и основывается на понятии отношение (relation).

Достоинство реляционной модели данных заключается в простоте, понятности и удобстве физической реализации на ЭВМ. Именно простота и понятность для пользователя явились основной причиной их широкого использования. Проблемы же эффективности обработки данных этого типа оказались технически вполне разрешимыми.

Реляционная модель

Основными недостатками реляционной модели являются следующие: отсутствие стандартных средств идентификации отдельных записей и сложность описания иерархических и сетевых связей.

Примерами зарубежных реляционных СУБД для ПЭВМ являются следующие: все БД семейства dBase, DB2 (IBM),R:BASE (Microrim), FoxPro, FoxBase, Visual FoxPro, Microsoft Access, Clarion, Ingres, Oracle и т.д.

Постреляционная модель

Постреляционная модель данных представляет собой расширенную реляционную модель, снимающую ограничение неделимости данных, хранящихся в записях таблиц. Постреляционная модель данных допускает многозначные поля - поля, значения которых состоят из подзначений. Набор значений многозначных полей считается самостоятельной таблицей, встроенной в основную таблицу.

Достоинством постреляционной модели является возможность представления совокупности связанных реляционных таблиц одной постреляционной таблицей. Это обеспечивает высокую наглядность представления информации и повышение эффективности ее обработки.

.

Недостатком постреляционной модели является сложность решения проблемы обеспечения целостности и непротиворечивости хранимых данных.

К числу СУБД, основанных на постреляционной модели данных, относятся также системы Cashe, Bubba и Dasdb

Многомерная модель

Многомерные СУБД являются узкоспециализированными СУБД, предназначенными для интерактивной аналитической обработки информации.

Агрегируемостъ данных означает рассмотрение информации на различных уровнях ее обобщения. В информационных системах степень детальности представления информации для пользователя зависит от его уровня: аналитик, пользователь-оператор, управляющий, руководитель.

Историчность данных предполагает обеспечение высокого уровня статичности (неизменности) собственно данных и их взаимосвязей, а также обязательность привязки данных ко времени.

Объектно-ориентированная модель

В объектно-ориентированной модели при представлении данных имеется возможность идентифицировать отдельные записи базы. Между записями базы данных и функциями их обработки устанавливаются взаимосвязи с помощью механизмов, подобных соответствующим средствам в объектно-ориентированных языках программирования.

Структура объектно-ориентированной БД графически представима в виде дерева, узлами которого являются объекты. Свойства объектов описываются некоторым стандартным типом (например, строковым - string) или типом, конструируемым пользователем (определяется как class).

- Основным достоинством объектно-ориентированной модели данных в сравнении с реляционной является возможность отображения информации о сложных связях объектов. Объектно-ориентированная модель данных позволяет идентифицировать отдельную запись базы данных и определять функции их обработки.
- Недостатками объектно-ориентированной модели являются высокая понятийная сложность, неудобство обработки данных и низкая скорость выполнения запросов.

Инкапсуляция ограничивает область видимости имени свойства пределами того объекта, в котором оно определено. Так, если в объект типа КАТАЛОГ добавить свойство, задающее телефон автора книги и имеющее название телефон, то мы получим 2 Зак.925. одноименные свойства у объектов АБОНЕНТ и КАТАЛОГ. Смысл такого свойства будет определяться тем объектом, в который оно инкапсулировано.

Наследование, наоборот, распространяет область видимости свойства на всех потомков объекта. Так, всем объектам типа КНИГА, являющимся потомками объекта типа КАТАЛОГ, можно приписать свойства объекта-родителя: isbn, удк, название и автор. Если необходимо расширить действие механизма наследования на объекты, не являющиеся непосредственными родственниками (например, между двумя потомками одного родителя), то в их общем предке определяется абстрактное свойство типа abs. Так, определение абстрактных свойств билет и номер в объекте БИБЛИОТЕКА приводит к наследованию этих свойств всеми дочерними объектами АБОНЕНТ, КНИГА и ВЫДАЧА. Не случайно поэтому значения свойства билет классов АБОНЕНТ и ВЫДАЧА, показанных на рисунке, будут одинаковыми - 00015.

Полиморфизм в объектно-ориентированных языках программирования означает способность одного и того же программного кода работать с разнотипными данными. Другими словами, он означает допустимость в объектах разных типов иметь методы (процедуры или функции) с одинаковыми именами, Во время выполнения объектной программы одни и те же методы оперируют с разными объектами в зависимости от типа аргумента. Применительно к нашей объектно-ориентированной БД полиморфизм означает, что объекты класса КНИГА, имеющие разных родителей из класса КАТАЛОГ, могут иметь разный набор свойств. Следовательно, программы работы с объектами класса КНИГА могут содержать полиморфный код.

Классификация (по месту)

По технологии обработки данных базы данных подразделяются на:

- централизованные
- распределенные.

Централизованная БД

Хранится в памяти одной вычислительной системы. Эта вычислительная система может быть мэйнфреймом - тогда доступ к ней организуется с использованием терминалов - или файловым сервером локальной сети ПК.

Распределенная БД

Состоит из нескольких, возможно, пересекающихся или даже дублирующих друг друга частей, которые хранятся в различных ЭВМ вычислительной сети. Работа с такой базой осуществляется с помощью системы управления распределенной базой данных (СУРБД).

Классификация (по доступу)

По способу доступа к данным базы данных разделяются на :

- **базы данных с локальным доступом**
- **базы данных с сетевым доступом**.

Для всех современных баз данных можно организовать сетевой доступ с многопользовательским режимом работы.

Классификация (по архитектуре)

Централизованные базы данных с сетевым доступом могут иметь следующую архитектуру:

- файл-сервер;
- клиент-сервер базы данных;
- сервер приложений - сервер базы данных (трехуровневая архитектура).

12 правил Кодда

Реляционная СУБД должна быть способна полностью управлять базой данных через ее реляционные возможности.

- **Информационное правило** - вся информация в реляционной БД (включая имена таблиц и столбцов) должна определяться строго как значения в таблицах.
- **Гарантированный доступ** - любое значение в реляционной БД должно быть гарантированно доступно для использования через комбинацию имени таблицы, значения первичного ключа и имени столбца
- **Поддержка пустых значений** (null value) - СУБД должна уметь работать с пустыми значениями (неизвестными или неиспользованными значениями), в отличие от значений по умолчанию и независимо для любых *доменов*.

5. **Онлайновый реляционный каталог** - описание БД и ее содержания должны быть представлены на логическом уровне как таблицы, к которым можно применять запросы, используя язык базы данных.
6. **Исчерпывающий язык управления данными** - по крайней мере, один из поддерживаемых языков должен иметь четко определенный синтаксис и быть всеобъемлющим. Он должен поддерживать описание структуры данных и манипулирование ими, правила целостности, авторизацию и транзакции
7. **Правило обновления представлений** (views) - все представления, теоретически обновляемые, могут быть обновлены через систему.

8. **Вставка, обновление и удаление** - СУБД поддерживает не только запрос на отбор данных, но и вставку, обновление и удаление
9. **Физическая независимость данных** - на программы-приложения и специальные программы логически не влияют изменения физических методов доступа к данным и структур хранилищ данных.
10. **Логическая независимость данных** - на программы-приложения и специальные программы логически не влияют, в пределах разумного, изменения структур таблиц.
11. **Независимость целостности** - язык БД должен быть способен определять правила целостности. Они должны сохраняться в онлайновом справочнике, и не должно существовать способа их обойти.
12. **Независимость распределения** - на программы-приложения и специальные программы логически не влияет, первый раз используются данные или повторно.
13. **Неподрывность** - невозможность обойти правила целостности, определенные через язык базы данных, использованием языков низкого уровня

Терминология реляционной модели

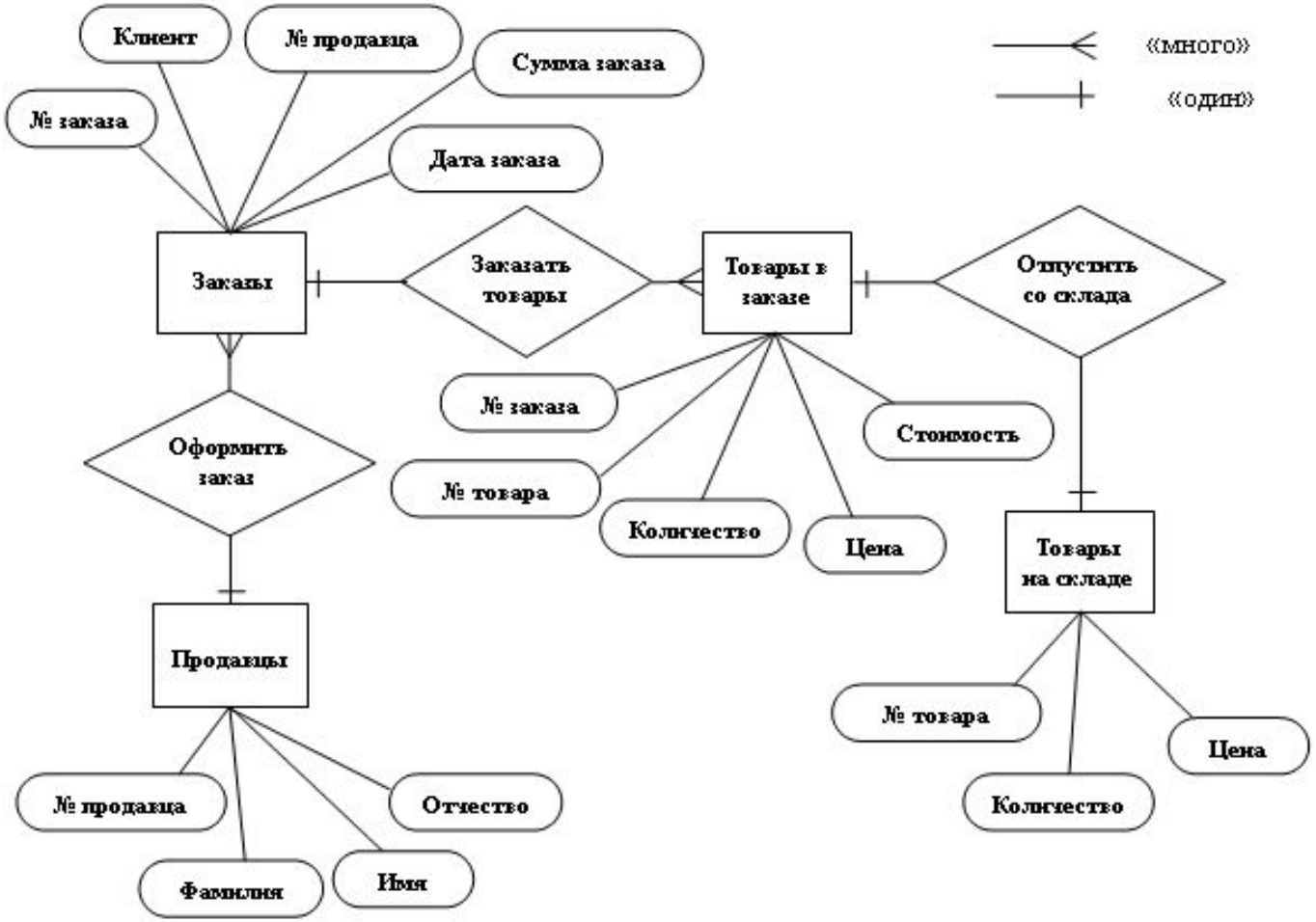
- ***Сущность*** - некоторый обособленный объект или событие, информацию о котором необходимо сохранять в базе данных, имеющий определенный набор свойств - *атрибутов*.
- Сущности могут быть как физические (реально существующие объекты: например, СТУДЕНТ, *атрибуты* - № зачетной книжки, фамилия, его факультет, специальность, № группы и т.д.),
- так и абстрактные (например, ЭКЗАМЕН, *атрибуты* - дисциплина, дата, преподаватель, аудитория и пр.).
- Для *сущностей* различают ее тип и экземпляр. Тип характеризуется именем и списком свойств, а экземпляр - конкретными значениями свойств.

Атрибуты *сущности* бывают:

- **Идентифицирующие и описательные**. Идентифицирующие *атрибуты* имеют уникальное значение для *сущностей* данного типа и являются потенциальными ключами. Они позволяют однозначно распознавать экземпляры *сущности*.
- Из потенциальных ключей выбирается один **первичный ключ** (ПК). В качестве ПК обычно выбирается потенциальный ключ, по которому чаще происходит обращение к экземплярам записи. ПК должен включать в свой состав минимально необходимое для идентификации количество *атрибутов*.
- Остальные ***атрибуты*** называются описательными.

- **Простые и составные.** Простой *атрибут* состоит из одного компонента, его значение неделимо. Составной *атрибут* является комбинацией нескольких компонентов, возможно, принадлежащих разным типам данных (например, адрес). Решение о том, использовать составной *атрибут* или разбивать его на компоненты, зависит от особенностей процессов его использования и может быть связано с обеспечением высокой скорости работы с большими базами данных.
- **Однозначные и многозначные** - могут иметь соответственно одно или много значений для каждого экземпляра сущности.
- **Основные и производные**. Значение основного *атрибута* не зависит от других *атрибутов*. Значение производного *атрибута* вычисляется на основе значений других *атрибутов* (например, возраст человека вычисляется на основе даты его рождения и текущей даты

Диаграмма «сущность-связи»



Нормализация

- ***Нормализация*** - это формальный метод анализа *отношений* на основе их первичного ключа и существующих связей. Ее задача - это замена одной схемы (или совокупности *отношений*) БД другой схемой, в которой *отношения* имеют более простую и регулярную структуру.

При работе с реляционной моделью для создания *отношений* приемлемого качества достаточно выполнения требований первой нормальной формы.

1НФ

Первая нормальная форма (1НФ) связана с понятиями простого и сложного *атрибутов*. Простой ***атрибут*** **-** это *атрибут*, значения которого атомарны (т.е. неделимы). Сложный *атрибут* может иметь значение, представляющее собой объединение нескольких значений одного или разных доменов. В первой нормальной форме устраняются повторяющиеся *атрибуты* или группы *атрибутов*, т.е. производится выявление неявных *сущностей*, "замаскированных" под *атрибуты*.

Отношение приведено к 1НФ, если все его атрибуты - простые, т.е. значение *атрибута* не должно быть множеством или повторяющейся группой.Для приведения таблиц к 1НФ необходимо разбить сложные *атрибуты* на простые, а многозначные *атрибуты* вынести в отдельные *отношения*.

2 НФ

Вторая нормальная форма (2НФ) применяется к *отношениям* с составными ключами (состоящими из двух и более *атрибутов*) и связана с понятиями функциональной зависимости.

Если в любой момент времени каждому значению *атрибута* A соответствует единственное значение *атрибута* B, то B функционально зависит от A (A B). Атрибут (группа ***атрибутов***) A называется детерминантом.

Во второй нормальной форме устраняются *атрибуты*, зависящие только от части уникального ключа. Эта часть уникального ключа определяет отдельную *сущность*.

Отношение находится во 2НФ, если оно приведено к 1НФ и каждый неключевой атрибут функционально полно зависит от составного первичного ключа.

3 НФ

Третья нормальная форма (3НФ) связана с понятием транзитивной зависимости. Пусть A, B, C - *атрибуты* некоторого *отношения*. При этом A B и B C, но обратное соответствие отсутствует, т.е. C не зависит от B или B не зависит от A. Тогда говорят, что C транзитивно зависит от A (A C).

В третьей нормальной форме устраняются *атрибуты*, которые зависят от *атрибутов*, не входящих в уникальный ключ. Эти *атрибуты* являются основой отдельной *сущности*.

Отношение находится в 3НФ, если оно находится во 2НФ и не имеет атрибутов, не входящих в первичный ключ и находящихся в транзитивной зависимости от первичного ключа.

Существуют также нормальная форма Бойса-Кодда (НФБК), 4НФ и 5НФ. Однако наибольшее значение имеет 1НФ, т.к. последующие НФ связаны с понятиями о составных ключах и сложных зависимостях от ключей, а на практике встречаются обычно более простые случаи.

Моделирование структуры базы данных при помощи алгоритма *нормализации* имеет серьезные недостатки:

Методика *нормализации* предполагает первоначальное размещение всех *атрибутов* проектируемой предметной области в одном *отношении*, что является очень неестественной операцией. Интуитивно разработчик сразу проектирует несколько *отношений* в соответствии с обнаруженными *сущностями*. Даже если совершить насилие над собой и создать одно или несколько *отношений*, включив в них все предполагаемые *атрибуты*, то совершенно неясен смысл полученного *отношения*.

Невозможно сразу определить полный список *атрибутов*. Пользователи имеют привычку называть разными именами одни и те же вещи или наоборот, называть одними именами разные вещи.

Для проведения процедуры *нормализации* необходимо выделить зависимости *атрибутов*, что тоже очень нелегко.

Ссылочная целостность

Соблюдение условий ссылочной целостности в реляционной базе данных

- **Правило соответствия внешних ключей первичным** - основное правило соблюдения условий ссылочной целостности. Для каждого значения внешнего ключа должно существовать соответствующее значение первичного ключа в родительской таблице

- Ссылочная целостность может нарушиться в результате операций вставки (добавления), обновления и удаления записей в таблицах. В определении ссылочной целостности участвуют две таблицы - родительская и дочерняя, для каждой из них возможны эти операции, поэтому существует шесть различных вариантов, которые могут привести либо не привести к нарушению ссылочной целостности.

Для родительской таблицы:

- **Вставка.** Возникает новое значение первичного ключа. Существование записей в родительской таблице, на которые нет ссылок из дочерней таблицы, допустимо, операция **не нарушает ссылочной целостности**.
- **Обновление**. Изменение значения первичного ключа в записи **может привести к нарушению ссылочной целостности**.
- **Удаление**. При удалении записи удаляется значение первичного ключа. Если есть записи в дочерней таблице, ссылающиеся на ключ удаляемой записи, то значения внешних ключей станут некорректными. Операция **может привести к нарушению ссылочной целостности.**

Для дочерней таблицы:

- **Вставка.** Нельзя вставить запись в дочернюю таблицу, если для новой записи значение внешнего ключа некорректно. Операция **может привести к нарушению ссылочной целостности**.
- **Обновление.** При обновлении записи в дочерней таблице можно попытаться некорректно изменить значение внешнего ключа. Операция **может привести к нарушению ссылочной целостности.**
- **Удаление.** При удалении записи в дочерней таблице **ссылочная целостность не нарушается**.

Таким образом, ссылочная целостность в принципе может быть нарушена при выполнении одной из четырех операций:

- Обновление записей в родительской таблице.
- Удаление записей в родительской таблице.
- Вставка записей в дочерней таблице.
- Обновление записей в дочерней таблице.