

# Базы данных и СУБД

Курс лекций

Радыгин В.Ю.

[radigin@msiu.ru](mailto:radigin@msiu.ru)

# Рекомендуемая литература

1. "Системы баз данных полный курс", Г.Гарсиа-Молина, Дж. Ульман, Дж. Уидом, Вильямс, 2003
2. "Введение в системы баз данных", К.Дж. Дейт, Вильямс, 2001 (или любое другое издание)
3. "PostgreSQL для профессионалов", Дж. Уорсли, Дж. Дрейк, Питер, 2003
4. "Основы PostgreSQL", Р. Стоунз, Н. Мэттью, СПб., 2002

# Базы данных и СУБД

Точное понятие базы данных не существует. В разных ситуациях под этим термином подразумевают разные вещи. Мы попробуем дать следующее абстрактное определение:

**База данных** – это набор порций информации, существующий в течении длительного времени.

Это понятие очень обширное поэтому в нашем случае этот термин будет немного усложнен. Мы будем работать лишь с компьютерными базами данных, т.е. такими, взаимодействие с которыми осуществляется посредством специального управляющего программного обеспечения – **СУБД**.

# СУБД

Для создания нормальной базы данных нужны три основные составляющие:

- сами данные;
- аппаратное обеспечение;
- программное обеспечение.

Под программным обеспечением мы будем подразумевать все те средства, которые позволяют конечным пользователям получать доступ к данным и редактировать их. Кроме того это программное обеспечение может решать и другие задачи, такие как например: обеспечение безопасности данных, одновременный доступ и т.д. Весь этот комплекс программ обычно называют **системой управления базами данных – СУБД**.

# Требования, предъявляемые к современному СУБД

Современная СУБД должна предоставлять возможность работы пользователя:

- на ЭВМ разной архитектуры с установленными на них различными операционными системами;
- в компьютерных сетях разных типов, работающих по различным протоколам;
- с различными графическими и символьными системами представления информации.

# Функции СУБД

Современная СУБД должна обеспечивать очень широкий набор функций. Вот некоторые из них:

- поддержка логической модели данных (определение данных и оперирование с ними);
- восстановление данных (транзакции, журналирование, контрольные точки);
- управление одновременным доступом;
- конфиденциальность данных (безопасность с точки зрения несанкционированного доступа);
- самостоятельная оптимизация выполнения операции;
- другие функции (администрирование, статистика, распределение данных и т.д.).

# Модели данных

Большинство объектов физического мира невероятно 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.

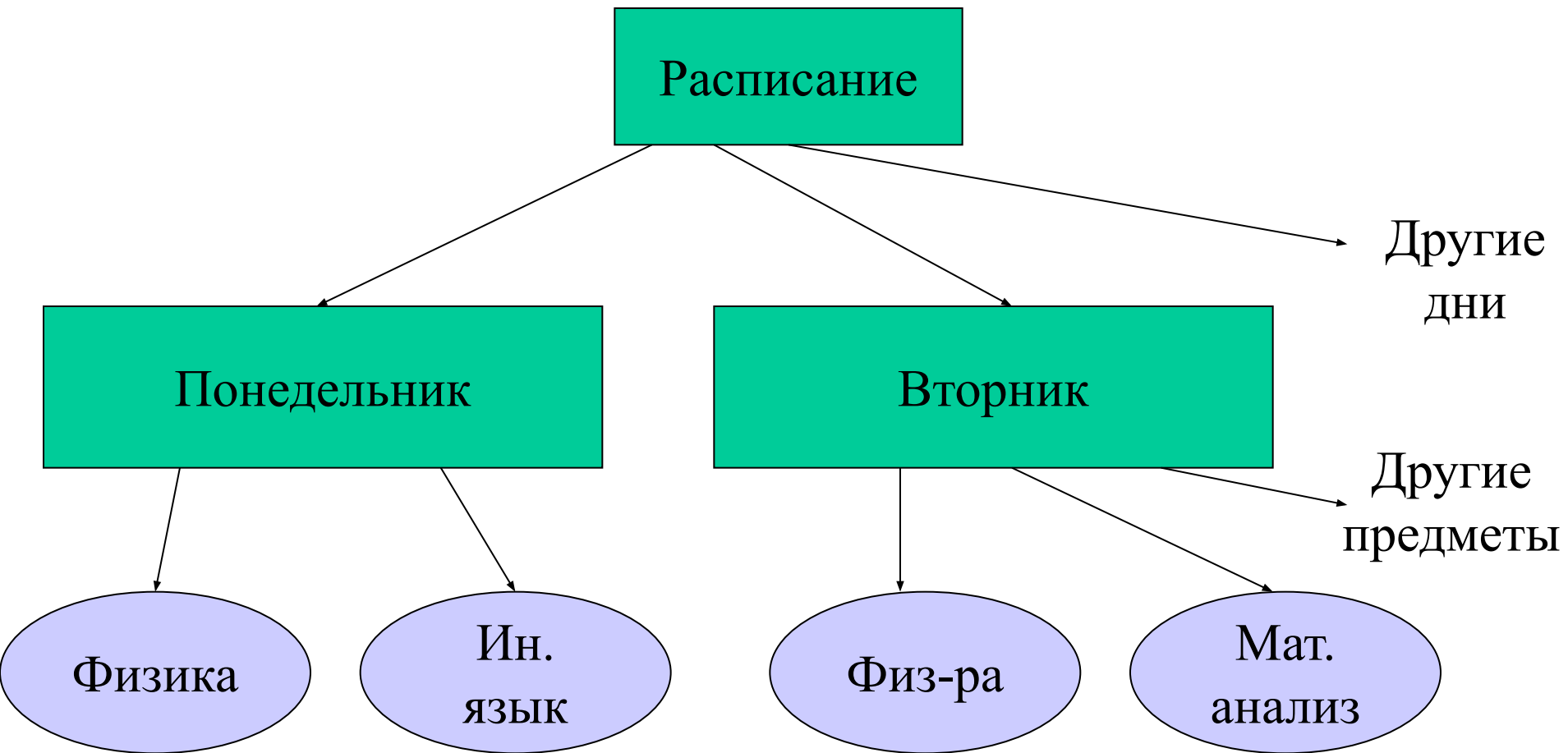
Сейчас актуальны в основном реляционная и объектная модели. Причем хотя наибольшее распространение имеет первая из них, обе модели имеют применение в современном программном обеспечении.



# Иерархическая модель

Модель представляет собой неоднородное дерево (группу деревьев), каждый узел которого обозначает некоторую сущность. Узел может иметь только одного родителя и 0 или более порожденных узлов. Каждая связь “родитель-ребенок” означает наличие отношения 1:М между соответствующими сущностями.

# Пример



# Некоторые определения

Пусть какой-то класс объектов из жизни может быть представлен набором свойств  $(A_1, A_2 \dots A_N)$ . Тогда набор значений, описывающий определенного представителя данного класса мы будем называть **записью**. А каждое из свойств  $A_1, A_2 \dots A_N$  – **атрибутом**.

Пример: пусть нашим объектом будут собаки. Будем считать, что для наших целей достаточно знать имя, породу, возраст и хозяина пса. Тогда имя, порода, возраст и хозяин – это атрибуты, а например набор значений (Бобик, Дворняжка, 10, Дядя Федя) – это запись.

# Некоторые определения

Атрибут  $A$  функционально зависит от атрибута  $B$ , если по заданному значению атрибута  $B$  можно однозначно определить значение атрибута  $A$ .

Пример: пусть у каждого хозяина будет только одна собака. Тогда мы можем сказать, что атрибут имя функционально зависит от атрибута хозяин.

# Некоторые определения

Атрибут или набор атрибутов от которого функционально зависят все остальные атрибуты записи называется **ключом**.

В нашем предыдущем примере ключом был хозяин.

Если атрибут (набор атрибутов)  $B$  функционально зависит от набора атрибутов  $(A_1, A_2)$ , но не зависит функционально от каждого из них в отдельности, то атрибуты  $A_1, A_2$  образуют **составной ключ**.

# Недостатки иерархической модели

- неадекватность отображения взаимосвязей между объектами;
- частая необходимость в искусственной избыточности;
- проблемы поиска по дереву в обратном направлении.

Несложно понять, что в иерархической модели наряду с ее простотой имеется функциональная зависимость неключевых атрибутов от пути в дереве.

# Сетевая модель

Сетевая модель подчиняется следующим правилам:

1. Один и тот же объект может находиться с другими объектами более чем в одном отношении.
2. Допускаются отношения N:M.
3. Нет иерархии.
4. Запись может не находиться ни в одном отношении с другими объектами.

В общем случае сетевая модель представляет собой произвольно ориентированный граф, возможно с петлями, узлы которого обозначают типы объектов, а дуги связи между ними.

# Внутренняя архитектура сетевой модели данных

Основные правила построения сетевых баз данных были сформулированы в 1971 рабочей группой по базам данных DTBG ассоциации по языкам систем обработки данных CODASYL. Они носят название CODASYL DTBG.

Сетевая СУБД оперирует следующими объектами: элементами данных, агрегатами данных, записями, и наборами.

В настоящее время широко известна сетевая СУБД db\_Vista, работающая на персональных компьютерах в DOS и Windows.



# Элементы данных и агрегаты данных

Элемент данных (ЭД) - наименьшая поименованная единица данных (аналог поля), с помощью которой осуществляется построение всех остальных структур. Имя элемента данных используется для его идентификации в схеме структуры данных более высокого уровня. Значение элемента данных может быть одного из встроенных типов: числовым (целым, вещественным) или нечисловым (символьным, логическим). В некоторых приложениях может использоваться «неопределенное» значение элемента данных и говорит о том, что значение соответствующего свойства объекта не определено в БД.

# Элементы данных и агрегаты данных

Агрегат данных - поименованная совокупность элементов данных внутри записи, которую можно рассматривать как единое целое. Имя агрегата используется для его идентификации в схеме структуры данного более высокого уровня. Агрегат данных может быть простым если состоит только из элементов данных, и составным если включает в свой состав другие агрегаты.

# Виды агрегатов данных

Персона			
Фамилия	Имя	Отчество	Пол

Простой агрегат  
данных

Составной агрегат  
данных

Кабинет				
Название	Персона			
	Фамилия	Имя	Отчество	Пол

# Виды агрегатов данных

Различают агрегаты типа «вектор» и типа «повторяющаяся группа». Агрегат, повторяющийся компонент которого является простым элементом данных, называется «вектором». Агрегат, повторяющийся компонент которого представлен совокупностью данных, называется повторяющейся группой. В повторяющуюся группу могут входить отдельные элементы данных, векторы, агрегаты или повторяющиеся группы.

Максимальное количество элементов данных (или агрегатов данных) для вектора и повторяющейся группы ограничено и задается при спецификации схемы записи.

# Виды агрегатов данных

## Будильник по дням недели

Время	Время	Время	Время	Время	Время	Время
-------	-------	-------	-------	-------	-------	-------

Вектор

## Отдел

Отчество	Имя	Фамилия	Должность	Отчество	Имя	Фамилия	Должность	Отчество	Имя	Фамилия	Должность
----------	-----	---------	-----------	----------	-----	---------	-----------	----------	-----	---------	-----------

Повторяющаяся  
группа

# Запись

Запись - поименованная совокупность элементов данных и/или агрегатов. Иными словами, запись - это агрегат, который не входит в состав никакого другого агрегата и может иметь сложную иерархическую структуру, поскольку допускается многократное применение агрегации. Имя записи используется для идентификации типа записи в схемах типов структур более высокого уровня. Экземпляр записи – это набор конкретных значений, которые принимают все входящие в запись агрегаты и элементы данных.

# Набор

Набор записей - поименованная совокупность записей, образующая двухуровневую иерархическую структуру подчинения: каждый тип набора представляет собой отношение (связь) между двумя или несколькими типами записей. В силу иерархичности набора для каждого его типа один тип записи объявляется «владельцем», тогда остальные типы записей - «члены», т.е. имеют место «запись-владелец» и «запись-член (набора)». Каждый экземпляр набора должен содержать только один экземпляр записи-владельца и любое количество экземпляров записей-членов.

Различают сингулярные, двухтипные и множественные наборы. Сингулярный набор содержит только тип записи члена. Записью владельцем выступает система. Двухтипный набор подразумевает одну запись с типом-владельцем и много записей с типами-членами. Множественный набор подразумевает несколько типов записей членов набора. Тогда экземпляр набора включает в себя один экземпляр записи владельца и все связанные с ним экземпляры записей-членов заданных типов.

# Виды наборов

Сингулярный

Константа Система\_Константа

Название	Значение
----------	----------

Руководитель\_Подчиненные  
Персона

--	--	--

Персона

Фамилия	Имя	Отчество
---------	-----	----------

Должность

Название	Обязанность	Договор
----------	-------------	---------

Зарплата

Сумма
-------

Множественный



# Реляционная модель

В реляционной модели все данные представлены в виде таблиц. Строки таблиц – это отдельные записи, а колонки – это атрибуты. Каждая таблица представляет набор объектов (записей) удовлетворяющих определенному отношению (соответствующему таблице). Отсюда и название – реляционная модель. Значения атрибутов таблиц удовлетворяют некоторым заранее предопределенным доменам – областям определения. Реляционная база данных представляет из себя набор таких отношений-таблиц.

Более подробно данную модель мы рассмотрим позже.

# Объектная модель

Объектную модель иногда называют также объектно-ориентированной моделью.

Основные понятия с которыми оперирует данная модель:

- объекты, обладающие внутренней структурой и однозначно идентифицируемые уникальным внутрисистемным ключом;
- классы, являющиеся по сути типами объектов;
- операции над объектами одного или разных типов называемые методами;
- инкапсуляция структурного и функционального описания объектов, позволяющая разделять внутреннее и внешнее описания;
- наследуемость внешних свойств объектов на основе отношения “класс-подкласс”.

# Недостатки и преимущества объектной модели

## Преимущества:

- Возможность определять сколь угодно сложные типы данных;
- Наличие наследуемости свойств объектов;
- Повторное использование программного описания типов объектов при обращении к другим типам, на них ссылающимся.

## Недостатки:

- Отсутствие строгой математической модели;
- Более сложные механизмы поиска и взаимодействия.

# История баз данных

До конца 1960-х

файловые системы.

Конец 1960-х

первые коммерческие СУБД (системы бронирования авиабилетов, банковские системы и т.д.). В основном иерархическая и сетевая модели данных.

1970 год

Э.Ф. Кодд вводит реляционную модель данных.

1971 год CODASYL сформулировал основные принципы сетевых СУБД.

Конец 1980-х - настоящее время

развитие объектной модели данных.