Лекция 6.1, 6.2.Основы баз данных(4 ч.)

План

1. Основные понятия. Модели данных.
2. Реляционная модель данных
3. Нормализация данных
4. Язык SQL

1. Основные понятия. Модели данных.

1. Основные понятия. Модели данных.

Базу данных (БД) можно определить как унифицированную совокупность данных, совместно используемую различными задачами в рамках некоторой единой автоматизированной информационной системы (ИС).

Теория управления базами данных как самостоятельная дисциплина начала развиваться приблизительно с начала 50-х годов двадцатого столетия. За это время в ней сложилась определенная система фундаментальных понятий. Приведем некоторые из них.

1. Основные понятия. Модели данных.

Предметной областью принято называть часть реального мира, подлежащую изучению с целью организации управления в этой сфере и последующей автоматизации процесса управления.

Объектом называется элемент информационной системы, сведения о котором хранятся в базе данных. Иногда объект также называют **сущностью** (от англ, entity).

Классом объектов называют их совокупность, обладающую одинаковым набором свойств.

Атрибут - это информационное отображение свойств объекта. Каждый объект характеризуется некоторым набором атрибутов.

1. Основные понятия. Модели данных.

Ключевым элементом данных называются такой атрибут (или группа атрибутов), который позволяет определить значения других элементов-данных.

Запись данных (англ, эквивалент record) - это совокупность значений связанных элементов данных.

Первичный ключ - это атрибут (или группа атрибутов), который уникальным образом идентифицируют каждый экземпляр объекта (запись).

Вторичным ключом называется атрибут (или группа атрибутов), значение которого может повторяться для нескольких записей (экземпляров объекта). Прежде всего, вторичные ключи используются в операциях поиска записей.

1. Основные понятия. Модели данных.

Процедуры хранения данных в базе должны подчиняться некоторым общим принципам, среди которых в первую очередь следует выделить:

- целостность и непротиворечивость данных, под которыми понимается как физическая сохранность данных, так и предотвращение неверного использования данных, поддержка допустимых сочетаний их значений, защита от структурных искажений и несанкционированного доступа;
- минимальная избыточность данных обозначает, что любой элемент данных должен храниться в базе в единственном виде, что позволяет избежать необходимости дублирования операций, производимых с ним.

1. Основные понятия. Модели данных.

Программное обеспечение, осуществляющее операции над базами данных, получило название СУБД - **система управления базами данных**. Очевидно, что его работа должна быть организована таким образом, чтобы выполнялись перечисленные принципы.

1. Основные понятия. Модели данных.

Модели организации данных

Набор принципов, определяющих организацию логической структуры хранения данных в базе, получил название **модели данных**. Модели баз данных определяются тремя компонентами:

- допустимой организацией данных;
- ограничениями целостности;
- множеством допустимых операций.

В теории систем управления базами данных выделяют модели четырех основных типов: иерархическую, сетевую, реляционную и объектно-реляционную.

1. Основные понятия. Модели данных.

Иерархическая модель данных

Терминологической основой для иерархической и сетевой моделей являются понятия: атрибут, агрегат и запись. *Под атрибутом* (элементом данных) понимается наименьшая поименованная структурная единица данных.

Поименованное множество атрибутов может образовывать *агрегат данных*. В некоторых случаях отдельно взятый агрегат может состоять из множества экземпляров однотипных данных, или, как еще говорят, являться множественным элементом. Наконец, *записью* называют составной агрегат, который не входит в состав других агрегатов.

1. Основные понятия. Модели данных.

Рассмотрим на примерах введенные понятия.

Пусть какой-то класс объектов из жизни может быть представлен набором свойств (A_1, A_2 … A_N). Тогда набор значений, описывающий определенного представителя данного класса это запись. А каждое из свойств A_1, A_2 … A_N – атрибут.

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.

Атрибут А функционально зависит от атрибута В, если по заданному значению атрибута В можно однозначно определить значение атрибута А.

1. Основные понятия. Модели данных.

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

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

В нашем предыдущем примере ключом был хозяин.

1. Основные понятия. Модели данных.

Если атрибут (набор атрибутов) В 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.

В иерархической модели все записи, агрегаты и атрибуты базы данных образуют иерархически организованный набор, то есть такую структуру, в которой все элементы связаны отношениями подчиненности, и при этом любой элемент может подчиняться только одному какому-нибудь другому элементу. Такую форму зависимости удобно изображать с помощью древовидного графа (схемы, состоящей из точек и стрелок, которая связна и не имеет циклов). Пример иерархической структуры базы данных приведен на рис. 1.

1. Основные понятия. Модели данных.

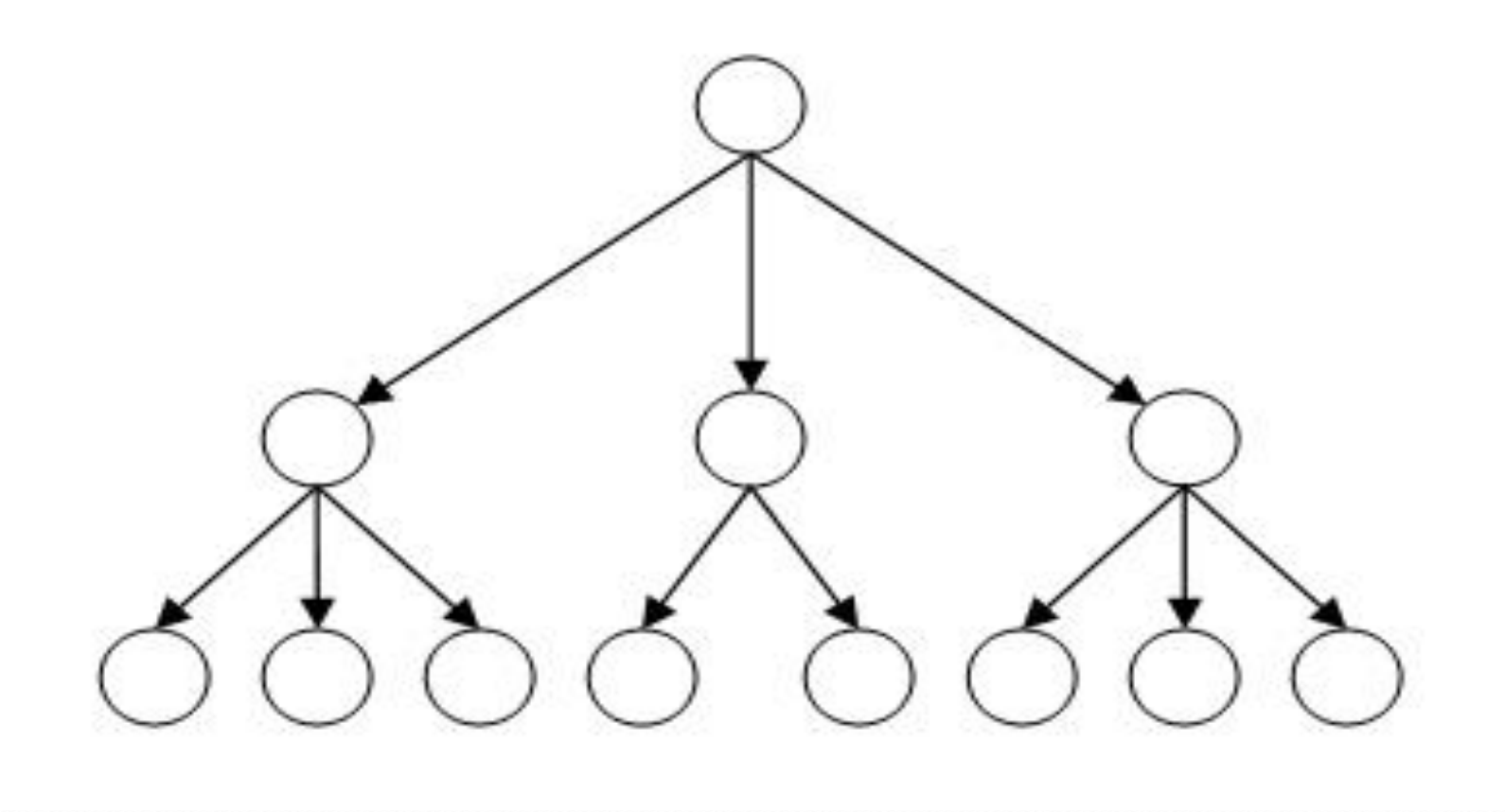

Рис. 1. Схема иерархической модели данных

1. Основные понятия. Модели данных.

Типичным представителем семейства баз данных, основанных на иерархической модели, является Information Management System (IMS) фирмы IBM, первая версия которой появилась в 1968 г.

Среди основных недостатков иерархической модели можно выделить следующие:

- 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.

1. Основные понятия. Модели данных.

Сетевая модель данных

Концепция сетевой модели данных связана с именем Ч. Бахмана. 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 (рис. 2).

1. Основные понятия. Модели данных.

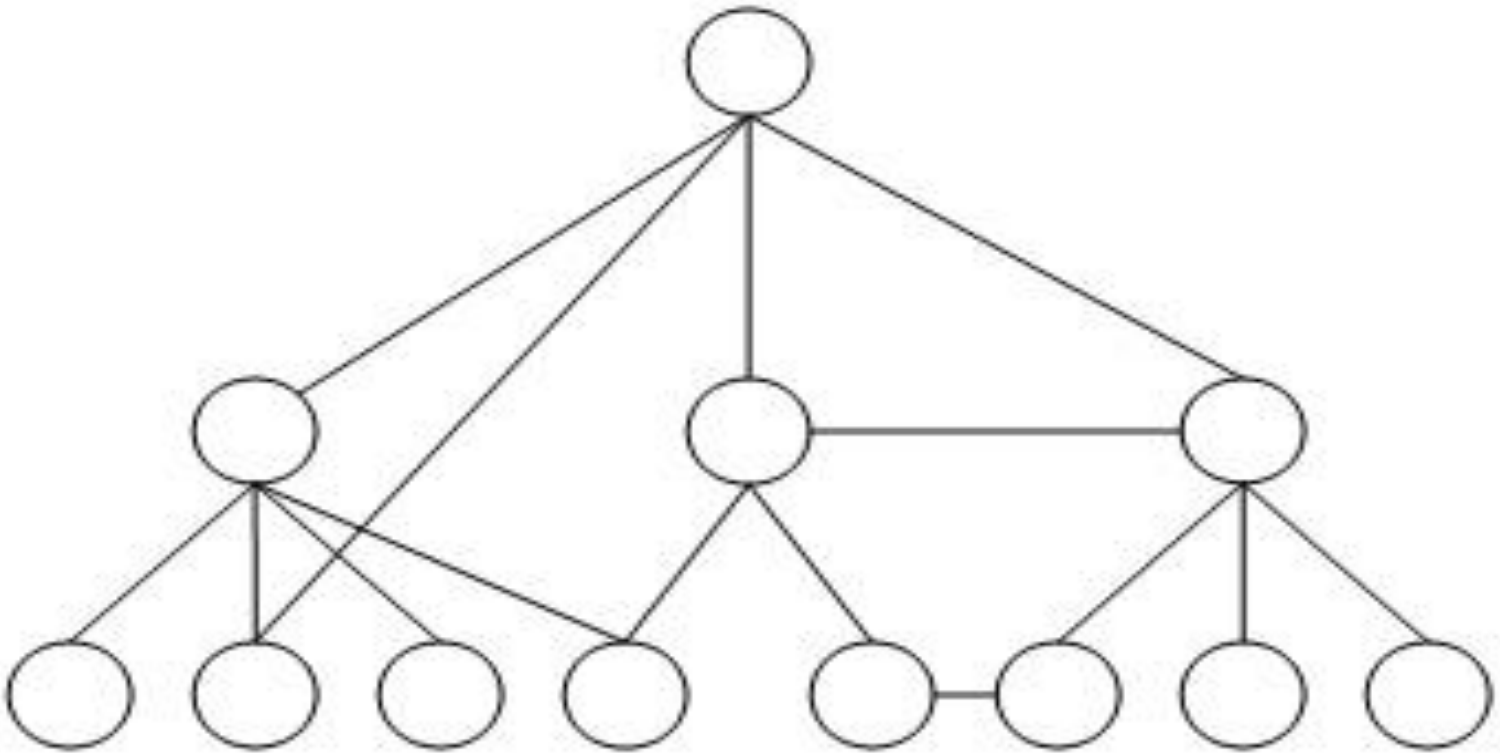

Рис. 2. Схема сетевой модели данных

1. Основные понятия. Модели данных.

Сетевая модель подчиняется следующим правилам:

- один и тот же объект может находиться с другими объектами более чем в одном отношении.
- допускаются отношения N:M.
- нет иерархии.
- 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.

1. Основные понятия. Модели данных.

Внутренняя архитектура сетевой модели данных:

- Основная рабочая единица сетевой модели данных – это так называемый набор. Набор характеризуется своим типом и множеством экземпляров.
- Тип набора – это тип владельца набора и тип записей набора.
- Экземпляр состоит из владельца и о или более записей.
- Тип записи может быть владельцем или членом нескольких типов наборов.
- Экземпляр записи может быть владельцем не более одного набора одного типа и членом не более одного набора данного типа.
- Существует одна предопределенная запись (system).
- Навигация осуществляется путем перехода по спискам, когда из записи одного списка мы получаем владельца нового списка.

1. Основные понятия. Модели данных.

Сетевая БД состоит из набора записей и набора связей между этими записями, точнее, из набора экземпляров записей заданных типов (из допустимого набора типов) и набора экземпляров из заданного набора типов связи.

Примером системы управления данными с сетевой организацией является Integrated Database Management System (IDMS) компании Cullinet Software Inc., разработанная в середине 70-х годов. Она предназначена для использования на "больших" вычислительных машинах. Архитектура системы основана на предложениях Data Base Task Group (DBTG), Conference on Data Systems Languages (CODASYL), организации, ответственной за определение стандартов языка программирования Кобол.

1. Основные понятия. Модели данных.

Среди достоинств систем управления данными, основанных на сетевой модели, могут быть названы их компактность и, как правило, высокое быстродействие, а среди недостатков - неуниверсальность, высокая степень зависимости от конкретных данных.

1. Основные понятия. Модели данных.

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”.

1. Основные понятия. Модели данных.

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.

2. Реляционная модель данных

2. Реляционная модель данных

Концепции реляционной модели впервые были сформулированы в работах американского ученого Э. Ф. Кодда. Откуда происходит ее второе название - модель Кодда.

В реляционной модели объекты и взаимосвязи между ними представляются с помощью таблиц (рис. 3). Для ее формального определения используется фундаментальное понятие отношения. Собственно говоря, термин "реляционная" происходит от английского relation - отношение.

Атрибуты отношения – это столбцы таблицы.

Записи объектов, удовлетворяющих данному отношению, - это строки таблицы.

2. Реляционная модель данных

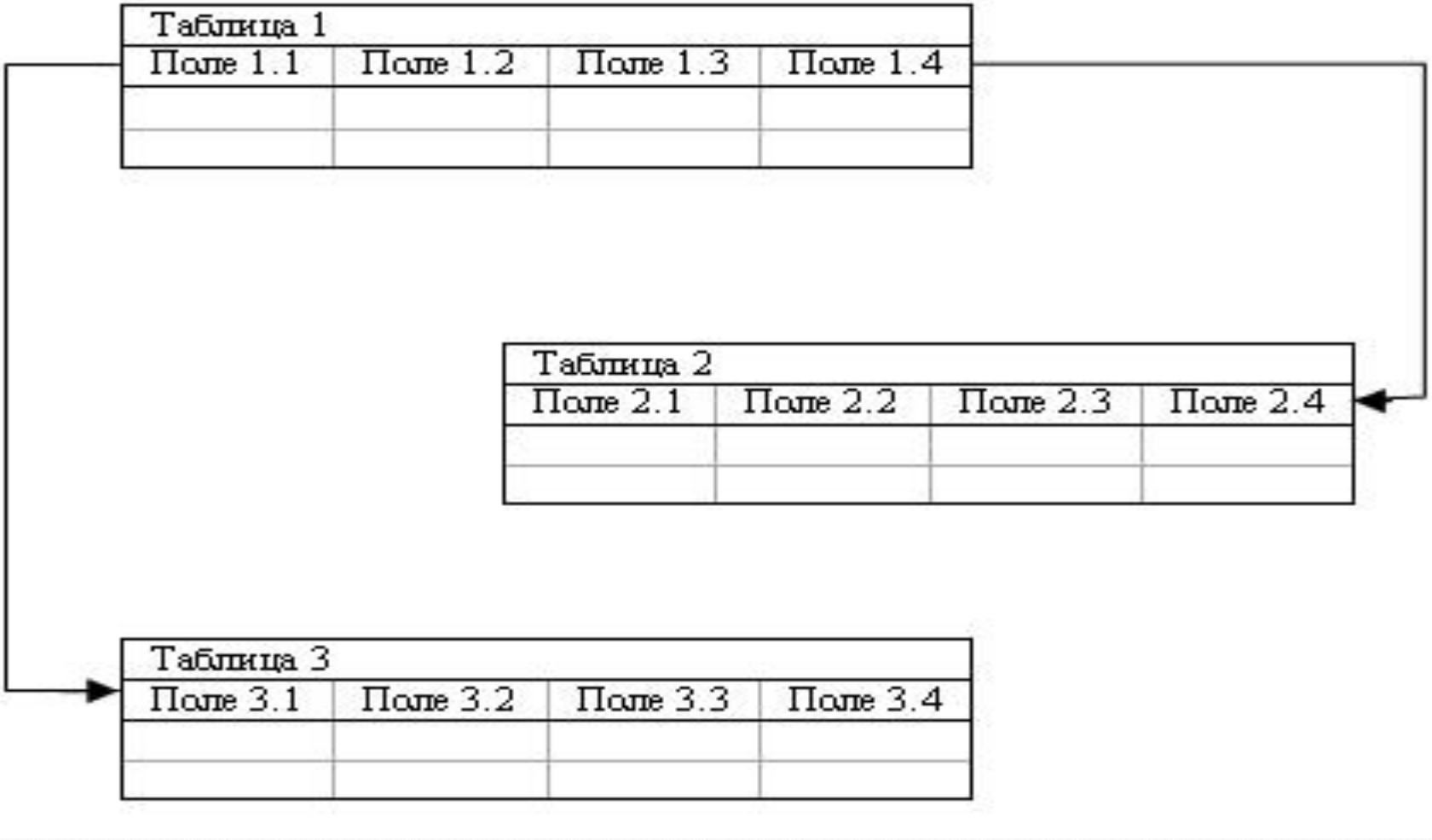

Рис. 3. Схема реляционной модели данных

2. Реляционная модель данных

Записи в реляционной модели принято называть *кортежами*.

Кардинальность отношения – это количество кортежей (записей) удовлетворяющих данному отношению.

Степень отношения – это количество атрибутов отношения.

Домен, которому принадлежит атрибут – совокупность допустимых значений, которые может принимать данный атрибут.

2. Реляционная модель данных

Реляционная алгебра

Основные действия, которые можно делать с реляционными отношениями и их кортежами, описаны реляционной алгеброй.

Операции реляционной алгебры

Выборка

Пусть задано отношение A(X, Y, ...) и операция Θ: (X,Y)→ {T|F}. Тогда Θ-выборкой из отношения A по атрибутам X, Y называется новое отношение B, с теми же атрибутами и содержащие все такие кортежи отношения A, для атрибутов X, Y которых Θ(X, Y) = T.

2. Реляционная модель данных

Проекция

Пусть задано отношение A(X, Y, …). Тогда проекцией отношения A по атрибутам X, Y называется новое отношение B, состоящее только из атрибутов X и Y и содержащие все кортежи вида {X:x, Y:y}, такие, что множество кортежей отношения A содержит хотя бы один кортеж вида {X:x, Y:y, ….}.

2. Реляционная модель данных

Произведение

Пусть заданы отношение $A(A_1, A_2, \ldots A_n)$ и отношение $B(B_1, B_2, \ldots B_n)$. Тогда произведением отношения A на отношение B называется новое отношение C, состоящее только из множества атрибутов $A_1, A_2, \ldots A_n, B_1, B_2, \ldots B_n$ и содержащее все кортежи, полученные путем дописывания к каждому кортежу отношения A каждый кортеж отношения B.

2. Реляционная модель данных

Объединение

Пусть заданы отношение $A(X_1, X_2, \ldots X_n)$ и отношение $B(X_1, X_2, \ldots X_n)$. Тогда объединением отношении A и B называется новое отношение C, состоящее из тех же атрибутов $X_1, X_2, \ldots X_n$, и содержащее все такие кортежи k, что k принадлежит множеству кортежей отношения A или множеству кортежей отношения B.

2. Реляционная модель данных

Пересечение

Пусть заданы отношение $A(X_1, X_2, \ldots X_n)$ и отношение $B(X_1, X_2, \ldots X_n)$. Тогда пересечением отношении A и B называется новое отношение C, состоящее из тех же атрибутов $X_1, X_2, \ldots X_n$, и содержащее все такие кортежи k, что k принадлежит множеству кортежей отношения A и множеству кортежей отношения B.

2. Реляционная модель данных

Разность

Пусть заданы отношение $A(X_1, X_2, \ldots X_n)$ и отношение $B(X_1, X_2, \ldots X_n)$. Тогда разностью отношении A и B называется новое отношение C, состоящее из тех же атрибутов $X_1, X_2, \ldots X_n$, и содержащее все такие кортежи k, что k принадлежит множеству кортежей отношения A и не принадлежит множеству кортежей отношения B.

2. Реляционная модель данных

Естественное соединение

Пусть заданы отношение $A(A_1, A_2, \ldots A_n, X, \ldots Y)$ и отношение $B(X, Y, B_1, B_2, \ldots B_m)$. Тогда естественным соединением отношения A и отношения B называется новое отношение C, состоящее только множества атрибутов $A_1, A_2, \ldots A_n, X \ldots Y, B_1, B_2, \ldots B_m$ и содержащее все кортежи, полученные путем дописывания к каждому кортежу отношения $A - (A_1 : a_1, A_2 : a_2, \ldots A_n : a_n, X : x, \ldots Y : y)$ недостающей части каждого кортежа r отношения $B – (B_1 : b_1, B_2 : b_2, \ldots B_m : b_m)$ для таких k из A и r из B, что значения атрибутов X,Y у k и r совпадают.

2. Реляционная модель данных

Θ-соединение

Соединением отношений $A(A_1, A_2 \ldots A_n)$ и $B(B_1, B_2 \ldots B_n)$ по операции $\Theta: A_1 x A_2 x \ldots A_n x B_1 x B_2 \ldots B_n$ $\{T|F\}$ называется новое отношение, содержащее все кортежи, получаемые путем соединения кортежей из исходных A и B и удовлетворяющие условию $\Theta(A:B) = T$.

2. Реляционная модель данных

Пример

Заказы		
Id_клиента	Товар	Статус
1	Картошка	Доставлено
1	Огурцы	На складе
2	Помидоры	На складе

Клиенты	
Id	Имя
1	Вася Пупкин
2	Петя Васечкин

⊖ : Заказы.id_клиента == Клиенты.id &&
Заказы.Статус != Доставлено

Заказы ⊖ Клиенты				
Id_клиента	Товар	Статус	Id	Имя
1	Огурцы	На складе	1	Вася Пупкин
2	Помидоры	На складе	2	Петя Васечкин

2. Реляционная модель данных

Эквисоединение

Если операция Θ представляет из себя равенство каких-либо полей (наборов полей), то такое соединение называется эквисоединением.

Пример:

Θ : Заказы.Id_клиента = Клиента.Id

2. Реляционная модель данных

Деление

Результатом деления отношения $A(A_1, A_2 \dots A_n)$ на отношение $B(B_1, B_2 \dots B_n)$ по отношению $C(A_1, A_2 \dots A_n, B_1, B_2 \dots B_n)$ называется новое отношение, содержащее все такие кортежи x из отношения A для которых выполняется условие: для любого y, принадлежащего множеству кортежей B, отношение C содержит кортеж x:y.

2. Реляционная модель данных

Пример

A	
F1	F2
1	2
2	3

B	
F3	F4
1	1
2	2

C			
F1	F2	F3	F4
1	2	1	1
1	2	2	2
2	3	1	1
1	2	4	5

A divide by B per C	
F1	F2
1	2

2. Реляционная модель данных

Реляционная БД

Реляционная база данных – это набор реляционных отношений (таблиц) и только их. Никаким другим образом (переменные, массивы) данные не представлены.

Над реляционными таблицами можно осуществлять только операции, описанные реляционной алгеброй. Причем результатом всех операции также являются реляционные отношения.

2. Реляционная модель данных

Правила разработки реляционной СУБД

Основное (подразумеваемое правило). Система, которая рекламируется или провозглашается поставщиком как реляционная СУБД должна управлять базами данных исключительно способами, соответствующими реляционной модели.

1. Информационное правило. Вся информация, хранимая в реляционной базе данных, должна быть явно, на логическом уровне, представлена единственным образом: в виде значений в реляционных таблицах.
2. Правило гарантированного логического доступа. К каждому имеющемуся в реляционной базе атомарному значению должен быть гарантирован доступ с помощью указания имени реляционной таблицы, значения первичного ключа и имени столбца.

2. Реляционная модель данных

3. Правило наличия значения. В полностью реляционной СУБД должны иметься специальные индикаторы (отличные от пустой символьной строки или строки из одних пробелов и отличные от нуля или какого-либо другого числового значения) для выражения (на логическом уровне, систематично и независимо от типа данных) того факта, что значение отсутствует по меньшей мере по двум различным причинам: его действительно нет, либо оно неприменимо к данной позиции. СУБД должна не только отражать этот факт, но и распространять на такие индикаторы свои функции манипулирования данными независимо от их типа.

2. Реляционная модель данных

4. Правило динамического диалогового реляционного каталога. Описание базы данных выглядит логически как обычные данные, так что авторизованные пользователи и прикладные программы могут употреблять для работы с этим описанием тот же реляционный язык, что и при работе с обычными данными.

2. Реляционная модель данных

5. Правило полноты языка работы с данными. Сколько бы много в СУБД ни поддерживалось языков и режимов работы с данными, должен иметься по крайней мере один язык, выразимый в виде командных строк в некотором удобном синтаксисе, который бы позволял формулировать:

- Определение данных;
- Определение правил целостности;
- Манипулирование данными (в диалоги и из программы);
- Определение выводимых таблиц (в том числе возможности их модификации);
- Определение правил авторизации;
- Границы транзакций.

2. Реляционная модель данных

6. Правило модификации таблиц. В СУБД должен существовать корректный алгоритм, позволяющий автоматически для каждой таблицы определять во время ее создания, может ли она использоваться для вставки и удаления строк и какие из столбцов допускают модификацию, и заносящий полученную таким образом информацию в системный каталог.

2. Реляционная модель данных

7. Правило множественности операций. Возможность оперирования базовыми или выводимыми таблицами распространяется полностью не только на выдачу информации из БД, но и на вставку, модификацию и удаление данных.

8. Правило физической независимости. Диалоговые операторы и прикладные программы на логическом уровне не должны страдать от каких-либо изменений во внутреннем хранении данных или в методах доступа СУБД.

9. Правило логической независимости. Диалоговые операторы и прикладные программы на логическом уровне не должны страдать от таких изменений в базовых таблицах, которые сохраняют информацию и теоретически допускают неизменность этих операторов и программ.

2. Реляционная модель данных

10. Правило сохранения целостности. Диалоговые операторы и прикладные программы не должны изменяться при изменении правил целостности в БД (задаваемых языком работы с данными и хранимых в системном каталоге).

11. Правило независимости от распределенности. Диалоговые операторы и прикладные программы на логическом уровне не должны страдать от совершаемого физического разнесения данных (если первоначально СУБД работала с нераспределенными данными) или перераспределения (если СУБД действительно распределенная).

2. Реляционная модель данных

12. Правило ненарушения реляционного языка. Если в реляционной СУБД имеется язык низкого уровня (для работы с отдельными строками), он не должен позволять нарушать или “обходить” правила, сформулированные на языке высокого уровня (множественном) и занесенные в системный каталог.

2. Реляционная модель данных

Проектирование реляционной БД.

Модель описания данных “сущность-связь” (ER-модель)

Основные элементы:

- множества сущностей,
- атрибуты,
- связи.

2. Реляционная модель данных

Сущность – это абстрактный объект определенного вида. Набор однородных сущностей образует множество сущностей.

Множеству сущностей отвечает набор атрибутов, являющихся свойствами отдельных его представителей.

Связи – это отношения между двумя или большим числом множеств сущностей.

2. Реляционная модель данных

2. Реляционная модель данных

Пусть X и Y произвольные подмножества атрибутов отношения R. Говорят, что Y функционально зависимо (ФЗ) от X (X→Y), тогда и только тогда, когда по значению атрибутов X кортежа можно однозначно определить значение атрибутов Y. Или, говоря другими словами, для любого допустимого значения отношения R, если два кортежа совпадают по значению атрибута X, то они совпадают и по значению атрибута Y.

Замечание: в этом определении не говорится, что кортежей с парой {x, y} не может быть несколько.

В записи функциональной зависимости X→Y множество X мы будем называть детерминантом, а Y - зависимой частью

2. Реляционная модель данных

Функциональная зависимость называется тривиальной, если Y является подмножеством X.

Замечание: подмножество Y может быть и несобственным.

Некоторые функциональные зависимости подразумевают другие функциональные зависимости.

Например,если $\{S, P\} \rightarrow \{X, Y\}$, то

$\{S, P\} \rightarrow \{X\}$ и $\{S, P\} \rightarrow \{Y\}$.

2. Реляционная модель данных

Аксиомы Армстронга

Каждое множество ФЗ может быть несколько упрощено. Из него могут быть удалены не только тривиальные ФЗ, но и некоторые другие. Рассмотрим правила вывода для ФЗ (Аксиомы Армстронга).

- Рефлексивность: если B подмножество A, то $A \rightarrow B$
- Дополнение: $A \rightarrow B \Rightarrow \{A, C\} \rightarrow \{B, C\}$
- Транзитивность: $A \rightarrow B$ и $B \rightarrow C \Rightarrow A \rightarrow C$

2. Реляционная модель данных

Дополнительные правила

Хотя этих правил достаточно, но для удобства они могут быть расширены дополнительными правилами вывода.

- Самоопределение: $A \rightarrow A$
- Декомпозиции: $A \rightarrow \{B, C\} \Rightarrow A \rightarrow B$ и $A \rightarrow C$
- Объединение: $A \rightarrow B$ и $A \rightarrow C \Rightarrow A \rightarrow \{B, C\}$
- Композиция: $A \rightarrow B$ и $C \rightarrow D \Rightarrow \{A, C\} \rightarrow \{B, D\}$
- $A \rightarrow B$ и $C \rightarrow D \Rightarrow A \cup (C \setminus B) \rightarrow \{B, D\}$

2. Реляционная модель данных

Замыкание множества

Множество всех функциональных зависимостей, которые можно вывести из заданного множества функциональных зависимостей - S по этим правилам, называется замыканием множества S (S+).

Неприводимые слева ФЗ

ФЗ называется неприводимой слева, если ни один атрибут из детерминанта не может быть опущен без изменения замыкания множества полученного из этой ФЗ.

2. Реляционная модель данных

Неприводимое множество ФЗ

Множество ФЗ S называется неприводимым если:

- Правая часть всех ФЗ содержит только один атрибут.
- Каждый детерминант является неприводимым слева.
- Ни одна ФЗ не может быть удалена из S без изменения S+.

3. Нормализация данных

3. Нормализация данных

Нормализация данных - это разложение отношений на большее количество более простых таблиц.

Цели нормализации

- Основной целью теории нормальных форм первоначально была экономия места на диске.
- Данные должны быть устроены так, чтобы при их редактировании или удалении, необходимо было исправлять только в одном месте БД.
- Группировка данных по содержанию (Каждая таблица - определенная тематика).
- Принцип модульности (Несколько унифицированных независимых блоков).

3. Нормализация данных

Примеры

1НФ

Преподаватели-предметы	
Преподаватель	**Предмет**
Иванов	Физика
Иванов	Химия
Петров	Физ-ра
Петров	Ин-яз

Не удовлетворяет 1НФ

Преподаватели-предметы	
Преподаватель	**Предмет**
Иванов	Физика Химия
Петров	Физ-ра Ин-яз

3. Нормализация данных

Декомпозиция

Основой нормализации является процесс разбиения - или декомпозиции. Причем нас будет интересовать не просто процесс декомпозиции, а процесс декомпозиции без потерь.

Процесс декомпозиции будем называть декомпозицией без потерь, если из полученных отношений можно полностью восстановить исходное отношение.

По своей сути декомпозиция представляет собой проекцию. А обратное преобразование - операция соединения.

3. Нормализация данных

Теорема Хита

Пусть дано отношение R(A, B, C) где A, B и C – непересекающиеся подмножества множества атрибутов R. Причем множество атрибутов R равно объединению подмножеств A, B, C. Если R удовлетворяет функциональной зависимости $A \rightarrow B$, то R равно соединению его проекций {A, B} и {A, C}.

3. Нормализация данных

Пример

Рассмотрим отношение R:

Группа	Кол-во	Специальность
4311	15	5201
4361	15	3515
2311	21	5201

3. Нормализация данных

Пример

Декомпозиция без потерь:

Группа	Кол-во
4311	15
4361	15
2311	21

Группа	Специальность
4311	5201
4361	3515
2311	5201

3. Нормализация данных

Пример

Декомпозиция с потерями

Группа	Кол-во
4311	15
4361	15
2311	21

Кол-во	Специальность
15	5201
15	3515
21	5201

3. Нормализация данных

У рассмотренной таблицы легко заметить следующие проблемы:

- Добавление - нельзя сделать запись о том, что студент учится на какой-то специальности, не добавив информацию о хотя бы одном уроке.
- Удаление - обратное, если мы удалим все записи о предметах для какого-либо студента, то потеряем информацию и о его группе и специальности.
- Изменение - при изменении группы у студента, надо исправить не одно значение, а значение всех записей, соответствующих его урокам.

3. Нормализация данных

2НФ

Отношение находится во второй нормальной форме тогда и только тогда, когда оно находится в 1НФ и каждый неключевой атрибут неприводимо зависит от первичного ключа. Атрибут называется неключевым, если он не входит в состав ни одного возможного ключа. Иначе он называется ключевым.

Замечание

Определение 2НФ было дано только для отношения с одним возможным ключом.

3. Нормализация данных

Если говорить простым языком, то оно может быть трактовано следующим образом.

Если для каждой функциональной зависимости вида:

ключ → неключ. Атрибуты

слева нельзя опустить ни один атрибут, без потери части смысла, то отношение находится во 2НФ.

3. Нормализация данных

Определение для отношений с несколькими ключами

Это определение легко расширяется и на отношение с несколькими ключами.

Отношение находится во второй нормальной форме тогда и только тогда, когда оно находится в 1НФ и каждый неключевой атрибут неприводимо зависит от любого возможного ключа.

3. Нормализация данных

Вернемся к примеру

Рассмотрим одно из двух получившихся отношений. У него все еще есть недостатки!

Персоны		
Студент	Специальность	Группа
Иванов	5102	1311
Петров	3515	1361

3. Нормализация данных

Проблемы

- Добавление - нельзя добавить информацию о том, что студенты группы такой-то учатся по специальности такой-то, если нет ни одной записи о студентах этой группы.
- Удаление - при удалении всех студентов группы, теряется информация о том, какую специальность изучала данная группа.
- Изменение - при изменении номера специальности надо проводить изменения ни одной записи, а всех, соответствующих студентам данной группы.

3. Нормализация данных

3НФ

Отношение находится в 3НФ тогда и только тогда, когда оно находится во 2НФ и отсутствуют транзитивные зависимости неключевых атрибутов от ключевых.

Пояснение: транзитивной зависимостью неключевых атрибутов от ключевых будем считать следующее:

$A \rightarrow B$ $B \rightarrow C$,

где A - набор ключевых атрибутов (ключ), B и C - различные множества неключевых атрибутов.

3. Нормализация данных

Поэтому мы можем сказать, что необходимо убедиться лишь в отсутствии взаимозависимостей между неключевыми атрибутами.

3. Нормализация данных

Модифицируем наш пример

Персона-группа	
Специальность	Группа
5102	1311
3515	1361

Группа-специальность	
Специальность	Группа
5102	1311
3515	1361

3. Нормализация данных

Нормальная форма Бойса-Кодда

Хотя данное определение для третьей нормальной формы применимо как для случая с одним потенциальным ключом, так и для случаев с большим количеством потенциальных ключей, тем не менее, во втором случае некоторые проблемы еще остаются.

Поэтому в случае, когда отношение содержит два или более потенциальных ключа обычно говорят не о 3НФ, а о нормальной форме Бойса-Кодда.

3. Нормализация данных

3 условия

- Отношение имеет два (или более) потенциальных ключа.
- Эти ключи – составные.
- Два или более потенциальных ключа перекрываются.

Если хотя бы одно из этих условий не выполняется, то НФБК совпадает с 3НФ.

3. Нормализация данных

НФБК

Определение: отношение находится в НФБК тогда и только тогда, когда детерминанты всех его ФЗ являются потенциальными ключами.

3. Нормализация данных

Пример

Студенты			
Ф.И.О.	N зачетки	Предмет	Оценка
Иванов И.И.	123456	Химия	2
Иванов И.И.	123456	Физика	2
Иванов И.И.	123456	Рус.-яз.	2
Петров П.П.	654321	Химия	2

3. Нормализация данных

Исследуем этот пример

Если мы предположим, что номера зачеток и Ф.И.О. студентов не повторяются, то у этого отношения можно выделить два потенциальных ключа:

- К1 - {Ф.И.О., Предмет} и
- К2 - {N зачетки, Предмет}.

Тогда множество ФЗ будет выглядеть следующим образом:

- К1 →К2 | Оценка
- К2 →К1 | Оценка
- Ф.И.О. →N зачетки

3. Нормализация данных

Последние две ФЗ и дают нам некоторую аномалию, которую не замечает 3НФ. Например, для изменения имени студента Иванова надо найти все записи с номером 123456.

В дальнейшем нормальная форма Бойса-Кодда и нормальные формы более высоких порядков не получили широкого распространения на практике.

Основным достоинством реляционной модели является ее простота. Именно благодаря ей она положена в основу подавляющего большинства реально работающих СУБД.

4. Язык SQL