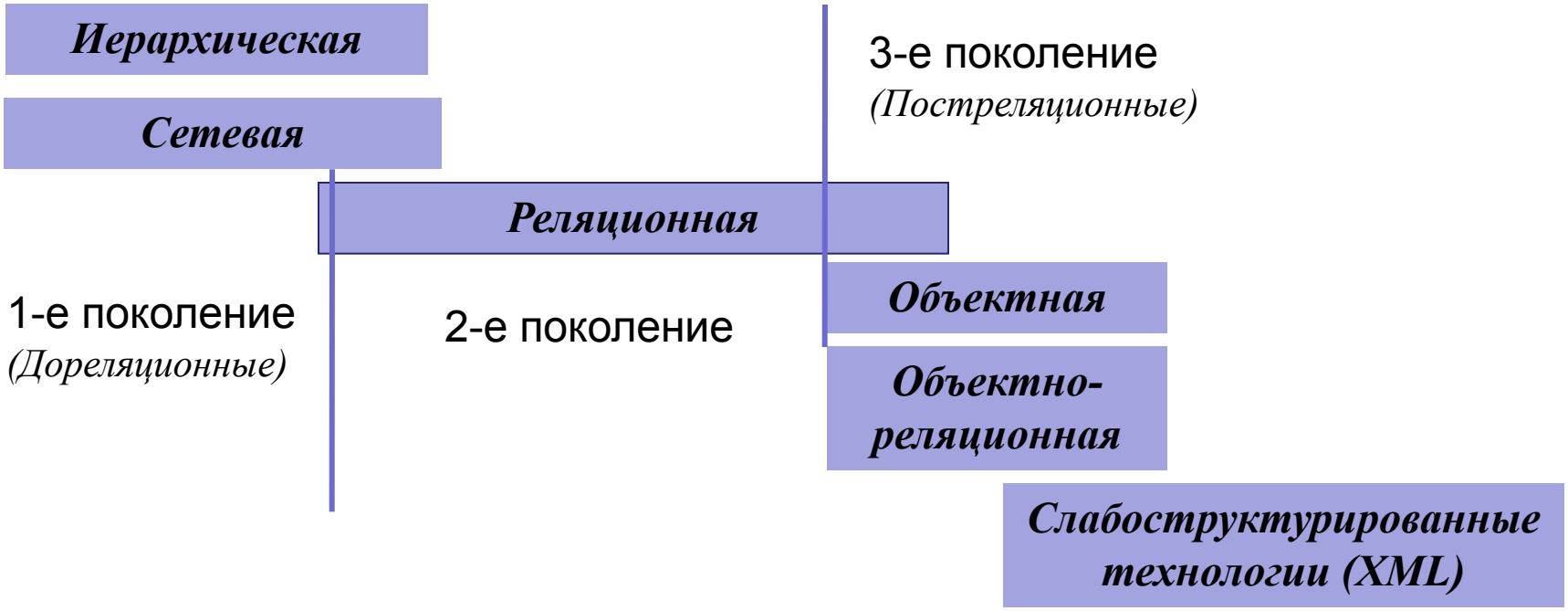
Модели данных

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь.
Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных.

Каждая БД и СУБД строится на основе некоторой явной или неявной модели данных. Все СУБД, построенные на одной и той же модели данных, относят к одному типу.



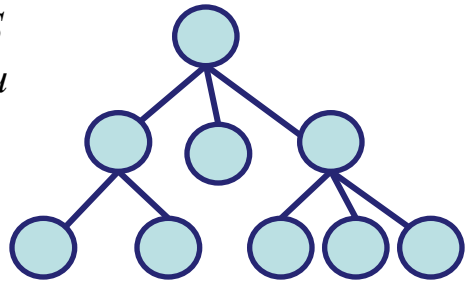
Иерархическая модель данных

Иерархическая модель данных — логическая модель данных в виде совокупности элементов, расположенных в порядке их подчинения от общего к частному и образующих перевернутое дерево (граф).

Объединяет записи, хранимые в общей древовидной структуре с одним корневым типом записи, который имеет нуль или больше подчиненных типов записей. Каждый подчиненный тип записи также может иметь нуль или больше подчиненных типов записей.

Иерархическая модель данных ориентирована на последовательный доступ к данным, хранимым на магнитных лентах.

Первая и «основная» коммерческая реализация – СУБД IMS (Information Management System) разработана при реализации космической программы Appolo



Соглашений об архитектуре иерархической модели и синтаксисе ее языка не существует.

Не смотря на это, системы на основе иерархической модели используются до настоящего времени.

Проблемы иерархической модели:

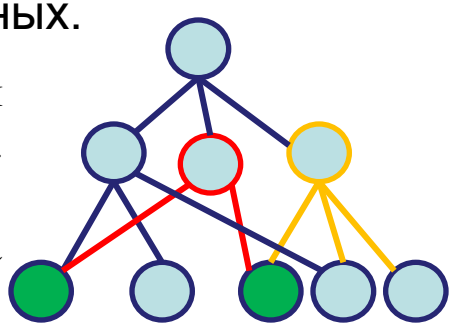
- необходимость дублирования (повторения) данных для обеспечения представления реальных объектов;
- ориентированность структуры на один конкретный вид обработки;
- навигационный механизм доступа.

Сетевая модель данных

Сетевая модель данных — логическая модель данных , состоящая из записей, элементов данных и связей типа «один ко многим», установленных между записями

Сетевая модель ориентирована на дисковое хранение данных.

Сетевая БД состоит из набора экземпляров определенного типа записи и набора экземпляров определенного типа связей между этими записями. Тип связи определяется для двух типов записи: предка и потомка. Экземпляр типа связи состоит из одного экземпляра типа записи предка и упорядоченного набора экземпляров типа записи потомка.



В 1971 году для сетевой модели был разработан «стандарт» (набор соглашений) организацией CODASYL (conference on data systems languages)

Введены понятия :

Сетевая схема – логическая организация всей БД.

Подсхема – часть БД, представляема пользователю или приложению.

Язык управления данными DML

Язык определения данных DDL (

Проблемы сетевой модели:

- сложность модификации структуры;
- навигационный характер доступа к элементам структуры

Модели первого поколения не обеспечивали полной независимости от данных и не имели проработанного математического базиса.

Реляционные базы данных

Расширение использования БАЗ ДАННЫХ и недостатки иерархических и сетевых моделей предопределили необходимость разработки способа описания данных, который:

- Понятен пользователю, не имеющему особых навыков в программировании
- Позволяет расширять БД без изменения имеющейся части
- Допускает максимальную гибкость при обработке непредсказуемых или случайных запросов

В качестве решения предложен реляционный подход

- **небольшой набор абстракций, позволяющий моделировать большую часть предметных областей, допускают точные формальные определения, оставаясь интуитивно понятными (*табличное представление*);**
- **простой и мощный *математический аппарат*, опирающегося главным образом на теорию множеств и математическую логику и обеспечивающего теоретический базис реляционного подхода к организации баз данных;**
- **возможность *ненавигационного манипулирования данными* без необходимости знания конкретной физической организации баз данных во внешней памяти.**

Реляционные модель данных

Формальной основой реляционной модели является аппарат математической теории множеств

Тип ***прямого произведения (декартова произведения)*** – запись:

D1 × D2 ×…× Dn = {<d1,d2,…,dn>|di∈Di, i=1,2,…,n} (1).

представляет собой множество, состоящее из кортежей, каждый из которых, в свою очередь, состоит из последовательно расположенных элементов **di**, взятых по одному из каждого множества **Di (di∈Di).**

Запись - удобная структура данных для представления набора значений атрибутов, описывающих некоторую сущность (объект) реального мира.

Совокупность всех возможных значений некоторого свойства сущности образуют область определения этого свойства – ***домен (именованное множество скалярных значений одного типа)***.

Декартово произведение (1) включает кортежи, полученные на основе всех возможных комбинаций элементов множеств D1, D2,…Dn.

Реляционные модель данных

При описании предметной области реально имеют смысл только те кортежи, которые представляют совокупность значений свойств ***реально существующих экземпляров сущностей предметной области****.*

То, что кортеж удовлетворяет этому условию, выражают предикатом ***P(d1,d2,…,dn).***

Отношением **называется подмножество кортежей прямого произведения, для которых высказывание P (d1,d2,…,dn) является истинным:**

$$R(D1, D2,\ldots Dn) = \{< d1,d2,\ldots,dn > \mid \mid di \in Di,\ i=1,2,\ldots,n \ \&\ P(d1,d2,\ldots,dn) = TRUE\}. \quad (2)$$

(Другие названия: ***реляционная структура***, или ***реляционный тип***.)

Имя отношения - имя, которое представляет *реляционный тип* данных.

Имя атрибута или просто ***атрибут*** - имя, представляющее один тип области определения. *Атрибут является оператором, с помощью которого из данных реляционного типа получают составляющие - данные простого типа*.

Для данных одного реляционного типа ***одноименные атрибуты не допускаются***. *Каждый атрибут должен быть определен только на одном домене. Но, различные атрибуты могут иметь одну область определения*.

Реляционная модель данных: Структура реляционного отношения

Заголовок– фиксированное **множество пар <имя-атрибута : имя домена >:**

$$\{<A_1:D_1>, <A_2:D_2>,\ldots,<A_n:D_n>\}, \quad (3)$$

(каждый атрибут A_j соответствует только одному домену D_j, и все имена атрибутов – разные).

При построении БД заголовок отношения называют ***схемой отношения***.

Если понятие домена не поддерживается СУБД, то вместо имени домена указывается имя типа данных, поддерживаемого СУБД, и, возможно, дополнительное ограничение на значения.

Тело отношения – множество из **m** строк данных (кортежей). Каждый i-й (i=1,…, m) кортеж содержит

множество пар <имя атрибута : значение атрибута >:

$$\{<A_1:V_{i1}>, <A_2:V_{i2}>,\ldots,<A_n:V_{in}>\}. \quad (4)$$

Значение атрибута является допустимым значением, взятым из домена данного атрибута (или типа данных, если понятие домена не поддерживается).

Кортеж - это набор именованных значений заданного типа.

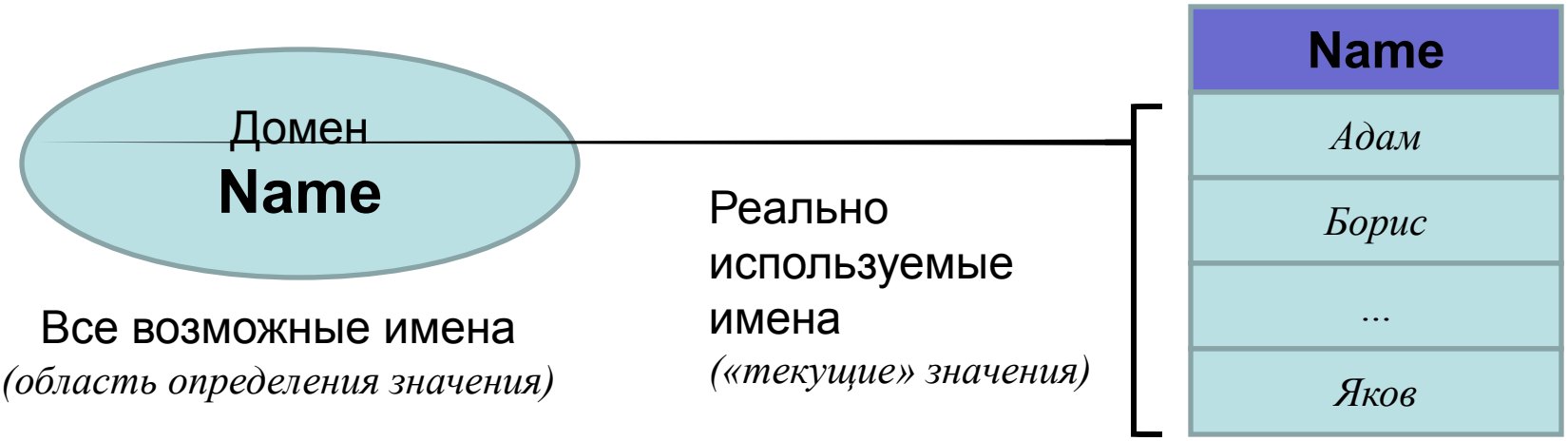
Значение **m** (число кортежей в теле отношения) называют ***кардинальным числом отношения***.

Значение **n** (число атрибутов, входящих в отношение) называются ***степенью отношения.***

Реляционные модель данных: Домены и отношения

В классических реляционных базах данных после определения схемы базы данных изменяются, как правило, только отношения-экземпляры. В них могут появляться новые и удаляться или модифицироваться существующие кортежи.
Однако во многих реализациях допускается и изменение схемы базы данных: определение новых и изменение существующих схем отношения. Это принято называть *эволюцией схемы базы данных*.

Различия между доменом и унарным отношением заключается в том, что домены статичны (т.е. значения области определения не меняются), а отношение динамично (т.е. может меняться как содержание кортежей, так и их количество (кардинальное число отношения)).



Реляционные модель данных: Таблицы и отношения

Для наглядного представления всех элементов множества, являющегося отношением удобно использовать *двумерные таблицы*. Если отношению сопоставить таблицу, то:

- ***название таблицы*** соответствует ***имени отношения***;
- ***кортеж*** отношения представляется ***строкой таблицы*** (или ***записью*** в терминологии СУБД);
- ***атрибут*** – ***имя столбца таблицы***;
- ***значения атрибута*** – ***множество значений в*** конкретном ***столбце*** таблицы (в СУБД – ***поле***);
- ***домен*** – область определения ***одного или нескольких атрибутов***, т.е. множество всех допустимых значений для столбца (нескольких столбцов) таблицы.

Атрибут 1	Атрибут 2	….	Атрибут n
знач.атрибута 1	знач.атрибута 2	…	знач.атрибута n
знач.атрибута 1	знач.атрибута 2	…	знач.атрибута n
…	…	…	…

*Таблица – наглядная **модель** отношения*

Реляционные модель данных:
Фундаментальные свойства отношений

Фундаментальные свойства отношений следуют из того, что отношения являются множествами.
Поэтому в любом отношении:

1. ***Нет одинаковых кортежей*** – т.к. тело отношения – это множество кортежей, а множества по определению не содержат одинаковых элементов. Из этого следует, что всегда можно определить уникальный идентификатор кортежа (в крайнем случае - как полный набор атрибутов кортежа).
2. ***Кортежи не упорядочены сверху вниз***. Это так же следует из свойств простого множества.
3. ***Атрибуты не упорядочены слева направо***, т.к. заголовок – множество атрибутов.
4. ***Все значения атрибутов атомарные.*** Это следует из атомарности значений доменов (отношения не содержат групп повторения). Отношения, удовлетворяющие этому условию, называются нормализованными, или представленными в *первой нормальной форме*.

Реляционные модель данных:

Фундаментальные свойства отношений

В реляционной модели *<u>ВСЕ</u>* отношения должны быть нормализованными, а отношение в математическом смысле может быть и не нормализованным

Поставщик	Прайс-лис	
П1	**товар**	**Цена**
	Т1	Ц1
	Т2	Ц2
П2	**товар**	**Цена**
	Т2	Ц3
	Т3	Ц4

Поставщик	товар	Цена
П1	Т1	Ц1
П1	Т2	Ц2
П2	Т2	Ц3
П2	Т3	Ц4

Нормализованные отношения составляют основу классического реляционного подхода к организации баз данных.

Они обладают некоторыми ограничениями (не любую информацию удобно представлять в виде плоских таблиц), но существенно упрощают манипулирование данными.

Структура реляционной модели данных

Реляционная модель состоит из трех четко выраженных частей, описывающих разные аспекты реляционного подхода: ***структурной, манипуляционной и целостной*.**

В ***структурной части*** модели фиксируется, что единственной структурой данных, используемой в реляционных БД, является ***нормализованное n-арное отношение***.

Манипуляционная часть модели включает два фундаментальных механизма манипулирования реляционными БД - ***реляционную алгебру*** и ***реляционное исчисление***.

Основной функцией манипуляционной части реляционной модели является обеспечение меры реляционности любого конкретного языка реляционных БД: язык называется реляционным, если он обладает не меньшей выразительностью и мощностью, чем реляционная алгебра или реляционное исчисление.

В ***целостной части*** реляционной модели данных фиксируются два базовых требования целостности, которые должны поддерживаться в любой реляционной СУБД: ***требование целостности сущностей и требованием целостности по ссылкам*** .

Требование целостности сущностей **состоит в том, что** ***любой кортеж любого отношения отличим от любого другого кортежа этого отношения*****,**

Потенциальный ключ - атрибут или группу атрибутов, на основе значений которых можно уникальным образом идентифицировать экземпляр сущности.

Ключ, состоящий из одного атрибута, называют ***простым***, а из группы атрибутов – ***составным***. *В общем случае сущность может иметь несколько потенциальных ключей.*

Если **R** – некоторое отношение, то ключ **K** – это подмножество множества атрибутов отношения **R (K ⊂ {Ai})**, обладающее следующими свойствами:

Уникальность – в **R** нет двух различных кортежей с одинаковым значением **K**

Неизбыточность или ***неприводимость*** (для составных ключей) - никакое из подмножеств K не обладает свойством уникальности.

(*в ключе не должно быть «лишних» атрибутов, т.е. таких атрибутов, которые можно исключить из состава ключа, сохранив свойство уникальности*).

Потенциальные ключи обеспечивают основной механизм адресации на уровне кортежей в реляционной схеме.

Первичный ключ - используется в качестве идентификатора экземпляров сущности (один из потенциальных)

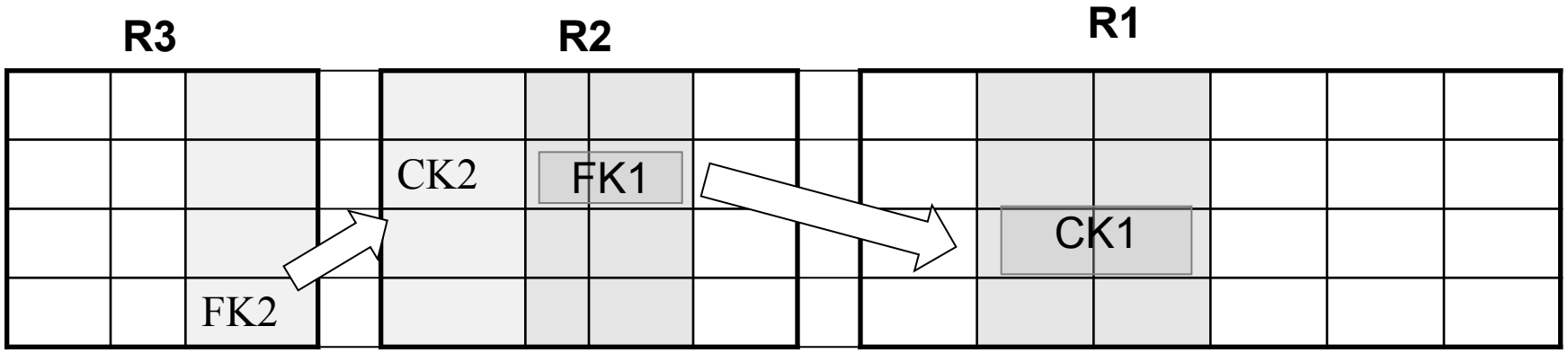
Требование целостности по ссылкам

Описание сложных сущностей реального мира может быть представлено в реляционной БД в виде нескольких кортежей нескольких отношений. Для того чтобы определить, что эти кортежи относятся к одной сущности (объекту) реального мира необходимо иметь специальный аппарат описания связей между этими кортежами. В качестве такого аппарата используются ***внешние (или ссылочные) ключи.***

Пусть R2 – базовое отношение. Тогда *внешний ключ* FK в отношении R2 - это подмножество атрибутов R2, такое что:

Существует другое базовое отношение R1 с потенциальным ключем CK;

Каждое значение FK в текущем значении R2 всегда совпадает со значением CK некоторого кортежа в текущем значении R1.



между отношениями R1 и R3 существует ссылочный путь (R3 → R2→ R1), позволяющий идентифицировать соответствующие друг другу кортежи в отношениях R1 и R3.

Ссылочная целостность базы данных требует, чтобы БД не содержала несогласованных значений внешних ключей *(для каждого значения внешнего ключа FK, появляющегося в ссылающемся отношении R2, в отношении R1, на которое ведет ссылка, должен найтись кортеж с таким же значением потенциального ключа CK)*

Для обеспечения ссылочной целостности при выполнении операций удаления и модификации записей в базе данных существует несколько механизмов:

Ограничения – запрещается производить удаление кортежа, на который существуют ссылки

Каскадирование - при удалении кортежа из отношения, на которое ведет ссылка, из ссылающегося отношения автоматически удаляются все ссылающиеся кортежи.

Использование неопределенного значения - при удалении кортежа, на который имеются ссылки, во всех ссылающихся кортежах значение внешнего ключа автоматически становится неопределенным (**null-значения**).

По мнению ряда авторов **null-значения** нарушают формальную модель РБД. В качестве альтернативного решения предлагается использовать некоторое **значение по умолчанию** (возможно – фиктивное) вместо неизвестных значений.

Отношения, используемые в БД можно подразделить на следующие виды:

1. ***Именованное отношение*** – переменная отношения определенная в СУБД с помощью соответствующего оператора CREAT
2. ***Базовое отношение*** – именованное отношение, являющееся важной автономной частью БД (существует как самостоятельная единица структуры БД).
3. ***Производное отношение*** – определяется через другие именованные (базовые) отношения посредством реляционных выражений
4. ***Представление*** – именованное производное отношение. Представления виртуальны – представлены в системе через определение в терминах других именованных отношений.

Отношения, используемые в БД можно подразделить на следующие виды:

5. ***Снимки (snapshot)*** – именованные производные отношения, но в отличии от представлений они реальны, а не виртуальны, т.е. представлены в структуре своими данными. (Обычно доступны только для чтения).
6. ***Результат запроса*** – неименованное производное отношение. Результаты запроса можно сохранить присвоив его некоторому именованному отношению.
7. ***Промежуточный результат*** – неименованное производное отношение, являющееся результатом некоторого реляционного выражения и вложенное в более сложное реляционное выражение.
8. ***Выражаемое отношение*** – отношение, которое можно получить из набора именованных отношений посредством некоторого реляционного выражения..
9. ***Хранимое отношение*** – поддерживается во внешней физической памяти.