Основы баз данных и знаний

Основные понятия реляционной модели БД

Лекция 2

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Реляционный подход

Реляционный подход к организации баз данных был заложен в конце 1960-х гг. Эдгаром Коддом.

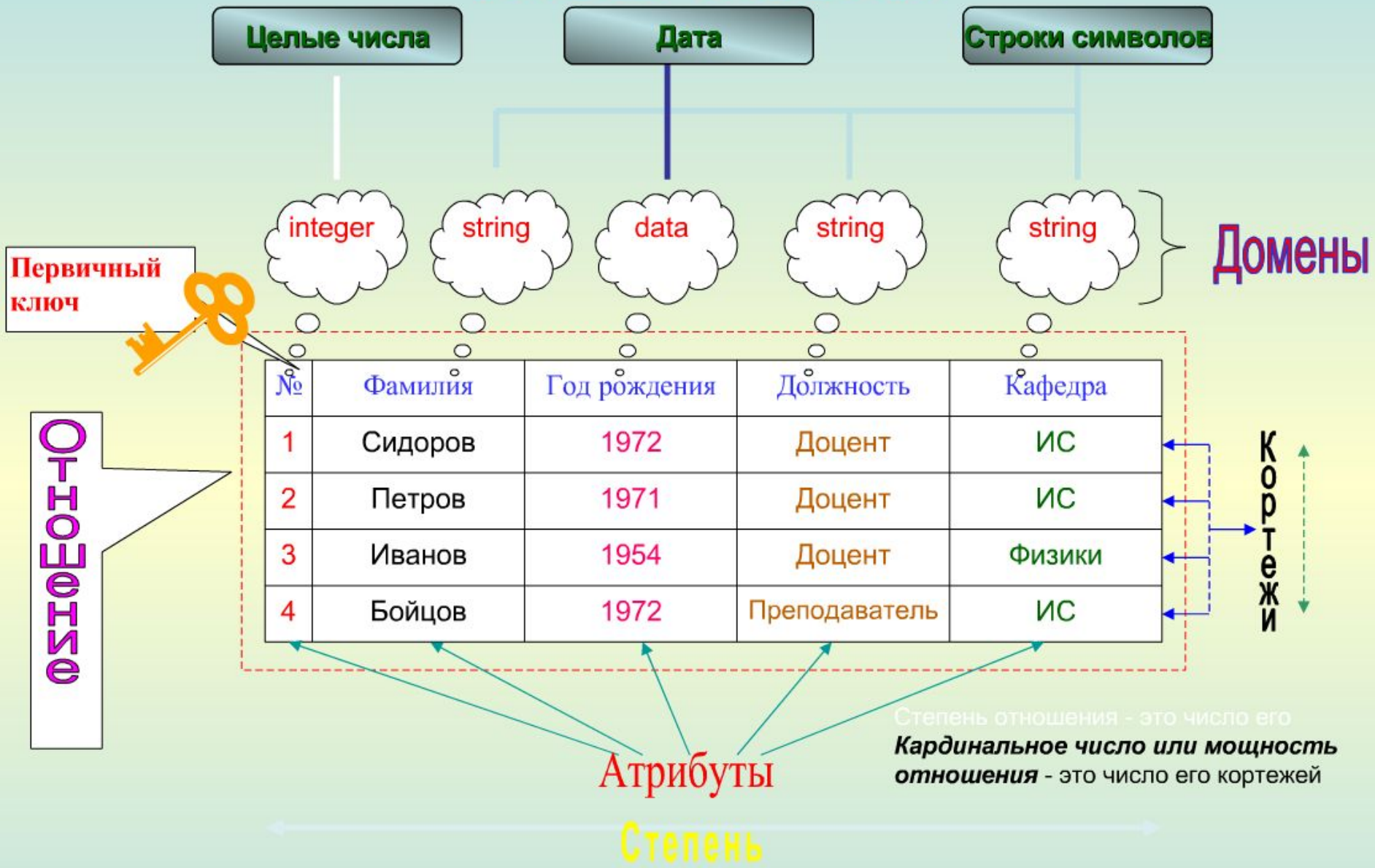
Будучи математиком по образованию, Э.Кодд предложил использовать для обработки данных аппарат теории множеств (объединение, пересечение, разность, декартово произведение). Он показал, что любое представление данных сводится к совокупности двумерных таблиц особого вида, известного в математике как **отношение – relation**.

Достоинствами реляционного подхода принято считать следующие свойства:

- реляционный подход основывается на небольшом числе интуитивно понятных абстракций, на основе которых возможно простое моделирование наиболее распространенных предметных областей;
- эти абстракции могут быть точно и формально определены;
- теоретическим базисом реляционного подхода к организации баз данных служит простой и мощный математический аппарат теории множеств и математической логики;
- реляционный подход обеспечивает возможность ненавигационного манипулирования данными без необходимости знания конкретной физической организации баз данных во внешней памяти.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Основные обозначения



Степень отношения - это число его

Кардинальное число или мощность отношения - это число его кортежей

Каждая строка, содержащая данные, называется **кортежем**, каждый столбец отношения называется **атрибутом** (на уровне практической работы с современными реляционными БД используются термины "запись" и "поле").

Тип данных

Значения данных, хранимые в реляционной базе данных, являются **типизированными**, т. е. известен тип каждого хранимого значения. Понятие типа данных в реляционной модели данных полностью соответствует понятию типа данных в языках программирования.

Традиционное (нестрогое) определение **типа данных** состоит из трех основных компонентов:

определение множества значений данного типа;

определение набора операций, применимых к значениям типа;

определение способа внешнего представления значений типа (литералов).

Обычно в современных реляционных базах данных допускается хранение символьных, числовых данных (точных и приблизительных), специализированных числовых данных (таких, как «деньги»), а также специальных «темпоральных» данных (дата, время, временной интервал). Активно развивается подход к внедрению в реляционные системы возможностей определения пользователями собственных типов данных.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Основными понятиями реляционных баз данных являются

1. ***тип данных***
2. ***домен***
3. ***атрибут***
4. ***кортеж***
5. ***первичный ключ***
6. ***отношение***

Домен - это набор всех допустимых значений, которые может содержать *атрибут*.

Атрибут - поименованная характеристика сущности

Сущность - объект и (или) факт предметной области, информацию о котором необходимо хранить в БД

Каждая строка, содержащая данные, называется ***кортежем***

Первичный (идентифицирующий, уникальный) ключ - атрибут, однозначно определяющий экземпляр сущности.

Отношение R можно определить как множество **кортежей**, каждый из которых представляет собой вектор-строку **v** = $(v_1, v_2, \ldots, v_k)$ отношения R. Элементы vl кортежа **v** выбираются из определенных доменов, $v_l \in D_l$.

Отношение

Понятие **отношения** является наиболее фундаментальным в реляционном подходе к организации баз данных, поскольку **n-арное отношение** является единственной родовой структурой данных, хранящихся в реляционной базе данных.

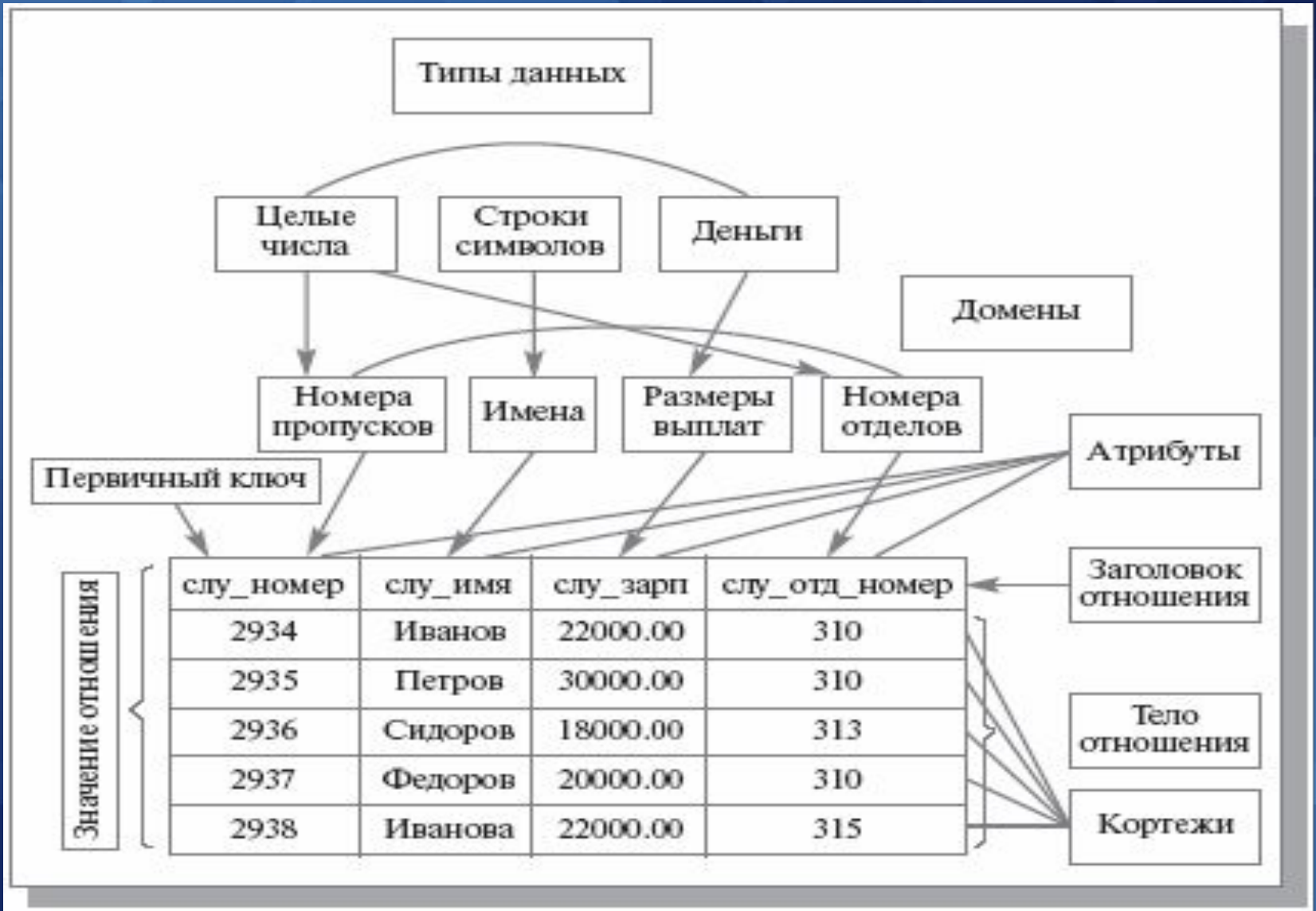
Для уточнения термина отношение выделяются понятия заголовка и тела отношения, значения отношения и переменной отношения.

Заголовком (или схемой) отношения r (Hr) называется конечное множество пар вида <A, T>, где A называется именем атрибута, а T обозначает имя некоторого базового типа или домена. По определению требуется, чтобы все имена атрибутов в заголовке отношения были различны.

В примере заголовком отношения СЛУЖАЩИЕ является множество пар {<слу_номер, номера_пропусков>, <слу_имя, имена>, <слу_зарп, размеры_выплат>, <слу_отд_номер, номера_отделов>}.
Если все атрибуты заголовка отношения определены на разных доменах, то, чтобы не плодить лишних имен, разумно использовать для именования атрибутов имена соответствующих доменов (не забывая, конечно, о том, что это всего лишь удобный способ именования, который не устраняет различия между понятиями домена и атрибута).

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Отношение



Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Отношение

Кортежем tr, соответствующим заголовку Hr, называется множество упорядоченных триплетов вида **<A, T, v>**, по одному такому триплету для каждого атрибута в Hr. Третий элемент – v – триплета <A, T, v> должен являться допустимым значением типа данных или домена T.

Заголовку отношения СЛУЖАЩИЙ соответствуют, например, следующие кортежи: {<слу_номер, номера_пропусков, 2934>, <слу_имя, имена, Иванов>, <слу_зарп, размеры_выплат, 22.000>, <слу_отд_номер, номера_отделов, 310>}, {<слу_номер, номера_пропусков, 2940>, <слу_имя, имена, Кузнецов>, <слу_зарп, размеры_выплат, 35.000>, <слу_отд_номер, номера_отделов, 320>}.

Телом Br отношения r называется произвольное множество кортежей tr. Одно из возможных тел отношения СЛУЖАЩИЙ показано на рис.

В общем случае могут существовать такие кортежи tr, которые соответствуют Hr, но не входят в Br.

Значением Vr отношения r (или просто отношением r, или отношением-экземпляром) называется пара множеств Hr и Br. Одно из допустимых значений отношения СЛУЖАЩИЙ показано на рис. 2.1.

(Отношение – это множество кортежей tr, соответствующих одной схеме отношения Hr).

В изменчивой реляционной базе данных хранятся отношения, значения которых изменяются во времени.

Переменной VARr называется именованный контейнер, который может содержать любое допустимое значение Vr. Естественно, что при определении любой VARr требуется указывать соответствующий заголовок отношения Hr.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Отношение

Степенью, или «арностью», заголовка отношения, кортежа, соответствующего этому заголовку, тела отношения, значения отношения и переменной отношения является мощность заголовка отношения (количество атрибутов).

1 атрибут – отношение унарное, 2 атрибута – отношение бинарное, 3 – тернарное, 4 – кватернарным, с большим количеством атрибутов – n-арное.

Например, степень отношения СЛУЖАЩИЕ равна четырем, т. е. оно является 4-арным (кватернарным).

Кардинальность (или кардинальное число) - мощность отношения – количество кортежей.

Реляционная база данных – набор нормализованных отношений r, которые различаются по именам.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

1. Правила Кодда для реляционных баз данных

В статье, опубликованной в журнале "Computer World", **Тэд Кодд** сформулировал двенадцать правил, которым должна соответствовать настоящая реляционная база данных. Двенадцать правил Кодда являются полуофициальным определением понятия *реляционная база данных.* Перечисленные правила основаны на теоретической работе Кодда, посвященной реляционной модели данных.

1. Правило информации. Вся информация в базе данных должна быть предоставлена исключительно на логическом уровне и только одним способом - в виде значений, содержащихся в таблицах

2. Правило гарантированного доступа. Логический доступ ко всем и каждому элементу данных (атомарному значению) в реляционной базе данных должен обеспечиваться путём использования комбинации имени таблицы, первичного ключа и имени столбца.

3. Правило поддержки недействительных значений. В настоящей реляционной базе данных должна быть реализована поддержка недействительных значений, которые отличаются от строки символов нулевой длины, строки пробельных символов и от нуля или любого другого числа, и используются для представления отсутствующих данных независимо от типа этих данных.

4. Правило динамического каталога, основанного на реляционной модели. Описание базы данных на логическом уровне должно быть представлено в том же виде, что и основные данные, чтобы пользователи, обладающие соответствующими правами, могли работать с ним с помощью того же реляционного языка, который они применяют для работы с основными данными.

5. Правило исчерпывающего подъязыка данных. Реляционная система может поддерживать различные языки и режимы взаимодействия с пользователем (например, режим вопросов и ответов). Однако должен существовать по крайней мере один язык, операторы которого можно представить в виде строк символов в соответствии с некоторым четко определенным синтаксисом и который в полной мере поддерживает следующие элементы:

- определение данных;
- определение представлений;
- обработку данных (интерактивную и программную);
- условия целостности;
- идентификация прав доступа;
- границы транзакций (начало, завершение и отмена).

6.Правило обновления представлений.

Все представления, которые теоретически можно обновить, должны быть доступны для обновления.

7. Правило добавления, обновления и удаления.

Возможность работать с отношением как с одним операндом должна существовать не только при чтении данных, но и при добавлении, обновлении и удалении данных.

8. Правило независимости физических данных. Прикладные программы и утилиты для работы с данными должны на логическом уровне оставаться нетронутыми при любых изменениях способов хранения данных или методов доступа к ним.

9. Правило независимости логических данных. Прикладные программы и утилиты для работы с данными должны на логическом уровне оставаться нетронутыми при внесении в базовые таблицы любых изменений, которые теоретически позволяют сохранить нетронутыми содержащиеся в этих таблицах данные.

10. Правило независимости условий целостности. Должна существовать возможность определять условия целостности, специфические для конкретной реляционной базы данных, на подъязыке реляционной базы данных и хранить их в каталоге, а не в прикладной программе.

11. Правило независимости распространения. Реляционная СУБД не должна зависеть от потребностей конкретного клиента.

12. Правило единственности. Если в реляционной системе есть низкоуровневой язык (обрабатывающий одну запись за один раз), то должна отсутствовать возможность использования его для того, чтобы обойти правила и условия целостности, выраженные на реляционном языке высокого уровня (обрабатывающем несколько записей за один раз).

Пояснения к 12 правилам Кодда

Правило 1 (информации) напоминает неформальное определение реляционной базы данных.

Правило 2 (*гарантированного доступа*) указывает на роль первичных ключей при поиске информации в базе данных. Имя таблицы позволяет найти требуемую таблицу, имя столбца позволяет найти требуемый столбец, а первичный ключ позволяет найти строку, содержащую искомый элемент данных.

Правило 3 (*недействительных значений*) требует, чтобы отсутствующие данные можно было представить с помощью недействительных значений (NULL)

Правило 4 (*динамического каталога*) гласит, что реляционная база данных должна сама себя описывать. Другими словами, база данных должна содержать набор системных таблиц, описывающих структуру самой базы данных.

Пояснения к 12 правилам Кодда

Правило 5 (*исчерпывающего подъязыка данных*) требует, чтобы СУБД использовала язык реляционной базы данных, например SQL, хотя явно SQL в правиле не упомянут. Такой язык должен поддерживать все основные функции СУБД — создание базы данных, чтение и ввод данных, реализацию защиты базы данных и т.д.

Правило 6 (*обновления представлений*) касается представлений, которые являются виртуальными таблицами, позволяющими показывать различным пользователям различные фрагменты структуры базы данных. Это одно из правил, которые сложнее всего реализовать на практике.

Правило 7 (*добавления, обновления и удаления*) акцентирует внимание на том, что базы данных по своей природе ориентированы на множества. Оно требует, чтобы операции добавления, удаления и обновления можно было выполнять над множествами строк. Это правило предназначено для того, чтобы запретить реализации, в которых поддерживаются только операции над одной строкой.

Пояснения к 12 правилам Кодда

Правила 8 и 9 (*независимости*) означают отделение пользователя и прикладной программы от низкоуровневой реализации базы данных. Они утверждают, что конкретные способы реализации хранения или доступа, используемые в СУБД, и даже изменения структуры таблиц базы данных не должны влиять на возможность пользователя работать с данными.

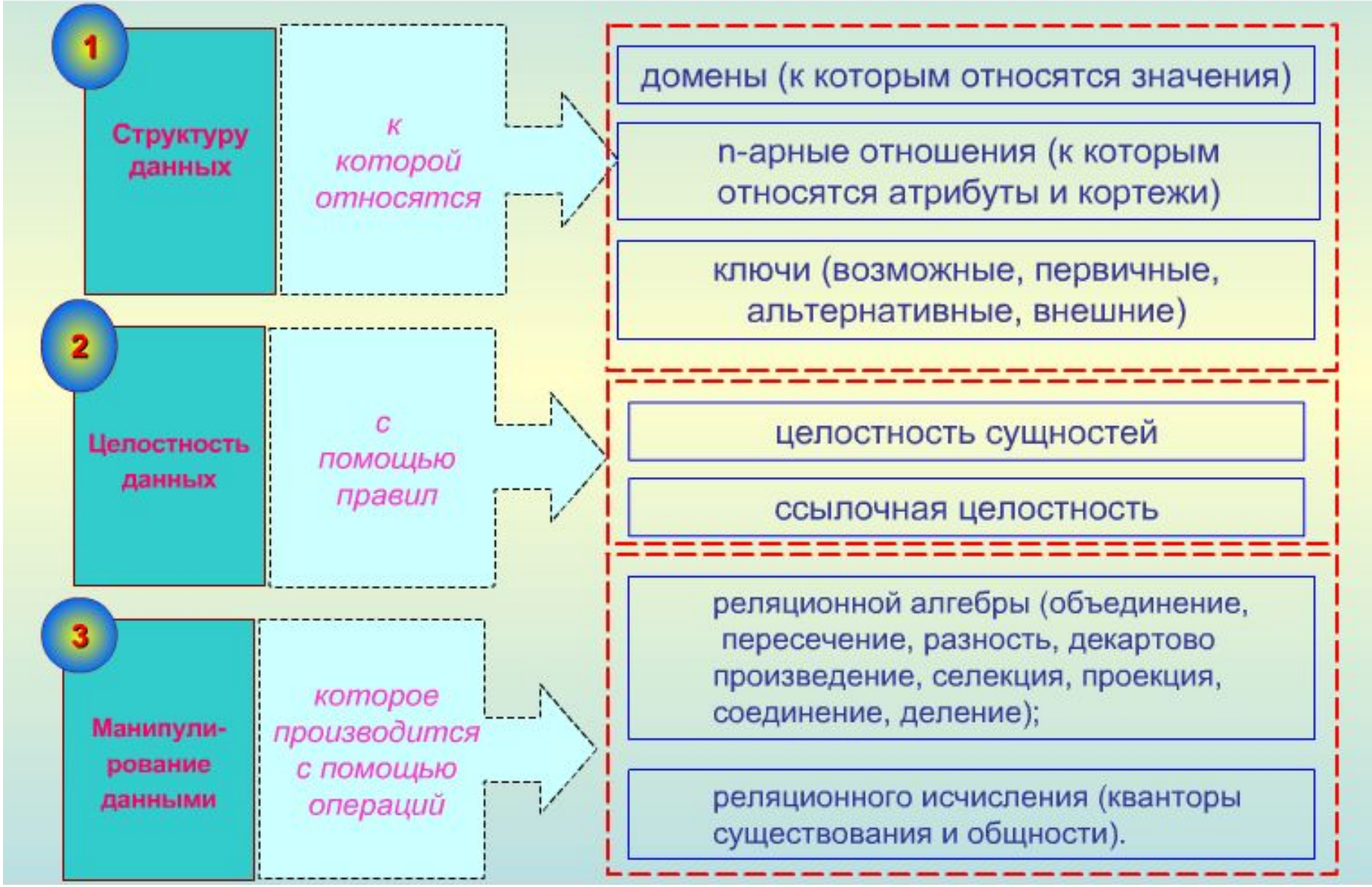
Правило 10 (*независимости условий целостности*) гласит, что язык базы данных должен поддерживать ограничительные условия, налагаемые на вводимые данные и действия, которые могут быть выполнены над данными.

Правило 11 (*независимости распространения*) гласит, что язык базы данных должен обеспечивать возможность работы с распределенными данными, расположенными на других компьютерных системах.

Правило 12 (*единственности*) предотвращает использование других возможностей для работы с базой данных, помимо языка базы данных, поскольку это может нарушить ее целостность.

2. Реляционная структура данных

Реляционная **модель определяет**



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

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

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

Реляционная база данных и ее схема

Пользовательский уровень понятия реляционной БД

Реляционная БД

Это конечная совокупность реляционных таблиц, объединенных смысловым содержанием, а так же процедурами контроля целостности и обработки данных в интересах решения некоторых прикладных задач.

Понятие реляционной БД на языке математики

Пусть R - конечное множество имен атрибутов, значения которых требуется хранить в БД.

Реляционной БД над R называется конечная совокупность отношений

$d(\rho) = r_1(R_1), r_2(R_2), \ldots, r_m(R_m),$

где $\rho = \{ R_1, R_2, \ldots, R_m \mid U R_i = R \}$ - ***схема БД.***

Считается, что среди R_i могут быть равные и такие, что

$R_i \cap R_j \neq \varnothing \ (1 \leq i < j \leq m)$.

U- объединение множеств

∩- пересечение множеств

Ø- пустое множество

Используется терминология:

R_i - схема отношения r_i,

R_i - подсхема схемы БД.

Пример

Рассмотрим следующую БД.

r_1 - СПИСОК ДОМОВ

Улица	Дом	Площадь, $м^2$	Сложность ремонта
U	D	P	C
Ленина	105A	500	1
Маркса	24	200	0.8

r_2 - СПИСОК МАСТЕРОВ

Табельный номер	Ф.И.О.	Разряд
Tn	F	R
100001	Иванов И.И.	5
100002	Петушко К.А.	4
100050	Иванов С.С.	2

r_3 - ТАРИФНАЯ СЕТКА

Разряд	Тариф, руб
R	Tr
1	100
2	150
3	200
4	230
5	280

r_4 - СВЕДЕНИЯ О ВЫПОЛНЕНИИ РЕМОНТА

Табельный номер	Дата	Улица	Дом	Объем работы, $м^2$
Tn	Data	U	D	V
100001	01.10.01	Ленина	105A	10.5
100001	02.10.01	Ленина	105A	10.5
100001	10.10.01	Маркса	24	7.0
100002	03.10.01	Ленина	105A	8.5
100002	10.10.01	Маркса	24	7.0

Здесь реляционная БД состоит из четырех таблиц (отношений) $d(\rho) = \{ r_1(R_1), r_2(R_2), r_3(R_3), r_4(R_4) \}$ и имеет схему $\rho = \{ R_1, R_2, R_3, R_4 \}$, где

$R_1 = \{ \underline{U, D}, P, C \} = \underline{UD}PC,$

$R_2 = \{ \underline{Tn}, F, R \} = \underline{Tn}FR,$

$R_3 = \{ \underline{R}, Tr \} = \underline{R}Tr,$

$R_4 = \{ \underline{Tn, Data}, U, D, V \} = \underline{TnData}\ U\ D\ V.$

Схему БД можно записать по другому:

$\rho = \{R_1(\underline{UD}PC), R_2(\underline{Tn}FR), R_3(\underline{R}Tr), R_4(\underline{TnData}UDV)\}.$

Сравнение терминологий

Пользовательский уровень	Логический уровень	Физический уровень
Реляционная таблица с именем r	Отношение с именем r	Файл записей последовательного доступа с именем r
Столбец таблицы с именем A и множеством допустимых значений D	(A, D) - атрибут отношения	Поле записи с именем A и множеством допустимых значений D
Заголовок таблицы - строка имен столбцов $A_1, A_2, \ldots, A_n$	Схема отношения $R = A_1 A_2 \ldots A_n$, где A_i $(i = 1, 2, \ldots, n)$ - имена атрибутов отношения	Структура записи файла, где $A_1, A_2, \ldots, A_n$, - имена полей записи
Ключ таблицы	Ключ отношения	Ключ файла
Строка таблицы	Кортеж отношения	Запись файла
Поле таблицы	Значение атрибута	Значение поля записи
Реляционная БД, как конечная совокупность реляционных таблиц	Реляционная БД, как конечная совокупность отношений	Реляционная БД, как конечная совокупность файлов последовательного доступа

Реляционная алгебра и исчисление

Основные операции над отношениями

Объединение

Объединение (UNION) отношений R и S дает множество кортежей, которые принадлежат R или S , или им обоим.
Математическая запись: REZ = R ∪ S . Операция объединения применяется только к отношениям одной арности, причем

$$k_{REZ} = k_R = k_S .$$

Отметим, что имена атрибутов отношений-операндов могут быть различными. В этом случае операция объединения отношений выполнима, если соответствующие значения компонент кортежей принадлежат одному домену. Например, для

R = A B C	и	S = D E F	REZ = R ∪ S =
a b c		b g a	a b c
d a f		d a f	d a f
c c d			c c d
			b g a ,

где значения атрибутов (A,D) выбраны из одного домена, например, D1 , (B,E) - из D2 и (C,F) - из D3 . В этом случае атрибуты результирующего отношения REZ не именуются.

ПРИМЕР:

Пусть отношение R - множество кортежей поставщиков, поставляющих деталь Д1, а S - множество кортежей поставщиков из Киева:

R =

PN	PIM	ST	GOROD
П1	Иванов	20	Киев
П2	Петров	40	Харьков

S =

PN	PIM	ST	GOROD
П1	Иванов	20	Киев
П3	Сидоров	10	Киев

Тогда REZ = R ∪ S есть множество кортежей поставщиков, которые находятся в Киеве, либо поставляют деталь Д1, либо и то и другое:

REZ =

PN	**PIM**	**ST**	**GOROD**
П1	**Иванов**	**20**	**Киев**
П2	**Петров**	**40**	**Харьков**
П3	**Сидоров**	**10**	**Киев**

Можно определить язык запросов реляционной алгебры, близкий к естественному, например английскому, языку.
Тогда оператор объединения отношений можно записать в виде

R UNION S GIVING REZ

Используя этот оператор, можно к отношению REZ добавить новый кортеж, например для поставщика (П4, Морозов, 20, Полтава):
REZ UNION (“П4”, “Морозов”, 20, “Полтава”) GIVING REZ

Вычитание

Вычитание (MINUS). Результат включает только те кортежи первого отношения, которых нет во втором.

Математическая запись: REZ = R \ S. В результирующее отношение включаются кортежи из R , не принадлежащие S , и $k_{REZ} = k_R = k_S$. Например,для

R = A	B	C
a	b	c
d	a	f
c	c	d

и

S = D	E	F
b	g	a
d	a	f

REZ = R \ S =

a	b	c
c	c	d

Пусть отношение R - множество кортежей поставщиков, поставляющих деталь Д1, а S - множество кортежей поставщиков из Киева:

R = PN	PIM	ST	GOROD
П1	Иванов	20	Киев
П2	Петров	40	Харьков

S = PN	PIM	ST	GOROD
П1	Иванов	20	Киев
П3	Сидоров	10	Киев

Результирующее отношение REZ = R \ S состоит из кортежей поставщиков, поставляющих деталь Д1, но не проживающих в Киеве:

REZ = PN	PIM	ST	GOROD
П2	Петров	40	Харьков

Соответствующий оператор языка запросов реляционной алгебры может быть записан в виде R MINUS S GIVING REZ

Операция вычитания используется для удаления ненужных кортежей отношения.

Пересечение

Пересечение (INTERSECTION) отношений R и S дает множество кортежей, которые принадлежат как R, так и S.

Математическая запись: $REZ = R \cap S$, и $k_{REZ} = k_R = k_S$.

Операция пересечения отношений выполняется при тех же ограничениях, что и операция объединения. Например, для

R = A B C		S = D E F	REZ = R ∩ S =
a b c		b c c	b c c
b c c		a a a	a a b
a a b		a a b	

Соответствующий оператор языка запросов реляционной алгебры может быть записан в виде

R INTERSECTION S GIVING REZ

Операция пересечения может быть выражена через основные операции реляционной алгебры следующим образом

$$REZ = R \cap S = R \setminus (R \setminus S)$$

Это означает, что в результирующее отношение включаются кортежи из R не (не принадлежащие S), т.е. принадлежащие S.

Декартово произведение

Декартово произведение (TIMES).

Результирующее отношение REZ = R x S состоит из кортежей, первые kR компонент которых - кортежи из R , а последние k_S компонент - кортежи из S и $k_{REZ} = k_R + k_S$ ("склеивание" кортежей операндов). Например, для

R = A B C и S = D E

A	B	C
a	b	c
b	b	a

D	E
a	b
b	b

REZ = R x S =

A	B	C	D	E
a	b	c	a	b
a	b	c	b	b
b	b	a	a	b
b	b	a	b	b

Соответствующий оператор языка запросов может быть записан в виде

R TIMES S GIVING REZ

Используя операцию декартова произведения, можно определить отношение так:

Отношение R есть подмножество декартова произведения доменов D1, ... , Dk , не обязательно различных, т.е. $R \subseteq D1 \times D2 \times \ldots \times Dk$.

Проекция

Проекция (PROJECT) дает возможность построить вертикальное подмножество отношения.

Математическая запись: $REZ = \pi_{I1, I2, \ldots, Im}(R)$,

где I_J - атрибуты отношения R , заданные именами или номерами (целыми числами из диапазона $[1, k_R]$). Повторяющиеся кортежи из результирующего отношения исключаются. Арности отношений REZ и R связаны условием $k_{REZ} \le k_R$. Например, для

R = A	B	C	D
a	b	c	a
b	b	c	c
a	a	b	b

$REZ = \pi_{BC}(R) = \pi_{2,3}(R) =$

B	C
b	c
a	b

Соответствующий оператор языка запросов может быть записан в виде
PROJECT R OVER B, C GIVING REZ

Пример . Пусть

R = PN	PIM	ST	GOROD
П1	Иванов	20	Киев
П2	Петров	40	Харьков
П3	Сидоров	10	Киев

Ответ на запрос: “Найти все города, в которых размещены поставщики” может быть получен как результат выполнения операции проекции R на атрибут GOROD:

REZ = πGOROD (R) = GOROD

Киев

Харьков

Селекция

Селекция, или выбор (SELECT) дает возможность построить горизонтальное подмножество отношения. Отбор кортежей может производится с использованием операций: арифметических (+ - / *), отношений (< ≤ = # ≥ >), логических (NOT (¬), AND (∧), OR (∨)).

Математическая запись: $REZ = \sigma B = \text{“b”} (R)$,

что означает селектирование (выбор) из отношения R тех кортежей, для которых значение атрибута B равно “ b”. Например, для

R =

A	B	C
5	8	4
3	2	1

$REZ = \sigma A < (B - 1) (R) =$

A	B	C
5	8	4

Соответствующий оператор языка запросов может быть записан в виде

SELECT R WHERE A < (B - 1) GIVING REZ

Пример. Пусть R =

PN	PIM	ST	GOROD
П1	Иванов	20	Киев
П2	Петров	40	Харьков
П3	Сидоров	10	Киев

Ответ на запрос “Найти все имена поставщиков из Киева”

можно получить последовательным выполнением операций селекции и проекции:

$REZ = \sigma_{GOROD} = \text{“Киев”} (R)$ и далее

$REZ = \pi_{PIM} (REZ)$ или $REZ = \pi_{PIM} (\sigma GOROD = \text{“Киев”} (R))$.

Соответствующие операторы языка запросов запишутся в виде

SELECT R WHERE GOROD = “Киев” GIVING REZ

PROJECT REZ OVER PIM GIVING REZ

или в виде одного вложенного оператора

PROJECT (SELECT R WHERE GOROD = “Киев”) OVER PIM GIVING REZ

Соединение

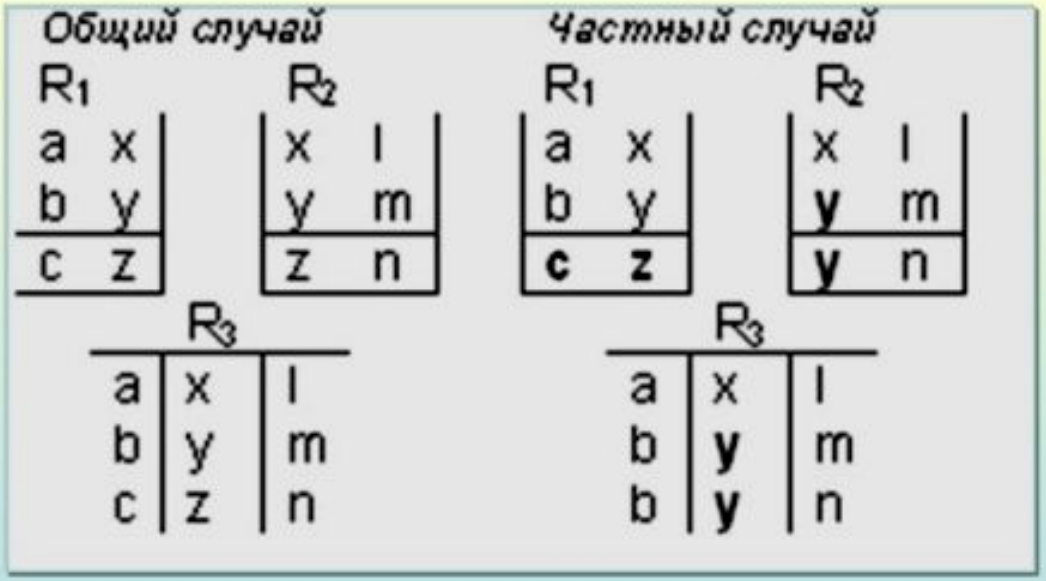
Операция Соединения (Join) имеет сходство с ДЕКАРТОВЫМ ПРОИЗВЕДЕНИЕМ.

Однако, здесь добавлено условие, согласно которому вместо полного произведения всех строк в результирующее отношение включаются только те строки, которые удовлетворяют определенному соотношению между атрибутами соединения.

Результатом соединения отношений R_1 и R_2 является отношениеR_3, кортежи которого представляют собой сцепление двух кортежей (принадлежащих соответственно R_1 и R_2), имеющих общее значение для одного или нескольких общих атрибутов R_1 и R_2.

Причем эти общие значения в результирующем отношении появляются только один раз.

R3 = R_1 JOIN R_2



Свойства соединений

Соединение обладает *свойствами*:

ассоциативность: $(R_1$ JOIN $R_2)$ JOIN $R_3 = R_1$ JOIN $(R_2$ JOIN $R_3)$;
коммутативность: $(R_1$ JOIN $R_2)$ JOIN $R_3 = R_1$ JOIN R_2 JOIN R_3.

Эта операция имеет несколько *разновидностей*, но самое распространенное – естественное соединение (на схеме). Есть еще *θ(тэта)-соединение*. Оно предназначено для случаев, когда два отношения соединяются на основе некоторых условий (***xθy***), отличных от эквивалентности.

В этом случае обозначение выглядит так: **(R_1 TIMES R_2) WHERE xθy**, т.е. сочетание произведения и выборки.

Пример θ-соединения: соединение отношений Студенты и Группы по атрибуту Группа так, чтобы получить информацию о студентах только групп 5 курса.

Проекция, соединение

Проекция выполняется над одним отношением на некоторые атрибуты. Результирующее отношение включает часть атрибутов исходного, на которые выполняется проекция, например, «Отдел» и «Должность». Кортежи-дубликаты отсутствуют:

АДМИНИСТРАТИВНАЯ ЧАСТЬ

Фамилия	Отдел	Должность
Иванова	общий отдел	начальник
Михайлова	общий отдел	сотрудник
Соколова	общий отдел	сотрудник
Котусева	отдел кадров	начальник
Ковальчук	отдел кадров	сотрудник
Егорова	отдел кадров	сотрудник

ДОЛЖНОСТИ

Отдел	Должность
общий отдел	начальник
общий отдел	сотрудник
отдел кадров	начальник
отдел кадров	сотрудник

Операция **соединения** выполняется над двумя отношениями. В каждом отношении выделяется атрибут, по которому будет производиться соединение. Если в качестве атрибута для соединения выбрать «№ зачетной книжки», то результирующее отношение будет включать все атрибуты обоих отношений:

ГРУППЫ

Специальность	№
прикл. матем.	053275
история	054117
педагогика	055421

СТУДЕНТЫ

№	Фамилия	Курс
053275	Токарев	1
053576	Мельников	1
053975	Поляков	2
054117	Карпова	1
055421	Морозова	3

СТАРОСТЫ ГРУПП

Специальность	№	Фамилия	Курс
прикл. матем.	053275	Токарев	1
история	054117	Карпова	1
педагогика	055421	Морозова	3

ДЕЛЕНИЕ ОТНОШЕНИЙ (DIVISION)

Пусть отношение **R** , называемое делимым, содержит атрибуты ***(A1,A2,...,An)***. Отношение **S** - делитель содержит подмножество атрибутов **A**: ***(A1,A2,...,Ak) (k<n)***. Результирующее отношение **C** определено на атрибутах отношения **R**, которых нет в **S**, т.е. ***Ak+1,Ak+2,...,An***. Кортежи включаются в результирующее отношение **C** только в том случае, если его декартово произведение с отношением **S** содержится в делимом **R**. Обозначается ***R[A1/A2]S***

Например, в БД Факультет есть отношение ***Занятия*** *(Группа, Дисциплина, Преподаватель). Чтобы получить список групп, которые изучают заданный набор дисциплин можно применить деление этого отношения на специально созданное унарное отношение, содержащее заданный набор дисциплин*

Декартово произведение, деление

Декартово произведение. Здесь отношения-операнды могут иметь разные схемы. Степень результирующего отношения равна сумме степеней отношений операндов, а мощность - произведению их мощностей:

СТУДЕНТЫ

Фамилия
Данилов
Козлов
Лукин

×

ЭКЗАМЕНЫ

Предмет	Дата
История	13 / 01 / 05
Физика	18 / 01 / 05

=

ЭКЗ. ВЕДОМОСТЬ

Фамилия	Предмет	Дата
Данилов	История	13 / 01 / 05
Данилов	Физика	18 / 01 / 05
Козлов	История	13 / 01 / 05
Козлов	Физика	18 / 01 / 05
Лукин	История	13 / 01 / 05
Лукин	Физика	18 / 01 / 05

Деление. Отношение-делитель должно содержать подмножество атрибутов отношения-делимого. Результирующее отношение содержит только те атрибуты делимого, которых нет в делителе. В него включают только те кортежи, декартовы произведения которых с делителем содержатся в делимом:

ЭКЗ. ВЕДОМОСТЬ

Фамилия	Предмет	Оценка
Данилов	История	5
Данилов	Физика	5
Козлов	История	4
Козлов	Физика	4
Лукин	История	4
Лукин	Физика	5

:

РЕЗУЛЬТАТЫ

Предмет	Оценка
История	5
Физика	4

=

СТУДЕНТЫ

Фамилия
Данилов
Козлов