Реляционная модель данных (РМД)

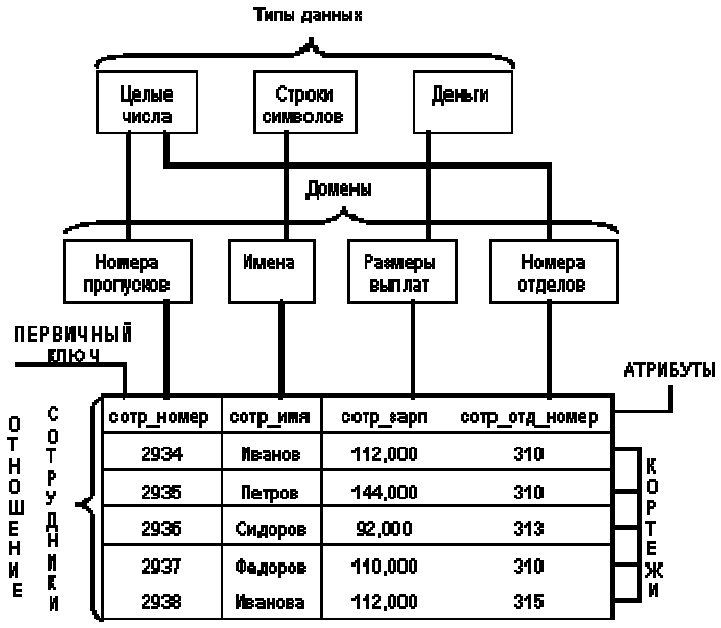
Оглавление

Реляционная модель данных была предложена в 1970 г. математиком Эдгаром Коддом (Codd E.F.). РМД является наиболее широко распространенной моделью данных и единственной из трёх основных моделей данных, для которой разработан теоретический базис с использованием теории множеств.

Основные понятия

Основными понятиями реляционных баз данных являются тип данных, домен, атрибут, кортеж, первичный ключ и отношение.

Для начала покажем смысл этих понятий на примере отношения СОТРУДНИКИ, содержащего информацию о сотрудниках некоторой организации:



Тип данных

Понятие *тип данных* в реляционной модели данных полностью адекватно понятию типа данных в языках программирования. Обычно в современных реляционных БД допускается хранение символьных, числовых данных, битовых строк, специализированных числовых данных (таких как "деньги"), а также специальных "темпоральных" данных (дата, время, временной интервал). Достаточно активно развивается подход к расширению возможностей реляционных систем абстрактными типами данных (соответствующими возможностями обладают, например, системы семейства Ingres/Postgres). В нашем примере мы имеем дело с данными трех типов: строки символов, целые числа и "деньги".

Домен

Понятие *домена* более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в некоторых языках программирования. В самом общем виде домен определяется заданием некоторого базового типа данных, к которому относятся элементы домена, и произвольного логического выражения, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат "истина", то элемент данных является элементом домена.

Наиболее правильной интуитивной трактовкой понятия домена является понимание домена как допустимого потенциального множества значений данного типа. Например, домен "Имена" в нашем примере определен на базовом типе строк символов, но в число его значений могут входить только те строки, которые могут изображать имя (в частности, такие строки не могут начинаться с мягкого знака).

Следует отметить также семантическую нагрузку понятия домена: данные считаются сравнимыми только в том случае, когда они относятся к одному домену. В нашем примере значения доменов "Номера пропусков" и "Номера групп" относятся к типу целых чисел, но не являются сравнимыми

Понятие отношения

Итак, базовой структурой РМД является **отношение**, основанное на декартовом произведении доменов. **Домен** – это множество значений, которое может принимать элемент данных (например, множество целых чисел, множество дат, множество комбинаций символов длиной N и т.п.). Домен может задаваться перечислением элементов, указанием диапазона значений, функцией и т.д.

Пусть D_1, D_2 ,…, D_k – произвольные конечные и не обязательно различные множества (домены). **Декартово произведение** этих множеств определяется следующим образом:

$$D_1 \times D_2 \times ... \times D_k = \{(d_1, d_2, ..., d_k) \mid d_i \in D_i, i=1,\ldots,k\}.$$

Таким образом, декартово произведение позволяет получить все возможные комбинации значений элементов исходных множеств.

Пример. Для доменов $D_1 = (1, 2)$, $D_2 = (A, B, C)$ декартово произведение $D = D_1 \times D_2$ будет таким: $D = \{(1,A), (1,B), (1,C), (2,A), (2,B), (2,C)\}$.

Подмножество декартова произведения доменов называется **отношением**.

Отношение содержит данные о сущностях определённого типа. Поясним это на примере. Если построить произведение трёх доменов *Должности* ('директор', 'бухгалтер', 'водитель', 'продавец'), *Оклады* (x | 20000≤x≤80000), *Надбавки* (1.1, 1.2, 1.3), то мы получим 4*60001*3=720012 комбинаций. Но реально отношение «Штатное расписание» содержит по одной строке на каждую должность, т.е. является именно подмножеством декартова произведения доменов.

Элементы отношения называют *кортежами* (или *записями*). Каждый кортеж отношения соответствует одному экземпляру сущности определённого типа. Элементы кортежа принято называть *атрибутами* (или *полями*).

Свойства отношений

Отношение обладает двумя основными свойствами:

1. В отношении не должно быть одинаковых кортежей, т.к. это множество.
2. Порядок кортежей в отношении несущественен.
   Таким образом, в отношении не бывает первого, второго или последнего кортежа: при выводе данных отношения кортежи выводятся в произвольном порядке, если не задано упорядочение по значениям полей.

Отношение удобно представлять как таблицу, где строка является кортежем, а столбец соответствует домену (рис. 1, отношение *СТУДЕНТЫ*). Количество строк в таблице (кортежей в отношении) называется **мощностью отношения**, количество столбцов (атрибутов) – **арностью**.

домен 1	домен 2	домен 3 (ключ)	домен 4	домен 5
Группа	*ФИО студента*	***Номер зачётной книжки***	*Год рождения*	*Размер стипендии*
С–72	Волкова Елена Павловна	С-12298	1991	1550.00
С–91	Белов Сергей Юрьевич	С-12299	1990	1400.00
		. . .		
С–72	Фролов Юрий Вадимович	С-14407	1991	0

Рис.1. Пример табличной формы представления отношения

Отношение имеет имя, которое отличает его от имён всех других отношений. Атрибутам реляционного отношения назначаются имена, уникальные в рамках отношения. Обращение к отношению происходит по его имени, а обращение к атрибуту – по имени отношения и имени атрибута.

Каждый атрибут определён на некотором домене, несколько атрибутов отношения могут быть определены на одном и том же домене (например, номера рабочего и домашнего телефонов). Домен задаётся типом данных, размером и ограничениями целостности: например, пол – это символьное поле дли-

ной 1, которое может принимать значения из множества ('м', 'ж'). В реляционных базах данных поддерживаются такие типы данных как символьный, числовой, дата и некоторые другие (конкретный перечень типов зависит от СУБД).

Атрибут может быть обязательным и необязательным. Значение обязательного атрибута должно быть определено в момент внесения данных в БД. Если атрибут необязательный, то для таких случаев предусмотрено специальное значение – NULL, которое можно интерпретировать как "неизвестное значение". Значение NULL не привязано к определённому типу данных, т.е. может назначаться данным любых типов.

Перечень атрибутов отношения с их типами данных и размерами определяют **схему отношения**. Отношения, построенные по одинаковой схеме, называют **односхемными**; по различным схемам – **разносхемными**.

Ключ отношения – это атрибут (группа атрибутов), значения которого классифицируют или идентифицируют кортеж. Например, значение атрибута *Группа* отношения *СТУДЕНТЫ* позволяет выделить среди всех студентов института студентов конкретной группы. Если ключ состоит из нескольких атрибутов, он называется *составным*. Если значения ключа уникальны в рамках столбца отношения, то такой ключ называется *потенциальным*. Потенциальных ключей может быть несколько (или не быть ни одного), но для отношения выделяется один основной ключ – первичный. **Первичный ключ** идентифицирует экземпляр сущности, его значение должно быть уникальным (*unique*) и обязательным (*not null*). (На рис. 1 первичный ключ выделен полужирным шрифтом). Неуникальные ключи ещё называют *вторичными*.

РМД не поддерживает групповые отношения (по версии CODASYL). Для связей между отношениями используются внешние ключи. Внешний ключ (*foreign key*) – это атрибут подчинённого (дочернего) отношения, который является копией первичного (*primary key*) или уникального (*unique*) ключа родительского отношения. (Пример – отношение *ОЦЕНКИ*, связанное с отношением *СТУДЕНТЫ* внешним ключом *Номер зачётной книжки*, рис. 2).

Номер зачётной книжки	*Дисциплина*	*Оценка*
С-12298	Программирование	5
С-12298	Дискретная математика	4
С-14407	Программирование	3
…	…	…

Рис.2. Связь отношений "Оценки" и "Студенты" по внешнему ключу

Если связь необязательная, то значение внешнего ключа может быть неопределённым (*null*).

Фактически внешние ключи логически связывают экземпляры сущностей разных типов (родительской и подчинённой сущностей).

Внешний ключ – это ограничение целостности, в соответствии с которым множество значений внешнего ключа является подмножеством значений первичного или уникального ключа родительской таблицы.

Ограничение целостности по внешнему ключу проверяется в двух случаях:

- при добавлении записи в подчинённую таблицу СУБД проверяет, что в родительской таблице есть запись с таким же значением первичного ключа;
- при удалении записи из родительской таблицы СУБД проверяет, что в подчинённой таблице нет записей с таким же значением внешнего ключа.

Примечание: внешний ключ может ссылаться на первичный ключ этой же таблицы. Это позволяет описывать унарную связь – иерархию однотипных сущностей. Например, если в таблицу *СОТРУДНИКИ* добавить поле *Руководитель* и описать его как внешний ключ на эту же таблицу, то в этом поле будет храниться идентификатор руководителя данного сотрудника (рис. 3). Атрибут *Руководитель* является необязательным.

Табельный номер	*№ отдела*	*ФИО*	*Должность*	***Руководитель***
002	1	Сухов К.А.	директор	
034	1	Петрова К.В.	секретарь	002
988	2	Рюмин В.П.	начальник отдела	002
909	2	Серова Т.В.	вед. программист	988

Рис.3. Внешний ключ "Руководитель", ссылающийся на первичный ключ этой же таблицы

Все операции над данными в РМД выполняются над отношением и требуют задания имени отношения. Если операция применяется к части отношения, то может потребоваться идентификация кортежа или группы кортежей и задание имён атрибутов. В РМД используются следующие операции:

- *запомнить*: внесение информации в БД (требует формирования значений уникального ключа и обязательных атрибутов кортежа);
- *извлечь*: чтение данных;
- *обновить*: модификация данных – изменение значений атрибутов кортежей;
- *удалить*: физическое или логическое удаление данных (кортежей).

Структуризация данных в РМД существенно отличается от структуризации данных по версии CODASYL (см. табл. 1).

Таблица 1. Сравнение структуризации данных в РМД и по версии CODASYL

Термины версии CODASYL	*Термины (и синонимы) РМД*
Элемент данных	Атрибут (поле)
Агрегат	—
Запись (группа)	Кортеж (запись, строка)
Совокупность записей одного типа	Отношение (таблица)
Набор (групповое отношение)	—
База данных	База данных

Примечание: в реляционной модели данных набор (групповое отношение) моделируется с помощью внешнего ключа, описывающего связь между двумя таблицами.

Достоинства и недостатки РМД

Широкое распространение реляционной модели объясняется в первую очередь простотой представления и формирования базы данных, универсальностью и удобством обработки данных, которая осуществляется с помощью декларативного языка запросов SQL (Structured Query Language).

Моделирование предметной области в рамках реляционной модели создаёт некоторые сложности, т.к. в этой модели нет специальных средств для отображения различных типов связей и агрегатов. Отсутствие агрегатов приводит к тому, что при проектировании реляционной БД приходится проводить нормализацию отношений. После нормализации данные об одной сущности предметной области распределяются по нескольким таблицам, что усложняет работу с БД.

Отсутствие специальных механизмов навигации (как в иерархической или сетевой моделях), с одной стороны, ведёт к упрощению модели, а с другой – к многократному увеличению времени на извлечение данных, т.к. во многих случаях требуется просмотреть всё отношение для поиска нужных данных.

В РМД нет понятий "режим включения" и "класс членства". Но с помощью внешних ключей и дополнительных возможностей СУБД их можно эмулировать.

Понятие реляционной алгебры

Итак, реляционная модель данных – это модель данных, основанная на представлении данных в виде набора отношений, каждое из которых является подмножеством декартова произведения определённых множеств. Манипулирование данными в РМД осуществляется с помощью операций реляционной алгебры (РА) или реляционного исчисления. Реляционная алгебра основана на теории множеств, а реляционное исчисление базируется на математической логике (вернее, на исчислении предикатов первого порядка).

В реализациях конкретных реляционных СУБД сейчас не используется в чистом виде ни реляционная алгебра, ни реляционное исчисление. Фактическим стандартом доступа к реляционным данным стал язык SQL (Structured Query Language). Язык SQL представляет собой смесь операторов реляционной алгебры и выражений реляционного исчисления, использующий синтаксис, близкий к фразам английского языка и расширенный дополнительными возможностями, отсутствующими в реляционной алгебре и реляционном исчислении. Вообще, ***язык доступа к данным называется реляционно полным, если он по выразительной силе не уступает реляционной алгебре*** (или, что-то же самое, реляционному исчислению), т.е. любой оператор реляционной алгебры может быть выражен средствами этого языка.

Рассмотрим операции реляционной алгебры.

Операции реляционной алгебры

Операндами для операций реляционной алгебры являются реляционные отношения. Результатом выполнения операций РА также является отношение.

Таким образом, механизм реляционной алгебры замкнут относительно понятия отношения. Это позволяет применять операции РА каскадно.

Реляционный оператор f выглядит как функция с отношениями в качестве аргументов:

$$R = f(R_1, R_2, \ldots, R_n)$$

Реляционная алгебра является замкнутой, т.к. в качестве аргументов в реляционные операторы можно подставлять другие реляционные операторы, подходящие по типу:

$$R = f(f_1(R_{11}, R_{12}, \ldots), f_2(R_{21}, R_{22}, \ldots), \ldots);$$

Таким образом, в реляционных выражениях можно использовать вложенные выражения сколь угодно сложной структуры.

Использование операций РА накладывает на отношения два ограничения:

- порядок столбцов (полей) в отношении фиксирован;
- отношения конечны.

Каждое отношение обязано иметь уникальное имя в пределах базы данных. Имя отношения, полученного в результате выполнения реляционной операции, определяется в левой части равенства. Однако можно не требовать наличия имен от отношений, полученных в результате реляционных выражений, если эти отношения подставляются в качестве аргументов в другие реляционные выражения. Такие отношения будем называть ***неименованными отношениями***. Неименованные отношения реально не существуют в базе данных, а только вычисляются в момент вычисления значения реляционного оператора.

Традиционно, по Кодду, определяют восемь реляционных операторов, объединенных в две группы.

Теоретико-множественные операторы:

- Объединение
- Пересечение
- Вычитание
- Декартово произведение

Специальные реляционные операторы:

- Выборка
- Проекция
- Соединение
- Деление

Можно выделить также пять основных операций реляционной алгебры – проекция, селекция, декартово произведение, разность, объединение, – и три вспомогательных: соединение, пересечение и деление. Вспомогательные операции могут быть выражены через основные, но в некоторых системах реализуются с помощью специальных команд (ключевых слов) для удобства пользователей.

1. **Проекция** (projection).

Определение. ***Проекцией*** отношения *A* по атрибутам *X,Y,..,Z*, где каждый из атрибутов принадлежит отношению *A*, называется отношение с заголовком *(X,Y,..,Z)* и телом, содержащим множество кортежей вида *(x,y,..,z)*, таких, для которых в отношении *A* найдутся кортежи со значением атрибута *X* равным *x*, значением атрибута *Y* равным *y*, …, значением атрибута *Z* равным *z*.

Другими словами, это унарная операция (выполняемая над одним отношением), служащая для выбора подмножества атрибутов из отношения R. Она уменьшает арность отношения и может уменьшить мощность отношения за счёт исключения одинаковых кортежей.

Операция проекции дает "*вертикальный срез*" отношения, в котором удалены все возникшие при таком срезе дубликаты кортежей.

Пример 1. Пусть имеется отношение R(A,B,C) (рис. 3,а).

Тогда проекция $\pi_{A,C}(R)$ будет такой, как показано на рис.3,б.

Отношение R

A	B	C
a	b	c
c	a	d
c	b	d

а)

Проекция $\pi_{A,C}$(R)

A	C
a	c
c	d

б)

Рис.3. Пример проекции отношения

Пример 1.1 Пусть дано отношение *A* с информацией о поставщиках, включающих наименование и месторасположение:

Номер поставщика	**Наименование поставщика**	**Город поставщика**
1	Иванов	Уфа
2	Петров	Москва
3	Сидоров	Москва
4	Сидоров	Донецк

Проекция *A[Город поставщика]* будет иметь вид:

Город поставщика
Уфа
Москва
Донецк

2. **Селекция** (selection).

Определение. ***Выборкой (ограничением, селекцией)*** на *A* отношении с условием *c* называется отношение с тем же заголовком, что и у отношения *A*, и телом, состоящем из кортежей, значения атрибутов которых при подстановке

в условие *c* дают значение ИСТИНА. *c* представляет собой логическое выражение, в которое могут входить атрибуты отношения *A* и (или) скалярные выражения.

Другими словами, это унарная операция, результатом которой является подмножество кортежей исходного отношения, соответствующих условиям, которые накладываются на значения определённых атрибутов.

В простейшем случае условие c имеет вид $X \Theta Y$, где Θ - один из операторов сравнения ($=, \neq, <, \leq, >, \geq$ и т.д.), а X и Y - атрибуты отношения A или скалярные значения. Такие выборки называются Θ-***выборки*** (***тэта-выборки***) или Θ-***ограничения***, Θ-***селекции***.

Пример 2. Для отношения R(A,B,C) (рис.4,а) селекция $\sigma_{C=d}(R)$ (при условии "значение атрибута C равно d") будет такой (рис.4,б):

Отношение R

A	B	C
a	b	c
c	a	d
c	b	d

а)

Селекция $\sigma_{C=d}(R)$

A	B	C
c	a	d
c	b	d

б)

Рис.4. Пример селекции отношения

Пример 2.1 Пусть дано отношение *A* с информацией о сотрудниках:

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Петров	2000
3	Сидоров	3000

Результат выборки *A* WHERE Зарплата < 3000 будет иметь вид:

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Петров	2000

Смысл операции выборки очевиден - выбрать кортежи отношения, удовлетворяющие некоторому условию. Таким образом, операция выборки дает "*горизонтальный срез*" отношения по некоторому условию.

3. **Декартово произведение** (Cartesian product).

Определение 5. ***Декартовым произведением*** двух отношений $A(A_1, A_2, \ldots, A_n)$ и $B(B_1, B_2, \ldots, B_m)$ называется отношение, заголовок которого является ***сцеплением заголовков*** отношений A и B:

$$(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m),$$

а тело состоит из кортежей, являющихся ***сцеплением кортежей*** отношений A и B:

$$(a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m),$$

таких, что $(a_1, a_2, \ldots, a_n) \in A$, $(b_1, b_2, \ldots, b_m) \in B$.

Другими словами, это бинарная операция над разносхемными отношениями, соответствующая определению декартова произведения для РМД.

Мощность произведения $A \times B$ равна произведению мощностей отношений A и B, т.к. каждый кортеж отношения A соединяется с каждым кортежем отношения B.

Если в отношениях A и B имеются атрибуты с одинаковыми наименованиями, то перед выполнением операции декартового произведения такие атрибуты необходимо переименовать.

Перемножать можно любые два отношения, совместимость по типу при этом не требуется.

Пример 3. Пусть имеются отношение R(A,B) и отношение S(C,D,E) (рис.5,а).

Тогда декартово произведение R×S будет таким (рис.5,б).

Отношение R

A	B
a	b
c	a
b	d

Отношение S

C	D	E
1	2	3
4	5	6

а)

Декартово произведение R×S

A	B	C	D	E
a	b	1	2	3
a	b	4	5	6
c	a	1	2	3
c	a	4	5	6
b	d	1	2	3
b	d	4	5	6

б)

Рис.5. Пример декартова произведения отношений

Пример 3.1. Пусть даны два отношения A и B с информацией о поставщиках и деталях:

Номер поставщика	**Наименование поставщика**
1	Иванов
2	Петров
3	Сидоров

Номер детали	**Наименование детали**
1	Болт
2	Гайка
3	Винт

Декартово произведение отношений *A* и *B* будет иметь вид:

Номер поставщика	**Наименование поставщика**	**Номер детали**	**Наименование детали**
1	Иванов	1	Болт
1	Иванов	2	Гайка
1	Иванов	3	Винт
2	Петров	1	Болт
2	Петров	2	Гайка
2	Петров	3	Винт
3	Сидоров	1	Болт
3	Сидоров	2	Гайка
3	Сидоров	3	Винт

Сама по себе операция декартового произведения не очень важна, т.к. она не дает никакой новой информации, по сравнению с исходными отношениями. Для реальных запросов эта операция почти никогда не используется. Однако операция декартового произведения важна для выполнения специальных реляционных операций.

4. **Объединение** (union).

Определение. ***Объединением*** двух совместимых по типу отношений *A* и *B* называется отношение с тем же заголовком, что и у отношений *A* и *B*, и телом, состоящим из кортежей, принадлежащих или *A*, или *B*, или обоим отношениям.

Или, объединением двух односхемных отношений R и S называется отношение T = R ∪ S, которое включает в себя все кортежи обоих отношений без повторов.

Пример 4. Пусть даны два отношения A и B с информацией о сотрудниках:

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Петров	2000

3	Сидоров	3000

Таблица 1 Отношение A

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Пушников	2500
4	Сидоров	3000

Отношение B

Объединение отношений A и B будет иметь вид:

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000
2	Петров	2000
3	Сидоров	3000
2	Пушников	2500
4	Сидоров	3000

Отношение A UNION B

Замечание. Как видно из приведенного примера, потенциальные ключи, которые были в отношениях A и B *не наследуются* объединением этих отношений. Поэтому, в объединении отношений A и B атрибут "Табельный номер" может содержать дубликаты значений. Если бы это было не так, и ключи наследовались бы, то это противоречило бы понятию объединения как "объединение множеств". Конечно, объединение отношений A и B имеет, как и любое отношение, потенциальный ключ, например, состоящий из всех атрибутов.

5. **Разность** (minus).

Определение. ***Вычитанием*** двух совместимых по типу отношений A и B называется отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих отношению A и не принадлежащих отношению B.

Другими словами, разностью односхемных отношений R и S называется множество кортежей R, не входящих в S.

Пример 5. Пусть имеются отношение R(A,B,C) и отношение S(A,B,C) (рис.6,а). Тогда разность R–S будет такой (рис.6,б):

Отношение R

A	B	C
a	b	c
c	a	d

Отношение S

A	B	C
g	h	a
a	b	c

Разность R–S

A	B	C
c	a	d
c	h	c

c	h	c

а)

h	d	d

б)

Рис.6. Пример разности отношений

Пример 5.1. Для тех же отношений *A* и *B*, что в примере 4 вычитание имеет вид:

Табельный номер	**Фамилия**	**Зарплата**
2	Петров	2000
3	Сидоров	3000

Следующие три операции являются вспомогательными операциями РА.

6. **Пересечение** (intersection).

Определение. ***Пересечением*** двух совместимых по типу отношений A и B называется отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих одновременно обоим отношениям A и B.

Другими словами, пересечение двух односхемных отношений R и S есть подмножество кортежей, принадлежащих обоим отношениям. Это можно выразить через разность:

$$R \cap S = R - (R - S).$$

Пример 6. Для тех же отношений A и B, что в примере 4 пересечение имеет вид:

Табельный номер	**Фамилия**	**Зарплата**
1	Иванов	1000

7. **Соединение** (join).

Операция соединения отношений, наряду с операциями выборки и проекции, является одной из наиболее важных реляционных операций.

Обычно рассматривается несколько разновидностей операции соединения:

- Общая операция соединения
- Тэта-соединение
- Экви-соединение
- Естественное соединение

Наиболее важным из этих частных случаев является операция естественного соединения. Все разновидности соединения являются частными случаями общей операции соединения.

Общая операция соединения

Эта операция определяет подмножество декартова произведения двух разносхемных отношений. Кортеж декартова произведения входит в результирующее отношение, если для атрибутов разных исходных отношений выполняется некоторое условие F. Соединение может быть выражено так:

$$R \underset{F}{\triangleright\triangleleft} S = \sigma_F (R \times S).$$

Таким образом, операция соединения есть результат последовательного применения операций декартового произведения и выборки. Если в отношениях *R* и *S* имеются атрибуты с одинаковыми наименованиями, то перед выполнением соединения такие атрибуты необходимо переименовать.

Тэта-соединение

Определение. Пусть отношение A содержит атрибут X, отношение B содержит атрибут Y, а Θ - один из операторов сравнения ($=, \neq, <, \leq, >, \geq$ и т.д.). Тогда Θ-***соединением*** отношения A по атрибуту X с отношением B по атрибуту Y называют отношение

$$(A \text{ TIMES } B) \text{ WHERE } X \Theta Y$$

Это частный случай операции общего соединения.

Иногда, для операции Θ-соединения применяют следующий, более короткий синтаксис:

$$A[X \Theta Y]B$$

Пример 7. Рассмотрим некоторую компанию, в которой хранятся данные о поставщиках и поставляемых деталях. Пусть поставщикам и деталям присвоен некий статус. Пусть бизнес компании организован таким образом, что поставщики имеют право поставлять только те детали, статус которых не выше статуса поставщика (смысл этого может быть в том, что хороший поставщик с высоким статусом может поставлять больше разновидностей деталей, а плохой поставщик с низким статусом может поставлять только ограниченный список деталей, важность которых (статус детали) не очень высока).

Номер поставщика	Наименование поставщика	X (Статус поставщика)
1	Иванов	4
2	Петров	1
3	Сидоров	2

Отношение A (Поставщики)

Номер детали	Наименование детали	Y (Статус детали)
1	Болт	3
2	Гайка	2

3	Винт	1

Отношение B (Детали)

Ответ на вопрос "какие поставщики имеют право поставлять какие детали?" дает ⍟ -соединение $A[X \geq Y]B$:

Номер поставщика	Наименование поставщика	X (Статус поставщика)	Номер детали	Наименование детали	Y (Статус детали)
1	Иванов	4	1	Болт	3
1	Иванов	4	2	Гайка	2
1	Иванов	4	3	Винт	1
2	Петров	1	3	Винт	1
3	Сидоров	2	2	Гайка	2
3	Сидоров	2	3	Винт	1

Отношение "Какие поставщики поставляют какие детали"

Экви-соединение

Наиболее важным частным случаем ⍟ -соединения является случай, когда ⍟ есть просто равенство.

Если условием является равенство атрибутов исходных отношений, такая операция называется *эквисоединением*.

Пример 7.1. Пусть имеются отношения P, D и PD, хранящие информацию о поставщиках, деталях и поставках соответственно (для удобства введем краткие наименования атрибутов):

Номер поставщика PNUM	Наименование поставщика PNAME
1	Иванов
2	Петров
3	Сидоров

Отношение P (Поставщики)

Номер детали DNUM	Наименование детали DNAME
1	Болт
2	Гайка
3	Винт

Отношение D (Детали)

Номер поставщика PNUM	*Номер детали DNUM*	Поставляемое количество VOLUME
1	*1*	100
1	*2*	200
1	*3*	300
2	*1*	150
2	*2*	250
3	*1*	1000

Отношение PD (Поставки)

Ответ на вопрос, какие детали поставляются поставщиками, дает экви-соединение $P[PNUM = PNUM]PD$. На самом деле, т.к. в отношениях имеются одинаковые атрибуты, то требуется сначала переименовать атрибуты, а потом выполнить экви-соединение. В результате имеем отношение:

Номер поставщика PNUM1	Наименование поставщика PNAME	Номер поставщика PNUM2	Номер детали DNUM	Поставляемое количество VOLUME
1	Иванов	1	1	100
1	Иванов	1	2	200
1	Иванов	1	3	300
2	Петров	2	1	150
2	Петров	2	2	250
3	Сидоров	3	1	1000

Отношение "Какие детали поставляются какими поставщиками"

Недостатком экви-соединения является то, что если соединение происходит по атрибутам с одинаковыми наименованиями (а так чаще всего и происходит), то в результирующем отношении появляется два атрибута с одинаковыми значениями. В нашем примере атрибуты PNUM1 и PNUM2 содержат дублирующие данные. Избавиться от этого недостатка можно, взяв проекцию по всем атрибутам, кроме одного из дублирующих. Именно так действует естественное соединение.

Естественное соединение – эквисоединение по одинаковым атрибутам исходных отношений.

Определение. Пусть даны отношения $A(A_1, A_2, \ldots, A_n, X_1, X_2, \ldots, X_p)$ и $B(X_1, X_2, \ldots, X_p, B_1, B_2, \ldots, B_m)$, имеющие одинаковые атрибуты $X_1, X_2, \ldots X_p$ (т.е. атрибуты с одинаковыми именами и определенные на одинаковых доменах).

Тогда ***естественным соединением*** отношений A и B называется отношение с заголовком $(A_1, A_2, \ldots, A_n, X_1, X_2, \ldots, X_p, B_1, B_2, \ldots, B_m)$ и телом, содержащим множество кортежей $(a_1, a_2, \ldots a_n, x_1, x_2, \ldots, x_p, b_1, b_2, \ldots b_m)$, таких, что $(a_1, a_2, \ldots a_n, x_1, x_2, \ldots, x_p) \in A$ и $(x_1, x_2, \ldots, x_p, b_1, b_2, \ldots b_m) \in B$.

Естественное соединение эквивалентно следующей последовательности реляционных операций:

1. Переименовать одинаковые атрибуты в отношениях
2. Выполнить декартово произведение отношений
3. Выполнить выборку по совпадающим значениям атрибутов, имевших одинаковые имена
4. Выполнить проекцию, удалив повторяющиеся атрибуты
5. Переименовать атрибуты, вернув им первоначальные имена

Можно выполнять последовательное естественное соединение нескольких отношений. Естественное соединение (как и соединение общего вида) обладает свойством ***ассоциативности***, т.е.

$(A \text{ JOIN } B) \text{ JOIN } C = A \text{ JOIN } (B \text{ JOIN } C)$

поэтому такие соединения можно записывать, опуская скобки:

$A \text{ JOIN } B \text{ JOIN } C$

Пример 7.2. Пусть имеются отношения R(A,B,C) и S(A,D,E) (рис.7,а). Тогда естественное соединение R ▷◁ S будет таким, как показано на рис.7,б.

Отношение R

A	B	C
a	b	c
c	a	d
c	h	c
g	b	d

Отношение S

A	D	E
g	h	a
c	b	c
h	d	d

а)

Соединение R▷◁S

A	B	C	D	E
c	a	d	b	c
c	h	c	b	c
g	b	d	h	a

б)

Рис.7. Пример естественного соединения отношений

Пример 7.3. В примере 7.1 ответ на вопрос "какие детали поставляются поставщиками", более просто записывается в виде естественного соединения трех отношений $P \text{ JOIN } PD \text{ JOIN } D$ (для удобства просмотра порядок атрибутов изменен, это является допустимым по свойствам отношений):

Номер поставщика PNUM	Наименование поставщика PNAME	Номер детали DNUM	Наименование детали DNAME	Поставляемое количество VOLUME
1	Иванов	1	Болт	100
1	Иванов	2	Гайка	200
1	Иванов	3	Винт	300

2	Петров	1	Болт	150
2	Петров	2	Гайка	250
3	Сидоров	1	Болт	1000

Отношение P JOIN PD JOIN D

8. **Деление** (division).

Определение. Пусть даны отношения $A(X_1,X_2,\ldots,X_n,Y_1,Y_2,\ldots,Y_m)$ и $B(Y_1,Y_2,\ldots,Y_m)$, причем атрибуты $Y_1,Y_2,\ldots Y_m$ - общие для двух отношений. ***Делением отношений*** A на B называется отношение с заголовком $(X_1,X_2,\ldots,X_n)$ и телом, содержащим множество кортежей $(x_1,x_2,\ldots,x_n)$, таких, что для *всех* кортежей $(y_1,y_2,\ldots y_m)\in B$ в отношении A найдется кортеж $(x_1,x_2,\ldots,x_n,y_1,y_2,\ldots y_m)$.

Отношение A выступает в роли ***делимого***, отношение B выступает в роли ***делителя***. Деление отношений аналогично делению чисел с остатком.

Пример 8. Пусть имеются отношения R(A,B,C) и S(A,B) (рис.8,а). Тогда частное R/S будет таким, как показано на рис.8,б.

Отношение R

A	B	C	D
a	b	c	b
c	f	g	h
a	v	c	b
a	b	g	h
c	v	g	h
c	f	c	b

Отношение S

C	D
g	h
c	b

Частное R/S

A	B
a	b
c	f

а) б)

Рис.8. Пример операции деления

Типичные запросы, реализуемые с помощью операции деления, обычно в своей формулировке имеют слово "все" - "какие поставщики поставляют *все* детали?".

Пример 8.1. В примере с поставщиками, деталями и поставками ответим на вопрос, "какие поставщики поставляют *все* детали?".

В качестве делимого возьмем проекцию $X = PD[PNUM, DNUM]$, содержащую номера поставщиков и номера поставляемых ими деталей:

Номер поставщика PNUM	Номер детали DNUM
1	1
1	2
1	3
2	1

2	2
3	1

Проекция X=PD[PNUM,DNUM]

В качестве делителя возьмем проекцию $Y = D[DNUM]$, содержащую список номеров *всех* деталей (не обязательно поставляемых кем-либо):

Номер детали DNUM
1
2
3

Проекция Y=D[DNUM]

Деление X DEVIDEBY Y дает список номеров поставщиков, поставляющих *все* детали:

Номер поставщика PNUM
1

Отношение X DEVIDEBY Y

Оказалось, что только поставщик с номером 1 поставляет все детали.

Как уже упоминалось, языком обработки данных, основанным на реляционной алгебре, является SQL, который мы рассмотрим в следующих лекциях.