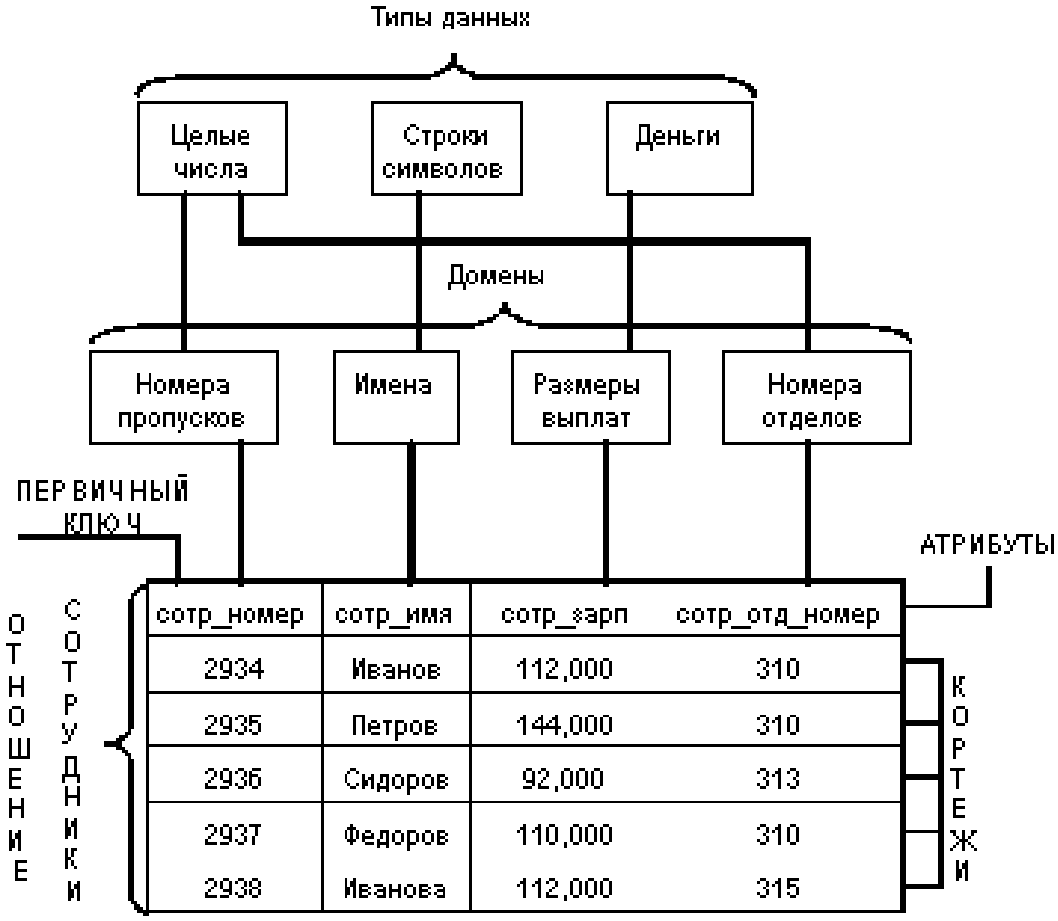
3 РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ

Введем на сравнительно неформальном уровне основные понятия реляционных баз данных, а также определим существо реляционной модели данных.

3.1 Базовые 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:



3.1.1 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».

3.1.2 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. Заметим, что в большинстве реляционных СУБД понятие домена не используется, хотя в Oracle V.7 оно уже поддерживается.

3.1.3 Схема отношения, схема базы данных

Схема отношения R – это именованное множество пар (имя атрибута, имя домена):

$$S_R = \{(A_i, D_i)\}.$$

Схема отношения представляет в реляционной БД *модель* объекта предметной области. Понятие схемы отношения ближе всего к понятию структурного типа данных в языках программирования. Так, если понятие домена не поддерживается в конкретной реализации СУБД, то вместо него в схеме указывается тип данных для каждого из атрибутов.

Степень отношения – это мощность (количество элементов) схемы отношения. Степень отношения СОТРУДНИКИ равна четырем, то есть оно является 4-арным. Если все атрибуты одного отношения определены на разных доменах, осмысленно использовать для именования атрибутов имена соответствующих доменов (не забывая, конечно, о том, что это является всего лишь удобным способом именования и не устраняет различия между понятиями домена и атрибута).

Схемы двух отношений S_R и S_Q называют *эквивалентными* ($S_R \sim S_Q$), если:

- степени отношений R и Q одинаковы;
- возможна такая перестановка местами атрибутов в схемах, чтобы на одинаковых местах стояли атрибуты с одинаковыми доменами.

Схема БД – это набор именованных схем отношений.

3.1.4 Кортеж, отношение-экземпляр

Кортеж – это множество пар (имя атрибута, значение), которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. Попросту говоря, кортеж – это набор именованных значений атрибутов. Каждый кортеж относится к конкретному объекту предметной области.

«Значение» является допустимым значением домена данного атрибута. Тем самым, степень или «арность» кортежа, т.е. число элементов в нем, совпадает с «арностью» соответствующей схемы отношения.

Отношение-экземпляр – это множество кортежей, соответствующих одной схеме отношения. Иногда схему отношения называют *заголовком отношения*, а отношение-экземпляр – *телом отношения*.

Было бы вполне логично отдельно определять схему отношения, а затем одно или несколько отношений-экземпляров с данной схемой. Однако в реляционных базах данных это не принято. Имя схемы отношения в таких базах данных всегда совпадает с именем соответствующего отношения-экземпляра. В классических реляционных базах данных после определения схемы базы данных изменяются 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*.

Обычным житейским представлением отношения является таблица, заголовком которой является схема отношения, а строками – кортежи отношения-экземпляра; в этом случае имена атрибутов именуют столбцы этой таблицы. Поэтому иногда говорят «столбец таблицы», имея в виду «атрибут отношения». Когда мы перейдем к рассмотрению практических вопросов организации реляционных баз данных и средств управления, мы будем использовать эту житейскую терминологию. Этой терминологии придерживаются в большинстве коммерческих реляционных СУБД.

Реляционная база данных – это набор отношений, имена которых совпадают с именами схем отношений в схеме БД.

Как видно, основные структурные понятия реляционной модели данных (если не считать понятия домена) имеют очень простую интуитивную интерпретацию, хотя в теории реляционных БД все они определяются абсолютно формально и точно.

3.2 Фундаментальные свойства отношений

Остановимся теперь на некоторых важных свойствах отношений, которые следуют из приведенных ранее определений.

3.2.1 Отсутствие кортежей-дубликатов

То свойство, что отношения не содержат кортежей-дубликатов, следует из определения отношения как множества кортежей. В классической теории множеств по определению каждое множество состоит из различных элементов.

Из этого свойства вытекает наличие у каждого отношения так называемого *первичного ключа* – набора атрибутов, значения которых однозначно определяют кортеж отношения. Для каждого отношения, по крайней мере, полный набор его атрибутов обладает этим свойством. Однако при формальном определении первичного ключа требуется обеспечение его «минимальности», т.е. в набор атрибутов первичного ключа не должны входить такие атрибуты, которые можно отбросить без ущерба для основного свойства – однозначно определять кортеж. Понятие первичного ключа является исключительно важным в связи с понятием целостности баз данных.

Забегая вперед, заметим, что во многих практических реализациях РСУБД допускается нарушение свойства уникальности кортежей для промежуточных отношений, порождаемых неявно при выполнении запросов. Такие отношения являются не множествами, а мультимножествами, что в ряде случаев позволяет добиться определенных преимуществ, но иногда приводит к серьезным проблемам.

3.2.2 Отсутствие упорядоченности кортежей

Свойство отсутствия упорядоченности кортежей отношения также является следствием определения отношения-экземпляра как множества кортежей. Отсутствие требования к поддержанию порядка на множестве кортежей отношения дает дополнительную гибкость СУБД при хранении баз данных во внешней памяти и при выполнении запросов к базе данных. Это не противоречит тому, что при формулировании запроса к БД, например, на языке SQL можно потребовать сортировки результирующей таблицы в соответствии со значениями некоторых столбцов. Такой результат, вообще говоря, не отношение, а некоторый упорядоченный список кортежей.

3.2.3 Отсутствие упорядоченности атрибутов

Атрибуты отношений не упорядочены, поскольку по определению схема отношения есть множество пар {имя атрибута, имя домена}. Для ссылки на значение атрибута в кортеже отношения всегда используется имя атрибута. Это свойство теоретически позволяет, например, модифицировать схемы существующих отношений не только путем добавления новых атрибутов, но и путем удаления существующих атрибутов. Однако в большинстве

существующих систем такая возможность не допускается, и хотя упорядоченность набора атрибутов отношения явно не требуется, часто в качестве неявного порядка атрибутов используется их порядок в линейной форме определения схемы отношения.

3.2.4 Атомарность значений атрибутов

Значения всех атрибутов являются атомарными. Это следует из определения домена как потенциального множества значений простого типа данных, т.е. среди значений домена не могут содержаться множества значений (отношения). Принято говорить, что в реляционных базах данных допускаются только нормализованные отношения или отношения, представленные в первой нормальной форме. Потенциальным примером ненормализованного отношения является следующее:

НОМЕР_ОТДЕЛА	ОТДЕЛ		
	СОТР_НОМЕР	СОТР_ИМЯ	СОТР_ЗАРП
310	2934	Иванов	112,000
	2935	Петров	112,500
313	2937	Федоров	110,000
315	2938	Иванова	112,000

Можно сказать, что здесь мы имеем бинарное отношение, значениями атрибута ОТДЕЛЫ которого являются отношения. Представим нормализованный вариант исходного отношения:

СОТР_НОМЕР	СОТР_ИМЯ	СОТР_ЗАРП	НОМЕР_ОТДЕЛА
2934	Иванов	112,000	310
2935	Петров	144,000	310
2936	Сидоров	92,000	313
2937	Федоров	110,000	310
2938	Иванова	112,000	315

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. Рассмотрим, например, два идентичных оператора занесения кортежа:

- Зачислить сотрудника Кузнецова (пропуск номер 3000, зарплата 115,000) в отдел номер 320 и
- Зачислить сотрудника Кузнецова (пропуск номер 3000, зарплата 115,000) в отдел номер 310.

Если информация о сотрудниках представлена в виде нормализованного отношения, оба оператора будут выполняться одинаково (вставить кортеж в отношение). Если же работать с ненормализованным отношением, то первый оператор выразится в занесение кортежа, а второй – в добавление информации о Кузнецове в множественное значение атрибута ОТДЕЛ кортежа с первичным ключом 310.

3.3 Компоненты реляционной модели

Когда мы говорили об основных понятиях реляционных баз данных, мы не опирались на какую-либо конкретную реализацию. Эти рассуждения в равной степени относились к любой системе, при построении которой использовался реляционный подход. Другими словами, мы использовали понятия так называемой *реляционной модели данных*. Модель данных описывает некоторый набор родовых понятий и признаков, которыми должны обладать все конкретные СУБД и управляемые ими базы данных, если они основываются на этой модели. Наличие модели данных позволяет сравнивать конкретные реализации, используя один общий язык.

Хотя понятие модели данных является общим, и можно говорить о иерархической, сетевой, некоторой семантической и т.д. моделях данных, нужно отметить, что это понятие было введено в обиход применительно к реляционным системам и наиболее эффективно используется именно в этом контексте.

Наиболее распространенная трактовка реляционной модели данных принадлежит Дейту, который воспроизводит ее (с различными уточнениями) практически во всех своих книгах. Согласно Дейту, реляционная модель состоит из трех частей, описывающих разные аспекты реляционного подхода: *структурной части, манипуляционной части и целостной части*.

В *структурной части* модели фиксируется, что единственной структурой данных, используемой в реляционных БД, является нормализованное *n*-арное отношение. По сути дела, в предыдущих двух разделах этой лекции мы рассматривали именно понятия и свойства структурной составляющей реляционной модели.

В *манипуляционной части* модели утверждаются два фундаментальных механизма манипулирования реляционными БД – реляционная алгебра и реляционное исчисление. Первый механизм базируется в основном на классической теории множеств (с некоторыми уточнениями), а второй – на классическом логическом аппарате исчисления предикатов первого порядка. 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. Первое требование называется *требованием целостности сущностей*. Объекту или сущности реального мира в реляционных БД соответствуют кортежи отношений. Конкретно требование состоит в том, что любой кортеж любого отношения отличим от любого другого кортежа этого отношения, т.е., другими словами, любое отношение должно обладать первичным ключом. Как мы видели в предыдущем разделе, это требование автоматически удовлетворяется, если в системе не нарушаются базовые свойства отношений.

Второе требование называется *требованием целостности по ссылкам* и является несколько более сложным. Очевидно, что при соблюдении нормализованности отношений сложные сущности реального мира представляются в реляционной БД в виде нескольких кортежей нескольких отношений. Например, нам требуется представить в реляционной базе данных сущность ОТДЕЛ с атрибутами ОТД_НОМЕР (номер отдела), ОТД_КОЛ (количество сотрудников) и ОТД_СОТР (набор сотрудников отдела). Для каждого сотрудника нужно хранить СОТР_НОМЕР (номер сотрудника), СОТР_ИМЯ (имя сотрудника) и СОТР_ЗАРП (заработная плата сотрудника). При правильном проектировании соответствующей БД в ней появятся два отношения ОТДЕЛЫ и СОТРУДНИКИ со схемами:

$S_{ОТДЕЛЫ}$ = (<u>ОТД_НОМЕР</u>, ОТД_КОЛ),
где первичный ключ – ОТД_НОМЕР и
$S_{СОТРУДНИКИ}$ = (<u>СОТР_НОМЕР</u>, СОТР_ИМЯ, СОТР_ЗАРП,
НОМЕР_ОТДЕЛА),
где первичный ключ – СОТР_НОМЕР.

Как видно, атрибут НОМЕР_ОТДЕЛА появляется в отношении СОТРУДНИКИ не потому, что номер отдела является собственным свойством сотрудника, а лишь для того, чтобы иметь возможность восстановить при необходимости полную сущность ОТДЕЛ. Значение атрибута НОМЕР_ОТДЕЛА в любом кортеже отношения СОТРУДНИКИ должно соответствовать значению атрибута ОТД_НОМ в некотором кортеже отношения ОТДЕЛЫ. Атрибут такого рода называется *внешним ключом*, поскольку его значения однозначно характеризуют сущности, представленные кортежами некоторого другого отношения (т.е. задают значения их первичного ключа). Говорят, что отношение, в котором определен внешний ключ, ссылается на соответствующее отношение, в котором такой же атрибут является первичным ключом.

Требование целостности по ссылкам, или требование внешнего ключа состоит в том, что для каждого значения внешнего ключа, появляющегося в ссылающемся отношении, в отношении, на которое ведет ссылка, должен

найтись кортеж с таким же значением первичного ключа, либо значение внешнего ключа должно быть неопределенным (т.е. ни на что не указывать). Для нашего примера это означает, что если для сотрудника указан номер отдела, то этот отдел должен существовать.

Ограничения целостности сущности и по ссылкам должны поддерживаться СУБД. Для соблюдения целостности сущности достаточно гарантировать отсутствие в любом отношении кортежей с одним и тем же значением первичного ключа. С целостностью по ссылкам дела обстоят несколько более сложно.

Понятно, что при обновлении ссылающегося отношения (вставке новых кортежей или модификации значения внешнего ключа в существующих кортежах) достаточно следить за тем, чтобы не появлялись некорректные значения внешнего ключа. Но как быть при удалении кортежа из отношения, на которое ведет ссылка?

Здесь используются три стратегии поддержки целостности по ссылкам, определенные для соответствующих сущностей инфологической модели БД (п. 2.5): каскадирование, ограничение, установка.

В развитых реляционных СУБД можно выбрать способ поддержания целостности по ссылкам для каждого внешнего ключа. Конечно, для принятия такого решения необходимо анализировать требования конкретной прикладной области.

3.4 Реляционная алгебра

Основная идея реляционной алгебры состоит в том, что коль скоро отношения являются множествами, то средства манипулирования отношениями могут базироваться на традиционных теоретико-множественных операциях, дополненных некоторыми специальными операциями, специфичными для баз данных.

Существует много подходов к определению реляционной алгебры, которые различаются набором операций и способами их интерпретации, но в принципе более или менее равносильны. Мы опишем немного расширенный начальный вариант алгебры, который был предложен Коддом. В этом варианте набор основных алгебраических операций состоит из восьми операций, которые делятся на два класса – *теоретико-множественные операции* и *специальные реляционные операции*.

В состав теоретико-множественных операций входят операции:

1) объединения отношений;
2) пересечения отношений;
3) взятия разности отношений;
4) Декартова (прямого) произведения отношений.

Специальные реляционные операции включают:

1) ограничение (выборка, фильтрация) отношения;
2) проекцию отношения;
3) условное соединение отношений;
4) деление отношений.

Кроме того, в состав алгебры включается операция присваивания, позволяющая сохранить в базе данных результаты вычисления алгебраических выражений, и операция переименования атрибутов, дающая возможность корректно сформировать заголовок (схему) результирующего отношения.

3.4.1 Реляционные операции

Для описания результатов операций реляционной алгебры будем использовать нотацию теории множеств. Следующая запись означает, что множество R состоит из элементов r:

$$R = \{r \mid \text{условие}\} ,$$

где «условие» определяет принадлежность элемента r к множеству R.

Мы будем рассматривать множества-отношения, состоящие из элементов-кортежей. Условие – это логическое выражение, которое может включать логические операции $\wedge$ -»И» и $\vee$-»ИЛИ».

При выполнении операции *объединения* отношений R и Q производится отношение, включающее все кортежи r, входящие хотя бы в одно из отношений-операндов:

$$R \cup Q = \{r \mid r \in R \vee r \in Q\} .$$

Операция *пересечения* отношений R и Q производит отношение, включающее все кортежи r, входящие в оба отношения-операнда:

$$R \cap Q = \{r \mid r \in R \wedge r \in Q\} .$$

Отношение, являющееся *разностью* отношений R и Q, включает все кортежи r, входящие в R, и не входит в Q:

$$R \setminus Q = \{r \mid r \in R \wedge r \notin Q\} .$$

Очевидно, что $R \setminus Q \neq Q \setminus R$. Заметим, также, что если $R \setminus Q = \varnothing$, то $R \subseteq Q$. Поэтому разность можно использовать для проверки: является ли R подмножеством Q.

Следует отметить, что операции объединения, пересечения и разности определены *только для эквивалентных отношений*. В противном случае найти пересечение либо разность отношений невозможно, а результат объединения не будет являться отношением.

При выполнении *Декартова произведения* отношений $R = \{r\}$ и $Q = \{q\}$ производится отношение, кортежи которого являются конкатенацией (сцеплением) кортежей первого и второго операндов (r,q):

$$R \oplus Q = \{(r, q) \mid r \in R \wedge q \in Q\} .$$

Результатом *ограничения (выборки или фильтрации)* отношения R по некоторому условию α(r) является отношение, включающее кортежи r, удовлетворяющие этому условию:

$$R[\alpha(r)] = \{r \mid r \in R \wedge \alpha(r) = True\} .$$

Пусть S – множество атрибутов отношения R, и множество A является подмножеством S: $A \subseteq S$. При выполнении *проекции* отношения на множество его атрибутов A производится отношение со схемой, состоящей из множества атрибутов A, кортежи которого производятся путем удаления из них значений, не принадлежащих атрибутам из множества A:

$$R[A] = \{r[A]\} .$$

При *условном соединении* отношений $R = \{r\}$ и $Q = \{q\}$ по некоторому условию α(r,q) производится отношение, кортежи которого являются конкатенацией кортежей (r,q) и удовлетворяют этому условию:

$$R[\alpha(r,q)]Q = \{(r,q) \mid r \in R \wedge q \in Q \wedge \alpha(r,q) = True\} .$$

Очевидно, что условное соединение сводится к последовательному применению Декартова произведения и выборки (фильтрации).

Пусть A_R и A_Q – множества атрибутов отношений R и Q, соответственно. Предположим, что

- $A \subseteq A_R$, $B \subseteq A_Q$;
- $A^1 \subseteq A_R$, $A^1 \cup A = A_R$, $A^1 \cap A = \varnothing$, т.е. A^1 состоит из всех атрибутов из A_R, не вошедших в A.

Множеством образов кортежа x проекции R[A] называют множество таких кортежей y проекции $R[A^1]$, для которых сцепление (x,y) является кортежем отношения R:

$$Im[A(x)] = \{y \mid y \in R[A^1] \wedge (x,y) \in R\}.$$

Например: Im[НОМЕР_ОТДЕЛА(2)] для отношения СОТРУДНИКИ – это отношение со схемой (СОТР_НОМЕР, СОТР_ИМЯ, СОТР_ЗАРП), кортежи которого содержат сведения о сотрудниках отдела № 2.

Предположим, что схемы проекций R[A] и Q[B] эквивалентны: $S_{R[A]} \sim S_{Q[B]}$. Операция *деления* R на Q по наборам атрибутов A и B производит отношение, состоящее из тех кортежей проекции $R[A^1]$, для которых проекция Q[B] является подмножеством множества образов кортежа:

$$R[A:B]Q = \{x \mid x \in R[A^1] \wedge Q[B] \subseteq Im[A^1(x)]\}.$$

Операция деления является наиболее сложной, и ее выполнение сводится к более простым операциям. Возможна следующая схема выполнения деления. Во-первых, находятся проекции $R[A^1]$ и $Q[B]$. Затем, для каждого картежа x из $R[A^1]$ строится множество образов $Im[A^1(x)]$ путем выборки из R по условию совпадения значений атрибутов и проекции $R[A]$. Далее, из $R[A^1]$ производится выборка по условию $Q[B] \subseteq Im[A^1(x)]$. Для проверки этого условия выполняется нахождение разности, т.к. условие эквивалентно следующему: $Q[B] \setminus Im[A^1(x)] = \varnothing$.

Предположим, что в базе данных сотрудников поддерживаются два отношения: СОТРУДНИКИ(ИМЯ, ОТД_НОМЕР) и ИМЕНА(ИМЯ), причем унарное отношение ИМЕНА содержит все фамилии, которыми обладают сотрудники организации. Тогда после выполнения операции реляционного деления отношения СОТРУДНИКИ на отношение ИМЕНА будет получено унарное отношение, содержащее номера отделов, сотрудники которых обладают всеми возможными в этой организации именами.

Операция *переименования* производит отношение, тело которого совпадает с телом операнда, но имена атрибутов изменены.

Операция *присваивания* позволяет сохранить результат вычисления реляционного выражения в существующем отношении БД.

Поскольку результатом любой реляционной операции (кроме операции присваивания) является некоторое отношение, можно образовывать *реляционные выражения*, в которых вместо отношения-операнда некоторой реляционной операции находится вложенное реляционное выражение.

3.4.2 Замкнутость реляционной алгебры

Каждое отношение характеризуется схемой (или заголовком) и набором кортежей (или телом). Поэтому, если действительно желать иметь алгебру, операции которой замкнуты относительно понятия отношения, то каждая операция должна производить отношение в полном смысле, т.е. оно должно обладать и телом, и заголовком. Только в этом случае будет действительно возможно строить вложенные выражения.

Заголовок отношения представляет собой множество пар (имя-атрибута, имя-домена). Если посмотреть на определения реляционных операций, то видно, что домены атрибутов результирующего отношения однозначно определяются доменами отношений-операндов. Однако с именами атрибутов результата не всегда все так просто.

Например, представим себе, что у отношений-операндов операции Декартова произведения имеются одноименные атрибуты с одинаковыми доменами. Каким был бы заголовок результирующего отношения? Поскольку это множество, в нем не должны содержаться одинаковые элементы. Но и

потерять атрибут в результате недопустимо. А это значит, что в этом случае вообще невозможно корректно выполнить операцию Декартова произведения.

Аналогичные проблемы могут возникать и в случаях других двуместных операций. Для их разрешения в состав операций реляционной алгебры вводится операция переименования. Ее следует применять в любом случае, когда возникает конфликт именования атрибутов в отношениях – операндах одной реляционной операции. Тогда к одному из операндов сначала применяется операция переименования, а затем основная операция выполняется уже безо всяких проблем.

В дальнейшем изложении мы будем предполагать применение операции переименования во всех конфликтных случаях.

3.4.3 Особенности теоретико-множественных операций

Хотя в основе теоретико-множественной части реляционной алгебры лежит классическая теория множеств, соответствующие операции реляционной алгебры обладают некоторыми особенностями.

Начнем с операции объединения (все, что будет говориться по поводу объединения, переносится на операции пересечения и взятия разности). Смысл операции объединения в реляционной алгебре в целом остается теоретико-множественным. Но если в теории множеств операция объединения осмысленна для любых двух множеств-операндов, то в случае реляционной алгебры результатом операции объединения должно являться отношение. Если допустить в реляционной алгебре возможность теоретико-множественного объединения произвольных двух отношений (с разными схемами), то, конечно, результатом операции будет множество, но множество разнотипных кортежей, т.е. не отношение. Если исходить из требования замкнутости реляционной алгебры относительно понятия отношения, то такая операция объединения является бессмысленной.

Все эти соображения приводят к появлению понятия *совместимости отношений по объединению*: два отношения совместимы по объединению в том и только в том случае, когда обладают одинаковыми заголовками. Более точно это означает, что в заголовках обоих отношений содержится один и тот же набор имен атрибутов, и одноименные атрибуты определены на одном и том же домене.

Если два отношения совместимы по объединению, то при обычном выполнении над ними операций объединения, пересечения и взятия разности результатом операции является отношение с корректно определенным заголовком, совпадающим с заголовком каждого из отношений-операндов. Напомним, что если два отношения «почти» совместимы по объединению, т.е. совместимы во всем, кроме имен атрибутов, то до выполнения операции типа соединения эти отношения можно сделать полностью совместимыми по объединению путем применения операции переименования.

Заметим, что включение в состав операций реляционной алгебры трех операций объединения, пересечения и взятия разности является очевидно избыточным, поскольку известно, что любая из этих операций выражается

через две других. Тем не менее, Кодд решил включить все три операции, исходя из интуитивных потребностей потенциального пользователя системы реляционных БД, далекого от математики.

Другие проблемы связаны с операцией взятия Декартова произведения двух отношений. В теории множеств Декартово произведение может быть получено для любых двух множеств, и элементами результирующего множества являются пары, составленные из элементов первого и второго множеств. Поскольку отношения являются множествами, то и для любых двух отношений возможно получение прямого произведения. Но результат не будет отношением! Элементами результата будут являться не кортежи, а пары кортежей.

Поэтому в реляционной алгебре используется специальная форма операции взятия Декартова произведения – *расширенное Декартово произведение* отношений. При взятии расширенного Декартова произведения двух отношений элементом результирующего отношения является кортеж, представляющий собой *конкатенацию (или слияние)* одного кортежа первого отношения и одного кортежа второго отношения.

Но теперь возникает второй вопрос – как получить корректно сформированный заголовок отношения-результата? Очевидно, что проблемой может быть именование атрибутов результирующего отношения, если отношения-операнды обладают одноименными атрибутами.

Эти соображения приводят к появлению понятия *совместимости по взятию расширенного Декартова произведения*. Два отношения совместимы по взятию прямого произведения в том и только в том случае, если множества имен атрибутов этих отношений не пересекаются. Любые два отношения могут быть сделаны совместимыми по взятию прямого произведения путем применения операции переименования к одному из этих отношений.

Следует заметить, что операция взятия прямого произведения не является слишком осмысленной на практике. Во-первых, мощность ее результата очень велика даже при допустимых мощностях операндов, а во-вторых, результат операции не более информативен, чем взятые в совокупности операнды. Как мы увидим немного ниже, основной смысл включения операции расширенного прямого произведения в состав реляционной алгебры состоит в том, что на ее основе определяется действительно полезная операция соединения.

По поводу теоретико-множественных операций реляционной алгебры следует еще заметить, что все четыре операции являются *ассоциативными*. То есть, если обозначить через OP любую из четырех операций, то (A OP B) OP C = A (B OP C), и следовательно, без введения двусмысленности можно писать A OP B OP C (A, B и C – отношения, обладающие свойствами, требуемыми для корректного выполнения соответствующей операции). Все операции, кроме взятия разности, являются коммутативными, т.е. A OP B = B OP A.

Несмотря на то, что теоретико-множественные операции довольно просты, их комбинации позволяют решать достаточно сложные задачи.

Примеры.

Пусть для учета абитуриентов во время сдачи вступительных экзаменов используется база данных, содержащая три отношения с эквивалентными схемами:

R1(ФИО, Паспорт, Школа) – список абитуриентов, сдававших репетиционный экзамен;

R2(ФИО, Паспорт, Школа) – список абитуриентов, сдававших вступительный экзамен;

R3(ФИО, Паспорт, Школа) – список абитуриентов, зачисленных в вуз.

Зачисление в вуз возможно как по результатам репетиционного, так и вступительного экзамена. Тогда:

R1∩R2\R3 – список абитуриентов, сдававших экзамены дважды, но не поступивших в вуз;

(R1\R2∩R3)∪(R2\R1∩R3) – список абитуриентов, поступивших с первой попытки;

R1∩R2∩R3 – список абитуриентов, поступивших со второй попытки.

3.4.4 Особенности специальных реляционных операций

Рассмотрим подробнее специальные реляционные операции реляционной алгебры: ограничение, проекция, соединение и деление.

Операция ограничения требует наличия двух операндов: ограничиваемого отношения и простого условия ограничения. Простое условие ограничения может иметь либо вид (a comp-op b), где a и b – имена атрибутов ограничиваемого отношения, для которых осмысленна операция сравнения comp-op, либо вид (a comp-op const), где a – имя атрибута ограничиваемого отношения, a const – литерально заданная константа.

В результате выполнения операции ограничения производится отношение, заголовок которого совпадает с заголовком отношения-операнда, а в тело входят те кортежи отношения-операнда, для которых значением условия ограничения является true.

Пусть UNION обозначает операцию объединения, INTERSECT – операцию пересечения, а MINUS – операцию взятия разности. Для обозначения операции ограничения будем использовать конструкцию A WHERE comp, где A – ограничиваемое отношение, а comp - простое условие сравнения. Пусть comp1 и comp2 – два простых условия ограничения. Тогда по определению:

A WHERE comp1 AND comp2 обозначает то же самое, что и (A WHERE comp1) INTERSECT (A WHERE comp2)

A WHERE comp1 OR comp2 обозначает то же самое, что и (A WHERE comp1) UNION (A WHERE comp2)

A WHERE NOT comp1 обозначает то же самое, что и A MINUS (A WHERE comp1)

С использованием этих определений можно использовать операции ограничения, в которых условием ограничения является произвольное булевское выражение, составленное из простых условий с использованием логических связок AND, OR, NOT и скобок.

На интуитивном уровне операцию ограничения лучше всего представлять как взятие некоторой «горизонтальной» вырезки из отношения-операнда.

Операция взятия проекции также требует наличия двух операндов – проецируемого отношения A и списка имен атрибутов, входящих в заголовок отношения A.

Результатом проекции отношения A по списку атрибутов a_1, a_2, ..., a_n является отношение, с заголовком, определяемым множеством атрибутов a_1, a_2, ..., a_n, и с телом, состоящим из кортежей вида $<a_1:v_1, a_2:v_2, ..., a_n:v_n>$ таких, что в отношении A имеется кортеж, атрибут a_1 которого имеет значение v_1, атрибут a_2 – значение v_2, ..., атрибут a_n – значение v_n. Тем самым, при выполнении операции проекции выделяется «вертикальная» вырезка отношения-операнда с естественным *уничтожением потенциально возникающих кортежей-дубликатов*.

Общая операция условного соединения (называемая также соединением по условию) требует наличия двух операндов – соединяемых отношений и третьего операнда – простого условия. Пусть соединяются отношения A и B. Как и в случае операции ограничения, условие соединения comp имеет вид либо (a comp-op b), либо (a comp-op const), где a и b – имена атрибутов отношений A и B, const – литерально заданная константа, а comp-op – допустимая в данном контексте операция сравнения.

Тогда по определению результатом операции условного соединения является отношение, получаемое путем выполнения операции ограничения по условию comp Декартова произведения отношений A и B.

Если внимательно осмыслить это определение, то станет ясно, что в общем случае применение условия соединения существенно уменьшит мощность результата промежуточного прямого произведения отношений-операндов только в том случае, когда условие соединения имеет вид (a comp-op b), где a и b – имена атрибутов разных отношений-операндов. Поэтому на практике обычно считают реальными операциями соединения именно те операции, которые основываются на условии соединения приведенного вида.

Хотя операция соединение в нашей интерпретации не является примитивной (поскольку она определяется с использованием прямого произведения и проекции), в силу особой практической важности она включается в базовый набор операций реляционной алгебры. Заметим также, что в практических реализациях соединение обычно не выполняется именно как ограничение прямого произведения. Имеются более эффективные алгоритмы, гарантирующие получение такого же результата.

Имеется важный частный случай соединения – эквисоединение и простое, но важное расширение операции эквисоединения – естественное соединение. Операция соединения называется *операцией эквисоединения*, если

условие соединения имеет вид (a = b), где a и b – атрибуты разных операндов соединения. Этот случай важен потому, что (a) он часто встречается на практике и (b) для него существуют эффективные алгоритмы реализации.

Операция *естественного соединения* применяется к паре отношений A и B, обладающих (возможно составным) общим атрибутом *c* (т.е. атрибутом c одним и тем же именем и определенным на одном и том же домене). Пусть ab обозначает объединение заголовков отношений A и B. Тогда естественное соединение A и B – это спроектированный на ab результат эквисоединения A и B. Если вспомнить введенное нами в конце предыдущей главы определение внешнего ключа отношения, то должно стать понятно, что основной смысл операции естественного соединения – возможность *восстановления сложной сущности*, декомпозированной по причине требования первой нормальной формы. Операция естественного соединения не включается прямо в состав набора операций реляционной алгебры, но она имеет очень важное практическое значение.