УДК 004.047:681.3.01

Л.А. Пономаренко, С.С. Танянский

Обработка данных произвольной структуры языковыми средствами реляционной модели

Рассмотрены вопросы организации доступа к данным со структурой, отличной от реляционной модели. Определены основные возможности использования выражений реляционного исчисления с переменными доменами для структур данных, основанных на графах. Исследована задача логического преобразования таблиц сетевой структуры к виду, допускающему использование операционной и языковой спецификации реляционной модели данных.

The problems of the organization of the access to the data with a structure different from a relational model are considered. The basic possibilities of the use of expressions of the relational calculation with variable domains for structures of the data based on graphs are defined. The problem of the logic transformation of tables of a network structure to a kind allowing the use of the operational and language specification of the relational model is described.

Розглянуто питання організації доступу до даних зі структурою, відмінною від реляційної моделі. Визначено основні можливості використання виражень реляційного числення зі змінними доменами для структур даних, заснованих на графах. Досліджено задачу логічного перетворення таблиць мережної структури до виду, що допускає використання операційної та мовної специфікації реляційної моделі даних.

Введение. Способы формулирования запросов, используемые в реляционной алгебре и реляционном исчислении, внешне выглядят по-разному. Существуют правила преобразования запросов из одного представления в другое, использующиеся, в частности, для оптимизации запросов по средствам приведения их к виду, подходящему для эффективной обработки. Такая концепция встречается в компиляторах, меняющих порядок подвыражений в операторах и устраняющих избыточные вычисления.

Важным доводом в пользу преобразования запросов является то, что сегодня широкое использование получают распределенные и интегрированные системы, основанные на базах данных (БД). Если предположить, что в двух информационных системах (ИС) используются БД для хранения одинаковых данных, но организованных по-разному, т.е. структуры БД неоднородны, то необходимо иметь возможность сформулировать запрос в форме, не привязанной к структурным особенностям данных. Такой подход также может быть использован в случаях, когда несколько отдельных БД функционируют в едином информационном пространстве с едиными принципами обработки данных.

Один из способов организации совместной обработки данных – построение интегрированной среды, объединяющей структурные особенности локальных БД. Для реализации интеграции можно использовать различные методы и средства. К ним относятся ряд аналитических и графических представлений, среди которых выделяются моделирование предметной области с помощью диаграмм «сущность–связь», семантических сетей, онтологий и др. Отметим, что интеграцию можно реализовать посредством объединения локальных систем или использовать средства, позволяющие организовывать доступ к данным в едином формате. В условиях, когда структуры локальных данных не эквивалентны и допускается значительная динамика их изменения, представление интегрированной среды посредством единой модели концептуализации данных (концептуализация рассматривается как множество правил, ограничивающих структуру действительности [1]) не является эффективной, так как требует пересмотра структуры обобщенной интегрированной системы.

Для организации совместной обработки без концептуального представления необходимо средство для единого формирования запросов к неоднородным структурам данных. В большинстве используемых ИС базы данных реализованы на основе реляционной модели и, следовательно, язык запросов использует операции реляционной алгебры и исчисления. Очевидно, что на формулировку запроса влияет структура БД, и, следовательно, необходимо иметь возможность описывать данные различной структуры (реляционной модели) в одном формате в терминах одинаковой операционной спецификации.

Исследования альтернативных средств формулирования запросов относятся ко времени разработанных математических методов описания данных. В частности, в запросах реляционной модели использование формул исчисления предикатов предложено Дж. Кунсом [2]. Практика использования современных технологий хранения данных постоянно расширяет границы информационного пространства, что в свою очередь требует расширения возможностей обработки данных из удаленных серверов средствами, допустимыми в локальных ИС.

Использование формул исчисления для формулировки запроса во многих случаях не требует знаний обо всех структурных деталях БД (таблицах, связях, ключах и т.п.), достаточно знать, какие данные и какие условия определяют конечный результат. Но, с другой стороны, необходимо учитывать, что ИС может обрабатывать только тот язык, который использует встроенная система управления базами данных (СУБД). Таким образом, запрос, записанный как формула исчисления предикатов, должен быть приведен к виду, понятному транслятору соответствующей СУБД (в данном случае – реляционной).

Обзор публикаций по методам обработки данных

Исследовать достоинства и недостатки совместной обработки данных, используя консолидированный подход, включая онтологии и семантические сети, можно по работам А.С. Клещева, И.Л. Артемьева [3] и Д. Цикритзиса, Ф. Лоховски [4].

Существующие работы по исследованию методов доступа к данным своими истоками уходят к началу развития компьютерных технологий. История развития языков манипулирования данными показала, что в технологиях БД наблюдается тенденция унификации средств формулирования запросов, что позволяет использовать гетерогенные системы в одном информационном пространстве.

Наиболее значимыми работами, определившими дальнейшее развитие технологий обработки данных, являются исследования Э. Кодда [5], Б. Стонебрекейра и Э. Вонга [6], а также работы Л. Готлиба [7], в которых изучены свойства модифицированных запросов и исследованы алгоритмы для реализации операций реляционной алгебры и исчисления.

Среди авторов, исследовавших вопросы доступа к данным с использованием современных методов и средств, можно выделить работы М. Зельцера [8] и В. Есина [9], в которых рассмотрены задачи создания единого информационного пространства на основе разработки универсальной БД со средствами обработки данных, основанных на едином методе описания и манипулирования данными.

Постановка задачи

Анализ указанных источников показал, что исследования в области унификации описания и обработки неоднородных распределенных данных недостаточно формализованы и классифицированы. Это связано в основном с попытками построить логическую «надстройку», используя методы и средства анализа предметных областей, целостное представление обо всех свойствах и ограничениях локальных ИС.

С другой стороны, в рассматриваемой задаче допускаются изменения в схемах локальных БД, и несогласованность структур данных может возникнуть в процессе функционирования ИС. При использовании методов единой концептуализации данных необходимо реструктуризировать представление предметной области при любом изменении структурных элементов локальных БД. Возможность корректно сформулировать запрос к конкретной БД в любой момент позволит уменьшить затраты на перепрограммирование средств доступа к данным при изменениях структур БД.

Таким образом, материал, рассматриваемый в статье, актуален как для гетерогенных ИС, использующих БД для накопления информации, так и для задач повышения эффективности управления распределенными данными с динамически изменяемой схемой.

Цель статьи – исследование и модификация методов и средств описания доступа к данным, основанная не на единой универсальной модели или структуре данных, а на едином подходе к формулированию запросов. Учитывая тот

факт, что средствами реляционных СУБД можно обрабатывать таблицы не реляционной структуры (например, со скрытой интерпретацией домена), для формального описания запросов более гибкий – аппарат реляционного исчисления, формулы которого в итоге легко трансформируется в *SQL*.

Табличное представление документов иерархической и сетевой структуры

Рассмотрим некоторые структурные особенности таблиц. Табличные документы можно разделить на три традиционных класса в соответствии с их структуризацией: реляционные, иерархические и сетевые.

Реляционные структуры соответствуют классической модели Кодда и не требуют дополнительных комментариев [5]. Документом иерархической структуры назовем таблицу *T*, представленную на рис. 1,*а*, соответствующее графическое изображение представлено на рис. 1,*б*.

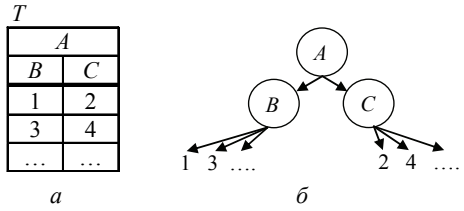

Рис. 1. Схема таблицы иерархической структуры

Представим таблицу *T* (рис. 1,*а*) средствами реляционной модели. Для этого необходимо задать набор доменов. В соответствии со схемой таблицы имеем три именованных домена (атрибута): $A = \{B, C\}$; $B = \{1, 3, \ldots\}$; $C = \{2, 4, \ldots\}$. Очевидно, что рассматривая задачу на теоретическом уровне, декартово произведение доменов $R = A \times B \times C$ даст результат, не соответствующий исходному документу. С другой стороны, на практическом уровне (некоторые ослабления теоретических требований на практике допускаются) описать данные также невозможно, так как сформировать схему документа, представленного на рис. 1, средствами реляционной СУБД нельзя, и на этапе создания она должна быть нормализована, например, приведена к виду таблицы *T* на рис. 2.

Для работы с табличным документом иерархической структуры необходимо выполнить преобразование на уровне описания доменов. Отметим, что в задаче не рассматривается иерархия на уровне данных, т.е. реляционные таблицы с рекурсивной организацией хранения информации [10].

T

A.B	*A.C*
1	2
3	4
…	…

Рис. 2. Нормализованная структура иерархической таблицы

Табличным документом сетевой структуры назовем структуру таблицы *T*, представленную на рис. 3,*а*. Факт сетевой организации такого документа графически демонстрируется схемой на рис. 3,*б*.

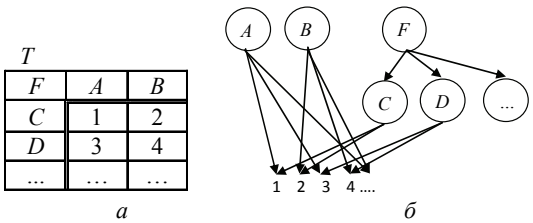

Рис. 3. Схема таблицы сетевой структуры

Данная структура характеризуется следующими особенностями: для идентификации данных, которыми являются строчные элементы <1, 2> <3, 4>, необходимы два атрибута, вертикальный и горизонтальный, т.е. теоретически схема таблицы определяется двумя наборами атрибутов; при этом элементы данных не имеют собственной интерпретации.

Для хранения такой структуры можно использовать реляционную таблицу с добавлением «фиктивного» атрибута *F*. Таким образом, с учетом физической структуры таблица *T* соответствует реляционной модели, а с точки зрения семантики данных – таблица не соответствует реляционному отношению.

Практический пример такой структуры можно представить в виде таблицы *Экзамен*, в которой хранится информация об успеваемости учеников (рис. 4).

Экзамен

ФИО	*МАТЕМАТИКА*	*ФИЗИКА*	…
САВЧЕНКО	4	5	…
ШЕВЧЕНКО	5	4	…
…	…	…	…

Рис. 4. Структура сетевой организации таблицы

О смысловой нагрузке цифр, стоящих на пересечении строк и столбцов, можно догадаться только из названия таблицы. Один из способов корректной работы с такой таблицей – воспользоваться методом нормализации и привести ее в соответствие реляционной модели. Такой подход легко осуществим при проектировании БД до накопления данных. При условии, что БД успешно функционирует некоторое время, вопрос реорганизации структуры затрудняется при переносе информации в новые таблицы, так как гарантировать отсутствие потери данных при проекции нельзя из-за свойств операций реляционной алгебры [11].

Наилучший способ организации доступа к таблицам нереляционной структуры – найти возможность формирования запросов средствами реляционной СУБД. При этом необходимо гарантировать правильность результата за счет использования формальных языков реляционной модели.

Описание данных сетевой структуры языковыми средствами реляционного исчисления

Хотя реляционная алгебра – основа многих языков запросов, большинство из них строится также и на исчислении. Как известно, выделяются два вида реляционного исчисления: с переменными кортежами и с переменными доменами [12].

Выражение исчисления с переменными кортежами обозначим как $\{t(R) | f(t)\}$, где f – некоторый предикат над кортежем t. Такая запись определяет отношение $r(R)$, состоящее из всех кортежей $t(R)$, для которых предикат $f(t)$ является истинным.

Для определения выражения исчисления с переменными доменами заменим унарную переменную t несколькими переменными, количество которых зависит от арности отношения $r(R)$. При этом необходимо фиксировать порядок переменных относительно порядка атрибутов в отношении. Если $r(R)$ имеет арность n, необходимо ввести n переменных $t_1, t_2, \ldots, t_n$, и тогда выражение исчисления с переменными доменами имеет вид $\{t_1, t_2, ..., t_n \mid f'(t_1, t_2, ..., t_n)\}$, где f' – предикат, аналогичный предикату f, в котором любой атом $R(t)$ заменен атомом $R(t_1, t_2, \ldots, t_n)$, а каждое свободное вхождение $t[i]$ заменено переменной t_i.

Для построения рекурсивных формул в исчислении с переменными кортежами использование связок $\wedge$ (и), $\vee$ (или), $\neg$ (не), а также кванторов $\exists$ (существования) и $\forall$ (всеобщности) также можно привести к эквивалентному виду в исчислении с переменными доменами. Для каждого квантора в формуле исчисления с переменными кортежами $\exists (u)$ и $\forall (u)$ вводится k новых переменных на доменах $u_1, u_2, ..., u_k$. В области действия квантификации $u[i]$ заменяется на u_i, а $R(u)$ заменяется атомом $R(u_1, u_2, \ldots, u_k)$. Соответствующие замены связных переменных в формулах исчисления с переменными доменами имеют вид $\exists (u_1, u_2, ..., u_k)$ и $\forall (u_1, u_2, ..., u_k)$.

Пусть R и S – некоторые таблицы БД, имеющие арность n, тогда формула, реализующая вычитания $R - S$, имеет вид:

$$\left\{t^{(n)} \left| \exists(u)\left(R(u) \wedge t[1] = u[1] \wedge ... \wedge t[n] = u[n]\right) \wedge \neg\exists(v)\left(S(v) \wedge u[1] = v[1] \wedge ... \wedge u[n] = v[n]\right)\right.\right\}. \quad (1)$$

После соответствующих замен формула исчисления с переменными доменами примет вид:

$$\left\{t_1, t_2, ..., t_n \left| \exists(u_1, u_2, ..., u_n)\left(R(u_1, u_2, ..., u_n) \wedge t_1 = u_2 \wedge ... \wedge t_n = u_n\right) \wedge \neg\exists(v_1, v_2, ..., v_n) \cdot \left(S(v_1, v_2, ..., v_n) \wedge u_1 = v_1 \wedge ... \wedge u_n = v_n\right)\right.\right\}. \quad (2)$$

Рассмотрим примеры организации таблиц различной структуры, в частности, реляционной и сетевой.

Пусть таблица R соответствует реляционной модели, а T – сетевой модели, т.е. для T заданы множество доменов (допустимых значений) $D = \{D_1, ..., D_m\}$ и множество имен (интерпретаций) $A = \{A_1, ..., A_m\}$, для которых установлено отображение вида $\rho : A \to D$. При этом отношение реляционной модели (таблица) имеет такой вид:

$$R \subseteq D_1 \times D_2 \times ... \times D_m. \quad (3)$$

Результатом определения подмножеств декартова произведения (3) является множество кортежей $r = \langle d_1, d_2, \ldots, d_m \rangle$, где $d_i \in D_i$ – экземпляр отношения, и каждый кортеж r_j отражает единицу информации предметной области. Задав пару (A_i, D_i), называемую атрибутом в обозначении Atr_i, построим таблицу, строки которой соответствуют кортежам из R, а схема таблицы соответствует множеству атрибутов

$$S = \bigcup_{i=1}^{m} Atr_1 . \qquad (4)$$

Таким образом, таблицу реляционной модели обозначам выражением

$$R = S(r). \qquad (5)$$

Сетевые свойства таблицы T представим в виде графа $G(V, E, \mu)$, где $V = \{v_1, \ldots, v_k\}$ – множество вершин, соответствующих элементам данных, и $E = \{e_1, \ldots, e_p\}$ – множество ребер, соответствующих связям или отношениям между элементами данных. Для каждого ребра определим имя. Множество имен обозначим как $N = \{n_1, \ldots, n_p\}$. Соответствие имен и ребер определяется биективным отображением вида $\mu : N \to E$. С учетом табличного представления имя ребра представляет собой значение элемента данных таблицы.

Граф $G_1(V_1, E_1, \mu)$ назовем подграфом (фрагментом) графа G, $G_1 \subseteq G$, если его множества V_1 и E_1 являются подмножествами соответствующих множеств V и E. При этом отображение μ должно сохранять свойства биекции. Таким образом, G_1 – это подграф, где E_1 состоит из всех ребер графа G, у которых оба конца принадлежат V_1, и из всех вершин, инцидентных ребрам из E_1.

Рассмотрим множество $\overline{G} = \{\overline{G}_1, \ldots, \overline{G}_r\}$ как подмножество графа G, где каждый $\overline{G}_i \subseteq G$ является порожденным подграфом и соответствует пути, определяющему единицу информации из БД, т.е. набор данных, соответствующий некоторому запросу. Таким образом, модель таблицы сетевой структуры можно представить как выражение следующего вида:

$$T = \bigcup_{i=1}^{r} \overline{G}_i . \qquad (6)$$

Формализация средств доступа к данным

В предложенной нотации к таблицам сетевой структуры могут быть сформулированы запросы в виде выражений исчисления с переменными доменами, где в качестве домена выступает элемент данных таблицы.

Пусть задана таблица сетевой структуры $T(G(V, E, \mu))$, где $V = \{A, B, C, D\}$, $E = \{e_1, e_2, e_3, e_4\}$ и $e_1 = (A, C)$, $e_2 = (B, C)$, $e_3 = (A, D)$, $e_4 = (B, D)$, $N = \{n_1, n_2, n_3, n_4\}$ и $n_1 = 1$, $n_2 = 2$, $n_3 = 3$, $n_4 = 4$. При этом соответствие между именами и ребрами, устанавливаемое множеством отображений $\mu = \{\mu_1, \mu_2, \mu_3, \mu_4\}$, определяется значениями $\mu_1 = 1$, $\mu_2 = 2$, $\mu_3 = 3$, $\mu_4 = 4$. Необходимо найти все значения элемента данных A, другими словами, необходимо найти значения $\{\mu_i\}$, соответствующие подграфу $\overline{G_1}\left(\overline{V_1}, \overline{E_1}, \mu\right)$, где $\overline{V_1} = \{A, C, D\}$, $\overline{E_1} = \{e_1, e_3\}$ и $e_1 = (A, C)$, $e_3 = (A, D)$. Тогда выражение в исчислении с переменными доменами будет иметь вид:

$$\left\{t_1, t_3, t_4 \mid \exists(u_1)\left(T(u_1) \wedge t_1 = u_1 \wedge \right.\right.$$
$$\wedge\left(\exists(u_3)\left(T(u_3) \wedge u_1 = u_3 \wedge t_3 = u_3\right)\right) \vee$$
$$\left.\left. \vee\left(\exists(u_4)\left(T(u_4) \wedge u_1 = u_4 \wedge t_4 = u_4\right)\right)\right)\right\}. \qquad (7)$$

Выражение (7) показывает, что в результирующую таблицу будет помещен столбец с именем элемента данных A, которому в (7) соответствует переменная u_1, и связные с ним элементы данных C и D, которым соответствуют переменные u_3 и u_4. Поскольку два ребра подграфа выходят из одной вершины, то необходимо найти объединение двух элементов данных C и D, что соответствует операции «$\vee$». Таким образом, результирующими значениями будут $\mu_1 = 1$, $\mu_3 = 3$.

При использовании формулы исчисления для формирования запроса к таблице сетевой структуры необходимо отметить, что для получения корректного результата необходимо обязательно указывать два элемента данных или в терминах реляционной модели – два домена. Один домен является элементом для формирования схемы таблицы, другой – для формирования условия выбора значений, при этом являясь параметром выбора значения.

Для обеспечения возможности доступа к данным не реляционной структуры рассмотрим два случая преобразования таблицы.

Первый случай – логическое преобразование. Пусть $T(G(V, E, \mu))$ – таблица сетевой структуры и пусть $G = \bigcup_{i=1}^{r} \overline{G_i}$ – совокупность подграфов, определяющих информационные элементы предметной области. И пусть, исходя из рассмотренного примера, необходимо найти все значения элемента данных B, т.е. найти значения $\{\mu_i\}$, соответствующие подграфу $\overline{G_2}\left(\overline{V_2}, \overline{E_2}, \mu\right)$, где $\overline{V_2} = \{B, C, D\}$, $\overline{E_2} = \{e_2, e_4\}$ и $e_3 = (B, C)$, $e_4 = (B, D)$. Так как количество строк таблицы в данном примере фиксировано, то рассматриваются две строки (хотя всегда можно говорить о выборке нужного количества элементов данных). Тогда выражение запроса примет следующий вид:

$$\left\{t_2, t_3, t_4 \mid \exists(u_2)\left(T(u_2) \wedge t_2 = u_2 \wedge \right.\right.$$
$$\wedge\left(\exists(u_3)\left(T(u_3) \wedge u_1 = u_3 \wedge t_3 = u_3\right)\right) \vee$$
$$\left.\left. \vee\left(\exists(u_4)\left(T(u_4) \wedge u_1 = u_4 \wedge t_4 = u_4\right)\right)\right)\right\}. \quad (8)$$

По аналогии с предыдущим примером, выражение (8) показывает, что в результирующую таблицу будет помещен столбец с именем элемента данных B и связные с ним элементы данных C и D, при этом результатом будут значения $\mu_1 = 2$, $\mu_3 = 4$.

Запрос для получения всех данных из таблицы T соответствует операции объединения подграфов $\overline{G_1} \cup \overline{G_2}$, для которой выражение исчисления можно представить в следующей записи:

$$\left\{t_1, t_2, t_3, t_4 \mid \exists(u_1, u_2)\left(T(u_1, u_2) \wedge t_1 = u_1 \wedge t_2 = \right.\right.$$
$$= u_2) \wedge \left(\exists(u_3)\left(T(u_3) \wedge u_1 = u_3 \wedge t_3 = u_3\right)\right) \vee$$
$$\vee\left(\exists(u_4)\left(T(u_4) \wedge u_1 = u_4 \wedge t_4 = u_4\right)\right) \vee$$
$$\vee\left(\exists(u_3)\left(T(u_3) \wedge u_1 = u_3 \wedge t_3 = u_3\right)\right) \vee$$
$$\left.\left. \vee\left(\exists(u_4)\left(T(u_4) \wedge u_1 = u_4 \wedge t_4 = u_4\right)\right)\right\}. \quad (9)$$

Очевидно, что выражение (9) эквивалентно следующему выражению:

$$\left\{t_1, t_2, t_3, t_4 \mid \exists(u_1, u_2)\left(T(u_1, u_2) \wedge t_1 = u_1 \wedge \right.\right.$$
$$\wedge t_2 = u_2) \wedge \left(\exists(u_3)\left(T(u_3) \wedge u_1 = u_3 \wedge t_3 = u_3\right)\right) \vee$$
$$\left.\left. \vee\left(\exists(u_4)\left(T(u_4) \wedge u_1 = u_4 \wedge t_4 = u_4\right)\right)\right\}. \quad (10)$$

Однако, в реляционных языках запросов прямой эквивалентной формулировки не существует. Но, с другой стороны, рассматриваемые примеры реализуются средствами реляционных СУБД, что позволяет установить некоторые соответствия между выражением исчисления и языковыми конструкциями *SQL*.

Пусть схема таблицы T соответствует схеме, изображенной на рис. 5.

T

	A	*B*
C	1	2
D	3	4

Рис. 5. Схема таблицы T сетевой структуры

Для реализации запроса по выражению (10) установим следующие соответствия: атрибуту A соответствует свободная переменная t_1, атрибуту B – t_2, атрибуту C – t_3 и атрибуту D – t_4, а также определим фиктивную переменную t_0, которая задает атрибут, содержащий в качестве своих элементов C и D. Тогда конструкция запроса на *SQL* может иметь вид:

```
SELECT A, B
FROM T AS a
WHERE EXISTS (SELECT * FROM T AS b WHERE
a.t0 = b.t0 AND a.t0 = C) OR
      EXISTS (SELECT * FROM T AS c WHERE
a.t0 = c.t0 AND a.t0 = D)
```

Как уже отмечалось, иногда возможно корректно (с учетом сохранения данных) нормализовать структуру исходной таблицы. Воспользуемся примером, рассмотренным в первом случае. Преобразуем с помощью перекрестного запроса таблицу T к двум таблицам T_1 и T_2, где каждый элемент данных – аналог домена реляционной модели (рис. 6).

T_1

A	*B*
1	2
3	4

T_2

C	*D*
1	3
2	4

Рис. 6. Декомпозиция таблицы сетевой структуры

Тогда связность графа $G(V, E, \mu)$ определяется операцией поэлементного объединения. Причем объединение выполняется по принципу сравнения строк таблицы T_1 со столбцами таблицы T_2.

Используя ранее введенные обозначения, выражение запроса для получения всех данных (т.е. по двум элементам данных из T_1 и T_2) имеет следующий вид:

$$\{t_1, t_2, t_3, t_4 \mid \exists(u_1, u_2)(T_1(u_1, u_2) \wedge t_1 = u_1 \wedge t_2 = u_2) \wedge (\exists(u_3)(T_2(u_3) \wedge (u_1 = u_3 \vee u_2 = u_3) \wedge t_3 = u_3)) \vee (\exists(u_4) \cdot (T_2(u_4) \wedge (u_1 = u_4 \vee u_2 = u_4) \wedge t_4 = u_4))\}. \quad (11)$$

Исходя из свойства дизъюнкции, условия ($u_1 = u_3 \vee u_2 = u_3$) и ($u_1 = u_4 \vee u_2 = u_4$) можно заменить одним любым дизъюнктом, входящим в выражения. Очевидно, что результат при этом не изменится, а формула сократится и примет такой вид:

$$\{t_1, t_2, t_3, t_4 \mid \exists(u_1, u_2)(T_1(u_1, u_2) \wedge t_1 = u_1 \wedge t_2 = u_2) \wedge (\exists(u_3)(T_2(u_3) \wedge u_1 = u_3 \wedge t_3 = u_3)) \vee (\exists(u_4)(T_2(u_4) \wedge u_1 = u_4 \wedge t_4 = u_4))\}. \quad (12)$$

Соответствующий запрос, реализованный на *SQL*, имеет вид:

SELECT A, B
FROM T_1 *AS a*
WHERE EXISTS (*SELECT * FROM* T_2 *AS b WHERE a.A = b.C*) *OR*
EXISTS (*SELECT * FROM* T_2 *AS c WHERE a.B = c.D*)

Анализируя выражение (12) и соответствующий запрос на *SQL*, необходимо отметить, что условия $u_i = u_j$ и $t_j = u_j$ в программной реализации сводятся к условию *a.A = b.C* и *a.B = c.D*, которое выполняет функцию соединения и выборки данных.

Таким образом, рассмотрев два варианта организации работы с таблицами сетевой структуры, приходим к выводу, что после соответствующих сокращений выражения (10) и (12 – эквивалентны, хотя в нотации *SQL* формат запроса отличается за счет источника данных (одна таблица T и две таблицы T_1 и T_2 соответственно). Этот факт дает возможность использовать языковые средства реляционной модели для обработки БД с неоднородной (не реляционной) структурой без приведения ее к реляционному соответствию.

Заключение. При использовании различных источников информации одни и те же сущности реального мира могут моделироваться с помощью различных структур и типов данных. При попытке интеграции разнородных источников необходимо либо приводить данные к общей структуре, либо в результате интеграции данные будут иметь неправильную структуру. Приведение к общей структуре при большом количестве источников может стать невозможным или обобщенная структура будет очень сложной, а ее использование неэффективным. Таким образом, один из способов совместной обработки множества независимых БД, спроектированных с различной степенью детализации, – работа с данными, имеющими несогласованную структуру.

Рассмотренные в статье вопросы организации доступа к данным определяют дальнейшее направление исследования задач подобного класса. Это, прежде всего, относится к изучению вопросов поддержки целостности данных в интегрированной системе с неоднородными структурами данных, а также разработки прикладных средств для расширения традиционных методов управления распределенными неоднородными данными. Среди вопросов, требующих дальнейшего изучения, необходимо выделить задачи, связанные с определением свойств моделей данных, для которых применимы математические средства реляционного исчисления при формировании запросов и последующей корректной трансформацией в реляционные языки манипулирования данными.

1. *Guarino N.* Understanding, building and using ontologies // Intern. J. of Human–Computer Studies. – 1997. – **46**, Iss. 2–3. – P. 293–310.
2. *Kuhns J.L.* Answering questions by computer: a logical study. RM-5428-PR, Rand Corp., Santa Monica, Calif., 1967. – 137 p.
3. *Клещев А.С.*, *Артемьева И.Л.* Математические модели онтологий предметных областей. Ч. 1. Существующие подходы к определению понятия «онтология» // Научно-техническая информация. Сер. 2. – 2001. – № 2. – С. 20–27.
4. *Цикритзис Д.*, *Лоховски Ф.* Модели данных. – М.: Финансы и статистика, 1985. – 343 с.
5. *Codd E.F.* Relational completeness of data base sublanguages. – IBID, 1972. – P. 65–98.
6. *Stonebraker B.*, *Wong E.* Access control in a relation database management system by query modification. ACM National conf., 1974. – P. 180–187.
7. *Gotlieb L.R.* Computing joins of relations. ACM Sigmod conf., 1975. – P. 55–63.
8. *Зельцер М.* За пределами реляционных баз данных: доступ к базам данных не ограничивается возможностями *SQL* // Data engineering. – 2005. – **3**, № 3. – P. 21–29.
9. *Єсін В.І.* Можливі шляхи створення єдиної бази даних діяльності ХВУ на основі універсальної моделі даних. – Харків: Наук.-метод. зб. ХВУ. – 2003. – № 3(89). – С. 3–12.
10. *Селко Дж.* Стиль программирования Джо Селко на *SQL*. – СПб.: Изд. дом «Питер», 2006. – 196 с.
11. *Мейер Д.* Теория реляционных баз данных. – М.: Мир, 1987. – 608 с.
12. *Ульман Дж.* Основы систем баз данных. – М.: Финансы и статистика, 1983. – 334 с.

Поступила 08.07.2010
Тел. для справок: (044) 526-0069 (Киев)
E-mail: *laponomarenko@ukr.net*