На правах рукописи

СОКОЛОВА Вероника Валерьевна

ТЕОРИЯ И АЛГОРИТМЫ ОБРАБОТКИ РЕКУРСИВНЫХ ИНФОРМАЦИОННЫХ СТРУКТУР

Специальность 05.13.01 – «*Системный анализ, управление и обработка информации* (*отрасль*: *информация и информационные системы*)»

АВТОРЕФЕРАТ

диссертации на соискание ученой степени кандидата технических наук

Томск – 2007

Работа выполнена на кафедре оптимизации систем управления Томского политехнического университета

Научный руководитель: ***НОВОСЕЛЬЦЕВ Виталий Борисович,*** кандидат физико-математических наук, доцент

Официальные оппоненты: ***КОЧЕГУРОВ Владимир Александрович***, доктор технических наук, профессор

КАЛАЙДА Владимир Тимофеевич, кандидат технических наук, доцент

Ведущая организация: ***Томский государственный университет***

Защита состоится «28» марта 2007 г. в **$15^{\underline{00}}$** часов на заседании диссертационного совета **Д212.269.06** при Томском политехническом университете по адресу: 634034, г. Томск, ул. Советская, 84, институт «Кибернетический центр» ТПУ.

С диссертацией можно ознакомиться в научно-технической библиотеке Томского политехнического университета по адресу: 634034, г. Томск, ул. Белинского, 53.

Автореферат разослан «27» февраля 2007 г.

Ученый секретарь диссертационного совета, к.т.н., доцент ***М.А. Сонькин***

1. Общая характеристика диссертации

Актуальность темы. В современных информационных системах, в частности, работающих с экономическими, медицинскими, химическими наборами данных, крайне важной является проблема построения и обработки рекурсивных информационных структур (структур, элементами которых выступают логически подобные комплексы). Данные структуры поддерживают большое количество скрытых зависимостей, которые важны для корректной обработки информации, например, в информационных системах поддержки менеджмента качества все процессы должны соответствовать циклу непрерывного улучшения (PDCA). Манипулирование рекурсивными наборами данных в стандартных (де-факто – реляционных) комплексах осуществляется, как правило, внесистемными средствами: внешними модулями, триггерами и т.д. Объясняется это отсутствием в классической алгебре Кодда механизмов, естественным образом поддерживающих оперирование рекурсивными комплексами. В настоящее время доступен некоторый инструментарий (опять же, технического характера), который можно найти в объектно-ориентированных оболочках. Однако отсутствие строгого теоретического аппарата, подобного реляционной алгебре, не обеспечивает уверенности в реализационных характеристиках конечного программного продукта (целостности, корректности, неизбыточности и т.д.), что, очевидно, вызывает необходимость дальнейших исследований в данной области.

Вторым фактором, определяющим актуальность работы, является то, что существующие на сегодняшний день CASE-средства, не предлагают эффективных решений в области моделирования рекурсивных наборов данных и их адекватного отображения в стандартное реляционное представление. Таким образом, необходим программный комплекс, интегрирующий процессы проектирования и представления в физическую структуру именно рекурсивных наборов данных.

Дополнительным важным моментом, подчеркивающим актуальность развиваемого во второй части диссертации подхода на базе конечных автоматов, является возможность естественной реализации связанных автоматных структур многопроцессорными исполнителями (кластерами, сетевыми структурами, мэйнфреймами и т.д.).

В работе строго показывается, что использование автоматного подхода позволяет эффективно обрабатывать рекурсивные зависимости. Учитывая, что общая задача обработки рекурсивного списка (развертка произвольной рекурсивной структуры) *NP*-трудна, предлагается корректная ограниченная стратегия достаточно эффективного (полиномиального) ее решения. Можно утверждать, что в проведенном исследовании предлагается совершенно новый подход к обработке рекурсивных структур данных.

Объектом исследования работы являются рекурсивные структуры данных. **Предметом исследования** – алгоритмы обработки и механизмы моделирования рекурсивных данных.

Целью работы является построение и анализ алгоритмов манипулирования рекурсивными информационными структурами данных и программного обеспечения для поддержки процесса их проектирования.

Для достижения поставленной цели в работе решаются следующие основные задачи:

1. Расширение алгебры Кодда средствами описания рекурсивных структур.
2. Доказательство замкнутости расширенной реляционной алгебры.
3. Дополнение теории организации данных с использованием формализмов конечных автоматов.
4. Разработка и программная реализация процесса проектирования схем баз данных с рекурсивно-определяемыми атрибутами.

Методы исследования. В процессе исследования использовался следующий инструментарий: методы теории множеств (теоретическая основа описания алгебр реляционного класса), аппарат математической логики для доказательства соответствующих теорем, теория алгоритмов, теория конечных автоматов, а также методы функционального и структурного программирования в качестве базиса практической реализации.

Научная новизна полученных результатов определяется следующими положениями:

- Предложено расширение реляционной алгебры (формальной теории) рекурсивными конструкциями, позволяющее организовывать и обрабатывать рекурсивные данные по типам атрибутов таблицы.
- Сформулирована и конструктивно доказана теорема о замкнутости расширенной алгебры, устанавливающая, что предложенное расширение алгебры Кодда является корректным.
- Предложен оригинальный формализм работы с рекурсивными структурами данных, обеспечивающий его реализацию с использованием аппарата конечных автоматов.
- Реализован программный комплекс, обеспечивающий проектирование и автоматический перевод обобщенных рекурсивных описаний в стандартное реляционное представление.

Практическая значимость. Все теоретические положения, приведенные в диссертации, формально строго обоснованы. Полученные во второй главе результаты, могут являться основой для создания промышленных систем управления базами данных и базами знаний с поддержкой рекурсии в качестве специализированного программного обеспечения. Разработанный комплекс предназначается для автоматизации процесса проектирования схем баз данных с рекурсивно-определяемыми атрибутами. При этом приложение обеспечивает преобразование исходных рекурсивных структур в стандартное реляционное представление.

Апробация работы. Основные положения и результаты диссертации докладывались и обсуждались на следующих конференциях международного и российского уровней:

- 8th Korea-Russia International Symposium on Science and Technology «KORUS 2004» (г. Томск, ТПУ, 26 июня – 3 июля 2004 г.).
- V Всероссийская конференция «Системы и средства автоматизации» (г. Томск, ТПУ, 21 – 22 октября 2004 г.).
- III Всероссийская научно-практическая конференция «Молодежь и современные информационные технологии» (г. Томск, ТПУ, 15 – 17 февраля 2005 г.).
- Международная конференция студентов и аспирантов по фундаментальным наукам «Ломоносов-2005» (г. Москва, МГУ, 12 – 15 апреля 2005 г.).
- X Байкальская Всероссийская конференция с международным участием «Информационные и математические технологии в науке, технике и образовании» (г. Северобайкальск, ИСЭМ СО РАН, 12 – 19 июля 2005 г.).
- 9th Asian Logic Conference (г. Новосибирск, НГУ, 16 – 19 августа 2005 г.).
- III Всероссийская конференция молодых ученых в рамках Российского научного форума с международным участием «Демидовские Чтения» (г. Томск, ИОА СО РАН, 3 – 6 марта 2006 г.).
- IV Всероссийская научно-практическая конференция студентов, аспирантов и молодых ученых «Молодежь и современные информационные технологии» (г. Томск, ТПУ, 28 февраля – 2 марта 2006 г.).
- Международная конференция «Инженерное образование и наука в мировом пространстве (GEER)», (г. Томск, ТПУ, 1 – 2 июня 2006 г.).

Публикации. Результаты диссертационного исследования изложены в 19 работах, в том числе 2 из перечня журналов, рекомендованных ВАК. Личный вклад автора в каждой публикации составляет 45-100%.

Личный вклад автора. Основные результаты диссертационной работы получены автором лично. Программный комплекс «RecModel 1.0» для проектирования схем баз данных с рекурсиями и представления их в стандартном реляционном описании разработан автором лично.

Внедрение результатов. Результаты работы используются в учебном процессе на кафедре Оптимизации систем управления ТПУ и в отделе Менеджмента качества Томского политехнического университета (КПР (ЦПР) № 3.3.3.1.2. «Развитие СМК ТПУ на базе ISO 9001-2000»), внедрены в Институте оптики атмосферы СО РАН (г. Томск), в компании «ЭлеСИ» (г. Томск) и в ОАО «ТЭЛЗ» (г. Томск).

Результаты, выносимые на защиту:

1. Расширение алгебры (формальной теории) Кодда рекурсивными конструкциями.
2. Доказательство замкнутости и корректности модифицированной реляционной теории.
3. Представление и обработка рекурсивных структур данных посредством аппарата конечных автоматов.
4. Программная реализация проектирования схем баз данных с рекурсивно-определяемыми атрибутами и перевода обобщенных рекурсивных описаний в стандартное реляционное представление.

Объем и структура диссертации. Диссертация включает в себя: введение, четыре главы, заключение, список литературы (120 наименований) и приложения, иллюстрирующие технические детали реализации программного комплекса. Общий объем работы составляет 150 страниц, включая 32 рисунка и 5 таблиц.

2. СОДЕРЖАНИЕ ДИССЕРТАЦИИ

Во введении обоснована актуальность реализуемой тематики, определены предмет и цели исследования, приведены содержательные формулировки сопутствующих задач, а также указаны возможные варианты использования полученных результатов.

Глава 1

В разделе систематизированы распространенные подходы и методы, применяемые в области обработки данных. Приведен аналитический обзор современных направлений исследований в области развития баз данных, включающих, главным образом, объектно-ориентированные и пост-реляционные информационные системы. Обсуждаются наиболее популярные подходы к хранению и обработке данных в информационных комплексах, определяются их преимущества и подчеркиваются недостатки.

В современной теории баз данных имеется большое количество различных моделей, отражающих особенности реальных систем (предметных областей). Наиболее распространенная классическая модель определяется реляционной алгеброй Кодда. Первоначально этот подход полностью удовлетворял потребности пользователей при обработке информации, но с появлением новых видов данных (графических, медиа- и видео-данных) стали появляться новые направления и в технологиях баз данных. Развитие средств вычислительной техники, но не математической теории, привело к разработке новых подходов.

Материалы первой главы перекликаются с работами С.Д. Кузнецова[1] и других ученых, чьи результаты связаны с проблемами развития технологий баз данных. В упомянутых работах отмечается необходимость выполнения рекурсивных запросов, но не приведены механизмы их реализации в существующих СУБД.

Исходя из анализа доступных источников, сделано заключение, что предпринимаемые попытки расширения реляционного подхода являются непринципиальными, так как носят, преимущественно, технический характер. Выявлено, что информационные системы следующего поколения (дедуктивные, объектно-ориентированные и другие) – это прямые, хотя и более технически-развитые по сравнению со своими предками наследники реляционных систем со всеми их недостатками.

Материал главы позволяет сформулировать цель работы, а также выделить ключевые задачи и основные направления их решения.

Глава 2

В главе с учетом проведенного ранее анализа предпринимается попытка расширения классической алгебры, призванная обеспечить возможность обработки рекурсивных наборов данных для широкого круга предметных областей. Для достижения этой цели переопределяются основные понятия алгебры Кодда, выделяются существенные для дальнейшего изложения положения реляционной теории и доказывается замкнутость расширенной алгебры.

Фиксируется сигнатура.

Определение 2.1. Сигнатурой $\Sigma = <\mathcal{A}, \mathcal{D}, \mathcal{F}>$ назовем тройку, где $\mathcal{A}$ – множество имен сущностей, $\mathcal{D}$ – множество имен доменов, $\mathcal{F}$ – множество имен функциональных связей над атрибутами.

Все множества сигнатуры являются не более чем счетными. Во множестве $\mathcal{D}$ зафиксировано непустое конечное подмножество $\mathcal{T} \subset \mathcal{D}$ – имен первичных доменов (предопределенных типов).

Определение 2.2. Под *модельным объектом (модельной сущностью)* понимается формальное в рамках рассматриваемой предметной области определение конкретной сущности, отличимой от другой. Все объекты обладают свойствами, называемыми *атрибутами*.

Определение 2.3. Под термином *атрибут* будем понимать пару (имя сущности, тип). *Интерпретированный атрибут* – суть подмножество элементов соответствующего типа, *денотативно* выделенное из последнего.

Определение 2.4. Под термином *кортеж*, будем понимать список атрибутов.

Определение 2.5. *Ключ* – это минимальный набор именованных атрибутов, определяющий оставшуюся часть кортежа.

---

[1] *Кузнецов, С.Д. Основы баз данных. Курс лекций: Учебное пособие – М. : ИНТУИТ, 2005. – 488 с.*

Определение 2.6. *Домен данных* – это (не более, чем счетное) множество некоторых однородных допустимых значений. Например, можно рассматривать множество целых чисел как домен. Для предметной области «Сотрудники» домен «разряд» будет принимать числовые значения, определенные Единой тарифной сеткой.

Домен можно трактовать как подмножество значений некоторого типа данных имеющих определенный смысл. Домен характеризуется следующими свойствами:

1. Домен имеет уникальное имя (в пределах базы данных).
2. Домен определен на некотором простом типе данных или на другом домене.
3. Домен может иметь некоторые логические свойства, позволяющие специфицировать его (домена) подмножества.
4. Домен несет определенную смысловую нагрузку.

При задании конкретной интерпретации **I** именам доменов (в том числе базовых) сопоставляются конкретизированные типы (множества). Предполагается, что именам базовых доменов при этом сопоставляются традиционные для программирования типы (например, *integer*, *real*, *char*, *Boolean* и т.д.).

Непервичными доменами могут выступать именованные подмножества базовых (например, $Name \subset string$), либо подмножества декартовых произведений ранее определенных доменов ($Person \subset Name \times Age \times Sex$).

Суммируя сказанное, уточним определение домена следующим образом.

Определение 2.7. Пусть задана интерпретация **I**, $P \in \mathcal{T}$, $R \in \mathcal{D} \setminus \mathcal{T}$, тогда

1) $P|_{\mathbf{I}}$ – домен;
2) если $R|_{\mathbf{I}} \subset P|_{\mathbf{I}}$, то $R|_{\mathbf{I}}$ – домен;
3) если $R_i|_{\mathbf{I}}$ $(i=\overline{1,n})$ – домены, то $R_0|_{\mathbf{I}} \subset R_1|_{\mathbf{I}} \times \ldots \times R_n|_{\mathbf{I}}$ – домен.

Заметим, что все высказанные ранее соображения относительно семантической целостности доменов остаются актуальными.

Прежде чем переходить к фиксации базового для реляционной алгебры понятия таблицы, отметим, что наследники примитивных типов импортируют все операции и отношения от порождающих. – Так, для $Age \subset Positiv\_Integer$ имеет место отношение “>” и все операции для целых положительных.

Определение 2.8. *Отношением* является некоторое подмножество декартова произведения одного или более доменов. Отношение обычно записывается в следующем виде: $R(A_1, A_2, \ldots, A_n)$.

Отношение обладает двумя важными свойствами:

1) в отношении не содержится совпадающих кортежей;
2) порядок кортежей в отношении несущественен.

Определение 2.9. Назовем *таблицей* подмножество некоторого фиксированного домена, актуализированное применением допустимых в теории операций.

Термины «*отношение*» и «*таблица*» в значительной степени являются синонимичными, тем не менее, их имеет смысл различать.

Определение 2.10. Под *реляционной базой данных* будем подразумевать совокупность взаимосвязанных отношений.

Содержательным представлением (нерекурсивного) отношения является плоская таблица, заголовок которой определяется упорядоченным списком атрибутов, а строки – кортежи – соответствующим образом упорядоченных значений, при этом атрибуты именуют столбцы таблицы. Поэтому иногда говорят «столбец таблицы», имея в виду «интерпретированный атрибут отношения». Этой терминологии придерживаются в большинстве коммерческих реляционных СУБД.

Определение 2.11. *Схема таблицы (домена)* – это именованное множество пар (имя атрибута, имя домена) вида $D(r)=r.(D_1(a_1),\ldots,D_n(a_n))$, где D, $D_i \in \mathcal{D}$ $(i=\overline{1,n})$ – имена доменов, r, $a_i \in \mathcal{A}$ $(i=\overline{1,n})$ – имена атрибутов. В правой части определения r может «проноситься» в скобки на любую глубину, так что $r.(D_1(a_1),\ldots,)$ $(r.D_1(a_1),\ldots,)$ $(D_1(r.a_1),\ldots,)$.

Иногда для большего технического удобства используется индуцируемое схемой таблицы понятие *кортеж-тип*, при этом осуществляется переход от функционального представления к типизированному.

Определение 2.12. *Кортеж-тип* для схемы D из определения 2.11 представляется формой $C=<a_1:D_1, a_2:D_2,..., a_n:D_n>$ с аналогичными функциями принадлежности. *Степень* или *арность* схемы отношения определяется количеством атрибутов n схемы.

Определение 2.13. *Схема БД* – это набор именованных схем отношений.

Например, схема отношения «СОТРУДНИК» выглядит так: сотрудник((фамилия, *text*(20)), (имя, *text*(15)), (отчество, *text*(15)), (дата рождения, *date*), (пол, *Boolean*)). При этом заранее может быть задан домен *пол* с областью значений (М, Ж).

Определение 2.14. Под *реляционной алгеброй* понимаем совокупность объектов и бинарных операций над ними, сохраняющие рефлексивность, симметричность и транзитивность.

Согласно Кодду[1], набор операций алгебры определяется восемью дефинициями, которые делятся на два класса: теоретико-множественные и специальные реляционные операции. В состав теоретико-множественных входят операции объединения, пересечения, взятия разности и декартова произведения. Специальными реляционными операциями являются: ограничение отношения, проекция, соединение и деление отношений. Кроме того, в состав алгебры неявно включается операция присваивания (*связывания*), позволяющая сохранить в базе данных результаты вычисления алгебраических выражений, и операция переименования атрибутов, дающая возможность корректно сформировать заголовок (схему) результирующего отношения.

Приведем определение операции объединения (остальные операции алгебры формулируются аналогично).

---

[1] *Codd, E. F. A Relational Model of Data for Large Shared Data Banks. // CACM. – 1970. - № 13, n 6. - P. 30-42.*

Определение 2.15. Пусть даны два непервичных домена $T_1 \subset D_1$ и $T_2 \subset D_2$, где $D_1 \triangleq D_2$ синтаксически равны, тогда результатом операции объединения двух отношений является отношение, включающее все кортежи, входящие хотя бы в одно из отношений-операндов. Таким образом, результат объединения это таблица $T|_{\mathbf{I}}$ $T_1|_{\mathbf{I}} \mathrm{Y} T_2|_{\mathbf{I}}$, где $(T|_{\mathbf{I}}:(t|_{\mathbf{I}} \in T_1|_{\mathbf{I}}) \vee (t|_{\mathbf{I}} \in T_2|_{\mathbf{I}}))$. Считается, что совпадающие кортежи элиминируются.

Поскольку результатом любой реляционной операции является некоторое отношение, возможно образовывать реляционные выражения, в которых вместо отношения-операнда некоторой реляционной операции находится вложенное реляционное выражение.

Реляционная алгебра представляет собой набор операторов, использующих отношения в качестве аргументов, и возвращающие отношения в качестве результата. Таким образом, реляционный оператор f выглядит как функция типа «отношения» с отношениями в качестве аргументов: $R=f(r_1,r_2,\ldots,r_n):r_0$.

Реляционная алгебра является замкнутой, поскольку в качестве аргументов в реляционные операторы можно подставлять другие реляционные операторы, подходящие по типу: $R=f(f_1(r_{11},r_{12},\ldots,r_{1n}),f_2(r_{21},r_{22},\ldots,r_{2n}),\ldots)$.

Резюмируем вышесказанное: в реляционных выражениях можно использовать вложенные подвыражения сколь угодной глубины.

Рекурсивным называется объект, частично определяемый с помощью самого себя. Пример такого объекта представлен на рисунке 1.

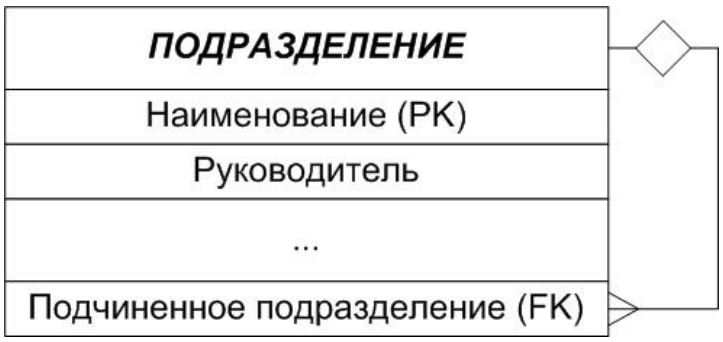

Рис. 1. Схема рекурсивного объекта

В диссертации рассматриваются рекурсивные таблицы. Везде ниже допускается только явная рекурсия в смысле следующего определения.

Определение 2.16. Таблица $T(r)=r.(a_1:D_1,a_2:D_2,\ldots,a_n:D_n)$ является *допустимой*, если синтаксическое равенство D_i T, определяющее рекурсию, выполняется не более, чем для n-1 доменов.

Заметим, что *кортеж* не может ссылаться на самого себя ни непосредственно, ни опосредованно.

Расширим реляционную алгебру, переопределив понятие таблицы следующим образом:

Определение 2.17. *Таблица* – это совокупность конечного числа кортежей конечной длины, таких что

1. если $C=<a_1:D_1,a_2:D_2,...,a_n:D_n>$ – кортеж-тип, где все атрибуты a_i попарно различны, то C назовем ос'новным кортежем-типом или основой;
2. обобщенной таблицей *основной* таблицы T будем называть конечную совокупность регулярных кортежей-значений C_r определяемых *основным* кортежем–типом C таблицы T, при этом
   - кортежи–значения C_r попарно различны;
   - каждый кортеж $C_r|_{\mathbf{I}}$ m-кратно ($m\geq 1$) повторенный набор атрибутов-значений $<a_{11},a_{12},...,a_{1n},<a_{21},a_{22},...,a_{2n},...,<a_{m1},a_{m2},\ldots,a_{mn}\ldots>>>$, таких что $a_{ik}\in D_k$ ($1\leq k\leq n$) основного кортежа C и никакие два подкортежа $<a_{i1},a_{i2},\ldots,a_{in}>$ и $<a_{j1},a_{j2},...,a_{jn}>$ не совпадают при $i\neq j$;
   - хотя бы один из атрибутов первого подкортежа имеет значение *nil* в позиции рекурсивной ссылки (выход из рекурсии).

Частный случай основной рекурсивной таблицы можно представить в виде, изображенном на рис. 2.

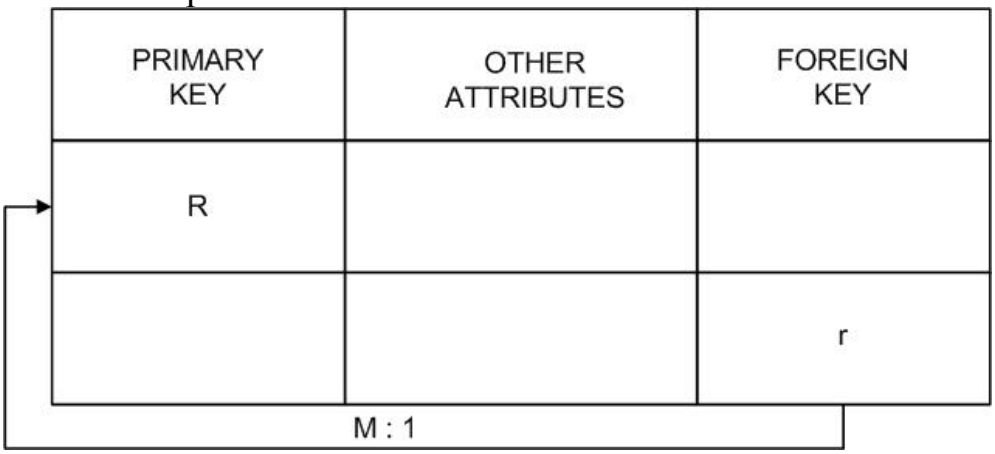

Рис. 2. Схема основной рекурсивной таблицы

Определение 2.18. Для обобщенной таблицы T стандартным образом определены ключевой набор K и индексный набор I , такие что (1) $D(K)\subset D(I)$, (2) $K\cap I=\varnothing$ и (3) определено отображение $R:I\rightarrow K$, причем неподвижной точкой такого транзитивного замыкания R^* является пустое значение *nil*.

Тогда рекурсивным эквисоединением таблицы T по отношению R называется операция, результатом которой выступает обобщенная таблица $T_{\boldsymbol{R}}$, такая что для каждого её кортежа – значения $C_r|_{\mathbf{I}}$, состоящего более чем из двух кортежей основной таблицы $<c_1,c_2,\ldots,c_m>$:

1) $I^+(c_m)=nil$;
2) $I^+(c_{k-1})=K(c_k)$, где $k\leq m$.

Итак, мы *расширили реляционную алгебру*, переопределив понятие таблицы и соответствующих операций.

Далее приводится доказательство теоремы о замкнутости расширенной алгебры: объекты данных и операции над ними в смысле определения 2.17 образуют замкнутую систему.

Приведем доказательство теоремы для операции объединения. Формулировка теоремы означает, что результатом применения каждой из введенных операций (объединение, пересечение, вычитание, проекция,

селекция, соединение) к произвольным таблицам T_1 и T_2 будет таблица в смысле того же определения 2.17.

Теорема 2.1. Рассмотрим таблицы T_1 и T_2. Пусть $G=T_1 \Upsilon T_2$. Покажем, что G удовлетворяет определению 2.17.

Доказательство (схема).

а) Поскольку T_1 и T_2 – таблицы в смысле определения 2.17, то каждая из них состоит из попарно различных кортежей C'_r ($1 \le r \le n$) и C''_r ($1 \le r \le m$) соответственно. Тогда по определению операции объединения таблица G будет также состоять из попарно различных кортежей C_r ($1 \le r \le k$, $k \le m+n$ – повторяющиеся кортежи исключаются).

б) Аналогично, в таблице G каждый кортеж есть m-кратно повторенный набор атрибутов-значений $<a_{11},a_{12},...,a_{1n},<a_{21},a_{22},...,a_{2n},\ldots,<a_{m1},a_{m2},...,a_{mk}>>>$, таких, что $a_{is} \in D_s$ ($1 \le s \le n$) основного кортежа C и никакие два подкортежа $<a_{i1},a_{i2},\ldots,a_{ik}>$ и $<a_{j1},a_{j2},...,a_{jk}>$ не совпадают при $i \ne j$.

в) T_1 – таблица в смысле определения 2.17. Следовательно, хотя бы один из атрибутов первого подкортежа имеет значение *nil*. T_2 – также таблица в смысле определения 2.17. Следовательно, хотя бы один из атрибутов её первого подкортежа имеет значение *nil*. Тогда таблица G будет также содержать хотя бы один такой подкортеж (по действию операции объединения).

г) T_1 – таблица в смысле определения 2.17, следовательно, в ней заданы ключевой набор K' и индексный набор I', такие что $D(K')=D(I')$ и $K' \cap I'=\varnothing$. T_2 – также таблица в смысле определения 2.17, следовательно, и в ней заданы ключевой набор K'' и индексный набор I'', такие что $D(K'')=D(I'')$ и $K'' \cap I''=\varnothing$. Из требования совместимости таблиц T_1 и T_2 по объединению и действию операции объединения вытекает, что $D(K)=D(I)$ и $K \cap I=\varnothing$, где $K=K' \cup K''$, $I=I' \cup I''$.

д) T_1 –таблица в смысле определения 2.17, следовательно, существует отображение $R':I' \to K'$, причем неподвижной точкой такого транзитивного замыкания R'^* является значение *nil*. T_2 – таблица в смысле определения 2.17, следовательно, существует отображение $R'':I'' \to K''$, причем неподвижной точкой такого транзитивного замыкания R''^* является значение *nil*. В таблице G можно определить отображение $R:I \to K$, которое действует по правилу $R(i)=R'(i)$ при $i \in I'$ и $R(i)=R''(i)$ при $i \in I''$. Транзитивное замыкание R^* будет иметь неподвижные точки, значение которых *nil*.

е) Для каждого кортежа–значения $C_r|_{\mathbf{I}}$ таблицы G, состоящего более чем из двух кортежей основной таблицы $<c_1,c_2,\ldots,c_m>$ будут выполняться условия $I^+(c_m)=nil$ и $I^+(c_{k-1})=K(c_k)$, $k \le m$.

Таким образом, результатом операции объединения является таблица в смысле определения 2.17. Аналогичные доказательства приводятся для остальных операций.

Глава 3

В данной главе рассматривается подход к решению задачи обработки рекурсивных данных, который основан на теории конечных автоматов. Рассмотрены основные понятия теории автоматов. Показано, как методы конечных автоматов могут быть использованы для обработки рекурсивных объектов.

В рамках предлагаемого подхода основная роль отводится функциональной стороне. Считается, что любые данные появляются в результате интерпретации первичной составляющей модели, либо применения некоторого автомата к другим данным. Таким образом, необходимо переопределить понятие запроса.

Определение 3.1. *Атомарный запрос* есть первичное значение, допустимое в заданной интерпретации. *Непервичный (автоматный) запрос* определяется некоторым конечным автоматом и реализуется применением этого автомата к входным данным.

Входной поток для фиксированного запроса (автомата) представляет собой список, сформированный другими атомарными либо автоматными запросами. Термины *запрос* и *автомат* считаются синонимами, если не оговорено противного.

Различаются явное описание запроса и его реализация. Согласно классическому определению[1], конечный автомат – это система с ограниченным набором состояний и определенной дисциплиной переходов, поэтому любая система, обладающая ограниченным набором возможностей, каждая из которых может быть отмечена отдельным состоянием, может быть представлена в виде конечного автомата (КА).

Для строгого введения последующих понятий зафиксируем сигнатуру:

Определение 3.2. Сигнатурой назовем тройку $\Sigma=(\mathcal{A}, \mathcal{E}, \mathcal{T})$, здесь $\mathcal{A}$ – множество символов атрибутов, $\mathcal{E}$ – множество символов состояний, $\mathcal{T}$ – множество символов доменов (возможных типов). Полагаем, что в $\mathcal{T}$ зафиксировано непустое подмножество $\mathcal{D}\subset\mathcal{T}$ символов первичных доменов ($\mathcal{D}\neq\varnothing$), содержащих базовые значения атрибутов (например, *integer*, *date*, *text* и т. д.). Считается, что при любой интерпретации первичному домену сопоставляется перечислимое множество элементов.

Определим схему обобщенной (автоматной) рекурсивной таблицы.

Определение 3.3. Обобщенная рекурсивная таблица T определяется записью $T=(a_1{:}T_1,a_2{:}T_2,\ldots,a_n{:}T_n)$, при этом $T_i\in\mathcal{T}$ – принадлежат множеству символов типов сигнатуры и, как минимум, один $T_i\in\mathcal{D}$, $a_i\in\mathcal{A}$ – принадлежат множеству символов атрибутов. Разрешено синтаксическое равенство (текстуальное совпадение) T и одного или нескольких (но не более, чем n-1) T_i. При рассмотрении набора таблиц допускается появление неявной или

[1] *Карпов, Ю. Г. Теория алгоритмов и автоматов. – СПб. : Геликон Плюс, 2000. – 256 с.*

опосредованной рекурсии, причем требование наличия в описании каждой схемы хотя бы одного первичного атрибута гарантирует конечность реализации любой таблицы.

Списочная реализация рекурсивной таблицы может быть представлена следующим образом:

$((a_{11}\ a_{12}\ (a_{21}\ a_{22}\ nil))\ \ (a_{31}\ a_{32}\ (a_{41}\ a_{42}\ (a_{51}\ a_{52}\ nil))))$.

Автомат, воздействуя на входную строку (домен), может активизировать другой КА или самого себя (рекурсивно и, возможно, опосредованно):

K_{00}:(a_{00}:D_{00},…,a_{0k}:D_{0k},a_{0k+1}:↓D_{0k+1},…,a_{0k0}…↓…)
(K_{11}:(a_{10}:D_{10},…,a_{1k}:D_{1k},a_{1k+1}:↓D_{1k+1},…,a_{1k0}…) (K_{12}:(…)…)…).

Определение 3.4. С учетом заданной сигнатуры непервичный автомат Q определяется следующим образом: Q (A,R,Δ,σ,F), где $Q\in\mathcal{T}\setminus\mathcal{D}$ – Q не является именем первичного домена; A – алфавит над декартовым произведением $\mathcal{A}\times\mathcal{T}$, то есть это все комбинации допустимых пар из декартова произведения типов и имен атрибутов; $R\in\mathcal{E}$ – конечное множество состояний автомата; Δ – функция перехода, определенная на множестве $A\times R$; $\sigma\in R$ – начальное состояние; $F\subseteq R$ – непустое множество конечных состояний.

В каждый фиксированный момент времени (такт) конечный автомат находится только в одном из возможных состояний, при этом число состояний, в которых может находиться конечный автомат – конечно. Автомат последовательно считывает элементы из входной строки. Каждый считанный символ либо переводит автомат в новое состояние, либо оставляет его в прежнем, одновременно КА генерирует элемент в выходную строку.

Функция перехода, ассоциированная с автоматом Q, является частичной функцией Δ:$A\times R\to A\times R$. Областью определения данной функции является декартово произведение атрибутов и состояний, допустимых для конечного автомата Q, а множеством значений – преобразованные конечным автоматом пары <*элемент_алфавита, состояние*>. При необходимости подчеркнуть связь Δ с автоматом Q используется запись Δ^{Q}**.**

Определение 3.5. Считается, что функция перехода, как правило, дискретна, т.е. может быть задана множеством пар: Δ $\{(\alpha,\delta),(\alpha',\delta')\}$.

На рис. 3 данным парам соответствуют x – текущий анализируемый атрибут, y – атрибут, который автомат записал в выходной поток после обработки, и переход между состояниями $\delta\to\delta'$.

Входом и выходом любого автомата являются регулярные списки, которые могут быть вложенными. Двухуровневые списки, при соблюдении некоторых условий, сопоставляются плоским таблицам реляционного подхода – ниже рассматриваются именно такие списки. Списки большей вложенности также не запрещены, так, трехуровневые структуры могут использоваться при моделировании темпоральных баз данных.

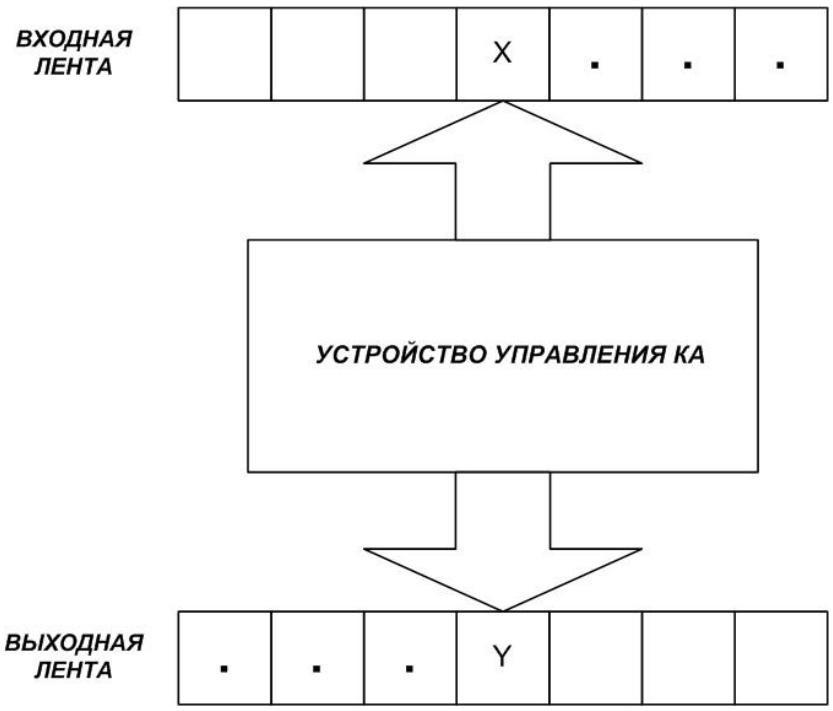

Рис. 3. Структурная схема конечного автомата

Введем уточненное понятие атрибута, которое понадобится для более строгого определения входного и выходного потоков автомата и других необходимых структур. Будем различать *имя* атрибута и его *значение (интерпретацию)*, а также *первичные (примитивные)* и *непервичные* атрибуты. Прежде всего, определим рекурсивно понятие *допустимого имени*.

Пусть $a \in \mathcal{A}$, α – допустимое имя, тогда a и $\alpha.a$ – также допустимые имена, во втором случае α и a называются префиксом и суффиксом, соответственно. Допустимое имя α становится *именем атрибута*, только если оно связывается с некоторым доменом $T \in \mathcal{T}$. Факт связывания отмечается записью $T(\alpha)$ или $\alpha.T$ (точечная пара, согласно нотации языка ЛИСП) – первая форма подчеркивает функциональность связывания, а вторая – его ссылочность. Если $T \in \mathcal{D}$, и α – примитивный атрибут, то $\alpha.T$ заменяется на α^T либо на α, когда ссылка на домен не существенна или очевидна из контекста. Связь именованного объекта с интерпретацией **I** изображается стандартной записью $(имя)|_{\mathbf{I}}$. Интерпретация задается традиционно: примитивному атрибуту сопоставляется константа из первичного домена $\alpha^T|_{\mathbf{I}} \in T|_{\mathbf{I}}$, тогда автоматной функции Δ^Q соответствует отображение $\Delta^Q|_{\mathbf{I}}: A|_{\mathbf{I}} \times R|_{\mathbf{I}} \to A|_{\mathbf{I}} \times R|_{\mathbf{I}}$.

Определение 3.6. Любой непервичный домен T, где $T \in \mathcal{T} \setminus \mathcal{D}$ является структурой над *именованными* компонентами: $a.T \leftrightharpoons a.(a_1.T_1,\ldots,a_n.T_n)$, где a – параметр-модификатор имен элементов списка согласно точечной нотации, a, $a_i \in \mathcal{A}$ – попарно-различные, $T \in \mathcal{T} \setminus \mathcal{D}$, $T_i \in \mathcal{T}$. В определении домена допускается рекурсия (в том числе, и неявная).

Интерпретация $\alpha.T|_{\mathbf{I}}$ определяет пустой или сколь угодно длинный (но конечный) *регулярный* список вида $\alpha.((a_1.t_{11}|_{\mathbf{I}},\ldots,a_n.t_{n1}|_{\mathbf{I}})\ldots(a_1.t_{1k}|_{\mathbf{I}},\ldots,a_n.t_{nk}|_{\mathbf{I}}))$, где

для всех значений индексов $i=(1,\ldots,n)$, $j=(1,\ldots,k)$, $a_i \in A$, $t_{ij}|_{\mathbf{I}} \in T_i|_{\mathbf{I}}$, $T_i \in \mathcal{T}$. Регулярность списка определяется тем, что различные t_{ij} и t_{im} принадлежат одному и тому же домену T_i. Префикс может быть продолжен в скобочные фрагменты на любую глубину, так что $\alpha.((a_1.t_{11}\ldots)\ldots) \leftrightharpoons (\alpha.(a_1.t_{11}\ldots)\ldots) \leftrightharpoons ((\alpha.a_1.t_{11}\ldots)\ldots)$. Такие структуры можно считать *таблицами*, подразумевая, что подсписки соответствуют строкам (кортежам) таблиц реляционного подхода. Ясно, что t_{ij}, в свою очередь, могут являться списками, возможно, пустыми.

В *последнем* параграфе главы суммируются выводы. Представленный подход к обработке рекурсивных наборов данных является весьма привлекательным для обработки данных широкого круга предметных областей.

Глава 4

В этой главе рассматриваются реализационные аспекты, использующие функциональный аппарат системы программирования ЛИСП. В главе также приведено описание разработанного программного приложения «RecModel 1.0» и процесса проектирования схемы базы данных на примере информационной системы поддержки менеджмента качества.

Зададим описание автомата λ-термом в стиле функциональных языков.

Определение 4.1. Автомат определяется функциональной записью вида $Q=(\lambda x.\Delta[x])$ при этом $Q \in \mathcal{T}\mathcal{D}$, где Q – имя автомата не могущее быть именем из первичного домена, а $\Delta[x]$ традиционно обозначает вхождение (элементов) списка x в форму Δ.

В наиболее общем случае структура формы Δ определяется следующей конструкцией согласно нотации языка ЛИСП:

(*DO* (<*init_form*>) (<*stop_condition*> <*actions*>) <*actions*>).

Интерпретацией автомата Q, как отмечалось выше, является применение его к списку S *интерпретированных* атрибутов – это обозначается традиционной для функционального подхода записью $(Q\ S|_{\mathbf{I}})$ или $(Q\ S)$, если интерпретация очевидна из контекста. Результатом интерпретации автомата является также список (возможно, пустой), который, в свою очередь, может использоваться в качестве домена при спецификации нового атрибута. Поскольку атрибуты S могут являться применениями автоматов, возникает вопрос о трактовке операции суперпозиции автоматов, который решается через вложенный функциональный вызов.

Рассмотрим ряд дополнительных соглашений относительно рекурсивных структур данных. Пусть регулярный список L некоторым образом порождается доменом $T_0=(a_1.T_1,\ldots,a_n.T_n)$, состоящим из первичных атрибутов. Условимся рассматривать только такие списки, у которых первый атрибут связан с первичным доменом, то есть $T_1 \in \mathcal{D}$ (*синтаксическое ограничение*). Соответственно, первый атрибут $(a_1.T_1)$ будем называть *первичным ключом* и

считать его *уникальным именем* любого подсписка списка L верхнего уровня, так что в $L=\alpha.((a_1.t_{11}|_{\mathbf{I}},\ldots,a_n.t_{n1}|_{\mathbf{I}})\ldots(a_1.t_{1k}|_{\mathbf{I}},\ldots,a_n.t_{nk}|_{\mathbf{I}}))$ значения всех $a_1.t_{1i}|_{\mathbf{I}}$ – попарно-различны, т. е. уникальны (семантическое ограничение).

Определение 4.2. Множество K_L $\{a1.t1_i|_{\mathbf{I}}\}$ будем называть *ключевым множеством* списка L. Считается, что на элементах K определено отношение полного порядка ("$<K$"), как правило, индуцируемое соответствующим отношением на T_1, а также операция инкрементирования. Важно подчеркнуть, что в набор параметров автомата x определения автомата в качестве обязательного параметра входит системный ключ a_1. Отметим, что отношения порядка могут быть определены на домене любого атрибута (группы атрибутов), – это позволяет организовывать необходимые индексированные цепочки, что аналогично понятию ключа в реляционном подходе.

Подчеркнем еще раз, что любой домен является порождением автомата, при этом первичные домены трактуются, как реализации *is-a* отношений (автоматов) вида (*is-a integer x*), доставляющих истину, если предъявленный объект распознается корректно.

Определение 4.3. Переопределим понятие таблицы данных в предложенных (функциональных) соглашениях. Для этого зададим понятие схемы таблицы для объекта данных r – это поименованный список тэгов вида: $T(r)=r.(\ldots,T_i(a_i),\ldots)$. При рассмотрении набора таблиц допускается появление неявной или опосредованной рекурсии, причем требование присутствия в описании каждой схемы хотя бы одного первичного атрибута гарантирует конечность реализации любой таблицы.

Определение 4.4. Для «свертки данных» используется автомат-конструктор: (CONSTR($T(a_1,a_2,\ldots,a_n)$)) ($T(a_1,a_2,\ldots,a_n)$). Во внешней по отношению к конструктору переменной запоминается прохождение всех рекурсивных ответвлений при обработке данных, что сокращает время обработки.

Далее необходимо определить обратную операцию – развертку данных на непервичном атрибуте.

Определение 4.5. Пусть $T(r)=r.(\ldots,T_{ij}(a_{ij}),\ldots)$ – описание домена, и $T_{ij}\notin D$. Развёрткой домена T на атрибуте a_{ij} будем называть список, получающийся в результате подстановки в описание исходного домена на место имени $T_{ij}(a_{ij})$, раскрытого соответствующим автоматом, подсписка-домена $T_{ij}(a_{ij})$. Все добавленные в развернутый домен T в результате такого действия атрибуты модифицируются префиксом $r.a_{ij}$, что индуцирует и модификацию автомата, доставляющего реализацию домена $T_{ij}(a_{ij})$. Процесс построения развёртки на атрибуте может повторяться многократно – пока имеются атрибуты, связанные с неэлементарными схемами.

Зададим операцию полной развертки**.**

Определение 4.6. Под *полной развёрткой T* понимается завершенный процесс применения соответствующего автомата к интерпретации домена.

Полная развёртка реализует систему строго вложенных стеков, которая для рекурсивных доменов, вообще говоря, может оказаться неограниченной.

Для сохранения конструктивности построений как раз и вводятся понятия *первичного ключа* и введенное понятие *замыкания атрибутов* относительно автоматного домена (запроса).

На следующем рисунке приведена развёртка рекурсивной схемы.

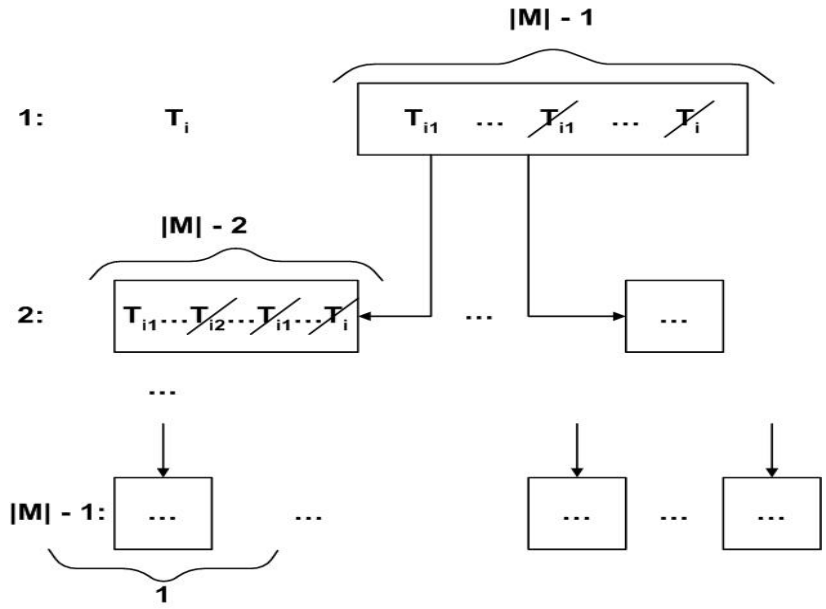

Рис. 4. Развёртка рекурсивной схемы таблицы

Определение 4.7. Пусть T – автоматный (возможно, рекурсивный) домен и пусть A – некоторый набор атрибутов домена. *Замыканием* A^+ для A относительно T будем называть объединение всех таких атрибутов X, что автоматное отображение $Q{:}A{\to}X$ может быть получено из информации о домене T. Здесь Q – (автоматный) терм в сигнатуре Σ, построенный по специальным правилам, A и X – непустые наборы атрибутов. Замыкание A^+ используется для обеспечения семантической необходимости выхода из рекурсии в практических случаях.

В предлагаемом подходе результат обработки произвольного списка вычисляется «по частям», т.е. вместо вычисления полного списка $\alpha.(a_1.t_1,a_2.t_2,\ldots,a_n.t_n)$ можно найти первый элемент $\alpha.(a_1.t_1)$ и выполнить вычисления остальных элементов с помощью автоматов, реализующих конкретные операции. При этом получается принципиальная экономия памяти ценой незначительного перерасхода времени на вспомогательное построение.

Общий алгоритм обработки рекурсивных запросов в базах данных приведен на рисунке 5, а более подробные схемы – в приложениях к работе.

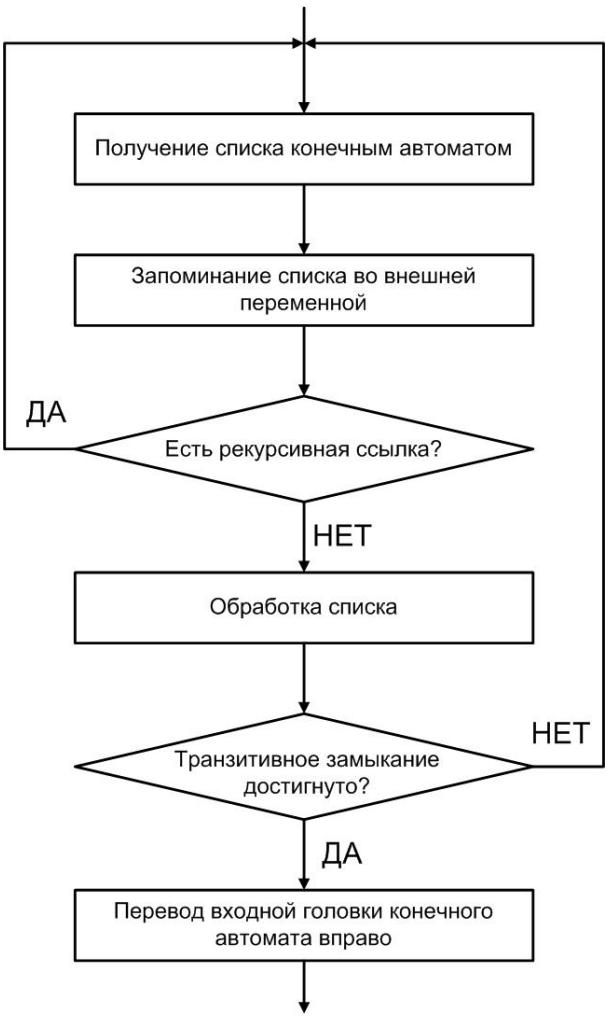

Рис. 5. Общий алгоритм выполнения запросов

В данной главе также рассматривается программная реализация процесса проектирования схемы баз данных с рекурсивными структурами с помощью разработанного программного комплекса «RecModel 1.0». В целом программное моделирование рекурсивных зависимостей соответствует нотации IDEF1X.

Программа выполняет следующие основные функции:

- предоставляет возможность автоматического построения схемы базы данных из описаний структур данных;
- обеспечивает построение ER-диаграммы (модель «сущность-связь»);
- позволяет создать корректную нормализованную модель данных;
- предоставляет возможность генерации SQL-кода для создания таблиц базы данных с учетом ограничений целостности.

Пример окна программы «Создание таблицы» представлен на рисунке 6. В данном диалоговом окне пользователь может задать описание полей таблицы по основным параметрам: название поля, тип поля, размер поля, значение по умолчанию, описание поля, признак ключевого поля, тип ключа (первичный, внешний, рекурсивный), уникальность и ограничения на вводимые значения.

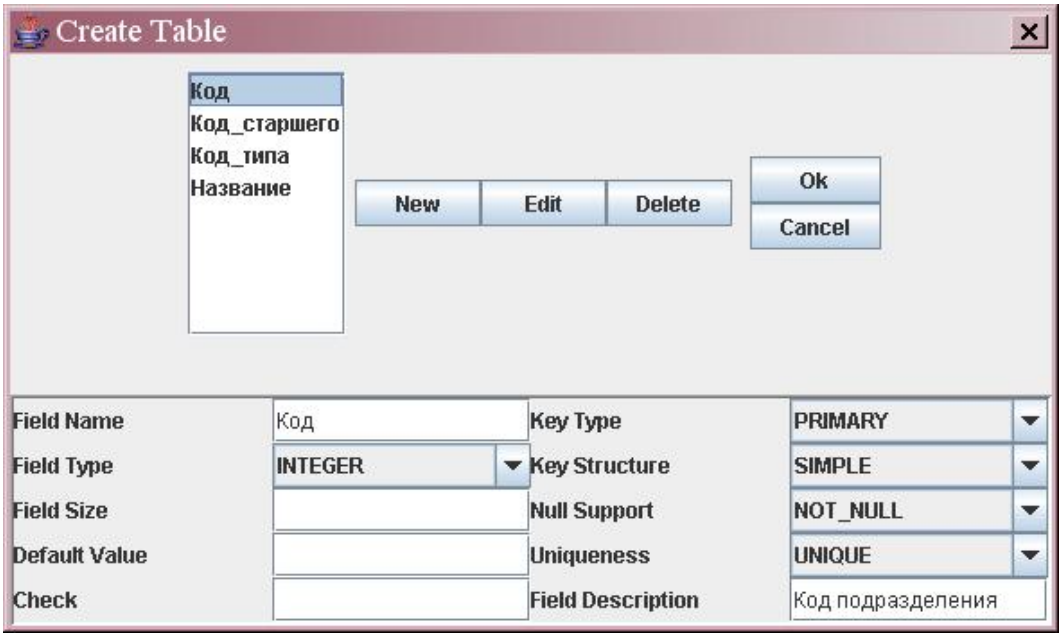

Рис. 6. Диалоговое окно «Создание таблицы»

Для связывания таблиц между собой, пользователь должен установить их в диалоговом окне «Создание связей» как показано на рисунке 7.

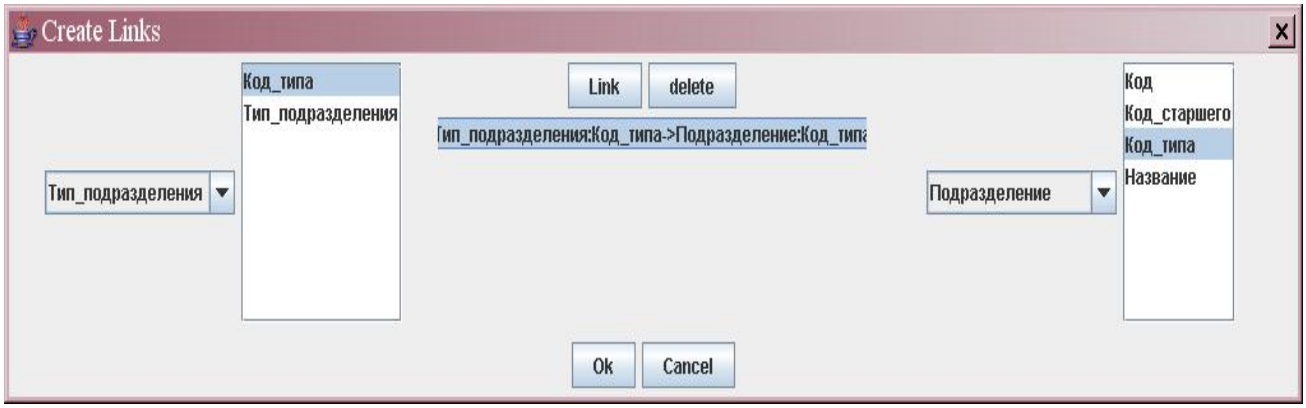

Рис. 7. Диалоговое окно «Создание связей»

На следующем рисунке представлена схема класса установления связей между таблицами.

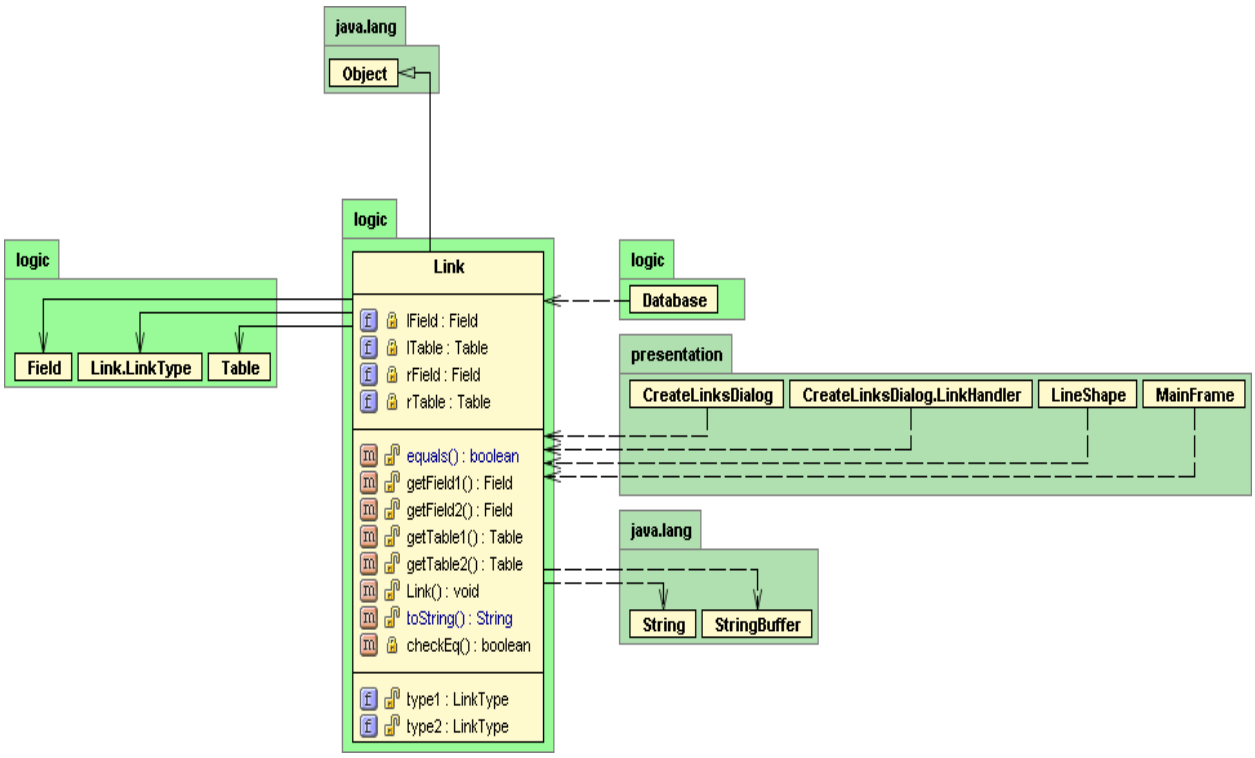

Рис. 8. Класс установления связей между таблицами

Материал, иллюстрирующий этапы работы программного комплекса, оформлен в виде приложений в диссертации.

Полученные результаты суммируются в заключительном параграфе. Основным преимуществом данного приложения является возможность моделирования рекурсивных наборов данных и автоматическое представление полученного описания для выбранного типа СУБД.

3. ОСНОВНЫЕ РЕЗУЛЬТАТЫ РАБОТЫ

Предлагаемые алгоритмы обладают достаточной универсальностью и могут быть использованы при проектировании различных предметных областей, содержащих рекурсивные информационные структуры. Полученные во второй главе результаты, могут являться основой для создания промышленных систем управления базами данных и базами знаний с поддержкой рекурсии в качестве специализированного программного обеспечения.

1. На основании исследований, проведенных в диссертации, установлено, что подход, основанный на расширенной теории реляционной алгебры Кодда, обеспечивает работу с рекурсивными зависимостями. Установлено также, что этот подход, характеризуется полиномиальной (не более третьей степени по описанию схемы) вычислительной сложностью обработки.
2. Сформулированная и доказанная теорема о замкнутости расширенной реляционной алгебры Кодда, предоставляет возможность выполнения запросов на обобщенной таблице, состоящей из вложенных списков. Теорема положена в основу предлагаемого подхода.

3. Предлагаемый формализм обработки вложенных рекурсивных данных посредством двухленточных конечных автоматов позволяет избежать повторной обработки рекурсивных ссылок.
4. Разработанный программный комплекс позволяет решить задачу проектирования схем баз данных с рекурсивно-определенными таблицами. Комплекс может использоваться для упрощения включения рекурсивных таблиц в существующие СУБД.
5. Предложенные формализмы, алгоритмы обработки и разработанный программный комплекс использованы при проектировании предметной области «Информационная система менеджмента качества кафедры» в отделе Менеджмента качества Томского политехнического университета, при проектировании и модификации баз данных метеорологических параметров атмосферы в системах «ТОР-станция» и «Аэрозольная станция» в Институте оптики атмосферы СО РАН (г. Томск) и при разработке программного комплекса «Управление основным производством» для ОАО «ТЭЛЗ» г. Томск, а также могут быть использованы при проектировании схемы базы данных «Профили компетентности специалистов» в компании «ЭлеСи» (г. Томск).

СПИСОК ПУБЛИКАЦИЙ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Govorushko, V. V. (Sokolova, V. V.) Expansion of Codd algebra operations with recursive objects / V. V. Govorushko, V. B. Novoseltsev // «KORUS 2004»: Proc. of 8th Korea-Russia International Symposium on Science and Technology, 26 june – 3 july, 2004. / TPU, – Tomsk, 2004. – P. 39-42.
2. Говорушко, В. В. (Соколова, В. В.) Расширение алгебры Кодда операциями с рекурсивными объектами / В. В. Говорушко, В. Б. Новосельцев // Вестник Томского государственного университета. – 2004. № 284, серия «Математика. Кибернетика. Информатика». – С. 18-21.
3. Соколова, В. В. Использование рекурсивных структур данных при построении сложных организационных систем // «Интеллектуальные технологии в образовании, экономике и управлении-2004»: Труды международной научно-практической конференции, 2004 г. / ВГУ, – Воронеж, 2004. – С. 175-177.
4. Соколова, В. В. Моделирование рекурсивных структур в реляционных базах данных / В. Б. Новосельцев, В. В. Соколова // «Информационные технологии и математическое моделирование (ИТММ-2004)»: Труды III Всероссийской научно-практической конференции, 11 – 12 декабря 2004 г. / Филиал КемГУ, – Анжеро-Судженск, 2004. – С. 107-109.
5. Соколова, В. В. Моделирование и реализация рекурсивных структур / В. Б. Новосельцев, В. В. Соколова // «Современные средства и системы автоматизации»: Труды V научно-практической конференции, 21 – 22 октября 2004 г. / ТУСУР, – Томск, 2004. – С. 133-136.
6. Соколова, В. В. Применение операций с рекурсивными объектами в реляционной алгебре Кодда // «Инфотелекоммуникационные технологии в

науке, производстве и образовании»: Труды I международной научно-технической конференции, 19 – 20 декабря 2004 г. / СевКавГТУ, – Ставрополь, 2004. – С. 494-498.
7. Соколова, В. В. Моделирование рекурсивных структур в базах данных конечными автоматами / В. Б. Новосельцев, В. В. Соколова // «Ломоносов 2005»: Тезисы докладов международной конференции студентов и аспирантов по фундаментальным наукам, 12–15 апреля 2005 г. / МГУ им. М. В. Ломоносова, – Москва, 2005. – С. 68-69.
8. Соколова, В. В. Внедрение системы менеджмента качества на кафедре вуза / В. Б. Новосельцев, В. В. Соколова // «IT-инновации в образовании»: Труды Всероссийской научно-практической конференции, 27 июня – 1 июля 2005 г. / ПетрГУ, – Петрозаводск, 2005. – С. 218-221.
9. Соколова, В. В. Моделирование рекурсивных структур конечными автоматами / В. Б. Новосельцев, В. В. Соколова // «Информационные и математические технологии в науке, технике и образовании»: Труды X Байкальской Всероссийской конференции, 12–19 июля 2005 г. / ИСЭМ СО РАН, – Северобайкальск, 2005. – С. 128-134.
10. Соколова, В. В. Таксономия рекурсивных зависимостей // «Молодежь и современные информационные технологии»: Труды III Всероссийской научно-практической конференции студентов, 15 – 17 февраля 2005 г. / ТПУ, – Томск, 2005. – С. 137 -138.
11. Соколова, В. В. К вопросу об информационной поддержке системы менеджмента качества / Е. И. Громаков, В. В. Соколова // «Качество высшего образования и подготовки специалистов к профессиональной деятельности»: Труды международного симпозиума, 9 – 11 ноября 2006 г. / Московский государственный институт радиотехники, электроники и автоматики, – Москва, 2006. – С. 272- 275.
12. Соколова, В. В. Применение автоматного подхода при обработке рекурсивных структур данных / В. Б. Новосельцев, В. В. Соколова // «Фундаментальные проблемы новых технологий в 3-ем тысячелетии»: Труды III Всероссийской конференции молодых ученых в рамках Российского научного форума с международным участием «Демидовские Чтения», 3 – 6 марта 2006 г. / ИОА СО РАН, – Томск, 2006. – С. 674-677.
13. Соколова, В. В. Автоматный подход к организации многомерных рекурсивных СУБД // «Молодежь и современные информационные технологии»: труды IV Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых, 28 февраля – 2 марта 2006 г. / ТПУ, – Томск, 2006. – С. 220-221.
14. Соколова, В. В. Система поддержки принятия решений руководителя кафедры / А. Р. Вахитов, В. В. Соколова // «Молодежь и современные информационные технологии»: Труды IV Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых, 28 февраля – 2 марта 2006 г. / ТПУ, – Томск, 2006. – С. 186-187.

15. Соколова, В. В. Информационная система поддержки менеджмента качества / Е. И. Громаков, В. Б. Новосельцев, В. В. Соколова // «Инженерное образование и наука в мировом пространстве»: Труды международной конференции GEER, 1 – 2 июня 2006 г. / ТПУ, – Томск, 2006. – С. 371-372.
16. Соколова, В. В. Информационные технологии в системах поддержки принятия решений / А. Р. Вахитов, В. В. Соколова // «Математика, информатика, управление» (МИУ-06): Труды Всероссийской конференции с международным участием, 8 – 12 июля 2006 г. / БГУ, – Улан-Удэ, 2006. – С. 38-42.
17. Соколова, В. В. Система поддержки принятия решений для менеджмента качества / А. Р. Вахитов, В. В. Соколова // «Энергия молодых – экономике России»: труды VII Международной научно-практической конференции студентов и молодых ученых, 17 – 21 марта 2006 г. / ТПУ, – Томск, 2006. – С. 401-402.
18. Соколова, В. В. Обработка рекурсивных данных конечными автоматами / В. Б. Новосельцев, В. В. Соколова // Известия Томского политехнического университета. – 2006. Т. 309, №7. – С. 130-134.
19. Соколова, В. В. Моделирование рекурсивных структур в базах данных / В. Б. Новосельцев, В. В. Соколова // «INTERMATIC – 2006»: материалы IV Международной научно-технической конференции «Фундаментальные проблемы радиоэлектронного приборостроения», 14 – 18 ноября 2006 г. / МИРЭА, – Москва, 2006. – С. 187-190.