НАЦИОНАЛЬНАЯ АКАДЕМИЯ НАУК УКРАИНЫ
Институт кибернетики им. В.М. Глушкова НАН Украины

На правах рукописи

Панченко Борис Евгеньевич

УДК 004.652

Каркасная модель данных и ее применение для разработки и внедрения CASE-средств и приложений

01.05.03 - математическое и программное обеспечение
вычислительных машин и систем

Диссертация на соискание ученой степени
доктора физико-математических наук

Научные консультанты:

Перевозчикова Ольга Леонидовна

член-корреспондент НАН Украины, доктор физико-математических наук, профессор

Летичевский Александр Адольфович
академик НАН Украины, доктор физико-математических наук, профессор

Киев – 2013

СОДЕРЖАНИЕ

СПИСОК УСЛОВНЫХ ОБОЗНАЧЕНИЙ

1НФ – первая нормальная форма

2НФ – вторая нормальная форма

3НФ – третья нормальная форма

4НФ – четвертая нормальная форма

5НФ – пятая нормальная форма

АИС - автоматизированная информационная система

БД – база данных

ВФЗ – взаимная функциональная зависимость

ДЗ – декартова зависимость

ДКНФ – доменно-ключевая нормальная форма

ЗПС – зависимость проекции-соединения

КМД – каркасная модель данных

МЗ – многозначная зависимость

ММД – многомерная модель данных

МХД – многомерное хранилище данных

НФ – нормальная форма

НФБК – нормальная форма Бойса-Кодда

ФЗ – функциональная зависимость

ООБД – объектно-ориентированная база данных

ООМД- объектно-ориентированная модель данных

ПО - программное обеспечение

ПрО – предметная область

ПТС – перемещаемая телевизионная студия

РБД – реляционная база данных

РК - реляционный каркас

РМД – реляционная модель данных

СИУ – система интегральных уравнений

СЛАУ – система линейных алгебраических уравнений

ТЗ – техническое задание

ХД – хранилища данных

CASE – Computer Aided Software Engineering

DD – domain dependence

KD – key dependence

MPEG – movie picture

OLAP – Online Analytical Processing

OLTP – Online Transaction Processing

SQL – Structure Query Language

SWS – Server Workstation Systems

SDI – Serial Digital Interface

3G-SDI – 3 gigabit serial digital interface

HD-SDI – high definition serial digital interface

СПИСОК БОЗНАЧЕНИЙ ЦИТИРУЕМЫХ ОПРЕДЕЛЕНИЙ И ТЕОРЕМ

F – Р. Фейджин [377, 378]

K – А.Г. Курош [144]

V – В.С. Выхованец [47]

U – В.А. Успенский [301]

ВВЕДЕНИЕ

Актуальность темы. Проблема анализа предметной области (ПрО) для ее концептуального и логического моделирования, обеспечивающего в дальнейшем проектировании высокую степень модифицируемости схемы реляционной базы данных (БД) и автоматизацию генерации приложений, занимает важное место в развитии современной науки. Для проектирования больших программных систем, систем управления, электронных устройств и т.п. решение этой проблемы имеет большое значение. До настоящего времени не было предложено универсального шаблона, позволяющего синтезировать в произвольной ПрО схемы БД, соответствующие безаномальной доменно-ключевой нормальной форме (ДКНФ). В дальнейшем говоря о БД, понимается только реляционная БД, если это не оговорено отдельно.

При работе с низко модифицируемыми схемами БД, основанными на низко нормализованных формах, не выше 3-й (3НФ), увеличивается число ошибок, повышается вероятность внесения некорректных данных – нарушается их целостность, появляются избыточные данные, увеличивается объем памяти, выделяемых под хранимые данные, неверно обрабатываются запросы, снижается итоговая производительность системы и т.д. Например, проблема модификации возникает при решении задач по развитию схемы БД в процессе эксплуатации, ведь обновление требует значительных затрат. Поэтому важной является разработка такого метода проектирования схем БД, которая обеспечивала бы выполнение большинства запросов пользователей с помощью операций индексного выбора, а не с использованием ресурсоемких соединений.

Все сказанное выше и обосновывает актуальность проблемы, исследуемой в настоящей диссертационной работе.

Проблемам формальных преобразований алгоритмов и их анализа на основе известных моделей посвящены роботы В.М. Глушкова [62], И.В. Сергиенко [272], А.А. Летичевского [149, 150], Е.Л. Ющенко [62], Ю.В. Капитоновой [105], Г.Е. Цейтлина [62, 321], Ф.И. Андона [5], В.Н. Редько [263], Д.Б. Буя [36], А.Е. Дорошенко [90], А.И. Провотаря [272].

Исследованиям в области моделирования предметной области - работы Г. Буча [38-40], К. Гейна [58], Э. Йордона [516], Д. Румбаха [483], Т. Сарсона [58], И. Якобсона [420], А.М. Вендрова [45], Г.Н. Калянова [108-110]. Исследования моделей данных и методов автоматизированного проектирования приложений содержатся в работах Т. Тиори [291], Дж. Фрай [291], Дж.-Д. Смит [276, 495], Ф. Бернштейна [349, 350], В.О. Гречко [70, 394], Ю.А. Пергаменцева [238], О.Л. Перевозчиковой [239-241], В.Г. Тульчинського [241, 295, 394]. Построение алгоритмов безаномальных схем БД предложены в работах Д. Кренке [131], Р. Фейджина [367, 377, 378], А. Алтайбек [2, 294], А.О. Голосова [64], У.А. Тукеєва [294], А.Ю. Филиповича [306]. Разработке семантических моделей анализа ПрО посвящены работы С. Абитбоула [337], Р. Кодда [361, 362], Я. Паридаенса [465, 466], Д. Сови [496], В. Харинарайна [404], П. Чена [359], С. Шапиро [486,487], А.М. Бабанова [12, 13], Д.Б. Буя [36], Е.А. Гришенкова [74], В.С. Выхованца [47], А.В. Замулина [94-96], Б.И. Плоткина [245], М.Ш. Цаленко [319], А.Л. Яловца [335]. Разработке онтологических моделей анализа ПрО – работы Т. Грубера [396-398], Н. Гуарино [399, 400], Д. Макгуиннесс [477, 460], Н. Ной [457-460], И.Л. Артемьевой [6], Г.Г. Белоногова [25], О.О. Варламова [43], Т.А. Гавриловой [50], Н.Н. Глибовца [60], Л.А. Калиниченко [106, 107], А.С. Клещова [6], А.В. Палагина [200] С.Л. Крывого [200]. Исследованиям многомерных хранилищ и темпоральных данных – работы Е. Баралиса [345], Д. Грея [371, 395], У. Инмона [414, 415], Р. Кимбалла [423-425], Т. Педерсена [469, 470], Р. Снодграсса [448], В. Харинарайна [404], М.Р. Когаловського [119-121], В.В. Пасичника [236], Н.Б. Шаховської [236]. Исследованиям денормализованных схем БД – работы Ф. Бреха [433], З. Лакроикса [433], К. Малинса [452,453], М. Пуле [258], С. Чаудхари [324, 325], Г. Хансена [310], О.Б. Кунгурцева [139-141], С.Л. Зиноватной [139-141]. Синтезу логических моделей объектно-ориентированных БД – работы М. Стоунбрейкера [500, 501], В.М. Линькова [166], А.М. Копейкина [127].

В обзоре использованной литературы по теме диссертации отмечены основные совпадения известных результатов и результатов диссертанта. Это

модель «сущность-связь» Чена (1976 г.) [359], реляционная модель высказываний Смитов (1977 г.) [276, 495], декартова зависимость Паридаенса (1979 г.) [465], многоместные предикаты Белоногова (1983 г.) [25], материализованные представления Блейкли-Ларсона (1986 г.) [354], денормализация Малинса (1992 р.) [452], пост-агрегация Грея (1995 р.) [395], многомерная решетка отношений Харинарайяна (1996 г.) [404], булеанные семантические запросы Абитбоула (1996 г.) [337], многоарные ключи Кимбалла (1996 г.) [423], реляционно-объектные БД Дейта (2000 г.) [368], модифицируемость схем БД и эволюционирующие приложения Варламова (2001 г.) [43], булеанная модель систем Степанова (2004 г.) [283], алгебра и исчисление понятий Выхованца (2004 г.) [47], семантически значимые отображения Бабанова (2004 г.) [12], ER-подход к синтезу ДКНФ Алтайбек (2008 г.) [2], фоновая агрегация Бадмаевой (2009 г.) [15], динамический изоморфизм Зинченко (2010 г.) [92, 98], семантический шаблон Гришенкова (2010 г.) [74].

Отмечено, что исходя из публикаций 1992–1994 гг. [222-225, 227], а также соответствующих актов внедрения, которые подтверждают промышленную применимость инструментального средства, а значит и каркасной модели данных (КМД), которая лежит в его основе, диссертант обладает некоторым приоритетом в указанных результатах.

Таким образом, диссертационная работа в области моделирования данных является развитием работ С. Абитбоула, Д. Грея, К. Малинса, Р. Кимбалла, Я. Паридаенса, Дж.-Д. Смит, Р. Фейджина, В. Харинараяна, П. Чена, А. Алтайбек, А.М. Бабанова, Г.Г. Белоногова, О.О. Варламова, Е.А. Гришенкова, О.Л. Перевозчиковой. А в области разработки программных приложений, моделирующих поведение объектов механики сплошных сред – работ А.Ф. Верланя, А.Н. Гузя, Ю.Г. Кривоноса, В.Д. Кубенко, А.М. Назаренко, З.Т. Назарчука, В.В. Панасюка, М.П. Саврука, Л.А. Фильштинського, Н.А. Шульги.

Связь работы с научными программами, планами, темами. Исследования, представленные в диссертации, выполнены в соответствии с рабочими планами научно-исследовательских работ в отделе автоматизации программирования Института кибернетики имени В.М. Глушкова

Национальной академии наук Украины, где автор был исполнителем, в рамках следующих тем:

1. «Высокопроизводительные методы анализа и спецификации пространств атрибутов предметной области для организации вычислений» (государственный регистрационный номер 0107U000800 ВФ.145.09.11, 2007 – 2011).

2. «Разработка метода интеллектуализации информационных технологий для оптимизации параллельных вычислений и верификации дедуктивными методами параллельных программ, которые масштабируются» (государственный регистрационный номер 0107U003570 ВФ.105-145.07.11, 2007 – 2011).

3. «Создания отечественного энергоэффективного суперкомпьютера для решения сложных научно-технических задач и задач государственного управления» (государственный регистрационный номер 0112 U002720 ВК 145.17, 2012).

Цель и задача исследования. Основная цель работы – разработка на основе решетки отношений (реляционного каркаса, РК) новой концептуальной и логической модели данных как подмножества реляционной модели (РМД), анализ и разработка каркасного алгоритма построения безаномальных схем БД, разработка соответствующих CASE-оболочек, а также практическая апробация полученных результатов.

В диссертационной работе цель исследования достигается благодаря двум направлениям. Первый – в разработке методов проектирования высоко нормализованных и безаномальных схем БД и создании на этой основе новой модели данных. Второй – в разработке и внедрении инструментальных средств нового типа (CASE-оболочек), схем БД и приложений традиционного корпоративного типа, а также других приложений, прямо не связанных с БД. Таким образом, второе направление – это проверка на практике результатов теоретических исследований.

Объект исследования – нормализация схем БД.

Предмет исследования – это новая концептуальная и логическая модели данных, которые позволяют единым автоматизированным методом

осуществлять декомпозицию ПрО и проектирование высоко-нормализованных модифицируемых схем и приложений БД, а также других информационных систем.

Метод исследования. Теоретическую основу выполненных исследований составляют: теория решеток, булева алгебра, алгебра и исчисление предикатов, теория алгоритмов, теория рекурсивных функций, теория нормализации и теория аномалий БД, метод Чена, теория интегральных уравнений.

Научная новизна полученных результатов. В диссертационной работе в процессе исследования получены такие новые научные результаты:

1) решена классическая проблема РМД об алгоритме синтеза безаномальных схем БД для произвольных ПрО (считается нерешенной);

   разработаны:

2) теоретические основы КМД, которое базируется на РК (решетке отношений) – универсальном шаблоне для проектирования высоко-нормализованных схем БД;
3) каркасный метод проектирования схем БД и приложений, причем не только для БД;
4) алгоритм минимизации операций соединения, вместо которых используются фоновые отношения для моделирования связей в реальном времени;

   исследованы и предложены:

5) каркасный метод моделирования абстракций (обобщений и ассоциаций), чем обеспечивается моделирования связей без выхода за границы непротиворечивости (вследствие, например, необоснованного расширения РМД и т.д.);
6) каркасный метод моделирования рекурсивных связей;
7) каркасный метод моделирования темпоральных данных;
8) замена ER-диаграммы П. Чена R-диаграммой, где именно связи являются единственной категорией;

9) экспериментальным путем подтверждено существование CASE-технологии, которая основана на безлистинговом принципе синтеза приложений.

Практическая ценность. Диссертационная работа имеет как теоретический, так и прикладной характер.

Разработанные алгоритмы позволяют использовать формально обоснованную каркасную R-диаграмму вместо слабо-формализованной ER-диаграммы П. Чена, проектировать БД в безаномальной ДКНФ-схеме, минимизировать операции соединения, заменив их предварительно синтезированными фоновыми отношениями-связями (материализованными представлениями), формально моделировать абстракции, рекурсивные и темпоральные данные, а также разрабатывать CASE-оболочки, которые синтезируют приложения по безлистинговому принципу.

Предложенный подход может использоваться в разных направлениях современного программирования.

Личный вклад соискателя. Шестнадцать работ опубликовано без соавторов. В коллективном учебном пособии [113] автору принадлежат разделы 2 – 6 и список литературы. В опубликованных в соавторстве научных работах соискателю принадлежит: в [222, 223] – схемы ПрО, общая формализация, обзор литературы; в [185 – 188, 230] – постановка задач, численная реализация, схема вычислений, анализ ПрО, анализ полученных результатов, разработка программных приложений, в [232 – 234] – постановки задач, формализация модели, формулировка теорем, общий ход доказательства теорем; в [221] постановка задач, формализация модели, каркасная схема ПрО, общая концепция инструментального средства, алгоритм фонового взаимодействия таблиц, алгоритм инициализации атрибутов в реальном времени в соответствии с ограничениями целостности, алгоритм фоновых групповых функций, сопровождение внедрений; в [229] – постановка задач, каркасная схема ПрО, постановка численных экспериментов, анализ результатов; в [231] – постановка задач, формализация и каркасная схема ПрО, общая концепция способов коммутации; в [225] – разработка общей концепции инструментального

средства, его схемы, алгоритм фонового взаимодействия таблиц, алгоритм on-line-инициализации атрибутов, соответствующий ограничениям целостности, алгоритм фоновых групповых функций.

Апробация результатов диссертационной работы. Основные результаты диссертации были апробированы на следующих международных конференциях: II Международная конференция «Технология программирования 90-х» (Киев, 1992); I Международная конференция «Компьютерные технологи в промышленности» (Киев, Общество «Знание», 4–6 октября 1994); V международная конференция «Глобальное информационное пространство: ресурсы, технологии, инновации» (Киев, УкрИНТЭИ, 22–23 октября 1998); XI международная конференция «Системный анализ и информационные технологии» (SAIT-2009) (Киев, КПИ, 26–30 мая 2009); IV межвузовская конференция «Информатика, математика, механика» (Сумы, СумДУ, 21–24 апреля 2009); III международная конференция «Вычислительная и прикладная математика» (Киев, КПИ, 11–12 сентября 2009); II международная научно-техническая конференция «Интеллектуальные системы в промышленности и образовании» (Сумы, СумДУ, 3–5 декабря 2009); XIII Международная научная конференция им. академика Михаила Кравчука (Киев, КПИ, 13 мая 2010); Юбилейная международная научно-практическая конференция «Распределенные компьютерные системы РКС-2010» (Киев, КПИ, 6–8 апреля 2010); I Всеукраинская научно-практическая конференция «Системный анализ. Информатика. Управление» (Запорожье, 4–5 марта 2010); XII Международная конференция «Системный анализ и информационные технологии» (SAIT-2010) (Киев, КПИ, 25–29 мая 2010); V Межвузовская конференция «Информатика, математика, механика» (Сумы, СумДУ, 21–24 апреля 2010); III Международная конференция «Инновационное развитие общества в условиях кросс-культурных взаимодействий» (Сумы, 26–29 апреля 2010); VII Международная конференция «Теоретические и прикладные аспекты построения программных систем» (TAAPSD-2010), (Киев, КГУ, 4–8 октября 2010); XIII международная конференция «Системный анализ и информационные

технологии» (SAIT-2011) (Киев, КПИ, 23–28 мая 2011); Международная конференция «Моделирование и исследование устойчивости динамических систем» (DSMSI-2011) (Киев, КГУ, 25–27 Мая 2011); IIX Международная конференция «Теоретические и прикладные аспекты построения программных систем» (TAAPSD-2011), (Киев-Ялта, КГУ, 19–23 сентября 2011); Международная конференция «Высокопродуктивные вычисления» (HPC-UA'2011) (Киев, КПИ, 12–14 октября 2011); IX Международная конференция «Теоретические и прикладные аспекты построения программных систем» (TAAPSD-2012), (Киев, КГУ, 3–7 декабря 2012); Международная конференция, посвященная 90-летию акад. В.М. Глушкова «Сучасна інформатика: проблеми, досягнення та перспективи розвитку».

В полном объеме диссертация докладывалась на научных семинарах: в Институте кибернетики имени В.М. Глушкова НАН Украины, Институте прикладной математики и механики НАН Украины (г. Донецк), Киевском государственном университете им. Т.Г. Шевченко, Национальном университете «Киево-Могилянская академия», Одесском государственном университете, Одесском политехническом университете, Национальном университете «Львовская политехника», Сумском государственном университете.

Публикации. Результаты диссертационных исследований опубликованы в 60 научных работах. Из них 21 статья – в научных изданиях из Перечня научных специализированных изданий Украины, в том числе 9 статей в изданиях, которые включены в международные научно-метрические базы (в частности, базу «Скопус»); 6 статей в других научных изданиях, которые до 2009 года входили в Перечень научных специализированных изданий Украины; одно учебное пособие с грифом МОН Украины; 3 патента Украины, 1 свидетельство на авторские права ПО; 28 докладов на международных и межвузовских конференциях.

Структура и объем диссертации. Диссертационная работа состоит из вступления, семи разделов, разбитых на подразделы, выводов, списка использованных источников (517 ссылок) и приложения. Работа содержит 33 рисунка. Объем основной части диссертации – 303 страницы.

РАЗДЕЛ 1
РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ

1.1. Введение к первому разделу

Нормализованная схема реляционной БД (в дальнейшем – просто схема БД), полученная в результате проектирования, должна, прежде всего, удовлетворять двум критериям – кроссплатформность и интероперабельность [239, 240]. В этих работах под «кроссплатформностью» понимается независимость приложений, как от аппаратного, так и от операционного программного обеспечения: «написанное однажды выполняется везде». А под «интероперабельностью» понимается способность приложений, интерфейсы которых полностью открыты, взаимодействовать и функционировать с другими приложениями без каких-либо ограничений доступа и реализации. А также способность приложения выполнять набор функций, определённых в его внешнем описании и удовлетворяющих заданным или подразумеваемым потребностям пользователей. Эти свойства сводятся к тому, что разные приложения БД, работающие на разных платформах, интегрируются в одно приложение, а БД при этом воспринимается как единое целое. Обеспечит эту потребность унификация схем для разных ПрО. При этом понимается, что ПрО обладает произвольными структурой и объемом.

Широко известны традиционные алгоритмы проектирования схем БД, которые базируются на классических технологиях [359, 361, 445]. Но эти методы не решают вопроса получения универсальной и модифицируемой схемы БД, а создают ее зависимой от семантики ПрО.

Наиболее важной частью любой информационной системы является ее БД. Синтез схемы БД – это многокритериальный процесс, основанный на анализе ПрО [239]. Основными критериями оценки модели БД являются «целостность данных», «отсутствие аномалий» и «отсутствие избыточности данных» [361]. Однако со временем любой программный продукт подвержен изменениям. И схему БД, не отвечающую новым требованиям,

необходимо подвергать обновлениям. Как правило, изменение схемы БД приводит к серьезным затруднениям и существенно повышает стоимость сопровождения. Следовательно, частое перепроектирование схемы БД приводит к реинжинирингу всей СУБД, что нежелательно для пользователей ввиду больших временных и финансовых затрат.

Одним из решений указанной проблемы является синтез безаномальной схемы БД [208]. Опыт разработок позволяет утверждать, что эксплуатационные характеристики приложений (скорость доступа к данным, скорость внесения модификаций в схему БД и в само приложение в динамическом режиме, кроссплатформность и интероперабельность приложений) при этом существенно повышаются.

Анализ ПрО, основанный на классическом методе декомпозиции [361], приводит к схемам БД, существенно отличающимся одна от другой в зависимости от специфики ПрО. Более того, субъективный фактор проектирования является решающим. Это приводит к большой зависимости приложения от всевозможных модификаций.

Однако, хотя классическая РМД основана на очень жестком алгоритме, слабо поддающемся унификации, идея универсальности не нова. Ее проявление состоит в алгоритме синтеза схемы отношения, предложенном Ф.А. Бернштейном [349, 350]. Эта же идея есть как в выводах работы П.П. Чена [359], так и в самом ее названии.

Поэтому разработка новой, более универсальной методики анализа ПрО, позволяющей синтезировать такие безаномальные схемы БД, которые обладали бы при этом еще и высокой степенью модифицируемости и подобия для разных ПрО, является важной задачей.

1.2. Обзор литературы по теме диссертации

Совершенствование методов проектирования интегрированных корпоративных БД актуально сегодня по причине качественного повторения проблемной ситуации 60–70 годов с разрозненностью данных применительно

к разнообразным СУБД – источникам интегрирования данных по технологии Data Warehouse процессами ETL (извлечение, преобразование, загрузка).

Явное выделение концептуального уровня абстракции данных придало данным смысл целостного корпоративного ресурса, отделенного от программ их обработки. Этот уровень объединяет разнообразие взглядов на данные пользователей, прикладных программистов и разнообразие решений физического уровня.

Однако, несмотря на достижения ER-моделирования [359] и теории нормализации БД [361], за 40 последующих лет никому не удалось построить единую нормализованную корпоративную БД, хотя цель такая ставилась. В дальнейшем изложении некие концепты ПрО (сущности, объекты, факты и т.п.) будем называть сущностями до того момента, пока не дадим новое определение. Понятие «сущность» традиционно используют в концепциях, близких к РМД, а понятие «объект» – ООБД. Поскольку целью настоящего исследования является обобщающая модель, в следующем разделе будет определена категория «сущность-объект».

Разрозненность данных. Подходы концептуального проектирования БД для АИС качественно мало изменились с времени формирования концептуальных положений [85, 291, 368], т.е. с 70-х годов прошлого века. В основу положено осмысление накопленного к середине 60-х годов опыта автоматизированной обработки данных файловыми системами, широко распространившимися в различных отраслях деятельности. При попытках получения сводной отчетности обнаружились недостатки файловых систем: трудности сбора данных из разных систем, отсутствие целостности данных, дублирование данных и усилий по их сбору и обработке и т.п.

С точки зрения корпорации в целом, требовалось повышать эффективность автоматизации обработки данных [309], роль автоматизации для управления производством осмыслена в работах С. Бира [27, 28], системный подход - в работе С. Оптнера [196].

Все вышеизложенное повторяется в многочисленных монографиях и учебниках, в части проблем проектирования структур данных [33, 44, 71, 85, 111, 113, 121, 159, 165, 236, 268, 291, 300, 309, 310, 317, 322, 493].

Концептуальное проектирование модели данных сначала было надежнее реализовать в ручном режиме, как и выверку работы сложной программы. Вместе с концептуальным проектированием зародилась и автоматизация этого процесса [309], получившая развитие в работах [125, 173]. Близкими по тематике являются работы по распределению ресурсов авторов [226] при использовании комбинаторных, эвристических методов и методов исследования операций [10, 23, 59, 81, 172, 250, 264].

Проблемная ситуация. С 2007 года в литературе по компьютерным наукам возродилась дискуссия по поводу пересмотра принципов организации обработки данных и построения СУБД [135, 308, 383, 419, 468, 500, 517]. При этом 2010 ожидался годом «эпохи перемен в технологиях БД» [135]. Дискуссия базируется на диспропорциях развития оборудования и методологии его применения, утверждениях типа «существующие технологии создавались для других данных, в другое время, для других целей» [500].

В частности, критика делает упор на необходимости сближения модели данных с вычислениями [500]. Такой подход применен и в настоящей работе. Раздельное рассмотрение моделей приводит к отрыву системы координат (ключевые атрибуты, определяющие сущности) от задаваемых в ней данных (числовые атрибуты). При этом поддержание вычислительной целостности целиком передается на усмотрение разработчика программ, то есть убирается с концептуального на физический уровень.

Ведение БД в режиме коллективного взаимодействия и действия по регистрации событий возможны только на новом витке технологического развития. Использование соответствующих представлений также позволяет четко разграничить полномочия различных лиц, работающих с БД.

Соответствующие представления позволяют описать «видение» БД разными лицами, работающими с ней:

- внешнее представление - представление специалиста ПрО (пользователя);
- внешнее представление и логическая модель - представление прикладного программиста, разрабатывающего приложение для пользователя;
- логическая модель и внутреннее представление - представление системного программиста, администрирующего БД.

В соответствии с этим были развиты два подхода концептуального проектирования: сущностный и объектно-ориентированный. Последний получил развитие в виде UML языка [22, 38, 45, 73-77], который рекомендуется применять для описания модели перед реализацией программ.

Понимание сущности подходов концептуального проектирования неотделимо от абстрактных понятий агрегации и обобщения [47, 55, 291]. Последовательное применений абстракций агрегации и/или обобщения позволяет получать иерархии понятий (сущностей, объектов и т.п.).

Получили широкое распространение так называемые инфологические (состав и логика использования данных) или, другими словами, семантические (смысл используемых данных) модели данных. В настоящий момент среди инфологических моделей наиболее распространена «сущность-связь» (ER), предложенная в [359], ставшая основным средством формализации требований пользователей [119, 291]. Предполагается, что ПрО состоит из сущностей с их свойствами (атрибутами), и связей, которые описывают отношения между сущностями. Атрибуты делятся на ключевые (набор которых определяет сущность) и простые .

Однако при таком подходе присутствует значительный субъективизм при выборе: один и тот же элемент данных может быть представлен и атрибутом, и сущностью, и связью. В связи с этим авторы [291] допускают возможность концептуального описания ПрО без деления элементов данных на сущности, атрибуты и связи.

В настоящей работе предлагается иной подход, отличающийся от ER-моделирования более строгим определением сущностей и атрибутов.

Теория нормализации БД. В основе разработки ER-подхода лежат идеи, которые во многом очень близки к идеям, послужившим Кодду [361] неформальной основой при создании им исходной формальной реляционной модели.

С ней связан язык SQL, обеспечивающий операции с данными на основе формализма реляционной теории [65, 66, 75, 160]. Следует заметить, что управление данными эффективно реализовывалось и до, и после появления языка SQL, например, язык Base SAS [485].

Кодд нашел подходящую формальную модель для определенных аспектов реального мира. В противоположность этому ER-модель не является формальной моделью. А проектирование БД не может быть завершено без применения формальных объектов и правил.

ER-модель не может заменить РМД, поскольку в ее рамках нет формальных механизмов, обеспечивающих выполнение основных операций обработки данных (какие были включены в состав РМД) и отсутствуют средства описания ограничений целостности данных. В то же время графические компоненты ER - модели полезны для представления «всей картины в целом».

Главная цель нормализации БД - устранение аномалий. В идеале при нормализации можно добиться того, что ни одно из отношений не было бы ниже 5НФ, а то и ДКНФ [131, 169, 361, 367, 377, 378, 368].

Процесс нормализации состоит из последовательного преобразования исходной БД к НФ, при этом каждая следующая НФ обязательно включает в себя предыдущую. Это позволяет разбить процесс на этапы и производить его однократно, не возвращаясь к предыдущим этапам. Всего в реляционной теории насчитывается 8 НФ.

Простое, но, тем не менее, формальное определение, например, 5НФ такое: это форма отношения, в которой устранены зависимости соединения.

В подавляющем большинстве случаев в приложениях, реализованных для промышленной эксплуатации, нормализованных таблиц уровня 5НФ не наблюдается. Не будет большим преувеличением сказать, что вряд ли найдется хоть один учебник, который недвусмысленно указал бы стажерам-проектировщикам, что НФ ниже 5-й в проекте схемы БД – это очень плохо.

В результате проведения нормализации БД удается добиться устранения (или, по крайней мере, серьезного сокращения) аномалий вставки, удаления, а также неуправляемой (нецелостной) избыточности данных [131, 169, 361, 367, 368]. Тут под управляемой (не аномальной) избыточностью понимаются строгое копирование в отношениях ключевых атрибутов, с помощью которых моделируются связи и ссылки на связи.

Как следствие этого, значительно сокращается вероятность появления противоречивых данных, облегчается администрирование базы и обновление информации в ней, сокращается объем дискового пространства.

Пределы применимости. Как и всякая формальная теория, теория нормализации имеет свои пределы применимости и ограничения. Они проявляют себя в обнаруживаемых на практике недостатках. На протяжении последних 20 лет никому так и не удалось построить теоретически обоснованную высоко нормализованную корпоративную БД. Хотя цели такие ставились и «иллюзии довлели умами» ИТ разработчиков [44]. На роль исключения претендует данное исследование.

К недостаткам теории нормализации можно отнести:

- сложность извлечения информации из нормализованной БД;
- сложность конструирования запросов;
- медленная работа запросов из-за большого числа операций соединений отношений;
- медленная работа запросов из-за сложных вычислений, особенно при использовании группировок и агрегатных функций (Sum, Max и т.п.);
- отсутствие теоретически обоснованного хранения агрегатных значений.

По этим причинам для увеличения скорости выборки данных и упрощения программирования запросов, нередко приходится идти на выборочную денормализацию базы [258, 452]. Это творческий и плохо формализуемый процесс. Авторы большинства учебных пособий утверждают, что денормализация - это искусство [71, 126, 268, 329, 331, 383].

Интеграция данных. На практике для решения проблемы интеграции данных было развито два технологических подхода: киоски данных (Data Marts или DM) [423, 424] и хранилище данных (Data Warehouse или DW) [414, 415], их сравнение широко обсуждалось [26, 151, 364, 373, 374, 434, 517].

DW основано на идее проектирования корпоративной аналитической БД, физическом разделении учетных и аналитических систем. Обработка данных также разделена на транзакционную (OLTP) и аналитическую (OLAP) части, различающиеся: объемами транзакций, оптимизацией на ввод или чтение, типом пользователей и др. Заполнение структуры DW основано на обширной ETL обработке, на массовых пересылках данных предельно большого объема.

При DM подходе выборочные подключения к СУБД источникам позволяют «на лету» формировать слой предметно ориентированных данных (Universe). Подход DM легок в реализации, но с ростом числа киосков данных быстро нарастает множество связей между киосками и источниками вместе с разногласиями в пользовательских представлениях.

Подходы DM и DW характеризуются как нисходящий и восходящий способы проектирования [373].

Однако на практике результаты их не стыкуются - вместо единого DW удается строить разрозненные склады детальных данных Detail Data Storage (DDS) для каждой предметной области (отраслевые модели) с ограниченным кругом задач [485].

Модель данных DDS может содержать сотни таблиц и тысячи показателей, объединение моделей для расширения круга задач на практике представляется очень сложной задачей.

Такое развитие свидетельствует о качественном воспроизводстве сложностей и недостатков систем, обусловленным теоретическим вакуумом в сфере концептуального моделирования данных. Причина – наличие непреодолимого семантического разрыва между методологией и практикой, между наукой и реалиями сферы БД.

Термин «семантический разрыв» в научный обиход ввел Г. Майерс [155]. Семантический разрыв был определен как мера различия принципов, лежащих в основе языков программирования высокого уровня, и тех принципов, которые определяют архитектуру ЭВМ. Однако этот фактор несложно продлить на более широкий круг проблем, трактуя семантический разрыв как меру различия разных принципов всей отрасли.

Накопленный опыт проектирования информационных систем показывает сложность и трудоемкость этого процесса, длительного во времени и требующего высокой квалификации участвующих в нем специалистов.

Недостатком известных методологий и технологий декомпозиции ПрО, затрудняющим решение проблемы, остается сохранение семантического (смыслового) разрыва между моделями различных уровней. Например, между высокоуровневой моделью ПрО и языком программирования, с помощью которого эта модель реализуется на этом языке; или между высокоуровневой моделью (спецификацией) и ее описанием на естественном языке, отражающем содержательные представления пользователей о ПрО [47].

В итоге, проектирование информационных систем до сих пор выполняется в основном на интуитивном уровне с применением неформализованных методов, основанных на искусстве разработчиков, их

практическом опыте, экспертных оценках и дорогостоящих экспериментальных проверках получаемых результатов.

Кроме того, в процессе жизненного цикла информационная система подлежит постоянному изменению в соответствии с изменяющимися потребностями пользователей и развитием их представлений о предметной области, что еще более усложняет разработку и сопровождение таких систем.

Что же касается использования методики онтологий [26, 157, 216, 221], т.е. построения параметризованных толковых словарей ПрО, достаточный обзор способов и методик изложен в [216]. Во всех этих подходах не рассматривается алгоритм, позволяющий автоматизировано создавать гибкие, максимально быстро модифицируемые схемы БД. Тем более, на базе текстового или даже звукового описания ПрО на естественном языке.

Основные совпадения известных результатов и результатов диссертанта. Как известно, до времени публикации [131] не существовало строго обоснованного алгоритма синтеза максимально нормализованной ДКНФ-схемы, что и указал Д.М. Кренке в [131]. Сегодня на эту роль претендует способ [204, 206], основанный на РК и КМД. Впервые это способ был официально предложен в 2001 году в [204]. Хотя начало работ по экспериментальному внедрению способа было зафиксировано в [222, 223, 225] в 1992-1994 гг., за 2 года до опубликования работ [337, 404, 423].

В 1996 году именно булеан связей («power-set») был выбран Сержем Абитбоулом в [337] как дополнительная алгебраическая операция, расширяющая РМД именно с целью ее «семантизации» («реляционной» формализации семантики языка SQL) и Венки Харинараяном как модель ХД в [404], названный там как «многомерная решетка». Позднее очень близкие решения были предложены также А.М. Бабановым [12] (2003 год, семантически значимые отображения) и Е.А Гришенковым [74] (2010 год, семантический шаблон). Однако для построения концептуальной и логической моделей данных и разработки серии инструментальных средств эта идея не была развита.

Отметим, что в монографии [283] также описана модель поведения систем [49, 239], основанная на множестве всех подмножеств связей. Однако автор монографии [283] в своих дальнейших работах этот метод не развивал.

Таким образом, при подготовке литературного обзора к диссертационному исследованию были обнаружены следующие совпадения результатов диссертанта с классическими и новыми решениями. Модель «сущность-связь» Чена (1976 г.) [359], реляционная модель высказываний Смитов (1977 г.) [276, 495], декартова зависимость Паридаенса (1979 г.) [465], многоместные предикаты Белоногова (1983 г.) [25], материализованные представления Блейкли-Ларсона (1986 г.) [354], денормализация Малинса (1992 р.) [452], пост-агрегация Грея (1995 р.) [395], многомерная решетка отношений Харинарайяна (1996 г.) [404], булеанные семантические запросы Абитбоула (1996 г.) [337], многоарные ключи Кимбалла (1996 г.) [423], реляционно-объектные БД Дейта (2000 г.) [368], модифицируемость схем БД и эволюционирующие приложения Варламова (2001 г.) [43], булеанная модель систем Степанова (2004 г.) [283], алгебра и исчисление понятий Выхованца (2004 г.) [47], семантически значимые отображения Бабанова (2004 г.) [12], ER-подход к синтезу ДКНФ Алтайбек (2008 г.) [2], фоновая агрегация Бадмаевой (2009 г.) [15], динамический изоморфизм Зинченко (2010 г.) [92, 98], семантический шаблон Гришенкова (2010 г.) [74].

В связи с тем, что в 1988 году (начало разработки) у диссертанта не существовало технологической возможности иметь доступ к литературным источникам, а также тем фактом, что ни один из указанных авторов не развил высказываемые идеи до нового инструментального средства, некоторые пересечения подходов определяются тут как совпадения, а не как заимствование.

1.3. Проблемы проектирования БД – постановка задач

Объединение СУБД, приложений БД, операционной системы и аппаратных средств в одну систему для информационного

обслуживания пользователей известно под названием система БД [291]. Одной из самых важных проблем, стоящих перед администратором БД, является наиболее эффективное использование этой системы. Решение этой проблемы включает [291]:

- требования пользователей;
- преобразование требований в эффективную схему БД;
- частота и способ перестройки схемы БД в соответствии с новыми и/или изменяющимися требованиями.

Процесс разработки схемы БД в соответствии с требованиями пользователей называется проектированием БД. Тут под проектированием будем понимать формирование логической схемы БД, которая поддерживается СУБД и описывает представление пользователя о данных, а также выбор физической структуры, которая включает представление данных или кодирование, методы доступа и физическое группирование (кластеризацию) данных.

1.3.1. Анализ ПрО

Формирование и анализ требований являются плохо формализованным и наименее изученным процессом, однако наиболее трудоемким и длительным по времени этапом в проектировании. Основной задачей является сбор требований, предъявляемых к содержанию и процессу обработки данных всеми известными и потенциальными пользователями БД. Анализ требований обеспечивает согласованность целей пользователей, а также согласованность их представлений об информационных потоках в ПрО.

Концептуальное проектирование - построение независимой от СУБД информационной структуры в доступной пользователю форме путем объединения информационных требований пользователей. Результат концептуального проектирования называется также концептуальной схемой, поскольку она является представлением точки зрения пользователей на ПрО и не зависит ни от программного

обеспечения СУБД, ни от технических решений, но реализуемой несколькими системами.

Проектирование логической схемы БД. Получение схемы БД как СУБД-ориентированного описания данных, выраженного в терминах языка описания данных.

Крайне важной является задача разработки такой новой или усовершенствования старой логической модели данных, которая возможно полно использовала концепции и для которой этап концептуального проектирования был максимально близким. В этом смысле ER-подход хотя и является повсеместно признанным, тем не менее, не удовлетворяет этому требованию. Перестроение концептуальных ER-схем при переводе их, например, на логическую схему БД в РМД, зачастую практически полностью меняет концепции. И соответственно нарушает интероперабельность и кроссплатформность приложений.

Первая проблема заключается в том, что может быть построено практически бесконечное число разных схем БД, удовлетворяющих одному и тому же множеству системных требований.

Вторая проблема состоит в том, что альтернативы чрезвычайно трудно поддаются оценке. Существует много признаков оптимальности, являющихся не измеряемыми свойствами, которые трудно выразить в количественном представлении или в виде целевой функции. Поэтому оценочные критерии принято делить на количественные и качественные.

Выпишем из [291] некоторые критерии. *Количественные*: время отклика на запрос; стоимость обновления; время, затраченное на создание приложения; стоимость реорганизации. А также *качественные*: гибкость; адаптивность; понимаемость проекта для новых пользователей; совместимость с другими системами; интероперабельность и кроссплатформность (возможность

конвертирования для использований в другой вычислительной среде); возможность членения или расширения структуры.

Одна из трудностей в оценке альтернатив заключается в том, что время действия различных критериев различно. Критерии эффективности обычно являются краткосрочными, поскольку они чрезвычайно чувствительны к проводимым изменениям, в то же время такие понятия, как конвертируемость и адаптивность, требуют более длительной перспективы для рассмотрения и менее чувствительны к кратковременным изменениям в ПрО.

Поэтому крайне важным свойством приложений является именно модифицируемость.

1.3.2. Концептуальное проектирование.

Этап концептуального проектирования связан с преобразованием разнообразных информационных требований пользователей в первоначальный проект БД. Результатом этого этапа является высокоуровневое представление информационных требований, например, такое как ER-диаграмма П. Чена [359]. Основным концептуальным носителем информации являются сущности, которые могут быть описаны атрибутами, детализирующими свойства сущности. Связи между сущностями отображают функциональные аспекты информации, представленной сущностями [359].

Существует несколько подходов к построению концептуальных моделей. Общим для всех подходов является набор из четырех основных шагов: определение сущностей (или объектов), определение атрибутов сущностей (или объектов), идентификация ключевых атрибутов сущностей (или объектов), определение связей между сущностями (или объектами).

Подход к концептуальному проектированию обычно предполагает, что рассматривается представление одного пользователя. Альтернативным является подход, в котором объединение

представлений выполняется как часть процесса концептуального проектирования [291].

Моделирование локальных представлений. Требования каждого пользователя анализируются и представляются в некоторой общей форме. Объекты и события, ассоциированные с представлением каждого пользователя, моделируются множеством сущностей, атрибутов и связей между сущностями.

Объединение локальных представлений. Требования отдельных пользователей, определенные в некоторой унифицированной форме, такой, как например, в виде ER-диаграммы [359]. При этом выявляются и устраняются противоречивые и избыточные данные.

Для целей проектирования БД используются абстрактные объекты или понятия, т.е. концепции. Целью концептуального проектирования является определение взаимосвязанной структуры абстрактных объектов.

Абстракция (лат. abstractio – отвлечение) определяется как один из основных процессов умственной деятельности человека, позволяющий мысленно вычленить и превратить в самостоятельный объект рассмотрения отдельные свойства, стороны, элементы или состояния предмета [171]. Различают уровни абстрагирования: абстрагирование, приводящее к образование простых понятий, и абстрагирование, результатом которого является образование составных понятий. В соответствии с этим существует два способа формализации абстракций: агрегация и обобщение. [47, 55]

Агрегация [291] формирует объект как связь между другими объектами из разных классов, каждый из которых не имеет общих признаков.

В математическом смысле понятие агрегации соответствует понятию декартова произведения.

Причем, в определении [291] допускается оговорка: если объект имеет общие признаки, то это носит случайный характер и не влияет на процесс агрегирования.

Рассмотрим несколько примеров, которые иллюстрируют идею агрегации. Например, связь между четырьмя сущностями ЧЕЛОВЕК, КОМНАТА, ГОСТИНИЦА и ГОРОД выражается через сущность БРОНИРОВАНИЕ. Этим самым выражается тот факт, что человек бронирует некоторый номер в некоторой гостинице некоторого города. Причем атрибутом этой связи будет некоторый отрезок времени - определенные даты начала и завершения бронирования. В этой конкретной агрегации наименования индивидуальных сущностей отбрасываются. И связь именуется как целое.

Агрегацию можно рассматривать и по-другому, как именную форму глагола в связи [291]. Например, БРОНИРОВАНИЕ является именной формой глагола «бронировать». Однако в речевой агрегации этот прием используется далеко не всегда. Так, например, для агрегации «преподаватель читает предмет в течение семестра», как правило, используют наименование КУРС. Видимо, считается, что этот термин подходит лучше, чем термин ЧТЕНИЕ.

Поэтому наименования агрегатных объектов в ПрО выбирается по разным критериям. И, как правило, непредсказуемо. Из практики проектирования известно, что именно агрегаты являются одним из наиболее проблемных описателей ПрО. Именно они порождают значительное многообразие формализмов и моделей ПрО, что, в свою очередь, приводит к значительному многообразию схем БД. Но для развития и сопровождения большинства промышленных приложений это недопустимое свойство, так как агрегаты заключают в себе семантику агрегации.

Таким образом, связь между объектами: «человек бронирует номер в гостинице города на определенные даты», «преподаватель читает предмет в

течение семестра», «автомашина перевозит груз от места загрузки к месту назначения» - это агрегатные объекты БРОНИРОВАНИЕ, КУРС (ЧТЕНИЕ), ПЕРЕВОЗКА (РЕЙС).

При этом любой компонент объекта может в свою очередь быть или атомарным (не агрегированным) объектом, или агрегатом. Например, ПРЕПОДАВАТЕЛЬ и ПРЕДМЕТ являются атомарными компонентами КУРСА, а компонент СТУДЕНТЫ на КУРСЕ для некоторых ПрО может интерпретироваться или как атомарный, или как агрегированный.

Таким образом, агрегированное понятие вводится пользователем в языковый обиход и делопроизводство для того, чтобы в процессе функционирования ПрО экономить ресурсы некомпьютерной обработки информационных потоков – документооборота, устных сообщений, экспертной аналитики и т.п. Очевидно, что агрегация сущностей-объектов как методология естественного языка направлена на упрощение восприятия.

Однако такое упрощение приводит к избыточности языка и неоднозначности толкования терминов-заменителей. Поэтому при проектировании высоко-нормализованных схем БД все агрегированные термины ПрО, на которые, так или иначе, вынужденно ссылаются атомарные и слабые сущности-объекты, должны учитываться концептуальной и логической моделью. Очевидно, что если таких ссылок нет, то данный агрегированный термин обозначает несвязанную с функционированием ПрО сущность-объект (тривиальный случай).

Обобщение [291] формируется так же как связь, но как связь, объединяющая некоторые объекты в общий класс по некоторому признаку. Обобщение своим наименованием заменяет несколько разных наименований объектов.

Рассмотрим на примере. Класс объектов {СОБАКА, КОШКА, СЛОН} может быть обобщен в объект ЖИВОТНОЕ. В этом обобщении подчеркивается общая природа объектов СОБАКА, КОШКА и СЛОН, а

их индивидуальные отличия (например, собака лает, слон имеет хобот) игнорируются. Класс объектов {СОБАКА, КОШКА, СЛОН}. Родовой объект ЖИВОТНОЕ, ДОРОЖНОЕ СРЕДСТВО ПЕРЕДВИЖЕНИЯ, ФРУКТ: {ПОВОЗКА, ТЯГАЧ, ВЕЛОСИПЕД, ...}, {ЯБЛОКО, БАНАН, ГРУША}.

В общем случае, если класс $\{D_1, ..., D_n\}$ может быть обобщен в D, говорят, что D_i является *категорией* (относится к категории) D. Например, СОБАКА является категорией (относится к категории) ЖИВОТНЫХ.

Из предыдущих рассуждений может быть выведено несколько важных заключение, а именно: крайне важно однозначно интерпретировать объект из ПрО как атомарную сущность, связь-агрегат, связь-обобщение или компоненту. И поскольку разные пользователи могут по-разному интерпретировать объекты, такая интерпретация должна основываться на некоторой формальной модели, которая не зависит от точки зрения конкретного пользователя.

Таким образом, для того, чтобы развивать интероперабельность и кроссплатформность, необходима унификация схем БД вплоть до ее стандартизации. А потому нужна строгая формальная интерпретация таких «заменителей смысла». Нельзя допускать, чтобы «различные пользователи выбрали различные объекты в качестве сущностей». В этом и миссия «концептуальных моделей».

Связи [291, 359]**.** Под связями понимаются ассоциации между одинаковыми или различными типами сущностей. Сущности соотносятся в ПрО между собой, а механизм связей используется для отображения этого соответствия в модели. Важными являются следующие характеристики: наименование связи, степень ассоциативности, избирательность, однозначность, время существования связи и ее идентификатор.

Связь имеет *наименование.* Выбор наименования заключает в себе определенный смысл. Например, связь БЫТЬ-ЗАПОЛНЕННЫМ предоставляет пользователю некоторую информацию. Она содержит также определенную направленность, например «дом заполнен людьми, домашними животными и грызунами». Связь между сущностями СЛУЖАЩИЕ и ПРОФЕССИИ может указывать направление от конкретного служащего к набору профессий, которыми он владеет. С другой стороны, СЛУЖАЩИЕ-С-ПРОФЕССИЕЙ указывает другое направление связи, ассоциируя каждую профессию со служащими, владеющими этой профессией.

Основная цель объединения представлений заключается в идентификации и выделении общих аспектов различных представлений, а также в обнаружении и разрешении их основных противоречий. Этот процесс включает анализ и принятие решений на нескольких уровнях [291].

Несогласованность наименований. Идентификация синонимов и омонимов среди элементов данных.

Несогласованность идентификации. Различная идентификация одних и тех же типов сущностей (например, служащие могут однозначно идентифицироваться номером страхового полиса в одном приложении и номером служащего — в другом).

Несогласованность агрегации. Ограничение различных групп элементов на структурном уровне или операций над значениями элементов на уровне экземпляров (например, означает ли «суммарные закупки» суммарные для округа, страны и т. д.).

Дополняющие подмножества. Распознавание взаимодополняющих друг друга подмножеств данных, таких, как «служащие, работающие неполный рабочий день», «служащие, работающие полный рабочий день» и «уволенные служащие».

Противоречивость требований обновления. Обнаружение

несогласованных правил добавления/исключения среди различных представлений пользователей.

Противоречивость ограничений целостности. Идентификация различий в правилах поддержания целостности данных. Например, каждый новый проект создает новый экземпляр сущности СЛУЖАЩИЙ, вызывая тем самым дублирование.

1.4. Основные определения реляционной модели данных

В РМД [361] даны определения основных категорий модели. Выпишем их. При этом оговорим, что в работе будет использован следующий принцип нумерации утверждений. Определения общего характера отдельной нумерацией выделяться не будут. Определения, используемые непосредственно в формулировке лемм и теорем, будут нумероваться отдельно. При этом в номерах цитируемых утверждений будет использован символ фамилии автора в соответствии со списком обозначений.

Сущность – уникально идентифицированная категория ПрО, информация о которой интересует пользователя для накапливания и хранения.

Предметная область – множество всех сущностей, свойства которых и отношения между которыми моделируются схемой РБД.

Как правило, под ПрО понимается система, обладающая свойствами целостности, полноты, замкнутости и связности [137, 159].

Атрибут – именованная характеристика сущности, с помощью которой моделируются ее свойства.

Таким образом, каждая сущность обладает хотя бы одним атрибутом.

Составной атрибут – объединение (сцепление, далее - конкатенация) нескольких атрибутов в один.

Ключ, ключевой атрибут – атрибут, значение которого уникально идентифицирует экземпляр сущности.

Составной ключ – составной атрибут, только конкатенированное значение которого уникально идентифицирует экземпляр сущности.

Арность (составность) ключа – минимальное число атрибутов, конкатенация которых формирует ключевой атрибут.

Суррогатный атрибут – атрибут, искусственно внесенный проектировщиком в совокупность атрибутов заданной сущности.

Связь – ассоциация сущностей в ПрО, представленная в схеме БД наравне с сущностями.

Таким образом, связь сущностей – это такая же часть данных ПрО, как и сущность.

Атрибут связи – именованная характеристика связи, с помощью которой моделируются ее свойства.

В моделях данных связи, которые не имеют атрибутов, являются неактуальными, так как являются функцией времени. Тогда одним из обязательных атрибутов связи является отрезок времени актуальности связи.

Степень связи – дискретный показатель типа отображения экземпляра одной сущности на экземпляры других сущностей, принимает четыре значения: 1:1 («один к одному»), $1:H$ («один ко многим»), $G:1$ («многие к одному»), $G:H$ («многие ко многим»).

Очевидно, что второе и третье значение степени связи равнозначны.

Кортеж – набор данных заранее оговоренных типов, так что экземпляр каждой сущности ПрО представляется в БД в виде кортежа.

Отношение (домен кортежей, реляционная таблица и т.д.) – двумерная таблица, имеющая уникальное имя и состоящая из строк и столбцов; строки соответствуют кортежам, а столбцы – атрибутам.

Домен – множество допустимых значений атрибута отношения

Схема отношения – имя отношения и состав его атрибутов $R(A,B,\ldots,X,Y,Z,\ldots)$.

Функциональная зависимость (ФЗ) – атрибут X функционально определяет Y ($X \rightarrow Y$), если каждому значению X соответствует единственное значение Y, но единственное значение Y соответствует набору X.

Детерминант ФЗ – определитель ФЗ, т.е. левая (независимая) часть ФЗ, аналог области определения аргумента функции.

Зависимая часть ФЗ – правая часть, аналог области значений функции.

Многозначная зависимость (МЗ) [377] – для данного отношения $R(A,B,C)$ МЗ $A \rightarrow\rightarrow B$ выполняется тогда и только тогда, когда выполняется МЗ $A \rightarrow\rightarrow C$.

Поэтому МЗ всегда образуют связанные пары, обычно их представляют совместно в символьном виде: $A \rightarrow\rightarrow B|C$.

Верно и еще одно утверждение. В отношении $R(A,B,C)$ существует МЗ $A \rightarrow\rightarrow B|C$ тогда и только тогда, когда множество значений B, соответствующее паре значений A и C, зависит только от A и не зависит от C.

Зависимость «проекции-соединения» (ЗПС) [169, 378] - отношение $R(X,Y,...,Z)$ удовлетворяет ЗПС $*(X,Y,...,Z)$ тогда, и только тогда, когда R восстанавливается без потерь путем соединения своих проекций на $X, Y, ..., Z$.

1.5. Нормализация схем БД

Получение схем БД, обладающих свойствами производительности при манипулировании данными, является актуальной задачей. Проявление идеи универсальности схемы БД состоит в алгоритме синтеза, предложенном Ф.А. Бернштейном [349, 350]. Эту же идею сформулировал и П.П. Чен [359]. А Ю.А. Пергаменцев высказал эту мысль в [238]. Основной посыл – унификация моделирования данных - по мнению многих авторов [300] весьма перспективен [2, 131].

Как указано в [197], главный фактор декомпозиции в процедуре нормализации [361] – ФЗ, т.е. не что иное, как связи. Рассмотрим причину аномалий [361] и отличий схем БД для разных ПрО. Для этого рассмотрим традиционные процедуры нормализации [169, 361].

Пример 1.1. 2НФ.

Рассмотрим отношение *ПОСТАВКИ (Код_Поставщика, Товар, Цена)*. Анализируя его, проектировщик отмечает следующие факты:

1) ключ отношения: *Код_Поставщика+Товар*;

2) две ФЗ: *Код_Поставщика+Товар→Цена*, *Товар→Цена*

Известно, что наличие в отношении частичной зависимости неключевого атрибута *«Цена»* от части ключа *«Товар»* приведет к аномалиям включения, удаления и модификации. Устраняется декомпозицией по атрибуту *«Товар»*. Свободными от аномалий будут отношения: *ПОСТАВКИ (Код_Поставщика, Товар), ЦЕНА ТОВАРА (Товар, Цена)*.

Причина проблемы – наличие избыточной ФЗ в отношении.

Пример 1.2. 3НФ

Пусть в отношении *ЛЕЧЕНИЕ (Код_Врача, Код_Пациента, Лекарство, Дата, Побочный_Эффект)*:

1) ключ отношения: *Код_Врача+Код_Пациента+Дата*;
2) две ФЗ: *Код_Врача+Код_Пациента+Дата →Лекарство*,
   Лекарство →Побочный_Эффект .

Известно, что наличие в отношении транзитивной зависимости типа $X \to Y$, $Y \to Z$ (но при этом Y−/→X) приводит к аномалиям:

1) нельзя включить в приведенное отношение сведения о лекарстве, не назначенном ни одному больному;
2) при изменении сведений о побочном эффекте лекарства требуется пересмотреть все отношения для корректного изменения данных.

Алгоритм нормализации тот же. Устранив транзитивные ФЗ и ФЗ атрибутов от части ключа, Э. Кодд получил 3НФ [361]. Причина проблемы – транзитивность ФЗ.

Пример 1.3. НФБК

Рассмотрим отношение *ПОСТАВКА* (*Код_Поставщика, Имя_Поставщика, Код_Товара, Объем_Покупки)*. При условии, что *Имя_Поставщика* - уникальный атрибут, имеем:

1) ключи отношения: *Код_Поставщика+Код_Товара*,
   Имя_Поставщика+Код_Товара.
2) две ФЗ: *Код_Поставщика →Имя_Поставщика* ,
   Имя_Поставщика →Код_Поставщика.

Наличие этих ФЗ означает, что в отношении присутствуют два детерминанта. Но каждый из детерминантов не является полностью ключом. Аномалии проявятся при изменении имени поставщика. Нормализация – проекция по атрибуту *Имя_Поставщика*.

Причина проблемы - зависимость части ключа от неключевых детерминантов. Несмотря на то, что детерминантами являются части ключей, они сами по себе ключами не являются. То есть причина все та же - «проблемные» ФЗ [361].

Таким образом, в каждом из приведенных случаев устранение нежелательных («паразитных») ФЗ приводит к нормализации отношений. Тем не менее, исходным для последовательной нормализации является отношение в 1НФ [361]. Как известно, такое отношение получается бессистемным формированием совокупности атрибутов. Поэтому, несмотря на универсальность подхода нормализации [361], результирующие нормализованные отношения не гарантируют универсальности схемы БД для ПрО.

1.6. Выводы к первому разделу

1. Нормализация по методу декомпозиции – это устаревший подход, приводящий к сложно-развиваемым и практически несовместимым приложениям.

2. Неуправляемые зависимости в отношениях приводят к аномалиям и неуправляемой избыточности данных.

3. Интероперабельность и кроссплатформность несовместимы с декомпозицией.

4. Декомпозиция стимулирует семантический разрыв между РМД и другими методами моделирования ПрО, например инженерию знаний.

5. Основным критерием качества приложений является высокая нормализованность схем БД, причем независимо от того, разрабатывается она в РМД, или в ООБД. Аномалии не зависят от модели данных.

6. В 90-х годах прошлого столетия разными авторами были предложены важные решения, которые могли бы лечь в основу новой трактовки РМД. Однако они не получили дальнейшего развития и не стали основой нового подхода к отображению семантики. Обнаружены частичные совпадения некоторых решений с основными результатами КМД.

7. Путем литературного исследования обнаружен и подтвержден приоритет диссертанта по самым важным подходам КМД: булеан как модель отображения семантики ПрО, многоарный единственный ключ отношения, on-line-агрегация и материализация представлений, модифицируемость булеанных схем БД, ориентированная на связь модель агрегаций, шунтирование как метод повышения эффективности БД.

РАЗДЕЛ 2
КАРКАСНАЯ МОДЕЛЬ ДАННЫХ

2.1. Введение ко второму разделу.

Разработка схем БД ограничивается классическим подходом Э. Кодда [361]. Попытка Кристофера Дейта и Хью Дарвена [366] создать формальную «надстройку» над РМД не находит практического применения. Уже традиционным стало построение либо модели «сущность-связь» (ER-модель) [359] либо так называемой семантической объектной модели (SOM) и последующий «перевод» получаемых семантических структур в схемы РБД [491, 492]. Вместе с тем этот подход отличает известная локальность построений, отсутствие универсальности, приводящие к сложностям при модификации схемы БД, вплоть до необходимости полного перепроектирования схемы [204, 206, 212].

Локальность ER/SOM построений заключается, прежде всего, в работе с фиксированным графом-схемой либо с заданными множествами семантически определенных сущностей-объектов. Это сказывается на гибкости и модифицируемости таких построений.

Дополнительные затруднения связаны с нечеткостью ключевых определений «атрибут», «сущность» и «объект» и с отсутствием общепринятой аксиоматики. Строгие алгебраические определения понятий по типу [144, 244, 319], которые для ER/SOM построений являются ключевыми, фактически отсутствуют. Это указывает на то, что традиционные решения не приспособлены для создания универсальных структурных схем, которые были бы формально полны и непротиворечивы.

Несмотря на то, что существуют развитые техники декомпозиции схем БД [368], подобный подход к конструированию имеет кардинальный недостаток – отсутствие гибкости. Именно, изменение совокупности атрибутов и/или взаимосвязей между ними (например, при расширении ПрО, что на практике случается чаще всего), как правило, вызывает необходимость повторного формирования нормализованной схемы БД. В подобных случаях

логическая схема БД либо трудно модифицируема, либо вообще не поддается модификации (поскольку между старой и новой схемами БД нет строгой преемственности).

Для отображения в БД новых дополнительных сущностей-объектов и связей между ними требуется сформировать расширенную совокупность сущностей-объектов и связей, а затем выполнить процедуру нормализации, т.е. фактически решить задачу декомпозиции заново.

При этом корректная процедура нормализации схемы БД должна обеспечивать сохранность информации - как соединение без потерь, так и сохранение исходных зависимостей, что далеко не всегда осуществимо для НФБК и выше. Можно утверждать, что подход к построению схемы БД, основанный на декомпозиции, неустойчив по отношению к исходному множеству атрибутов и связей.

Альтернативные алгоритмы синтеза нормализованных схем по Бернштейну [349, 350] отказывают, если семантика ПрО предполагает наличие МЗ или ЗПС.

2.2. Постановка задачи

Известно, что со временем любую схему БД, не отвечающую новым требованиям, необходимо обновлять, что обычно существенно повышает стоимость сопровождения. Поэтому частое перепроектирование схемы БД приводит к реинжинирингу приложений, что нежелательно для пользователей. Этот процесс приводит к значительным временным и финансовым затратам. Поэтому модифицируемость схемы БД становится одним из самых важных критериев качества приложения.

Под *модифицируемостью* схемы БД понимается [204, 206] возможность независимого добавления дополнительных или удаление существующих подсхем, а также возможность внесения изменений в схему отношений, причем минимальным числом операций. Иными словами, добавление или удаление новых отношений в БД не должно затрагивать уже существующие отношения.

Пусть в ПрО имеем набор N некоторый сущностей ПрО. Необходимо построить алгоритм, который проектировщику схемы БД позволяет:

- получать совокупность реляционных отношений, каждое из которых удовлетворяет критериям не ниже 5НФ [169, 378],

- анализировать ПрО для выявления противоречий, в том числе неатомарность атрибутов,

- учитывать начальные иерархические зависимости поднабора сущностей (так называемых «слабых» сущностей),

- без вспомогательных графических средств моделировать n-арные связи (в том числе рекурсивные) между сущностями, степень которых в общем случае $G:H$,

- интегрировать нужное количество атрибутов связей,

- обеспечить модифицируемость схемы БД так, чтобы в процессе эксплуатации не изменялось обслуживаемое приложение.

2.3. Нормализация схемы отношения посредством шунтирования.

В работе [205] введено понятие «шунтирования МЗ» и доказана теорема о шунтировании.

Определение 2.1. *Шунтированием* МЗ назовем добавление в отношение $R(X, Y, Z)$, где содержится МЗ $X \to\to Y \setminus Z$, дополнительного поднабора $\{A\}$ неключевых атрибутов так, что $(X+Y+Z) \to \mathrm{A}$. Тут X, Y, Z могут быть составными атрибутами, т.е. $X = \{X\}, Y = \{Y\}, Z = \{Z\}$.

Тут знаком «+» обозначено конкатенацию (сцепление) экземпляров столбцов множества атрибутов (по строке). Например, в [300, 349, 361] эта операция обозначена записью атрибутов рядом XY или запятой между ними X,Y. Но для большего понимания действия этой операции воспользуемся символом суммирования.

Для дальнейшего построения новой модели данных сформулируем и докажем базовую теорему.

Теорема 2.1. Если в отношении $R(X, Y, Z)$ имеется нетривиальная МЗ $X \to\to Y \setminus Z$, причем X, Y, Z могут быть составными множествами, то есть

$X = \{X\}$, $Y = \{Y\}$, $Z = \{Z\}$, то после добавления дополнительных неключевых атрибутов {A}, причем так, что $(X + Y + Z) \to A$, все зависимости в $R(X,Y,Z,A)$ становятся только функциональными.

Доказательство. Предположим обратное, что в отношении $R(W, Y, Z)$, где $W = \{X + A\}$, имеется МЗ $W \to\to Y/Z$. Пусть R – реляционная схема, W и Y – не пересекающиеся подмножества R. И пусть $Z = R - (W + Y)$. Следуя [377], отношение R удовлетворяет МЗ $W \to\to Y$, если для любых двух кортежей t_1 и t_2 из R, для которых $t_1(W) = t_2(W)$, в R существуют еще и кортеж t_3, для которого выполняется соотношение $t_3(W) = t_1(W),\ t_3(Y) = t_1(Y),\ t_3(Z) = t_1(Z)$.

То есть, поскольку в отношении $R(W, Y, Z)$ имеются кортежи:

$$r_1 = (w, y, z_1),$$

$$r_2 = (w, y_1, z),$$

то из МЗ должны следовать и кортежи:

$$r_3 = (w, y, z),$$

$$r_4 = (w, y_1, z_1).$$

Но эквивалентность кортежей в МЗ нарушается, так как:

$$r_1 = (x + a_1, y, z_1),$$

$$r_2 = (x + a_2, y_1, z),$$

$$r_3 = (x + a_3, y, z),$$

$$r_4 = (x + a_4, y_1, z_1).$$

Значит, в полном отношении МЗ отсутствует. □

Из доказанного следует, что если отношение имеет МЗ, ее можно «шунтировать», то есть, перекрыть ее влияние добавлением нового элемента – естественного или суррогатного атрибута. «Физический» смысл такого элемента - атрибут связи. Поэтому в дальнейшем будем называть доказанное выше утверждение теоремой о шунтировании МЗ, а отношения с «перекрытой» МЗ – шунтированными отношениями.

В качестве примера рассмотрим ПрО из [236]. Пусть «абитуриенты» сдают «вступительные экзамены» по списку «предметов» в «институты».

При этом дополнительным условием является разрешение любому абитуриенту подавать документы в любой институт и выбирать при этом день сдачи экзамена. Данное упрощение мы вводим лишь для того, чтобы не загромождать пример дополнительными ограничениями, которые лишь отвлекут от сути. Заметим, однако, что данный пример не упрощает ситуацию, а наоборот усложняет её. В реальности же степень связи $G:H$ по произвольному числу сущностей встречается крайне редко. А это значит, что появляются дополнительные ФЗ в ключевых атрибутах, которые лишь упрощают шунтирование МЗ.

X – «Абитуриенты» - $X=\{1,\ 2,\ 3\}$.

Y – «Предметы» - $Y=\{1,\ 2,\ 3\}$.

Z – «Институты» - $Z=\{1,\ 2,\ 3\}$.

$A_{1,2}$ – оценка на экзамене, дата или время его проведения и т.п.

Табл. 2.1

Отношение с шунтированной МЗ

X	Y	Z	A_1	A_2
1	1	1	3	12.10
1	1	2	5	16.30
1	1	3	2	10.30
1	2	1	3	11.00
1	2	2	5	15.00
1	2	3	4	12.10
1	3	1	2	10.30
1	3	2	3	11.00
1	3	3	3	15.00
2	1	1	5	14.10
2	1	2	3	12.30
2	1	3	5	11.45
2	2	1	5	12.10

2	2	2	4	10.30
2	2	3	2	16.30
2	3	1	3	11.00
2	3	2	5	15.00
2	3	3	2	13.00
3	1	1	5	12.10
3	1	2	3	11.00
3	1	3	4	11.45
3	2	1	2	10.15
3	2	2	3	15.00
3	2	3	2	11.00
3	3	1	5	10.30
3	3	2	2	12.10
3	3	3	5	16.30

В табл. 2.1. приведено полное отношение, исходя из максимально возможного объема кортежей, полученное декартовым произведением [144] атрибутов, чтобы наглядно показать основной вывод теоремы. Очевидно, что на практике многие кортежи вообще не появились бы. Условно «абитуриент 2» не явился бы на экзамен «2+2+3» и не получил бы двойку, зная, что он не готов к «предмету 3».

Заметим, что никакая часть A_i не зависит ни от какой комбинации частей ключа, кроме как от самого ключа. И сам ключ - так же. Получена форма, подобная НФБК [361], но без МЗ. Как известно из [377], это - 4НФ.

Впервые эта теорема была доказана в [205].

2.4. Оператор синтеза реляционного каркаса

Как известно, алгоритм формирования отношений в РМД [361] основан на операции декартова произведения множеств. Рассмотрим совокупность отношений, которая может быть получена путем сочетания декартовых произведений.

Определение 2.2. *Реляционным каркасом* (РК) будем называть совокупность отношений, порождаемую сочетанием декартовых произведений конечного множества унарных атрибутов.

Введем следующий оператор $L_l^m(X_j)$.

Определение 2.3. Оператор $L_l^m(X_j)$, формирующий в соответствии с индексом $m = 1,...,N$ множество сочетаний декартовых произведений унарных атрибутов X_j, где $j = 1,...,N$ – номер унарного атрибута, m – текущая арность декартового произведения, $l = 1,...,L_m$ – номер декартового произведения m-й арности, назовем *оператором синтеза РК*.

Заметим, что в дальнейшем моделировании j – номер *атомарной сущности* из ПрО, а $m = 1,...,N$ - арность ключей *составных сущностей*. Их строгое определение будет приведено далее.

Совокупность имеет вид: $L_1^1(X_j) = (X_1)$, $L_2^1(X_j) = (X_2)$, …, $L_N^1(X_j) = (X_N)$, $L_l^2(X_j) = (X_1 + X_2)$, …, $L_1^3(X_j) = (X_1 + X_2 + X_3)$, …, $L_1^{N-1}(X_j) = (X_1 + X_2 + X_3 + X_4 + ... + X_{N-1})$, $L_2^{N-1}(X_j) = (X_2 + X_3 + X_4 + ... + X_N)$, …, $L_1^N(X_j) = (X_1 + X_2 + X_4 + ... + X_N)$.

Общее число S полученных групп декартовых произведений (а в схемах БД – конкатенаций) определяется выражением числа сочетаний [204, 206] $2^N - 1$, где N – число унарных атрибутов. Очевидно, что оператор синтеза РК построен на множестве всех подмножеств (булеане) отношений.

Известно [144], что структура множества всех подмножеств обладает тем свойством, что пересечения и объединения определены в ней не только для двух элементов (и поэтому, в силу ассоциативности, для любого конечного числа элементов), но и для любых бесконечных подмножеств.

Иными словами, эта структура булеана является *полной структурой* в следующем смысле.

Определение 2.К1. Частично упорядоченное множество S называется полной структурой, если для любого непустого подмножества $A \subseteq S$ в S существуют элементы c и d со следующими свойствами:

1. $\forall a \in A$ выполняется неравенство $c \le a$, причем если некоторый элемент c' также удовлетворяет условию $c' \le a \ \ \forall a \in A$, то $c' \le c$;

2. $\forall a \in A$выполняется неравенство $d \ge a$, причем если некоторый элемент d'также удовлетворяет условию $d' \ge a \ \forall a \in A$, то $d' \ge d$.

Там же [144] в общем виде доказана **Теорема 2.К1.** (о полноте). Частично упорядоченное множество S тогда и только тогда будет полной структурой, если оно обладает единичным элементом и если в нем существует пересечение элементов для каждого непустого подмножества.

Тогда для оператора $L_l^m(X_j)$ сформулируем базовое утверждение, позволяющее говорить о нем как о структуре КМД.

Теорема 2.2 (о полноте РК)**.** Совокупность отношений $R_l^m(L_l^m(X_j))$, которая построена на операторе синтеза РК, является полным множеством (полной структурой) отношений.

Доказательство.

Используем [144]. Пусть существует множество всех подмножеств отношений $\mathrm{R}^{\mathrm{n}}(X_1, X_2 ...)$ некоторых n элементов $\mathrm{X_i}(i = 1, n)$ так, что их общее число равно 2^n. Т.е. имеем совокупность отношений от n унарных до одного *n*-арного, в том числе и «пустое» отношение: унарные $\mathrm{R}(X_i)$, бинарные $\mathrm{R}(X_i, X_j)$, тернарные $\mathrm{R}(X_i, X_j, X_m)$, квартарные $\mathrm{R}(X_i, X_j, X_m, X_l)$,..., единственное *n*-арное $\mathrm{R}(X_1, X_2, ...X_N)$ и единственное пустое $\mathrm{R}()$, где по алгоритму декартового произведения $i = j = l = m = ... = 1, n.$

Для описанной частично упорядоченной совокупности отношений $\mathrm{R}^{\mathrm{n}}(X_1, X_2, ...)$ (т.е., решетки [144, 156]) существует «единичный элемент» и «ноль». Роль нуля и единичного элемента в структуре булеана — пустое подмножество $R()$ и само множество $R^n(X_1, X_2, ...)$. Но тогда, поскольку существуют нижняя и верхняя границы, в совокупности отношений $R^n(X_1, X_2, ...)$ обязательно существует пересечение элементов для каждого непустого подмножества. Тогда по теореме о полноте частично

упорядоченного множества совокупность отношений, построенных на операторе синтеза РК, полная. □

Заметим, что ниже будет приведено иное доказательство этой теоремы.

На рис. 2.1. показана общая схема РК. Как отмечалось в разделе 1, понятие «сущность-объект» будет определено далее.

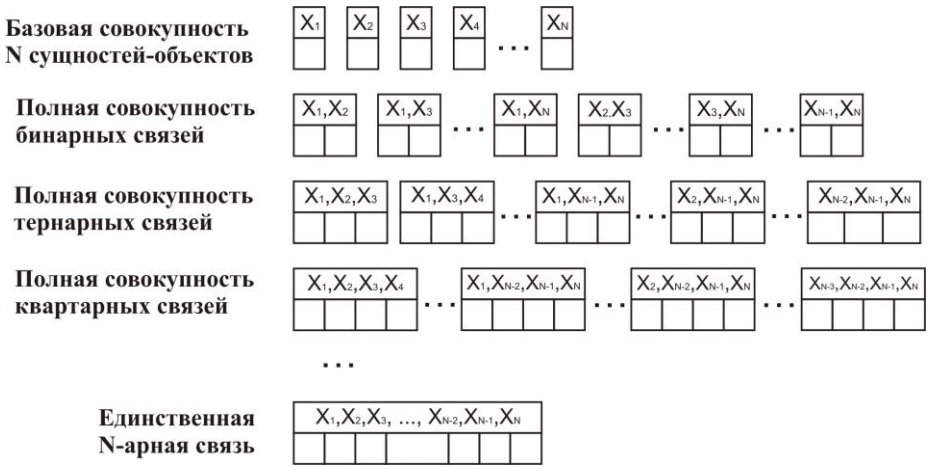


Рис. 2.1. РК для N сущностей-объектов

Очевидно, в контексте конкретных схем БД большинство отношений неактуальны, но их актуализация, по сути, модифицирует эти схемы БД. Из этого следует, что модификация схемы БД сводится к двум типам операций: актуализация-аннулирование отношения и актуализация-аннулирование произвольного множества неключевых атрибутов в произвольной группе отношений. При этом целостность БД сводится, прежде всего, к уникальности совокупности ключевых атрибутов и их строгого соответствия в логически связанных отношениях. Присвоив группе отношений статус неактуальных, проектировщик снижает количество используемых отношений.

Очевидно, что особенность приведенных отношений заключена именно в том, что они образуют полный [144] РК всех схем отношений ПрО. Заметим, что в каждом отношении из совокупности $R_l^m(L_l^m(X_j))$ ключом отношения является вся строка (вся конкатенация) X_j. Однако, если каждое отношения расширить добавлением неключевых атрибутов, которые полностью зависят от ключа (шунтировать это отношение), совокупность

$R_l^m(L_l^m(X_j),A_{li})$, где $i=1,...,I_l-$ это номер неключевого атрибута для l-го отношения, будет также полной структурой. Последнее утверждение в связи с очевидностью его справедливости оформлять отдельной теоремой не будем.

Тогда на совокупности $R_l^m(L_l^m(X_j),A_{li})$ можно построить схему БД, отображающую данные от N атомарных сущностей ПрО на множество отношений, количество которых S определяется вышеприведенной формулой числа сочетаний.

Отметим также, что в соответствии с теоремой 2.1. все отношения этой совокупности нормализованы не ниже 4НФ [377]. Если ключевые атрибуты отношений отвечают одному предикату, то физический смысл таких отношений - это совокупности атрибутов соответствующих атомарных сущностей ПрО. Такие отношения в практике проектирования БД принято называть «справочниками», указывая на их свойство независимости (инвариантности) от приложений. Они не зависят от специфики ПрО, т.е. также инвариантны и ей.

Если каждая часть ключа такого отношения отвечает разным предикатам, это отношение моделирует связь сущностей. То есть отношения с шунтированной МЗ, полученные декартовым произведением ключевых атрибутов сущностей ПрО, моделируют произвольные связи любой степени – от 1:1 до $G:H$. Как отмечено, неключевые атрибуты таких отношений являются атрибутами этой связи.

Следовательно, совокупность всех отношений S моделирует весь объем данных обо всех сущностей, входящих в эту ПрО.

П.П. Чен в [359] предложил моделировать связи разной степени между сущностями с помощью отдельных виртуальных сущностей, в которых функции связи реальных сущностей он назвал *ролями*. Тогда в соответствии с ER-моделью Чена множество $R_l^m(L_l^m(X_j),A_{li})$ есть не что иное, как полная совокупность сущностей, связей и их ролей.

Отметим, что описанная выше схема позволяет моделировать лишь «одноразовые» [300] связи каждой сущности с произвольной группой. Отсутствие ограничений на количество *видов связей* на любом уровне арности в каждой конкретной группе сущностей, свойственное такой сущности, как «люди» и приводящее к *рекурсивным* связям, в изложенном алгоритме учитывается с помощью *копий* сущностей, дополнительных отношений, смысл которых – *маски для ролей* в связях сущностей. Очевидно, что учет этих особенностей не внесет принципиальных изменений в схемы БД построенных на РК. Этот подход рассмотрен в разделе 4.

Для описанного подхода принципиально важным является его совпадение с методом рекурсивного решета. В [264] показано, что это метод комбинаторного программирования, который рассматривает конечное множество и исключает все элементы этого множества, не представляющие интереса. Является логическим дополнением к процессу поиска с возвращением, который перечисляет все неэквивалентные элементы множества.

Пусть из данного множества объектов необходимо исключить максимальное подмножество попарно изоморфных объектов. Имеем множество объектов $\{a_1,a_2,a_3,a_4,a_5,\ldots,a_n\}$. В соответствии с алгоритмом решета находим все $a_i = a_1,\ i>1$, и вычеркиваем их. Для a_{l_1} – первого, не вычеркнутого после a_1, находим все $a_i = a_{l_1},\ i > l_1$, и вычеркиваем их. Продолжая далее, получаем максимальное множество неэквивалентных объектов. Т.о., алгоритм решета, отбраковывающего изоморфные объекты, подобен алгоритму исключения неактуальных ячеек каркаса [209] из полной каркасной совокупности.

2.5. Категории каркасной модели данных

Гипотеза 2.1. Причина отличий схем отношений в БД для разных ПрО – не контролируемые зависимости

Исходя из этого утверждения, несколько иначе определим основные категории РМД.

Сущностями-объектами назовем множества, каждый элемент которых имеет общий набор параметров, соответствующих общему предикату объединения элементов.

Такое определение упрощает понимание отличий сущностей-объектов от атрибута как еще одной важной категории модели, схожей с сущностью-объектом.

Атрибутом назовем характеристику сущности-объекта или связи сущностей-объектов, один из тех их параметров, набор которых соответствует единому предикату. Дополнительным признаком атрибута является тот факт, что атрибут, в отличие от сущности-объекта, не имеет никаких атрибутов.

Связь – зависимость между сущностями-объектами, а также ФЗ атрибутов от своих сущностей-объектов.

Таким образом, связь рассматривается также как сущность-объект.

Предикатом сущности-объекта (предикатом объединения элементов множества в сущность-объект) назовем такой предикат, который является истинным (возвращает истину) только для переменных, которые объединяются в это множество.

Таким образом, предикат сущности-объекта – это некий формальный индикатор, который определяет, принадлежит ли данная переменная (атрибут) к этой сущности-объекту или нет.

Заметим, что на практике сформировать такой индикатор, как правило, бывает крайне сложно. Однако методы моделирования функций, которые в соответствии с численным значением исследуемой переменной отличают наличие или отсутствие корреляции с иными значениями переменной данного множества, изучает в частности математическая статистика [3]. Такой подход позволяет сделать начальную оценку сущности-объекта и ее места в ПрО более объективной.

Специфика ПрО – совокупность связей между сущностями-объектами произвольной степени – от 1:1 до $G:H$.

Предполагается, что все многообразие атрибутов сущностей-объектов, а также характеристики этих атрибутов не влияют на специфику ПрО.

Ключ сущности-объекта – минимальная совокупность атрибутов сущности-объекта, достаточная для уникальной идентификации экземпляра сущности-объекта.

Арность (*составность*) *ключа* – минимальное число атрибутов, конкатенация которых формирует ключ сущности-объекта (ключевой атрибут).

Суррогатный атрибут – искусственно внесенный в отношение атрибут (например, ключевой), имеющий гарантированные свойства. Если суррогатный атрибут вносится в качестве ключа, то обеспечивается его уникальность для любого экземпляра сущности-объекта, а также минимальная достаточность числа разрядов значения, пропорционального проектному объему экземпляров отношения.

Простейшей назовем схему отношения $R(X, A)$, когда $X \to A$, X – простой (унарный, односоставный) ключ отношения, A – простой унарный неключевой атрибут.

Особой назовем схему отношения $R(X_j, A_i)$, когда $X_j \to A_i$, $\{X_j, j = 1,...,J\}$ – в общем случае единственный составной ключ отношения, форма - не ниже 5НФ [378], причем так, что ни один атрибут сам по себе не является ни ключом, ни детерминантом. Множество неключевых атрибутов $\{A_i, i = 1,...,L\}$ такое, что ни один из них не является детерминантом.

Подобными назовем схемы отношений с равным числом ключей и равным числом ФЗ.

При этом тип и число состава неключевых атрибутов не влияют на подобие схем отношений. Отметим, что если в сравниваемых отношениях существует разное число ключей, то отношения не являются подобными.

Коэффициентом подобия схем отношений назовем целую часть от деления большей арности ключей на меньшую. Если в сравниваемых отношениях существует несколько ключей (однако их число - равно), то сравнению подлежат ключи старшей арности. Для сравнения с простейшим отношением смысл коэффициента подобия – это превышение единичной арности.

Заметим при этом, что наличие нескольких ключей в одном отношении является, вообще говоря, аномалией [306]. Проектировщик может избавиться от аномалии декомпозицией отношения. Поэтому, исследование вопроса подобия отношений, необходимое, например, при объединении нескольких схем БД от разных приложений в одну, лучше проводить уже после исключения такой аномалии.

Например, 3 - коэффициент подобия простейшей схемы и схемы с тернарным ключом в отношении $R(X, Y, Z, A_i)$, где $X+Y+Z$ – тернарный ключ, а A_i - множество неключевых атрибутов.

Таким образом, подобными будут схемы, у которых одинаковое число ключей и ФЗ, а отличия могут быть лишь в арности ключей. Соотношение арностей назовем коэффициентом подобия.

Внешний атрибут (след внешней сущности-объекта) – атрибут (возможно, ключевой или даже суррогатный), предикат которого отличается от предиката сущности-объекта.

Внутренняя связь – ФЗ совокупности неключевых атрибутов от составного ключа сущности-объекта.

Внешняя связь – связь, предикат неключевых атрибутов которой отличается от предикатов каждого ключевого атрибута сущностей-объектов, входящих в эту связь.

Таким образом, внешняя связь – это связь сущности-объекта с иными сущностями-объектами. В описаниях ПрО такая связь, как правило, заменяется новой сущностью-объектом (новым термином), которую в

концептуальных моделях данных принято называть «составной». К составной сущности-объекту сводятся и абстракции.

Как отмечалось в разделе 1, известны следующие способы абстрагирования понятий [55]: обобщение – специализация, типизация – конкретизация, агрегация – декомпозиция и ассоциация – индивидуализация.

Обобщение и типизация и обратные им специализации и конкретизации выражают общность понятий, проявляющуюся при дифференциации. Ассоциация и агрегация, а также противоположные им - индивидуализация и декомпозиция, раскрывают интеграцию понятий.

Важным свойством РК, которое будет рассмотрено далее, является то, что все указанные типы связей моделируются единым универсальным механизмом, который также является элементом РК.

Кратность сущности-объекта (*кратность внешних связей*) – число связей у одной сущности-объекта с иными сущностями-объектами.

Номер связи (*вид связи*) – идентификатор связи в группе связей сущности-объекта, кратность которой больше единицы.

Роль – функция сущности-объекта в связи данного номера.

Маска – отношение, сформированное на частичной копии группы атрибутов сущности-объекта, достаточное для моделирования связи данного номера.

2.6. Подобие реляционных схем

Справедливой является следующая

Теорема 2.3. Все отношения, имеющие особую схему, подобны друг другу и подобны простейшей схеме.

Доказательство. Рассмотрим группу отношений со схемами $R_0(X_0,\ A_i)$, $R_1(X_1,\ A_i)$, $R_2(X_1+X_2,\ A_i)$, $R_3(X_1+X_2+X_3,\ A_i)$, … $R_j(X_j,\ A_i)$, причем, каждое из них находится в 5НФ [378] и атрибуты X_j являются различными частями составных ключей отношений (кроме унарных ключей). Множество A_i – неключевые атрибуты, где $j=1,...,J;\ i=1,...,I_j$, причем j – номер отношения и

число ключевых атрибутов (частей составного ключа) в каждом *j*-м отношении, а I_j – число неключевых атрибутов каждого *j*-го отношения.

Необходимо показать, что из R_0 следует R_1 и формально $R_2 \sim 2 \times R_0$ и $R_3 \sim 3 \times R_0$, где символом $n \times$ обозначим коэффициент подобия, то есть составность ключа.

То, что $R_1 \sim R_0$ с коэффициентом подобия *k*=1 следует из определения подобия. Ключевые атрибуты X_0 и X_1 унарны. Арность неключевых атрибутов не влияет на схему отношения.

Далее, поскольку отношения находятся в НФБК, то каждый из X_i сам по себе не может быть детерминантом – это противоречит условию НФБК [361]. То есть каждый из этих атрибутов может быть лишь частью составного ключа. Но и среди A_i также нет детерминантов. Это значит, что в R_2 и R_3 имеет место единственная ФЗ $X_j \to A_i$, то есть, в частности, для R_3 зависимость имеет вид $(X_1 + X_2 + X_3) \to A_i$.

Значит, при замене конкатенированного атрибута $X_1 + X_2 + X_3$ на атрибут W той же размерности, составляющие схемы отношения не изменятся. То есть выполняется условие подобия. А коэффициент подобия определится отношением наибольшего состава ключей. Поскольку в исходном и анализируемом отношениях есть только один ключ, отношение составов - 2 и 3 для R_2 и R_3 соответственно. □

Вывод о подобии важен для сравнения отношения с особой схемой отношений с МЗ-ЗПС.

2.7. Роль многозначной зависимости в схемах баз данных

В работе П.П. Чена [359] был предложен алгоритм исследования ПрО путем отделения сущностей-объектов в статическом состоянии от их связей. Однако вопрос нормализации отношений, каждое из которых отдельно моделирует каждую связь, не был изучен. Алгоритм, предложенный в серии работ авторов [2, 294], также основанный на идее П.П. Чена, строго не обоснован. И вопрос исключения МЗ и ЗПС, явно присутствующих в

отношениях для связей более высокой арности, в работах [2, 294] не рассмотрен. Хотя общий вывод о том, что схема БД, каждое отношение которой находится в ДКНФ, еще и обладает максимальной степенью модифицируемости и потому «может служить основой типизации и унификации в проектировании» [2, 294], частично совпадает с выводами, полученными ранее [203, 204, 225]. Однако лишь для единственной совокупности отношений – каркасной.

Видимо, то обстоятельство, что Р. Фейджин не рассмотрел один важный частный случай, когда конкатенация атрибутов с МЗ или ЗПС [377, 378] является детерминантом непустого множества неключевых атрибутов, и является причиной того, что ДКНФ, по словам Д.М. Кренке [131], «до настоящего времени является скорее искусством, чем наукой».

Исследования показывают, что РК [204, 207] – это формализованный носитель полного многообразия связей атомарных сущностей-объектов, каждая актуальная ячейка которого строго соответствует критериям безаномальности [378]. Проанализируем более подробно МЗ. Для этого рассмотрим одну из теорем Фейджина [377].

Теорема 2.F.1 (о МЗ). МЗ $X \to\to Y$ выполняется для отношения $R(X, Y, Z)$ тогда и только тогда, когда R является соединением своих проекций $R_1(X, Y)$ и $R_2(X, Z)$.

В работе [377] автор приводит нестрогое доказательство этой теоремы. Простой проверкой того факта, что $R(X, Y, Z)$ является соединением своих проекций $R_1(X, Y)$ и $R_2(X, Z)$обосновывается утверждение, что это условие выполняется. «Всякий раз, когда R содержит (x,y,z) и (x,y',z'), обязательно будут присутствовать и кортежи (x,y',z) и (x,y,z'). Это последнее условие выполняется тогда и только тогда, когда в отношении кортежи тождественны $Y_{xz} = Y_{xz'}$. Теорема следует из определения МЗ». □

Опишем один важный частный случай. Для этого на базе теоремы Фейджина о МЗ сформулируем новое утверждение в виде **Леммы 2.1.** В

отношении со схемой $R(X,Y,Z)$, которое получено операцией декартова произведения унарных отношений $R_1(X) \times R_2(Y) \times R_3(Z)$, где X,Y,Z в общем случае непустые составные атрибуты, для любых (X,Y,Z) всегда существуют кортежи: $r_1 = (x,y,z)$, $r_2 = (x,y_1,z)$, $r_3 = (x,y,z_1)$, $r_4 = (x,y_1,z_1)$.

Обратное утверждение, вообще говоря, неверно. Отличие – возможность существования в $R(X,Y,Z)$ ФЗ между атрибутами, которые лишь сокращают экстенсионал отношения, однако не исключают оговариваемых леммой закономерностей в кортежах.

Для доказательства леммы 2.1. достаточно рассмотреть операцию декартова произведения. Путем прямого применения операции над тремя произвольными одноместными множествами, в каждом из которых имеется не менее двух элементов (тривиальный случай, когда множества пусты или состоят из одного элемента, не рассматривается), немедленно получим выполнение утверждения леммы. □

А тот факт, что к аналогичному же чередованию элементов итогового множества приводит и МЗ, когда в результирующем отношении как *ограничение* [377] существуют еще ФЗ между одним из атрибутов, например X и каждым из двух других – Y и Z (но, тем не менее, между собой у атрибутов Y и Z никакой зависимости не существует), говорит о том, что описанная зависимость является частным случаем МЗ.

По аналогии с многозначной будем в дальнейшем называть ее ДЗ, понимая тот факт, что ДЗ:

- моделирует полную независимость атрибутов между собой,

- обеспечивает полноту набора кортежей в результирующем отношении.

Ниже будет дано строгое определение ДЗ и ее свойств. В соответствии с утверждением леммы 2.1 и МЗ, и ЗПС мы будем называть *обобщением* ДЗ. Потому, что именно при такой формулировке понятным становится физический смысл и МЗ, и ЗПС. Как отмечено в разделе 1, впервые эту

зависимость описал Ян Паридаенс [465, 466]. Однако по аналогии с работой [377] и на основании леммы 2.1. дадим ей иное определение.

Определение 2.4. (ДЗ). Пусть R – реляционная схема, X и Y – непересекающиеся подмножества R, причем так, что $Z = R - (X + Y)$, а также такая, что между атрибутами X,Y,Z отсутствует ФЗ. Отношение R удовлетворяет ДЗ $X \to\to Y//Z$, если *для любых* двух кортежей $r_1 = (x,y,z)$, $r_2 = (x,y_1,z_1)$, в R обязательно найдутся и кортежи $r_3 = (x,y,z_1)$, $r_4 = (x,y_1,z)$.

Будем обозначать ДЗ дополнительной наклонной чертой между именами атрибутов, подчеркивая тем самым, что в контексте РМД ДЗ является частным случаем МЗ. Имеет место следующее утверждение.

Лемма 2.2. ДЗ атрибутов в отношении $R(X,Y,Z)$ моделирует *полную* совокупность кортежей атрибутов X, Y и Z этого отношения R.

А это означает, что моделируется полная совокупность связей. Доказательство не проводим в связи с его очевидностью.

На основании этих лемм проектировщик получает возможность использовать отношение, в основе которого лежат обобщения ДЗ - или МЗ, или ЗПС, которые также моделируют связь степени $G:H$. Но не между полным сочетанием всех атрибутов, а лишь между сочетанием части из них.

Таким образом, отношение $R(X,Y,Z)$, где по аналогии с [377] под X в общем случае понимается совокупность $\{x_1,....x_n\}$, под Y – $\{y_1,....y_m\}$, а под Z – $\{z_1,....z_l\}$, которое содержит ДЗ, является предельным случаем отношения, моделирующего полную связь между соответствующими сущностями-объектами ПрО. Причем, единственное отношение моделирует одну связь указанной степени. А полная совокупность таких отношений моделирует всю полноту связей. На этом принципе и построена КМД.

Заметим, что большинство классических монографий по БД, большой обзор которых приведен в серии учебных пособий автора [135], не поясняют проектировщику физический смысл ЗПС. А между тем, именно ДЗ дает

неформальное представление и о МЗ, и о ЗПС как о подмножестве кортежей из отношения с ДЗ.

Действительно, если в отношении $R(X,Y,Z)$ существует МЗ $X \to\to Y/Z$, то при полном наборе кортежей, МЗ тождественна ДЗ.

В отношении $R(X,Y,Z)$ может отсутствовать МЗ $X \to\to Y/Z$, но существовать ЗПС: $*[R_1(X,Y), R_2(X,Z), R_3(Y,Z)]$. Т.е. такая зависимость, что операция соединения любой пары из трех проекций $R_1(X,Y), R_2(X,Z), R_3(Y,Z)$ возвращает отношение $R'(X,Y,Z)$, строго тождественное $R(X,Y,Z)$, что означает, что в соединенном отношении кортежей ни больше, и ни меньше, чем в исходном.

Это, в свою очередь, означает, что из полного отношения, обладавшего ДЗ, «вычеркнули» кортежи так, что осталась только ЗПС.

Единство МЗ и ЗПС в [377, 378] показано в виде необходимого условия выполнения в отношении $R(U)$ ЗПС $*\lfloor R_1, R_2,...,R_p \rfloor$, где U - это множество всех атрибутов отношения R ($U = R_1 R_2 ... R_p$).

Согласно [169, 377, 378], отношение R удовлетворяет МЗ $X \to\to Y/Z$ тогда и только тогда, когда R декомпозируется без потерь на $R_1(X,Y)$ и $R_2(X,Z)$, где $Z = R - (X,Y)$. Как видно, это условие совпадает с условием зависимости соединения $*[XY,XZ]$. С другой стороны, зависимость соединения $*[R_1, R_2]$ имеет тот же смысл, что и МЗ: $R_1 \cap R_2 \to\to R_1$.

Для ЗПС можно привести определение, аналогичное определению МЗ.

Пусть R удовлетворяет зависимости соединения $*\lfloor R_1, R_2,...,R_p \rfloor$. Если R содержит кортежи $t_1, t_2,...,t_p$ такие, что для всех (i, j) $t_i(R_i \cap R_j) = t_j(R_i \cap R_j)$, то R содержит и кортеж t такой, что $t(R_i) = t_i(R_i)$, где $1 \le i \le p$.

Ограничения ПрО сокращают полноту отношения, моделирующего связь. Это важное обстоятельство необходимо для понимания роли ДЗ в синтезе ДКНФ.

Уникальность ДЗ для моделирования связей между сущностями-объектами заключается в следующем. Сама зависимость является минимально возможным «многозначным» частным случаем МЗ. Но отношение с ДЗ обладает максимально возможным набором кортежей. ЗПС же, наоборот, является предельным случаем многозначного обобщения ФЗ. А отношение с ЗПС – частным случаем набора кортежей. Из этого следует, что для моделирования произвольных связей в ПрО необходимо в качестве *шаблона* использовать только отношения с ДЗ.

Другими словами, если в отношении с ДЗ ПрО «произвольно вычеркивает» некоторые наборы кортежей, то в отношении возникает или лишь ФЗ, или МЗ, или ЗПС. Поскольку ДЗ – частный случай и МЗ, и ЗПС, то справедливо следующее утверждение: если отношение обладает ДЗ, оно автоматически обладает и МЗ, и ЗПС. Очевидно, что обратное – неверно. Поэтому само отношение R является обобщающим для отношения R', обладающего только МЗ (а значит и ЗПС), а также для отношения R'', обладающего только ЗПС.

Тогда для отношения R, содержащего ДЗ, справедливы следующие свойства.

Теорема 2.4. ДЗ $X \to\to Y//Z$ выполняется для отношения $R(X,Y,Z)$ тогда и только тогда, когда R является соединением без потерь проекций по всем своим атрибутам $R_1(X)$, $R_2(Y)$ и $R_3(Z)$.

Доказательство проведем по аналогии с [169].

Необходимость. Пусть имеется ДЗ $X \to\to Y//Z$. Докажем, что декомпозиция отношения R на проекции $R_1(X)$, $R_2(Y)$ и $R_3(Z)$ является декомпозицией без потерь. Нужно доказать, что $R = R_1(X) x R_2(Y) x R_3(Z)$ для *любого состояния* отношения R. Пусть кортеж $r = (x,y,z) \in R_1(X) x R_2(Y) x R_3(Z)$. Это означает, что в проекции $R_1(X)$ содержится кортеж $r_1' = x$, в проекции $R_2(Y)$ содержится кортеж $r_2' = y$, а в проекции $R_3(Z)$ кортеж $r_3' = z$. По определению проекции, найдется такое значение z_1 атрибута Z, что

отношение R содержит кортеж $r_1 = (x,y,z_1)$. Аналогично, найдется такое значение y_1 атрибута Y, что отношение R содержит кортеж $r_2 = (x,y_1,z)$. Тогда по определению ДЗ кортеж $r = (x,y,z) \in R$.

Достаточность. Пусть декомпозиция отношения R на проекции $R_1(X)$, $R_2(Y)$ и $R_3(Z)$ является декомпозицией без потерь. Докажем что отношение содержит ДЗ $X \to\to Y//Z$. Предположим, что отношение R содержит кортежи $r_1 = (x,y,z_1)$ и $r_2 = (x,y_1,z)$. Необходимо доказать, что кортеж $r_3 = (x,y,z)$ также содержится в R. По определению проекций, кортеж $r_1' = x$ содержится в $R_1(X)$, кортеж $r_2' = y$ содержится в $R_2(Y)$, а кортеж $r_3' = z$ содержится в $R_3(Z)$. Тогда кортеж $r_3 = (x,y,z)$ содержится в декартовом произведении $R_1(X) x R_2(Y) x R_3(Z)$. А в силу того, что декомпозиция является декомпозицией без потерь, этот кортеж содержится и в R.□

Теорема 2.5. Отношение $R(X,Y,Z)$, в котором содержится ДЗ $X \to\to Y//Z$, является соединением без потерь всех типов проекций: или проекций только по всем своим атрибутам $R_1(X)$, $R_2(Y)$, $R_3(Z)$, или двух проекций только по паре атрибутов $R_4(X,Y)$, $R_5(X,Z)$, или проекций по всем парам атрибутов $R_4(X,Y)$, $R_5(X,Z)$, $R_6(X,Z)$. Доказательство не приводим в связи с его очевидностью.

Поэтому в дальнейшем, говоря о разных типах многозначности в кортежах отношений без потребности конкретизации их, будем употреблять запись ДЗ (МЗ, ЗПС), что означает один из видов зависимости – ДЗ, МЗ, ЗПС.

Применяя следующее небольшое обобщение теоремы 2.1. о шунтировании ДЗ (МЗ, ЗПС) [205], проектировщик избавляется от аномалий в указанном отношении R.

Теорема 2.6. Если в отношении $R(X,Y,Z)$ имеется нетривиальная ДЗ $X \to\to Y//Z$, причем X, Y, Z могут быть составными множествами, то есть $X = \{X\}, Y = \{Y\}, Z = \{Z\}$, то при добавлении в это отношение дополнительных

неключевых атрибутов $\{A\}$, причем так, что $(X+Y+Z)\to A$, зависимость в отношении $R(X,Y,Z,A)$ становится функциональной.

Доказательство полностью аналогично приведенному выше доказательству теоремы 2.1. Из доказанного следует, что если отношение имеет ДЗ (МЗ, ЗПС), ее можно «шунтировать», то есть, перекрыть ее влияние добавлением нового элемента – естественного или суррогатного атрибута, полностью зависимого от всей конкатенированной строки, «физический» смысл которого - характеристика связи.

Поэтому в дальнейшем будем называть эти доказанные утверждения «теоремой о шунтировании ДЗ (МЗ, ЗПС)». А отношения с «перекрытой» ДЗ (МЗ, ЗПС) – шунтированными отношения.

Заметим, что аномалия в понимании Р. Фейджина [377] заключается в том, что после соединения проекций появляются избыточные кортежи или пропадают, что названо «потерей информации». Тем самым обеспечивается корректность работы SQL-запроса к схеме БД. А пользователь мотивируется декомпозировать отношения, моделирующие многоарные связи ПрО.

Это означает, что аномалия «потери информации» [377, 378] в результирующих отношениях, которые построены на обобщениях ДЗ, - не что иное, как результат некорректных соединений при выполнении запросов пользователей. Но использование таких соединений может быть минимизировано.

И как будет показано ниже, аномалии вставки и удаления в совокупности отношений-связей и отношений-сущностей не возникают лишь при определенных ограничениях. Это значительно уточняет выводы серии работ авторов [2, 294].

2.8. Связи в схемах баз данных

На основании теоремы 2.1. приведем без доказательства несколько очевидных утверждений.

Лемма 2.3. Отношение $R(X_j, A_i)$, полученное шунтированием ДЗ (МЗ, ЗПС) в ключевых атрибутах X_j так, что $\{X_j \rightarrow A_i,\ j=2,\ldots,J;\ i=1,\ldots,I\}$, моделирует связь арностью j и степенью $G:H$.

Поднабор A_i составляют атрибуты связи, моделирующие специфику ПрО. Действительно, если есть поднабор атрибутов связи A_i, то существует и сама связь.

Лемма 2.4. Если в отношении $R(X_j)$ есть только набор ключевых атрибутов и для $j \geq 3$ в R содержится ДЗ (МЗ, ЗПС), отношение $R(X_j)$ неактуально для ПрО.

Иными словами, такое отношение $R(X_j)$ не моделирует ни сущности-объекта, ни связи. Его можно либо исключить из совокупности отношений, либо шунтировать атрибутами. Именно найденные в ПрО атрибуты придадут семантику такому отношению.

Отметим, что в практике проектирования схем БД используют такие «недозаполненные» отношения для отражения лишь вероятности факта связи сущностей-объектов. Но отсутствие в ПрО несуррогатных атрибутов такой связи говорит о возможных аномалиях использования таких отношений.

Лемма 2.5. Отношение $R(X_j, A_i)$, полученное шунтированием ДЗ (МЗ, ЗПС) в ключевых атрибутах X_j так, что $\{X_j \rightarrow A_i,\ j=2,\ldots,J;\ i=1,\ldots,I\}$, является подобным особому отношению с составным ключом X_j и тем самым моделирует некоторую сущность-объект с атрибутами A_i.

Такую сущность-объект считают виртуальной и называют «постсвязной» [300] (построенной по «отглагольному существительному»).

Заметим, что леммы 2.3 и 2.5. показывают единство категорий связи и сущности-объекта, что согласуется с идеей П. Чена [359]. Это вполне закономерно, поскольку обе категории моделируются совокупностью атрибутов, а они инвариантны ПрО. Поэтому метод «сущностей и связей» [359] можно рассматривать как метод «связей».

2.9. Математический анализ реляционного каркаса

2.9.1. Булеан связей и РК

Отметим особенности каркасной совокупности отношений, обладающей эффективностью в части модифицируемости их схем.

Пусть $\{a_i\}$ - конечное множество атрибутов атомарных сущностей-объектов в смысле [232]. Там под атомарностью понимается свойство недекомпозируемости (отсутствие внутренней структуры) сущности-объекта, построенной на атомарном (одноместном или многоместном) предикате [25, 232]. Комбинирование атрибутов a_i порождает полную совокупность $\{r_l\}$ схем, представляющих именованные отношения. В свою очередь, комбинирование схем r_i порождает полную совокупность схем $\{D_j\}$ БД:

$$\{a_i\} \Rightarrow \{r_l\} \Rightarrow \{D_j\} \qquad (2.1)$$

Выражение (2.1) представляет собой ядро каркасной схемы. Следует отметить ключевое отличие между каркасной схемой и ER/SOM построениями: для заданного множества атрибутов в ER/SOM-моделях никогда не задействованы *полные* совокупности отношений $\{r_l\}$ и схем $\{D_j\}$, а только их подмножества. В этом и состоит локальность ER/SOM моделей. Такая локальность объясняется тем, что выбор атрибутов и конструирование «начальных» отношений (и их совокупностей) при «трансляции» из ER/SOM моделей определяются семантикой соответствующей ПрО. Другими словами, ER/SOM-конструкции являются решениями частных задач.

В силу этого полная совокупность $\{D_j\}$ в разделе 2.4. названа РК. Он может быть также носителем любых схем БД, которые могут быть построены на фиксированном множестве атрибутов $\{a_i\}$ при любых зависимостях между атрибутами. Обсуждение понятия «РК схем БД» и логические предпосылки к его введению имеется в работах [204, 206] .

Пусть в экземплярах R_i любого из отношений r_l каждый атрибут a_i принимает значения $[a_i]_p$ из непустого множества (домена) A_i, $\forall p : [a_i]_p \in A_i$. Здесь индекс p идентифицирует любое из возможных значений атрибута в

домене A_i, причем $\max p(A_i)$ характеризует алгебраическую мощность домена A_i, т.е. является его кардинальным числом.

Рассмотрим преобразование $\{a_i\} \Rightarrow \{r_l\}$. В отличие от традиционного синтеза схем r_l как декартовых произведений *отдельных* доменов, преобразование $\{a_i\} \Rightarrow \{r_l\}$ приводит к построению полной совокупности схем.

На первом шаге, для заданной ПрО формируется множество $\{a_i\}$ атрибутов и множество $\{A_i\}$ соответствующих доменов. Как правило, атрибуты просто перечисляются, а соответствующие им домены соотносятся с определенными типами, формами, определяющими свойствами. Каждый новый элемент в домене индексируем. Атрибуты и их домены индексируются (i) в произвольном порядке. При этом:

$$\forall A_i \; \bar{\exists} p^* \neq p : [a_i]_{p^*} = [a_i]_p , \qquad (2.2)$$

т.е. в каждом домене все значения атрибутов уникальны.

В процессе разработки и сопровождения БД состав множества атрибутов $\{a_i\}$, равно как и состав отдельных доменов A_i из (2.2), может изменяться. Подобная модификация может быть обусловлена как выделением в ПрО дополнительных атрибутов и новых сущностей-объектов, так и исключением атрибутов и сущностей-объектов, ставших неактуальными. Поэтому удобно говорить о *состоянии* схемы БД – совокупности $\langle \{a_i\} \; \{A_i\} \rangle$, актуальной в некоторый момент времени. Заметим, что схема БД «эволюционирует» от состояния к состоянию за счет изменения $\{a_i\}$, но не $\{A_i\}$. Модификация самой БД (а не ее схемы) - добавление новых кортежей, изменение или удаление существующих - может привести к изменению лишь кардинальных чисел отдельных доменов A_i, а также к изменению нормальной формы, в которой находится то или иное отношение.

Понятие зависимости между атрибутами (значит, и понятие соответствия нормальным формам) относится только к *экземплярам* схем, но не к самим схемам. Заметим, что на этой стадии построения РК $\{D_j\}$ никакие

сведения о зависимостях между атрибутами *не учитываются*. Это упрощает процедуру синтеза.

Далее конструируется полная совокупность $\{r_l\}$ схем. Традиционно отношения интерпретируются как подмножества декартовых произведений доменов [491]. Итак, пусть имеются некоторые множества A и B, имеющие кардинальные числа p_A и p_B соответственно. Пусть оператор синтеза РК L, определенный в разделе 2.4., действует следующим образом:

$$\{C_k\} = L_A(B),\ C_k = B \cup a,\ a \in A \setminus B. \qquad (2.3)$$

Тогда он продуцирует индексированную совокупность $\{C_k\}$ множеств C_k, содержащих все элементы множества B и один из элементов множества A, не принадлежащий B. Здесь и далее множество A назовем источником, множество B - базовым, а оператор L синтеза РК будем просто называть оператором синтеза (при использовании аббревиатуры, например, ОС могут возникать неверные ассоциации).

Заметим, что базовое множество может представлять собой *совокупность* множеств (в этом случае оператор синтеза действует на каждое из множеств базовой совокупности). Поясним эту процедуру на примерах.

Пример 2.1. Для источника $A = \{1,2,3,4,5,6,7\}$ и базового множества $B = \{1,3,5,6,8\}$ получаем

$$L_A(B) = L_A\{1,3,5,6,8\} = \{\{1,2,3,5,6,8\},\{1,3,4,5,6,8\},\{1,3,5,6,7,8\}\},$$

$$L_A^2(B) = L_A(L_A(B)) = L_A\{\{1,2,3,5,6,8\},\{1,3,4,5,6,8\},\{1,3,5,6,7,8\}\} = \\ = \{\{1,2,3,4,5,6,8\},\{1,2,3,5,6,7,8\},\{1,3,4,5,6,7,8\}\},$$

$$L_A^3(B) = L_A(L_A^2(B)) = L_A\{\{1,2,3,4,5,6,8\},\{1,2,3,5,6,7,8\},\{1,3,4,5,6,7,8\}\} = \{1,2,3,4,5,6,7,8\},$$

$$L_A^4(B) = L_A(L_A^3(B)) = L_A\{1,2,3,4,5,6,7,8\} = \{1,2,3,4,5,6,7,8\} = L_A^3(B).$$

Рассмотрим основные свойства оператора L. Прежде всего, оператор синтеза никогда не уменьшает мощность базового множества. Если $A \setminus B \neq \varnothing$, то мощность базового множества возрастает на единицу, что дает мощность любого из множеств C_k совокупности $\{C_k\}$, т.е. $\forall k: p(C_k) = p_B + 1$. Базовое множество является подмножеством любого из порожденных C_k. Фактически

оператор синтеза обеспечивает «перекачку» элементов от источника к базовому множеству. В силу того, что реальный источник предполагается конечным (т.е. p_A является конечным числом), m-кратное действие $L_A^m(B)$ оператора синтеза всегда приводит к «насыщению» базового множества. Этот эффект выражается в том, что $C_k \to B$ при $A \setminus B = \varnothing$, т.е. когда базовое множество уже содержит источник ($A \subseteq B$; источник «исчерпан»). Пороговым является значение $M = p(A \setminus B)$. При $A \setminus B = \varnothing$ получаем $M = p(\varnothing) = 0$, что указывает на «насыщение» базового множества для всех $m \geq M$.

Важны две особенности оператора L: во-первых, оператор синтеза позволяет получить все возможные комбинации атрибутов a_i, так как в алгебраическом смысле РК является булеаном источника A, а во-вторых, полная совокупность этих комбинаций оказывается структурированной.

Действительно, каждое m-кратное действие $L_A^m(B)$ порождает совокупности $\{C_i\}_m$ множеств C_i, имеющих равные кардинальные числа, но взаимно отличающихся (если множеств C_i оказывается более одного) не более чем $m-1$ элементами. Всякую совокупность $\{C_i\}_m$ назовем m-уровнем. Ясно, что множества C_i каждого m-уровня можно идентифицировать значением m, например, индексами элементов a_i источника A. Если схема получаемой совокупности $\{C_i\}_m$ множеств несущественна, можно представить ее в виде $\{C_k\}$ простым перечислением порожденных множеств.

Пример 2.2. Пусть элементы рассмотренного выше источника $A = \{1,2,3,4,5,6,7\}$ проиндексированы следующим образом: $a_1 = 2$, $a_2 = 5$, $a_3 = 1$, $a_4 = 3$, $a_5 = 6$, $a_6 = 4$, $a_7 = 7$ (здесь содержание элементов не имеет значения). Для базового множества $B = \{1,3,5,6,8\}$ получаем трехуровневую совокупность множеств, а именно:

1) $L_A(B)$ порождает тройку

$$C_1 \equiv C(1) = \{1,2,3,5,6,8\} = B \cup a_1,$$

$$C_2 \equiv C(6) = \{1,3,4,5,6,8\} = B \cup a_6,$$

$$C_3 \equiv C(7) = \{1,3,5,6,7,8\} = B \cup a_7;$$

2) $L_A^2(B)$ порождает тройку

$$C_4 \equiv C(1,6) = \{1,2,3,4,5,6,8\} = B \cup \{a_1, a_6\} = C(1) \cup a_6 = C(6) \cup a_1,$$

$$C_5 \equiv C(1,7) = \{1,2,3,5,6,7,8\} = B \cup \{a_1, a_7\} = C(1) \cup a_7 = C(7) \cup a_1,$$

$$C_6 \equiv C(6,7) = \{1,3,4,5,6,7,8\} = B \cup \{a_6, a_7\} = C(6) \cup a_7 = C(7) \cup a_6;$$

3) $L_A^3(B)$ порождает единственное множество

$$C_7 \equiv C(1,6,7) = \{1,2,3,4,5,6,7,8\} = B \cup \{a_1, a_6, a_7\} =$$
$$= C(6,7) \cup a_1 = C(1,7) \cup a_6 = C(1,6) \cup a_7 =$$
$$= B \cup (A \setminus B) = A \cup B,$$

после чего наступает состояние «насыщения» базового множества.

Ясно, что каждое «дочернее» множество может иметь несколько «родительских» множеств. Как правило, мощность $A \setminus B$, велика (т.е. велико число атрибутов, выделяемых в ПрО), поэтому обычно граф, отражающий эту взаимосвязь между множествами, не будет планарным. Весь формализм теории графов (связность, диаметр, маршруты и т.п.), естественно, применим для конструируемой совокупности множеств.

Рассмотрим, как можно использовать оператор синтеза L при построении схем БД.

Пусть базовое множество B пусто ($B = \varnothing$ и $p_B = 0$), а источник A – индексированная совокупность $\{a_i\}$ атомарных атрибутов ($p_A = N$, где N - количество сущностей-объектов, выделяемых в ПрО). Имеем $A \setminus B = A$ и $\{C_i\} = L_A(\varnothing) \equiv L_A^1$, $C_i = a_i \in A$. Таким образом, в случае однократного применения оператора синтеза (L^1) пустое базовое множество можно не учитывать, рассматривая лишь совокупность атрибутов A. Отметим, что при многократном действии оператора синтеза ($L^{m>1}$) аргументом будет совокупность множеств, полученных в результате предыдущих действий оператора. Следовательно, сам оператор синтеза определен индуктивно.

Итак, действие L_A^1 для пустого базового множества порождает $S_1 = N$ унарных отношений $\{a_i\}$, однозначно соотнесенных с атрибутами a_i из источника A. Действие L_A^2 оперирует совокупностью $\{a_i\}$ как аргументом, порождая элементы 2-уровня, т.е. все возможные бинарные отношения. При этом $B = L_A^1$ и $S_2 = \frac{1}{2}(N-1)N$. Далее, совокупность бинарных отношений является аргументом для действия L_A^3, которое порождает элементы 3-уровня, т.е. все возможные тернарные отношения. Аналогично, $B = L_A^2$ и $S_3 = \frac{1}{6}(N-2)(N-1)N$. Вообще, m-уровень $(m \le N)$,синтезированный действием L_A^m с базовой совокупностью $B = L_A^{m-1}$ в качестве аргумента, содержит $S_m = \frac{N!}{m!(N-m)!}$ m-арных отношений. Как отмечено выше, общее число отношений в синтезируемом каркасе составляет:

$$S(N) = \sum_{m=1}^{N} S_m \qquad (2.4)$$

Т.е., при $m = N$ получаем $S_N = 1$, что указывает на «исчерпывание» совокупности атрибутов A. Например, $S(5) = 31$, $S(10) = 1023$, $S(20) = 1048575$, $S(30) = 1073741823$. Очевидно, что в практически значимых случаях, когда в ПрО выделяются десятки и сотни атрибутов сущностей-объектов, РК не может быть реализован целиком. РК как естественным образом структурированная совокупность множеств атрибутов является аналогом множества «состояний» физической системы, в пределах которого разворачивается весь процесс синтеза схемы БД. Для схемы, состоящей из $R \le S(N)$ схем $\{r_i\}$, удобно ввести понятие *заполнения* ξ_R РК:

$$\xi_R = \frac{R}{S(N)} = \frac{R}{2^N - 1}. \qquad (2.5)$$

Как правило, $\xi_R << 1$. Причем вопрос о распределении $\xi_R(N)$ (2.5) для реальных БД оставляем открытым.

2.9.2. Теорема о полноте и единственности РК

Расширим и переформулируем теорему 2.2. как теорему о полноте и единственности.

Теорема 2.7. РК, порождаемый оператором синтеза на конечном множестве унарных атрибутов, единственен и полон.

Доказательство.

Полнота РК. Пусть заданы источник A и базовое множество B. Пусть существует такое множество $C=\{c_i\}$, что $C \supseteq B$ и $\exists c \in A \setminus B$, но $\forall m: C \neq L_A^m(B)$. Тогда из определения оператора синтеза (2.3), $C \neq B \cup C_m^*$, где $\forall m: C_m^* \subseteq A \setminus B$, а значит $\exists c \in C_m^*$. Но это противоречит допущению $\exists c \in A \setminus B$. Следовательно, действие $L_A^m(B)$, порождающее множество C, действительно существует. Поэтому для РК имеем $R = S(N)$, т.е. его заполнение $\xi_R = 1$.

Единственность РК. Доказательство основано на его полноте. Пусть имеется пара РК $\{C_k\} = L_A^m(B)$ и $\{C_k^*\} = L_A^m(B)$. В силу полноты каждого из РК имеем $\{C_k\} \setminus \{C_k^*\} = \varnothing$ и $\{C_k^*\} \setminus \{C_k\} = \varnothing$, следовательно, $\{C_k\} = \{C_k^*\}$, т.е. каркасы идентичны или, другими словами, на конечном множестве атрибутов сущностей-объектов оператор синтеза порождает единственный РК. □

Единственность и полнота совокупности отношений, порождаемых оператором L_A из множества A атрибутов сущностей-объектов, выделяемых в ПрО, указывает на исключительное положение РК в процессе разработки схемы БД. Действительно, любое отношение на заданном множестве атрибутов принадлежит РК. Более того, результат операций над элементами РК также будет элементом РК. Это свойство замкнутости непосредственно следует из свойства полноты. Поэтому РК является вполне естественным (хотя и формальным) компонентом при построении унифицированного алгоритма синтеза схем БД. Отметим, что сам РК как структурированная совокупность всех возможных множеств атрибутов предельно избыточен ($\xi = 1$), поэтому для его эффективного использования необходим инструментарий «навигации по РК», преобразований, редукций, локализаций и оперирования в целом.

Итак, преобразование $\{a_i\} \Rightarrow \{r_l\}$ в каркасной схеме заключается в том, что заданное множество атрибутов порождает структурированную

совокупность отношений. Следующее преобразование каркасной схемы, а именно $\{r_l\} \Rightarrow \{D_j\}$, состоит в синтезе всех возможных схем БД. Ясно, что для этого можно использовать оператор синтеза L (2.3) с полной совокупностью отношений в качестве аргумента.

Пусть базовое множество B пусто ($B = \varnothing$ и $p_B = 0$), а источником A является индексированная совокупность $\{r_l\}$ отношений ($p_A = S(N) = 2^N - 1$ - полное число синтезированных отношений). Тогда действие L_A^1 порождает совокупность схем БД, образованных простейшими отношениями (фактически это – каркас отношений, рассмотренный выше), действие L_A^2 - всеми возможными парами отношений, действие L_A^m - всеми возможными m-компонентными совокупностями отношений ($m \le p_A$). Общее число схем РБД составит $S_D = 2^{S(N)} - 1 = 2^{2^N - 1} - 1$. Естественно, что всякая РБД имеет в основе единственную схему, с необходимостью являющуюся элементом РК схем. Это объясняется тем, что согласно теореме о полноте и единственности РК совокупность схем БД, порождаемая оператором синтеза, также обладает свойствами единственности и полноты.

Пример 2.3. Пусть в ПрО выделены 3 атомарных атрибута a, b, c. Совокупность этих атрибутов образует источник A. РК отношений состоит из $2^3 - 1 = 7$ отношений, а именно: a, b, c (L_A^1, т.е. унарные отношения), ab, bc, ac (L_A^2, т.е. бинарные отношения), abc (L_A^3, т.е. единственное тернарное отношение). РК схем БД состоит из $2^7 - 1 = 127$ элементов, синтезируемых из 7-элементного РК отношений как аргумента оператора синтеза L, например, a, abc (для действия L_A^1), $a + ac$, $abc + ac$, $ac + ab$ (для действия L_A^2), $abc + a + bc$, $a + b + c$, $ab + ac + b$ (для действия L_A^3).

РК схем БД также занимает исключительное положение при построении унифицированной каркасной схемы. Действительно, *любая* схема БД, синтезируемой для заданного множества атрибутов, является элементом этого РК. Поэтому решение любой задачи синтеза или задачи модификации уже содержится в РК схем БД.

2.9.3 Топология путей нормализации РК

Выбранная для оператора L начальная схема $D^{(0)}$, как правило, требует нормализации в силу наличия избыточных ФЗ и/или ДЗ (МЗ, ЗПС) между отдельными атрибутами и/или их совокупностями. По сути, нормализация является преобразованием от начальной схемы $D^{(0)}$ БД (как совокупности схем БД) к некоторой конечной схеме $D^{(k)}$, каждое из отношений которой удовлетворяет известным критериям нормализации. В большинстве случаев (для нетривиальных множеств атрибутов и нетривиальных зависимостей между ними) нормализация представляет собой многошаговую процедуру, т.е., вначале нормализуется одно отношение, а затем другие. При этом вся схема БД, естественно, изменяется:

$$D^{(0)} \to D^{(1)} \to \ldots \to D^{(k)} \qquad (2.5)$$

Основной акцент при проектировании схемы БД ставится именно на нормализацию, т.е. на получение конечной схемы $D^{(k)}$, все отношения которой находятся в требуемой НФ. Поскольку для каркасной схемы совокупность $\{D_j\}$ является полной (т.е. представляет собой множество всех возможных комбинаций всех возможных схем r_l, построенных комбинированием атрибутов a_i), она уже содержит необходимую конечную схему $D^{(k)}$. Таким образом, объект исследования – РК схем $\{D_j\}$ и его свойства.

Классический способ нормализации – это декомпозиция отношений. Суть декомпозиции состоит в замене каждой схемы r^*, не удовлетворяющей критериям требуемой НФ, на другие схемы $\{r_l\} \sim r^*$, в совокупности эквивалентные исходной схеме r^*, но по отдельности соответствующие критериям НФ. Всю совокупность схем БД в РК $\{D_j\}$ удобно проиндексировать. Любую нормализацию (2.5), а значит и соответствующую декомпозицию отношений, идентифицируем последовательностью $Q(j_0, j_1, \ldots, j_k)$ индексов схем БД, которую далее назовем *путем нормализации*. Для изучения свойств РК $\{D_j\}$ целесообразно проанализировать топологию

путей Q. Метод такого исследования – анализ топологии путей нормализации в РК$\{D_j\}$.

Каждый путь нормализации определяется двумя факторами: начальной схемой $D^{(0)}$ БД, а также зависимостями в отношениях начальной схемы (и, возможно, последующих схем). Эти ФЗ / ДЗ (МЗ, ЗПС), как правило, тесно связаны с ключами (унарными либо составными), которые можно выделить в экземплярах отношений. Поскольку в общем случае всякое нетривиальное отношение может содержать несколько зависимостей (равно как и несколько ключей), для каждой начальной схемы можно предположить наличие нескольких путей нормализации. Далее, не для всяких начальной $D^{(0)}$ и конечной $D^{(k)}$ схем может существовать путь нормализации. Для нетривиальных зависимостей на множестве атрибутов $\{a_i\}$ топология путей нормализации в РК $\{D_j\}$ может оказаться достаточно сложной. Заметим, что рассматриваемое начальное подмножество $D^{(0)}$ является элементом РК схем БД.

Предположим, что зная топологию путей нормализации (например, особенности влияния мощностей отношений в РК на принадлежность отношений заданным нормальным формам, связь между множеством зависимостей в некотором отношении и областью в РК, куда приведет декомпозиция этого отношения и т.п.) удастся существенно упростить процедуры построения и модификации конечных схем БД. Во-первых, за счет использования новых унифицированных методик дизайна этих схем. И, во-вторых, за счет эксплуатации тех топологических особенностей унифицированной схемы, которые ранее были совершенно незаметны при решении частных задач. Например, введение адекватной меры $\|Q\|$ для путей нормализации позволит говорить о выборе оптимальных путей нормализации схем БД.

Рассмотрим пути нормализации РК. Пусть дана индексированная совокупность схем $\{C_k\}$, $k = 1,2,...,S(N)$, образующих РК отношений для

множества из N сущностей-объектов в некоторой произвольной ПрО. Внесение реального содержания в РК отношений (т.е. заполнение схем сведениями о значениях реальных экземпляров) приводит к формированию совокупности экземпляров отношений. Обозначим экземпляр отношения $\{C_k\}$ символом $[C_k]$. Для каждого из экземпляров $[C_k]$ говорим о множестве Φ_k ФЗ либо ДЗ (МЗ, ЗПС) между атрибутами (или множествами атрибутов) соответствующего отношения $\{C_k\}$. Как правило, множество зависимостей Φ_k считается независимым от экземпляра $[C_k]$, т.е. любая модификация $[C_k]$ не меняет Φ_k.

Пример 2.4. Пусть для источника $A=\{a,b,c,d\}$ ($N=4$) сформирован РК, состоящий из $2^4-1=15$ отношений $\{C_k\}$, и пусть образованы соответствующие экземпляры $[C_k]$ этих отношений. Множество зависимостей Φ_k может состоять, например, из: ФЗ между атрибутами отношений 1-уровня (для действия L_A^1), т.е. из $a \to a$, $c \to c$, которые всегда будут тривиальными; из зависимостей вида $a \to b$, $c \to\!\!\to d$ между атрибутами отношений 2-уровня (для действия L_A^2); из зависимостей вида $ab \to c$, $a \to\!\!\to bd$ между атрибутами отношений 3-уровня (для действия L_A^3) и т.д. Важно, что все элементы множества зависимостей Φ_k между атрибутами (и множествами атрибутов) отношения можно перечислить комбинаторными методами, вводя тем самым полное множество зависимостей Φ_k и его подмножество $\tilde{\Phi}_k \subseteq \Phi_k$, актуальное для конкретного экземпляра $[C_k]$ отношения $\{C_k\}$.

Рассмотрим отношение $\{C_k\}$ и его экземпляр $[C_k]$ с зависимостями Φ_k. Пусть $K=\{K_1,K_2,K_3,K_4,K_5,\ldots\}$ - упорядоченная совокупность традиционных НФ, а также всех возможных их модификаций. Пусть ψ - множество классических критериев, относящих C_n к НФ из совокупности K:

$$\psi_n = \psi(C_n),\ \psi_n \in K. \qquad (2.6)$$

Учитывая взаимосвязь $K_1 \subset K_2 \subset K_3 \subset \ldots$ между НФ из совокупности K, обозначим через $\Psi_k = \max \psi_k$ наивысшую НФ, в которой находится экземпляр $[C_n]$ отношения $\{C_n\}$.

Состоянием схемы $D^{(0)}$ назовем совокупность $\{\Psi_k\}$ НФ экземпляров $[C_k]$ всех отношений $C_k \in D^{(0)}$. Ясно, что НФ, в которой находится схема $D^{(0)}$ (в общем случае весь РК отношений), определяется величиной $\max\{\Psi_k\}$. Именно поэтому все отношения схемы БД приводятся (как правило, путем декомпозиции) к одной, наибольшей НФ.

В результате нормализации (2.5) получаем последовательность элементов РК схем БД, которую можно интерпретировать как путь нормализации. Выше уже указано, что формально путь нормализации $Q(j_0, j_1, ..., j_k)$ – это последовательность индексов схем БД, которая описывает переход от начального элемента $D^{(0)}$ к конечному $D^{(k)}$ РК схем БД (2.5), причем этот переход осуществляется только декомпозицией отношений, принадлежащих элементам пути нормализации.

Конечный элемент $D^{(k)}$ пути нормализации назовем *решением*. Далее, *топологией* путей нормализации для заданной начальной схемы $D^{(0)}$ назовем совокупность решений $D^{(k)}$, которые можно получить путем декомпозиции при заданных зависимостях $\{\Phi_k\}$ между атрибутами отношений. Ясно, что топологию будут определять как $D^{(0)}$, так и $\{\Phi_k\}$.

Тогда определенный ранее оператор синтеза L позволяет сформулировать еще одно важное свойство РК в виде следующей теоремы, непосредственно следующей из полноты РК.

Теорема 2.8. (о замкнутости путей нормализации). Для заданных источника A и совокупности $\{\Phi_k\}$ зависимостей между атрибутами в экземплярах $[C_k]$ РК отношений, а также для заданной упорядоченной совокупности НФ $K = \{K_1, K_2, K_3, K_4, K_5, \ldots\}$ существует путь нормализации $D^{(0)} \to \ldots \to D^{(k)}$, решение которого находится в требуемой НФ $\max\{\Psi_k\}$ из совокупности K.

Доказательство.

Для доказательства вновь рассмотрим пример 2.4 - источник $A=\{a,b,c,d\}$ $(K=4)$, для которого сформирован РК из 15 отношений $\{C_k\}$. Пусть $D^{(0)}$ - начальная схема БД, содержащая среди прочих единственное отношение $C=\{abcd\}$ 4-уровня (т.е. синтезированное действием L_A^4), $C\in D^{(0)}$. Пусть семантика ПрО такова, что для экземпляра $[C]$ выполняется следующее множество Φ функциональных и МЗ: $abc\to d$, $a\to\to b$, $a\to\to c$. Каким может быть решение $D^{(1)}$ для простейшего, одношагового пути нормализации?

Поскольку все атрибуты из A являются атомарными, то $\psi(C)=K_1$, т.е. C находится в 1НФ. Далее, для функциональной зависимости $abc\to d$ множество атрибутов abc является суперключом, поэтому $\psi(C)=\{K_1,K_2,K_3,K_4\}$, т.е. C находится также в 2НФ, 3НФ и НФБК. Поскольку для каждой из МЗ $a\to\to b$ и $a\to\to c$ в отношении C ни a ни b не являются суперключами, $\psi(C)\neq K_5$, т.е. C не находится в 4НФ. Поэтому $\Psi=K_4$. Для дальнейшего увеличения Ψ на единицу, т.е. для получения 4НФ, можно было бы потребовать, чтобы в отношениях искомого решения $D^{(1)}$ отсутствовали МЗ, такие как $a\to\to b$ и $a\to\to c$.

Заметим, что МЗ может существовать для отношений не ниже 3-уровня (т.е. синтезированных действиями $L_A^{m\geq 3}$), т.е. когда отношение имеет минимум 3 атрибута. В силу насыщения оператора синтеза наибольшим уровнем является 4-й, где и находится отношение C. Ясно, что решение $D^{(1)}$, для которого $\Psi=K_5$ (4НФ), не должно содержать отношения C. Можно попробовать редуцировать схему $D^{(1)}$ до совокупностей отношений, порождаемых действиями $L_A^{m<3}$, т.е. применить т.н. ограничение РК отношений сверху. Выполняя декомпозицию отношения C, получаем

$$\{abcd\}\mapsto\{ab\}\cup(\{acd\}\mapsto\{ac\}\cup\{ad\})=\{ab\}\cup\{ac\}\cup\{ad\},$$

т.е. производится замена отношения 4-уровня на совокупность отношений 2-уровня, которые и будут присутствовать в решении $D^{(1)}$. При этом для

отношений $\{ab\}$ и $\{ac\}$ МЗ $a \to\to b$ и $a \to\to c$ будут уже тривиальными, т.е. критерий 4НФ будет выполняться. □

Из примера 2.4 также видно, что исходное отношение C 4-уровня $\{abcd\}$ после декомпозиции на совокупность отношений 2-уровня уже не содержит ФЗ $abc \to d$, т.е. информация об этой зависимости для $\{ab\} \cup \{ac\} \cup \{ad\}$ теряется. Если под сохранностью информации понимать обеспечение соединения без потерь [85, 169, 300] и, одновременно, обеспечение сохранения зависимостей, то такую декомпозицию можно считать декомпозицией с потерей информации. Имеем тройку критериев, которые в случае корректной декомпозиции должны быть выполнены одновременно: возможность восстановления кортежей (т.е. обеспечение соединения без потерь), сохранение имеющихся зависимостей, а также соответствие критериям «целевой» НФ (в примере 2.4 это 4НФ) [358, 377].

Если между критериями из этой совокупности возникает противоречие, т.е. если невозможно одновременно выполнить хотя бы два критерия из трех, декомпозиция будет некорректной. При получении желаемой НФ такая некорректность может быть выражена либо искажениями при соединении дочерних отношений, либо искажениями в зависимостях между атрибутами.

2.10. Предикатная модель реляционного каркаса

Здесь и далее БД рассмотрена как структурированная совокупность. Используем логическую интерпретацию БД [368, 451] как естественно подходящую для синтеза универсальной структуры, вмещающей схемы БД.

Формально всякая сущность-объект является n-местным предикатом $P_j^n(x_i)$, принимающим истинное значение, $P_j^n(x_i) = 1$ [25, 351]. Здесь x_i – аргументы предиката P, взятые без конкретизации в качестве атрибутов сущностей-объектов ПрО и, тем более, без конкретизации каких бы то ни было взаимосвязей между ними. При этом аргументы x_i предикатов - так называемые предметные (индивидные) переменные [170]. Заметим, что появление в БД ложного предиката, $P_j^n(x_i) = 0$, недопустимо и может

указывать на некорректное функционирование БД. Например, если сама схема БД не нормализована и допускает аномалии вставки, модификации или удаления. Аргументом x_i предиката может быть любая переменная (или, в частном случае, постоянная) с единичной кардинальностью, выделяемая в ПрО. При этом не уточняется, относится ли семантически аргумент предиката к какому-либо атрибуту сущности-объекта.

Пусть семантика ПрО предполагает наличие K таких переменных x_i. Индекс $i = 1,...,K$ является идентификатором каждого аргумента x_i, вне зависимости от привязки к предикатам. Естественно, для предикатов $P_j^n(x_i)$ $\forall j : n \le K$. Например, $P_9^3(x_1,x_5,x_{17})$, $P_2^5(x_3,x_4,x_{11},x_{12},x_{108})$. Как и ранее, тут индекс $j = 1,...,N$ идентифицирует n-местный предикат $P_j^n(x_i)$ в схеме БД.

Очевидно, что для ПрО с K аргументами x_i возможно до 2^K предикатов, различных только по количеству аргументов: одноместных, двуместных, и т.д., и единственного K-местного предиката. Однако, предикаты из одной группы «вместимости» (арности) $P_A^3(x_1,x_2,x_3)$ и $P_B^3(x_1,x_2,x_3)$ могут быть высказываниями, имеющими отличную один от другого семантику. Например: «Человек X(x_1) в городе Y(x_2) владеет машиной с номером Z(x_3)» и «Человек X(x_1) в городе Y(x_2) пострадал в ДТП с участием машины с номером Z(x_3)». Однако, как правило, такие разные по смыслу, но одинаковые по схеме высказывания, содержащие одинаковые переменные, чаще всего относятся к разным ПрО. Тем не менее, ситуация, когда в БД отношения с одинаковыми наборами атрибутов, моделируют разные связи, встречается. Этот случай будет рассмотрен при проектировании ХД с помощью масок сущностей-объектов.

Такое построение соответствует методологии Дейта [85, 368], когда БД задана совокупностью истинных высказываний (но не «экземпляров» объектов), а оперирование БД интерпретируется как получение новых высказываний из имеющейся совокупности. При этом есть прямая аналогия с

РМД: всякий выполнимый предикат $P_j^n(x_i)$ является аналогом схемы R_j, а каждое истинное высказывание для этого предиката – это аналог кортежа в схеме R_j. Если для каждого из аргументов x_i предикатов $P_j^n(x_i)$ ввести свою область значений D_i, $x_i \in D_i$, аналогия с кортежами становится очевидной.

Здесь, естественно, допустима привязка к семантике ПрО: «Формулы предикатов имеют смысл только тогда, когда имеется какая-нибудь интерпретация входящих в них символов. Под *интерпретацией* мы понимаем всякую систему, состоящую из непустого множества D, называемого *областью* интерпретации, и какого-либо соответствия, относящего каждой предикатной букве P_j^n некоторое n-местное отношение в D, каждой функциональной букве p_j^n - некоторую n-местную операцию в D (т.е. $p_j^n : D^n \to D$) и каждой предметной переменной x_i - некоторый элемент из D. При заданной интерпретации предметные переменные x_i мыслятся пробегающими область D этой интерпретации, а… …всякое n-местное отношение в D может рассматриваться как некоторое подмножество множества D^n всех n-ок элементов из D» [170, с. 57]. При этом истинное высказывание $P_j^n(x_i) = 1$ в БД справедливо только для определенных значений аргументов предиката, т.е. предикат в БД не должен быть тавтологией.

Введем особый класс A предикатов, называемых семантически атомарными.

Определение 2.5. Многоместный предикат $P_j^n(x_i)$ является *семантически атомарным* (просто *атомарным*), если никакие подмножества его аргументов (и, в частном случае, отдельные аргументы) не фигурируют ни в каких зависимостях, как между собой, так и с подмножеством аргументов иных предикатов.

Здесь атомарность предиката можно интерпретировать как свойство, означающее отсутствие какой бы то ни было внутренней семантической структуры. Такое упрощение совершенно естественно, поскольку зависимости

целостности не влияют на проектирование схем БД. А их учет – его отдельные процедуры. В дальнейшем это определение будем также называть правилом *взаимной семантической атомарности.*

Формально, для подмножеств аргументов атомарного предиката нельзя написать непротиворечивую логическую формулу с помощью связок отрицания ($\neg$) и следования ($\supset$), а также кванторов существования ($\exists$) и всеобщности ($\forall$). Точнее, всякая формула на подмножествах аргументов семантически атомарного предиката должна представлять собой логическое противоречие, т.е. быть ложной для всех возможных значений аргументов. Другими словами, семантически атомарный предикат вообще не должен содержать непротиворечивых субпредикатов.

На языке схем отношение, эквивалентное семантически атомарному предикату, не может содержать зависимостей, кроме лишь ФЗ каждого атрибута (аргумента) от конкатенации всех. При этом единственным неизбыточным ключом такого отношения будет конкатенация атрибутов. Для атомарного предиката ${}_A P_j^n(x_i)$ это полная совокупность его аргументов. А это значит, что единственный ключ должен быть строго n-значным. Естественно, понятие атомарности как «отсутствия внутренней структуры» не является новым [418]. Но здесь используется атомарность в обобщенном смысле: атомарные предикаты могут быть n-местными ($n \geq 1$), но взаимодействовать - образовывать семантически значимые отношения - могут лишь как цельные сущности-объекты.

Таким образом, семантическим атомом ${}_A(x_i)_j$, назовем всякий факт в БД, справедливый для атомарного предиката, т.е. ${}_A P_j^n\left(x_i = {}_A(x_i)_j\right) = 1$. Как и ранее, индекс i в обозначении семантического атома – это глобальный (взят по всей ПрО) идентификатор аргумента, а индекс j - идентификатор предиката.

На языке схем БД семантический атом есть не что иное, как отдельный кортеж в отношении, эквивалентном атомарному предикату. Атомы ${}_A(x_i)_j$

каждого j-предиката ${}_A P_j^n(x_i)$ удобно проиндексировать. Соответствующий индекс ${}_A\alpha_j$ является ключом семантического атома (далее для простоты пишем α_j вместо ${}_A\alpha_j$). На языке схем БД это означает введение функциональной зависимости $(\alpha \to x_i)_j$, что эффективнее, поскольку ключ семантического атома является унарным (принимающим целые положительные значения), а ключ соответствующего атомарного предиката ${}_A P_j^n(x_i)$ – строго n-арным (причем далеко не всегда числовым).

Определение 2.6. *Ключом* семантически атомарного предиката $\alpha_j(x_i)$ назовем такой одноместный предикат (в дальнейшем – «ключевой предикат»), конъюнкция с которым семантически атомарного предиката равна истине и такой, что существует ФЗ $(\alpha \to x_i)_j$ для любых переменных x_i семантически атомарного предиката.

На языке отношений, это такой минимально достаточный унарный ключевой атрибут X_j, что в схеме отношения $R_j(X_j, A_j^m)$ возникает новый ключ $X_j \to A_j^m$. Тогда унарное отношение $R_j(X_j, A_j^m)$, эквивалентное семантически атомарному предикату, не содержит никаких неконтролируемых («паразитных») зависимостей.

Пусть ПрО, где выделено K аргументов x_i, допускает введение N атомарных предикатов ${}_A P_j^n(x_i)$, $j = 1,..,N$. Заметим, что за счет группировки аргументов ПрО в атомарные предикаты можно получить существенно меньшее число предикатов, чем изначально: хотя бы в силу того, что $2^N \le 2^K$ при $N \le K$; а при $N < K$ для $N, K >> 1$, $2^N << 2^K$. Получаем N ключей α_j соответствующих семантических атомов. Для описания как потенциальных, так и актуальных семантических связей между семантическими атомами введем по аналогии с разделом 2.5 каркасную схему $\mathrm{B}(\alpha)$, или 2^α:

α_j, $j = 1,..,N$;

$\alpha_{j_1} \times \alpha_{j_2}$, $j_1 \ne j_2$;

$\alpha_{j_1} \times \alpha_{j_2} \times \alpha_{j_3}$, $j_1 \neq j_2 \neq j_3$;

…

$\alpha_{j_1} \times \alpha_{j_2} \times \alpha_{j_3} \times ... \times \alpha_{j_N}$, $j_1 \neq j_2 \neq j_3 \neq ... \neq j_N$.

Схема 2^{α} есть множество всех подмножеств ключей семантических атомов. Универсальность схемы состоит в единообразии описания всех возможных отношений между семантическими атомами .

Естественно, реальная ПрО содержит только некоторое актуальное подмножество $2^{\alpha+} \subseteq 2^{\alpha}$ из всех возможных семантических связей. Это так называемая *актуальная часть* булеана ключей семантических атомов. Связи из $2^{\alpha+}$, не противоречащие семантике ПрО, назовем семантически актуальными, а все остальные - семантически потенциальными. Вся совокупность потенциально возможных связей будет описываться множеством $2^{\alpha-} = 2^{\alpha} \setminus 2^{\alpha+}$. Это - *потенциальная часть* булеана ключей семантических атомов. Технически такое разбиение можно реализовать индексацией элементов булеана 2^{α} с последующим введением бинарной переменной δ_m, принимающей значение 0 для потенциальной связи $\left(2^{\alpha}\right)_m$ и значение 1 для семантически актуальной связи $\left(2^{\alpha}\right)_m$.

Заметим, что в некотором смысле изложенный подход снимает дилемму Кодда о «выборе между единственным универсальным отношением и множеством бинарных отношений» [361, с. 468].

Рассмотрим, каким образом «межатомные» связи в их классической интерпретации, т.е. в виде ФЗ и ДЗ (МЗ, ЗПС), переносятся на множество $2^{\alpha+}$. В общем виде механизм такой трансляции связей заключается в следующем. Пусть заданы атомарные предикаты $P_A \in A$ и $P_B \in A$. Пусть между $x_A \subseteq (x_i)_A$ и $x_B \subseteq (x_i)_B$ существует (другими словами, актуальна) некоторая зависимость λ, т.е. $x_A \lambda x_B$. По определению ключа семантического атома имеем $(\alpha \to x_i)_A$ и $(\alpha \to x_i)_B$, следовательно, $\alpha_A \to x_A$ и $\alpha_B \to x_B$. Поэтому зависимость $x_A \lambda x_B$ между подмножествами аргументов предикатов P_A и P_B транслируется в

зависимость $\alpha_A \lambda \alpha_B \in 2^{\alpha+}$ между ключами α_A и α_B соответствующих семантических атомов.

Пусть $x_{A_1} \subseteq (x_i)_A$, $x_{A_2} \subseteq (x_i)_A$, $x_{A_1} \neq x_{A_2}$ и $x_B \subseteq (x_i)_B$, тогда $x_{A_1} \lambda x_B$ и $x_{A_2} \lambda x_B$. В этом случае имеем $\alpha_A \to x_{A_1}$, $\alpha_A \to x_{A_2}$ и $\alpha_B \to x_B$. Пара семантически различных зависимостей $x_{A_1} \lambda x_B$ и $x_{A_2} \lambda x_B$ может транслироваться в одну зависимость $\alpha_A \lambda \alpha_B \in 2^{\alpha+}$ между ключами α_A и α_B семантических атомов, т.е. в этом случае актуальная часть $2^{\alpha+}$ булеана 2^{α} не будет способна отобразить указанную пару зависимостей. Из правила взаимной семантической атомарности следует, что $x_A = (x_i)_A$ и $x_B = (x_i)_B$, и поэтому никаких затруднений с трансляцией зависимости λ на $2^{\alpha+}$ не возникает. Отсюда очевидно, что конструирование класса A семантически атомарных предикатов необходимо начинать с рассмотрения наименьших (по числу аргументов) детерминант зависимостей, выделенных в ПрО.

Синтез РК сводится к построению множества $2^{\alpha+}$ на булеане ключей семантических атомов и выражается следующей последовательностью действий:

1) в ПрО выделяется множество x_i аргументов предикатов;

2) формируется совокупность предикатов ${}_A P_j^n(x_i)$, семантически атомарных (т.н. базовое множество предикатов РК);

3) для атомарных предикатов ${}_A P_j^n(x_i)$ вводятся ключи α_j;

4) для отображения связей строится булеан 2^{α} ключей семантических атомов;

5) строится актуальная часть $2^{\alpha+}$ булеана ключей семантических атомов;

6) выполняется построение логической схемы БД, эквивалентной $2^{\alpha+}$.

Возможность использования РК в качестве универсального «носителя» данных для ПрО с произвольно заданной семантикой обусловлена такими свойствами РК, как полнота, единственность и *устойчивость к модификации* базового множества атомарных предикатов.

Для доказательства этих характеристик используем представление об операторе синтеза (2.3) применительно к ключам α_j семантических атомов. В общем виде оператор синтеза L действует как $\{C_k\} = L_A(B)$, $C_k = B + a$, $a \in A \setminus B$.

Оператор L продуцирует индексированную совокупность $\{C_k\}$ множеств C_k, содержащих все элементы базового множества B и один из элементов множества источника A, не принадлежащий B. Многократное (m раз) действие оператора L на базовое множество B обозначаем как $L_A^m(B)$. Заметим, что базовое множество может быть совокупностью множеств (тогда оператор синтеза действует на каждое из множеств базовой совокупности). Поэтому при многократном действии оператора синтеза его аргументом будет совокупность множеств, полученных в результате предыдущих действий оператора. Таким образом, сам оператор синтеза L определяется индуктивно.

Детальный анализ свойств оператора синтеза L указывает на то, что структуры, порождаемые L на множествах любой природы, эквивалентны схеме $\mathrm{B}(\alpha)$ или 2^α, т.е. собственно РК. Так в качестве элементов и базового множества, и источника могут выступать ключи семантических атомов.

2.10.1. Теорема о полноте и единственности РК

Докажем на изложенном подходе теорему 2.7. о полноте и единственности РК. И хотя логика доказательства будет аналогичной приведенной ранее, сформулируем ее несколько иначе.

Теорема 2.9. РК, порождаемый оператором синтеза на конечном множестве атомарных предикатов, полон и единственен.

Доказательство.

Полнота РК. Пусть заданы источник A и базовое множество B. Пусть также существует такое множество $C = \{c_i\}$, что $C \supseteq B$ и $\exists c \in A \setminus B$, но $\forall m : C \neq L_A^m(B)$, т.е. допустим, что не существует действия, порождающего множество C. Тогда из определения оператора синтеза следует $C \neq B \cup C_m^*$, где

$\forall m: C_m^* \subseteq A \setminus B$, значит $\exists c \in C_m^*$. Но это противоречит допущению $\exists c \in A \setminus B$. Следовательно, действие $L_A^m(B)$, порождающее множество C, действительно существует. Поэтому схема 2^α потенциально реализуема и полна. И фактически является булеаном отношений между семантическими атомами.

Единственность РК. Доказательство основано на его полноте. В силу полноты каждого из РК $\{C_k\} = L_A^m(B)$ и $\{C_k^*\} = L_A^m(B)$ имеем $\{C_k\} \setminus \{C_k^*\} = \varnothing$ и $\{C_k^*\} \setminus \{C_k\} = \varnothing$, следовательно, $\{C_k\} = \{C_k^*\}$, т.е. РК идентичны. Это означает, что на конечном множестве семантических атомов оператор синтеза порождает единственный РК. □

2.10.2. Теорема о модифицируемости каркасной схемы БД

Практическое значение имеет модифицируемость логических схем БД при изменении ПрО, например, при добавлении новых сущностей-объектов, но не их экземпляров, или удалении существующих, что встречается гораздо реже.

В этом контексте важна **Теорема 2.10.** (об устойчивости РК к модификациям). Если выполняется правило взаимной семантической атомарности, то изменение базового множества атомарных предикатов на единицу не влияет на исходную схему РК.

Доказательство. Пусть к базовому множеству $B = \{P_j\}$ атомарных предикатов P_j, на котором построен РК 2_B^α, добавляется некоторый новый предикат $X \notin B$. Согласно правилу взаимной семантической атомарности каждый из элементов нового базового множества $B + X$ является атомарным. Поэтому на этом новом базовом множестве реализуем единственным образом новый РК 2_{B+X}^α. Устойчивость РК при добавлении нового атомарного предиката состоит в независимости исходной структуры РК от дополнения, определяемого предикатом X, т.е. выражается условием $2_{B+X}^\alpha = 2_B^\alpha + 2_X^\alpha$, причем $2_B^\alpha \cap 2_X^\alpha = \varnothing$. Здесь РК 2_B^α является полным, или «насыщенным» [233], а РК 2_X^α состоит из единственного элемента, т.е. $2_X^\alpha = L_X(\varnothing)$. Допустим, что ключ α_X содержится в РК 2_B^α, т.е. $2_B^\alpha \cap 2_X^\alpha \neq \varnothing$. Согласно теореме о полноте РК

(применительно к РК 2_B^α), для некоторого базового множества B^* существует источник A^*, содержащий ключ α_X: $2_B^\alpha = L_{A^* \supset \alpha_X}(B^*)$. Это означает, что предикат X является элементом базового множества, на котором построен РК 2_B^α, т.е. $X \in B$. Но это противоречит исходному допущению о добавлении нового предиката: $X \notin B$. Следовательно, $2_B^\alpha \cap 2_X^\alpha = \varnothing$, т.е. дополнение каркаса при увеличении базового множества атомарных предикатов на единицу не имеет общих элементов с исходным РК и поэтому никак не влияет на исходную структуру отношений между атомарными предикатами. □

Аналогично доказывается устойчивость РК относительно модификации базового множества атомарных предикатов при «сжатии» ПрО, т.е. при устранении из базового множества B одного из атомарных предикатов. Если ПрО изменяется не на один, а сразу на несколько атомарных предикатов, то такая модификация сводится к последовательности единичных изменений.

2.11. Семантика формальных теорий

Отметим, что любую формальную теорию, путём введения новых понятий или чрезмерного расширения объёма существующих понятий можно сделать противоречивой. Поэтому введение новых понятий не только в формальную, но и в содержательную теорию должно сопровождаться тестом на непротиворечивость. Как известно диссертанту, расширенная реляционная модель данных (РРМД), в которую ее автор [361, 362] ввел несколько новых категорий, до сих пор не проверена на противоречивость.

Известно, что попытка построения универсальной метамодели столкнулась с проблемой принципиальной непополнимости множества первичных семантических категорий, ранее обнаруженной К. Геделем в арифметике и заключающейся в неполноте и непополнимости ее аксиоматики [170].

2.11.1. Формальное описание семантики

Определение семантики основывается на выделении некоторой типологии смыслов – категорий значений (семантических категорий, примитивов), через которые и посредством которых на семантическом языке

задается более сложный смысл. В основе описания семантики формальных языков лежит *принцип композиционности* Г. Фреге [31], заключающийся в определении семантики целого через семантику частей, т.е. между семантическими правилами интерпретации и синтаксическими правилами образования выражений языка может быть установлено соответствие.

В работе [47] показано, что на данный момент не существует универсального общепринятого подхода для формального описания семантики. Хотя разработано множество различных моделей и методов:

- грамматические модели, основанные на добавлении расширений к грамматике, определяющей язык: атрибутные грамматики [117], императивный или операционный метод [509], венский метод [512], W-грамматики [514];
- аппликативные модели, в которых непосредственно конструируется определение функции, которую вычисляет каждая программатика, написанная на этом языке: аксиоматический метод [409], метод денотационной семантики [502], метод функциональной семантики [154];
- модели спецификаций, в которых описываются отношения между разными функциями языка, и если программа реализует эти отношения, то она корректна по отношению к спецификации: метод спецификаций [251], алгебраический метод [94-96].

2.11.2. Понятийный анализ

Для спецификации ПрО в [47] использованы две формальные системы. Первая – *исчисление понятий* применена там для выражения инвариантных свойств ПрО. К инвариантным свойствам относят свойства структур понятий ПрО.

Вторая формальная система – специализированный предметный язык. В [47] предложено строить для каждой ПрО отдельно, с использованием ее специфических свойств.

Понятия, выявленные в процессе анализа ПрО, разделены на две группы: терминальные (или сигнификативные), выражаемые

последовательностью знаков терминального алфавита специализированного предметного языка, и нетерминальные (или денотационные), соответствующие нетерминальным знаками порождающей грамматики этого языка.

На основе выявления способов абстрагирования денотационных понятий построена понятийная структура ПрО, где под абстракцией понимается одно из четырех отображений вида $N^i \to N$, где N^i – декартова степень i множества понятий N, которые соответствуют четырем фундаментальным способам образования понятий: *обобщению, типизации, агрегации и ассоциации.*

Именно этот подход хорошо согласуется с методом КМД. Как будет показано в разделе 4, КМД также позволяет единым приемом моделировать перечисленные типы абстракций.

Для каждой такой абстракции в [47] приведено формальное и семантически прозрачное (инвариантное) определение, не требующее предметной интерпретации. Это имеет место в других концептуальных моделях, где используется множество связей между понятиями, несущими различную семантическую нагрузку. Но там это существенно усложняет моделирование. Поэтому для сравнения КМД с существующими формальными системами был выбран именно понятийный анализ. Выпишем из [47] основные определения.

Сущность определена там как устойчивое и уникальное представление о выделенной части ПрО. Сущность воспринимается в виде своих признаков. В [244] приведена следующая классификация сущностей: предмет, свойство (атрибут), состояние, процесс, событие, оценка, модификатор, квантификатор, модальность. Очевидно, что тут термин сущность имеет несколько иную трактовку, нежели в КМД. Но этим различием пренебрегаем, поскольку это не повлияет на итоговые выводы.

Признак – именованная сущность, характеризуемая множеством своих проявлений (значений) и имеющая проблемную интерпретацию

(семантическую роль). Признаки по своей сути являются элементарными сущностями, с точностью до которых производится описание предметной области.

Понятие определено как именованное множество сущностей, объединенных на основе общности своих признаков. Понятия именуются и определяются схемой, интенсионалом и экстенсионалом.

Имя (или знаковое представление понятия) рассматривается как языковая единица, отражающая некоторый смысл в семантическом плане и некоторую конкретную сущность в плане синтаксическом.

Схема понятия задана набором признаков, на которых понятие определено. Признаки в этом случае интерпретированы как некоторые элементарные понятия, на которых определяется схема. Последние, возможно, являются составными и строятся на основе других более простых понятий. Таким образом, понятие в общем случае может быть определено как именованное множество других понятий, имеющих подсхему, принадлежащую схеме образуемого понятия.

Интенсионал (или содержание понятия) представим набором значений взаимосвязанных признаков, позволяющим отличать сущности, принадлежащие понятию, от других сущностей ПрО. Тем самым интенсионал понятия косвенно задает тот смысл, который вкладывается в это понятие.

Экстенсионал (или объем понятия) будем задавать множеством сущностей, принадлежащих понятию.

Сущности, составляющие экстенсионал понятия различаются с помощью признаков. Так как каждый признак можно рассматривать как некоторое специализированное (первичное) понятие, то множество допустимых значений понятия-признака образует его экстенсионал, или *домен*, а *семантическая роль* понятия-признака представляется его интенсионалом.

В итоге, интенсионал признака – это те минимальные смысловые единицы, на которых строится описание ПрО, а экстенсионал признака задает элементарные синтаксические единицы (термы), которые используются для выражения составных понятий через простые.

Заметим, что в этой теории деление понятий на «простые» и «составные» является относительным и, по большей части, следует из онтологических представлений познающего субъекта.

Известны следующие способы абстрагирования понятий [55]: обобщение-специализация; типизация-конкретизация; агрегация-декомпозиция; ассоциация-индивидуализация. Выпишем из [47] определения, несколько отличные от приведенных в разделе 1.

Обобщение (от англ. generalization) – порождение понятия-обобщения на основе пересечения схем обобщаемых понятий и расширенного объединения их экстенсионалов.

При *специализации*, наоборот, – для заданного понятия-обобщения выделяются похожие на него понятия.

При обобщении подобные видовые понятия соотносятся с родовым, а при специализации, наоборот, – родовые понятия делятся на видовые.

При специализации схема понятия расширяется, и в нее включаются новые признаки, которые ранее были несущественными, или схема остается прежней, но происходит некоторое ограничение интенсионала исходного понятия.

Типизация – порождение понятия-типа на основе пересечения схем типизируемых понятий и строгого объединения их экстенсионалов.

При *конкретизации* из понятия-типа извлекаются типизированные в нем понятия. Для идентификации исходного понятия у сущностей понятия-типа используется подмножество признаков, называемое *ключом*.

Ассоциация – порождение понятия-ассоциации на основе объединения схем ассоциируемых понятий и ограничения декартова произведения их экстенсионалов.

При *индивидуализации* понятие-ассоциация разъединяется на ассоциируемые понятия. Для установления взаимосвязи сущностей, принадлежащих ассоциируемым понятиям, используется набор признаков, называемый *связью*.

Агрегация является предельным случаем ассоциации. В агрегацию могут входить произвольные комбинации сущностей агрегируемых понятий, а при ассоциации – только взаимосвязанные.

Агрегация – порождение понятия-агрегата на основе объединения схем агрегируемых понятий и декартова произведения их экстенсионалов.

При *декомпозиции* понятие агрегат разделяется на составляющие его агрегированные понятия.

Под *интерпретацией* формул логики первого порядка (язык которой состоит из множества функциональных символов F и множества предикатных символов P) понимается [19, 170, 192, 333] такое отображение, которое определяется следующим: задается несущее множество D и семантическая функция σ, отображающая каждый n-арный функциональный символ f из F в n-арную функцию $\sigma(f): D\times...\times D\to D$, каждый n-арный предикатный символ p из P в n-арное отношение $\sigma(p)\subseteq D\times...\times D$.

2.11.3. Семантическая теория понятий

Отличительной особенностью формальных теорий (исчислений, или синтаксических теорий) является то, что в последовательность знаков, получаемых в этих теориях, не вкладывается никакого смысла, пока не введена их явная интерпретация. Но введение интерпретации не относится к зоне ответственности самой синтаксической теории. Для этого требуется привлечение другой, более общей и более мощной теории (метатеории, или семантической теории), в которую в качестве фрагмента входит интерпретируемая синтаксическая теория.

Только в этом случае, на языке метатеории, становится возможных формулировка свойств синтаксической теории. По этой причине любая синтаксическая теория подразумевает наличие некоторой стандартной области интерпретации, относительно которой делаются все содержательные утверждения (мета-утверждения).

Так, в математической логике различают исчисление высказываний (исчислении предикатов) и алгебру высказываний (алгебру предикатов), причем алгебра высказываний (алгебра предикатов) рассматривается как семантическая теория, поставляющая исчислению высказываний (исчислению предикатов) стандартную область интерпретации [155, 333].

В свою очередь исчисление высказываний (исчисление предикатов) является синтаксической теорией и используется как логическое средство в других математических теориях.

Однако все высказывания о полноте и непротиворечивости исчисления высказываний (исчисления предикатов) формулируются не относительно прикладной математической теории, для описания которой оно предназначено, а относительно алгебры высказываний (алгебры предикатов), которая, в этом случае, является семантической теорией и непременно должна быть включена в синтаксические средства прикладной теории.

Разработанные в [47] семантическая и синтаксическая теории понятий, применяемые для формализации понятий и связей между ними, могут быть использованы для доказательства аналогичных утверждений о КМД. И основные базовые выводы о семантической и синтаксической полноте, непротиворечивости и разрешимости формул КМД могут считаться доказанными, поскольку доказанными являются аналогичные утверждения там. Выпишем из [47] эти утверждения.

В отличие от известных формализмов (семантических сетей, концептуальных схем, и др.), где на понятиях задается большое множество отношений различной природы, в [47] введен в использование новый формализм – понятийная структура, которая определена множеством

понятий с четырьмя видами отображений, единственное назначение которых – показать способы образования понятий.

Определение 2.V1. Понятийной структурой $S=\langle N,G,T,A,C\rangle$ будем называть конечное множество понятий $N=\{N_0, N_1 ,\ldots, N_{n-1}\}$, на которых заданы четыре конечные множества отображений абстрагирования: обобщения $G=\{g_0, g_1 ,\ldots, g_{n-1}\}$, типизации $T=\{t_0, t_1,\ldots, t_{n-1}\}$, ассоциации $A=\{a_0,a_1,\ldots, a_{n-1}\}$ и агрегации $C=\{c_0,c_1,\ldots,c_{n-1}\}$, *где* $g(N_g)\rightarrow N_g$, $t(N_t)\rightarrow N_t$, $a(N_a)\rightarrow N_a$, $c(N_c)\rightarrow N_c$

Если по условиям решаемой задачи не требуется уточнение способа абстрагирования понятий, то понятийную структуру будем рассматривать как множество понятий, на котором установлены два типа отображений: отображения дифференциации и отображения интеграции: $S=\langle N,D,I\rangle$, где $D=\{d_0, d_1,\ldots, d_{n-1}\}$, $I=\{i_0, i_1 ,\ldots, i_{n-1}\}$ множества отображений дифференциации и интеграции, заданные на понятиях N.

Заметим, что понятийная структура очень близка к КМД. Она также близка к моделям данных «сущность-связь» (ER-модель) [359] и расширенная «сущность-связь» (EER-модель) [274]. То, что и в КМД, и в ER-моделях элементами являются не понятия, а данные, не снижает важности аналогии. В EER-модели для детализации используется трудно формализуемая семантическая разметка в виде дополнительных нотаций и ограничений. Но КМД таких ограничений не имеет.

2.11.4. Алгебра понятийных структур

Совокупность понятий и четырех множеств отображений одних понятий в другие определено как понятийная структура. Учитывая то, что типизация является частным случаем обобщения, которое, в свою очередь выражает дифференциацию понятий, а агрегация является частным случаем ассоциации, выражающей интеграцию понятий, абстрагируемся от частных способов образования понятия и будем далее использовать только дифференциацию и интеграцию.

В алгебре понятийных структур различают понятия и операции над понятиями. При этом под понятием понимается знак (слово), о котором можно вполне определенно сказать, что он (оно) обозначает некоторое множество других понятий. В алгебре понятий отвлекаются от конкретного содержания понятий и интересуются лишь вопросом, из каких других понятий образовано каждое понятие [47].

Заметим, что эта задача совпадает с задачей автоматизированного определения формализованной каркасной структуры (*этимологии смысла*, которая будет введена в разделе 4) каждой конкретной сущности-объекта некоторой ПрО. Именно структура составных сущностей-объектов и является искомым результатом синтеза схемы БД на РК.

Для обозначения понятий в [47] использованы знаки N_i , где i – натуральное число, включая ноль. Для того, чтобы показать, что понятие N_i образовано из n понятий N_1, N_2, ..., N_n путем их дифференциации использована запись вида $\delta_n(N_1, N_2, ..., N_n) \to N_i$, а путем их интеграции – запись вида $\lambda_n(N_1, N_2, ..., N_n) \to N_i$. В этой нотации знак δ_n выдран для обозначения дифференциации понятий по его внешней схожести со знаком дифференциала, а λ - по его схожести с символом интегрирования.

Определение 2.V2. Пусть n – любое натуральное число. Тогда n-местным дифференциалом (интегралом) понятий, определенным на множестве N, называется всякое однозначное отображение n-ой декартовой степени множества N в само множество

$$\delta_n : N^n \to N \; (\lambda_n : N^n \to N).$$

Если имеются некоторые понятия, то из них при помощи операций интеграции и дифференциации можно образовать (выразить) другие понятия. Например, если понятия N_1 и N_2 служат для выражения понятия N_3 путем дифференциации, запишем $\delta_2(N_1, N_2) \to N_3$, а если понятие N_4 образовано при интеграции понятия N_1, N_2 и N_3, то $\lambda_3(N_1, N_2, N_3) \to N_4$.

Определение 2.V3. Алгеброй понятийных структур называется тройка $\langle N, \Delta, \Lambda \rangle$, где N – множество понятий $N = \{N_1, N_2, ...\}$, а $\Delta(\Lambda)$ – множество операций дифференциации (интеграции) понятий различной местности:

$$\Delta = \{\delta_0, \delta_1, \delta_2, ...\}, (\Lambda = \{\lambda_0, \lambda_1, \lambda_2, ...\}).$$

Определение 2.V4. Формулой алгебры понятийных структур называется множество отображений дифференциации и интеграции, заданное на некотором конечном подмножестве N.

Из определения понятия следует, что каждое понятие обладает схемой – набором признаков, на которых понятие определено.

Рассмотрим задачу определения схем понятий по имеющейся понятийной структуре. Для ее решения схему понятия определим как набор простых понятий, на которых понятие определено. Схему понятий будем вычислять на основе отображений абстрагирования, заданных в понятийной структуре, по следующей рекуррентной процедуре:

- схема простого понятия N равна (N);
- схема понятия, полученного в результате дифференциации, равна пересечению схем дифференцируемых понятий;
- схема понятия, полученного в результате интеграции, равна объединению схем интегрируемых понятий;
- схема понятия, полученного в результате дифференциации и интеграции, равна объединению схем интегрируемых понятий, принадлежащему пересечению схем дифференцируемых понятий.

Операции объединения и пересечения множеств в рассматриваемом случае выполняются с учетом повторения элементов. Одновременная интеграция и дифференциация понятий является выразительным средством, которое позволяет уточнить схему определяемого понятия путем ограничения пересечения схем дифференцируемых понятий указанием некоторого подмножества этого пересечения.

Очевидно, если схема, получаемая через интеграцию понятий, не содержится в схеме, вычисленной на основе дифференциации, то последнее говорит о недопустимом описании понятийной структуры.

2.11.5. Синтаксическая теория понятий

Всякая формальная теория определяется формальным языком, порождающим формулы, имеющими смысл с точки зрения этой теории, и совокупностью теорем, интерпретируемых в некоторой предметной области как выполнимые (имеющие место). Собственно формальной теорией называется множество теорем, которое замкнуто относительно правил вывода.

Формальная теория называется *разрешимой*, если существует конструктивная процедура, которая по любой строке в алфавите T позволяет судить, является ли эта строка формулой теории или нет.

Для конструктивного построения формальной теории V:

- фиксируется конечный алфавит знаков T, называемый терминальным;
- определяется перечислимое множество формул F, каждая из которых является строкой в алфавите T;
- выделяется разрешимое множество аксиом A, являющееся подмножеством множества формул F, $A \subset F$;
- задается конечное множество правил вывода P, которые рассматриваются как вычислимые отображениями различной конечной местности i на множестве формул F, позволяющие получать новые теоремы на базе имеющихся по схеме $P \subset (\bigcup_{i=0}^{\infty} F^i \to F)$, где F^i – i-я декартова степень множества F.

Под конструктивностью традиционно понимается существование эффективной процедуры (алгоритма), предназначенной для решения той или иной задачи, например, задания множеств, определения принадлежности элемента множеству, и др.

Алгоритмом [302] назовем предписание, позволяющее по каждому исходному данному, или аргументу, из некоторой совокупности возможных (для данного алгоритма) исходных данных (аргументов) получить результат в случае, если таковой существует, или не получить ничего в случае, если для рассматриваемого исходного данного не существует результата.

Если для выбранного исходного данного результат существует, говорят, что алгоритм применим к этому исходному данному и перерабатывает его в этот результат.

Множество называется *перечислимым* (рекурсивно) [302], если оно - либо пусто, либо является множеством элементов какой-нибудь вычислимой последовательности, т.е. множеством значений какой-нибудь вычислимой функции, определенной на натуральном ряду.

О такой функции (последовательности) говорят, что она перечисляет рассматриваемое множество. Функция, которая вычисляется некоторым алгоритмом, называется вычислимой.

Таким образом, множество будет перечислимым [47], если существует алгоритм, порождающий это множество, т.е. такой алгоритм, который последовательно выдаёт элементы данного множества (быть может, с повторениями) и только их. Предполагается, что любой элемент множества рано или поздно будет получен.

Множество называется *разрешимым* [47, 302] если существует единый способ, позволяющий относительно любого элемента определить, принадлежит он этому множеству, или нет. Формальным уточнением этого понятия является понятие рекурсивного множества – такого множества, для которого существует (формальный) алгоритм, разрешающий это множество, т.е. дающий для любого элемента ответ, принадлежит он этому множеству или не принадлежит.

Очевидно, что всякое разрешимое множество перечислимо. Обратное неверно, поскольку существуют примеры перечислимых, но не разрешимых множеств.

В абстрактных формальных системах конструктивные средства для перечисления множества формул могут не задаваться. В этом случае предполагается, что произвольная строка в алфавите *T* является формулой теории. Тем самым неявно предполагается наличие некоторой конструктивной процедуры перечисления или распознавания произвольных строк над этим алфавитом.

2.11.6. Полнота и непротиворечивость

Основными свойствами любой формальной теории являются ее полнота и непротиворечивость.

Полнота формальной теории рассматривается как свойство, характеризующее достаточность для каких либо целей ее выразительных средств . Для любой интерпретации теории необходимо отображение, устанавливающее соответствия между формулами теории и сущностями (объектами) ПрО, которую в этом случае называют *областью интерпретации* теории.

В прикладных теориях значения исходных терминов даны с самого начала, т.е. интерпретацию данной теории полагают фиксированной. Полнота формальной теории в данном случае называется *семантической полнотой*.

Прежде всего, необходимы условия, при которых полные формальные системы не являются противоречивыми. Выпишем эти условия из [302]. При этом все обозначения сохраним без изменений.

Под *алфавитом* понимается конечный список элементарных (т. е. считающихся не членимыми далее) знаков, называемых буквами этого алфавита. Конечная цепочка следующих друг за другом букв некоторого алфавита называется словом в этом алфавите.

Пусть имеется алфавит В. Множество всех слов в алфавите В будем обозначать В*. Предполагается, что для каждого языка имеется такой алфавит, что все выражения этого языка (т. е. имена тех или иных предметов, утверждения об этих предметах и т. п.) суть слова в этом алфавите.

Предполагается, что в множестве B*, где B — алфавит рассматриваемого языка, задано подмножество T, называемое множеством «истинных утверждений» (или «истин»). Будет достаточным считать язык полностью заданным, коль скоро задан алфавит B и подмножество T множества B*. Всякую такую пару ⟨B, T⟩ мы будем называть фундаментальной парой.

В дальнейшем будем пользоваться термином «доказательство». Хотя этот термин является основополагающим, он не имеет точного определения. Но это не сказывается на выводах. Не претендуя на то, чтобы дать точное определение, будем использовать его формальный аналог, для которого и сохраним термин.

Будучи записанным, доказательство становится словом в некотором алфавите D; все доказательства образуют некую совокупность в D*.

Итак, пусть:

1. Имеются алфавиты B (алфавит языка) и D (алфавит доказательств).

2. В множестве D* выделено подмножество d, элементы которого называются доказательствами; предполагается наличие алгоритма, позволяющего по произвольному слову в алфавите D узнавать, принадлежит оно d или нет.

3. Имеется функция δ (функция выделения доказанного), у которой область определения Δ удовлетворяет соотношению $d \subseteq \Delta \subseteq D^*$ и которая принимает свои значения в B*; предполагается наличие алгоритма, вычисляющего эту функцию; доказательство d_i из d называется доказательством слова $\delta(d_i)$.

Дедуктикой над алфавитом B называют тройку ⟨D, d, δ⟩, удовлетворяющую условиям 1—3. Естественно потребовать, чтобы доказуемыми были лишь «истинные утверждения», т. е. слова, принадлежащие множеству T.

Дедуктику ⟨D, d, δ⟩ называют непротиворечивой относительно фундаментальной пары ⟨B, T⟩, если $\delta(D) \subseteq T$.

Если имеется язык, то необходимо найти такую непротиворечивую дедуктику, в которой каждое истинное утверждение было бы доказуемым. Теорема Гёделя [302] в интересующем нас варианте именно и утверждает, что при определенных условиях, налагаемых на фундаментальную пару, этого сделать нельзя.

Дедуктику $\langle D, d, \delta \rangle$ называют полной относительно фундаментальной пары $\langle B,T \rangle$, если $\delta(D) \supseteq T$.

Для каждого алгоритма можно указать некоторый алфавит исходных данных, так что все возможные исходные данные являются словами в этом алфавите, и некоторый алфавит результатов, так что все результаты являются словами в этом алфавите.

В части 3 определения понятия доказательства говорится о том, что функция выделения доказанного должна быть вычислимой функцией.

В силу сделанных предположений относительно понятия алгоритма для каждой вычислимой функции можно указать такие два алфавита, что все ее аргументы суть слова в первом из этих алфавитов, а все ее значения — слова во втором из этих алфавитов.

Особый интерес представляют функции, аргументы и значения которых суть натуральные числа. Такие функции условимся называть числовыми.

В [302] доказана следующая **Теорема 2.U**. Чтобы для $\langle B, T \rangle$ существовала полная и непротиворечивая дедуктика $\langle D, d, \delta \rangle$, необходимо и достаточно, чтобы множество T было перечислимым.

Тогда справедливой будет следующая **Теорема 2.11.** (о непротиворечивости РК). РК, построенный на перечислимом множестве N, полон и непротиворечив.

Доказательство немедленно следует из теоремы 2.U.

В теориях, конструируемых на основаниях формальной аксиоматики, значения исходных терминов остаются неопределенными во время вывода. В этом случае формальная теория называется *семантически полной*

относительно заданной интерпретации, если из нее могут быть выведены все формулы, необходимые и соответствующие этой интерпретации.

Наряду с понятием семантической полноты определяется и другое ее понятие, которое рассматривается как внутреннее свойством формальной теории, не зависящее ни от одной из ее интерпретаций. Формальную теорию называют *синтаксически полной*, если порождаемое ею множество формул достаточно для произвольной области интерпретации.

Будем назвать доказанные выше свойства РК синтаксической полнотой и непротиворечивостью КМД. А для исследования семантической полноты и непротиворечивости этой модели данных воспользуемся исследованием этих же свойств теории понятий.

Формальную теорию называют *дедуктивно полной*, если для всякой формулы, выводимой в данной теории, можно показать ее истинность (является в таком случае теоремой).

Во многих формальных теориях имеется различие между множеством выводимых формул и множеством теорем. В этом случае формальная теория дополняется конструктивными средствами, позволяющими получить произвольную теорему. При логической интерпретации *теоремой* называется формула, выводимая из аксиом и интерпретируемая как истинная. А конструктивные средства для этого предоставляются логическими правилами вывода. Заметим, что в математической логике распознание в произвольной формуле теоремы исчисления предикатов первого порядка неразрешимо [37].

Понятно, что в этом случае подразумевается логическая интерпретация, которая ставит в соответствие каждой формуле значение ее истинности.

Как отмечено ранее, формальная теория понятий нас интересует как предельный случай КМД. В [47] она построена на основе алгебры понятийных структур, которая предназначена для выражения способов абстрагирования понятий и дополненная описанием интенсионалов понятий, задающих конструктивные процедуры для разрешения их экстенсионалов.

Определение 2.V5. Формулой формальной теории понятий называется понятийная структура, которая дополнена описанием интенсионалов всех входящих в нее понятий.

Так как в формализме понятийной структуры предусмотрены средства для выражения всех известных абстракции [47, 55], определение синтаксической полноты теории там осуществлено путем выявления условий, при которых такое определение становится неприменимым. Как это имеет место и в исчислении предикатов, различают полноту и непротиворечивость самой формальной теории.

Однако в противоположность исчислению предикатов, используем понятия полноты и непротиворечивости понятийной структуры, а также ее интенсионалов, выражаемой формулой этой теории.

Определение 2.V6. Формула формальной теории понятий называется *синтаксически полной*, если в ней отсутствуют понятия, которые не содержат ни одной сущности, т.е. экстенсионал которых пуст.

В частности, непосредственно из определения 2.V6. следует, что если у некоторого понятия схема пуста, то это свидетельствует о синтаксической неполноте понятийной структуры, так как из пустоты схемы следует и невозможность определить хотя бы одну сущность, принадлежащую экстенсионалу этого понятия, кроме разве что пустого понятия, не содержащего ни одной сущности.

Определение 2.V7. Формула формальной теории понятий называется *семантически полной*, если в ней определены все понятия, способы их абстрагирования и выражения, необходимые для описания заданной области интерпретации.

Определение 2.V8. Формула формальной теории понятий называется *противоречивой*, если в ней описана сущность, выраженная как принадлежащая, так и как не принадлежащая экстенсионалу одного и того же понятия.

Определение 2.V9. Формальную теорию понятий будем называть:

– *непротиворечивой*, если все выводимые в теории формулы непротиворечивы;

– *семантически полной*, если для любой области интерпретации существует в теории вывод соответствующей ей формулы;

– *синтаксически полной*, если для любой выводимой в теории формулы существует область интерпретации;

– *разрешимой*, если существуют конструктивные средства для распознавания формул, выводимых в теории.

Из данного определения видно, что понятия полноты и непротиворечивости формальной теории понятий несколько отличаются от принятых в математической логике.

В формальной теории понятий, по аналогии с исчислением предикатов, в качестве семантической теории выступает алгебра понятийных структур, поставляющая синтаксической части теории понятий стандартную область интерпретации. Поэтому для оценки состоятельности теории по описанию полных и непротиворечивых понятийных структур введено понятие семантической полноты, а для оценки невозможности получения описаний неполных и противоречивых понятийных структур – понятие синтаксической полноты.

Выписанное из [47] определение синтаксической полноты формальной теории понятий является некоторым обобщением синтаксической полноты исчисления высказываний в узком смысле. Непротиворечивая теория называется *синтаксически полной в узком смысле*, если добавление к ее аксиомам любой невыводимой в этой теории формулы с сохранением всех правил вывода, приводит к противоречивой теории [333].

Правила вывода исчисления понятий сохраняют синтаксическую полноту и непротиворечивость каждой выводимой в исчислении понятийной структуры. При этом аксиома рассматривается как минимальная понятийная структура, являющаяся синтаксически полной и непротиворечивой по определению. Следовательно, каждая выведенная в исчислении понятийная

структура является ее теоремой. Отсюда в 47 доказана справедливость следующего утверждения.

Теорема 2.V1. Исчисление понятий полно и непротиворечиво.

Достаточность используемых в исчислении понятий четырех абстракций для выражения любой понятийной структуры имеет такое же обоснование, как и достаточность логических связок и кванторов исчисления предикатов для выражения всех высказываний относительно любой области интерпретации.

Поскольку КМД также базируется на этих четырех абстракциях, данный вывод очень важен для понимания ее достаточности, не смотря на то, что КМД является лишь частным случаем РМД.

Под *разрешимостью* формальной теории обычно понимается возможность определить для произвольной формулы, выводима или нет эта формула из множества аксиом теории. Известно, что свойством разрешимости обладает исчисление высказываний, но не обладает исчисление предикатов [333].

Аналогично разрешимость в формальной теории понятий определена как возможность установить принадлежность произвольной строки множеству формул (понятийных структур), выводимых в этой теории. В [47] доказано следующее утверждение.

Теорема 2.V3. Исчисление понятий разрешимо.

Выше непротиворечивость и полнота исчисления понятий установлена исходя из предположения о непротиворечивости и полноте интенсионалов понятий. Последнее может быть обеспечено использованием для выражения интенсионалов известных непротиворечивых теорий, например, исчисления предикатов, формальной арифметики, и др.

Логическую теорию понятий определим как расширение исчисления понятий за счет дополнения формализма исчислением предикатов первого порядка, которое используется для выражения интенсионалов.

Заметим, что определение тождественной ложности формул в исчислении высказываний разрешимо, что следует из существования эффективной процедуры, которая по любой формуле позволяет судить, является ли она теоремой исчисления высказываний или нет [333].

Однако проблема разрешимости исчисления предикатов решается отрицательно [333]. Впрочем, как и проблема синтаксической полноты в узком смысле относительно алгебры предикатов. Не существует эффективной процедуры, выполнение которой гарантировало бы, что если имеется теорема исчисления предикатов, то существует конечный шаг этой процедуры, на котором она будет доказана. Т.е. множество теорем исчисления (тавталогий) не является рекурсивно перечислимым.

По этой причине формальная теория понятий, построенная на основе исчисления предикатов, будет обладать синтаксической неполнотой. Последнее является следствием того, что проверка формулы на противоречивость вынуждено сводится к проверке тавтологичности ее отрицания.

Определение 2.V10. *Противоречием* называется формула логики предикатов, которая ложна при всех интерпретациях, т.е. является тождественно-ложной.

Но имеется один частный случай, относительно которого исчисление предикатов разрешимо – это *одноместное* исчисления предикатов первого порядка [134].

Следовательно, только одноместное исчисление предикатов может служить формальным средством для синтаксически полного выражения интенсионалов понятий. В противном случае определение синтаксической полноты произвольной формулы логической теории понятий становится неразрешимым, как и неразрешимым определение принадлежности произвольной строки формулам этой теории.

На основании разрешимости исчисления понятий, рассматриваемом как исчисление понятийных структур, в [47] доказана следующая **Теорема**

2V4. Логическая теория понятий является разрешимой относительно одноместного исчисления предикатов.

Для обеспечения синтаксической полноты логической теории необходимо устанавливать отсутствие противоречия при выражении интенсионала каждого понятия. Очевидно, для такой проверки не требуется знание области интерпретации, так как противоречие является тождественно ложной формулой во всех интерпретациях. Следовательно, одноместное исчисление предикатов сохраняет и синтаксическую полноту логической теории.

После такого уточнения в [47] доказана следующая **Теорема 2.V5.** Логическая теория понятий является непротиворечивой и синтаксически полной относительно одноместного исчисления предикатов.

Вопрос о семантической полноте логической теории понятий может быть переформулирован следующим образом: существуют ли экстенсионалы понятий произвольной ПрО, которые не выражаются на языке одноместного исчисления предикатов . Иными словами, существует ли высказывание о принадлежности той или иной сущности экстенсионалу некоторого понятия, которое является истинным в ПрО, но не выразимым в одноместном исчислении предикатов.

Общий ответ на этот вопрос оказывается отрицательным даже для полного исчисления предикатов. Однако известны примеры перечислимых, но не разрешимых множеств, например, множество самоприменимых алгоритмов [134]. Другой пример – упомянутое ранее множество формул исчисления предикатов первого порядка, которое перечислимо, но не разрешимо.

Последнее означает, что существуют множества, которые можно задать в виде перечисляющей процедуры, но не существует процедуры, которая позволяет определяет принадлежность произвольного элемента этому множеству. Перенос этого результата в логическую теорию понятий

означает, что существуют понятия, экстенсионалы которых не могут быть заданы на языке исчисления предикатов.

Определение 2.V10. Областью *логической интерпретации* называется такая область интерпретации, в которой экстенсионал произвольного понятия выразим на языке одноместного исчисления предикатов.

Таким образом, семантическую полноту логической теории понятий следует рассматривать относительно не области интерпретации, а относительно того подмножества, в котором интенсионал понятий представим формулами одноместного исчисления предикатов.

На основании этого в 47 доказана **Теорема 2.V6.**. Логическая теория понятий является семантически полной относительно области логической интерпретации.

Как будет показано в разделе 3, семантически атомарные многоместные предикаты равносильны конъюнкции одноместных предикатов. Поэтому семантически полная логическая теория понятий, интенсионал которых представим формулами одноместного исчисления предикатов, является своеобразной верхней границей разрешимости КМД как формальной теории.

2.12. Выводы ко второму разделу

1. Эксплуатационные характеристики приложений (скорость доступа к данным, скорость внесения модификаций в схему БД и в само приложение в динамическом режиме, кроссплатформность и интероперабельность приложений) при безаномальной схеме БД существенно повышаются.

2. Изложен алгоритм, в котором рассматривается отсутствие ограничений на уровень арности и количества связей для каждой группы сущностей, приводящий к рекурсивным связям. Учет этих особенностей не внесет принципиальных изменений в схемы БД построенных на КМД.

3. Разработанная концепция позволяет проектировщику минимизировать и листинги будущих запросов пользователей, и листинг приложений. Спроектировав максимально возможное число моделей связей, заранее предусмотреть большинство запросов.

4. Аномалия «потери информации» в полученных отношениях, которые построены на обобщениях ДЗ - результат некорректных соединений при выполнении запросов пользователей. Использование таких соединений может быть минимизировано.

Аномалии вставки и удаления в совокупности отношений-связей и отношений-сущностей не возникают лишь при определенных ограничениях.

5. Обнаруженная и обоснованная ДЗ, которая являясь частным случаем МЗ и ЗПС, тем не менее, является обобщением в смысле полноты числа кортежей, в шунтированном виде может быть использована в качестве шаблона для моделирования связей между сущностями-объектами в ПрО. Как элемент РК такая модель позволит моделировать соединения при проектировании предварительно настроенных запросов. И тем самым существенно экономить на соединениях при обращении к БД.

6. Любая схема РБД, синтезируемой для заданного множества атрибутов, является элементом этого РК. Решение любой задачи синтеза или задачи модификации уже содержится в РК схем БД.

7. Ключевой РК строго соответствует критериям 5НФ. Проделав над множеством ключевых атрибутов процедуры, несложно из РК получить ключевой.

8. РК, единственным образом синтезируемый на базовом множестве атомарных предикатов, выступает в качестве «носителя» данных о фактах ПрО. Такое свойство РК, как его полнота гарантирует, что все (актуальные и потенциальные) отношения между парами, тройками, четверками и т.д. атомарных предикатов можно единообразно описать в рамках одной структуры – РК.

9. Свойство РК - устойчивость к модификациям базового множества атомарных предикатов гарантирует, что при изменении ПрО не возникнет потребность в тотальной реорганизации всей структуры. Семантически атомарные предикаты играют роль оптимальных «сущностей-объектов» для представления данных из ПрО, а сам РК является оптимальным «носителем» данных о взаимосвязях между этими сущностями-объектами.

РАЗДЕЛ 3
КАРКАСНОЕ ПРОЕКТИРОВАНИЕ ДОМЕННО-КЛЮЧЕВОЙ СХЕМЫ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

3.1. Введение к третьему разделу

Семантический анализ произвольных ПрО и современное проектирование схемы БД осуществляется в основном с использованием модели П. Чена «сущность-связь» [359] или расширенной РМД Э. Кодда [362]. Хороший обзор методов проектирования предоставлен в классических учебниках по БД Д.М. Кренке [131] и С.Д. Кузнецова [135]. Однако модель «сущность-связь» обладает некоторыми недостатками – она не дает строгих формальных определений сущности и атрибута сущности, а также не учитывают функциональных требований к приложению на стадии проектирования схемы БД.

Для некоторых приложений такой подход применим, когда можно абстрагироваться от операций над данными и моделируются только сами данные. Но основой приложений являются функциональные требования к манипулированию данными. И если проектировщик вынужден сопровождать схему БД, постоянно учитывая всевозможные модификации, растет и стоимость проекта .

Как отмечено в обзоре в разделе 1, существующий классический подход к проектированию схемы БД имеют следующие недостатки.

1. Сложность и трудоемкость идентификации ФЗ.
2. Зависимость конечного результата проектирования от опыта и субъективного взгляда проектировщика, а не от метода проектирования.
3. Проблема идентификации сущностей и атрибутов сущностей;
4. Модифицируемость схемы БД.
5. Отсутствие четких технологий получения высоко-нормализованных схем БД.

При значительном числе классов сущностей-объектов и атрибутов число всевозможных ФЗ существенно возрастает. Как правило, на практике все ФЗ не рассматриваются. И практически все отношения из-за экономии времени проходят процесс нормализации не выше 3НФ, что приводит к значительным сбоям и ошибкам на этапе эксплуатации. Очевидно, что для их устранения требуются значительные затраты временных и человеческих ресурсов.

Указанные недостатки и проблемы определяют актуальность разработки метода проектирования схемы БД, устраняющего ошибки проектирования и значительно снижающего трудозатраты. На наш взгляд, именно безаномальность схемы БД [378] является одним из самых важных критериев качества внедряемых приложений. В [206] обоснован алгоритм синтеза схемы БД, находящихся в безаномальной ДКНФ [378] так, что одновременно удовлетворяются и критерии безаномальности, и модифицируемости.

В [2, 294] впервые сформулирована гипотеза о том, что максимально возможная степень модифицируемости обеспечена только схемой БД, находящейся в ДКНФ [378]. Но далее в работах авторов [2, 294] эта гипотеза должным образом не формализована, алгоритм строго не обоснован, а процедура синтеза не является единственной и полной.

Как показывает опыт практической эксплуатации, все существующие на рынке инструментальные средства синтеза приложений, а также все серверы БД предлагают пользователю самостоятельно определиться с выбором схемы БД – хоть в РМД, хоть в ООМД. Причем для большинства серверов БД, разработанных в соответствии с РМД, не накладывается существенных ограничений на нормальность отношений. А значит и на итоговую схему БД. Однако такая «маркетинговая» концепция инструментальных средств, как правило, приводит к полной несовместимости приложений даже в рамках одной платформы. И к невысокой эффективности работы приложения.

Для решения указанных проблем существует два подхода в использовании безаномальной схемы БД. Первый – это разработка приложений на базе существующих на рынке инструментальных средств и серверов БД, однако с использованием высоко-нормализованной (безаномальной) схемы БД. Опыт разработок позволяет утверждать, что эксплуатационные характеристики приложений (скорость доступа к данным, скорость внесения модификаций в схему БД и в само приложение в динамическом режиме, кроссплатформность и интероперабельность приложений) существенно повышаются.

Второй подход – это разработка уникального инструментального средства, непосредственно использующего все указанные преимущества именно безаномальной схемы БД, в том числе и минимально необходимое число операций соединений для ответов на запросы.

Описываемый метод может быть использован, прежде всего, для генерации схем БД, удовлетворяющих условиям ДКНФ. Однако его можно также использовать для построения устройств распознавания речи, устройств-переводчиков, экспертных систем, систем автоматизированного аудита корректности работы введенных в эксплуатацию комплексов приложений и т.п. А также систем автоматизированного проектирования дейтацентров для ПрО с возможностью максимально гибкой модификации схемы БД.

3.2. Постановка задачи

Как уже отмечалось, проблемы эксплуатации приложений зачастую возникают из-за низкой модифицируемости схемы БД [204, 206]. Однако это свойство невозможно реализовать на низко-нормализованных схемах, так как, с одной стороны, низкая нормализация является результатом плохой осведомленности проектировщика о всех причинно-следственных связях в ПрО. А с другой – низко-нормализованная БД подвергается значительным рискам. Существующая высокая вероятность аномалий вставки и удаления снижает эксплуатационные характеристики и оперативных БД, и ХД. Но при

этом, постоянные модификации приложений вносят в низко-нормализованную схему дополнительные источники ошибок. Все это значительно повышает стоимость сопровождения приложений.

Но декомпозиция все чаще представляется чем-то несовершенным и требующим кардинального пересмотра [202, 217]. При этом сама РМД эволюционирует, вмещая в себе подходы, основанные на моделях универсального и бинарного отношений, модели «сущность-связь», объектно-ориентированной и семантической моделях [205, 209]. Каждый из этих подходов использует свою «точку видения» на ПрО и собственное модельное представление, отображаемое в схеме БД.

Для применения РК при проектировании схем БД необходимо знать условия отсутствия аномалий вставки и удаления. Поэтому исследования условий безаномальности является актуальной задачей.

3.3. Доменно-ключевая схема базы данных

Выпишем из [378] определение понятия ограничения схемы отношения. *Z-ограничением* (или просто *ограничением,* если подразумевается совокупность Z отношений) будем называть отображение набора всех S-отношений на домен {true, false}. Такие ограничения на отношения вводятся, как правило, с помощью высказываний, написанных на заданном языке – таком, как исчисление предикатов 1-го порядка. На эти ограничения могут быть наложены еще и дополнительные ограничения.

Будем говорить, что отношение R *удовлетворяет* ограничению g, или, что g *содержится* в R, если после этого отображения, отношение R принимает значение *true*. Иначе, мы говорим, что g ложно (*false)* в R. К ограничениям также относят традиционные зависимости – ФЗ, МЗ, ЗПС, а также введенные в [378] зависимости домена – DD и зависимости ключа – KD.

Пусть X – набор атрибутов, g – единственное ограничение и G – набор ограничений (или единственное ограничение). Тогда очевидно, что G *логически включает* g (*в контексте* X), или, что g *– логическое следствие* G

(*в контексте X*). Строго это означает, что всякий раз, когда G содержится в отношении с атрибутами X, тогда и g является таковым. Будем обозначать этот факт как обычное следствие $G \Rightarrow g$, если контекст X подразумевается. Например, $\{A \to B, B \to C\} \Rightarrow A \to C$.

Согласно [378] введем определение *реляционной схемы* как набора атрибутов вместе с набором ограничений. Таким образом, можно говорить не об ограничениях *в* схеме, а об ограничениях, которые являются логическим следствием тех, что в схеме. Удобно предположить, что набор ограничений схемы замкнут под действием логических ограничений. То есть, пусть G – набор ограничений схемы. Тогда если g – такое ограничение, что $G \Rightarrow g$, тогда g – также является ограничением этой же схемы.

Тогда отношение является *допустимым экземпляром* схемы, если оно имеет те же атрибуты, что и схема, и если удовлетворяет всем ограничениям схемы.

Как и в [378], будем обозначать зависимость домена $IN(A,S)$ (от слова Include), где A – атрибут и S – множество. Означает, что для каждого кортежа значение атрибута A должно принадлежать S.

Зависимость ключа $KEY(K)$, где K – множество атрибутов, означает, что K является *ключом* (*key*), т.е., что в отношении нет двух кортежей, для которых значения K одинаковы.

С понятием ограничения реляционной схемы связано и понятие *выполнимости* реляционной схемы [378]. Схема выполнима, если для нее найдется, по крайней мере, один допустимый экземпляр схемы, и набор ее ограничений непротиворечив.

Теперь, по аналогии с [378], введем определение ДКНФ.

Определение 3.F1. (ДКНФ) [378]. Пусть R^* – реляционная схема, находящаяся в 1НФ. И пусть G – набор зависимостей $DD(R)$ и $KD(R)$ схемы. R находится в ДКНФ, если $G \Rightarrow g$ для любого ограничения g схемы R^* .

По аналогии с [378], введем понятие аномалий вставки и удаления. Для этого прежде определим понятие кортежа, совместимого со схемой отношения.

Определение 3.F2. (совместимого кортежа). Пусть R^* – реляционная схема, R – допустимый экземпляр реляционной схемы R^*. Пусть t – произвольный кортеж, не принадлежащий R. Кортеж t будет *совместимым с* R (в контексте R^*), если:

- кортеж t имеет те же атрибуты, что и R,
- для любого атрибута A и любой зависимости $DD(R)$ $IN(A,S)$ схемы R^* значение атрибута A для кортежа $t \in S$,
- для любой зависимости $KD(R)$ $KEY(K)$ схемы R^* и любого кортежа s, принадлежащего отношению R, выполняется $t(K) \neq s(K)$.

Таким образом, совместимый кортеж является «вставляемым» в R без нарушения каких-либо зависимостей $DD(R)$ или $KD(R)$.

Определение 3.F3. (аномалии вставки). Реляционная схема R^* имеет *аномалию вставки*, если есть допустимый экземпляр R схемы R^* и есть кортеж t, совместимый с R, такой, что $R \cup \{t\}$ – отношение, полученное вставкой t в R, не являющееся допустимым экземпляром схемы R^* (то есть, этот экземпляр не удовлетворяет хотя бы одному ограничению R^*). Пример 3.1. Пусть R^* – реляционная схема с атрибутами *Преподаватель*, *Кафедра* и *Метод* и со следующими ограничениями (где *Char20* – множество строк символов, длиной не больше, чем в 20 символов):

$DD(R)$=IN(*Преподаватель*, *Char20*)$\cap$IN(*Кафедра*, *Char20*)$\cap$IN(*Метод*, *Char20*)

$KD(R)$=KEY(*Преподаватель*)$\Rightarrow$*Преподаватель*$\rightarrow$(*Кафедра* + *Метод*)

Кафедра$\rightarrow$*Метод*

Как отмечалось ранее, по умолчанию предполагается замкнутость ограничений под логическими следствиями, т.е. под $\Rightarrow$. Схема R^* не находится в НФБК, или даже в 3НФ [361], так как в ней существует «транзитивная зависимость» столбца *Метод* от *Преподаватель*:

Преподаватель $\rightarrow$ *Кафедра* и *Кафедра* $\rightarrow$ *Метод*. В [378] показано, что R^* имеет аномалию вставки. Пусть R – это первые три кортежа отношения, показанного в табл. 3.1.

Это отношение – допустимый экземпляр схемы R^*. Пусть t – кортеж (Александров, Математики, Эйлер). Кортеж t совместим с отношением R. Однако, если вставить кортеж t в отношение, оно не является допустимым экземпляром схемы, так как вставка нарушает ФЗ *Кафедра* $\rightarrow$ *Метод*.

Таблица 3.1.

ПРЕПОДАВАТЕЛИ+КАФЕДРЫ+МЕТОДЫ (*2НФ*)

Преподаватель	*Кафедра*	*Метод*
Иванов	Математики	Гаусса
Петров	Математики	Гаусса
Васильев	Компьютерных наук	Неймана
Александров	Математики	Эйлера

Таким образом, найдена аномалия вставки. Неформально объяснить аномалию вставки кортежа (Александров, Математики, Эйлер) можно так: не ясно в чем цель вставки: в том, что бы просто вставить информацию о новом преподавателе Александрове, или в том, чтобы также обновить имя автора метода, который преподается.

Пример 3.2. Пусть R^* – реляционная схема с атрибутами *ABC* и со следующими ограничениями (где *Nat* – множество натуральных чисел {0,1,2,…}):

$DD(R)$=IN(A, *Nat*)$\cap$IN(B, *Nat*)$\cap$IN(C, *Nat*)

$KD(R)$=KEY(*ABC*)

$A \rightarrow\rightarrow B/C$

Схема R^* не находится в 4НФ (хотя она находится в НФБК), так как в ней содержится МЗ $A \rightarrow\rightarrow B/C$, и A не является ключом. В [378] показано, что R^* имеет аномалию вставки.

Пусть R – отношение табл. 3.2. Это отношение – допустимый экземпляр схемы R^*. Пусть t – кортеж (0, 1, 1). Кортеж t совместим с отношением R. Однако, вставив кортеж t в R, мы бы получили отношение в табл. 3.2., которое не является допустимым экземпляром схемы, так как нарушена МЗ $A \to\to B/C$.

Таблица 3.2.

$A+B+C$ (*НФБК*)

A	B	C
0	0	0
1	0	0
0	1	1

Определение 3.F4. (аномалии удаления). Реляционная схема R^* имеет *аномалию удаления*, если есть допустимый экземпляр R схемы R^* и кортеж t в R такой, что отношение, полученное удалением t из R, не является допустимым экземпляром схемы R^* (то есть, нарушает ограничение схемы R^*).

Пример 3.3. Пусть R^* – та же реляционная схема, что и в примере 3.2. В [378] показано, что R^* имеет аномалию удаления. Пусть R – отношение приведенное в табл. 3.3. Это отношение – допустимый экземпляр схемы R^*. Пусть t – кортеж (1, 1, 1). Если удаляется кортеж t из отношения R, то полученное отношение, показанное табл. 3.3, не является допустимым экземпляром схемы, так как нарушена МЗ $A \to\to B/C$.

Таблица 3.3.

$A+B+C$ (*НФБК*)

A	B	C
1	0	0
1	0	1
1	1	0
1	1	1

Теперь выпишем из [378] основную теорему о ДКНФ. Она будет базовой для дальнейших выкладок.

Теорема 3.F1. (о ДКНФ). Выполнимая 1НФ-реляционная схема находится в ДКНФ тогда и только тогда, когда в ней нет аномалий вставки или удаления.

Из этой теоремы следует очень важный вывод. Если Σ - класс зависимостей, целиком состоящий из *DD*(*R*) и *KD*(*R*), то Σ определяет нормальную форму, равносильную ДКНФ. Очевидно также, что если $\vartheta_1 \subseteq \vartheta_2$, то ϑ_1-я нормальная форма заключается в ϑ_2 нормальной форме.

Таким образом, СУБД должна иметь возможность приводить в исполнение зависимости *DD*(*R*) и *KD*(*R*). Так как домены и ключи уже поддерживаются любой СУБД, необходимо, чтобы все ограничения схемы могли бы приводиться в исполнение, просто выполняя действия над доменами и ключами, которые в точности описывают ДКНФ. Таким образом, если схема находится в ДКНФ, то не нужно дополнительно сверх того организовывать еще и поддержку ограничений схемы ДКНФ. Более того, СУБД может вообще не иметь возможности поддержки других ограничений кроме тех, которые автоматически приводится в исполнение, выполняя действия над доменами и ключами.

3.4. Нормальные формы и ДКНФ

Пусть Δ – набор зависимостей ключей *KD(R),* Θ – набор зависимостей доменов *DD(R)*, и g – классические ограничения схемы R - ФЗ, МЗ, ЗПС. Приведем определения основных классических НФ.

НФБК. 1НФ схема отношения R находится в НФБК, если $\Delta \Rightarrow g$ для каждой ФЗ схемы R. Таким образом, в НФБК каждая ФЗ – следствие зависимостей ключей.

4НФ. 1НФ схема отношения R находится в 4НФ, если $\Delta \Rightarrow g$ для каждой МЗ схемы R. Таким образом, в 4НФ каждая МЗ – следствие зависимостей ключей.

5НФ. 1НФ схема отношения R находится в 5НФ, если $\Delta \Rightarrow g$ для каждой ЗПС схемы R. Таким образом, в 5НФ каждая ЗПС – следствие зависимостей ключей.

Таким образом, существует аналогия между определениями НФБК, 4НФ и 5НФ. Но в отличие от ДКНФ, ни одна из этих более ранних НФ не рассматривала роль доменов. Задача ДКНФ исправит это упущение.

Изменение состоит в том, что учитывается влияние зависимостей доменов *DD*(*R*). В [378] доказано, что первоначальные и модифицированные версии определений эквивалентны, если никакой из доменов не является слишком маленьким. И доказано также, что ДКНФ$\Rightarrow$5НФ$\Rightarrow$4НФ$\Rightarrow$НФБК$\Rightarrow\ldots\Rightarrow$1НФ. Таким образом, если все домены бесконечны, то ДКНФ заключает в себе 5НФ (и, конечно, 4NF, НФБК, 3НФ, 2НФ и 1НФ).

Пусть Ω – набор ограничений схемы. Для каждого атрибута A определим $DD_{\Omega}(A) = \cap\{S : \Omega \Rightarrow IN(A,S)\}$. Таким образом, $DD_{\Omega}(A)$ – это набор всех возможных значений, которые может принять значение атрибута A для кортежа отношения, если отношение удовлетворяет ограничению Ω. Если Ω - набор ограничений схемы отношения, то $DD_{\Omega}(A) = \cap\{S : IN(A,S) \in \Theta\}$, где Θ - набор зависимостей доменов *DD*(*R*) схемы. Это следует из того предположения, что набор ограничений схемы замкнут под логическим следствием.

В частности, тогда каждая зависимость домена *DD*(*R*), связанная с ограничениями схемы, находится в Θ (наборе зависимостей доменов *DD*(*R*) схемы). Точно так же, каждая зависимость ключей *KD*(*R*), связанная с ограничениями схемы, находится в Δ (наборе зависимостей ключей *KD*(*R*) схемы). Функцию домена обозначаем упрощенно $DD(A)$, а также под $|DD(A)|$ подразумеваем число элементов $DD(A)$.

Определения НФБК, 4НФ и 5НФ не рассматривают влияние зависимостей доменов *DD*(*R*). Таким образом, для некоторых введенных ранее ограничений *g*, определения рассматривают вопрос: выполняется или

нет условие $\Delta \Rightarrow g$, а не условие $G \Rightarrow g$. Предположим, что $|DD(A)| \geq 2$ для каждого атрибута A. Пусть g – ФЗ или МЗ.

Таким образом, для таких важных зависимостей, как ФЗ или МЗ (как ограничений схемы g) классические определения НФ рассматривают вопрос выполнения условия $KD(R) \Rightarrow g$, а не условие $G \Rightarrow g$.

Следующая лемма Фейджина, доказанная в [378], вводит условие, при котором между $KD(R) \Rightarrow g$ и $G \Rightarrow g$ нет никакого различия.

Лемма 3.F1. (о роли доменов). Ограничения схемы g являются следствием ограничений на ключи $\Delta \Rightarrow g$ тогда, и только тогда, когда $G \Rightarrow g$, где $G = \Delta \cup \Theta$.

Пример 3.4. Пусть R^* – схема отношения с атрибутами ABC, и со следующими ограничениями:

$DD(R)$=IN(A, {1})$\cap$IN(B, *Nat*)$\cap$IN(C, *Nat*)

$KD(R)$=KEY(BC)

Пусть g: $\varnothing \rightarrow A$, где $\varnothing$ - пустое множество. Согласно формальному определению ФЗ специальная ФЗ $\varnothing \rightarrow A$, где левая сторона - пустое множество, говорит то, что значение атрибута A для каждого кортежа одинаково. Тогда g - ограничение схемы, так как g - логическое следствие первой зависимости доменов $DD(R)$, указанной выше. Можно принять в качестве ограничения g $B \rightarrow A$. Этот пример будет точно так же работать. Пусть Δ – множество, состоящее из одного ключа $KEY(BC)$, пусть Θ – набор, состоящий из трех перечисленных выше зависимостей $DD(R)$, и пусть $G = \Delta \cup \Theta$. Тогда $G \Rightarrow g$, несмотря на то, что ложно $\Delta \Rightarrow g$. Очевидно, что $|DD(A)| = 1$. Таким образом, лемма 3.F1. не выполняется, если мы опускаем предположение, что $|DD(A)| \geq 2$.

3.5. Безаномальность реляционного каркаса

С учетом этих утверждений, доказанных в [378], рассмотрим особое отношение (названное так в [207]) со схемой $R(X,A)$, где X - единственный необязательно простой ключ. По аналогии с [207] назовем также *особыми* следующие ограничения этой схемы.

Определение 3.1. *Особыми* будем называть следующие ограничения схемы $R(X,A)$:

$$DD(R) = IN(A,\ \forall A\ (A \neq const)) \cap IN(X,\ \forall X\ (X \neq const)),$$

$$KD(R) = KEY(X) \Rightarrow X \to A$$

Таким образом, особые ограничения – это такая совокупность, в которой ограничения на домены: любые значения атрибутов X и A, не являющиеся константой (и, в том числе, пустым множеством), ограничения на ключи: X – единственный ключ отношения, а дополнительные ограничения: наличие ФЗ $X \to A$ – это естественное следствие из ограничения ключа.

Отметим, что любое отношение со схемой $R_i(X_1,X_2,...,X_i,A_1,A_2,...,A_j)$, где $(X_1 + X_2 + ... + X_i) \to (A_1 + A_2 + ... + A_j)$, на основании аксиом Армстронга [135] может быть заменено равносильным отношением со схемой $R_i'(X,A)$ и единственной ФЗ $X \to A$, где $X = (X_1 + X_2 + ... + X_i)$, а $A = (A_1 + A_2 + ... + A_j)$.

Сформулируем это утверждение в виде леммы**.**

Лемма 3.1. Выполнимое отношение со схемой $R_i(X_1,X_2,...,X_i,A_1,A_2,...,A_j)$ и ограничениями: $(X_1 + X_2 + \ldots + X_i)$ – единственный ключ отношения и ограничения на домены строго соответствуют ограничениям на домены в особых ограничениях, может быть заменено отношением с особой схемой $R_i'(X,A)$, где $X=(X_1+X_2+,...,+X_i)$ и $A=(A_1+A_2+,...,+A_j)$ тогда, и только тогда, когда любая совокупность ФЗ исходной схемы является следствием ключа.

Доказательство.

Необходимость. Пусть в схеме $R_i(X_1,X_2,...,X_i,A_1,A_2,...,A_j)$ имеется единственный ключ $(X_1 + X_2 + ... + X_i)$. Тогда следствием ключа в отношении имеется ФЗ $(X_1 + X_2 + ... + X_i) \to (A_1 + A_2 + ... + A_j)$. Из этого следует, что возможные оставшиеся ФЗ имеют своими детерминантами только части ключа X_i. Действительно, в данной схеме отсутствуют любые ФЗ с

составным детерминантом на произвольной совокупности A_j из неключевых атрибутов – в противном случае в схеме был бы не один ключ. Покажем, что, эти ФЗ являются следствием ключа, т.е. могут быть выведены из «ключевой» ФЗ. Пусть в схеме $R_i(X_1, X_2, ..., X_i, A_1, A_2, ..., A_j)$ имеется множество ФЗ $X_m \to X_k$. Тогда, на основании аксиомы Армстронга о пополнении, можем заменить любую произвольную ФЗ $X_m \to X_k$ на $(X_m + X_1 + X_2 + \ldots) \to (X_k + X_1 + X_2 + \ldots)$, что по аксиоме о самодетерминированности будет сведено к $(X_1 + X_2 + \ldots + X_i) \to (X_1 + X_2 + \ldots + X_i)$, т.е., к ключу.

Достаточность. Пусть в схеме $R_i(X_1, X_2, ..., X_i, A_1, A_2, ..., A_j)$ имеется множество ФЗ $X_m \to X_k$, которые не являются следствием ключа. Это означает, что их невозможно вывести из совокупности $(X_1 + X_2 + \ldots + X_i)$. Тогда в совокупности детерминанта X_m находятся еще и неключевые атрибуты. Но тогда это означает, что имеется еще один ключ. Противоречие доказывает достаточность. □

Тогда справедливо еще одно важное свойство каркасного отношения, которое также сформулируем в виде леммы.

Лемма 3.2. В выполнимом каркасном отношении со схемой $R_i(X_1, X_2, \ldots, X_i)$, полученном декартовым произведением атрибутов $(X_1, X_2, \ldots, X_i)$, (оператором роста [207]), с ограничениями: $(X_1 + X_2 + \ldots + X_i)$, – единственный ключ отношения и ограничения на домены соответствуют ограничениям на домены в особых ограничениях, любая совокупность ФЗ является следствием ключа.

Доказательства не приводим в связи с его очевидностью.

Это значит, что в схеме шунтированного отношения $R_i(X_1, X_2, ..., X_i, A_1, A_2, ..., A_j)$ любая совокупность ФЗ на частях единственного ключа как на детерминантах, причем так, что в этих зависимостях не участвует ни один из неключевых атрибутов $(A_1, A_2, \ldots, A_j)$, не противоречит ограничениям ключа. Это очень важно для проектировщика, так как означает, что такая замена будет адекватной для разных совокупностей ФЗ,

которыми обладают составные ключи при шунтировании обобщений ДЗ (МЗ, ЗПС) [205].

Это утверждение обосновывает адекватность моделирования многоарной связи (с атрибутами) между множеством атомарных сущностей отношением со схемой, аналогичной схеме отношения, моделирующего единственную атомарную сущность. И полностью подтверждает идею Чена [359] о равносильности сущностей и связей.

В работе [377] рассмотрен характерный пример, подтверждающий справедливость выводов лемм 3.1 и 3.2

Пример 3.5. Пусть R^* – реляционная схема, представленная в табл. 3.4. Схема R находится в 3НФ и даже в более «сильной» НФБК [377], так как это – «весь ключ». То есть, никакое собственное подмножество четырех столбцов не формируют ключ. Однако отношение не находится в 4НФ.

Таблица 3.4.

СЛУЖАЩИЙ+ЗАРПЛАТА+ДЕТИ+ГОД (*НФБК*)

Служащий	*Зарплата*	*Дети*	*Год*
Иванов	1000	Сергей	2013
Иванов	1500	Сергей	2013
Петров	1200	Александр	2012
Петров	1300	Александр	2013
Петров	1200	Татьяна	2012
Петров	1300	Татьяна	2013
Сидоров	1400	Анатолий	2012
Сидоров	1700	Анатолий	2013

В [377] указано, что «хотя в R нет никаких ФЗ, в нем есть МЗ $Служащий \rightarrow\rightarrow Дети$, и также $Служащий \rightarrow\rightarrow \{Зарплата, Год\}$. История зарплаты служащего полностью определена служащим и ортогональная к его множеству детей».

Таким образом, зависимость вида $Дети \rightarrow Служащий\ (G:1)$ или $(Служащий + Год) \rightarrow Зарплата\ (G:1)$ не считается ФЗ, так как их

детерминанты находятся в более сильной зависимости - МЗ. Хотя, тем не менее, «история зарплаты служащего полностью определена служащим» и «… множество детей служащего полностью определено служащим» [377].

Сформулируем несколько утверждений, которые непосредственно следуют из теоремы Фейджина о ДКНФ. При этом формально приведем доказательства этих утверждений, не смотря на их очевидность. Имеет место следующая лемма.

Лемма 3.3. Выполнимая простейшая реляционная схема $R(X,A)$ с единственным ключом X не имеет аномалий вставки или удаления тогда и только тогда, когда в ней отсутствуют ограничения, которые не следуют из особых ограничений.

Доказательство.

Достаточность. Простейшая схема $R(X,A)$ имеет унарные X,A. Предположим, что в $R(X,A)$ не содержится ограничений, которые логически не следуют из особых. Это означает, что в схеме $R(X,A)$ существуют только ограничения, которые строго следуют из ограничений на домены и ключи. Покажем, что $R(X,A)$ не имеет аномалий вставки или удаления. Пусть r_1 – допустимый экземпляр схемы $R(X,A)$. Мы должны показать, что отношение r_2, которое получено из $R(X,A)$ или вставкой кортежа, совместимого с $R(X,A)$, или удалением кортежа из $R(X,A)$, является также допустимым экземпляром. Пусть G – набор зависимостей схемы $R(X,A)$. В соответствии с леммой Фейджина о роли доменов для простейшей схемы $G = DD(R) \cup KD(R)$. Легко проверить, что, так как $R(X,A)$ удовлетворяет условиям G, то и r_1 удовлетворяет условиям G. Чтобы показать, что r_2 является допустимым экземпляром $R(X,A)$, мы должны показать, что r_2 удовлетворяет также и каждому из ограничений g. Но g – это единственная ФЗ $X \to A$, которая является естественным следствием ограничений на ключи $KD(R)$ отношения $R(X,A)$. То есть, в отношении отсутствуют ограничения, которые не следуют из G. Значит, по определению $R(X,A)$ находится в ДКНФ. Следовательно, так

как r_2 удовлетворяет G и так как $G \Rightarrow g$, то из этого следует, что r_2 удовлетворяет g.

Необходимость. Теперь предположим, что в отношении нет аномалий вставки и удаления. Покажем, что в схеме присутствуют ограничения, которые логически следуют из особых ограничений. Поскольку в схеме нет аномалий, она находится в ДКНФ. Тогда по теореме 3.F1. (Фейджина, [378]) ограничения, которые находятся в схеме отношения, являются логическим следствием доменов и ключей. Но это возможно только, если они являются следствием особых ограничений, потому что особые ограничения – это совокупность исключительно ограничений на домены и ключи.□

Необходимо заметить, что описанное леммой 3.3 «минимальное» отношение также было рассмотрено Р. Фейджиным (в работе [378] – пример 3.10). Однако дополнительные ограничения на такую схему, названные там *зависимостями вложения* (когда каждое вхождение в X должно также входить и в A, обозначается $X \subseteq A$), привели схему к аномалиям удаления. При этом в [378] указано: «Более общая схема, в которой единственными ограничениями являются ФЗ, может всегда простым способом преобразовываться в ДКНФ-схему БД». Для каждой ФЗ $X \rightarrow Y$ исходной схемы, мы формируем новую схему отношения с атрибутами XY и с единственным ограничением *KEY*(*X*)». Поэтому, по аналогии с [378], приведенное доказательство [208] будем считать исчерпывающим всюду до тех пор, пока нет *«контр-примера»* отношения *R*(*X*,*A*) такого, что особые ограничения удовлетворяют *R*, но и такого, что найдутся такие иные ограничения-следствия особых $DD^{'}(R)$ и $KD^{'}(R)$, что *R* станет аномальным.

Введем определение схемы БД. Согласно [378] *схема БД* – это набор реляционных схем, каждая из которых представляет собой набор атрибутов и набор ограничений.

Определение 3.2. Каркасную совокупность отношений будем называть *замкнутой*, если в ней нет ни одного отношения, у которого хотя бы одна

часть простого или составного ключевого атрибута встречается в совокупности единственный раз.

Интуитивно понятно, что такая совокупность моделирует замкнутые связные ПрО.

Доказанная лемма 3.3. [208], а также доказанная в [205] теорема о шунтировании ДЗ (теоремы 2.3., 2.6. и 2.7.) позволяют сформулировать итоговые утверждения, завершающее построение базовой части теории РК.

Имеет место очень важное для проектирования безаномальных схем БД утверждение, доказанное в [208].

Теорема 3.1. Схема БД, построенная на замкнутой совокупности шунтированных каркасных отношений, не имеет аномалий вставки и удаления, если и только если ограничения каждого отношения является логическим следствием особых ограничений.

Доказательство.

Необходимость. Пусть каркасная совокупность $R_{i-1}(X_1,...,X_{i-1},A_{i-1})$, $R_i(X_1,...,X_i,A_i)$, $R_{i+1}(X_1,...,X_{i+1},A_{i+1})$ является шунтированной. И пусть A_i – шунтирующие атрибуты каждого многоарного каркасного отношения. Тогда в соответствии с леммами 3.1. и 3.2. каждое шунтированное отношение может быть заменено равносильным отношением со схемой $R_i'(X_i,A_j)$ и единственной ФЗ $X_i \to A_j$, где $X_i = (X_1 + X_2 + \ldots + X_i)$, а $A_j = (A_1 + A_2 + \ldots + A_i)$. Пусть все ограничения на домены $DD_i(X_i,A_j)$ и ключи $DK_i(X_i)$ не противоречат особым ограничениям. Поскольку по определению ДКНФ каждое отношение всей совокупности находится в этой НФ, в соответствии с леммой 3.3 каждое такое отношение не имеет аномалий вставки и удаления.

Достаточность. Пусть шунтированная каркасная совокупность $R_{i-1}(X_1,...,X_{i-1},A_{i-1})$, $R_i(X_1,...,X_i,A_i)$, $R_{i+1}(X_1,...,X_{i+1},A_{i+1})$ не имеет аномалий. Тогда по теореме Фейджина она находится в ДКНФ. Но в соответствии с леммой 3.3 это возможно лишь при условии, что ограничения каждого отношения являются следствием особых ограничений. □

Из этого утверждения следует простой вывод для проектировщика: если для некоторой ПрО схема БД проектируется на каркасной совокупности шунтированных отношений, и если для некоторых отношений из этой совокупности обнаруживаются ограничения на домены и ключи, противоречащие особым ограничениям, то эти ограничения из схем отношений должны быть исключены. В противном случае такая схема БД будет иметь аномалии.

Однако из теорем о полноте, единственности и замкнутости РК следует еще одно, даже более сильное утверждение.

Теорема 3.2. (достаточное условие безаномальности реляционной схемы БД). Схема произвольной БД не имеет аномалий вставки и удаления, если она построена на замкнутой совокупности шунтированных каркасных отношений, каждое из которых имеет ограничения, логически следующие из особых.

Доказательство. Проведем от противного. Пусть имеется иная совокупность реляционных отношений, имеющая безаномальную схему для произвольной совокупности ограничений. Пусть тогда согласно леммам 3.2 и 3.3 имеется, в том числе, и совокупность отношений $Z_{i-1}(X_1,\ldots,X_{i-1},A_{i-1})$, $Z_i(X_1,\ldots,X_i,A_i)$, $Z_{i+1}(X_1,\ldots,X_{i+1},A_{i+1})$, схема каждого из которых является безаномальной. То, что лемма 3.3 может иметь множество «обратных решений», не имеет значения, так как нас интересует именно тот факт, что одним из обратных решений утверждения леммы 3.3 будет обязательно некоторая совокупность отношений со схемой $Z(Y,B)$, ограничения которой – особые. Но поскольку такая совокупность является замкнутой по условию теоремы – это схема БД некоторой связной ПрО, она образует фрагмент реляционного каркаса. Тогда имеются два разных фрагмента каркаса. Но это противоречит теореме о единственности реляционного каркаса [233]. Такое противоречие доказывает теорему.□

При этом совокупность отношений, каждое из которых находится в ДКНФ, будем называть ДКНФ-схемой БД. Проектировщику вполне

достаточно знать, что, по крайней мере, на РК он может получить безаномальную ДКНФ-схему БД при условии, что все ограничения ПрО будут являться следствием особых.

Отметим, что выводы теорем 3.1. и 3.2. полностью соответствует идее Д.М. Кренке [131] о «единственности темы отношения». Если отношение моделирует высказывание с единственной темой, оно соответствует критериям ДКНФ. Единственный унарный или составной ключевой атрибут в этом контексте является семантически атомарным или составным многоместным предикатом [232] такого высказывания. То есть – его «темой». Потому, что каждая шунтированная «ячейка» РК, то есть каждое актуальное отношение-связь, является высказыванием. Актуальность отношения-связи констатирует некий факт из ПрО.

По нашему мнению, именно Д.М. Кренке впервые ввел понятие «тема» как логическое следствие ключа: один ключ – одна тема. Он тем самым интуитивно подсказал, что у ключа отношения, если он единственен, появляется еще одна важная миссия, нежели только сохранность целостности отношения и доступ к данным, - быть носителем семантики отношения.

И вывод о том, что в отношении с ДКНФ-схемой не может быть более одного ключа и более одного детерминанта неключевых атрибутов, также предложил Д.М. Кренке. «Если в отношении имеется три атрибута $R(A,B,C)$ и при этом $(A,B)\rightarrow C$, то не должно быть еще и зависимости $A\rightarrow B$. В противном случае по этой зависимости отношении должно быть декомпозировано» [131]. Отношение, построенное на РК, автоматически обладает единственной ФЗ.

3.6. Актуализация ячеек каркаса

Наша цель – исследовать РК, предложенный и формализованный в [207, 208], а также продемонстрировать в его свойствах важные конкурентные преимущества, которые позволят проектировщику использовать эту совокупность отношений как базовый шаблон для проектирования схем БД произвольных ПрО. И при этом показать

диаметрально противоположный к общепринятому подход в проектировании схем БД – не подстраиваться под каждую, часто необоснованную, специфику ограничений ПрО, а наоборот, на первом этапе отобрать в ПрО те ограничения, специфика которых является логическим следствием заведомо корректных. А ограничения ПрО, которые противоречат безаномальной схеме, реализовать на втором этапе в отдельных отношениях. Но и их схемы проектировать также на РК. Опыт проектирования схем БД позволяет заключить, что на этом шаблоне может быть реализовано подавляющее большинство ограничений ПрО.

В работе [208] показано, что теория РК – это обоснование алгоритма синтеза ДКНФ-схемы БД [378]. А благодаря теореме об «устойчивости РК к модификациям» [232], еще и строгое обоснование гипотезы о максимальной степени модифицируемости ДКНФ-схемы БД, смело высказанной в [2, 294]. Однако, с той лишь оговоркой, что [2] является частным случаем [204, 208]. При этом, существенным в практике каркасного анализа ПрО является то обстоятельство, что РК показывает совокупность всех актуальных отношений. Поэтому статическая ER-диаграмма [359] становится менее эффективной, чем каркасная диаграмма.

В работе [131] Д.М. Кренке, цитируя Р. Фейджина, утверждает, что «поскольку отношения в ДКНФ не должны иметь аномалий модификации, предотвратить возникновение этих аномалий может сама СУБД, контролируя соблюдение ограничений доменов и ключей». В каркасной схеме БД ограничения ПрО естественным образом могут быть наложены на домены и ключи атомарных в смысле [232] сущностей-объектов.

Проектировщик может использовать фундаментальную природу доменов и ключей для РБД. А СУБД должна позволять приводить в исполнение зависимости ключей KD и зависимости доменов DD (например, таких DD, которые утверждают, что домен данного атрибута - строки шести символов). Так как домены и ключи так или иначе поддерживаются, необходимо, чтобы все ограничения ПрО приводились в исполнение просто

выполняя действия над доменами и ключами (которые в точности описывают ДКНФ).

Таким образом, если схема БД находится в ДКНФ, то не нужно дополнительно сверх того организовывать еще и поддержку ограничений схемы ДКНФ. Фактически, СУБД может не иметь возможности поддержки других ограничений кроме тех, которые автоматически приводится в исполнение, выполняя действия над доменами и ключами.

В КМД отношения, которые моделируют произвольные связи атомарных сущностей-объектов, обладают строгой обусловленностью доменами и ключами унарных отношений. Если бы это было не так, каркасная схема не соответствовала бы критериям целостности. И не моделировала бы ПрО – целостной полной замкнутой и связной системы [49, 239, 283].

Замкнутую каркасную совокупность шунтированных [205] отношений со схемами $R_{i-1}(X_1,...,X_{i-1},A_{i-1})$, $R_i(X_1,...,X_i,A_i)$, $R_{i+1}(X_1,...,X_{i+1},A_{i+1})$ будем называть *обусловленной* доменами и ключами этой совокупности, если ограничения на домены и ключи не противоречат особым ограничениям, а также, если выполняется строгое тождество значений соответствующих одноименных ключевых атрибутов.

Строгое тождество значений соответствующих одноименных ключевых атрибутов – это не что иное, как ссылочная целостность. Она подразумевает еще и ограничение на удаление кортежей, на которые ссылаются иные кортежи (хотя бы на один ключевой атрибут), пока они не удалены. Или разрешено каскадное удаление. Но это ограничение обеспечивает большинство СУБД автоматически.

В такой совокупности каждое отношение может выступать как в роли «поставщика» домена для ключевых атрибутов любой совокупности иных отношений, так и в роли «получателя» домена для своих ключевых атрибутов – ключевые атрибуты отношения следующего каскада каркасной совокупности обусловлены всеми предыдущими отношениями, то есть,

доменами этих ключей. Домены следующих каскадов отношений обусловлены ключами предыдущих каскадов. При этом шунтирующие атрибуты также строго определяются соответствующими доменами из ПрО.

Дадим определение понятию «актуальная связь в ПрО». При этом оговорим, что в этом разделе время как параметр динамики системы не рассматривается. Рассматриваются лишь статические срезы ПрО, и потому анализируются закономерности синтеза схем БД лишь для этих срезов.

Определение 3.3. *Актуальной ячейкой РК* называется многоарное отношение $R\ (X_1, X_2 \ldots, X_n, A_j)$, если оно имеет единственный составной ключ $(X_1 + X_2 + \ldots + X_n)$, шунтировано [205] хотя бы одним функционально зависящим от всего ключа атрибутом A_j, так, что $(X_1 + X_2 + \ldots + X_n) \to A_j$, а также входит в РК.

На рис. 3.1. показана совокупность актуальных ячеек РК – схема БД.

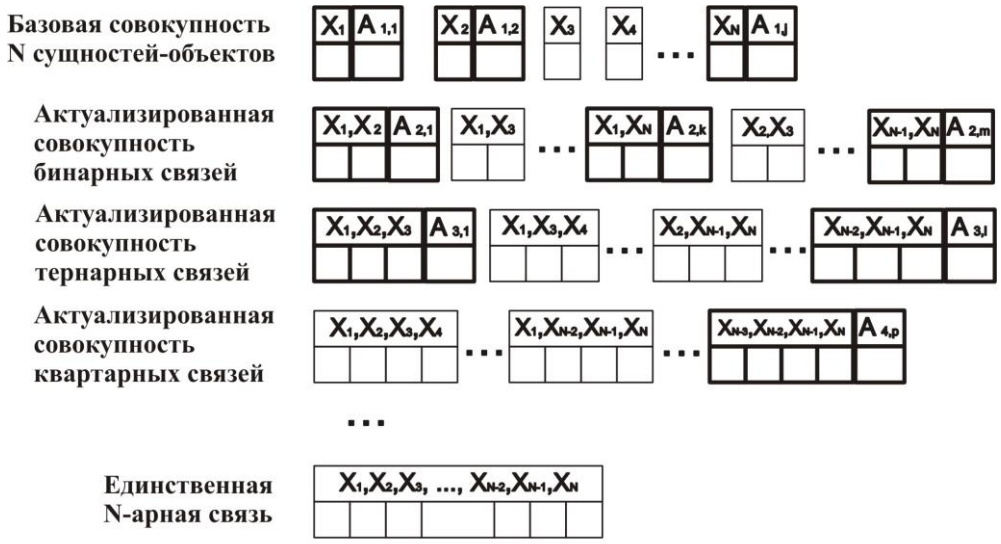


Рис. 3.1. Совокупность актуальных ячеек РК – схема БД.

Необходимо подчеркнуть, что условие непротиворечивости ограничений (обусловленность схем совокупности отношений) необходимы проектировщику, чтобы схемы БД были *выполнимыми* [378]. Там же показаны примеры того, как ограничения могут быть непротиворечивыми и схема выполнимой, но логическая взаимосвязь между ограничениями отсутствовать – тогда схема становится аномальной.

3.7. Модифицируемость и безаномальность

Именно совокупность актуальных ячеек РК, или, другими словами, актуальная каркасная совокупность отношений, лишена аномалий вставки и удаления кортежей для любой ПрО при условии, что никакие дополнительные ограничения из ПрО не противоречат *особым* ограничениям, веденным в [207].

Тогда имеет место следующая **Теорема 3.3.** (о модифицируемой ДКНФ-схеме БД). Для того, чтобы схема БД обладала одновременно свойствами и модифицируемости, и безаномальности, необходимо и достаточно, чтобы данная схема была сформирована на замкнутой совокупности актуальных ячеек РК, и ограничения на каждое отношение являлось бы логическим следствием особых ограничений.

Доказательство. Справедливость этого утверждения следует из справедливости доказанных выше теорем 3.1. и 3.2. Действительно, достаточность обеспечена тем, что по условиям теоремы ограничения актуальных ячеек РК не только обусловлены (т. е., не противоречивы), но еще и являются следствием особых.

А необходимость доказывается на основании основных свойств РК, также строго доказанных. Из того факта, что совокупность отношений сформирована на актуальных ячейках РК, следует, что данная совокупность обладает модифицируемостью РК. Но модифицируемость схемы БД, построенной на РК, является следствием его полноты, единственности и замкнутости [208]. Поэтому никакая иная совокупность реляционных отношений не имеет аналогичных свойств.□

Этот же вывод можно получить еще и на основании доказанных теорем о шунтировании обобщений ДЗ, полноте, единственности и замкнутости РК. Действительно, если бы РК не обладал возможностью исключать разновидности ДЗ (не был бы шунтируемым в смысле [205]), то имела бы место аномалия типа ДЗ (МЗ, ЗПС). Если бы РК в рамках реляционной модели был бы ни единственным, ни полным [233], то имелась бы

альтернативная совокупность отношений, обладающая аналогичными свойствами. Но тогда это противоречило бы целостности реляционной алгебры. Однако РК единственен. Если бы он не был бы полон, то не обладал бы свойством независимости от произвола ПрО. И не обладал бы инвариантностью для произвольной совокупности запросов к схемам. А из теоремы о замкнутости путей нормализации РК [234] следует, что на РК может быть построена вся совокупность схем. Пересечение всех перечисленных свойств обеспечивается единственно возможной схемой БД – 5НФ [378]. Соответствие шунтированного каркасного отношения для любых ПрО критериям 5НФ доказано строго [205].

Но теорема о замкнутости путей нормализации РК [234] позволяет обосновать еще и тот факт, что каркасная совокупность 5НФ-отношений - схема более сильная, потому, что каркас обеспечивает еще и полную взаимообусловленность всех атрибутов, чего, по сути, и требует определение ДКНФ [378]. В этой схеме присутствуют только ключевые атрибуты, обусловленные только друг другом и своими доменами. А также шунтирующие атрибуты, также обусловленные только ключевыми атрибутами, доменами ключевых атрибутов и своими доменами. Таким образом, для любых ограничений ПрО, наложенных на домены и ключи этой совокупности, каждое отношение из этой совокупности удовлетворяет критериям ДКНФ [378] только тогда, когда эти ограничения не противоречат особым.

3.8. Шунтирование в терминах ограничений на ключи

В [367] Дейт и Феджин в 1992 году предоставили проектировщику простые и эффективные критерии высокой нормализованности схем БД. Однако по некоторым причинам наиболее распространенным критерием качества схемы БД была признана усредненная 3НФ. Такая денормализация схем вызвана потребностью пользователя в оптимизации между скоростью доступа к данным и числом таблиц схемы БД, влияющих на число операций

соединения. Но, как показано ранее, иное решение этой проблемы основано на шунтировании ДЗ (МЗ-ЗПС) .

В [367] приведены следующие определения.

1. Реляционная схема $R(X,Y)$, где X – суперключ и $X \to Y$ является нетривиальной ФЗ, находится в НФБК тогда и только тогда, когда любая ФЗ схемы является логическим следствием ограничений ключа схемы.

2. Реляционная схема $R(X,Y)$, где X – суперключ и $X \to\to Y$ является нетривиальной МЗ, находится в 4НФ тогда и только тогда, когда любая МЗ схемы является логическим следствием ограничений ключа.

3. Реляционная схема $R(X,Y)$, которая содержит нетривиальную ЗПС и в которой где X – суперключ, находится в 5НФ тогда и только тогда, когда любая ЗПС схемы является логическим следствием ограничений ключа.

Под термином «суперключ» в РМД принято называть подмножество атрибутов отношения, удовлетворяющее требованию уникальности и минимальной составности.

Во-первых, не существует двух кортежей данного отношения, в которых значения этого подмножества атрибутов совпадают, а во-вторых, на суперключ не накладывается требование недекомпозируемости. Последнее требование означает, что в составе ключа отсутствует меньшее подмножество атрибутов, удовлетворяющее условию уникальности. Вследствие этого в состав суперключа не может входить другой, более «компактный» по количеству атрибутов суперключ.

Под термином «…зависимости должны быть логическим следствием ограничений ключа» понимается то, что детерминанты зависимости (левые части) должны быть детерминантами ключей. Например, в [378] «схема R не находится в 4НФ, хотя она находится в НФБК, так как в ней содержится МЗ $X \to\to Y$, и X не является ключом».

Это, по сути, формальная запись теоремы о шунтировании, только в терминах ограничений на ключ: когда составной ключ X многозначно («проективно-соединительно», декартово) определяет неключевой

атрибут (не обязательно простой), то такое отношение будет иметь 5НФ $(X_1 + X_2 + ... + X_n) \to\to (A_1 + A_2 + ... + A_j)$. Совокупность именно таких схем позволяет минимизировать операции соединения.

3.9. Каркасный анализ предметной области

Будем называть разделение сущностей-объектов по категориям *сепарацией* сущностей-объектов. Алгоритм автоматизированной сепарации построен в соответствии с теорией КМД [207, 208]. Задача решается тем, что на первом этапе осуществляется автоматизированная этимологическая сепарация данных, а на втором этапе, в соответствии с результатами этимологической сепарации, осуществляется автоматизированное каркасное размещение данных в БД [212].

При этом используется допущение, что для любой ПрО всегда существует ограниченная *базовая совокупность* сущностей-объектов, к которой относятся лишь атомарные и слабые сущности-объекты [300]. А все иные сущности-объекты (которых практически всегда - намного больше) синтезируются на этой совокупности благодаря каркасу связей, т.е. булеану связей сущностей-объектов из базовой совокупности. Остальные сущности-объекты являются следствием функционирования ПрО.

Задача автоматизированной сепарации сущностей-объектов не является тривиальной. И вопрос строгой формализации описанного ниже алгоритма – это вопрос дальнейших исследований. В настоящей работе приведено лишь его нестрогое описание.

Отметим, что именно *составные* сущности-объекты в ПрО более всего замаскированы. И именно они обладают наиболее противоречивым происхождением смысла. Поэтому их сепарация – это и есть основная цель каркасного проектирования схемы БД.

Таким образом, целью автоматизированной сепарации сущностей-объектов моделируемой ПрО является получение каркасной совокупности отношений $R_l(X_i, A_t)$. Тут X_i - это конкатенированные в общем случае

ключи каждого отношения, а A_i – атрибуты сущностей-объектов или их связей, $l=1,L$ – общее число отношений в схеме БД, $i=1,N$, N - число атомарних сущностей-объектов. Каждое отношение соответствует той или иной атомарной, слабой или составной сущности-объекту (т.е. многоарной в общем случае связи). Причем, синтезировав суррогатные ключи (и их структуру) в каждом отношении, а также установив однозначное соответствие между именами сущностей-объектов и именами отношений, пользователь получает возможность в динамическом режиме пополнять каждое отношение атрибутами (т.е., свойствами) каждой сущности-объекта.

При этом все необходимые документы ПрО моделируются как проекция фрагмента БД на носители (монитор, принтер, плоттер и т.п.). Неопределенные сущности-объекты на момент внедрения приложения исключены.

В работах [2, 294] также предложен алгоритм разделения сущностей-объектов моделируемой ПрО. Но лишь на две категории – на сущности-объекты в статическом состоянии и на их связи. Однако такое упрощение приводит схему БД к жесткости и немодифицируемости, не смотря на декларации авторов [294].

3.9.1. Классификация сущностей-объектов ПрО

В соответствии со схемой РК введем формализацию ПрО. Пусть в описываемом алгоритме все сущности-объекты распределяются на пять категорий. Дадим определения этим категориям.

1. *Атомарные* сущности-объекты – сущности-объекты, значения атрибутов которых не изменяются во времени, или этими изменениями в моделируемой ПрО можно пренебречь. Эти сущности-объекты не имеют также и зависимости существования [300]. В некоторых моделях данных такие сущности-объекты получили название стержневых или базовых [362].

Формально *атомарная* сущность-объект – семантически атомарный многоместный предикат [231] $I_j^M(A_j^m)$, где A_j^m – аргументы предиката -

совокупность атрибутов j-й сущности-объекта, где $j = 1, L$ – номер сущности-объекта в ПрО, а $m = 1, M_j$ номер места в M_j – местном предикате.

Семантически атомарный предикат и ключ семантически атомарного предиката $\alpha_j(x_j)$ определены в разделе 2 (определения 2.5. и 2.6.). Тогда семантически атомарный предикат может быть представлен в виде конъюнкции: $I_j^{M_j}(A_j^m) = \alpha_j(x_j) \bigcap F_j^{M_j}(A_j^m)$.

А вся N-совокупность P_I атомарных сущностей-объектов – так называемая «статическая» часть ПрО (по сути, структура системы P) – это предикат, полученный дизъюнкцией семантически атомарных многоместных предикатов: $P_I = \bigcup I_j^{M_j}(A_j^m) = \bigcup(\alpha_j(x_j) \bigcap F_j^{M_j}(A_j^m))$, $j = 1, N$.

Примером атомарных сущностей-объектов может быть сущность-объект *ЧЕЛОВЕК, ВСЕЛЕННАЯ, СОБАКА, КОШКА* и т.п. Причем принадлежность этих сущностей-объектов к определенным дальнейшим категориям (абстрагирование - классификация атомарных сущностей-объектов) является искусственной семантической надстройкой пользователя, которая и маскирует содержание сущности-объекта.

2. «*Слабые»* сущности-объекты – сущности-объекты, которые функционально зависят от атомарных [300]. И в моделях данных имеют аналогичное название. Причем такая зависимость может быть как лишь на уровне идентификации слабых атрибутов, так и на уровне всего существования зависимых слабых сущностей-объектов.

Определение 3.4. *Слабый ключевой предикат* (или *слабый ключ*) – предикат, который получен конъюнкцией ключевых предикатов некоторых семантически атомарных многоместных предикатов так, что существует строгая ФЗ между любыми ключевыми предикатами от первого до завершающего ключевого предиката.

Схематически слабый ключ имеет вид направленной связи от ключей-«предков» к ключам-«наследникам»: $\alpha = \alpha_1(x_1) \bigcap \alpha_2(x_2) \bigcap \alpha_3(x_3) \ldots \alpha_k(x_k)$, где

существует строгая ФЗ: $\alpha_k(x_k) \to (\alpha_{k-1}(x_{k-1}) \to (\ldots \alpha_3(x_3) \to (\alpha_2(x_2) \to \alpha_1(x_1))))$.

Определение 3.5. *Семантически слабый многоместный* предикат (или просто *слабый предикат*) – предикат, который получен конъюнкцией некоторого семантически атомарного предиката и слабого ключевого предиката.

Схематически семантически слабый многоместный предикат имеет вид: $Q_i^{(k+j)}(x_1, x_2, \ldots, x_k, b_1, b_2, \ldots, b_j) = \alpha_1(x_1) \bigcap \alpha_2(x_2) \ldots \alpha_k(x_k) \bigcap D_i^J(b_j)$, где k – число звеньев иерархии ключа, $D_i^J(b_j)$ – i-й семантически атомарный многоместный предикат, $j=1,J_i$ – число мест для собственных атрибутов предиката b_j и существует строгая ФЗ: $\alpha_k(x_k) \to (\alpha_{k-1}(x_{k-1}) \to (\ldots \alpha_3(x_3) \to (\alpha_2(x_2) \to \alpha_1(x_1))))$.

Тогда формально *слабая* сущность-объект – это семантически слабый многоместный предикат, обладающий собственными аргументами.

В начальном приближении модели ПрО слабые сущности-объекты будем относить к атомарным, так как их атрибуты также не зависят от времени.

Примером слабых, но, тем не менее, относимых к категории атомарных, могут быть сущности-объекты *ПОДРАЗДЕЛЕНИЕ, ОТДЕЛ, ЛАБОРАТОРИЯ, КВАРТИРА, КАФЕДРА* . Каждая из этих сущностей-объектов не является самодостаточной и функционально зависит от «предковой» сущности-объекта. Однако зависимостью ее атрибутов от времени в ПрО можно пренебречь.

Проектировщики часто делают исключение - некоторым слабым сущностям-объектам принудительно назначается суррогатный идентификатор, который уникально идентифицирует все атрибуты. Такие исключения являются своеобразной границей ПрО, когда пользователю известно, что на протяжении значительного времени эксплуатации приложения и БД (ХД), граница не будет расширяться. Однако именно такие исключения приводят к невозможности осуществлять модификации схемы

БД без изменений самого приложения. Как в процессе ее работы, так и после ее останова.

3. *Составные* («постсвязные») сущности-объекты – в моделях данных имеют еще и название многосторонних связей [300, 359] – связь атомарных сущностей-объектов.

Определение 3.6. *Составной ключевой предикат* (или *составной ключ*) – это предикат, который получен конъюнкцией ключевых предикатов от некоторых семантически атомарных многоместных предикатов.

Это означает, что наличие ФЗ в произвольном ключевом звене не является обязательным условием существования такого предиката.

Схема составного ключевого предиката имеет вид, аналогичный схеме из предыдущего пункта: $\alpha = \alpha_1(x_1) \bigcap \alpha_2(x_2) \bigcap \alpha_3(x_3) \ldots \alpha_k(x_k)$, но строгая ФЗ в виде: $\alpha_k(x_k) \to (\alpha_{k-1}(x_{k-1}) \to (\ldots \alpha_3(x_3) \to (\alpha_2(x_2) \to \alpha_1(x_1))))$ отсутствует.

Определение 3.7. *Семантически составной многоместный предикат* (или *составной предикат*) – это предикат, полученный конъюнкцией составного ключевого предиката и семантически атомарного многоместного предиката.

Как указано выше, это означает, что наличие ФЗ в произвольном ключевом звене не является обязательным условием существования такого предиката. При этом семантически составной многоместный предикат также имеет свои аргументы – аргументы связи.

Аналогично предыдущему пункту, схема составного предиката имеет вид: $S_i^{(k+j)}(x_1, x_2, \ldots, x_k, b_1, b_2, \ldots, b_j) = \alpha_1(x_1) \bigcap \alpha_2(x_2) \bigcap \alpha_3(x_3) \ldots \alpha_k(x_k) \bigcap T_i^J(b_j)$, где k – число звеньев ключа, $T_i^J(b_j)$ – i-й семантически атомарный многоместный предикат, $j = 1, J_i$ – число мест для собственных атрибутов b_j и в каждом звене отсутствует строгая ФЗ. Заметим, однако, что это не означает, что ФЗ не может существовать в некоторых звеньях.

Поэтому формально *составная сущность-объект* – это семантически составной многоместный предикат, обладающий собственными аргументами.

Одним из примеров составных сущностей-объектов являются событийные сущности-объекты – *ЭКЗАМЕН, КОНЦЕРТ, ВЫСТАВКА, СОГЛАШЕНИЕ, МИТИНГ* и т.п. Их содержание представляет собой «продукт» равноправного взаимодействия нескольких атомарных сущностей-объектов.

В [71] описаны объекты, полученные агрегацией данных. Там сущность-объект *ЭКЗАМЕН* так же, как и в [205], отнесена к категории агрегированных сущностей-объектов, т.е., составных. «Например, связь между сущностями *СТУДЕНТ, ДИСЦИПЛИНА, ПРЕПОДАВАТЕЛЬ, ОЦЕНКА* имеет смысловое описание: «студент по фамилии __ получил на экзамене по дисциплине __ у преподавателя по фамилии __ оценку __ » и может быть представлена агрегированным элементом: сущностью *ЭКЗАМЕН* с атрибутами *ФИО_СТУДЕНТА*, *НАЗВАНИЕ_ДИСЦИПЛИНЫ*, *ФИО_ПРЕПОДАВАТЕЛЯ*, *КОД_ОЦЕНКИ*» [71].

Очень схожий пример рассмотрен и в [89]. Там также делается вывод о том, что агрегат необходимо представлять конкатенацией атрибутов составляющих его сущностей-объектов. Хотя дальнейшего развития эта идея в работах автора [89] не нашла.

Таким образом, в каркасном алгоритме сущности-объекты формируются по следующей схеме: на базе атомарных («стержневых», «базовых» и т.п.) порождаются слабые, т.е. функционально (иерархически) зависимые от атомарных, но статичные во времени. Отношения, моделирующие слабые сущности-объекты, также имеют составной ключ. А на совокупности атомарных и слабых сущностей-объектов благодаря образованию разнообразных связей между ними создаются составные постсвязные сущности-объекты, характеризующие поведение ПрО.

Отметим, что на начальном этапе моделируются статические срезы ПрО, а на последующих учитывается и темпоральность данных.

Чаще всего подавляющее большинство составных сущностей-объектов проектировщиками относится к категории слабых, или даже атомарных, что,

в свою очередь, приводит к увеличенной жесткости приложений БД и невозможности их гибкого развития без коренных переработок текстов программ.

КМД предоставляет проектировщику совокупность отношений, каждое из которых моделирует либо статическое состояние сущности-объекта (тривиальную связь), либо одну из связей совокупности сущностей-объектов.

Любой семантически атомарный многоместный предикат рассматривается лишь как связь его атрибутов. Т.е., как тип тривиальной связи «самого с собой»: $I_j^{M_j}(A_j^m)$.

А нетривиальные связи рассматриваются как конъюнкция ключевых атрибутов от разных семантически атомарных многоместных предикатов. Тогда схематически фрагмент булеана связей имеет вид (с несколько более усложненными индексами):

$S_i^{(l_j+j)}(x_1,x_2,x_3,\ldots,x_k,b_1^j,b_2^j,\ldots,b_{l_j}^j)=\alpha_1(x_1)\cap\alpha_2(x_2)\cap\alpha_3(x_3)\ldots\alpha_j(x_j)\cap T_j^{l_j}(b_{l_j}^j)$, где $b_{l_j}^j$ – атрибуты связи, моделируемые $S_i^{(l_j+j)}$-м предикатом, обладающим (l_j+j) местами, j из которых занимают ключи с 1-го по j-й, а с $(j+1)$-го по (l_j+j)-й занимают аргументы связи – шунтирующие атрибуты соответствующего отношения $R_j(X_1,X_2,X_3,\ldots,X_j,B_1^j,B_2^j,\ldots,B_{l_j}^j)$.

Каркасными предикатами будем называть произвольную совокупность семантически атомарных, слабых или составных многоместных предикатов с произвольным составом.

Тогда схематически в общем виде модель ПрО можно представить так:

$P_I=I_1^{M_1}(A_1^m)\cup I_2^{M_2}(A_2^m)\ldots\cup I_N^{M_n}(A_N^m)$ – структура ПрО («бесконечно долгие» по времени связи ПрО),

$P_S=S_2^2\cup S_2^3\cup S_2^4\ldots\cup S_2^{L_2}\cup S_3^2\cup S_3^3\cup S_3^4\ldots\cup S_3^{L_3}\ldots\cup S_N^2\ldots\cup S_N^{L_n}$ – текущие связи ПрО.

Тогда: $P=P_I\cup P_S$

Т.е., ПрО как многоместный предикат P является результатом дизъюнкция конъюнкций каркасных предикатов.

С одной стороны, модель ПрО является актуальной частью РК. А с другой многоместный предикат P – это классическая нормальная форма высказывания в булевой логике, широко используемая в исследованиях в смежных областях [335].

Описанные типы сущностей-объектов, так или иначе, встречались в других концептуальных моделях. Однако, как показывает практика проектирования приложений, их не достаточно. Для более точного моделирования ПрО необходимо использование еще двух типов – артефакты и неопределенные сущности-объекты.

4. *Артефакты* - это сущности-копии, данные о которых будут условно размещаться в БД по решению пользователя.

Формально артефакт (документ) – это предикат $E_i(A_j, B_l, C_k, D_m, \ldots, Z_p)$, где каждый из аргументов $A_j, B_l, C_k, D_m, \ldots, Z_p$ – это фрагмент множества аргументов «порождающих» предикатов, участвующих в формировании предиката E_i. Т.е., это частичная копия многоместного предиката, моделирующего связь других сущностей-объектов. Имеет некоторое сходство с составной сущностью-объектом.

К артефактам, например, можно отнести любой документ, который пользователи ПрО создают именно ради того, чтобы скопировать те или иные атрибуты тех или иных сущностей-объектов. Причем не просто скопировать атрибуты одной конкретной сущности-объекта, а еще и объединить в этой новой искусственно созданной сущности-объекте несколько атрибутов от разных сущностей-объектов.

Артефакты – это, как правило «пост-следственные» сущности-объекты. Поэтому независимо регистрируя их в приложении, пользователь сталкивается со значительным дублированием данных. А это, в свою очередь, приводит к потребности дополнительного отслеживания целостности еще и избыточных данных. Проблема заключается в том, что в ПрО вероятнее всего уже существует группа неискусственных составных

сущностей-объектов, у которых имеется еще и множество атрибутов, которые не копируются в атрибуты совокупности искусственных сущностей-объектов.

Отслеживание целостности таких дублированных данных в этой совокупности естественных и искусственных сущностей-объектов очень затруднено. КМД позволяет исключить такие сущности-копии из схемы БД. А копии моделировать посредством проекций частей схемы БД при ответах на запросы. Однако если в ПрО отсутствуют дублирующие сущности-объекты, проектировщик может использовать составную сущность-объект под именем артефакта.

Примером артефактов могут быть *НАКЛАДНАЯ*, *СЧЕТ*, *АКТ* и т.д.

5. *Неопределенные* сущности-объекты – такие, семантика которых подлежит дальнейшему уточнению; или такие, на которые не ссылается ни одна из всей совокупности сущностей-объектов моделируемой ПрО.

Формально к неопределенным сущностям-объектам первого типа отнесем предикат $E_i(A_j,B_l,C_k,D_m,\ldots,Z_p)$, совокупность $A_j,B_l,C_k,D_m,\ldots,Z_p$ аргументов которого – пустое множество. К неопределенным сущностям-объектам второго типа отнесем предикат $E_i(A_j,B_l,C_k,D_m,\ldots,Z_p)$, никакая часть совокупности $A_j,B_l,C_k,D_m,\ldots,Z_p$ аргументов которого не встречается ни в одной связи других сущностей-объектов этой ПрО.

Неопределенные сущности-объекты возникают в описании ПрО на этапе незавершенного исследования функционирования ПрО.

Отметим, что впервые такая классификация, а также алгоритм сепарации, основанный на ней, были предложены в [206].

3.9.2. Текстовое описание ПрО

В работе подразумевается, что описание моделируемой ПрО должно быть выражено типизированной формой на естественном языке. Единицей считывания при этом является атомарное предложение (в дальнейшем - просто «предложение»), содержащее не более, чем две сущности-объекта,

которые кодируются существительными с уникальным побуквенным написанием. А глагол с уникальным побуквенным написанием символизирует исключительно бинарную связь между парой сущностей-объектов этого же предложения.

Предполагается, что существительные, которые повторяются, обозначают одну и ту же сущность-объект. Поэтому такое повторение в пределах одного предложения означает тривиальную пару, т.е. такую, которая несет лишь информацию о существовании сущности-объекта в ПрО без связей ее с другими. Предполагается также, что и глаголы, которые повторяются в разных предложениях, означают один и тот же класс связи. Поэтому основная миссия атомарного предложения - информировать о наличии сущностей-объектов в определенной ПрО и декларировать класс связи этой пары.

Предложения, которые включают в себя более чем две сущности-объекта, являются составными. Они подлежат автоматизированной декомпозиции. Для этого может использоваться любой известный алгоритм декомпозиции составных предложений. Однако, те составные предложения, которые невозможно автоматизировано преобразовать к бинарной форме по технологической причине (например, из-за отсутствия четкой структуры, которая объединяет их в одно составное предложение), из описания изымаются. Этот фрагмент описания подлежит дальнейшему уточнению.

Алгоритм не предусматривает верхнего ограничения количества предложений. Тем не менее, предполагается предварительный анализ наличия для каждой задекларированной сущности-объекта хотя бы одной связи с какой-либо иной сущностью-объектом – принцип замкнутости описания.

Таким образом, проектировщик схемы БД может получить описание ПрО в следующем виде:

- текстового файла, сформированного текстом на естественном языке,
- последовательных схем или графов,

- файла с записанным голосовым сигналом, который надиктован естественным языком,

- звукового голосового сигнала в реальном времени,

- последовательности файлов хранилищ данных, которые уже существуют и введены в эксплуатацию и т.п.

Файлы внедренных БД особенно важны для исследования возможных противоречий в схемах БД и прогнозирования затрат на модификации при дальнейшем развитии приложения. Очевидно, что каждый тип описания может использоваться не обособлено, а в сравнении один с другим.

3.9.3. Этимология смысла сущности-объекта

Для автоматизированного синтеза схемы БД в каждом проектируемом отношении $R_i(X_{k_i}^{m_k}, A_j)$ формируются *структурированные идентификаторы***,** структура каждого из которых не произвольная, не задана пользователем и не получена каким-то иным отличным способом, а строго исчисляется.

Для автоматизированного исчисления структуры идентификатора используются математические критерии, построенные в соответствии с закономерностями, выявленными в ПрО.

В основе этих критериев - единый обобщенный фактор происхождения смысла сущности-объекта, т.е. *этимология смысла* (в дальнейшем – просто этимология) сущности-объекта.

Определение 3.8. *Этимологией* сущности-объекта будем называть строковый структурированный идентификатор со следующей схемой: $X_1^{m_1} + X_2^{m_2} + X_3^{m_3} + ... + X_{k_i}^{m_k}$, где каждое звено $X_{k_i}^{m_k}$ - отделенный идентификатор факта происхождения *i*-го каркасного предиката, *i* – номер произвольного каркасного предиката, k_i – номер звена идентификатора *i*-го каркасного предиката (подстрочный индекс), m_k – номер соответствующего порождающего каркасного предиката из базовой совокупности каркасных предикатов - объединенной группы атомарных и слабых предикатов

(надстрочный индекс), причем каждое m_k может получить какое либо значение только из множества $\{1,2,...,N_0,...,N\}$, где N_0 - общее число атомарных предикатов, N – суммарное число атомарных и слабых предикатов.

Заметим, что в случае полной совокупности связей $i=\{1,2,...,N_0,...,N,(N+1),...,(2^N-1)\}$. Для атомарных предикатов этимологией является лишь одно звено X^i, в котором $m=i$. Т.е. атомарная сущность-объект порождает сама себя.

В описываемом алгоритме атомарные сущности-объекты получают в общей совокупности первые номера, т.е. для них $i=1,N_0$. Для слабых сущностей этимологией является вышеупомянутая строковая сумма звеньев, где каждому номеру k_i звено $X_{k_i}^{m_k}$ соответствует строго.

Для составных сущностей-объектов этимологией является вышеупомянутая строковая сумма звеньев, где место каждого звена $X_{k_i}^{m_k}$ не строгое, т.е. последовательность звеньев не имеет значения.

Каждое звено этимологии сущности-объекта означает связь с иными атомарными сущностями-объектами, которые принимали участие в происхождении данной сущности-объекта, если последняя представляет собой или слабую, или составную, т.е. постсвязную сущность-объект.

Таким образом, каждое звено $X_{k_i}^{m_k}$ идентификатора ячейки строится в строгом соответствии с происхождением сущностей-объектов из описания ПрО. Каждая сущность-объект в ПрО может быть атомарной, а значит иметь *унарный* X^i, или *составной* идентификатор $\sum X_{k_i}^{m_k}$, где суммирование ведется по k_i, $k_i=1,K_i$. Причем общее число звеньев K_i представляет собой арность отношения, моделирующего такую сущность-объект. При этом K_i в общем случае может равняться 2, 3,..., 10 и т.д. А в случае атомарной сущности-объекта $K_i=1$.

На основании значительного числа экспериментальных исследований, проведенных на разных ПрО [113, 241], сформулирована

Гипотеза 3.1. Все факторы, характеризующие семантику любой сущности-объекта в ПрО, являются функционально зависимыми от этимологии сущности-объекта.

3.10. Автоматизированная сепарация сущностей-объектов

3.10.1. Логическая сепарация сущностей-объектов

Гипотеза 3.2. Алгоритм последовательных приближений сепарации каркасных предикатов в виде численного определения этимологии каждого имеет на РК единственное решение при условии, что эта совокупность является замкнутой.

Такое допущение полностью согласуется с основными теоремами КМД. Начальное приближение алгоритма синтеза каркаса-шаблона может быть получено с использованием специализированного словаря. Назовем его *словарь вероятных этимологий*. Анализ основан на сравнении имени и контекста (совокупности сущностей-объектов в описании ПрО) каждой сущности-объекта с задекларированными в этом словаре сущностями-отношениями.

В словаре каждому существительному и глаголу заранее поставлено в соответствие наиболее вероятную этимологию. Причем те слова, которые обозначают пока что неизвестные словарю сущности-объекты и классы связей, для дальнейшего автоматизированного анализа отделяются. А в случае, если неизвестных сущностей-объектов и связей в описании не выявлено, автоматизированный логический анализ завершается.

Формально пусть имеется описание ПрО в виде множества сущностей-объектов в предикатной форме $E_l(A_j, B_l, C_k, D_m, \ldots, Z_p)$. Задача: для каждой l-й сущности-объекта численно определить, к какому типу она относится с учетом всей совокупности E_l, а после этого определить, какие атрибуты действительно принадлежат E_l, какие являются внешними и символизируют связь, а какие попали в совокупность по ошибке.

На следующем шаге из описания отделяются неизвестные словарю потенциальные *составные* сущности-объекты благодаря автоматизированному логическому сравнению каждой из неизвестных сущностей-объектов с теми, что образовываются из повторяющихся существительных и повторяющихся глаголов благодаря объединению их в одну составную, т.е. многостороннюю постсвязную сущность-объект.

Такое объединение возможно при условии совпадения класса связи, т.е. совпадения глаголов между различными парами, так как именно благодаря многоразовой повторяемости упомянутых существительных в нескольких различных связях с одного класса, т.е. для нескольких одинаковых глаголов, вероятность того, что эти сущности-объекты принадлежат именно к группе составных сущностей-объектов, значительно повышается.

Если же выяснится, что такое приближение - ошибочное, это не внесет значительной некорректности. Схема будет уточняться в дальнейшем.

На дальнейших шагах осуществляется последовательное или параллельное выполнение процедуры сравнения каждой сущности-объекта из каждого предложения, т.е., из каждой пары, с каждой другой сущностью-объектом.

Результат - это *логическая сепарация*, т.е., предоставление каждому отношению-проекту, где будут храниться данные от атрибутов каждой сущности-объекта моделируемой ПрО, соответствующих предварительных структурированных идентификаторов. И тем самым предварительная перегруппировка этих отношений по категориям.

Особенностью данного этапа является то, что синтез структуры каждого звена этимологии сущностей-объектов осуществляется автоматизированным логическим анализом существительных и глаголов, т.е. анализом содержания сущностей-объектов и содержания связей, без учета множеств конкретных значений конкретных атрибутов сущностей-объектов.

В этой части алгоритма завершается автоматизированный логический анализ описания ПрО. Последнее логическое сравнение - анализ группы тех сущностей-объектов и связей, которые оказались неизвестными словарю вероятных этимологий и остались после изъятия потенциально составных сущностей-объектов.

3.10.2. Статистическая сепарация сущностей-объектов

На следующем шаге от предварительно отобранных групп сущностей-объектов окончательно отделяются артефакты, т.е. сущности-копии, которые могли по ошибке остаться в этой совокупности. Для этого осуществляется автоматизированное статистическое сравнение, основанное на использовании известных [3] процедур статистического анализа для выявления детерминированных ФЗ, а также корреляционных или регрессивных зависимостей между значениями данных в атрибутах сущностей-объектов. Наличие или отсутствие таких зависимостей позволяет подтвердить или опровергнуть прямые совпадения групп атрибутов, а также замаскированную этимологию, полученную на предыдущих шагах.

Как свидетельствуют определенные исследования, для отслеживания наличия, например, прямых совпадений атрибутов-копий (в случае несовпадения имен) достаточно сравнить не больше десяти групп значений, т.е., не больше десяти групп кортежей для реляционного формата хранения значений атрибутов сущностей-объектов.

Для отслеживания закономерности на этом шаге алгоритма от каждой сущности-объекта достаточно не более двух естественных атрибутов. А для отслеживания, например, МЗ (ЗПС), которая наблюдается лишь между атрибутами составных сущностей-объектов и отдельно атрибутами каждого из их предков, которые принимали участие в образующих связях этих

составных сущностей-объектов, достаточно сравнить не более двухсот групп значений. Т.е., не более двухсот групп кортежей для реляционного формата хранения значений атрибутов сущностей-объектов.

Причем между каждым конкатенированным значением экземпляров общей совокупности всех отделенных атрибутов предков и значениями экземпляров какого либо (или даже каждого) из атрибутов составных сущностей-объектов возникает уже не МЗ, а детерминированная ФЗ, если именно эти предки образовывали именно эту составную сущность-объект.

Наличие такой детерминированной связи является достаточным критерием для идентификации и сепарации составных сущностей-объектов. Причем для отслеживания этой закономерности от каждой сущности-объекта также достаточно не больше двух естественных атрибутов.

3.10.3. Распределение сущностей-объектов на каркасе

Для корректности статистического анализа вся совокупность значений всех атрибутов от всех сущностей-объектов ПрО должны отвечать единому моменту *времени* жизни ПрО. Причем отрезок между соседними промежутками времени должен быть достаточным для возникновения действительно нового состояния ПрО. Если это условие не выполняется, закономерности будут некорректными.

На следующем шаге строится *уточненное* приближение сепарации, для чего отделяются группы значений атрибутов, которые зависят от *времени*, и группы значений атрибутов, которые от времени не зависят. Или, если и зависят, то лишь от очень значительных промежутков времени – их развитием и изменениями можно пренебречь в сравнении с другими группами значений атрибутов.

Причем, группа атрибутов, которая практически не зависит от времени, относится к группе сущностей-объектов, которые создают *структуру* ПрО. Структура какой либо системы значительно медленнее зависит от времени, чем ее функционирование, т.е. формирование определённых связей между сущностями-объектами. Таким образом, на этом шаге за уточненное приближение *составных* сущностей-объектов берется группа сущностей-объектов, которые зависят от времени. А другая группа получает статус совокупности атомарных и слабых.

На следующем шаге в группе, где отобраны атомарные и слабые сущности-объекты, повторно и более доказательно автоматизировано отделяются атомарные от слабых. Одновременно используется два критерия.

Первый критерий заключается в том, что для идентификации какого либо значения естественного атрибута атомарной сущности-объекта достаточно лишь названия сущности-объекта и названия атрибута, что невозможно именно в случае слабой сущности. Но такое сопоставление на этом шаге осуществляется на большом числе данных**.**

Второй критерий заключается в том, что между атрибутами потомка и конкатенированными атрибутами всех предков наблюдается ФЗ, а потому - детерминированная связь, которая дает возможность отслеживать не только сам факт слабости, а еще и конкретизировать звенья связей с более «старшими» сущностями-объектами. Причем, если связь от потомка к предку устанавливается однозначно, проверка наличия или отсутствия однозначной обратной связи от предка к множеству потомков возможна лишь благодаря интерполяции значений от атрибутов всех потомков следующего уровня. Т.е., преобразования множества этих значений в математическую функцию и проверки детерминированной зависимости на отрезке в окрестности

значений атрибута конкретного потомка (аналогично отслеживанию детерминированной связи, например, у периодической функции). Подтвержденная связь находит отображение в структуре идентификатора ячейки сущности-объекта.

Для дальнейшего уточнения не только характера и принадлежности к группе составных сущностей-объектов, а еще и окончательного восстановления структуры этимологии каждой составной сущности-объекта, на базе полученной совокупности атомарных и слабых сущностей-объектов в компьютерной памяти в качестве шаблона [212] строится каркас полной совокупности связей данных.

3.10.4. Результирующий алгоритм

В общем виде алгоритм автоматизированного каркасного проектирования схемы БД сводится к следующим шагам метода последовательных приближений, когда на каждом следующем шаге благодаря определенным логическим и математическим критериям осуществляется уточнение каждой предыдущей совокупности данных.

Шаг 1. Формирование взаимно-однозначного соответствия между каждым словом из описания ПрО, обозначающим сущность-объект, и группой слов, каждое из которых обозначает атрибут каждой сущности-объекта из ПрО. Эта часть алгоритма проводится пользователем вручную. На этом же этапе пользователь исключает артефакты и неопределенные сущности-объекты. Данный шаг играет роль ручной подготовки исходных данных.

Результат – список вида $E_l(A_j,B_l,\ldots,Z_p)$, где E_l – совокупность многоместных предикатов, соответствующих списку слов, обозначающих сущности-объекты из ПрО, каждое из которых встречается единственный

раз. Тут $A_j, B_l, \ldots, Z_p$ – совокупность слов, обозначающих атрибуты каждой сущности-объекта, $l = 1, L$ – число всех сущностей-объектов, $i = 1, I_l;\ j = 1, J_l;\ p = 1, P_l$ – значения текущего числа атрибутов каждой сущности-объекта.

Шаг 2. Автоматическое или автоматизированное изъятие начального приближения базовой совокупности сущностей-объектов. В описании ПрО эта группа сущностей-объектов, как правило, замаскирована разнообразными агрегированными терминами, категориями, вспомогательными существительными, синонимами и т.д.

Для этого к отобранной группе атомарных сущностей-объектов присоединяется еще и подгруппа виртуально атомарных сущностей-объектов, которая получается добавлением к идентификаторам слабых сущностей-объектов отдельного унарного идентификатора. Создается начальное множество простых унарных идентификаторов.

Это действие носит сугубо технологический характер и упрощает дальнейшие шаги относительно создания комбинаций идентификаторов ячеек: назначенные виртуально атомарные сущности-объекты, которые происходят от слабых, несут в себе обе этимологии - естественную, т.е. составную, и искусственную, т.е. унарную. Это действие коренным образом отличается от процедуры автоматического назначения унарного идентификатора без учета семантики любому объекту, что свойственно, например, объектно-ориентированной модели [38, 39, 445].

Шаг 3. Последовательно или параллельно сравниваются слова из описания моделируемой ПрО, обозначающие сущности-объекты и их связи, с теми, которые уже проанализированы в других проектах. На этом шаге используется пополняемый словарь. Тут же базовая совокупность

окончательно отделяется от случайно неотобранных пользователем артефактов и неопределенных сущностей-объектов.

Шаг 4. Если словарь для анализируемых сущностей-объектов или связей пуст, осуществляется процедура автоматизированного логического сравнения сущностей-объектов между собой. Число логических процедур и критериев для сравнений ничем не ограничено – они отделены во внешнюю библиотеку, которая пополняется.

Шаг 5. Для каждого унарного идентификатора каждой сущности-объекта из базовой совокупности в хранилище отводится унарный домен памяти для размещения элементов хранения идентификатора, структура которого строго унарная.

Таким образом, в памяти создается начальное множество простых унарных доменов. При этом идентификаторы от слабых сущностей-объектов могут быть обозначенными дополнительно. Тем не менее, способ установки подобных меток может быть произвольным, вплоть до их отсутствия.

Шаг 6. На базовой совокупности унарных доменов осуществляется синтез каркаса-шаблона - каркасных эталонных составных сущностей-объектов - построение булеана связей по принципу «все со всеми». Этой процедурой порождается система доменов с многоарными идентификаторами.

Шаг 7. Осуществляется наполнение отношений шаблона конкретными данными из ПрО. Причем группы данных отбираются и вносятся таким образом, чтобы каждая группа строго соответствовала новому состоянию ПрО.

Шаг 8. Проводится начальная сепарация составных сущностей-объектов благодаря процедурам статистического сравнения эталонных

составных сущностей-объектов, полученных на каркасе-шаблоне, и тех составных сущностей-объектов, которые на завершающем этапе отделены.

Благодаря процедурам статистического анализа с использованием конкретных значений данных осуществляется проверка групп атрибутов атомарных, составных и слабых сущностей-объектов из описания ПрО, а также сформированных атомарных и составных идентификаторов на соответствие друг другу.

Шаг 9. Осуществляется уточнение сепарации составных сущностей-объектов посредством процедуры последовательных приближений на каркасе-шаблоне.

Уточняется соответствие путем применения повторной процедуры последовательных приближений и многоразовой модификации базовой совокупности, то есть соответствующего каркаса-шаблона.

Шаг 10. Формируются рекомендации словарю о возможности пополнения его ресурсов новыми группами сущностей-объектов, если в окончательных группах никаких противоречий не выявлено.

В итоге алгоритм формирует этимологии всех сущностей-объектов из начального описания ПрО. Очевидно, что в зависимости от специфики конкретной ПрО некоторые шаги алгоритма могут не использоваться.

3.10.5. Внешняя библиотека критериев перераспределения

Для КМД доказаны теоремы о полноте и единственности реляционного каркаса, а также о его непротиворечивом росте [232]. Важным следствием этих теорем является следующая

Лемма 3.4. В замкнутой каркасной совокупности неключевые атрибуты A_j какой-либо актуальной ячейки каркаса со схемой $R(X_i, A_j)$ встречаются в каркасном отношении единственный раз.

Действительно, если для какой либо совокупности составных сущностей-объектов $R(X_i, A_j)$ искусственно назначить статус атомарных с искусственными унарным идентификатором $Y = X_i$, а затем вновь перемножить их, то образованные новые (искусственные) составные сущности-объекты (по сути – связи связей) можно получить и на продолжении каркаса $R(X_i, X_i + 1, B_j)$. Но при условии, что при новом перемножении продублированные идентификаторы из отношений исключаются, что соответствует реляционной модели (и здравому смыслу), отношения станут тождественными. □

Это означает, что составные сущности-объекты со схемой $R(X_i, A_j)$ между собой дальнейших связей не образовывают и сущностей-объектов не порождают. Т.е., и без переобозначения идентификаторов базовая совокупность сущностей-объектов - это еще и базовая совокупность идентификаторов. При таком ограничении синтезированные составные сущности-объекты не расширяют базовой совокупности.

Тем не менее, какое либо расширение базовой совокупности сущностей-объектов приводит к появлению новых составных сущностей-объектов. Поэтому, если все же возникает такая потребность, алгоритм позволяет искусственно моделировать дальнейшие связи именно расширением базовой совокупности идентификаторов. Например, прибавляя к начальной совокупности еще и искусственные атомарные сущности-объекты, полученные из составных путем установки в их структуре искусственных унарных идентификаторов.

Такая ситуация может возникнуть при условии, что для некоторых ПрО характерным является расширение их структуры за счет синтезированных составных сущностей-объектов. В этой ситуации важно

обязательное многократное добавление идентификаторов, отвечающих за различные состояния составных сущностей-объектов или их масок. А также учет номеров отрезков времени таких модификаций в этих идентификаторах, что в рамках настоящей работы не рассматривается.

Алгоритм предусматривает возможность развития процедур логического и статистического анализа. Для этого отдельно строится внешняя библиотека, которая пополняется новыми подчиненными способами как логического, так и статистического анализа со своими новыми критериями, которые разрабатываются пользователями. Поэтому перечень подчиненных способов сравнения данных между собой, а также перечень критериев сравнения ничем не ограничивается.

Не ограничивается также и последовательность выполнения упомянутых процедур. Очевидно, что наиболее точная сепарация может быть проведена либо благодаря словарю вероятных этимологий, либо благодаря автоматизированному статистическому анализу на каркасе-шаблоне.

Первый тип сепарации еще и самый быстрый, последний – самый длительный. Поэтому, при отсутствии сущностей-объектов в словаре, выполнение всех иных, т.е., промежуточных итераций, значительно ускоряют каркасную сепарацию. И позволяет всесторонне проанализировать данные. Если словарь вероятных этимологий на начальных стадиях своего существования не является полным, постоянная эксплуатация, которая пополняет его, в конечном итоге минимизирует потребность в автоматизированном логическом и статистическом анализе описания ПрО.

Этот механизм позволит вносить изменения в схему такой БД по полно-модифицируемому принципу, а не с существенным редизайном как самой схемы БД, так и приложения.

3.11. Выводы к третьему разделу

1. ДКНФ – это эталон качества схемы БД.

2. Модифицируемость и ДКНФ – равнозначные свойства.

3. Интероперабельность и кроссплатформность наиболее полно реализовать для приложений, основанных на ДКНФ-схемах БД.

4. Используя ДКНФ, проектировщик может обнаруживать аномалии в самой ПрО.

5. Полнота решетки отношений и их безаномальность дает возможность минимизировать листинг приложения, так как унифицирует большинство процедур.

РАЗДЕЛ 4
ТЕМПОРАЛЬНОСТЬ И ХРАНИЛИЩА ДАННЫХ

4.1. Введение к четвертому разделу

В [196] отмечено, что использование традиционной ММД [414, 423] в МХД сопряжено с определенными трудностями. Для ее реализации требуется больший объем памяти, так как при реализации физической многомерности используется большое количество технической информации. Поэтому объем данных, который может поддерживаться МХД, обычно не превышает нескольких десятков гигабайт. При этом, МХД труднее поддается модификации: при необходимости встроить еще одно измерение требуется выполнить физическую перестройку всего многомерного куба.

Из этого следует, что применение систем хранения, в основе которых лежит многомерное представление данных, целесообразно только в тех случаях, когда объем используемых данных сравнительно невелик, а сама многомерная модель имеет стабильный набор измерений.

РК как шаблон для схем и OLTP-БД, и OLAP-ХД, позволяет иначе решить эти вопросы. Проведем анализ возможности организации ХД с использованием РК.

4.2. Постановка задачи

Рассмотрим более подробно пример из [139]. Пусть в ПрО «Поликлиника» регулярно формируются отчеты по неключевому атрибуту *ПолПациента* из отношения ВИЗИТ К ВРАЧУ (*Пациент, Талон, ПолПациента, ...*). Для упрощения процедуры синтеза отчетов может быть использован принцип «маска для роли» атомарной сущности-объекта, предложенный в [205] – отдельное отношения, которое является частичной копией атомарного отношения *ПАЦИЕНТ*. Моделирование и поддержка таких отношений существенно сокращает потребность проектировщика разрабатывать листинги запросов, что в итоге улучшает эксплуатационные свойства приложения .

Отношение-связь с ключами *ПолПациента+КодТалона* может формироваться в реальном времени по факту обновления базового отношения. В формируемом отношении, которое универсальными процедурами в процессе эксплуатации поддерживается в целостном и актуальном состоянии, может храниться и необходимая статистика – формироваться естественные неключевые шунтирующие атрибуты.

В [141] такая декомпозиция отношения *ПАЦИЕНТ* на два отношения - *ПОЛ ПАЦИЕНТА* и, например, *ВОЗРАСТ ПАЦИЕНТА* - определена как денормализация [257]. Однако, Р. Фейджин в [378] аналогичный прием привел как пример нормализации схемы отношения: «…Схема может быть разложена на две … кортеж, содержащий данные о поле человека, входит в первое отношение, а кортежи, содержащие данные о его профессии, входят во второе отношение. Схема *SEX_ROLE* удовлетворяет функциональной зависимости (ФЗ) $PERSON \rightarrow ROLE$, в то же время схема *PROF_ROLE* не удовлетворяет этой зависимости. … Преимущество разложения состоит в том, что механизм обработки ключей системы может тогда предотвратить вставку двух ролей пола для одного человека».

Таким образом, моделирование абстракций, учет рекурсивных связей и темпоральности данных, а также возможность многократного добавления ключевых атрибутов, отвечающих за различные состояния составных сущностей-объектов, выполняющих разные роли в ПрО, в каркасной схеме БД могут быть осуществлены с использованием масок [205, 217]. При этом возможен учет номеров отрезков времени таких модификаций в таких отношениях.

Этот механизм позволяет вносить изменения в схему БД по полно-модифицируемому принципу, а не с существенным редизайном как самой схемы БД, так и приложения.

Схожий подход к проектированию рекурсивных связей описан и в [89]. Однако вопрос нормализации отношений, построенных на частичных копиях сущностей-объектов, там не исследован.

4.3. Маски сущностей-объектов

В [211, 215] предложена классификация сущностей-объектов произвольной ПрО, которая наиболее соответствует каркасной совокупности. Там же формально определены артефакты как сущности-копии, данные о которых будут условно размещаться в БД по решению пользователя. Важным частным случаем артефакта является маска сущности-объекта. Дадим определения.

Под *маской* сущности-объекта будем понимать такой многоместный предикат $M_j(X_k,B_j)$, ключевым предикатом которого является ключевой предикат некоторого семантически атомарного, слабого или составного предиката (т.е. исходной сущности-объекта), а аргументами - ограниченная совокупность аргументов этого же предиката.

Неформально, под маской сущности-объекта будем понимать такую частичную копию сущности-объекта (такой артефакт), которая является носителем ограниченной группы атрибутов этой сущности-объекта, отвечающих лишь за единственную роль [358] сущности-объекта.

На языке схем БД под масками отношения $R(X_k,A_n)$, в котором совокупность атрибутов X_k является единственным ключом, а совокупность атрибутов A_n не является ключом и полностью зависит от ключа, будем понимать совокупность отношений $M_j(X_k,B_j)$, в каждом из которых имеется лишь некоторая уникальная часть неключевых атрибутов $B_j \subset A_n$, где $j=1, n$ и $B_1+B_2+\ldots+B_j+\ldots=A_n$, а также имеется единственный ключ X_k – копия ключа отношения R.

Т.о., отношения $R(X_k,A_n)$ и $M_j(X_k,B_j)$ – подобны [206]. Процедура их формирования может иметь несколько частных случаев.

1. $M_j(X_i,B_j)$, где $i<k$ – рассечение отношения $R(X_k,A_n)$ операцией split [378] на слои при постоянстве некоторой части ключа $X_{k-i}=const$.

2. $M_j(X_k, X_{k+m}, A_n)$, где X_{k+m} – новое дополнительное звено единственного ключа, которое отвечает за этимологию смысла [215] роли моделируемой сущности-объекта. Очевидно, что такое звено может быть как унарным, так и многоарным. Последний случай отвечает за иерархическую маску.

3. $M_j(Y_j, X_k, A_n)$, где Y_j – новый дополнительный ключ отношения, как правило, суррогатный унарный атрибут.

Каждая сущность-объект может иметь в ПрО определенное число разных масок. Т.е., или множество, или несколько, или лишь одну. Тем не менее, число масок обусловливается числом ролей сущности-объекта в ПрО, т.е. связей, в которых принимает участие сущность-объект. Если, например, рассматривается сущность-объект «человек», то таких масок может быть значительное количество. Это - «специальность», «должность», «воинское звание», «научная степень» и т.д. Тем не менее, если это сущность-объект «животное», то масок может быть намного меньше: «домашние животные», «дикие животные», «скот» и т.д.

Маски необходимы лишь тем сущностям-объектам, которые могут принимать участие в разных ролях. Очевидно, что связи, которые моделируются составными сущностями-объектами, дальнейшего участия в новых связях не принимают. В разделе 3 это свойство каркасной совокупности доказано в виде леммы 3.4. [217]. Характерной же особенностью участия в разных связях атомарных и слабых сущностей-объектов, является их темпоральность. Иными словами, каждая роль, которую выполняют атомарные или слабые сущности-объекты в одной из связей, носит временный характер.

Способ простого декартова произведения, используемый в [203], также формирует РК и тем самым учитывает все возможные связи между группами сущностей-объектов, которые могут образовываться в ПрО. Однако он не

учитывает влияние разнообразия ролей каждой сущности-объекта на разнообразие связей, которое ограничивает его применение.

Для моделирования обобщений, специализаций, типизаций, конкретизаций и иных вариантов абстракций [47, 55, 290] маски могут применяться по следующим схемам.

1. Для регулярного формирования в отдельном отношении БД группы атрибутов как фрагмента одного исходного отношения (или группы отношений), выделенного по определенному условию, налагаемому на одну или несколько частей ключевого атрибута (например, равенство определенному значению). Однако в отношения-маски ключевые атрибуты-фильтры не копируются.
2. Для сокращения списка экземпляров справочной сущности-объекта (выделенных по определенному условию) для дальнейшего использования лишь в сокращенном виде.
3. Для совмещения метаданных и данных в одном отношении.
4. Для корректного моделирования рекурсивных связей между экземплярами одной и той же сущности-объекта, участвующими в разных ролях. Например, физические или юридические лица, одновременно выступающие в ролях покупатель-продавец, учащийся-преподаватель, врач-пациент и т.п.
5. Для переименования ключевых атрибутов.

Синтез отношений-масок может осуществляться или посредством пост-процедурного запроса, или запроса реального времени, выполнение которого синхронизировано с формированием исходного отношения. Аналогичная потребность в таком формировании отношений описана в [452]: «отношения должны быть доступны по-разному у разных пользователей за один и тот же промежуток времени».

4.3.1. Нормализация отношений, построенных на масках

Покажем то, что маски, построенные на актуальных ячейках РК [206], обладают 5НФ. А при некоторых условиях на ограничения - и ДКНФ [378].

Пусть отношение $M_j(X_k,B_j)$ является *j*-й маской отношения $R(X_k,A_n)$, Тогда совокупность атрибутов X_k является единственным ключом, а совокупность атрибутов A_n и B_j не является ключом и полностью зависит от ключа. При этом $B_j \subset A_n$. Если отношение $R(X_k,A_n)$ является актуальной ячейкой РК и его ограничения являются логическим следствием особых ограничений [206], то оно находится в безаномальной ДКНФ [378].

Отношение $M_j(X_k,B_j)$ подобно [206] отношению $R(X_k,A_n)$ с коэффициентом подобия $\nu \geq 1$. Подобными в [206] названы схемы отношений с равным числом ключей и ключевых ФЗ. При этом, если в сравниваемых отношениях существует разное число ключей, то отношения не являются подобными. А тип, число и арность неключевых атрибутов наподобие схем не влияют. Под *коэффициентом подобия* схем отношений в [206] понимается целая часть от деления большей арности ключей на меньшую.

Тогда форма нормализации маски $M_j(X_k,B_j)$ строго соответствует 5НФ, так как, исходя из подобия, имеет единственный ключ и единственную ключевую ФЗ $X_k \to B_j$. А условием принадлежности отношения к ДКНФ по теореме о безаномальности актуальных ячеек РК является логическое следствие ограничений на домены и ключи данного отношения особым ограничениям [206].

В работе [378] приведен пример, который подтверждает данный вывод. Пусть R – реляционная схема с атрибутами *КодComp*, *Статус*, *Зарплата* и ограничениями:

DD(*R*)=IN(*КодComp*,{*n: n is a two-digit integer*})∩

IN(*Зарплата*,{*n: 1000≤n≤10000*})∩IN(*Статус*, {0;1})

KD(*R*)=KEY(X)

$\forall t((t[STATUS]=0)\Rightarrow(t[Z]\leq 5000))$,

где атрибут *Статус* – переменная STATUS, а атрибут *Зарплата* - переменная Z.

Смысл ограничений выражается так: *КодСотр* - целое число из двух цифр, и *КодСотр* является ключом; имеются два статуса: 0 и 1; значение *Зарплата* может изменяться в диапазоне между 1000 и 10000. Исключение: сотрудники со статусом 0 имеют зарплату не больше 5000.

Эта схема находится в 4НФ, и даже в 5НФ. Однако, в схеме есть следующая аномалия вставки. Пусть *R* – отношение с первыми тремя кортежами в таблице. Это отношение – допустимый экземпляр схемы. Пусть *t* – четвертый кортеж (06, 0, 5600). Кортеж *t* совместим с отношением *R*. Однако, вставив кортеж *t* в *R*, мы бы получили отношение, которое не является допустимым экземпляром схемы, так оно нарушает ограничение, что любой работник со статусом 0 не может иметь зарплату больше, чем 5000.

Таблица. 4.1.

КодСотр+Статус+Зарплата (5НФ)

КодСотр	*Статус*	*Зарплата*
03	1	7300
04	0	3200
05	1	4600
06	0	5600

Следовательно, схема в примере не находится в ДКНФ. И для получения ДКНФ одного только подхода декомпозиции не достаточно. Здесь вновь может быть использован оператор разделения split [378]. Таким образом, Р. Фейджин в [378] заменяет первоначальную схему R примера двумя схемами R_1 и R_2, где экземпляры схемы R_1 содержат информацию о сотрудниках со статусом 0, а экземпляры схемы R_2 - о сотрудниках со статусом 1. Таким образом, отношение со схемой R_1 (первая маска), содержащее информацию о сотрудниках со статусом 0, имеет только атрибуты *КодСотр* и *Зарплата*. Атрибут *Статус* больше не является

необходимым, так как все значения тождественно равны 0. Ограничения отношения R_1 теперь имеют вид:

DD(R)=IN(X,{*n: n is a two-digit integer*})∩IN(Z,{*n: 1000≤n≤5000*})

KD(R)=KEY(X)

Отношение со схемой R_2 (вторая маска) содержит информацию о сотрудниках со статусом 1. Исключение лишь в том, что в ограничениях отношения R_2 значение 5000 заменено на 10000.

Проверим теперь, какова форма отношений-масок в частных случаях.

1. Слои $M_j(X_i, A_n)$ отношения $R(X_k, A_n)$ при постоянстве некоторой части ключа $X_{k-i} = const$, где $i < k$. Поскольку отношение $M_j(X_i, A_n)$ подобно отношению $R(X_k, A_n)$, имеет единственный многоарный ключ X_i, единственную ключевую ФЗ и шунтировано неключевыми атрибутами A_n, оно относится к 5НФ. А при ограничениях, являющихся логическим следствием особых ограничений, и к ДКНФ.

2. Отношение $M_j(X_k, X_{k+m}, A_n)$, где X_{k+m} – новое дополнительное звено единственного ключа, которое может быть как унарным, так и многоарным. Также подобно отношению $R(X_k, A_n)$, имеет единственный многоарный ключ $X_k + X_{k+m}$, единственную ключевую ФЗ и шунтировано неключевыми атрибутами A_n. Также относится к 5НФ. А при ограничениях, являющихся логическим следствием особых ограничений, и к ДКНФ.

3. Отношение $M_j(Y_j, X_k, A_n)$ с новым дополнительным независимым ключом Y_j. В связи с наличием транзитивной ФЗ от двух независимых ключей это отношение должно быть декомпозировано на 2 отношения.

В [215] это утверждение доказано в виде леммы. Таким образом, в большинстве случаев использование масок не приводит схему отношения к денормализации. Кроме лишь случая введения дополнительного ключа. Поэтому вывод о том, что такая замена всегда снижает форму нормализации отношений, ошибочен.

Однако при использовании масок могут возникать некоторые некритические проблемы программной реализации. Рассмотрим их более подробно.

1. Потребность объединить в одной операции (запросе) и данные, и метаданные: восстановить исключенную связь, но не между значениями исходных и результирующих атрибутов, а между значениями атрибутов исходных отношений и именами результирующих отношений-масок. Причем, как правило, разным значениям фильтруемых атрибутов соответствуют разные группы результирующих отношений-масок. Как уже упоминалось, аналогичный тип отношения уже применялся разными авторами [89, 452, 467]. «Такое рассечение … обычно требует разных имен отношений в запросах, в зависимости от значений в таблице». Все указанные авторы отнесли эти отношения к категории денормализованых. Однако, как показано ранее, маски, подобные актуальным ячейкам РК [206], имеют форму не ниже 5НФ.

Формально, пусть в отношении со схемой $R(X_1,X_2,\ldots,X_n,A_j)$ имеются атрибуты $X_k + X_l$, каждое текущее значение конкатенации которых соответствует некоторым константам из совокупности $\{X_{c_1},X_{c_2},\ldots,X_{c_s}\}$.

Пусть в приложении работает процедура, которая по факту появления в отношении R каждой новой записи проверяет условие равенства $X_k + X_l = \{X_{c_s}\}$. И при его выполнении, в зависимости от значения константы X_{c_s}, формирует новые записи только в *f*-й группе, состоящей из *p* отношений $Q_p^f(X_m^f,\ldots,X_n^f,B_j^f)$, где $p = 1,P^f$. Однако при этом в любом отношении Q_p^f атрибуты $X_k + X_l$ отсутствуют – в этом смысл частичной копии отношения, т.е. маски.

Тогда данная процедура будет выполнимой, если на момент выполнения условия $X_k + X_l = X_{c_s}$ для любых c_s, p и f существует однозначное соответствие $X_{c_s} \leftrightarrow Q_p{}^f$. Такое соответствие может быть

заранее организовано проектировщиком в виде отдельного отношения-маски $Z(X_{c_s}, Q_p)$ или $Z(X_{c_s}, D_p)$, где вместо имени отношения Q_p использован, например, номер отношения D_p – т.е., метаданные.

Однако, исходя из подхода [361], отношение $Z(X_{c_s}, Q_p)$ или $Z(X_{c_s}, D_p)$ может восприниматься как выходящее за рамки начальной РМД. Но исходя из концепции современного языка SQL такое объединение - типовая процедура.

2. В отношении-маске появляются взаимные ФЗ [305] в 2-х ключах – первичном и новом $X \leftrightarrow Y$. Это приводит к денормализации схемы БД, потому, что отношение-маска $R(X,Y,A)$ формально обладает «паразитной» транзитивной ФЗ неключевого атрибута от каждого из ключей: $X \rightarrow Y$, $Y \rightarrow A$ и $Y \rightarrow X$, $X \rightarrow A$. Тот же факт, что $X \leftrightarrow Y$ – ключи, не компенсирует такую транзитивность. Аномалия устраняется декомпозицией исходного отношения на 2 отношения $R_1(X,Y)$ и $R_2(Y,A)$, где Y – новый ключ. Но при этом появляется дополнительное суррогатное отношение $R_1(X,Y)$ – взаимно-однозначное соответствие ключевых атрибутов («хелпер-таблица» [423, 424]).

Однако, как правило, проектировщики не проводят такую декомпозицию, руководствуясь соображением, что схема $R(X,Y,A)$ более компактна, а вероятность проявления аномалии мала. При этом отношение-маска со схемой $R(X,Y,A)$ выполняет две функции одновременно – и моделирует однозначное соответствие между разными ключевыми атрибутами, и моделирует новую сущность-объект с атрибутами A.

4.3.2. Алгоритм проектирования схем с использованием масок

Число масок произвольной сущности-объекта не может быть любым или отделенным от числа других масок этой сущности-объекта или иных сущностей-объектов. При образовании бинарных, тернарних или связей более высокой арности со стороны каждой задействованной в этой связи

сущности-объекта должно быть «предоставлено» соответствующую маску. А это, в свою очередь, означает, что маски актуализируются или аннулируются синхронизировано с актуализацией или аннулированием соответствующих связей, т.е. ролей, в которых те или иные группы сущностей-объектов принимают участие. Это соответствие масок существенно упрощает построение модели ПрО.

Признаком отнесения характеристик-атрибутов к той или иной маске есть смысловая, то есть предикатная, зависимость конкретной характеристики-атрибута от конкретной маски сущности-объекта. Процедура такого отнесения отвечает КМД. Используется тот факт, что каждый атрибут принадлежит лишь одной уникальной сущности-объекту. А также то, что лишь общая совокупность всех атрибутов образовывает полную взаимно-независимую совокупность свойств. И что объединение разных групп характеристик от разных предикатов, в одну сущность-объект (в одно в одно отношение), что часто наблюдается в искусственных сущностях-объектах (в артефактах), зачастую приводит к появлению нежелательных «паразитных» межатрибутных ФЗ.

Таким образом, формальным признаком корректного отбора атрибутов сущности-объекта в отдельную маску есть отсутствие в совокупности атрибутов транзитивных зависимостей, а также отсутствие дополнительных потенциальных ключей в кортежах отношений. При таком принципе отбора атрибутов сущности-объекта в совокупность атрибутов маски сущности-объекта не возникает условий существования ФЗ частей составных ключей от неключевых атрибутов и наоборот.

Итак, сама маска есть не только поименованной частичной копией сущности-объекта, а и эксклюзивным носителем группы взаимно-независимых атрибутов именно этой сущности-объекта. Таким образом, каждое отношение-маска вмещает лишь единственный составной ключ и группу функционально независимых один от другого атрибутов маски, которые полностью зависят лишь от ключа.

Композиционный метод синтеза схем отношений благодаря алгоритму управления ФЗ предложил П.А. Бернштейн в 1975 году [348, 349] Там же отмечалось, что под ФЗ понимается связь между сущностями-объектами и между сущностями-объектами и их атрибутами. Тем не менее, поскольку входными факторами вышеупомянутого метода является набор ФЗ определенной ПрО, это есть его существенным недостатком. Реляционные схемы, которые образовываются в соответствии с этим алгоритмом, зависят от семантики ПрО. В отличие от упомянутого, описываемый алгоритм предоставляет процедуру абстрагирования от ФЗ.

Таким образом, в алгоритме синтеза РК как шаблона схемы БД, предложенного в [202 - 224], необходимо учесть маски сущностей-объектов. По аналогии с [216], будем употреблять термин «маска» в значении логической частичной копии сущности-объекта, а термин «домен-маска» в значении физического размещения данных из маски в участке памяти.

Шаг 1. Для каждой сущности-объекта в памяти отводится несколько участков для размещения элементов хранения. В каждом участке размещают домен-маску с идентификатором ячейки, структура которого строго соответствует структуре найденной этимологии. Таким образом, создается множество доменов-масок.

Для использования в схеме БД отношений-масок важно правильно моделировать слабые сущности-объекты. Поэтому домены-маски назначаются всем маскам базовой совокупности [215] сущностей-объектов, в том числе и маскам слабых сущностей-объектов.

Поскольку в общем случае слабые сущности-объекты зависят от цепи сущностей-объектов, где каждая сущность-звено является также слабой сущностью-объектом (исключая лишь наивысшую), на начальном этапе проектирования схемы БД маски назначают так, будто этой зависимости не существует. Т.е. аналогично процедуре получения базовой совокупности сущностей-объектов, игнорируя иерархическую зависимость. Алгоритм предусматривает дальнейший учет всех типов связей между масками, а

значит и иерархических связей между сущностями-объектами - это действие не приведет к потере иерархических связей.

При этом проектировщик схемы БД должен отслеживать семантическое соответствие каждой маски каждой роли, т.е. соответствие масок и связей: чтобы одна маска уникально отвечала единственной роли.

Шаг 2. Осуществляется формирование расширенного РК связей масок – сочетание декартовых произведений всех упомянутых доменов-масок между собой по принципу «все на все». Общее число $S(t)$ полученных таким образом отношений для БД существенно увеличивается в сравнении с другими способами.

С учетом множества масок каждой сущности-объекта и зависимости количества сущностей от номера промежутка времени актуальности схемы БД, общее число отношений определяется выражением:

$$S(t) = \sum_{K=1}^{NN(t)} \frac{NN(t)!}{K!(NN(t)-K)!} = 2^{NN(t)} - 1$$

где K – текущая арность связей групп доменов-масок, а $NN(t)$ – общее число доменов-масок, которое зависит от t – номера промежутка времени актуальности схемы БД, на протяжении которого эта схема не претерпевает модификации. Общее же число доменов-масок определяются формулой:

$$NN(t) = \sum_{i=1}^{N(t)} \sum_{j=1}^{M(i,t)} \alpha(i,j,t),$$

где, в свою очередь, $\alpha(i,j,t)$ – признаки актуальности домена-маски.

Признак актуальности – это формальный массив целых чисел, каждое из которых определяется совокупностью индексов (i,j,t). И в пределах данного алгоритма принимается или 0, что соответствует аннулированию домена-маски, или 1, что соответствует актуальность домена-маски. Тогда i – индекс, который определяет номер сущности, $N(t)$ – общее число сущностей-объектов на промежутке времени t, $M(i,t)$ – число доменов-масок каждой i-й сущности-объекта на промежутке времени t, а j - индекс, который

определяет номер домена-маски *i*-й сущности-объекта. Суммарное число доменов-масок для одной сущности-объекта формирует внутренняя сумма. Тогда внешняя сумма формирует общее число доменов-масок.

На рис. 4.1. приведена схема расширенного РК, построенного на булеане $NN(t)$ масок сущностей-объектов – схема БД, которая моделирует ПрО, где $K_{111},K_{121},K_{131},K_{141},\ldots,K_{NNM1}$ – совокупность структурированных идентификаторов бесконечных столбцов доменов-масок, а также структура многоарных отношений каждого уровня связей, которые получены путем декартова произведения доменов-масок между собой. При этом *M* обозначает зависимый от номера сущности-объекта индекс, который соответствует числу масок каждой сущности-объекта.

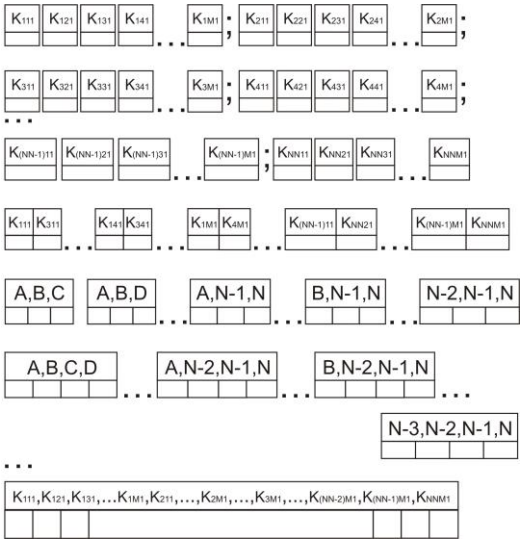


Рис. 4.1 Схема расширенного РК

Для экономии места на рисунке символ *i* не приведен. По этой же причине приведены лишь некоторые случайные отношения бинарных связей, а также на третьем и четвертом уровне арности отношений вместо трехмерного идентификатора K_{ijt} обобщенно показаны массивы $A,B,C,D,\ldots,N$, т.е., использованные символы имен сущностей-объектов, которые обобщают имена своих масок. Последнее *NN*-арное отношение показано с раскрытой структурой ключа.

4.4. Темпоральность каркасной базы данных

Еще одно отличие описанного алгоритма заключается в структуре идентификатора, который может иметь единое имя для всех отношений и сквозную трехмерную индексацию (i, j, t). Индексы имеют тот же смысл, что и в выражении общего числа доменов-масок. Каждый из индексов ключа уникально отвечает каждой маске каждой сущности-объекта.

Т.е., каждый из индексов отвечает за свой базовый фактор алгоритма, а именно:

$i = 1, N(t)$ – номер каждой сущности-объекта, где $N(t)$ – общее число сущностей-объектов за *t*-й промежуток времени,

$j = 1, M(i,t)$ – номер маски *i*-й сущности-объекта за *t*-й промежуток времени,

t - номер отрезка времени актуальности текущего состояния *t*-й модификации совокупности всех (i, j)-х отношений.

Таким образом, за промежуток времени, которое имеет номер *t*, схема всей БД остается без изменений, т.е., не модифицируется. А на моменте времени, которое имеет номер $t+1$, эта же совокупность отношений уже получает модификацию своего состояния.

Такая модификация может оказаться как в мизерном изменении лишь размера одного из столбцов уже существующего отношения, так и в появлении новой группы отношений. Пользователь получает возможность самостоятельно назначать и использовать любое формальное условие перехода к новому коду отрезка времени, который характеризует актуальность состояния схемы БД. А значит, к новой совокупности отношений и кортежей.

Благодаря кодированию промежутков времени, на протяжении которых состояние структуры совокупности отношений сохраняет актуальность, алгоритм предоставляет возможность анализировать все слои состояний схемы БД или отдельно один от другого, или в полной совокупности. Такая технология синтеза схемы БД предоставляет

возможность хранения каждого отдельного *t*-слоя совокупности отношений в целостном виде со всеми наработанными данными за этот промежуток времени. И построить *темпорально-слоевой архив* данных, что существенно отличается от концепции архива «кубов данных» [414, 423].

В приведенном алгоритме также не существует ограничений относительно момента добавления дополнительных доменов-масок от начальных, или даже от новых сущностей-объектов, которые не были учтены проектировщиком на начальном этапе. Такое добавление и является упомянутой модификацией дежурного состояния схемы БД.

Дополнительный «физический» смысл констант $\alpha(i,j,t)$ - это еще и факт размножения определенной маски, когда определенная константа равняется 2, 3, 4 и т.д. Это, в свою очередь, означает моделирование возможности многоразового одновременного выполнения одной сущностью-объектом одной роли, т.е. участие сущности-объекта своей одной маской в одном типе связи несколько раз. Такая ситуация не имеет аналогов в ПрО.

Как уже отмечалось, используется принцип уникальности - каждая маска используется лишь для одной роли, а в каждой роли, т.е. в каждом типе связи, сущность-объект принимает участие этой маской лишь один раз. Поэтому, даже рекурсивная связь произвольной арности одного и того же экземпляра сущности-объекта, которая в теории проектирования ХД считается одним из существенных противоречий ПрО, органически моделируется описываемым алгоритмом за счет разных масок, которые принадлежат одной сущности-объекту. Тем не менее, в пределах алгоритма размножение еще и масок - сугубо теоретическая ситуация - не создаст существенных структурных проблем и противоречий.

Описанный подход к моделированию темпоральности данных несколько отличается от общеизвестных [120].

4.5. Хранилища данных

Потребность в моделировании темпоральности данных вызвана динамическим характером функционирования любой системы. Исторически

это было важной причиной разделения всех приложений БД на 2 типа - оперативные, в которых моделирование времени может быть минимизировано, и исторические, в которых время играет принципиальную роль. Эти БД получили соответствующие названия – «оперативные» (или «транзакционные») и «хранилища» (или «аналитические»).

Принцип ХД и их определение предложил У. Инмон [415]. В его подходе ХД — это «предметно-ориентированная, интегрированная, содержащая исторические данные, целостная совокупность данных, предназначенная для поддержки принятия управленческих решений». Там же дана и классификация ХД. Из описанных типов нас более всего будет интересовать виртуальное ХД.

Виртуальное ХД [415] — это система, предоставляющая интерфейсы и методы доступа к оперативно-регистрирующей системе, которые эмулируют работу с данными как с ХД. Это означает, что виртуальное ХД можно организовать, создав ряд «обращений» (view) к БД, либо применив специальные средства доступа [423].

Главными достоинствами такого подхода являются простота и малая стоимость реализации, единая платформа с источником информации и отсутствие сетевых соединений между источником информации и ХД. А ХД, организованное в соответствии с каркасной схемой БД, избавлено от большинства недостатков, описанных в [283]. Схема такого ХД и приложения, его обслуживающие, могут быть динамически модифицируемы. А высокая производительность такой системы обеспечивается минимизацией операций соединения [207].

Следовательно, и модифицируемость схем БД, и интеграция данных с другими источниками, и отслеживание исторических измерений, и подобие схем БД [206], и гарантии чистоты данных [283], обеспечиваемые ограничениями на домены и ключи, позволяют сделать вывод о том, что объединение свойств оперативной (OLTP) и архивной (OLAP) БД в одной каркасной схеме становится возможным.

Авторы [196] считают доказанным утверждение о том, что «…РМД не является оптимальной с точки зрения задач анализа, поскольку предполагает высокую степень нормализации, в результате чего снижается скорость выполнения запросов» . Однако, как показано в [207, 208], при определенных условиях этот вывод не соответствует действительности. Скорость выполнения запросов снижается не из-за нормализации отношений, а из-за традиции проектирования всех оперативных связей ПрО только посредством бинарных отношений. Дальнейшее использование таких отношений в произвольных запросах требует значительного числа операций соединения. Это и приводит к значительным временным затратам. Высоко-нормализованные каркасные отношения (актуальные ячейки РК [208]), избавляющие пользователя от проектирования большинства оперативных запросов на соединение, доказывают обратное утверждение.

Проанализируем основные мотивы [135] разделения на оперативные БД и аналитические ХД. А также сопоставим их со свойствами каркасной БД.

1. «Для проведения анализа данных требуется привлечение внешних источников информации (например, статистических отчетов), поэтому ХД должно включать как внутренние корпоративные данные, так и внешние данные».

Современные интернет - приложения позволяют строить распределенные системы любой вложенности. Поэтому приложения могут формировать и поддерживать в актуальном состоянии все необходимые справочники непосредственно из «паутины». При этом объем таких отношений-справочников будет значительно меньшим по сравнению с объемом отношений оперативных данных.

2. «Для оперативной обработки требуются данные за несколько последних месяцев, поэтому объем аналитических БД как минимум на порядок больше объема оперативных».

Работая с типизированной и унифицированной схемой БД, построенной на РК, пользователь избавлен от проблемы соединений ЗНФ-отношений, зависимых от текущей семантики ПрО, посредством сложных SQL-запросов разных типов. Поэтому большинство операций на каркасной БД – это индексные выборки. При такой конфигурации БД скорость реакции системы существенно повышается. И пользовательские приложения не конкурируют между собой, потому что работают с разными фрагментами отношений.

3. «Оперативные БД могут содержать семантически эквивалентную информацию, представленную в разных форматах, а иногда даже противоречивую. ХД должно содержать единообразную и согласованную информацию, поэтому необходима компонента «очистки» информации».

Однако строгость ограничений на домены и ключи (взаимообусловленность данных [208]) не дает возможности использовать данные от сущностей-объектов в виде омонимов и синонимов.

4. «Большинство запросов к оперативной БД известно уже при проектировании, а набор запросов к ХД предсказать невозможно. Размеры ХД стимулируют использование запросов с агрегатами - сумма, минимальное, максимальное, среднее значение и т.д.»

В [205] показано, что каркасный анализ ПрО позволяет предсказать большинство актуальных связей в ПрО. И тем самым проектировать отношения, накапливающие в своих шунтирующих атрибутах все необходимые виды агрегированных данных. А также интегрировать новые запросы пользователей в новых отношениях. Поскольку КМД основана на булеане связей, проектировщик имеет возможность динамически интегрировать в хранимых отношениях данные для любого вновь возникшего запроса.

5. «Схемы оперативных БД являются изменчивыми, а при малой изменчивости ХД нужны быстрые методы индексации при массовой выборке и хранении агрегированных данных».

Те СУБД, которые поддерживают унифицированные схему оперативных и аналитических БД, имеют равноправные операции динамической модификации таких схем. Например, позволяют применять единые процедуры индексирования как оперативных, так и аналитических отношений. А также процедуры отслеживания целостности ключей и неключевых оперативных и агрегированных данных.

Таким образом, как показывает практика, ни один из перечисленных факторов не позволяет однозначно сделать вывод о том, что нецелесообразно эксплуатировать единое пространство БД и как оперативное, и как аналитическое хранилище.

Основная причина такого разделения вероятнее всего заключается в невозможности корректного совмещения совокупности 3НФ-отношений, которые навязаны большинством учебников по БД, и семантически-ориентированных схем ХД типа многомерных кубов [415] или реляционной «звезды»/«снежинки» [423]. Более того, потребность пользователей в таком объединении является общеизвестной. В [315] эта концепция описана в разделах «исчезновение отдельных ХД» и «ХД в режиме реального времени». А ранее аналогичная потребность была исследована и предложена в [221, 222]. Дальнейшие внедрения этого подхода разными пользователями подтвердили такой вывод.

Причина разделения БД на OLTP и OLAP заключается не в неудобстве использования, а в неприспособленности нормализованных в соответствии с алгоритмом [360] отношений к OLAP-запросам.

4.5.1. Модификация и деактуализация данных

Очевидно, что модификация БД может быть двух типов – метаданных и данных. В связи с возможностью объединения в каркасной БД и оперативных, и аналитических («исторически-архивных») данных, введем понятие корректной модификации.

Под *корректной модификацией метаданных* будем понимать такое изменение схемы БД, которое не снижает степени нормализации ни одного из отношений БД, а также не нарушает целостность существующих данных.

Примером корректной модификации метаданных является расширение поля для атрибута, которое не меняет тип и значения существующих в нем данных. Однако если модификация атрибута затрагивает значения данных, модификация будет некорректной.

Добавление нового шунтирующего атрибута, который не является детерминантом новой ФЗ, не изменит степени нормализации каркасного отношения. Но если в таком отношении существуют данные, то для корректности модификации добавление еще одного шунтирующего атрибута должно сопровождаться двумя действиями: инициализацией незаполненных полей нового атрибута данными, которые не разрушат целостности иных данных, а также фиксацией даты актуализации этого отношения и этого атрибута.

Для размещения даты актуализации метаданных и данных проектировщик использует отдельные отношения-маски, которые связывают метаданные и данные. Очевидно, что проектировщик должен заблаговременно позаботиться о таких ситуациях и подобрать те СУБД, которые поддерживают такие решения. Отказ уполномоченного лица от заполнения пустых полей нарушит целостность БД.

В настоящей работе более глубокие вопросы модификации метаданных (схем БД) не рассмотрены. Однако необходимо отметить, что некорректная модификация метаданных (редактирование схем «задним числом») вызвана или плохим предварительным анализом ПрО, или национальными традициями функционирования ПрО.

Под *корректной модификацией данных* будем понимать такое изменение значений одного или более атрибутов, которое не вызывает потерю целостности БД.

Поскольку и приложение, и БД, разработанные на основе КМД, обладают универсальными признаками – и оперативной системы, и ХД, примером некорректной модификации данных является операция каскадного удаления. Она также вызвана, как правило, плохим анализом ПрО, случайными ошибками проектировщиков или некорректной эксплуатацией системы. Эта операция разрешена только малому числу уполномоченных администраторов. Поэтому такое редактирование данных, моделирующих, например, структуру ПрО (так называемых «справочников»), в каркасной схеме БД должно быть сведено лишь к каскадной деактуализации кортежей.

Рассмотрим такой алгоритм на примере. Очевидно, что одной из групп данных, которые подвержены изменениям в ПрО, являются словесные наименования адресных категорий (улиц, переулков, площадей и т.п.), периодически присваиваемые и переименовываемые пользователями в соответствии со своими национальными традициями. В связи с этим в каркасном анализе ПрО наименования могут выступать как независимые атомарные или слабые сущности-объекты. Рассмотрим процедуру модификаций и деактуализации этих данных в БД на примере моделирования составной сущности-объекта АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ. Фрагмент схемы БД будет состоять из следующих отношений.

ТИПЫ АДРЕСНЫХ КАТЕГОРИЙ (*КодТипаАдрКатег*, Наименование, СокрНаимен, КодАктуальнТипаАдрКатег, ДатаАктуальнТипаАдрКатег) – атомарная сущность-объект. Содержит список наименований традиционных типов адресных категорий населенных пунктов городов, поселков или сел без привязки к типу национальной традиции и перечню государств: { бульвар, переулок, проспект, площадь, улица, …, тупик, avenue, boulevard, …, square, street, …}.

Данный список может быть только пополняемым. Потребность в аннулировании любой из этих категорий могла бы возникнуть лишь при условии однозначного отказа от ее использования во всех населенных пунктах всех национальностей. Однако автору не известен случай полной

отмены любой из категорий во всех национальных традициях одновременно. Поэтому отношение, моделирующее эту атомарную сущность-объект, вероятнее всего будет стабильным длительное время эксплуатации приложений.

СЛОВАРЬ НАИМЕНОВАНИЙ АДРЕСНЫХ КАТЕГОРИЙ (*КодТипаАдрКатег, КодНаименАдрКатег*, Наименование, СокрНаимен, КодАктНаимАдрКатег, ДатаАктНаимАдрКатег) – слабая сущность-объект. Содержит список собственных наименований традиционных объектов адресных категорий, связанных с традиционными типами, однако без привязки к конкретным населенным пунктам: { …, Alexander, …, Silver, …, Антонова, …, Перова, …, Туполева, … }. Данный список также может быть только пополняемым. Потребность в аннулировании любой из этих записей могла бы возникнуть лишь при условии однозначного отказа от использования данного наименования во всех населенных пунктах всех национальностей. Отношение, моделирующее эту слабую сущность-объект, также будет стабильным.

АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ (*КодГосуд, КодОбл, КодРайона, КодГорода, КодТипаАдрКатег*, *КодНаименАдрКатег,* КодАктАдресКатег, ДатаАктАдрКатег, Примечание) – составная сущность-объект – «справочник улиц в городах государств» с учетом истории изменений собственных наименований.

Для понимания более полной схемы этого фрагмента БД ниже будут приведены иные отношения-справочники, в которых представлены недостающие данные. Рассмотрим, как работает механизм корректного каскадного переименования, например, улицы, название которой уже длительное время используется в адресах значительного числа объектов – зданий, сооружений, физических лиц и т.п.

В соответствии с предварительно настроенным специализированным запросом приложение открывает группу отношений и их индексов. Все необходимые связи между отношениями по соответствующим одноименным

частичным ключевым полям также активизированы. Под активизацией связи между отношениями понимается строгое соответствие в буфере приложения групп записей друг другу по значениям ссылающихся ключей.

Уполномоченный пользователь (например, системный администратор) посредством отдельного пункта меню одного из приложений, работающих с этой БД, из группы записей в отношении АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ(), отфильтрованных именем выбранного города в необходимом государстве, посредством локального (дополнительного) поискового индекса выбирает ту запись, которая посредством части ключа *КодНаименАдрКатег* ссылается на необходимое прежнее наименование искомой адресной категории. Очевидно, что поиск по наименованию может осуществляться в открытом отношении СЛОВАРЬ НАИМЕНОВАНИЙ АДРЕСНЫХ КАТЕГОРИЙ() в этом же пункте меню посредством обособленного окна поиска.

Визуально активизировав найденное новое наименование, пользователь в текущей записи отношения АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ() заменяет прежнее значение ключа *КодНаименАдрКатег* на новое. Атрибут «наименование адресной категории» в этом и следующих по иерархии отношениях отсутствует. Этот принцип исключает избыточность данных, многозначность форматов, разночтение терминов и т.п. Если же в отношении СЛОВАРЬ НАИМЕНОВАНИЙ АДРЕСНЫХ КАТЕГОРИЙ() на момент такого поиска нового наименования еще нет, то уполномоченный пользователь вносит его посредством этого же окна поиска. Приложение посредством типовой процедуры предоставляет новому наименованию соответствующее новое значение ключа *КодНаименАдрКатег*.

По факту внесения такого изменения срабатывает одна из каскадных процедур целостного обновления взаимосвязанных данных во всех отношениях, задекларированных в метаданных приложения. Связь между

этими отношениями удерживается в активном состоянии по активным индексам одноименных ключей.

В схему каждого каркасного отношения внесены помимо прочих еще и естественные шунтирующие атрибуты – даты и код актуальности соответствующих связей. Под датой актуальности понимается день (а если нужно, то и час-минута этого дня) начала эксплуатации этой связи – т.е. этой записи в конкретном отношении. А под кодом актуальности понимается бинарная пара {true; false} ({1; 0}). Очевидно, что если значение этого атрибута – «ноль», запись или группа записей текущей совокупности отношений теряет актуальность.

Важной особенностью описанного выше редактирования значения ключа *КодНаименАдрКатег* в отношении АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ() (замены старого значения новым) является то, что старое значение ключа *КодНаименАдрКатег* фактически не редактируется, а сама запись лишь деактуализируется. То есть, в данном пункте меню этого приложения запрос к отношению АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ() настроен так, что после выполнения редактирования в этом отношении (а также во всех связанных с ним отношениях по данной группе частичных ключей) появляется новая запись, в которую вносится новое значение ключа *КодНаименАдрКатег*.

Атрибуты КодАктАдресКатег и ДатаАктАдрКатег этой новой записи получают соответствующие текущие значения, моделирующие актуальность этой записи – символ актуальности и текущую дату обновления. А в записи этого же отношения со старым значением ключа *КодНаименАдрКатег* (а также во всех группах записей во всех связанных с ним отношениях по данной группе частичных ключей) в атрибуте КодАктАдресКатег символ актуальности заменяется символом неактуальности. Атрибут ДатаАктАдрКатег в этой «старой» записи не изменяется.

Строгость отслеживания целостности данных требует хранения «дат деактуализации» в отдельных отношениях-масках. В фоновом режиме в

отдельных отношениях формируются записи с соответствующими значениями всех необходимых ключей и датой деактуализации. Такие дополнительные каркасные отношения-маски имеют вид: ДЕАКТУАЛИЗИРОВАННЫЕ НАИМЕНОВАНИЯ АДРЕСНЫХ КАТЕГОРИЙ (*КодТипаАдрКатег, КодНаименАдрКатег*, ДатаДеАктНаимАдрКатег), ДЕАКТУАЛИЗИРОВАННЫЕ АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ (*КодГосуд, КодОбл, КодРайона, КодГорода, КодТипаАдрКатег*, *КодНаименАдрКатег,* ДатаДеАктАдреснКатег), ДЕАКТУАЛИЗИРОВАННЫЕ АДРЕСА ПРОРЖИВАНИЯ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГосуд, КодОбл, КодРайона, КодГорода, КодТипаАдрКатег*, *КодНаименАдрКатег, НомЗдания, Литера, СубНомЗдания, СубЛитера, НомПомещен, Литера,* ДатаДеАктАдреса).

Таким образом в задекларированном в метаданных приложения искомом «оперативном» отношении АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ(), ради корректного использования записей которого, по сути, и существуют все вышеописанные отношения-справочники, в фоновом режиме все записи, ссылающихся на старое значения ключа *КодНаименАдрКатег*, будут деактуализированы. Также в фоновом режиме в этом же отношении появятся новые записи с новым значением ключа *КодНаименАдрКатег*.

Очевидно, что особенностью отношения АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ() является то, что у каждого адресата может быть множество адресов проживания. И некоторая часть из них с некоторого момента времени может быть уже неактуальной. Однако это не означает, что такие записи должны быть удалены из отношения. Принцип ХД подразумевает бессрочное хранение всех исторических фактов в целостном виде. Поэтому вышеописанное редактирование моделирует ситуацию, когда в ПрО искомые адресаты (например, физические лица) в массовом порядке меняют адрес проживания. И в отношении АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ() обновление группы соответствующих записей осуществляется

автоматизированной процедурой. В случае же моделирования ситуации единичной смены адреса искомого адресата, или появления у него дополнительного адреса, в это отношение соответствующие дополнительные записи вносятся вручную.

Таким образом, по факту выполнения вышеописанного редактирования – замены в отношении АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ() старого значения атрибута *КодНаименАдрКатег* на новое – процедуры каскадных обновлений связанных отношений типа АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ() могут выполняться автоматически в режиме «квази-реального времени». Под термином «квази» тут понимается задержка на выполнение всех необходимых операций обновления значительного числа данных и генерации значительного числа записей. Однако эти процедуры построены не на операциях соединения, а на менее ресурсоемких индексных пробежках по группам отфильтрованных записей. В дальнейшем, как и прежде, будем употреблять термин «реальное время», понимая тот факт, что современные вычислительные системы позволяют пренебрегать временем задержки на выполнение описанных процедур, если такая задержка не является критичной и это не оговорено особо.

Еще одной важной особенностью такого обновления отношения АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ() является то, что появление записей с новыми значениями части ключа *КодНаименАдрКатег* не нарушит реляционной целостности отношения. При этом никакого нового суррогатного ключа типа ID_Address, который в некаркасных схемах БД традиционно отвечает за такую целостность, не требуется. Это возможно потому, что в каркасной совокупности отношений все ключи автоматически обладают и целостностью, и полнотой, и единственностью, и семантичностью.

На описанном принципе может быть организовано целостное обновление любого ключа или неключевого атрибута. Это означает, что у пользователя каркасной БД имеется возможность моделировать

одновременное обновление всех задекларированных в метаданных групп записей и значений их атрибутов в группах отношений, связанных с редактируемой записью в текущем отношении. Тогда для оперативной обработки данных код актуальности играет роль фильтра. А для операций аналитической обработки этот атрибут не является определяющим.

Отметим, что иные каркасные отношения-справочники от атомарных и слабых сущностей-объектов фрагмента этой ПрО имеют вид: ПЕРЕЧЕНЬ ГОСУДАРСТВ (*КодГосуд*, Наименов, СокрНаим, КодАктНаимГос, ДатаАктНаимГос), СЛОВАРЬ НАИМЕНОВАНИЙ ОБЛАСТЕЙ (*КодНайменОбл*, Наименов, СокрНаим, КодАктНаимОбл, ДатаАктНаимОбл), СЛОВАРЬ НАИМЕНОВАНИЙ ГОРОДОВ (*КодНайменГор,* Наименов, СокрНаим, КодАктНаимГор, ДатаАктНаимГор), СЛОВАРЬ НАИМЕНОВАНИЙ РАЙОНОВ (*КодНайменРай,* Наименов, СокрНаим, КодАктНаимРай, ДатаАктНаимРай), ОБЛАСТИ В ГОСУДАРСТВАХ (*КодГосуд, КодОбл, КодНайменОбл*, КодАктОблГос, ДатаАктОблГос, Примечание), РАЙОНЫ В ОБЛАСТЯХ В ГОСУДАРСТВЕ (*КодГосуд, КодОбл, КодРайона, КодНайменРай,* КодАктРайГос, ДатаАктРайГос, Примечание), ГОРОДА В ГОСУДАРСТВАХ (*КодГосуд, КодОбл, КодРайона, КодГорода, КодНайменГор*, КодАктГроГос, ДатаАктГорГос, Примечание).

4.5.2. Каркасное хранилище данных

Рассмотрим представленную на рис. 4.2 классическую [423] схему ХД «снежинка», заимствованную из [196]. Представим ее в предикатной форме. При этом для атомарных сущностей-объектов учтем рекомендации большинства учебников по БД о присвоении не более чем унарного ключа [311, 312].

Однако заметим, что такой подход к назначению ключей отношений в данной схеме возможен лишь потому, что уникальной особенностью схемы «снежинка» является то, что любое отношение-измерение моделирует только или атомарные, или слабые сущности-объекты.

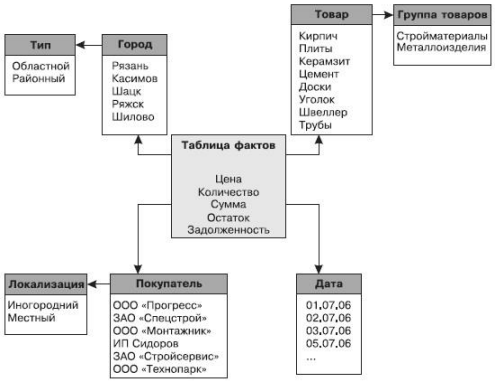


Рис. 4.2. Схема ХД «снежинка»

Имеем: ЛОКАЛИЗАЦИЯ (*КодЛокал*, Наименов), ПОКУПАТЕЛЬ (*КодПокуп*, Наименов), ДАТА (*НомГода*, *НамМес*, *НомДня*), ТИП ГОРОДА (*КодТипаГор*, Наименов), ГОРОД (*КодТипаГор, КодГор*, Наименов), ГРУППА ТОВАРОВ (*КодГруппы*, Наименов), ТОВАР (*КодГруппы*, *КодТовара*, Наименов), ФАКТЫ (*КодФакта*, Наименов, Значение).

Для получения итоговых документов необходимо посредством выполнения запросов к ХД осуществить соединения указанных отношений – некоторую совокупность декартовых произведений. Тогда интересующие пользователя отчеты будут иметь вид: ПОСТАВКА ТОВАРОВ В НЕКОТОРЫЙ ГОРОД ЗА ГРУППУ МЕСЯЦЕВ НЕКОТОРОГО ГОДА (*НомГода, НомМес, КатТипаГор, КодЛокал, КодГор, КодТовара, КодГрупТов,* Количество, НаСумму), ПОСТАВКА ТОВАРОВ НЕКТОРОМУ ПОКУПАТЕЛЮ ЗА ГРУППУ МЕСЯЦЕВ НЕКОТОРОГО ГОДА (*НомГода, НомМес, КодПокупат, КодТовара, КодГрупТов,* Количество, НаСумму), и т.д.

Тогда в общем виде отчет произвольной вложенности будет подобен отношению $R\text{EPORT}(X_1 + X_2 + \ldots + X_k, A_1 + A_2 + \ldots + A_n)$. Таким образом, на базе произвольной совокупности отношений-измерений классической схемы реляционного ХД «снежинка» всегда может быть получено произвольное отношение-факт, подобное актуальной ячейке РК [106]. При этом семантически неактуальные кортежи будут удалены. Это очень важное совпадение позволяет сделать следующие выводы.

1. Схема реляционного ХД типа «снежинка» является частным случаем схемы БД, полученной на РК.
2. Схема ХД может быть подобной в смысле [206] схеме оперативной БД, т.е. построена на РК – и оперативные, и аналитические данные могут храниться в БД с единой схемой и управляться приложениями, синтезированными единым каркасным подходом.
3. В современных условиях хранить данные в компактном виде нецелесообразно, так как скорость получения отчетов более важный фактор, нежели цена хранения.

4.6. Основные совпадения с современными тенденциями

4.6.1. Объектно-ориентированные БД

Проблема объединения в рамках единой модели данных свойств, присущих РМД и ООБД является актуальной на протяжении вот уже нескольких десятилетий [325, 368]. Описанная теория каркасной модели данных позволяет иначе решить эту проблему. Новая модель одновременно обладает SQL и OQL полнотой [299], позволяет проектировать информационные системы, имеющие объединенные свойства. А теорема о модифицируемой ДКНФ-схеме БД [208] дает возможность рассматривать КМД как единственно оптимальную платформу для такого объединения.

Идею, допускающую положительный результат такого объединения, высказал Дейт [368], где отношения реляционной модели можно рассматривать как объектные классы. Дейт утверждает, что "в реляционной модели не существует ничего, что может ограничить скалярные значения, существующие в доменах, … простыми формами. … Эти скалярные значения могут быть настолько сложными, насколько мы пожелаем".

В [113] указано, что под сущностью (объектом, фактом и т.п.) в ПрО понимается расширяемая совокупность атрибутов, объединенных единым уникальным в рамках этой ПрО многоместным предикатом. А под атрибутом сущности-объекта понимается некое абстрактное множество, объединенное одноместной частью этого уникального для ПрО многоместного предиката.

Там же описано одно из принципиальных свойств каркасной модели - фиксированная структура единого для каждого атрибута ключа, являющегося следствием многоместного предиката. Такой подход к декомпозиции ПрО дает возможность каждый атрибут ПрО рассматривать как отдельный класс сущностей-объектов со своим фиксированным классом ID, каждый экземпляр которого, во-первых, имеет строго фиксированную предопределенную порождающим предикатом схему, а во-вторых, уникально определяет экземпляр атрибута. При работе с SQL-запросами такая БД форматируется как реляционно-табличная – все группы атрибутов-столбцов в строгом соответствии со своими ID могут быть объединены в отношения. При выполнении OQL-запроса можно использовать механизм индексирования базы по структуре ID, что повышает скорость выполнения запроса.

Теоремой 2.10. об устойчивости РК к модификациям (о модифицируемости РК) рассмотрена классическая [254] задача о «неожиданном» дополнении сущности-объекта новым атрибутом, т. е., по сути, о модификации схемы БД [113, 231]. Автор [254] для этой проблемы определил красноречивую рубрику «смерть обоим», указывая на невозможность корректного моделирования ситуации ни в рамках классической реляционной модели [360], ни известным алгоритмом ООБД [38, 39]. Для каркасной же концепции подобные модификации являются естественным и корректным процессом, формально описанным теоремой, доказанной в [231].

Как указано в [113], модифицируемость схемы БД обеспечивается генерацией ключевых атрибутов по алгоритму сочетаний «всех со всеми», аналогично множеству всех подмножеств. Там же показано, что любые агрегированные сущности-объекты, замаскированные под атомарные, продуцируются конечным набором атомарных. Такой подход обобщает процедуру моделирования ООБД [38, 39], где любому искомому существительному (объекту) ставится в соответствие только атомарный

предикат (неструктурированный ID), вне зависимости от семантики ПрО - внутренних взаимосвязей. Поэтому каркасная модель значительно расширяет возможности ООБД. Составные сущности-объекты являют собой постсвязные многоместные предикаты. А слабые сущности-объекты – цепочки иерархически-зависимых многоместных предикатов. Каркас моделирует единой процедурой и тот, и иной тип.

Каркасный метод анализа ПрО дает подход более строгого моделирования сущностей-объектов. Как известно [38], теория проектирования ООБД не дает точных рекомендаций, какие сущности-объекты или факты из описания ПрО должны быть объектами, а какие являются лишь маскирующими категориями (словами). Важным является то обстоятельство, что такая методика применима к проектированию любого программного обеспечения – от приложений БД до произвольных кибернетических систем.

При таком подходе к проектированию приложений БД каждое отношения автоматически становится объектом. А их совокупность – иерархической структурой программной системы. Потому, что любое сечение каркаса – это иерархия, древовидный граф, в узлах которого расположены отношения. А связи по дугам графа моделируются ключами отношений. Каждое отношение моделирует сущность-объект.

Каркасный анализ позволяет разработчику приложений и программных комплексов обще-вычислительного характера, работающих не только с БД, проектировать структуру исходного текста программ и соответствующей библиотека объектов так, чтобы дальнейшее сопровождение и обязательные постоянные модификации не вызывали ощутимых затруднений и значительных затрат ресурсов. Использование совокупности объектов и процедур, построенных в соответствии с каркасной схемой семантики специализированной ПрО – стационарных динамических задач теории упругости [181], заложенное в листинг программного комплекса на начальных этапах (в 1990 году), позволило значительно расширить и

обобщить приложение [212, 219], а также решить вопрос его интероперабельности (для кластеров СКИТ-3 и «Инпарком-256») за очень короткий отрезок времени.

4.6.2. Денормализация

Из приведенной в [139, 140] подробной классификации наиболее схожими с проектированием безаномальных схем БД на реляционном каркасе является метод нисходящей и восходящей денормализации схем БД. А приведенный в [140] пример о включении в отношение-связь атрибута атомарной сущности-объекта, функционально зависимого только от части составного ключа, вносит не только постороннюю ФЗ и нарушает условия нормализации отношения-связи, но является чуждым исходя из семантики высказывания, моделируемого отношением-связью. В ней нарушается «тема» отношения.

Тем не менее, при такой денормализации пользователь получает дополнительное конкурентное преимущество эксплуатации БД, потому что скорость доступа к данным значительно повышается из-за исключения операций соединения, что и отмечается в работах [139, 140].

Каркасный синтез ДКНФ позволяет избегать денормализации, однако использовать нормализованные отношения-связи, которые также экономят операции соединения. Рассмотрим более подробно пример из [139]. Если в ПрО «ЛПУ» (*ПАЦИЕНТ, ТАЛОН, ПОЛ ПАЦИЕНТА*) регулярно синтезируются отчеты по неключевому атрибуту *Пол*, то может быть использован принцип "маска для роли атомарной сущности-объекта" [216] – отдельное отношения, которое является частичной копией атомарного отношения *ПАЦИЕНТ*. При этом формируется отношение-связь *ПОЛ ПАЦИЕНТА+ТАЛОН*, которое приложением БД поддерживается в целостном и актуальном состоянии универсальными процедурами. В нем также может сохраняться необходимая статистика – формироваться неключевые шунтирующие атрибуты.

И хотя такая декомпозиция отношения *ПАЦИЕНТ* на два отношения (*ПОЛ ПАЦИЕНТА* и, например, *ВОЗРАСТ ПАЦИЕНТА*) некоторыми авторами оценивается как денормализация [139], у Фейджина в [378] аналогичный прием приведен как пример нормализации схемы отношения: «…схема может быть разложена на две схемы… кортеж, содержащий данные о поле человека, входит в первое отношение, а кортежи, содержащие данные о его профессии, входят во второе отношение. Схема *SEX_ROLE* удовлетворяет ФЗ $Person \rightarrow Role$, в то же время схема *PROF_ROLE* не удовлетворяет этой зависимости. … Преимущество разложения состоит в том, что механизм обработки ключей системы может тогда предотвратить вставку двух ролей пола для одного человека».

При этом, как упоминалось в [139], тратится дополнительное время на поддержание ссылочной целостности. Однако эти потери, как и в [139], несоизмеримы с потерями при соединении.

Но если необходимо выполнить разовый отчет, создается запрос с операцией "выбор по выбору". Осуществляется выборка из отношения-связи *ТАЛОНЫ ПАЦИЕНТОВ* (соединения, в котором условно находятся миллионы записей) только тех ключевых значений атрибута *КодПациента*, для которых отфильтрована маленькая группа кортежей («пачка записей») в отношении *ПАЦИЕНТЫ* по любой совокупности вторичного ключа (скажем, по значению нескольких неключевых полей "*мальчики, младшие 12 лет*"). «Умные» СУБД выполнят эту операцию значительно быстрее прямого соединения, поскольку операция сводится лишь к выбору в «большом» отношении-связи отфильтрованной по первичному ключу (по индексу) группы кортежей в соответствии с малой группой «родительского» отношения. Однако, к сожалению, далеко не все СУБД предоставят такую возможность.

4.6.3. NoSQL - key-value БД

Современные модели БД типа «ключ-значение» (key-value – KV-модель), которые более всего схожи с каркасными отношениями, относятся к

новому, динамично развивающемуся направлению БД, объединенному красноречивым названием «не только SQL» (Not only SQL - NoSQL) [501]. Как правило, они представляют собой распределенные хеш-таблицы. Это самая сильная их сторона, т.к. предоставляет высокую производительность и модифицируемость. Тем не менее, эти свойства перестают быть преимуществами, когда необходимо обрабатывать списки.

В РМД же нижеследующие задачи являются типовыми:

- выбрать *k* последних покупателей,
- выбрать самые популярные товары текущей недели,
- найти склад, у которого максимальный оборот за последний месяц,
- найти покупателя по любому неключевому атрибуту (по номеру мобильного телефона, по адресу места жительства и т.п.)…

Но в KV-модели подобные задачи вызывают трудности. Например, когда необходимо осуществить выборку значений атрибутов одной сущности-объекта по вторичному ключу другой, ситуация становится схожей с подходом «нисходящей» или «восходящей» денормализации.

Таким образом, это - тот класс задач, для которых можно и нужно использовать РМД. На практике, поскольку такие задачи сильно взаимосвязаны, проектировщик передает БД в РМД-приложение. Потому что, как правило, применяется гибридное решение - комбинация KV-РМД. Все данные хранятся в KV, а те свойства, по которым необходимо осуществить агрегатные выборки, дублируются в РБД (с указателем на ключ). Но сложность заключается в том, что приходится следить за синхронизацией данных в двух БД, что сильно усложняет логику приложения.

Схема РБД, полученная на каркасной модели данных, автоматически обладает свойствами, подходящими и под модель NoSQL. Единственный структурированный ключ каждого отношения каркасной совокупности, полученный конъюнкцией одноместных ключевых предикатов (или, что то же самое, конкатенацией унарных ключевых атрибутов), являющийся

ключом отношения для SQL-запроса от приложения РМД, одновременно является еще и KV для соответствующих значений неключевых атрибутов при нетабличном запросе к БД от KV-приложения. А так же, как указывалось выше, еще и ID для OQL-запроса.

4.7. Выводы к четвертому разделу

1. КМД позволяет моделировать абстракции не выходя за рамки РМД .

2. КМД позволяет моделировать иерархии не выходя за рамки РМД .

3. Нормализация не входит в противоречия с потребностью моделирования концепций и понятий, не смотря на тот, что они в основном иерархической структуры

4. РМД не является противоречивой для моделирования темпоральности данных. Высокая гранулированность схемы БД не является проблемной, если существует механизм навигации по группам таблиц. КМД предоставляет такую возможность.

5. ХД и БД могу моделироваться на одной совокупности таблиц.

6. Приведенный алгоритм синтеза реляционного каркаса является алгоритмом анализа ПрО. И хотя данный алгоритм не претендует на единственность, своим существованием он показывает, что РК может быть использован не только для проектирования схем БД, а и для анализа более широкого класса ПрО.

РАЗДЕЛ 5
CASE-ОБОЛОЧКА НА РЕЛЯЦИОННОМ КАРКАСЕ

5.1. Введение к пятому разделу

Всесторонний анализ большинства типов ПрО [241] позволяет обнаружить в РМД возможность реализации двух разных подходов к решению задач пользователей. Это – механизм запросов к частично нормализованной схеме РБД (не выше БКНФ, а то и 3НФ [361]), а также механизм типизации большинства запросов, унификации основных алгоритмов, с ними связанных, и «упрятывания» этих алгоритмов в безаномальную схему БД [208].

Основной критерий применимости подходов – это коэффициент прогнозирования развития ПрО и разнообразия запросов пользователей (коэффициент прогнозирования запросов) [241]. Там под таким коэффициентом понимается отношение числа подтвержденных изменений к суммарному числу прогнозированных и спонтанных изменений за определенный период времени, например, год. Очевидно, что в такой формулировке число подтвержденных изменений не может превышать числа прогнозируемых.

Показательно, что удовлетворительной для второго подхода является ПрО с коэффициентом, равным единице. Это те ПрО, которые можно изучить, развить и прогнозировать. Приложения, моделирующие процессы ПрО, могут иметь минимальный интерфейс для неподготовленного пользователя, позволяющий минимизировать (или практически исключить) использование механизма внешних запросов к БД. К таким ПрО можно отнести всевозможные бизнес-приложения компаний и корпораций, которые развиваются в соответствии с прогнозируемыми рыночными факторами.

Что же касается ПрО с малым коэффициентом прогнозирования запросов, как, например, поисковые машины в Интернет, всевозможные социальные сети, Интернет-витрины данных, схемы БД которых зависят не от собственников систем и причинно-следственных связей ПрО, а от

хаотически обращающихся пользователей, задачи эффективно моделируются гибкими языковыми конструкциями.

5.2. Постановка задачи

Реляционный каркас позволяет представить любое приложение как среду управления данными с заданной целью. Но поскольку для обработки данных необходимо использовать ту или иную модель, наиболее привлекательной для указанных целей является именно РМД. Каркас же является частным случаем РМД. Причем, как показано, например, в [216], одним из важных свойств РК является возможность минимизировать объем запросов к БД, построенных на громоздких и вычислительно сложных операциях соединения.

Как показывают экспериментальные исследования, проведенные на значительном числе примеров [216], каркасная БД моделирует до 90% запросов без операции соединения и ее модификаций. Это позволяет значительную часть данных обрабатывать по заранее сформированным индексным таблицам. И тем самым существенно снизить объем вычислений. А также подавляющее большинство запросов пользователей формализовать, унифицировать и интегрировать в приложение.

Этот подход позволяет построить универсальную перенастраиваемую оболочку, управляемую группой метаданных. А массивы метаданных, отражающие ту или иную специфику разнообразных ПрО, создавать с помощью программной оболочки, называемой в дальнейшем инсталлятор.

Для проектирования инструментального средства, будущих приложений БД или иных вычислительных систем, связанных с БД лишь косвенно, имеем единую схему каждого отношения – схему, построенную на этимологии сущности-объекта [211]. То есть, по сути, на строковой конкатенации унарных атрибутов типа $A+B+C+D$ $...+Z…$ Известно, что для такой конструкции может быть построены элементарные индексные деревья. И именно такой механизм проектирования может быть унифицирован вплоть до стандартизации.

5.3. CASE-оболочка генерации метаданных

Основная технологическая особенность КМД сводится к пакету отношений, связанных между собой по индексированным ключевым атрибутам. Однако поскольку значительная часть каждого кортежа состоит именно из ключевых атрибутов (что является важным отличительным признаком каркасной БД), такими иерархическими связями пронизана вся проекция каркаса (совокупность актуальных отношений) на данную ПрО.

Это означает, что разрабатываемая новая CASE-оболочка - это не что иное, как генератор мета-отношений, в каждом из которых унифицированным способом хранятся метаданные конкретной ПрО: имена пользовательских отношений, имена их атрибутов (в дальнейшем в этом разделе – полей), группы индексных заголовков, иные специализированные данные .

Принципиально важным признаком каркасной группы отношений является отличительный элемент: жесткая взаимосвязь кортежа из верхнего отношения с группой кортежей нижнего отношения, которая осуществляется по индексированному общему ключевому атрибуту. По сути, именно к такому атомарному сочетанию сводится вся совокупность групп отношений.

Рассмотрим любое отношение, построенное в соответствии с КМД. В общем случае, его схема сводится к схеме особого отношения $R(X_i, A_{ij})$, где атрибуты и их подстрочные индексы имеет тот же смысл, что и в [208]. Связь верхнего кортежа и группы нижних кортежей осуществляется по индексированному общему ключевому атрибуту X_i. Причем, если A_{ij} - пустое множество и отношение не шунтировано, то имеет место аномалия типа ДЗ (МЗ, ЗПС). Или, что аналогично, отношение моделирует связь, которая не актуальна в этой ПрО в данный момент времени.

Подчеркнем, такая связь может быть лишь временно не актуальна. В любой момент эксплуатации текущего состояния схемы БД может возникнуть потребность в актуализации некоторой связи. У этой связи появятся атрибуты, т.е. отношение станет безаномальным. Это говорит о том,

что CASE-оболочка должна предоставлять пользователю возможность «на всякий случай» формировать всю полноту отношений. Выбор определяется лишь спецификой ПрО и проектировщиком.

При этом любое отношение для любого пункта меню может быть использовано в качестве так называемого «центра активности» приложения. Под центром активности понимается одно или группа отношений, в которые одним или группой уполномоченных пользователей осуществляется ввод новых или редактирование существующих данных. Эти две разновидности обработки данных завершаются аналогично.

В момент завершения такой операции, по принципу on-line-транзакции, осуществляются все необходимые процедуры: обновление всех ключевых атрибутов актуальной совокупности отношений, связанных с фоновым центром активности (за экраном), отслеживание целостности всех соответствующих по ключевым полям отношений, формирование всех необходимых итоговых полей, обновление журнала транзакций, перегруппировка блокировок и пользовательских буферов и т.п.

На описанном принципе разработано и всесторонне апробировано инструментальное средство - CASE-оболочка SWS (от Server Workstation System) [221]. Инструментальная средство SWS поддерживает бесконечно много центров активности. Однако в одном конкретном пункте меню интерфейса пользователя синтезируемого приложения представлен единственный центр активности. Именно пункт меню предоставляет пользователю его полномочия – на ввод новых данных, на их обновление, на просмотр и вывод на печать, на иные действия. Именно центр активности – это формализованный унифицированный запрос к серверу БД, который проектировщик подготовил после обследования ПрО.

Известно, что ввод или обновление данных является ответственным действием. Рассмотрим именно эту операцию более подробно. По факту завершения редактирования текущего поля одного из отношений приложения, синтезированного на CASE-оболочке SWS, осуществляется вся

совокупность унифицированных и дополнительно задекларированных пользователем действий. Причем осуществляется только по принципу on-line-транзакций. В эту совокупность действий входит: обновление всех участвующих в связи группы отношений ключевых полей (отслеживание их целостности), генерация всех новых кортежей во всех группах фоновых отношений, обновление индексов, формирование задекларированных в CASE-оболочке суммирований, иных хранимых пользовательских процедур или триггеров и т.п.

В такой процедуре центр активности – одно отношение схемы БД или совокупность отношений, связанных между собой в одно. Все остальные совокупности отношений распределяются на две группы – головные отношения или подчиненные.

Это означает, что любому связанному по достаточной совокупности ключевых полей кортежу головного отношения соответствует текущая группа кортежей активного (то есть, находящегося в этот момент в редактируемом пользователем состоянии) подчиненного отношения. В зависимости от специфики ПрО к любому из подчиненных отношений связь с центром активности может осуществляться как степенью $1:H$, так и $G:H$. Именно к такой паре «отношение-отношение» в конечном итоге сводится любой фрагмент ПрО. Поэтому унификация каскадных процедур навигации также не является сложной.

Приведем краткое описание основных принципов синтеза приложений БД, используемых в CASE-оболочке SWS.

5.4. Схема мета-БД и компоненты CASE-оболочки SWS

CASE-оболочка – инструментальная система SWS – это два модуля: инсталлятор, посредством которого формируется мета-БД таблиц-сценариев работы приложения, и непосредственно CASE-оболочка, интерпретирующая таблицы мета-БД, которая является унифицированным прототипом любого приложения.

Схема каркасной диаграммы инсталлятора, приведенная на рис. 5.1, представляет собой традиционную иерархию меню. На первом шаге проектирования приложения система обеспечивает выбор принципа работы со списками проектируемых отношений: локальные отношения, общие отношения или слияние схем БД разных ПрО.

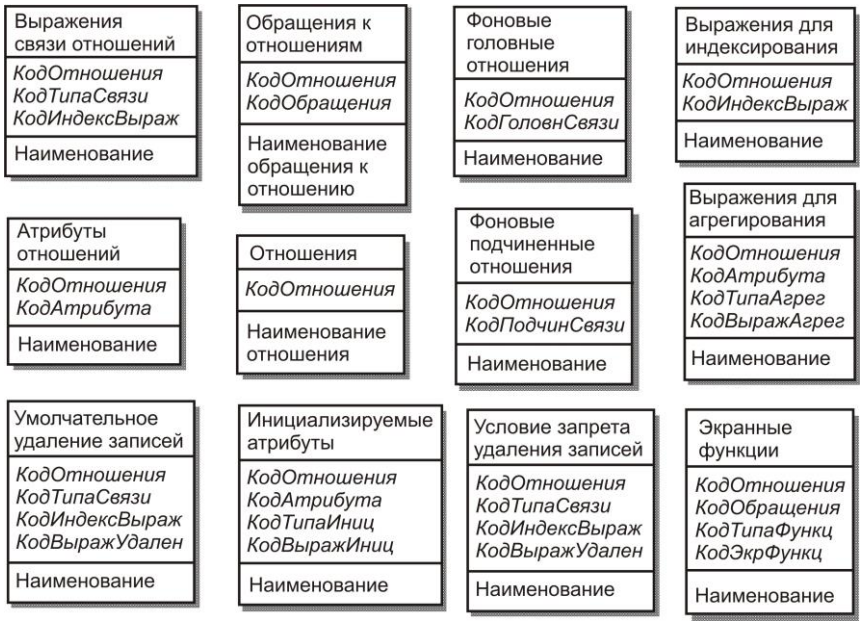


Рис. 5.1. Каркасная схема SWS -инсталлятора

В режиме «локальные отношения» производится настройка всей группы проектируемых отношений БД, которые обслуживаются только модулем приложения локально. Такие отношения будут недоступны при настройке другого модуля создаваемой подсистемы, в отличие от отношений БД, описываемых в режиме инсталлятора «общие отношения».

Определенный при первом конфигурировании модуля приложения вид работы с отношениями можно изменять, активизировав элемент подменю «режим работы с отношениями». Перестройка вида работы всегда возможна «снизу вверх». Также возможна полная переконфигурация (вплоть до режима «без отношений БД»). Но такой выбор будет сопровождаться отключением уже существующих реляционных связей.

5.5. Основы конфигурирования приложения

Основная работа по созданию комплексов приложений, производимая в среде инсталлятора CASE-оболочки - это конфигурирование схемы БД и

создание интерфейса пользователя, управляющей среды «меню-подменю» любого из отдельных обслуживающих эту БД приложений. Синтез схемы БД обеспечивается инсталлятором после выбора режима работы с отношениями – только как с монопольными отношениями, только как с общими или по смешанному типу. Сценарий функционирования приложения составляется пользователем посредством раздела «Меню».

Две установочные ветви традиционно логически связаны - каждая из них в отдельности позволяет строить иерархические структуры и для схемы БД, и для одного из приложений. Кроме того инсталлятор позволяет задавать специальные перекрестные графовые связи между элементами деревьев меню-подменю приложения и элементами деревьев отношений БД (как монопольных, так и общих).

Эти два режима инсталлятора - синтез схемы БД и описание конфигурации приложения - самые трудоемкие. Поэтому их тесная функциональная взаимосвязь требует частого перехода из одного режима в другой. С этой целью система естественно поддерживает подсказки взаимных данных.

5.5.1 Настройка БД: общие или локальные отношения

Режимы «Локальные отношения» и «Общедоступные отношения» обеспечивают равнозначную возможность конфигурирования схемы БД в этих двух режимах. Разделение этих режимов говорит лишь о принадлежности отношений - либо к локальной подсхеме данного модуля, либо к общедоступной подсхеме БД для всех модулей создаваемого приложения.

Можно было бы строить подсистемы обработки БД только с общедоступными отношениями. Однако разумное распределение данных, с применением локальных отношений, повышает эффективность функционирования сетевых комплексов.

В колонке «Имя отношения» содержится список соответствующих имен, в строке напротив имени отношения в колонке «Структура»

обеспечивается вход в табличный редактор структуры данного отношения, в колонке «Индексы» активизируется редактирование списка индексных выражений, приводящее впоследствии к созданию необходимых индексных таблиц. И в колонке «Обращения» обеспечивается редактирование описаний табличных форм ввода/вывода данных (предварительно настраиваемых пользовательских запросов), а также связей данного отношения с другими отношениями модуля.

Связи отношений строятся на основе их индексных выражений. А к записанным в определенном порядке связям и табличным формам (обращениям) впоследствии адресуются определенные элементы подменю модулей - из общедоступного списка отношений к обращениям могут адресоваться все модули комплекса. А к обращениям из локального списка отношений может адресоваться только данный модуль-владелец этого списка.

Необходимая активная индексная таблица, обеспечивающая связи между отношениями и быстрые поиски, также фиксируется в режиме настройки подменю модуля.

5.5.2. Имя отношения

При задании имен отношений БД система подразумевает следующие соглашения. Имена отношений, как общедоступных всем модулям, так и локальных, должны быть уникальны в разрезе всей БД. Если в соответствующем рабочем каталоге, например для локальных отношений при редактировании этого списка существуют сторонние (заимствованные из других приложений) отношения БД, имеется возможность перечитать их имена, включив, таким образом, эти таблицы в общий список БД разрабатываемой подсистемы.

5.5.3. Структура

Активизация колонки редактирования структуры определенного отношения БД позволяет сформировать или изменить структуру отношения. В колонке «Имя поля» производится набор списка полей отношения. В

колонке «Тип» описываются их типы, в колонках «Длина» и «Дес.» описываются размерности полей.

Имеются соглашения на имена полей - может состоять из не более чем 100 символов латинского алфавита (от A до Z), символа подчерка и символов цифр (от 0 до 9), причем символ цифры не может быть первым (ведущим) символом в имени поля.

При наборе структуры в среде инсталлятора действует простейший, но эффективный входной контроль, так что поля соответствующих типов могут получить только определенные размерности или размерности, входящие в определенный диапазон (например, максимальная длина строкового поля - не более 1000 символов).

Колонка «Дес.» актуальна только для полей типа «N» (числовых). Традиционно, размерность при значениях «Длина» - 5 и «Дес.» - 2, определяет структуру поля, позволяющего содержать максимальное число значением 99.99 (разделитель десятичной части - точка, и символ точки учитывается в числе определителя длины числового поля).

Повторно считывание структуры определенного отношения приводит к переопределению редактируемой таблицы в соответствии с уже существующей структурой данного отношения. Таким способом легко редактировать вновь включенные в список БД сторонние отношения. При этом содержимое такого отношения сохраняется.

5.5.4. Индексы

Очевидно, что самым важным механизмом поддержки целостности каркасной БД является индексный поиск. Поэтому колонка «индексы» является наиболее часто используемой. Она предназначена для заполнения списков выражений индексации отношений, что впоследствии приведет к созданию такого же количества индексных таблиц, позволяющих отслеживать структурные связи, а также производить необходимые быстрые поиски совокупностей записей в группах отношений.

Очевидным соглашением системы SWS является требование только строкового типа для выражения индексации. В простейшем случае такое выражение индексации отношения может быть записано как имя ключевого поля. В более сложных выражениях индексации применяется языковые операции, например "+" - конкатенация строковых выражений. Причем каждая часть выражения должна или иметь строковый тип, или должна быть приводимой определенной встроенной функцией системы или стандартной SQL-функцией к строковому типу.

Пример выражения индексации на языке типа dBase: *F1+DTOS*(*F2*). В этом выражении, кроме операции конкатенации, применена функция DTOS(), позволяющая преобразовать значение атрибута отношения (т.е., поля) F2 (типа "D") в строку вида "ГГГГММДД". Очевидно, что данное действие позволяет отсортировать записи отношения в хронологическом порядке.

Традиционно, результирующая маска индексированного поиска, определяющая позиционное содержимое индексной таблицы для текущей записи, может, например, выглядеть так: "XXX20120321", где "XXX" находится в поле F1 текущей записи, а поле F2 содержит значение даты 21 марта 2012 года. Кроме того, выражения индексации включаемых активных индексных таблиц позволят организовывать связи по ведущим ключам. Т.е. в SWS полностью отслежена специфика функционирования реляционных СУБД.

Еще одной важной особенностью SWS является отслеживание формата индексации отношения: $A \rightarrow (A+B) \rightarrow (A+B+C)$. Она полностью совпадает с концепцией предикатной этимологии смысла сущностей-объектов произвольной ПрО, обоснованной в рамках КМД [211].

Приведенный абстрактный пример расшифровывается следующим образом. Постановка задачи требует трехсвязанной табличной иерархии. Первое отношение индексировано по ключевому полю A. Второе - по ключевым полям A и B. Третье - по ключам A, B и C. Это позволило

представить для обработки только иерархически взаимосвязанные данные, принадлежащие только уникальным выбранным ключам. Причем, записи иерархически старших отношений «ведут» пакеты записей подчиненных (нижних, ведомых) отношений. Эта концепция системы SWS продиктована спецификой КМД, лежащей в ее основе и обоснованной ранее. А практика применения этой концепции станет более понятна при разборе рассматриваемых ниже «псевдофильтров» в «обращениях» к отношениям.

5.5.5. Обращения к отношениям

При последовательном описании окружения любого из отношений БД на завершающем этапе производится конфигурирование унифицированных запросов к фрагменту каркасной БД, озаглавленное в инсталляторе как «обращения». Приложения функционируют под управлением файлов мета-БД, т.е. сценариев установок, составленных в среде инсталлятора.

Листья ветви меню-подменю приложения, озаглавленные как «выводимые отношения», фиксируют графовые связи к листьям ветви конфигурирования БД, озаглавленных как «обращения». В определенных элементах подменю приложений происходят обращения к стандартно описываемым табличным формам ввода-вывода данных. Обращение заполняется с тем, чтобы ввести в мета-БД приложения данные о том, как в определенном активизированном элементе подменю представить экранную таблицу ввода-вывода данных. И как эти данные должны быть взаимосвязаны.

5.5.6. Выражение связи

В заголовке данной колонки инсталлятора, (как и многих других), куда могут быть внесены выражения, запрашивается уточнение типа, возвращаемого при выполнении внесенного выражения. Для обеспечения межтабличной связи или получения «вырезки» из отношения необходимо записать такое выражение строкового типа, которое позволит задействовать необходимое активное выражение индексации.

На получение необходимого «пакета» записей в проектируемой форме ввода-вывода оказывают влияние три установочных фактора: запись нужного выражения индексации таблицы, выбор необходимого активного выражения индексации в режиме инсталлятора «открываемые отношения» и фильтрующее выражение с учетом выражения индексации активной индексной таблицы

В подменю приложения, где фиксируются открываемые отношения, предполагается к включению индексная строка, содержащая ранее записанное выражение индексации вида A_1. Выражение связи, называемое «псевдофильтром», записываем, например, как выражение строковой константы "00". Тогда при активизации данного элемента подменю приложение сформирует таблицу «ввода-вывода» данных, где будут «видны» только те записи, в которых (согласно примеру) в поле A_1 содержатся данные, начинающиеся с символьной цепочки вида "00", такие как "00AAA".

Таким образом, псевдофильтр – унифицированный на уровне создаваемого модуля приложения запрос к данной таблице, построенный в соответствии с механизмами КМД и спроектированный не пользователем при эксплуатации, а разработчиком предварительно. В этом принципиальное отличие инструментальной CASE-оболочки SWS от аналогичных систем.

Концепция построения взаимосвязанных отношений в CASE-оболочке SWS, продиктованная КМД, требует определенного порядка внесения в метаданные выражений связи. Если предполагается необходимость иерархической структуры из двух отношений - R_1 и R_2, у которых ключевые индексные выражения имеют вид X и $X+Y$, то это означает, что запись верхнего отношения R_1 с ключевым полем X (содержащим определенное уникальное значение) должна отслеживать связь со всеми записями нижнего (ведомого отношения R_2), каждая из которых также содержит соответствующее значение в своем ключевом поле X.

В обращении к верхнему отношению R_1 (с выражением индексации X) выражение связи может отсутствовать, а в соответствующем для общей

таблицы обращении к нижнему отношению R_2 (с выражением индексации $X+Y$) следует записать выражение связи вида $R_1 \to X$.

Нотация $R_1 \to X$ в некоторых языках программирования читается как «алиас» поля X из отношения R_1. Означает, что текущее значение поля X необходимо взять из отношения R_1.

В данном примере, связь осуществляется из другого отношения, поэтому выражение связи указывается символом алиаса $R_1 \to$. Сортировка, требование уникальности значений или предполагаемая подобная связь с третьим отношением структуры в нижнем отношении R_2 определяется включением в выражение индексации поля Y.

При использовании сценария из примера настроенный модуль приложения выведет на экран монитора или на печать табл. 5.2. Первое отношение, приведенное в табл. 5.2, имеет схему $R_1(X, A)$, является ведущим и упорядочено по значению ключевого поля $R_1 \to X$ («Номер комплекта»). Второе поле отношения $R_1 \to A$ содержит наименования комплектов. В таблице приведено абстрактное наименование. В промышленных БД такое наименование, как правило, имеет значительное число символов.

Таблица 5.2.

Взаимосвязь двух отношений R_1 и R_2

Номер комплекта	Наименование комплекта
006	Комплект № 6

Номер комплекта	Номер комплектующих	Наименование комплектующих
006	01	Первая комплектующая комплекта № 6
006	02	Вторая комплектующая комплекта № 6
006	03	Третья комплектующая комплекта № 6
006	04	Четвертая комплектующая комплекта № 6
006	05	Пятая комплектующая комплекта № 6

Второе, подчиненное отношение имеет схему $R_2(X,Y,B)$, является ведомым и упорядочено по значениям ключевых полей $R_2 \to X$ («Номер комплекта») и $R_2 \to Y$ («Номер комплектующих»). Третье поле отношения $R_2 \to B$ содержит наименования некоторых комплектующих, каждое из которых являющихся частью соответствующих комплектов из ведущего отношения R_1

Из примера видно, что выводимая на экран текущая запись из верхнего отношения R_1 со значением ключевого поля 006 фильтрует из подчиненного отношения R_2 только те записи, которые принадлежат данному коду. Связующее поле $R_1 \to X$ в таких таблицах на экран или печать, как правило, не выводится, так как служит лишь для связи с верхним отношением. Во всей совокупности записей нижнего отношения, связанных с единственной записью верхнего отношения, значение связующего поля является тождественным (в приведенном примере 006).

Поля $R_1 \to A$ и $R_2 \to B$ - информативные, являются традиционной смысловой расшифровкой кодов. Конечный пользователь (оператор) работает в основном с кодами, организуя рабочую последовательность вида «код - данные». Экранные таблицы и выходные документы модуль формирует с использованием кодов записи и смысловых полей.

Таким образом, выражение связи применяется для организации структуры таблиц данных типа «дерево». Каждое отношение, как очевидно из теории КМД – это в общем случае связь. Поэтому система SWS поддерживает иерархическую совокупность отношений-связей, т.е., актуальный шунтированный фрагмент каркаса, моделирующий функционирование конкретной ПрО.

5.5.7. Фоновые головные отношения

В CASE-оболочке SWS проектировщику предоставляется возможность построения каркасной схемы БД, когда дерево связанных отношений БД строится «от листьев к корню» в соответствии с шунтированным фрагментом

каркаса, моделирующим функционирование данной ПрО. Для этого в корневом окне редактирования обращения к отношению для одинаковых значений ключевых полей (в соответствующих записях совокупности отношений) обеспечивается формирование уникального кортежа в каждом фоновом (скрытом за экраном) отношении. При использовании в одном из элементов меню приложения такого обращения формируется on-line-сценарий обработки группы отношений построением иерархической цепочки «снизу вверх».

Другие обращения для других элементов подменю могут обеспечить возврат на экран и печать таких структурированных данных. Чтобы запросить отношение R_1 как фоновое головное для отношения R_2, необходимо следующее. В колонке инсталлятора «фоновые головные отношения», обеспечивающей окно редактирования списка фоновых головных отношений, записывается имя отношения R_1, в котором ключ X должен быть уникальным и ведущим для пакета соответствующих кортежей в отношении R_2 . Выражение индексации отношения R_1 - указание имени ключевого поля X. Выражение связи из текущего отношения записывается тоже как X. В колонке «инициализируемые поля» вызывается редактор списка полей, и для поля X отношения R_1 инициализирующим будет также поле X, но взятое из отношения R_2.

В завершение построения в одном из подменю создаваемого приложения как открываемые заказываются оба отношения - и R_1, и R_2 . Но как выводимое на экран (в мониторинговую таблицу) заказывается только отношение R_2. В дальнейшем вся созданная таким образом конструкция функционирует в определенном подменю приложения следующим образом: на экране приложения показывается таблица для просмотра или редактирования отношения R_2, где в произвольном порядке можно вносить неуникальные значения для поля X, которое для R_2 не является полным ключом . Ввод каждого нового значения X приводит к созданию новой

записи в фоновом отношении R_1. Ключ X в этом отношении получает соответствующее уникальное значение.

Если при наборе в поле X отношения R_2 вводится значение, для которого уже создана ведущая запись в фоновом головном отношении R_1, новая запись в нем не добавляется. При этом происходит внутреннее позиционирование на запись с соответствующим ключом. Таким образом, создается схема БД, строго соответствующая фрагментам реляционного каркаса – совокупности групп данных по разделам ПрО. Такую схему легко применить для on-line-накопления расчетных данных.

Существует еще один способ использования фоновых головных отношений. Ранее описанные отношения-справочники не имеют жесткой индексированной связи с текущим отношением приложения. Поэтому для динамического связывания такое отношение-справочник следует объявить как фоновое головное отношение, указав соответствующее выражение связи с ним. Появляется возможность в колонках текущего отношения для определенного кода выводить на экран или печать расширенную информацию (например, наименование, иные атрибуты), но взятую из другого отношения – некоторого связанного отношения-справочника, что реализуется заказом непосредственного вывода соответствующих полей с указанием справочного отношения.

При создании такого фрагмента сценария работы приложения, инициализируемые поля фонового головного отношения и выражения для их инициализации заказывать не нужно, поскольку соответствующие справочные записи должны существовать. Но если пользователь приложения имеет право обновлять этот справочник самостоятельно, выражения инициализации заказываются обязательно. В такой связи набор кортежей в справочном отношении первичен.

5.5.8. Фоновые подчиненные отношения

Когда в обращении приложения, который сформирован унифицированными соглашениями SWS и транслирует некоторый

унифицированный запрос, предполагается редактирование какого-либо отношения, для которого есть иерархически подчиненные отношения, не выводимые на экран, тогда такие подчиненные отношения должны быть объявлены в инсталляторе как фоновые подчиненные.

Смысл такого конфигурирования заключается в том, что при изменении какого-либо ключевого значения в ведущем (активном) отношении все соответствующие значения в подчиненных, ведомых, отношениях также будут синхронно сменены с сохранением корректной ключевой связи данных. Порядок заполнения колонок в данном окне инсталляционного редактора тот же, что и для фоновых головных отношений. Для подменю приложения, естественно, такие фоновые подчиненные отношения должны быть объявлены как открываемые.

5.6. Основной перечень каскадных функций SWS

Анализ ПрО проектирования инструментального средства позволил сформировать список основных унифицированных каскадных функций. К ним относятся: инициализация и редактирование ключевых полей; проверка выполнения условий обязательности ввода или запрета на ввод; проверка условия целостности вводимого значения (минимум-максимум, положительное, уникальное и т.п.); установка полю статуса поискового и статуса выбираемого значения; установка связи для выбора значения в ином отношении (по полю, с фильтром, с показом поля, какое выражение связи по индексу и т.п.); условие, приводящее к замене значения одного поля при вводе и редактировании текущего (список заменяемых полей); условие генерации «сжатых» отчетов; условие каскадного удаления и т.п. Очевидно, что данный список не может быть полным.

5.6.1. Открываемые отношения

Открываемые отношения БД, записанные в данном редакционном окне Инсталлятора, необходимы при использовании режимов STD и SPEC-I. Гарантируется открытие отношений БД одновременно со всеми их индексными файлами, причем текущим индексным файлом явится

выбранный в колонке "№ главного ордера". Именно в таком порядке будут отсортированы кортежи открытого отношения БД. Именно определяемые показанным выражением главного ордера связи возможны для данного отношения. Если в каких-либо выражениях, записываемых, например, в «Формах обращения», производится переключение такого главного ордера для какой-либо специфической обработки, следует при окончании цепочки обработки вернуть значение данного главного ордера. И еще один важный момент - если производится модификация какого-либо стороннего открытого отношения в таком цепочечном выражении, модификацию следует предварять встроенной функцией MODIFY ("Имя_модифицируемого_файла"). Это необходимо для правильного отслеживания транзакций системой SWS.

5.6.2 Выводимые отношения

Порядок записи выводимых в STD-таблицу отношений обеспечивает порядок их вывода на экран – первое отношение будет располагать кортеж в верхней строке таблицы, второе - во второй и т.д. Такой формат следования кортежей и обеспечивает иерархическое представление данных в таблице обработки, что строго соответствует любому фрагменту реляционного каркаса. Обозначение индексных выражений и видов связи строго соответствует этимологии смысла сущностей-объектов, которые моделируются каждым отдельным отношением. Они имеют вид: A; A+B; A+B+C... Выбранный ранее номер обращения обеспечивает вид таблицы на мониторе, вид связи и прочие установки, касающиеся ранее описанных сценарийных настроек для режима инсталляции "Обращения STD". Точка выбора необходимого обращения и является замыкающей для полной записи сценария, обеспечивающей графовые связи для будущих обработок данных. Здесь существует определенная опасность появления труднораспознаваемой ошибки - при смене номера обращения или переупорядочивании обращений определенного отношения. Если производится переупорядочивание обращений, для исключения возможной ошибки в несоответствии обращения

и настроек следует синхронно произвести переназначение номера обращения в выводимом отношении.

5.6.3. Формы обращения

Раздел «Формы обращения» позволяет определить полный (при единичном отношении) или частичный (при многосвязанных отношениях), выводимый модулем на экран монитора вид таблицы строящегося модуля. При активизации для редактирования данной колонки, если этот режим активизирован впервые, необходимо перечитать полную структуру данного отношения посредством нажатия клавиши F5.

Благодаря этому появляется возможность выбора необходимых полей для вывода в таблице, определяемой данным обращением. Заполнение колонки "Заголовок колонки" является условием, которое позволяет включить текущее поле в будущую таблицу, которую модуль построит на основе данного обращения. При этом строка в окне инсталляции изменяет свой цвет. Порядок расположения таких ярко засвеченных строк в таблице Инсталлятора определит порядок расположения колонок в таблице ввода, редактирования или просмотра (в зависимости от указанного режима, о чем будет сказано далее).

При заполнении «Заголовка колонки» ведущие и завершающие пробелы всегда будут отсекаться Инсталлятором - такова специфика редактора. Если появится необходимость вывода «удлиненного» заголовка, можно в начале и в конце вводимой строки заголовка поставить какой-либо символ, отличный от пробела, например, символ точки.

Колонка "Поле, функция, переменная" содержит выражение, значение которого будет выводиться (или будет представлено для ввода данных, если это поле отношения и выбран соответствующий режим) в таблице настроенного модуля, в ее колонках для каждой записи соответственно. В основном здесь записываются имена полей, как с применением алиасов, так и без таковых. Если для данного обращения необходимо вывести поле другого, не текущего отношения, указание алиаса - обязательное условие.

Выражение, которое не является просто полем одного из отношений, необходимо записывать с применением круглых скобок. Если это функция, скобки само собой присутствуют, а если это выражение, например, суммы полей отношений, то такое выражение необходимо взять в скобки. Такие выражения модулем трактуются как только выводимые (т.е. в них, естественно, отсутствует возможность занесения каких-либо значений в режиме ввода или корректировки). Мощный способ - вывод выражений - позволяет динамически заполнять определенные колонки таблицы модуля, сэкономив при этом дисковое пространство за счет меньшего количества полей в отношениях БД. Если нет необходимости хранения таких динамически получаемых данных (обычно типа "N"), пользователь имеет возможность записывать их как выводимые выражения. Примером может служить такая строка в окне инсталляции: "Заголовок колонки" - Сумма; "Поле, ф-я, перем" - (KOL_VO*CENA).

Колонка "Длина" определяет ширину каждой колонки в таблице экранной формы ввода/вывода. Ее можно определить меньшей, чем длина поля (в режиме печати бывает необходимым такое отсечение). А колонки для строковых полей большой длины удобнее сжать по ширине, одновременно указав в колонке "PICTURE" функцию вертикальной прокрутки значения поля.

Колонка "PICTURE" определяет способ ввода/вывода поля. Здесь записывается строковое выражение форматирующей маски. Форматирующая маска может содержать непосредственно отображаемые/заносимые символы, может специальными символами осуществлять фильтрацию вводимых символов (например, только цифры при шаблоне вида "999"), а также может включать в себя функцию форматирования. Функция форматирования предваряется знаком @ (коммерческое АТ). Если присутствует строка функции и строка непосредственного шаблона, строка функции записывается первой и строка шаблона должна быть отделена одним пробелом. Полное

выражение форматирования в колонке "PICTURE" обязательно должно быть заключено в одинарные или двойные кавычки.

5.6.4. Выбор данных в группах отношений

Эта и последующие пять колонок обеспечивают настройку интерфейса со справочными отношениями, когда обеспечивается входной контроль вводимых данных по существующим ранее набранным кодированным ключевым полям, выбранным из других отношений. В этой опции и заключено основное свойство доменно-ключевой схемы данных, означающее зависимость ключевых полей из более старших отношений только от доменов и ключей младших отношений. Как показано в теории реляционного каркаса, если схема БД моделируемой предметной области построена в строгом соответствии с шунтированной частью каркаса, совокупность этих отношений является замкнутой с точки зрения зависимостей ключей и доменов этих ключей. Поэтому осуществить выбор полей более старших отношений из младших – это в своем роде концептуальное обязательство системы.

Правильно составленная схема БД – каркасная ДКНФ, когда проектировщик точно и однозначно расшифровал этимологию смысла всех сущностей-объектов и перевел ее в совокупность ключевых полей ДКНФ - совокупности отношений - предусматривает введение редко изменяющихся (условно постоянных) данных в отдельные отношения, моделирующие атомарные сущности-объекты ПрО. В обиходе проектировщиками их принято называть «справочниками». Такие отношения индексированы по ключевым полям, что обеспечивает работу оператора выбором наборов данных для определенного кода. Контроль на существование данного кода и обеспечивают описываемые выше шесть инсталляционных колонок. А заполнение колонки "Выбираемое" выражением, возвращающим "истину", обеспечивает активизацию всей цепи справочного контроля в будущем окне таблицы набора данных, которое модуль построит на основании данного обращения к отношению.

Для правильной работы справочника в окне таблицы модуля, заданном при заполнении текущего обращения, необходимы следующие условия:

1) отношение-справочник должно объявляться как открываемое в настройке "Подменю"/"Открываемые отношения";

2) для него должен быть включен индексный ордер с выражением индексации, где кодовое поле записано первым, если не заявлена константа/поле псевдофильтра;

3) тип и размерность контролируемого поля в структуре текущего отношения, а также выражение "PICTURE" в заполняемом текущем обращении должны строго соответствовать как типу и размерности поля в структуре справочника, так и выражению "PICTURE" в обращении для обеспечения набора справочника.

Кроме того, должны быть заполнены последующие описываемые ниже колонки окна редактора Инсталлятора.

Выбрать в отношении. Нажатие клавиши <Enter> в данной колонке включает окно выбора необходимого отношения-справочника. Это выбранное отношение впоследствии, в определенном настраиваемом при инсталляции элементе подменю модуля, должно быть объявлено как открываемое. Выбор производится и фиксируется как обычно в редакторе Инсталлятора, где задействован этот интерфейс.

Редактирование обращения для заполнения, просмотра или корректировки справочного отношения ничем не отличается от описываемой последовательности действий при настройке текущего обращения. И заполнение обращения для обработки справочного отношения может, в свою очередь, предусмотреть такой же выбор кодовых значений из других, более низких по каркасной иерархии отношений - справочников для справочников.

По полю. Настройка в данной колонке тесно связана с настройкой в предыдущей. Выбор производится аналогичным способом. К выбору предоставляются поля из назначенного справочника. И назначается именно то ключевое поле, по которому производится входной дублирующий

контроль. Обычно при анализе ПрО предусматриваются одинаковые (но это – не обязательно) имена полей и в текущем отношении, и в отношении-справочнике. А вот тип и размерность соответствующих полей обязательно должны быть строго идентичны: и в отношении-справочнике, и в текущем отношении, для которого настраивается обращение.

С фильтром. При определении схемы БД, при обеспечении в обращении выражения связи/псевдофильтра, уже было рассмотрено, как записывается такое выражение связи, обеспечивающее вырезку-пакет кортежей в индексированном отношении. Подобный эффект обеспечивает и настройка в данной колонке. С тем лишь отличием, что полученная благодаря выражению фильтра и индексному ордеру справочного отношения вырезка из справочника участвует во входном контроле и выводится на экран в специальную таблицу выбора лишь при ошибочном вводе или при принудительной активизации таблицы выбора. При этом актуальными для контроля являются только те записи, в которых более старшее (ведущее) ключевое поле по значению соответствует выражению фильтра. Строго говоря, фильтр действует таким образом: выражение индексации справочника позволяет системе составить его служебный индексный файл с управляющими строковыми выражениями определенной длины. А фильтрующее выражение делает видимыми только те кортежи, у которых ведущее не обязательно полное значение индексного выражения точно соответствует полному выражению фильтра. Например, фильтр вида "00" обеспечит выбор в совокупности кортежей, которые упорядочены (индексированы) ключевым полем с одинаковым начальным значением "00" - таким как "001", "002", "003" и т.д. Наиболее частой предпосылкой к применению такого фильтра является использование многоступенчатой справочно-контрольной схемы отношений, которая, по сути, и соответствует одному из важнейших принципов ДКНФ – зависимости от доменов и ключей. Фрагмент такой схемы состоит, как правило, из 3-4 иерархически

связанных реляционным каркасом отношений, обслуживаемых STD-обращениями для ввода/корректировки справочных данных.

С показом поля. При наборе ключевых данных в окне таблицы настроенного модуля, обеспечиваемом при заполнении данного обращения, возможен принудительный вызов специальной таблицы для выбора имеющихся кодовых полей из справочника. Такая таблица в простейшем случае выдает на экран монитора кортежи с данными, содержащимися в поле, имя которого указано в колонке "По полю". Однако, в большинстве случаев такой выбор малоинформативен. Поэтому желательно дополнительно выдать на экран обычно присутствующие в справочнике строковые поля, трактующие ключевое поле – некий атрибут. Для заполнения данной колонки в редакторе инсталлятора достаточно нажать <Enter> для активизации стандартного ранее рассмотренного интерфейса выбора из всех имеющихся в заказанном справочнике полей.

Как выбирать (выражение связи по индексу). Колонка выражения выбора из отношения-справочника предусматривает обязательную (если заказан справочный интерфейс) запись выражения строкового типа, которое из этого обращения предписывает модулю поиск необходимого уникального справочного кортежа. В большинстве случаев данное выражение - имя текущего поля, для которого обеспечивается ввод или корректировка в редактируемом обращении. При записи имени поля текущего обрабатываемого отношения алиас указывать необязательно - обращение предусматривает имя по умолчанию. Обычно выражение индексации справочного отношения осуществляется, например, по полю A, и если ключевые поля и в справочнике, и в текущем отношении одноименны, то выражение выбора по индексу в данном обращении также имеет вид A (A - поле текущего обрабатываемого отношения). Если соответствующие контрольно-дублируемые поля имеют нестроковый тип, тогда по соглашению системы и в выражении индексации справочника, и в выражении выбора по индексу должны быть записаны функции приведения

выражения к строковому типу ("C"). Если применен фильтр для обеспечения вырезки из справочного отношения, необходимо предварять выражение выбора выражением фильтра, что выглядит, как правило, так: если фильтр записан как строковая константа "00", выражение выбора по индексу может быть записано как "00"+A, где A - имя контрольно-дублируемого поля отношения текущего обращения. Такая запись логично вытекает из определения выражения индексации справочного отношения.

5.6.5. Удаление кортежей по умолчанию

Для каждого отношения, как неотъемлемая часть его описания в системе SWS, существует выражение удаления кортежей по умолчанию. Когда данная инсталляционная колонка пуста, логическое выражение удаления кортежей по умолчанию отношения трактуется системой как .F. (фальшь). Обычно кортежи отношения с оперативными данными имеют актуальное значение только лишь в течение определенного периода, после чего теряют свою актуальность. Поэтому иногда выражение условия удаления по умолчанию записывается как анализ даты в каком-либо поле совокупности (каскада) отношений. Например: DATE()-DATAPOS>365. Выражение в этом примере обеспечивает сохранность записей в течение года, после чего записи автоматически будут удалены модулем системы из указанного каскада отношений со строгим отслеживанием целостности всей схемы в один из моментов его активизации. Таким образом, обеспечивается защита от неконтролируемого разрастания объемов данных, и, как следствие, обеспечивается надежная работоспособность разработанного комплекса. Проектировщик не должен забывать составлять выражения условий удаления каскада кортежей для возможно разрастающихся отношений с неактуальными оперативными данными.

5.7. Генерация отчетов

При составлении обращения к отношению нужно понимать, для какого режима обработки данных составляется эта конфигурационная единица. Часто одно и то же обращение вполне обеспечивает конфигурирование

экранной таблицы или ее части как для ввода, так и для корректировки, просмотра и печати. Причем настройки "Масок STD" и другие правильно произведенные настройки никогда не конфликтуют при использовании данного обращения в различных режимах обработки модулем таблицы. Однако бывает, что существуют некоторые или даже значительные различия в представлении данных в экранной таблице модуля и в отчете. И тогда приходится создавать новые соответствующие обращения к отношению. Необходимость создания различных обращений к одному и тому же отношению проектировщик рассматривает в каждом конкретном случае. Речь идет о заполнении обращения, пригодного для любого из режимов модульной табличной обработки. Система правильно использует необходимые настройки в любом из режимов обработки данных. Здесь речь пойдет о настройке той части обращения, которое отвечает за генерацию "сжатых" отчетов при назначении соответствующего режима для настраиваемого модуля.

При наборе данных в иерархически подчиненное отношение, где для связанного ключевого поля верхнего отношения производится набор пакета текущих кортежей, часто бывает необходимым получение твердых копий (отчетов) с выборочным суммированием полей по данному опорному ключу. Есть два пути для решения этой задачи. Первый - назначение теневого "архивного" отношения, накапливающего нужные данные. Второй - непосредственное получение твердой копии документа (отчета) с прокруткой пакетов кортежей и получением необходимых просуммированных данных. Второй путь является типовым для большинства систем с БД. Это так называемые OLAP-запросы. Он обеспечивает некоторую экономию дискового пространства, однако хуже вписывается в каркасную модель размещения данных в системе SWS и on-line-концепцию. Учитывая, что на сегодня доступная аппаратная база обеспечивает более чем достаточные пространственные характеристики накопителей, не возникает никаких проблем в проектировании размещение данных так, чтобы получать

генерацию отчетов непосредственно из отношений БД. Как накапливать нужные данные в теневые отношения, будет описано далее.

Настройка STD-обращения, затрагивающая режим печати, обеспечивается в колонке "Выражение прокрутки записей при печати". Если редактируется обращение одного из подчиненных отношений с предполагаемым к включению выражением индексации вида A+B (т.е. с пакетами кортежей, сгруппированными по значению поля A), и необходимо получить отчет по уникальным значениям поля A, выражение прокрутки записей при печати будет иметь вид A. Модуль, опираясь на такую настройку, в заказанном режиме генерации твердых копий будет всегда распечатывать первую из записей сгруппированного пакета, последовательно прокручивая и анализируя остальные записи с таким же значением ключевого поля A. При этом существует возможность накопить определенные суммы в переменную, записанную в таблице редактирования "Масок STD". При смене значения поля A процесс повторяется. Таким образом, обеспечивается эффект, подобный индексации отношения с командной фразой SET UNIQUE ON.

5.8. OLAP – подход на каркасной модели данных

При обслуживании современных БД основным вопросом является скорость реакции системы. Такие бизнес-приложения, как ретроспективный анализ деятельности компании, анализ рынка, стратегическое планирование и прогнозирование и т.п. характеризуются необходимостью извлекать большое количество записей из очень больших наборов данных и вычислять на их основе с максимально возможной скоростью разнообразные итоговые показатели. Для этого требуется использование *централизованной* схемы БД. Предоставление поддержки таким приложениям является основным назначением всех OLAP-инструментов [93].

Международный комитет по стандартизации и международная электротехническая комиссия ISO/IEC в 2001 году расширили известный стандарт SQL-подобных языков [416]. Был добавлен пакет специфических

OLAP-функций. Как основные решения стандарт [416] задекларировал функции GROUPING SETS, CUBE BY и ROLLUP BY. Каждая из них позволяет получать произвольные выборки данных для ответов на запросы пользователей.

Однако основным недостатком пост-запросного подхода является плохое распараллеливание вычислительного процесса работы с терабайтами данных и, как следствие, невысокая итоговая производительность системы в целом. Подавляющее большинство запросов не является случайными, а предопределено спецификой ПрО. Но механизмы учета этого факта используются не в полной мере.

5.8.1. OLAP и механизм on-line-транзакций

Альтернативной является методика использования реального времени, предложенная в [223] (в 1994 году), когда в фоновых таблицах все необходимые итоги и подитоги формируются по факту ввода данных. При этом процесс распараллеливания осуществляется автоматически специализированной буферизацией транзакций. А за счет механизма индексирования скорость формирования фоновых таблиц - максимально возможная в рамках используемой операционной среды. Поскольку ключевые поля фоновых таблиц строятся на основании этимологии сущностей-объектов ПрО, а значит, соответствующей полной совокупности связей, вероятность появления непредусмотренных запросов в дальнейшей эксплуатации хранилища очень мала.

Данная методика позволяет также значительно усовершенствовать известный механизм унификации часто повторяющихся запросов. И on-line-формирование фоновых OLAP-итогов осуществлять на автоматизировано расширяющейся совокупности каркасных фоновых отношений, учитывающих также и новые запросы.

Еще одним важным свойством каркасной OLAP-методики является более естественный механизм отслеживания целостности данных, когда единицей атомарности является каскад данных, сформированный по всей

совокупности отношений. Протоколирование таких транзакций значительно упрощается. И механизмы восстановления потерянных групп при сбоях в системе сводятся к простому копированию.

Несложно показать, что при агрегировании значений шунтирующих атрибутов «нижних» каркасных отношений и фиксации единственного агрегированного значения в качестве шунтирующего атрибута в одном кортеже «верхнего» каркасного отношения, появление в этом кортеже дополнительных, необусловленных ограничениями на домены и ключи, функциональных зависимостей (ФЗ) между атрибутами невозможно.

Для монотонного агрегирования докажем это утверждение в виде **леммы 5.1.** Атрибут S каркасного отношения со схемой $R(X,S)$, где X является ключом, полученный произвольной монотонной функцией $F(R_i(X_i,A_{ij}))$ от произвольной совокупности $A_{i,j}$ неключевых атрибутов иных каркасных отношений R_i, каждое из которых удовлетворяет только особым ограничениям [208], не является детерминантом неключевой ФЗ отношения R.

Доказательство проведем от противного. Пусть существует некоторая функция F такая, что $S=F(R_i(X_i,A_{ij}))$ – также ключ отношения $R(X,S)$, как и X. Но тогда, поскольку функция монотонна, для нее найдется обратная функция $Z = F^{-1}$ такая, что $Z(R(X,S)) = A_{ij}$. Тогда атрибут A_{ij} - также ключ отношения $R_i(X_i,A_{ij})$, так как функция не устраняет ФЗ атрибутов-аргументов, в том числе и $A_{ij} \to X_i$. Но такая ФЗ противоречит особым условиям на домены и ключи каркасного отношения [208], в каждом из которых ключ единственный. Противоречие доказывает лемму. □

Из доказанного следует, что использование в одном кортеже нескольких агрегированных атрибутов от разных аргументов агрегирования не приводит отношения к денормализации, так как никаких «паразитных» ФЗ между агрегатами не возникает. Но следует также и то, что использование в одном кортеже нескольких агрегированных атрибутов от одного аргумента

агрегирования приведет к денормализации. В таком отношении появятся транзитивные ФЗ в неключевых шунтирующих агрегированных атрибутах.

С одной стороны, транзитивные ФЗ, которые появляются между несколькими агрегированными шунтирующими атрибутами от общего аргумента агрегирования, не являются критическими, так как устраняются обычной декомпозицией на несколько каркасный отношений, каждое из которых может хранить необходимый единственный агрегат. Но с другой стороны, хранение значительного числа результатов агрегирования от громоздких функций может привести к значительным временным затратам на операции по отслеживанию целостности этих данных. Поэтому решение проектировщик принимает исходя из статистики запросов.

Для выполнения основных OLAP-функций в SWS использован механизм пошаговых параллельных транзакций, построенный на групповых функциях [204, 218], аналогичных перечню стандартизованных вычислительных функций [416].

Благодаря интеграции рабочих станций в распределенную среду становится возможным более эффективное распределение функций в ней, когда прикладные программы выполняются на рабочих станциях, называемых *серверами приложений,* а БД обслуживаются выделенными компьютерами, называемыми *серверами БД*. Это служит источником развития таких распределенных архитектур, где в роли узлов выступают не просто компьютеры общего назначения, а специализированные серверы.

Параллельный компьютер, или мультипроцессор сам по себе является распределенной системой, составленной из узлов (процессоров, компонентов памяти), соединенных быстрой сетью внутри общего корпуса. Технология распределенных БД может быть естественным образом пересмотрена и распространена на параллельные системы БД, т. е. системы БД на кластерах [371]

Данная методика позволяет также значительно усовершенствовать известный механизм унификации часто повторяющихся запросов. И on-line-

формирование теневых OLAP-итогов осуществлять на автоматизировано расширяющемся поле теневых таблиц, учитывающих новые, случайно появившиеся в системе, запросы.

Еще одним важным преимуществом предложенной методики является более естественный механизм отслеживания целостности данных, когда единицей атомарности является каскад данных, сформированный по всей совокупности таблиц. Протоколирование таких транзакции значительно упрощается. И механизмы восстановления потерянных групп при сбоях в системе сводятся к простому копированию.

Проведен численный анализ скорости доступа к данным при пост-обработке и в on-line режиме. Заметное уменьшение времени получения итогов в режиме реального времени – более 10 % - наблюдается уже при значении суммарного количества записей в БД около 10^6 . При этом время задержки выполнения каждой транзакции при 10 параллельных процессах и более 100 дополнительных теневых таблиц, в которых по факту появления каждого новой атомарной группы данных в режиме реального времени формируются всевозможные статистические итоги, составляет 1-2 секунды на каждом рабочем месте.

При увеличении основных характеристик системы - количества хранимых данных, количества пользователей системы, количества формирующихся в квазиреальном времени теневых таблиц, эффективность параллельной обработки значительно повышается. И хотя при этом может наблюдаться незначительное увеличение времени задержки ввода данных на каждом рабочем месте, такие задержки не являются критичными.

При классическом OLAP-подходе может быть выбрана, например, схема «снежинка» [423], где отношения-«факты» содержат оперативные данные. А множественные отношения с «измерениями», присоединенные к «фактам», показывают, как оперативные данные могут агрегироваться. Эти отношения показывают, как могут анализироваться агрегированные реляционные данные. Число возможных агрегирований определяется числом

первоначальных иерархических отображений. Такой OLAP-механизм вынуждает пользователя тратить достаточно большой промежуток времени на ожидание ответа на запрос – от нескольких часов до нескольких суток. Но специфика работы современной ПрО чаще всего требует мгновенного анализа.

При использовании такого механизма OLAP пользователь вынужден тратить достаточно большой промежуток времени на ответ на запрос – от нескольких минут до нескольких часов. Но чаще всего специфика работы современной ПрО требует мгновенного анализа.

Иной результат дает OLAP–подход на основе РК и режима реального времени (синхронный OLAP), при котором скорость получения данных сравнима со скоростью обращения к одной таблице с поиском по нескольким полям – менее 1 секунды. Ниже приведены результаты численного эксперимента.

5.8.2. OLAP-реализация в CASE-оболочке SWS

Функционирование модуля CASE-оболочки SWS [221, 225] организовано таким образом, что все необходимые расчетные данные обрабатываются в режиме реального времени - в тот момент, когда производится ввод исходных данных. Для реализации такого положения есть как минимум три точки инсталляции - заполнение списка суммирующих полей в корне редактирования обращения, заполнение списка заменяемых полей в «настройки экрана», а также заказ функциональных выражений вида A*B в колонке «поле-функция-переменная» редактора экранных настрое. Как показала практика внедрений, значительное число пользователей большинство расчетных данных формирует посредством заполнения списка суммирующих полей.

Алгоритм этой унифицированной процедуры следующий.

1. Накопление может быть произведено только в поля либо текущего отношения, для которого описывается обращение, либо в поля иерархически верхних отношений – объявленных как фоновые головные, или

подключенные иным способом. В поля кортежей иерархически подчиненных отношений производить суммирование бессмысленно.

2. При каждом вводе в любое поле (редактировании любого поля) текущего кортежа активизируется описываемая накапливающая инициализация суммирующих полей.

3 Специфика накопления следующая - если данные в суммирующих полях существуют, то на первом шаге производится арифметическое вычитание указанных в этих колонках выражений с их аргументами до момента возбуждения (ввода в поле, редактирования поля). А на втором шаге производится арифметическое прибавление указанных в этих колонках выражений, но уже с участием измененных аргументов (после момента возбуждения). Такой алгоритм позволяет динамически поддерживать целостность вычисляемых данных, как при вводе, так и при редактировании полей с существующими данными.

4. Суммирование производится по порядку очередности списка сверху вниз - каждая последующая строка может использовать выражения с участием суммирующих полей с уже просчитанными данными из строк выше.

5.8.3. Численное исследование элементарной OLAP-процедуры

В работе проведен эксперимент, основной характеристикой которого является время доступа к любому атрибуту отношения-маски исследуемой БД. Как известно, самой ресурсоемкой операцией реляционной алгебры являются соединения отношений [300]. Эта операция применяется при выполнении запросов к БД без использования отношений-масок и более низкой «нормальности» – не выше 3НФ. Особенностью каркасной схемы является то, что в БД большинство исследованных связей моделируется уже соединенными отношениями. Поэтому отсутствие операции соединения приводит к значительной экономии времени доступа к БД.

Для экспериментального исследования каркасного метода проектирования схемы БД и проведения указанного численного

эксперимента была выбрана ПрО «Городская поликлиника». Описываемое приложение БД было разработано в г. Хмельницком на одной из городских поликлиник с помощью CASE-генератора SWS [221] сторонними пользователями этой инструментальной системы. В работе имеется соответствующий акт внедрения. Как показано в [221], само CASE-средство SWS проектировалось в строгом соответствии с каркасной моделью данных. При этом приложения БД, синтезируемые с помощью SWS и каркасной схемы БД, обладают высокой эффективностью.

Рассмотрим подробнее элементы синхронного формирования OLAP-данных на примере каркасной схемы БД (приведена на рис. 5.2.), которая моделирует ведение журналов начислений по абонентам из ПрО «Горгаз» [211]. Имеем следующие сущности-объекты.

ИТОГ ЗА ГОД (*НомГода,* Сумма, Прим, …) – атомарная сущность-объект,

ИТОГ ЗА КВАРТАЛ (*НомГода, НомКварт,* Сумма, Прим, …) - слабая сущность-объект,

ИТОГ ЗА МЕСЯЦ (*НомГода, НомМес,* Сумма, Прим, …) - слабая сущность-объект,

ИТОГ ЗА ДЕНЬ (*НомГода, НомМес, НомДня,* Сумма, Прим, …) - слабая сущность-объект,

ИТОГ ЗА ДЕНЬ ПО ОТДЕЛУ (*НомГода, НомМес, НомДня, КодОтдела,* Сумма, Прим, …) - слабая сущность-объект,

ВЫРУЧКА СОТРУДНИКА ОТДЕЛА ЗА ДЕНЬ (*НомГода, НомМес, НомДня, КодОтдела, КодСотрудн,* Сумма, Прим, …) - слабая сущность-объект,

СОТРУДНИКИ В ОТДЕЛАХ (*КодОтдела, КодСотрудн,* Прим, …) - слабая сущность-объект,

АБОНЕНТЫ ПО СОТРУДНИКАМ (*КодАбонента, КодСотрудн,* Прим,…) - составная сущность-объект (справочник),

МЕСЯЦЫ В КВАРТАЛАХ (*НомМесяца, НомКварт,* Прим, …) – составная сущность-объект (справочник),

ОПЛАТА АБОНЕНТОВ (*НомГода, НомМес, НомДня, КодАбонента,* Сумма, НомСчета, …) – составная сущность-объект.

В схеме БД исключены некоторые зависимые от времени атрибуты, а также отношения, моделирующие такие сущности-объекты. Эти упрощения не влияют на результаты эксперимента.

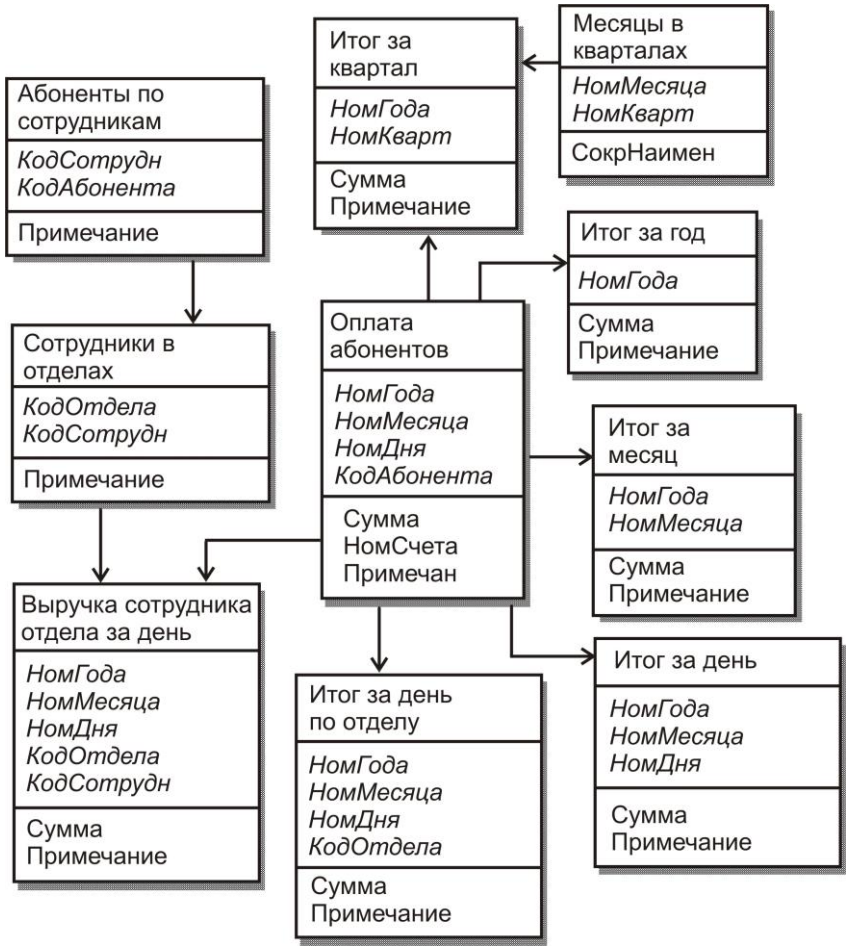


Рис. 5.2. Схема ПрО «Горгаз»

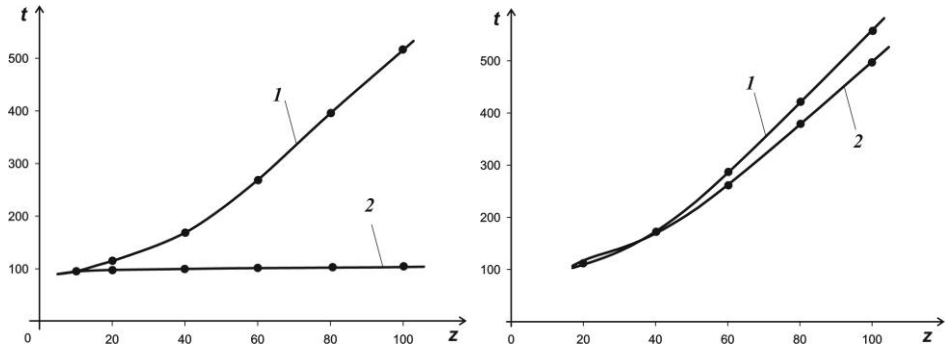
Процедура фонового формирования отношений следующая: при введении оператором в поле *СУММА* данных об оплате некоторым абонентом суммы за определенный счет, срабатывает унифицированная каскадная функция, которая пересчитывает итоговые данные во всех фоновых головных отношениях.

Проведен численный анализ скорости доступа к данным при пост-обработке и в on-line-режиме. Заметное уменьшение времени получения

итогов в режиме реального времени – более 50% - наблюдается для БД, состоящей из 100 отношений, уже при значении числа кортежей около 10^5 в половине отношений.

Исследовано также и время задержки на выполнение арифметической операции на каскаде отношений. При тестировании схемы моделировалось следующее: на протяжении года 200 операторов ежедневно вносят по 1000 кортежей в общее отношение, что в итоге составляет $73 \cdot 10^6$ записей. При этом формируется до 100 фоновых итоговых отношений. И на начальной, и в завершающей фазе общее усредненное время выполнения функции (время задержки системы) с учетом интернет-трафика не превышает 1 секунды на каждом рабочем месте, что не является существенным. Результат полностью согласуется с методикой индексного поиска.

При дальнейшем увеличении основных характеристик системы - количества хранимых данных, числа пользователей системы, а также числа формирующихся в реальном времени фоновых отношений, эффективность параллельной обработки значительно повышается.



а б

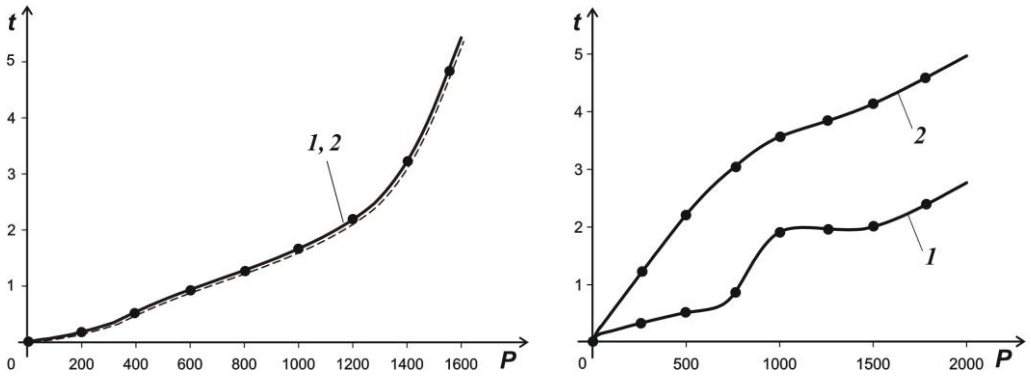
Рис. 5.3. Зависимость времени от числа фоновых отношений

На рис. 5.3., а приведен график роста времени задержки (в наносекундах) на выполнение процедуры фонового суммирования от числа формируемых отношений для 1 ядра (кривая 1) и 24 ядер (кривая 2) от 100 пользователей. Приведена зависимость времени выполнения параллельных операций, когда результат следующей операции не зависит от результата предыдущей. Видно, что сокращение времени транзакций прямо

пропорционально числу ядер. И общий прирост времени с увеличение нагрузки на БД в многоядерной конфигурации незначителен. Каркасная схема БД позволяет применять алгоритмы агрегирования с независимыми операциями.

Каждое отношение формируется до 10^7 кортежей. Видно, что распараллеливание процесса формирования агрегированных отношений значительно ускоряет выполнение фоновых процедур. Однако даже без распараллеливания сервер AMD OPTERON X12 с 2 процессорами по 12 ядер (3,26 GHz на ядро) с операционной системой UBUNTU SERVER 12.04 и сервером данных PostgreSQL 3.1, выполнил в описанном эксперименте транзакцию для 100 таблиц за 500 нСек – более чем удовлетворительное время. Все остальные затраты времени приходятся на работу сети.

На рис. 5.3., б приведена та же зависимость, но для схемы БД, где результат следующей операции зависит от результата предыдущей. Очевидно, что последовательные вычисления плохо подаются распараллеливанию. И время выполнения операции зависит только от быстродействия ядер, а не от их числа. Незначительное ускорение многоядерной архитектуры объясняется использованием параллельных процессов для выполнения вспомогательных операций - переноса данных с жёсткого диска в ОЗУ, формирование прерываний пользователя, накладные расходы самой СУБД и т.п.



а б

Рис. 5.4. Зависимость времени от числа фоновых отношений

На рис. 5.4. приведены графики роста времени задержки (в миллисекундах) на формирование фоновых таблиц от числа пользователей.

График на рис. 5.4., а показывает зависимость времени выполнения последовательных операций с 100 (кривая 1) и 500 (кривая 2) фоновыми отношениями от числа пользователей подключенных к БД. Видно, что рост времени выполнения операций зависит не столько от числа формируемых запросом таблиц, сколько от числа пользователей, что объясняется спецификой СУБД. Необходимость обработки прерываний инициатора транзакций и возврат ему данных вызывает дополнительные вычислительные расходы.

График на рис. 5.4. б демонстрирует выполнение той же процедуры, но с учетом сетевых издержек на передачу данных. Кривая 1 показывает время на выполнение операции и на отсылку данных клиенту, а кривая 2 – в том числе и время на получение данных и декодирование сетевого запроса на транзакцию. Из графика видно, что время собственно обработки данных очень мало по сравнению со временем передачи-приема запроса, что говорит о том, что основной причиной задержек в работе с БД является работа сети, а не недостатки алгоритма или вычислительная мощность ПК.

Заметим, что приведенная на рис. 5.2. схема БД имеет некоторую схожесть с известной [423] схемой ХД «снежинка». Однако отличием является возможность построения косвенных связей между отношениями по неполному соответствию ключей (на рис. 5.2. это связь между отношениями *ОПЛАТА АБОНЕНТА* и *ВЫРУЧКА СОТРУДНИКА ОТДЕЛА ЗА ДЕНЬ*), формирование параллельных веток табличных деревьев, моделирование рекурсивных связей сущностей-объектов и поддержка режима реального времени. Таким образом, схема «снежинка» является частным случаем РК. Поэтому схемы БД, подобные приведенной на рис. 5.2., могут использоваться не только как схемы ХД, поддерживая при этом лишь операции извлечения кортежей, но и как OLTP-базы. Стрелками на рис. 5.2.

показаны противоположные [423] потоки данных, что также является важным отличием.

5.9. История промышленной эксплуатации CASE-оболочки SWS

Эта разработка была начата в Украине в 1988 году. Истории разработки и всесторонней апробации результатов было бы необходимо выделить немалый раздел литературы. Поэтому приведем лишь некоторые ссылки.

Начало промышленных испытаний и широкого внедрения CASE-оболочки SWS, которое было разработано в полном соответствии с каркасной моделью данных, зафиксировано авторами в специализированный национальной и российской печати: «Деловые новости», № 17 (80), К, 1995, с. 10; ComputerWorld Киев, № 6 (29), К, 1995, стр. 11-14; «Монитор», № 8, М, 1994, с. 79-82, «Компьютера», № 35 (66), с. 15, № 42-43 (73-74), с. 16, - М, 1994, № 6 (86), с. 19, - М, 1995.

Ниже приведен список наиболее показательных задач, решаемых приложениями БД, которые были разработаны и внедрены с помощью SWS сторонними пользователями в соответствующих организациях за период с 1992 по 1998 гг. В Приложениях 9. и 10. имеются копии соответствующих публикаций.

Среди пользователей инструментального средства - Телевизионный технический центр «Останкино» [244], Редакция газеты «Аргументы и факты» [242], Московская мебельная фабрика [84], Российское федеральное дорожно-строительное управление, Шахта «Комсомолец Донбасса», Хмельницкая городская поликлиника, Дунайское речное пароходство [243], войсковая часть и машиностроительный завод.

Спектр ПрО, которые моделируются внедренными приложениями, очень широк. Это начисления по всем типам заработной платы, учет задолженности и начисления отпусков, система подготовки и обработки документов общего финансового учета (касса, банк, подотчетные лица, расчеты с дебиторами и кредиторами, валютные операции и т.п.), отчетность аппарата управления и сбыт готовой продукции (склад,

счета, накладные, расчеты по сбыту), основные фонды и планирование производства (расчет себестоимости, рентабельности, ценообразование), снабжение (взаиморасчеты с поставщиками, материалы, комплектующие) и производство (рабочее место технолога, движение сырья, материалов, деталей), кадры и ремонтные работы, учет травм и происшествий и т.п.

Среди внедренных приложений разработаны и уникальные системы: картотека первого отдела, розничное и подписное распространение прессы, анализ качества добытого угля, учет выработки тепла по котельным города и эксплуатация котельных (оборудование и теплотрассы), начисление денежного довольствия офицеров и прапорщиков, учет прививок, регистрация талонов амбулаторных больных и анализ заболеваемости и т.п.

У авторов имеются акты внедрения перечисленных систем, подтверждающие длительную актуальность промышленных приложений БД, которые были разработаны и внедрены пользователями. Особое значение имеет акт внедрения, который декабрем 2009 года фиксирует текущий период актуальности приложений на шахте «Комсомолец Донбасса», промышленная эксплуатация которых была начата в 1998 году.

5.10. Выводы к пятому разделу

1. Описанный подход к автоматизации проектирования и кодирования приложений БД принципиально отличается от известных [45, 109, 146] тем, что вместо избыточного синтезатора листингов в CASE-оболочке SWS использована строгая унификация схем БД и типизация функций; а вместо привязки схемы БД к специфике ПрО посредством ручного кодирования ER-диаграммы [359] использован типовой шаблон, значительно упрощающий декомпозицию ПрО .

2. Разработанная и всесторонне тестированная CASE-оболочка SWS подтвердила гипотезу о том, что эффективное решение проблемы унификации, типизации и минимизации инструментального средства, которое масштабируется метаданными и позволяет синтезировать приложения БД, моделирующие функционирование произвольных ПрО,

обеспечивает совокупность каркасных отношений. Созданный малогабаритный, но мощный CASE-генератор каркасных метаданных, управляющих универсальной оболочкой, практически полностью исключил потребность в ресурсоемких операциях соединения в большинстве запросов к БД и существенно упростил настройку приложений в прикладной разработке.

3. В рамках настоящего исследования базовые гипотезы КМД о единственности фактора этимологии и о сходимости алгоритма последовательных приближений на каркасе-шаблоне [216, 217] не являются строго доказанными утверждениями. Однако они широко применяются авторами и пользователями на практике. В настоящей работе приведены доказательства справедливости этих утверждений, полученные экспериментальным путем. Исследовано 15 разных ПрО [84, 242-244], в том числе не связанных непосредственно с БД [228, 230, 231] .

4. Проанализированы приложения и схемы БД, разработанные сторонними пользователями как самостоятельно, так и при непосредственном участии авторов. Решенные задачи подтверждают справедливость основных теоретических утверждений, доказанных в [208], - о подобии каркасных схем БД для разных ПрО, модифицируемости систем без останова приложений, а также об отсутствии аномалий .

5. Данный подход особенно важен для тех ПрО, где ограничения часто изменяются, так как позволяет существенно минимизировать затраты на разработку и сопровождение приложений .

РАЗДЕЛ 6
ПрО ВЫЧИСЛИТЕЛЬНОГО ХАРАКТЕРА - ПАРАЛЛЕЛЬНО-КОНВЕЙЕРНЫЕ СХЕМЫ ВЫСОКОТНЫХ КЛАСТЕРНЫХ ВЫЧИСЛЕНИЙ В ДИНАМИЧЕСКИХ ЗАДАЧАХ МЕХАНИКИ СПЛОШНЫХ СРЕД

6.1. Введение к шестому разделу

Современные вычислительные комплексы в сочетании с программными системами, базирующимися на хорошо обусловленных алгоритмах, позволяют высокоэффективно моделировать напряженно-деформированное состояние сред с усложненными свойствами. Однако вопрос автоматизированного синтеза приложений, гибко перенастраиваемых в зависимости от изменения конфигурации механических систем, практически не изучен. Большинство исследований посвящено развитию метода конечных элементов [1]. Однако существуют иные подходы, позволяющие существенно экономить вычислительные ресурсы и повышать тем самым точность вычислений. Поэтому при рассмотрении вопроса проектирования инструментальных программных средств (CASE-средств) [45, 109, 146], позволяющих синтезировать и сопровождать приложения (моделирующие динамическое поведение сложных механических систем), необходимо проанализировать именно эти методики решения задач механики сплошных сред.

Для анализа ресурсов конструкций, содержащих значительное число неоднородностей и работающих под воздействием динамических нагрузок, исследуют взаимодействие волн перемещений и напряжений в упругой среде с отверстиями, включениями, трещинами или линейными вставками. Поэтому изучение дифракции упругих волн на системах произвольных неоднородностей является вопросом важным и актуальным. Однако вследствие необходимости привлечения больших объемов вычислений и значительных ресурсов цифровой памяти такие задачи исследованы мало. В связи с этим особое значение приобретают эффективные параллельные

алгоритмы, в основе которых лежат обоснованные аналитические методы [46]. Для решения плоских и антиплоских задач теории дифракции [76, 271] большой эффективностью обладает метод интегральных уравнений [202, 182, 307]. Дополнительные преимущества этого метода заключаются в сокращении числа пространственных переменных, достаточно высокой скорости сходимости и возможности применения различных эффективных численных методов решения [202]. Кроме того, метод обладает большими возможностями при построении параллельных вычислительных схем [185].

6.2. Постановка задачи

При проектировании того или иного CASE-средства синтеза и сопровождения приложений как правило ПрО представляется в виде схемы БД [109, 208]. И все дальнейшие действия сводятся к проектированию инструментального приложения БД.

Целью настоящей части исследования является анализ специализированной ПрО - метода СИУ, а также алгоритма параллельного численного решения СИУ на примере моделирования задач динамической теории упругости. Результатом анализа является построение схемы БД и как следствие – проект CASE-средства параллельного решения систем СИУ, разработанный на ПрО моделирования поведения систем неоднородностей в изотропной неограниченной среде от воздействия гармонических волн плоской и антиплоской деформации.

Схема БД исследуемой ПрО разрабатывалась с использованием каркасного анализа, когда РК [208] использовался как шаблон схемы. Каркасный анализ ПрО позволил сформировать списки основных сущностей-объектов, а также всех связей между ними. И обеспечить структуре разрабатываемой инструментальной программной оболочки соответствие требованиям модифицируемости и безаномальности [208], что в свою очередь позволяет говорить о кроссплатформности и интероперабельности проектируемого средства.

Дополнительным конкурентным преимуществом такого подхода является возможность применения современных OLAP-решений к получаемым итоговым множествам (массивам решений), хранимым в РБД. И тем самым корректному решению обратных задач механики – идентификации и управления поведением систем [24].

Такой анализ позволяет учесть в схеме БД все необходимые совокупности сущностей-объектов – метаданных, геометрических характеристик исследуемых решеток, искомых величин, объектов интерфейса приложений и т.п. При этом основные математические структуры ПрО, которые положены в основу проектируемого инструментального средства, представлены в виде списков. А основные функции работают как процедуры отслеживания целостности данных.

Проведем подробное описание некоторых базовых задач, а также основных отличительных особенностей метода СИУ для каждой из них.

6.3. Антиплоская деформация

Под продольным сдвигом (или антиплоской деформацией) понимают напряженно-деформированное состояние цилиндрического тела, нагруженного по боковой поверхности усилиями, направленными и равномерно распределенными вдоль образующей. В предположении, что ось деформации направлена вдоль оси *OZ* декартовой прямоугольной системы координат *OXYZ*, отличными от нуля являются две компоненты тензора напряжений σ_{13}, σ_{23} и одна компонента перемещения *W(x,y,t)*, причем:

$$\frac{\partial\sigma_{13}}{\partial x}+\frac{\partial\sigma_{23}}{\partial y}+f=\rho\frac{\partial^2 W}{\partial t^2},\ \sigma_{13}=\mu\frac{\partial W}{\partial x},\ \sigma_{23}=\mu\frac{\partial W}{\partial y},$$

где ρ - плотность, μ - модуль сдвига среды, t - время, f - объемная сила.

Из этого вытекает, что перемещение *W(x,y,t)* удовлетворяет неоднородному волновому уравнению Гельмгольца [202]:

$$\nabla^2 W-\frac{1}{c^2}\frac{\partial^2 W}{\partial t^2}=-\frac{1}{\mu}f\ ,\ c^2=\frac{\mu}{\rho}.$$

6.3.1. Система цилиндрических неоднородностей

Рассмотрим неограниченное изотропное пространство, содержащее систему m бесконечных вдоль оси OZ неоднородностей, поперечное сечение которых представляет собой замкнутые (без общих точек) или разомкнутые контура произвольной формы. Обозначим L-совокупность указанных контуров и пусть положительное направление выбрано так, что при движении вдоль L область D остаётся слева (рис. 6.1.).

Совокупность L будем разделять на:

1. $L^{(1)}$ – совокупность контуров однородных упругих включений;
2. $L^{(2)}$ – совокупность контуров отверстий;
3. $L^{(3)}$ – совокупность замкнутых контуров однородных жёстких включений.
4. $L^{(4)}$ – совокупность разомкнутых контуров криволинейных жёстких вставок.
5. $L^{(5)}$ – совокупность криволинейных контуров трещин-разрезов.

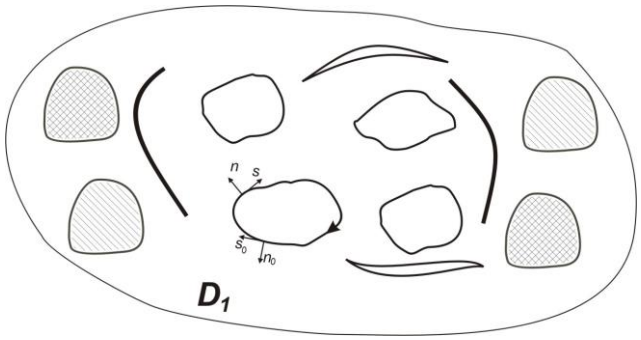


Рис. 6.1.

В качестве источника внешнего воздействия поля перемещений может выступать набегающая на цилиндры из бесконечности монохроматическая SH-волна или гармонический сосредоточенный источник заданной интенсивности [182]. В результате взаимодействия приходящей волны с указанной системой неоднородностей возникает сложное дифрагированное волновое поле. Считаем, что отражённая и проникающая (случай присутствия упругих включений) волны имеют ту же частоту колебаний, что и возбуждающий источник. Это позволяет сделать переход к амплитудам перемещений. Пусть W_0, W_1 и W_2 – амплитуды перемещений возбуждающего, отраженного и проникающего полей

соответственно. Тогда общее поле W равно $W^+ = W_0 + W_1$ в матрице (область D_1) и $W^- = W_2$ внутри упругих включений (область D_2). В случае гармонической зависимости от времени ($e^{-i\omega t}$) амплитуды W_1 и W_2 удовлетворяют уравнениям Гельмгольца [202].

Сформулируем краевые условия на L для разрешающих уравнений Гельмгольца.

1. На $L^{(1)}$: $\tau^+_{n_0} = \tau^-_{n_0}$, $W^+ = W^-$ – условие типа склейки соответствующих касательных напряжений и перемещений со стороны матрицы и включений.

2. На $L^{(2)}$: $\tau^+_{n_0} = \mu_1 \dfrac{\partial W^+}{\partial n_0} = 0$.

3. На $L^{(3)}$: $W^+ = const$.

4. На $L^{(4)}$: $W^\pm = const$.

5. На $L^{(5)}$: $\tau^\pm_{n_0} = 0$ – берега трещин-разрезов свободны от сил.

Таким образом, задача дифракции SH-волн на системе неоднородностей указанного типа в неограниченной изотропной среде сводится к решению уравнений Гельмгольца при выполнении краевых условий на L и дополнительных условий излучения типа Зоммерфельда на бесконечности [202].

В данной работе решение краевых задач 1 – 4 основано на представлении амплитуды W_1 рассеянного поля (и амплитуды W_2 приникающего поля при наличии $L^{(1)}$) в виде потенциала типа простого слоя [185]:

$$W_m(x,y) = \int_L f_m(s) G_m(x,y,\xi,\eta)\,ds,\ G_m = \frac{1}{4i} H_0^{(1)}(\gamma_m r),\ m = 1,2,$$

(6.1)

$$r = |z - \zeta|,\ z = x + iy,\ \zeta = \xi + i\eta \in L,$$

а краевой задачи 5 – в виде потенциала типа двойного слоя:

$$W_1(x,y) = -\int_{L^{(5)}} g(s)\frac{\partial G}{\partial n}ds\,. \qquad (6.2)$$

Приведенные интегральные представления автоматически удовлетворяют необходимым уравнением Гельмгольца и условиям излучения на бесконечности. Остаётся выполнить граничные условия на контурах неоднородностей.

Сделаем следующее замечание. Удовлетворение граничных условий по перемещениям приводит к интегральным уравнениям 1-го рода с логарифмическими ядрами, численная реализация которых не очень эффективна с точки зрения точности параллельных вычислений. С целью получения СИУ 1-го рода, для которых разработана высокоточная схема параллельных вычислений [185], граничное равенство по перемещениям дифференцировалось по дуговой координате s_0. Единственные решения СИУ 1-го рода, которые возникают в результате выполнения модифицированных граничных условий, получаются, если к этим уравнениям присовокупить некоторые дополнительные интегральные равенства, вытекающие из постановки соответствующей краевой задачи.

В случае, если совокупность $L^{(1)}$ содержит I контуров упругих включений, необходимые дополнительные условия вытекают из равенства средних перемещений на $L_i^{(1)}$, $i = 1,\ldots,I$ (l_i – длина контура $L_i^{(1)}$):

$$\frac{1}{l_i}\int_{L_i^{(1)}} W^+ ds_0^i = \frac{1}{l_i}\int_{L_i^{(1)}} W^- ds_0^i\,,\ i = 1,\ldots,I\,. \qquad (6.3)$$

Если совокупность $L^{(3)}$ содержит K контуров подвижных жёстких включений, дополнительные интегральные равенства следуют из закона движения включений как абсолютно жёстких тел. Уравнения движения включений здесь можно записать в виде:

$$\int_{L_k^{(3)}} \tau_{n_0}^+ ds_0^k = -\omega^2 \rho_0 S_0 \int_{L_k^{(3)}} W^+ ds_0^k\,,\ k = 1,\ldots,K\,, \qquad (6.4)$$

где ω – частота колебаний, ρ_0 и S_0 – плотность и площадь однородных

жёстких включений.

В случае $L^{(4)}$ – совокупность M контуров криволинейных жёстких вставок дополнительные условия вытекают из равенства нулю главного вектора сил, действующих на контуре $L_m^{(4)}$ m-й вставки ($m=1,\ldots,M$):

$$\int\limits_{L_m^{(4)}} f_1\left(s^m\right)ds^m = 0,\ f_1\left(s^m\right)=\left[\frac{\partial W}{\partial n}\right] \text{ на } L_m^{(4)}. \qquad (6.5)$$

Удовлетворение соответствующих граничных условий на L сводит рассматриваемые краевые задачи к системе интегральных уравнений.

1. На $L^{(1)}$ выполнение граничных условий по напряжениям приводит к интегральным уравнениям Фредгольма 2-го рода, а выполнение модифицированных граничных условий по перемещениям – к СИУ 1-го рода. Для замыкания алгоритма к последним необходимо присовокупить дополнительные условия (6.3).

2. На $L^{(2)}$ краевая задача сводится к решению системы интегральных уравнений Фредгольма 2-го рода.

3. На $L^{(3)}$ модифицированные граничные условия $\dfrac{dW^+}{ds_0}=0$ дают систему СИУ 1-го рода. Необходимые дополнительные условия для разрешимости этих уравнений вытекают из (6.4).

4. На $L^{(4)}$ также выполняются модифицированные граничные условия $\dfrac{dW^+}{ds_0}=0$ и получаются СИУ 1-го рода на системе разомкнутых контуров. Дополнительные интегральные равенства, обеспечивающие единственность решения, имеют вид (6.5). Здесь плотности интегральных уравнений равняются скачкам производной по нормали от амплитуды перемещения ($f_1\left(s^m\right)=\left[\dfrac{\partial W}{\partial n}\right]$ на $L_m^{(4)}$), и они имеют корневую особенность на концах m-й криволинейной вставки ($m=1,\ldots,M$).

5. На $L^{(5)}$ интегральное представление (2) автоматически

обеспечивает непрерывность производной по нормали от перемещения и скачок перемещения на $L^{(5)}$ ($g\left(s^n\right)=[W]$ на $L_n^{(5)}$, $n=1,\ldots,N$). Краевая задача сводится к сингулярным интегро-дифференциальным уравнениям 1-го рода. Необходимые дополнительные условия вытекают из условия равенства нулю скачков перемещений на концах $L_n^{(5)}$, $n=1,\ldots,N$ и записываются в виде

$$\int_{L_n^{(5)}} \frac{dg}{ds^n} = 0,\ n=1,\ldots,N. \tag{6.6}$$

Здесь решение имеет корневую особенность на концах *n*–го разреза.

В качестве примера запишем СИУ в случае $L=L^{(1)}+L^{(2)}$, где $L^{(1)}$ – совокупность I контуров упругих включений, $L^{(2)}$ – совокупность J контуров полостей. Имеем

$$\begin{gathered}
\frac{1}{2}\mu_1 f_1\left(s_{10}^i\right)+\frac{1}{2}\mu_2 f_2\left(s_{10}^i\right)+\sum_{p=1}^{I}\int_{L_p^{(1)}}\left(\mu_1 K^+\left(s_{10}^i,s_1^p\right)f_1\left(s_1^p\right)-\mu_2 K^-\left(s_{10}^i,s_1^p\right)f_2\left(s_1^p\right)\right)ds_1^p+ \\
+\sum_{r=1}^{J}\int_{L_r^{(2)}}\mu_1 K\left(s_{10}^i,s_2^r\right)f\left(s_2^r\right)ds_2^r=N_1\left(s_{10}^i\right), \qquad (6.7) \\
\sum_{p=1}^{I}\int_{L_p^{(1)}}\left(M^+\left(s_{10}^i,s_1^p\right)f_1\left(s_1^p\right)-M^-\left(s_{10}^i,s_1^p\right)f_2\left(s_1^p\right)\right)ds_1^p+ \\
+\sum_{r=1}^{J}\int_{L_r^{(2)}}M\left(s_{10}^i,s_2^r\right)f\left(s_2^r\right)ds_2^r=N_2\left(s_{10}^i\right),\ i=1,\ldots,I, \\
\frac{1}{2}\mu_1 f\left(s_{20}^j\right)+\sum_{p=1}^{I}\int_{L_p^{(1)}}\mu_1 K^+\left(s_{20}^j,s_1^p\right)f_1\left(s_1^p\right)ds_1^p+ \\
+\sum_{r=1}^{J}\int_{L_r^{(2)}}\mu_1 K\left(s_{20}^j,s_2^r\right)f\left(s_2^r\right)ds_2^r=N_3\left(s_{20}^j\right),\ j=1,\ldots,J.
\end{gathered}$$

Здесь ядра $M^+\left(s_{10}^i,s_1^p\right)$, $M^-\left(s_{10}^i,s_1^p\right)$ – сингулярны, остальные ядра – непрерывны. СИУ следует рассматривать совместно с дополнительными условиями (6.3).

В случае полупространства с защемлённой или свободной от сил границей $y = 0$ система СИУ может быть построена, если исходить из соответствующих интегральных представлений, рассмотренных в [178, 213, 214].

6.3.2. Периодическая система неоднородностей

Метод интегральных уравнений, развиваемый в данной работе, является универсальным. Поэтому предлагаемая методика исследования распространяется на случай дифракции гармонической волны сдвига $W_0 = \tau e^{-i\gamma_2 y}$ на $2d$-периодической решётке, составленной из неоднородностей указанного типа (рис. 6.2.).

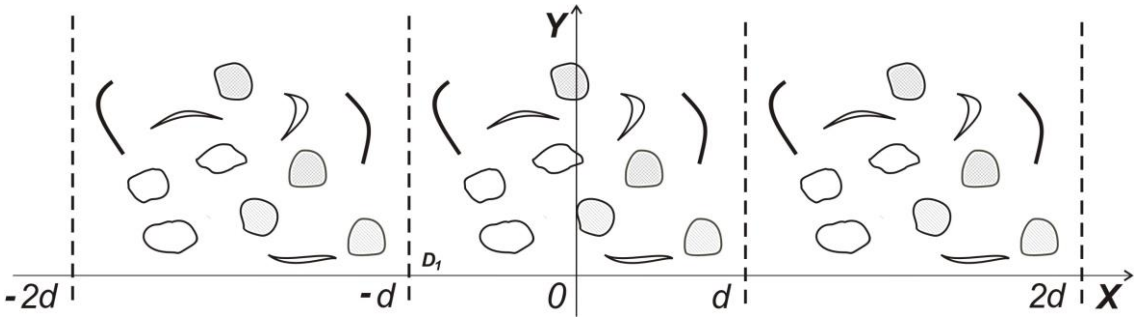


Рис. 6.2.

Здесь также можно использовать интегральные представления (6.1), (6.2), если в них в качестве функций источника G_1 и G_2 выбрать периодические функции источника для областей D^+ и D^- (случай присутствия в решётке упругих включений). Имеем [184]:

$$G_m(x,y,\xi,\eta) = \frac{\ln 2}{2\pi} + \frac{1}{4\pi} \ln\left(\sin\frac{\pi(z-\zeta)}{2d} \sin\frac{\pi(z-\bar{\zeta})}{2d} \right) + \frac{1}{2d} \sum_{k=0}^{\infty} f_k^{(m)}(y-\eta) \cos\alpha_k (x-\xi),$$

(6.8)

$$f_0^{(m)}(p) = -\frac{|t|}{2} + \frac{e^{i\gamma_m |p|}}{2i\gamma_m}, \quad f_k^{(m)}(p) = \frac{e^{i\beta_k^{(m)}|p|}}{i\beta_k^{(m)}} + \frac{e^{-\alpha_k |p|}}{\alpha_k}, \quad k \neq 0, \quad \alpha_k = \frac{\pi k}{d},$$

$$\beta_k^{(m)} = \sqrt{\gamma_m^2 - \alpha_k^2}, \ \gamma_m > \alpha_k, \ \beta_k^{(m)} = \sqrt{\alpha_k^2 - \gamma_m^2}, \ \alpha_k > \gamma_m, \ m = 1, 2.$$

Фигурирующий в (6.8) ряд сходится равномерно и абсолютно: в точке приложения источника общий член ряда ведёт себя как $\frac{1}{k^3}$. При указанном выборе $\beta_k^{(m)}$ для распространяющихся мод знаки групповой и фазовой скоростей в каждой моде совпадают во всей области частот. Одинаковая направленность фазовой скорости и скорости переноса энергии в гармонической волне обеспечивают выполнение условий

излучения на бесконечности [202].

Аналогичные представления можно получить и в случае полупространства [213, 214].

6.3.3 Система цилиндрических неоднородностей в волноводах

Представляет интерес исследование динамической напряженности нерегулярных волноводов при распространении в них гармонических волн сдвига. Предполагаем, что стенки волновода $x=0$ и $x=d$ свободны от сил ($A=-1$) или защемлены ($A=1$), а нерегулярность волновода вызвана наличием внутри него цилиндрических полостей, жёстких или упругих включений, криволинейных жёстких вставок или трещин-разрезов (рис. 6.3). В качестве возбуждающей нагрузки может выступать излучающаяся из бесконечности гармоническая SH-волна или линейный гармонический источник.

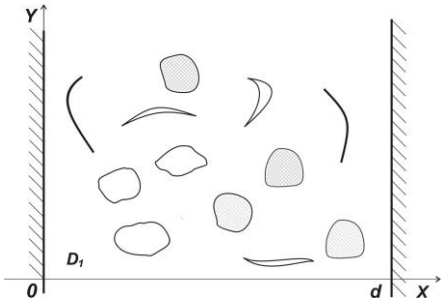


Рис. 6.3.

Следуя выбранной методике исследования, в интегральных представлениях (6.1) и (6.2) вместо функций источника G_1 и G_2 возьмём функцию Грина для соответствующего волновода ($A=-1$ или $A=1$). Имеем [183]:

$$G(x,y,\xi,\eta)=\frac{1-A}{2\pi}\ln 2+\frac{1}{2\pi}\ln\left|\sin\frac{\pi(z-\zeta)}{2d}\right|-\frac{A}{2\pi}\ln\left|\sin\frac{\pi(z+\bar{\zeta})}{2d}\right|+$$

$$+\frac{1}{d}\sum_{k=0}^{\infty}f_k^{(m)}(y-\eta)\begin{Bmatrix}\cos\alpha_k x\cos\alpha_k\xi\\ \sin\alpha_k x\sin\alpha_k\xi\end{Bmatrix},\ m=1,2. \qquad (6.9)$$

Здесь в фигурных скобках верхние значения соответствуют случаю волновода со свободными от сил стенками ($A=-1$), нижние – случаю волновода с защемлёнными стенками ($A=1$).

При указанном в (6.8) выборе ветви корня в выражении для $\beta_k^{(m)}$ излучаемая энергия ограничена при $y \to \pm\infty$ и волновое поле внутри волновода носит характер расходящихся волн, что соответствует условиям излучения (зависимость от времени $e^{-i\omega t}$). Точки $\alpha_k = \gamma_m$ являются точками возникновения новых волноводных распространяющихся гармоник. В случае защемлённого волновода ($A=1$) гармоники начинают возникать при $\gamma_m > \alpha_1$ и их число равно $\left[\frac{\gamma_m d}{\pi}\right]$; если стенки волновода свободны от сил ($A=-1$), то распространяющиеся волны наблюдаются при любой частоте и их число равно $1+\left[\frac{\gamma_m d}{\pi}\right]$ ($[x]$ означает целую часть числа x).

При подстановке функций Грина (6.9) в интегральное представление (6.1) автоматически будут удовлетворяться разрешающие уравнения Гельмгольца, обеспечиваться выполнение условий излучения на бесконечности и граничных условий на стенках волновода. Кроме того, в случае $L^{(4)}$ автоматически обеспечивается непрерывность перемещения и скачок производной по нормали от перемещения на системе криволинейных жёстких вставок. Аналогично в случае $L^{(5)}$ представление (6.2) автоматически даёт скачок перемещения на системе криволинейных трещин-разрезов. Подстановка интегральных представлений в граничные условия на L соответствующих краевых задач сводит последние к системе интегральных уравнений. Тип и структура этих интегральных уравнений описаны выше. Если используются модифицированные граничные условия по перемещениям (дифференцирование по дуговой координате s_0), к возникающим СИУ 1-го рода необходимо присовокупить дополнительные интегральные равенства, обеспечивающие однозначность решений.

6.4. Плоская деформация

Под плоской деформацией понимают напряженно-

деформированное состояние цилиндрического тела, нагруженного по боковой поверхности усилиями, действующими в плоскости поперечного сечения. В предположении, что ось цилиндра направлена вдоль оси 0Z декартовой прямоугольной системы координат, перемещение вдоль этой оси V_3 равно нулю, а перемещения V_1 и V_2 и компоненты тензора напряжений $\sigma_{i,j}$ (i,j = 1,2) зависят только от координат x и y. Отсюда немедленно следует, что для изотропного тела $\sigma_{13} = \sigma_{23} = 0$, а $\sigma_{33} = \nu(\sigma_{11} + \sigma_{22})$, где ν - коэффициент Пуассона среды.

6.4.1. Система неоднородностей в неограниченной среде

Пусть в неограниченной изотропной среде имеется система бесконечных вдоль оси OZ цилиндрических неоднородностей (рис. 6.1), причём внешнее поле перемещений действует перпендикулярно оси OZ. При таких предположениях мы находимся в условиях плоской деформации. В качестве внешнего воздействия будем рассматривать набегающую на цилиндры из бесконечности монохроматическую волну расширения-сжатия ($\tau = const$)

$$V_0^{(1)} = 0,\ V_0^{(2)} = \varpi e^{-i\gamma_1^{(1)} y},\ \gamma_1^{(1)} = \frac{\omega}{c_1^{(1)}},\ c_1^{(1)} = \sqrt{\frac{\lambda_1 + 2\mu_1}{\rho_1}} \qquad (6.10)$$

или волну сдвига ($\tau = const$)

$$V_0^{(1)} = \varpi e^{-i\gamma_1^{(2)} y},\ V_0^{(2)} = 0,\ \gamma_1^{(2)} = \frac{\omega}{c_1^{(2)}},\ c_1^{(2)} = \sqrt{\frac{\mu_1}{\rho_1}}. \qquad (6.11)$$

Здесь $c_1^{(1)}$ и $c_1^{(2)}$ – скорости продольной и поперечной волн в матрице (область D_1). Аналогичные скорости внутри однородных волокон (область D_2) будем обозначать $c_2^{(1)}$ и $c_2^{(2)}$ (λ_1, μ_1, ρ_1 и λ_2, μ_2, ρ_2 – коэффициенты Лямэ, а также плотности матрицы и инородных включений соответственно).

При взаимодействии приходящей волны с цилиндрами возникают отражённые и проходящие внутрь цилиндров (если они являются

упругими включениями) волны двух типов: продольные и поперечные, причём другие типы волн не образуются [179]. Пусть $V_k^{(1)}$ и $V_k^{(2)}$ – амплитуды перемещений отражённого ($k=1$) и проходящего ($k=2$) полей соответственно. Тогда компоненты общего поля перемещений $\vec{V}=\vec{V}^{+}$ при $k=1$ и $\vec{V}=\vec{V}^{-}$ при $k=2$ равны

$$V^{(1)}=V_k^{(1)}+(2-k)V_0^{(1)},\ V^{(2)}=V_k^{(2)}+(2-k)V_0^{(2)}. \qquad (6.12)$$

В случае гармонической зависимости от времени ($e^{-i\omega t}$) компоненты векторов амплитуд перемещений $\vec{V}$ удовлетворяют уравнениям движения [188]

$$(\lambda+2\mu)\frac{\partial^2 V^{(1)}}{\partial x^2}+\mu\frac{\partial^2 V^{(1)}}{\partial y^2}+(\lambda+\mu)\frac{\partial^2 V^{(1)}}{\partial x\partial y}+\rho\omega^2 V^{(1)}=0, \qquad (6.13)$$

$$\mu\frac{\partial^2 V^{(2)}}{\partial x^2}+(\lambda+2\mu)\frac{\partial^2 V^{(2)}}{\partial y^2}+(\lambda+\mu)\frac{\partial^2 V^{(2)}}{\partial x\partial y}+\rho\omega^2 V^{(2)}=0$$

при соответствующем задании параметров матрицы и включений.

На бесконечности ($y\to+\infty$) поле рассеянной волны должно выполнять условия излучения, т.е. компоненты $V_1^{(1)}$ и $V_1^{(2)}$ должны представлять собой расходящиеся волны.

Будем исходить из того, что амплитудные значения компонент тензора напряжений связаны с амплитудами перемещений формулами [179]

$$\tau_{11}+\tau_{22}=2(\lambda+\mu)\left(\frac{\partial\left(V^{(1)}+iV^{(2)}\right)}{\partial z}+\frac{\partial\left(V^{(1)}-iV^{(2)}\right)}{\partial\bar{z}}\right), \qquad (6.\ 14)$$

$$z=x+iy,\ \bar{z}=x-iy,\ \tau_{22}-\tau_{11}+2i\tau_{12}=-4\mu\frac{\partial\left(V^{(1)}-iV^{(2)}\right)}{\partial z},$$

$$\tau_{22}-\tau_{11}-2i\tau_{12}=-4\mu\frac{\partial\left(V^{(1)}+iV^{(2)}\right)}{\partial\bar{z}}.$$

Обозначим через S_1 и S_2 амплитуды тангенциальной и нормальной компонент вектора напряжений $\vec{S}$ на L. Тогда в произвольной точке

$\zeta_0 = \xi_0 + i\eta_0 \in L$ эти напряжения выражаются через компоненты тензора напряжений следующим образом:

$$2i(S_1 + iS_2) = (\tau_{11} + \tau_{22})e^{i\varphi_0} + (\tau_{22} - \tau_{11} - 2i\tau_{12})e^{-i\varphi_0}, \qquad (6.15)$$

$$-2i(S_1 - iS_2) = (\tau_{11} + \tau_{22})e^{-i\varphi_0} + (\tau_{22} - \tau_{11} + 2i\tau_{12})e^{i\varphi_0},$$

где φ_0 – угол положительной касательной к L в точке $\zeta_0 \in L$ с осью OX.

На границе цилиндров представляют интерес распределения амплитуд напряжений

$$\tau_{n_0} = S_1 \sin\varphi_0 - S_2 \cos\varphi_0, \ \tau_{n_0 s_0} = S_1 \cos\varphi_0 + S_2 \sin\varphi_0,$$

$$\tau_{s_0} = (\tau_{11} + \tau_{22}) - \tau_{n_0} \qquad (6.16)$$

Граничные условия на L имеют следующий вид.

1. На $L^{(1)}$: $\vec{S}^+ = \vec{S}^-$, $\vec{V}^+ = \vec{V}^-$ условия типа склейки компонент вектора напряжений $\vec{S}$ и вектора перемещений $\vec{V}$ со стороны матрицы и включений.

2. На $L^{(2)}$: $\vec{S}^+ = 0$.

3. На $L^{(3)}$: $V^{(1)} = B_1 - \omega_0\eta_0$, $V^{(2)} = B_2 + \omega_0\xi_0$, $(\xi_0, \eta_0) \in L^{(3)}$.

4. На $L^{(4)}$: $V^{(1)} = B_1 - \omega_0\eta_0$, $V^{(2)} = B_2 + \omega_0\xi_0$, $(\xi_0, \eta_0) \in L^{(4)}$.

5. На $L^{(5)}$: $\vec{S}^{\pm} = 0$ – берега трещин-разрезов свободны от сил.

Здесь в случаях 3 – 4 предполагается, что однородные жёсткие включения перемещаются и поворачиваются вместе с матрицей; B_1 и B_2 – амплитуды поступательного движения, ω_0 – амплитудное значение жёсткого поворота включений.

В случае 3, используя второй закон Ньютона, получим уравнения, которые характеризуют поступательное движение однородных жёстких включений:

$$\int_{L_k^{(3)}} S_r ds_0^k = -\omega_0^2 \rho_0 S_0 B_r , \qquad r = 1, 2, \; k = 1, \ldots, K . \qquad (6.17)$$

Уравнение, описывающее вращательное движение включений, имеет вид:

$$\int_{L_k^{(3)}} \left(S_1\left(\eta_0^k - y\right) - S_2\left(\xi_0^k - x\right) \right) ds_0^k = -\omega^2 J_A \omega_0 , \qquad (6.18)$$

где ρ_0 – плотность, S_0 – площадь однородных жёстких включений, J_A – момент инерции включений относительно произвольной точки $A(x,y)$.

В случае 4 считаем, что результирующие силы и крутящий момент, действующие на $L^{(4)}$, равны нулю, т.е.:

$$\int_{L_k^{(4)}} S_r ds_0^m = 0 , r = 1,2 ; \int_{L_m^{(4)}} \left(S_1\left(\eta_0^m - y\right) - S_2\left(\xi_0^m - x\right) \right) ds_0^m = 0 , m = 1, \ldots, M .$$

(6.19)

Как и в случае антиплоской деформации, решение краевых задач 1 – 4 основано на интегральных представлениях амплитуд перемещений отражённого ($k = 1$) и проходящего внутрь упругих включений ($k = 2$) полей в виде потенциалов типа простого слоя [179]:

$$V_k^{(m)}(x,y) = \int_L \left(f_1^{(k)}(s) G_{m1}^{(k)}(x,y,\xi,\eta) + f_2^{(k)}(s) G_{m2}^{(k)}(x,y,\xi,\eta) \right) ds , m = 1, 2 .$$

(6.20)

Здесь $f_1^{(k)}(s)$ и $f_2^{(k)}(s)$ – независимые функции ($k = 1, 2$); $G_{1j}^{(k)}$ и $G_{2j}^{(k)}$ – функции Грина ($j = 1, 2$), представляющие собой амплитуды перемещений j-го состояния в точке $(x,y) \in D_k$ при действии гармонической сосредоточенной силы, приложенной в точке $\zeta = \xi + i\eta \in L$ и направленной вдоль OX (1-ое состояние) или вдоль оси OY (2-ое состояние). В случае 4 представления (6.20) автоматически удовлетворяют условию непрерывности перемещений при переходе через $L^{(4)}$ и обеспечивают скачки напряжений: $[S_1] = f_1^{(1)}(s)$, $[S_2] = f_2^{(1)}(s)$.

При решении краевой задачи 5 используем интегральные

представления типа двойного слоя [189]:

$$V^{(m)}(x,y)=\int_{L^{(5)}}\left(g_1(s)S_{m1}(x,y,\xi,\eta)+g_2(s)S_{m2}(x,y,\xi,\eta)\right)ds\,. \qquad (6.21)$$

Здесь $g_1(s)$ и $g_2(s)$ – неизвестные функции; S_{1j} и S_{2j} – функции Грина ($j=1,2$), представляющие собой амплитуды напряжений в области D_1 при действии сосредоточенной силы, приложенной в точке $\zeta\in L$ и направленной вдоль оси OX (1-ое состояние) или вдоль оси OY (2-ое состояние). Представления (21) автоматически удовлетворяют непрерывность напряжений при переходе через $L^{(5)}$ и обеспечивают скачки перемещений: $\left[V^{(1)}\right]=g_1(s)$, $\left[V^{(2)}\right]=g_2(s)$.

Функции (6.20) и (6.21) удовлетворяют уравнениям движения (6.13) и условиям излучения на бесконечности при соответствующем выборе указанных функций Грина. Таким образом, решение сформулированных задач дифракции сводится к нахождению неизвестных плотностей интегральных представлений по заданным граничным условиям на L.

Для построения эффективного численного алгоритма при удовлетворении граничных условий по перемещениям на L вычислялись комбинации $V^{(1)}\pm iV^{(2)}$, которые дифференцировались по дуговой координате s_0. Аналогично граничные условия по напряжениям удовлетворялись для комбинаций $S_1\pm iS_2$. Такой подход позволил получить следующие системы интегральных уравнений.

1. На контурах упругих включений $L^{(1)}$ при выполнении граничных условий по напряжениям возникают СИУ 2-го рода, а удовлетворение модифицированных граничных условий по перемещениям приводит к СИУ 1-го рода. Необходимые дополнительные условия для разрешимости последних вытекают из равенства средних амплитуд перемещений на $L^{(1)}$.

2. На контурах полостей $L^{(2)}$ соответствующая краевая задача сводится к системе СИУ 2-го рода, которая имеет единственное

решение.

3. На замкнутых контурах жестких включений $L^{(3)}$ модифицированное граничное условие по перемещениям приводит к системе СИУ 1-го рода, к которым необходимо присовокупить дополнительные интегральные равенства, вытекающие из (6.17), (6.18).

4. На разомкнутых криволинейных контурах жёстких вставок $L^{(4)}$ также возникают СИУ 1-го рода. Здесь плотности интегральных уравнений являются скачками напряжений на $L^{(4)}$, и они имеют корневую особенность на концах разомкнутых контуров. Необходимые дополнительные условия возникают из условия равенства нулю главного вектора сил, действующих на берегах $L^{(4)}$. Жёсткий поворот однородных криволинейных жёстких вставок определяется из условия равенства нулю главного момента сил, возникающих на $L^{(4)}$ (6.19).

5. На системе криволинейных трещин-разрезов $L^{(5)}$ краевая задача сводится к сингулярным интегро-дифференциальным уравнениям 1-го рода. Дополнительные условия для их разрешимости вытекают из условия равенства нулю скачков перемещений на концах трещин-разрезов:

$$\int\limits_{L_n^{(5)}} \frac{dg_1}{ds^n} ds^n = 0\,, \quad \int\limits_{L_n^{(5)}} \frac{dg_2}{ds^n} ds^n = 0\,, \; n = 1,\ldots,N\,. \tag{6.22}$$

Здесь производные от скачков перемещений на $L^{(5)}$ имеют корневую особенность на концах разрезов.

6.4.2. Периодическая система неоднородностей

Элементы конструкций, используемые в технике и строительстве, работают под действием циклических нагрузок и часто содержат большое количество неоднородностей, ориентированных вдоль некоторой оси. Такую конечную решётку можно аппроксимировать бесконечной, а именно периодической решёткой, составленной из упругих и жёстких включений, полостей, криволинейных жестких вставок или трещин-

разрезов. Одним из наиболее распространённых и эффективных методов исследования периодических плоских задач дифракции в среднечастотном диапазоне волн является метод интегральных уравнений.

Рассмотрим неограниченную изотропную среду с плотностью ρ_1 и коэффициентами Ляме λ_1, μ_1, в которой содержится 2d-периодическая вдоль оси OX система цилиндрических неоднородностей указанного типа (рис. 6.2). Предполагаем, что из бесконечности на периодическую решётку набегает гармоническая P-волна (6.10) или SV-волна (6.11). Необходимо исследовать возникающую плоскую периодическую задачу дифракции методом интегральных уравнений и оценить динамические и механические характеристики периодической решётки.

При решении периодических краевых задач 1 – 4 интегральные представления амплитуд перемещений будем искать в виде (6.20), причём здесь следует положить: $G_{1j}^{(k)}$ и $G_{2j}^{(k)}$ – компоненты матриц Грина в области D_1 ($k=1$) и в области D_2 ($k=2$), которые представляют собой амплитуды перемещений в точке $(x,y)\in D_k$ при действии периодической системы гармонических сил, сосредоточенных в точках $(\xi+2dp,\eta)\in L_p$ ($p=0,\pm1,\pm2,\ldots$) и направленных вдоль оси OX ($j=1$) или вдоль оси OY ($j=2$) [153].

При решении периодической краевой задачи 5 будем использовать интегральные представления (6.21), если положить в них, что S_{1j} и S_{2j} – компоненты матрицы Грина, представляющие собой амплитуды напряжений в матрице (область D_1) при действии периодической системы гармонических сил, сосредоточенных в точках $(\xi+2dp,\eta)\in L_p$ ($p=0,\pm1,\pm2,\ldots$) и направленных вдоль оси OX ($j=1$) или вдоль оси OY ($j=2$) [180].

Подстановка интегральных представлений (6.20) или (6.21) в заданные краевые условия на L приводит к системе СИУ указанных выше

типов. В случае СИУ 1-го рода к ним необходимо присовокупить дополнительные условия, вытекающие из механической постановки исследуемой краевой задачи.

6.5. Схема вычислений

Как видно из описания ПрО, каждая локальная задача сведена к системе СИУ, структура каждого из которых строго типизирована. Более того, методика численной реализации позволяет использовать комбинирование типовых вычислительных процедур. Опишем общую схему численной реализации.

Элементы матрицы СЛАУ, к которой, в конечном итоге, сводятся система СИУ, являются результатом дискретизации контуров. Очевидно, что размер матрицы пропорционален числу неоднородностей. Применим распараллеливание алгоритма, в котором каждый элемент матрицы определяется координатами узлов дискретизации.

6.5.1. Схема параллельного алгоритма

Как показано в [185], данный метод в вычислительном смысле сводится к обходу каждого контура по точкам коллокации внеинтегральной переменной ζ_{k0} и одновременному же обходу каждого контура по аналогичным либо иным узлам переменной интегрирования ζ_k.

Рис. 6.4.

Параллельно-конвейерная схема вычислений показана на рис. 6.4.

Тут приведена пропорция интервалов времени вычислений на: синтез массивов исходных данных (время t_0 при количестве процессов P_1), синтез матрицы СЛАУ (время t_1 при количестве процессов P_1), решение СЛАУ методом Гаусса (t_2 – оптимальное время вычислений при оптимальном числе процессов P_0), синтез массивов итоговых решений (время t_3). Первый, второй и четвертый этапы макроконвейера не требуют пересылок данных, что означает независимость вычислений. На третьем этапе для решения СЛАУ существует оптимальное число процессов, определяемое спецификой матрицы. Это означает, что для 1, 2 и 4 этапов алгоритма оптимальным является число процессов, соответствующее числу коэффициентов СЛАУ.

В данной методике решения всех перечисленных краевых задачи основной операцией является определение текущего расстояния между точками коллокации и интегрирования, заданного на множестве значений параметрических координат неоднородностей. Указанное расстояние является аргументом функции Грина. И поскольку комбинации самих функций Грина и коэффициентов при них являются элементами матрицы СЛАУ, указанная процедура может быть базовой при разработке CASE-средства. Тем более, что, как показано в [185, 213, 214], алгоритм хорошо масштабируется по вычислительным узлам.

Вычислительный процесс решения СЛАУ, в свою очередь, распараллеливается согласно [314]. Параллельное вычисление итоговых искомых характеристик осуществляется путем подстановки массивов значений неизвестных функций $f_k(\beta_p)$ в интегральные представления решений аналогично процедурам формирования матрицы СЛАУ. При этом в зависимости от операционной среды, доступной пользователю, может применяться два типа формирования матрицы – поэлементное и построчное. Как показано на рис. 6.4, более оптимальным является

поэлементное параллельное формирование матрицы СЛАУ, когда число узлов равно числу элементов матрицы. Однако для решения СЛАУ эффективнее использовать построчное распараллеливание, когда пересылки и вычисления находятся в балансе. Таким образом, наиболее прогрессивной является операционная среда, в которой пользователь имеет возможность сочетать оба механизма, гибко изменяя число используемых узлов в соответствии с этапом вычислений.

6.5.2. Схема ПрО и CASE-оболочки

Из описания ПрО, приведенного выше, следует, что структурирование является естественным. Очевидно, что итоговое множество данных - массив значений искомых функций - является результатом взаимодействия всех базовых компонентов: функций Грина со своими аргументами, правых частей, дополнительных условий или внеитнегральных членов, геометрических характеристик исследуемых задач и т.п. Таким образом, функционирование приложения такой БД – это работа со списками этих сущностей-объектов.

Приведенный выше список задач механики сплошных сред может быть значительно расширен. Так к типам нагрузки могут быть добавлены еще и изгибные волны, а категория волн может быть расширена на «нестационарные». Область распространения может быть замкнутой. Свойства среды распространения могут быть не только изотропными, а и ортотропными и даже анизотропными. Модели описания распространения, например, изгибных волн в пластинах могут быть также различными - от модели Кирхгофа-Лява [181] до уточнённых моделей типа Тимошенко [177]. А, как показано в [20], на стыке механики и электрофизики исследуются новые среды и поля. Однако общим вычислительным базисом для этого многообразия задач является метод СИУ.

В [221, 224, 225] описана структура каркасной [207] инструментальной программной среды, которая масштабируется метаданными. Как показал анализ ПрО, для всех задач математической физики, решаемой методом СИУ, структура алгоритма и большинство вычислительных процедур типизируемы. Значит, для проектирования нового CASE-средства может быть применен РК. Рис. 6.5 показывает, что даже при условии неполноты анализа ПрО и незавершенности проекта, CASE-средство, основанное на каркасном шаблоне, может быть модифицируемо и развиваемо.

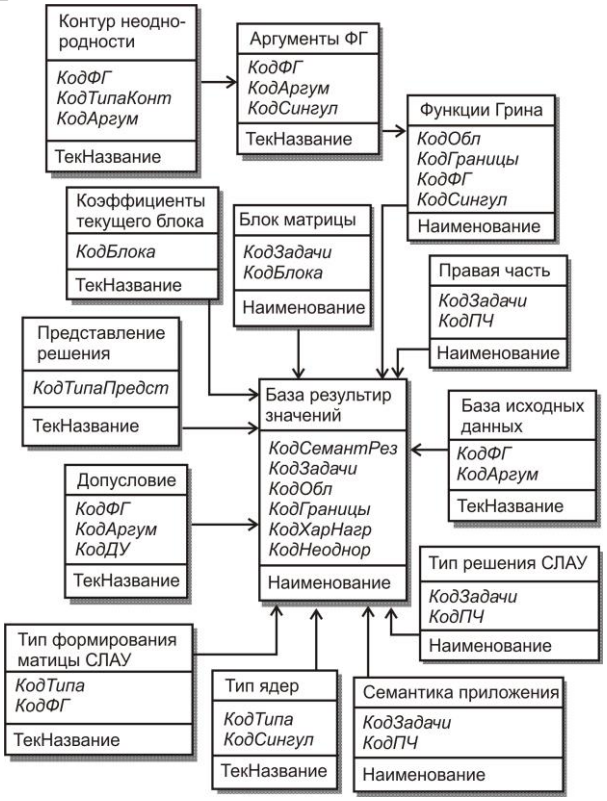


Рис. 6.5.

6.6. Результаты численных исследований

6.6.1. Система упругих включений

Привлечение формулы Сохоцкого – Племеля [202] для вычисления предельных значений интегралов типа Коши, возникающих при удовлетворении граничных условий (6.5) с учетом соотношений (6.6) – (6.8), приводит к системе СИУ относительно неизвестных функции $f_k(s)$:

$$\int_L f_1(s)[g(s,s_0)+B_1(s,s_0)]ds-\int_L f_2(s)[g(s,s_0)+B_2(s,s_0)]ds=N_k(s_0) \qquad (6.23)$$

$$\frac{1}{2}[\mu_1 f_1(s_0)+\mu_2 f_2(s_0)]+\int_L [f_1(s)E_1(s,s_0)-f_2(s)E_2(s,s_0)]ds=K_k(s_0)$$

$$g=\frac{1}{2\pi}Re\left(\frac{e^{i\varphi_0}}{\zeta-\zeta_0}\right),\ \zeta_0-\zeta=r_0e^{i\alpha_0},\ \zeta_0-\zeta=r_{10}e^{i\alpha_{10}};\zeta_0-\bar{z}_0=R_{10}e^{i\beta^*_{10}};\ \zeta_0-z_0=R_0e^{i\beta^*_0},$$

$$E_1=\gamma_1\mu_1H_1^{(1)}(\gamma_1r_0)sin(\varphi_0-\alpha_0)/4i,\ E_2=\gamma_2\mu_2[H_1^{(1)}(\gamma_2r_0)sin(\varphi_0-\alpha_0)+H_1^{(1)}(\gamma_2r_{10})sin(\varphi_0-\alpha_{10})]/4i,$$

$$B_1=\gamma_1H_1(\gamma_1r_0)cos(\varphi_0-\alpha_0)/4i,\ B_2=\gamma_2[H_1(\gamma_2r_0)cos(\varphi_0-\alpha_0)+H_1^{(1)}(\gamma_2r_{10})cos(\varphi_0-\alpha_{10})]/4i,$$

$$N_1=-i\gamma_2[W_0(s_0)cos(\varphi_0-\psi)+W_3(s_0)cos(\varphi_0+\psi)],\ N_2=P\gamma_2[H_1^{(1)}(\gamma_2R_0)cos(\varphi_0-\beta^*_0)+H_1^{(1)}(\gamma_2R_{10})cos(\varphi_0-\beta^*_{10})]/4i\mu_2,$$

$$K_1=-i\gamma_2\mu_2[W_0(s_0)sin(\varphi_0-\psi)+W_3(s_0)sin(\varphi_0+\psi)],\ K_2=P\gamma_2[H_1^{(1)}(\gamma_2R_0)sin(\varphi_0-\beta^*_0)+H_1^{(1)}(\gamma_2R_{10})sin(\varphi_0-\beta^*_{10})]/4i.$$

Здесь функции $N_k(s_0)$ и $K_k(s_0)$ отвечают случаям (1) и (2) соответственно, ядра $g(s,s_0)$ – сингулярны, ядра $B_k(s,s_0)$ и $E_k(s,s_0)$ – непрерывны (k=1, 2). В первой группе (9) интегральные уравнения являются сингулярными, а во второй – уравнениями Фредгольма 2-го рода.

Для выделения единственного решения СИУ присовокупим к ней дополнительные условия:

$$\int_{L_j} W_1 ds_0=\int_{L_j}(W_2+W_0+W_3)ds_0, \qquad (6.24)$$

выполнение которых обеспечивает непрерывность перемещений на каждом из контуров L_j.

Параллельно-конвейерная схема вычислений представлена в [230]. Первый, второй и четвертый этапы макроконвейера не требуют пересылок данных, что означает независимость вычислений. На третьем этапе для решения СЛАУ существует оптимальное число процессов, определяемое спецификой матрицы.

Для алгоритма решения СЛАУ искомых СИУ оптимальным числом оказалось 150-200 процессов при заданной точности 10^{-8}. Увеличение числа процессов приводит к незначительному снижению суммарного времени вычислений за счет части алгоритма без пересылок, но также и к приросту вычислительных расходов на балансировку загрузки процессоров.

На рис. 6.6. показан график зависимости времени кластерных вычислений массива контурных напряжений на ромбическом упругом включении от числа процессов для одного варианта нагрузки. По графику

видно, что как и в [230], весь алгоритм хорошо масштабируется.

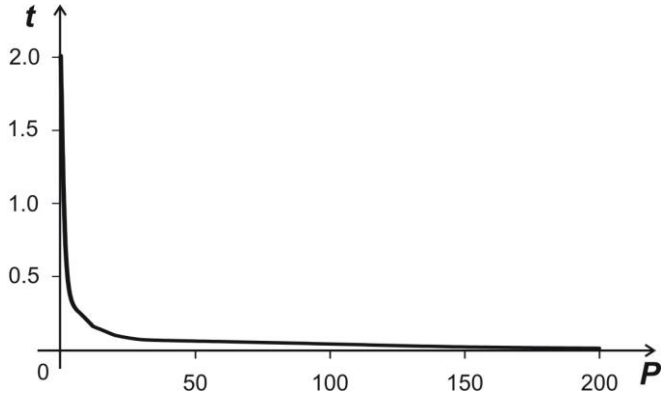


Рис. 6.6.

Так как для данной методики решения краевой задачи основная операция при вычислении каждого элемента матрицы – это определение расстояния между точками контуров (рис. 6.7.), которое является аргументом цилиндрических функций Ханкеля, а также вычисление самих этих функций и коэффициентов при них, на каждом клоне хоста запускаются цикл процедур определения указанных коэффициентов. Итоговая матрица собирается по факту завершения последнего.

Вычислительный процесс решения СЛАУ также распараллеливается согласно [230]. Параллельное вычисление итоговых искомых характеристик осуществляется путем подстановки массивов значений неизвестных функций $f_k(\beta_p)$ в представление (6.7) аналогично процедурам формирования матрицы СЛАУ.

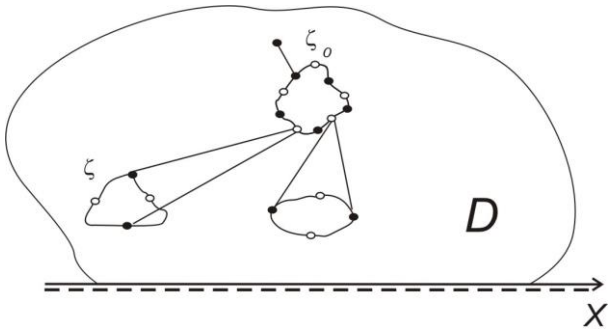


Рис. 6.7.

В исследовании достигалась точность вычислений порядка 10^{-10}. Такая точность обеспечена следующим: высокая сходимость самого алгоритма, разрешающая способность компиляторов языков высокого уровня и

разрешающая способность операционных сред. Метод СИУ обеспечивает быструю сходимость решения, а также функциональную зависимость стабилизации знаков результирующих данных от увеличения числа точек коллокации. Для указанной точности в описанной задаче достаточно 700–1000 точек коллокации каждого контура.

С целью исследования сходимости построенного алгоритма рассмотрим случай нормального падения ($\psi = \pi/2$) волны сдвига (6.1) на систему однородных эллиптических или ромбических включений, расположенных вдоль одной линии на одинаковом расстоянии d один от другого (рис. 6.8.).

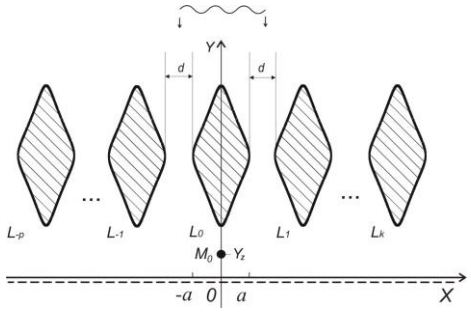


Рис. 6.8.

Используем известные [77] параметрические уравнения для задания основного контура L_0:

$$\xi(\beta) = b\sin\beta - \nu\sin 3\beta, \quad \eta(\beta) = a\cos\beta + \nu\cos 3\beta, \quad 0 \le \beta \le 2\pi, \tag{6.25}$$

где при $\nu = 0.14036$ контур имеет вид ромба со скругленными точками возврата. А в случае $\nu = 0$ контур имеет эллиптическую форму. Остальные контуры для простоты будем располагать симметрично относительно оси Y. При условии, что физические характеристики всех включений одинаковы, рассматриваемая дифракционная задача обладает свойством симметрии, что позволяет осуществлять первичное самотестирование получаемых результатов.

В ходе численной реализации вычислялись безразмерные контурные напряжения $\sigma_\beta = \tau_s / \mu_2$ и $\sigma_n = \tau_n / \mu_2$. Точность вычислений проверялась путем сравнения результатов при различных значениях N. Проводилось также сравнение полученных результатов с результатами, приведенными в [230]

как для случая системы эллиптических или ромбических отверстий (как частного случая упругого включения), так и для одиночного упругого включения.

Применение метода параллельных вычислений, проведенного на кластере «Инпарком–256», позволило подтвердить вывод о том, что сходимость решения СИУ практически не зависит от числа отражателей.

Численное исследование показало, что в случае границы, свободной от сил, при воздействии из бесконечности SH-волны в описанной системе эффект насыщения [145] наблюдается не строго (как и в [230]). И хотя при линейном и симметричном относительно нагрузки расположении вдоль границы физически одинаковых включений для усредненного исследования достаточно не более 11 неоднородностей, все же при дальнейшем наращивании числа включений наблюдаются незначительные пульсации в распределении напряжений. Обусловленность матриц при этом проверялась на основании алгоритма, описанного в [315].

При фиксированной размерности матриц СЛАУ число включений не влияет на оптимальное число процессов, поскольку в СИУ каждый контур неоднородности является частью суммарного контура интегрирования. Поэтому при прочих равных условиях свойства СЛАУ, полученных как для одного включения, так и для нескольких, не изменяются. Как и в случае [145, 219, 220, 230], от числа включений не зависит и сходимость алгоритма.

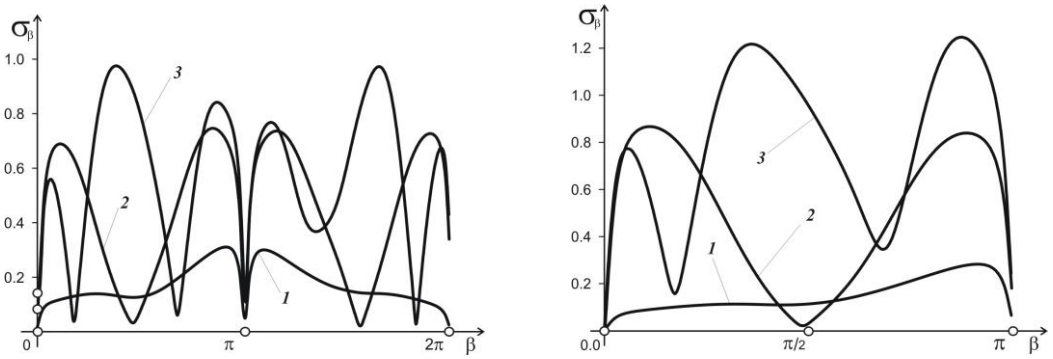
В работе проводились вычисления контурных напряжений σ_β и σ_n вдоль контуров центрального L_0 и крайнего L_k включений (рис. 6.7) в случае решетки, состоящей из нечетного числа неоднородностей ($p = k$). Отсчет угла β ведется от нуля (теневая точка) до π (лобовая точка) для центрального включения (учитывается симметрия в случае нормального распределения волны сдвига) и от 0 до 2π – для крайних волокон (в силу симметрии и равенства упругих постоянных распределения напряжений на контурах L_k и L_{-k} зеркальны). Рассматривается случай ромбов, вытянутых вдоль набегающей волны. Для всех рисунков $\mu_1/\mu_2=5.0$, $\rho_1/\rho_2=2.0$, $b/a=2.5$, а

расстояние от границы до центрального волокна $h=4$ (на рис. 6.7 не показано).

На рис. 6.9 а, б показаны распределения напряжения σ_β вдоль контура крайнего слева и центрального волокон соответственно в случае решетки, состоящей из трех ромбов. На рис. 6.10 а, б показаны распределения напряжений σ_n вдоль тех же контуров. Воздействие – волна из бесконечности. Значения безразмерного волнового числа $\gamma_2 a$ для кривой 1 составляет 0.3, для кривой 2 - 0.9, для кривой 3 – 1.5.

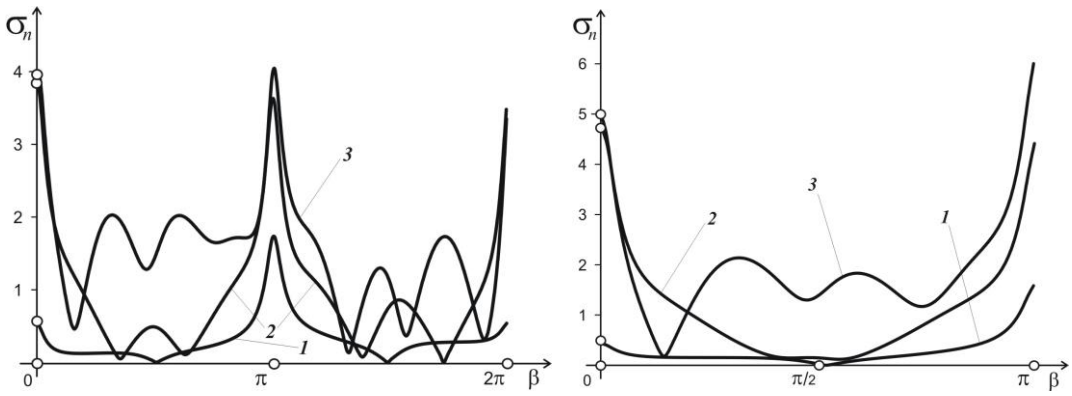
Результаты показывают, что, чем выше частота колебаний, тем больший вклад в напряженно-деформированное состояние контура волокна вносит напряжение σ_n. Это говорит о том, что разрушение, например, в композиционном материале может происходить вследствие отрыва по границам раздела фаз.

Если в теневой ($\beta = 0$) и лобовой ($\beta = \pi$) точках $\sigma_\beta = 0$, то в зоне соскальзывания с увеличением $\gamma_2 a$ число локальных максимумов σ_β также увеличивается, причем растет и максимальное значение σ_β. Напряжение же σ_n имеет обратный характер. Такой вывод полностью совпадает с результатами работы [178].



а) б)

Рис.6.9.



а) б)

Рис. 6.10.

6.6.2. Система упругих включений и отверстий

Проведем параметрическое исследование некоторых новых прикладных задач. С целью исследования сходимости построенного алгоритма рассмотрим случай нормального падения волны сдвига [230] на систему, состоящую из эллиптических или ромбических отверстий и упругих включений, поочередно расположенных в упругом полупространстве на одинаковом расстоянии d один от другого и ориентированных вдоль свободной от сил границы полупространства $y = 0$ (рис. 6.11.).

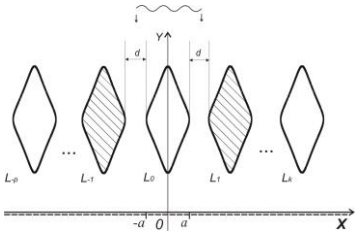


Рис. 6.11.

Используем известные [77] параметрические уравнения для задания основного контура L_0:

$$\xi(\beta) = b\sin\beta - \nu\sin 3\beta, \quad \eta(\beta) = a\cos\beta + \nu\cos 3\beta, \quad 0 \le \beta \le 2\pi, \tag{6.26}$$

где при ν = 0.14036 контур имеет вид ромба со скругленными точками возврата. А в случае ν = 0 контур имеет эллиптическую форму. Остальные контуры для простоты будем располагать симметрично относительно оси

Y. В этом случае рассматриваемая дифракционная задача обладает свойством симметрии, что позволяет осуществлять первичное самотестирование получаемых результатов.

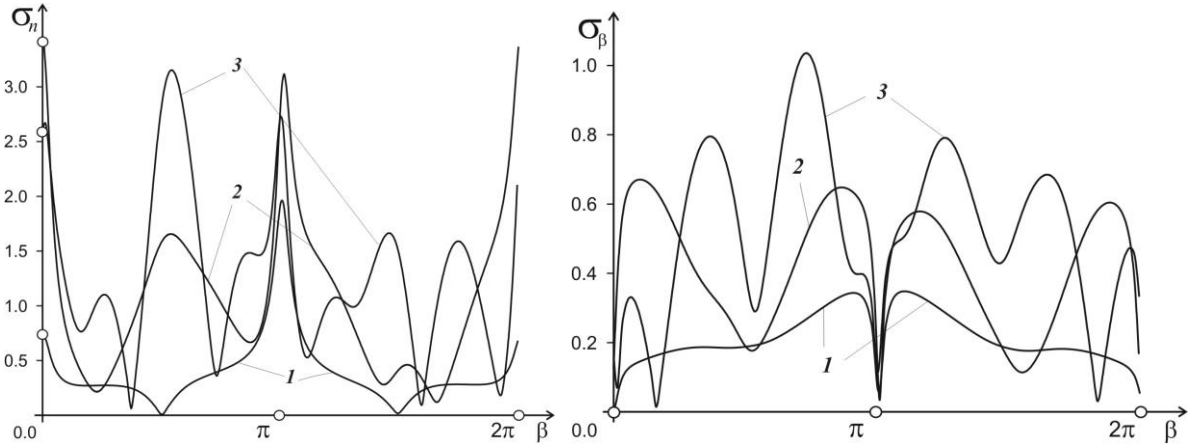
В ходе численной реализации вычислялись безразмерные напряжения $\sigma_\beta = \tau_s / \mu_1$ на контурах отверстий, а также безразмерные контурные напряжения $\sigma_\beta = \tau_s / \mu_1$ и $\sigma_n = \tau_n / \mu_1$ на контурах упругих включений. Точность вычислений проверялась путем сравнения результатов при различных значениях N. Проводилось также сравнение полученных результатов с результатами, приведенными в [214] как для случая системы ромбических отверстий в полупространстве со свободной от сил границей, так и для одиночного отверстия или упругого включения [178].

Численное исследование показало, что в полубесконечном случае с границей, свободной от сил, при воздействии на описанную систему SH-волной из бесконечности эффект насыщения [145], как и в [214], наблюдается не строго. И хотя при линейном и симметричном относительно нагрузки расположении неоднородностей вдоль границы для усредненного исследования достаточно не более 9 неоднородностей, все же при дальнейшем наращивании числа неоднородностей наблюдаются незначительные пульсации в распределении напряжений. Обусловленность матриц при этом проверялась на основании алгоритма, описанного в [315].

В работе проводились вычисления контурных напряжений σ_β вдоль контуров центрального L_0 и крайнего L_k включения (полого или упругого, рис. 6.11.) в случае решетки, состоящей из нечетного числа неоднородностей (p=k). Отсчет угла β ведется от нуля (теневая точка) до π

(лобовая точка) для центрального отверстия (учитывается симметрия в случае нормального падения волны сдвига) и от 0 до 2π – для крайних упругих включений (в силу симметрии распределения напряжений на контурах L_k и L_{-k} зеркальны). Рассматривается случай ромбов, вытянутых навстречу набегающей волне. Для всех графиков $\mu_1/\mu_2=5.0$, $\rho_1/\rho_2=2.0$, $b/a=2.5$, а расстояние от границы до центрального волокна $h=4$ (на рис. 6.10 не показано).

На рис. 6.12. показаны распределения безразмерных контурных напряжений σ_β и σ_n для крайнего ромбического упругого включения в случае 3-х неоднородностей и набегания из бесконечности волны сдвига в полупространства с границей, свободной от сил. Для кривых 1, 2 и 3 $\gamma_2 a$ равны 0,3, 0,9 и 1,5 соответственно.



а) б)

Рис. 6.12.

На рис. 6.13. показано распределение контурных напряжений σ_β на границе центрального ромбического отверстия в случае 3-х неоднородностей и набегания из бесконечности волны сдвига в полупространстве с границей, свободной от сил. Как и на рис. 6.28, тут кривые 1, 2, 3 соответствуют $\gamma_2 a$ = {0,3, 0,9, 1,5}.

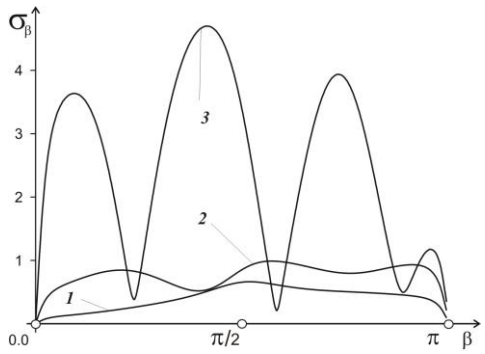


Рис. 6.13.

6.7. Выводы к шестому разделу

1. Как указывалось в [46, 230], в задачах математической физики параллельные алгоритмы позволяют значительно сократить время вычислений и более детально проанализировать характеристики исследуемых полей. Это важно, так как получение точных величин, например, максимумов напряжений вплоть до 10-го знака, а также точных координат их дислокации позволяет избежать разрушений конструкций, работающих в условиях динамических нагрузок.

2. Сочетание метода интегральных уравнений, позволяющего значительно ускорить решение задачи, и процедур распараллеливания, также приводящих к значительной экономии времени вычислений, существенно увеличивает эффективность предложенного алгоритма.

3. Построенные предложенным методом инструментальные программные средства синтеза приложений, численно решающие разнообразные интегральные уравнения, позволят своевременно и точно прогнозировать поведение различных систем с усложненными свойствами.

4. Анализ описанной ПрО показывает, что для широкого класса задач математической физики, решаемых методом СИУ, структура алгоритма и большинство вычислительных процедур типизируемы. Поэтому при рассмотрении вопроса проектирования инструментальных программных

средств (CASE-средств) [45, 109, 146], позволяющих синтезировать и сопровождать приложения, моделирующие динамическое поведение сложных механических систем, достаточно изучить несколько базовых алгоритмов [220, 230].

5. В задачах механики сплошных сред параллельные алгоритмы позволяют значительно сократить время вычислений и более детально проанализировать характеристики исследуемых полей. Это важно, так как получение точных величин, например, максимумов напряжений (вплоть до 8-го – 10-го знака), а также точных координат их дислокации позволяет избежать разрушений конструкций.

6. Построенные предложенным методом инструментальные программные средства синтеза приложений позволят своевременно и точно прогнозировать поведение различных систем с усложненными свойствами.

РАЗДЕЛ 7

РЕЗУЛЬТАТЫ ЧИСЛЕННЫХ ЭКСПЕРИМЕНТОВ

7.1. Введение к седьмому разделу

Концептуальная модель данных необходима проектировщику для обеспечения его потребности в синтезе максимально близкой схемы БД к реалиям ПрО, но, в то же время, и к формализму той логической модели, на которую опирается СУБД. Парадоксы, когда идеально логичная и максимально близкая к реалиям ПрО ER-диаграмма вынужденно разрушается нормализацией, и потому схема денормализуется и приобретает аномалии, не устраивают современного пользователя. Тратятся ресурсы на проектировку, затем на перепроектировку. Затем на исправление ошибок. Практическая апробация любой новой концептуальной модели, а тем более модели, которая претендует на роль более универсальной и более эффективной – это самый важный этап для подтверждения или опровержения теоретических закономерностей.

Существенным в описанном анализе является то обстоятельство, что каркас как таковой прямо не применяется. Он лишь символизирует совокупность таблиц для любой ПрО, давая проектировщику гарантии, что при полной декомпозиции сущностей полученная совокупность отношений будет иметь максимальную степень модифицируемости. Это означает, что каркасная логическая модель данных показывает направление анализа ПрО и выявляет схемы отношений. Очевидно, что в приведенной выше совокупности таблиц можно с помощью SQL-преобразований построить любую совокупность связей. Поэтому любой вид статической ER-диаграммы проектировщику вообще не нужен. Проектировщик имеет гарантии, что

любые разновидности связей можно смоделировать на совокупности ключевых атрибутов.

7.2. Постановка задачи

Основная задача экспериментального проектирования каркасных схем БД разных ПрО – всестороння апробация основных теоретических выводов и закономерностей КМД. Цель проектирования – сравнение схем БД на предмет их подобия, а также проведение сравнительного численного анализа времени доступа к данным. Эксперимент проведен на 7 ПрО по принципу роста сложности. Большинство приведенных схем БД отобрано из реальных внедренных приложений.

7.3. ПрО «Приемная комиссия университета»

В качестве первого эксперимента выбрана наиболее простая и широко описанная в литературе ПрО: «Приемная комиссия Университета». Классификация сущностей-объектов этой ПрО, также схема каждой сущности-объекта и фрагмент итоговой каркасной диаграммы позволил исследовать скорости доступа к данным и сравнить схемы БД по классическому подходу к проектированию и каркасному. Ниже приведен подробный словесный анализ каждой сущности-объекта. Для следующих ПрО будут приводиться только результирующие схемы.

7.3.1. Описание ПрО и схема БД

Сущность-объект *ВУЗ* – атомарная для данной ПрО - справочник *ВУЗ*ов данного университета. Тут уточняется понятие «данная ПрО» потому, что приведен лишь фрагмент ПрО. Атомарность или слабость устанавливается именно в сравнении границ ПрО. В данном исследовании схема сущности-объекта *ВУЗ* упрощена. И ей назначен статус атомарной. Ее зависимостью от сущности-объекта *ГОСУДАРСТВО*, а потому от ключевого атрибута

КодГосударства, также можно пренебречь. Таким образом, *КодВУЗа* – это суррогатный унарный ключ сущности *ВУЗ*. Наименование конкретных *ВУЗов* хранится в атрибуте *Название*. Атрибут *СокрНазв* – это аббревиатура *ВУЗа*. Очевидно, что все иные атрибуты могут быть встроены аналогично.

Сущность-объект *ОТДЕЛЕНИЯ* – также атомарная сущность. Отражает информацию об отделениях. Как правило, существует два отделения – дневное и заочное. Иногда имеется еще и вечернее. Ключ – унарный суррогатный – *КодОтдел*.

Сущность-объект *СПЕЦИАЛЬНОСТЬ* – атомарная. Это – справочник специальностей в соответствии с государственным классификатором. Таблица, которая моделирует эту сущность-объект, хранит данные о названии специальности и иных ее свойствах. *КодСпец* – унарный суррогатный ключевой атрибут.

Сущность-объект *ДИСЦИПЛИНЫ* – слабая сущность-объект – общегосударственный справочник дисциплин, каждая из которых зависит от сущности-объекта *СПЕЦИАЛЬНОСТЬ*. *КодСпец+КодДисц* – суррогатный бинарный ключевой атрибут.

Сущность-объект *АБИТУРИЕНТ* – атомарная сущность-объект, которая не зависит от иных сущностей-объектов. Его условной зависимостью от атрибута *КодГосударства* можно также пренебречь. Поэтому в данной ПрО имеет унарный суррогатный ключевой атрибут - *КодАбитур*. А отношение, которое моделирует эту сущность-объект – это перечень всех абитуриентов Украины, которые подали заявления именно в этот университет. Т.е., в те или иные его ВУЗы. Содержит атрибуты *ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, ГОД РОЖДНИЯ, МЕСТО РОЖДЕНИЯ* и т.п.

Сущность-объект *ФАКУЛЬТЕТ* – слабая сущность-объект, которая обладает зависимостью существования. Однако может моделироваться и постсвязной составной (то есть связью) от независимых между собой атомарных сущностей-объектов *ВУЗы* и *ФАКУЛЬТЕТЫ В ГОСУДАРСТВЕ*. Тут это справочник факультетов *ВУЗов* университета. *КодВУЗа+КодФак* – суррогатный составной бинарный ключевой атрибут. *НАЗВАНИЕ* факультета – один из неключевых атрибутов сущности-объекта.

Сущность-объект *ПРЕПОДАВАТЕЛИ* – атомарная сущность-объект – перечень преподавателей. Зависимостью от сущностей-объектов *ОБЛАСТЬ, ГОРОД* или *СПЕЦИАЛЬНОСТЬ* в этой ПрО можно пренебречь. Поэтому ключ – унарный атрибут *КодПрепод*. Неключевые атрибуты – ФИО, степень, звание, дата рождения и т.п.

Составная сущность-объект *ПРЕПОДАВАТЕЛИ В ВУЗе* – это связь сущностей-объектов *ПРЕПОДАВАТЕЛИ* и *ВУЗы*. Поэтому ключ – бинарный составной атрибут *КодВУЗа+КодПрепод*. Атрибутами связи могут быть *ДАТА, ПРИМЕЧАНИЕ* и т. п.

Сущность-объект *СПЕЦИАЛЬНОСТИ В ВУЗах* – составная бинарная сущность-объект (связь). Ключ – бинарный составной атрибут - *КодВУЗа+КодСпец*. Моделирующее ее отношение отфильтровывает пакет специальностей из отношения *СПЕЦИАЛЬНОСТИ* именно для данного университета. Атрибут связи – текущее *НАЗВАНИЕ* специальности именно в этом *ВУЗе, ДАТА* ее открытия и т. п.

Сущность-объект *ДИСЦИПЛИНЫ В ВУЗах* – составная тернарная сущность-объект (связь) между сущностями-объектами *ДИСЦИПЛИНЫ, ВУЗы* и *СПЕЦИАЛЬНОСТИ*. Ключ – тернарный составной атрибут -

КодВУЗа+КодСпец+КодДисц. Атрибуты связи – текущее *НАЗВАНИЕ* дисциплины, *ДАТА* открытия, суммарные плановые показатели и т.п.

Сущность-объект *АБИТУРИЕНТЫ ВУЗов* - составная сущность-объект (связь), которая связывает сущность-объект *АБИТУРИЕНТЫ* и сущности-объекты *ВУЗы* и *СПЕЦИАЛЬНОСТЬ*. Поэтому имеет тернарный ключевой атрибут - *КодВУЗа+КодСпец+КодАбитур*. Моделирующее отношение – это перечень абитуриентов, которые подали заявления именно в эти *ВУЗы* данного университета и именно на эти *СПЕЦИАЛЬНОСТИ*. Обладает атрибутами связи – *ДАТА* заявления, *ПРИМЕЧАНИЕ* и т.п.

Сущность-объект *УЧЕБНАЯ ПРОГРАММА* – является составной сущностью-объектом. В этой ПрО она является результатом связи семи сущностей-объектов - *ВУЗ, СПЕЦИАЛЬНОСТЬ, ОТДЕЛЕНИЕ, ФАКУЛЬТЕТ, ДИСЦИПЛИНА, ПРЕДМЕТ* И *СЕМЕСТР*. Составной септарный ключевой атрибут: *КодВУЗа+КодСпец+КодОтдел+КодФак+КодДисц+КодПредм+КодСеместр*. Атрибуты связи – *ЧИСЛО ЧАСОВ, ПРИМЕЧАНИЕ* и т.п. Причем сущность *УЧЕБНАЯ ПРОГРАММА* не зависит от сущностей-объектов *ПРЕПОДАВАТЕЛИ, ГРУППЫ СТУДЕНТОВ, АУДИТОРИИ, НЕДЕЛЯ* и *ДЕНЬ*. Если в состав ключевых атрибутов ввести еще и эти атрибуты, получим составную сущность-объект (связь) *РАСПИСАНИЕ ЗАНЯТИЙ*.

Сущность-объект *ВСТУПИТЕЛЬНЫЕ ЭКЗАМЕНЫ* – септарная составная сущность-объект (связь). Ключ – составной септарный атрибут: *КодВУЗа+КодСпец+КодФак+КодОтдел+КодПредм+КодПреп+КодАбитур*. А атрибуты связи – *ОЦЕНКА, ДАТА* и т.п.

Можно сравнить сущность-объект *ВСТУПИТЕЛЬНЫЕ ЭКЗАМЕНЫ* и сущность-объект *ЭКЗАМЕНЫ СЕССИИ*. Большая часть составного ключевого атрибута совпадет, так как совпадает происхождение смысла этих

различных сущностей. Однако имеется и отличительные звенья. Вместо абитуриента, так сказать, субъектом связи выступает уже сущность-объект *СТУДЕНТ*, которая именно в этой ПрО имеет свой тернарный ключевой атрибут – *КодВУЗа+КодСпец+КодСтуд*. А атрибут *КодАбитур*, который был ключом в таблице *АБИТУРИЕНТ*, в этой таблице уже является неключевым справочным атрибутом. Именно эта часть совокупности отношений уже относится к приложению *УЧЕБНЫЙ ПРОЦЕСС*.

В этой ПрО появляется и слабая сущность-объект *ГРУППА*, которая объединяет совокупность студентов одной специальности и года зачисления в *ВУЗ*. Она, как правило, имеет тернарный ключ *КодВУЗа+КодСпец+КодГруппы*. А год зачисления, то есть год формирования группы, как правило, является неключевым атрибутом (на диаграмме не показан). Следовательно, в этой части приложения обязательно моделируется и сущность-объект *СТУДЕНТЫ В ГРУППАХ*, то есть составная сущность-объект (связь) между сущностью-объектом *СТУДЕНТЫ* и сущностью-объектом *ГРУППЫ*. И, в свою очередь, ее ключ - квартарный составной атрибут *КодВУЗа+КодСпец+КодГруппы+КодСтуд*.

Однако, как правило, атрибут *КодСтуд* не зависит от атрибута *КодГруппы*, так как группы – это условное разделение множества студентов, которое упрощает управление ими. Поэтому группы имеют в своем составе небольшое число людей – элементов множества. Именно небольшой диапазон изменения значений этого атрибута и влияет на то, что пользователь ПрО назначает домен значений атрибута *КодСтуд* так, что одновременно кодирует в его структуре и группу. Поэтому эта связь является лишь справочной. А значит в данной ПрО в сущности-объекта *ЭКЗАМЕНЫ СЕССИИ* атрибут *КодГруппы* участия не принимает.

В качестве ключевых для сущности-объекта *ЭКЗАМЕНЫ СЕССИИ* добавляются еще и атрибуты, которые обуславливают специфику сессии – *НОМЕР ПОПЫТКИ* (первая, вторая или даже четвертая), а также *ТИП СЕССИИ* – зимняя или летняя.

Таким образом, сущность-объект *ЭКЗАМЕНЫ СЕССИИ* – составная нонарная сущность-объект (связь). Ключ – составной нонарный атрибут: *КодВУЗа+КодСпец+КодФак+КодОтдел+КодПредм+КодПреп+КодСтуд+ НомПопыт+КодСессии*. А атрибуты связи – оценка, дата и т.п.

На рис. 7.1 приведена не традиционная ER-диаграмма Чена [359], а каркасная. Здесь дуги играют роль не связей между сущностями-объектами, а указателей на основные взаимообусловленности доменов и ключей. А значит и маршруты отслеживания целостности для СУБД. Причем, очевидно, что соответствие имен ключевых атрибутов и означает их взаимность. Впервые аналогичные диаграммы были предложены в [222, 223] в 1993 году.

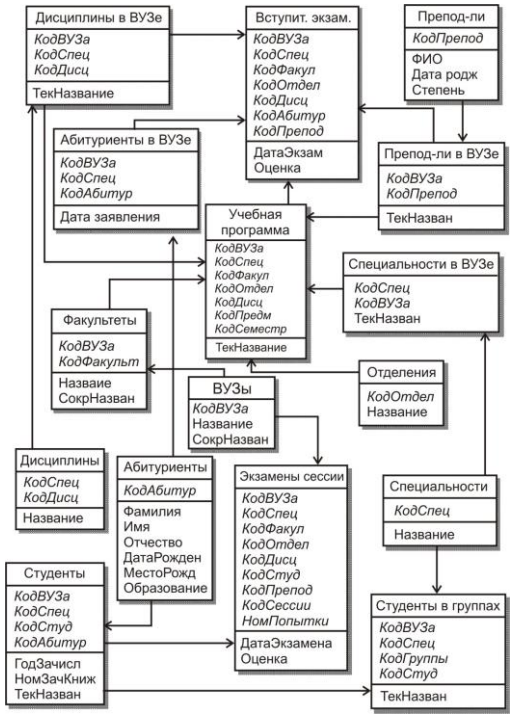


Рис. 7.1. Каркасная схема БД фрагмента ПрО «Университета»

7.3.2. Зависимости времени доступа к данным

На рис. 7.2 показан график, иллюстрирующий рост времени (t по оси ординат, в секундах) доступа к данным при получении одной записи из увеличивающихся групп кортежей (z по оси абсцисс, в миллионах записей) при запросе на соединение 4 отношений, находящихся в 3НФ (кривая 2).

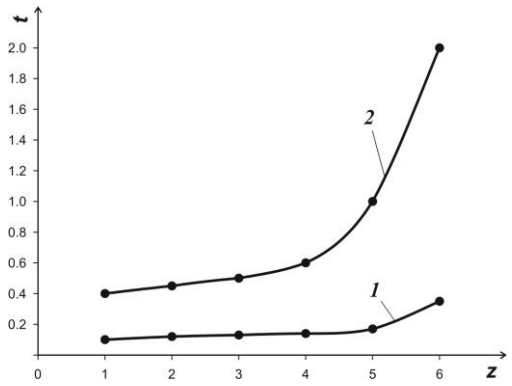


Рис. 7.2.

И значительную общую экономию времени на индексную выборку этой же записи из одного каркасного бинарного отношения (кривая 1), а также слабый прирост времени при увеличении числа записей. При этом результирующее отношение обладало в среднем от 100 до 200 кортежей.

7.4. ПрО «Склад завода «Стройиндустрии»

Схема каркасной БД показана на рис. 7.3.

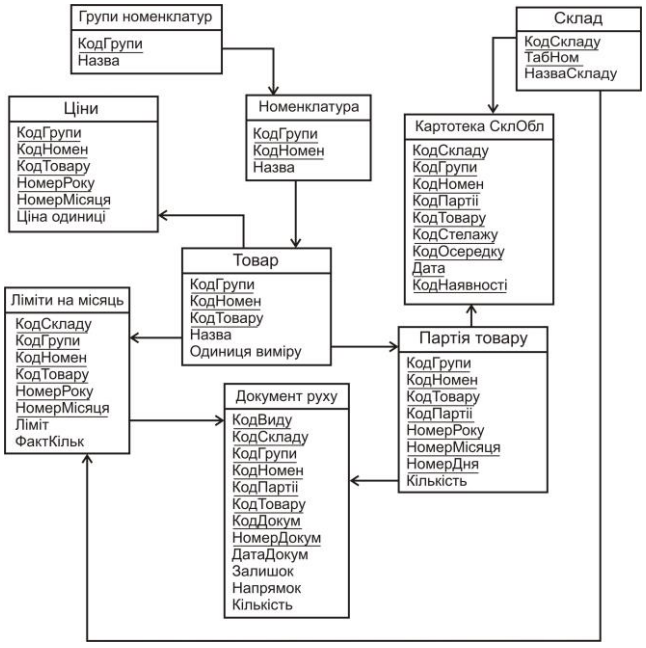


Рис. 7.3 Каркасная схема БД фрагмента ПрО

Схемы каждой сущности-объекта данной ПрО имеет следующий вид.

ГРУППЫ НОМЕНКЛАТУР (*КодГрупТовар*, Наимен) – атомарная сущность-объект,

НОМЕНКЛАТУРЫ (*КодГрупТовар*, *КодНоменкл*, Наимен) – слабая сущность-объект,

СОСТАВ (*КодСостава*, *НаимСостава*, *ТабНомМатОтвЛица*, Прим) – слабая сущность-объект,

ТОВАР (*КодГрупТовара*, *КодНоменкл*, *КодТовара*, Наимен) – слабая сущность-объект,

ЦЕНЫ ТОВАРА (*КодГрупТовара*, *КодНоменкл*, *НомПартии*, *КодТовара*, *НомГода*, *НомМесяц*, Цена) – слабая сущность-объект,

ПАРТИЯ ТОВАРА (КодГрупТовар, КодНоменкл, КодПартииТовар, КодТовара, Нименов) – слабая сущность-объект,

ЛИМИТ НА МЕСЯЦ (КодСостава, КодГрупТовар, КодНоменкл, КодТовара, НомГода,НомМес, Лимит, Факт) – слабая сущность-объект,

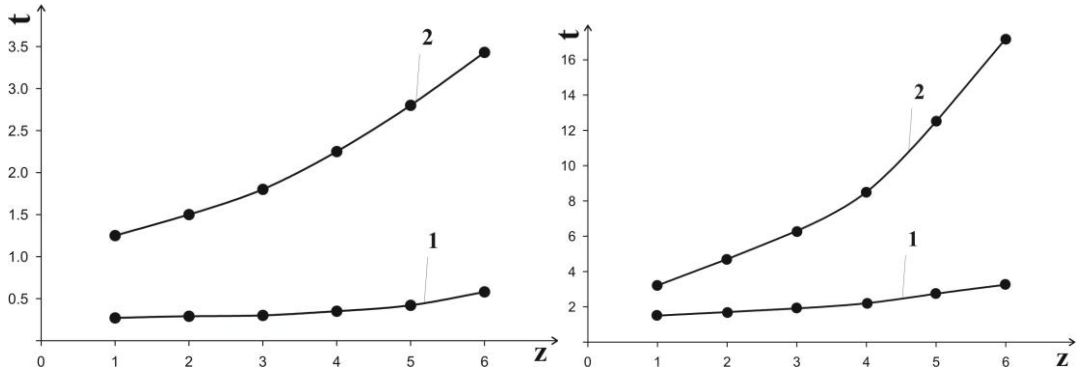
КАРТОТЕКА СКЛАДСКОГО УЧЕТА (*КодСостава*, *КодГрупТовар*, *КодПартииТовар*, *КодНоменкл*, *КодТовара*, *КодСтелажа*, *КодЯчейки*, *КодНаличияТовара*, Дата) – артефакт составной сущности-объекта

СКЛАДСКОЙ УЧЕТ (),

ДОКУМЕНТ ДВИЖЕНИЯ (*КодТипаДвижТов*, *КодСостава*, *КодГрупТовар*, *КодПартииТовар*, *КодНоменкл*, *КодТовара*, *КодТипаДокум*, *КодНапрДвиж*, НомДок, ДатаДок, ОстатокТов, КоличТов) – артефакт составной сущности-объекта ДВИЖЕНИЕ ТОВАРА().

На рис. 7.4, а) показан график, иллюстрирующий рост времени (*t* по оси ординат, в секундах) доступа к данным при получении одной записи из

увеличивающихся групп кортежей (z по оси абсцисс, в миллионах записей) при запросе на соединение 4 отношений, находящихся в 3НФ (кривая 2). И значительную общую экономию времени на индексную выборку этой же записи из одного каркасного квартарного отношения (кривая 1), а также слабый прирост времени при увеличении числа записей. При этом результирующее отношение обладало в среднем от 100 до 200 кортежей.



а) 4 отношения б) 8 отношений

Рис. 7.4

На рис. 7.4., б показан график, иллюстрирующий временя доступа к данным при получении одной записи из увеличивающихся групп кортежей при запросе на соединение 8 отношений, находящихся в 3НФ (кривая 2) и индексную выборку этой же записи из одного каркасного октарного отношения (кривая 1). Заметен слабый прирост времени при увеличении числа записей. Результирующее отношение также обладало в среднем от 100 до 200 кортежей.

7.5. ПрО «Горгаз – Участок обхода контролеров»

Для формирования унифицированного запроса к 3НФ БД, возвращающего группу данных для документа «Наряд на обход строений на участке», применяется комплексная операция выборка из соединения. А именно: «*выбрать строение для определенного сотрудника, которое находится на определенной улице, на определенном участке, в определенном административном районе города, в определенном населенном пункте, в определенном административном районе, в определенной*

административной области, в определенном территориальном районе, в определенной территориальной области». Схемы каждой сущности-объекта ПрО приведены в Приложении 2. Схема БД приведена на рис. 7.5.

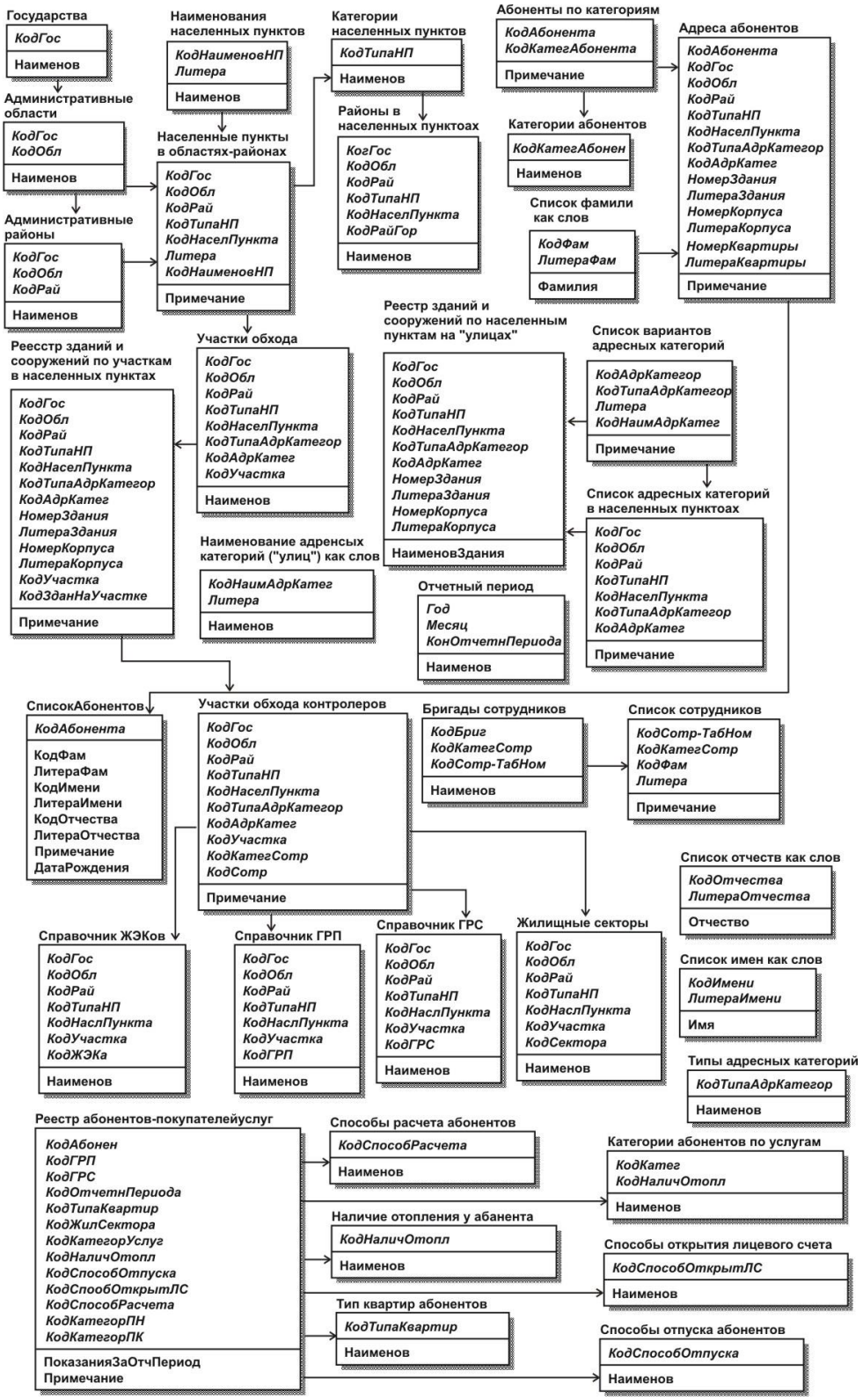


Рис. 7.5. Фрагмент каркасной схемы БД ОАО «Горгаз»

Для формирования унифицированного запроса к 3НФ БД, возвращающего группу данных для документа *НАРЯД НА ОБХОД СТРОЕНИЙ НА УЧАСТКЕ*, применяется комплексная операция *выборка из соединения*.

Таблица 7.1.

Время выполнения запросов к классической схеме БД в 3НФ (в с)

Число отношений	1 млн.	2 млн.	3 млн.	4 млн.	5 млн.	6 млн.
4	1.25	1.79	2.04	2.35	2.94	3.43
8	3.21	5.13	7.11	8.50	12.80	17.19
12	16.91	19.42	25.84	31.96	38.48	44.14
14	56.74	71.43	86.98	94.22	114.74	128.65

Таблица 7.2.

Время выполнения запросов к каркасной схеме БД (в с)

Число ключей в отношении	1 млн.	2 млн.	3 млн.	4 млн.	5 млн.	6 млн.
4	0.23	0.29	0.30	0.35	0.42	0.58
8	1.11	1.70	1.91	2.20	2.74	3.25
12	2.04	2.44	2.94	3.41	4.27	5.93
14	2.84	4.12	4.91	6.24	7.85	8.84

Но для каркасной схемы БД (рис. 7.5.) никаких дополнительных операций, кроме индексного поиска по сумме ключевых идентификаторов каждого из указанных признаков в многоарном отношении, не требуется. Результирующее отношение будет декарным (10-арным).

В эксплуатируемой БД исследуемой ПрО были заменены 29 3НФ-отношений на каркасные. При этом общее число отношений в БД возросло

лишь до 37. Но скорость доступа к данным по типовым запросам повысилась на несколько порядков.

На рис. 7.6 приведен график, иллюстрирующий рост времени доступа к данным при получении одной записи из увеличивающихся групп записей при запросе на соединение 10-и 3НФ-отношений (кривая 2). И слабый прирост времени на индексную выборку этой же записи из одного каркасного октарного отношения (кривая 1) при таком же увеличении числа записей.

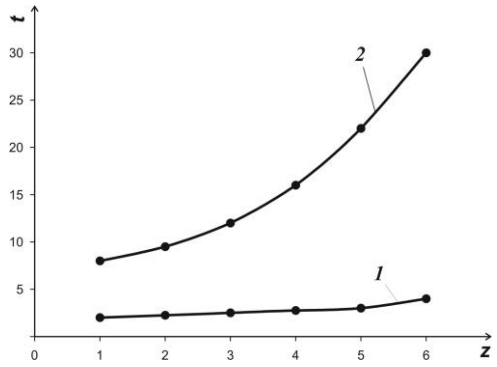


Рис. 7.6

7.6. ПрО «Шахта – Учет отгрузки угля»

Описываемое приложение БД было разработано на шахте «Комсомолец Донбасса» с помощью CASE-генератора SWS [225] сторонними пользователями этой инструментальной системы. В работе имеется соответствующий акт внедрения. Как показано в [221], само CASE-средство SWS проектировалось в строгом соответствии с каркасной моделью данных. При этом приложения БД, синтезируемые с помощью SWS и каркасной схемы БД, обладают максимальной эффективностью.

Для формирования унифицированного запроса к БД, возвращающего группу данных для анализа документов (артефактов) «Удостоверение партии» или «Счет к оплате потребителю», применяется единственная операция *выборка из соединения*. Причем, все соединения, присутствующие в схеме БД, сформированы не по факту выполнения запроса пользователя к БД, а по факту внесения текущих данных [221].

Запросы могут быть сформулированы на естественном языке. Например: «*выбрать группу категорий продукта за все месяцы конкретного года, добытого на конкретном участке конкретной шахты, показатели зольности, влажности, теплоты сгорания и процента серы которых не превышали бы такой-то величины».*

А также: *«выбрать все счета к оплате за текущий месяц текущего года, по которым для конкретной категории продукта, добытого на произвольных участках конкретной шахты разными бригадами и в разные смены, скидки-надбавки были бы не выше указанных».* На рис. 7.7. приведена каркасная схема БД указанной ПрО. С целью экономии места список всех сущностей-объектов с указанием их классификации и схемы приведен в Приложении 3. Там курсивом традиционно выделяются ключевые атрибуты.

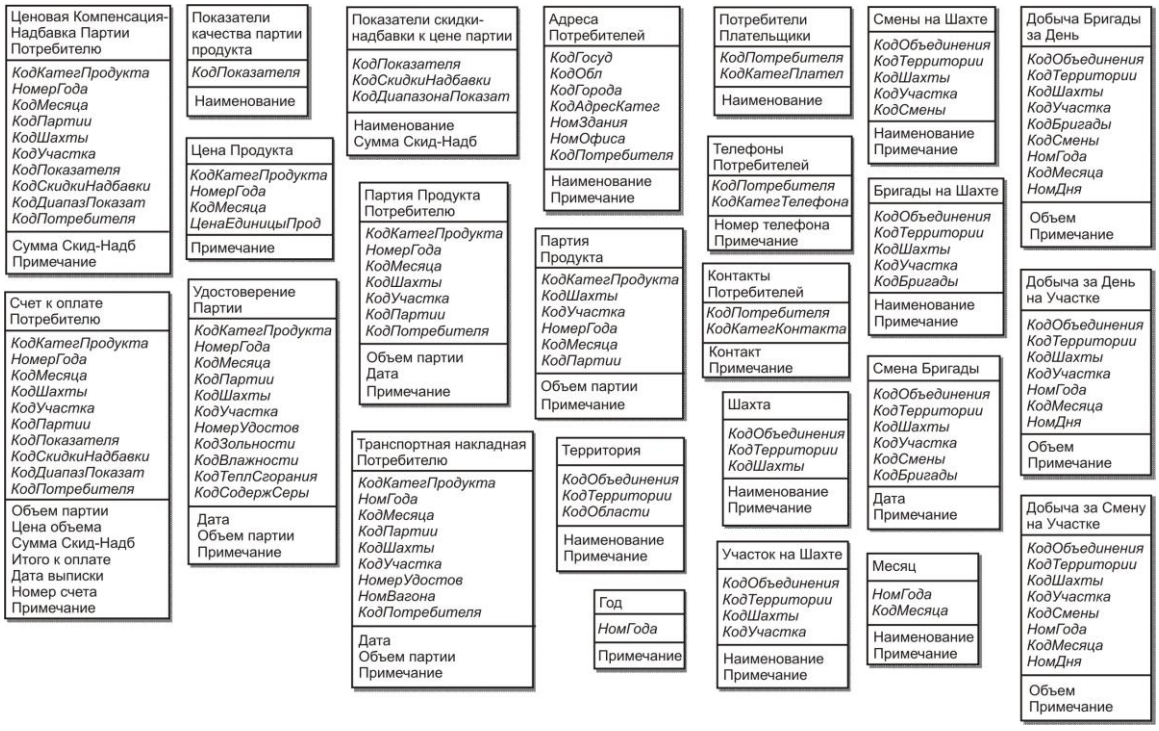


Рис. 7.7. Фрагмент каркасной схемы БД

На рис. 7.8. приведены графики, иллюстрирующие рост времени доступа к данным при получении одной записи из увеличивающихся пачек записей при запросе на соединение 12-и 3НФ-отношений (кривая 2).

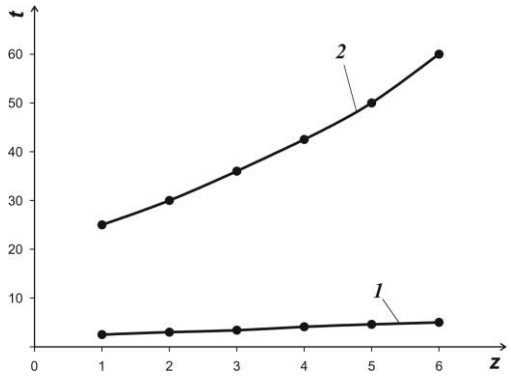


Рис. 7.8.

И заметно меньший прирост времени на индексную выборку этой же записи из одного 12-арного каркасного отношения (кривая 1) при таком же увеличении числа записей.

В приведенной схеме БД синтезировано 25 каркасных отношения. Скорость доступа к данным по типовым запросам повысилась на несколько порядков по сравнению со схемой, разработанной в соответствии с алгоритмом нормализации Кодда [361].

7.7. ПрО «Городская поликлиника»

Описываемое приложение БД было разработано в г. Хмельницком в одной из городских поликлиник с помощью CASE-оболочки SWS [225] сторонними пользователями этой инструментальной системы. В работе имеется соответствующий акт внедрения.

Схема каждой сущности-объекта данной ПрО приведена в Приложении. С целью экономии места во всех схемах приведенных там отношений атрибуты, отвечающие за актуальность записей, не приведены. Тут курсивом выделяются ключевые атрибуты. Для примера в некоторых отношениях приведены ключи-ссылки на метаданные (выделены иным шрифтом).

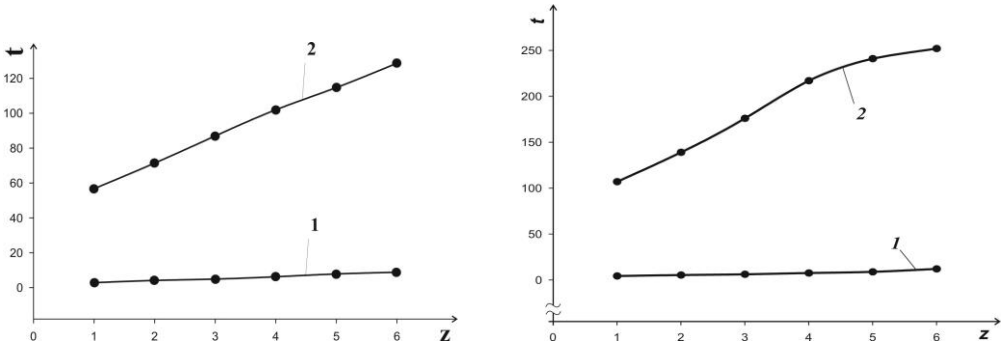
Ценность столь подробного разделения групп отношений на маски особенно проявляется при поддержке приложения режима реального времени [210 223], когда превалирующее большинство запросов пользователей проектируется заранее. Тогда на моменте ввода любого данного в отношения, моделирующие оперативные состояние ПрО, запускаются фоновые процедуры синтеза всех необходимых «архивных» масок, в которых в режиме реального времени обновляются соответствующие кортежи, связанные по индексам соответствующих ключей.

При такой конфигурации приложения потребность в написании значительного объема листингов запросов, так или иначе зависящих от семантики данных, значительно снижается.

Заметим, что вид схемы каркасной БД описанной ПрО полностью соответствует аналогичной схеме каркасной БД из работы [209]. Поэтому с целью экономии места тут диаграмма не приведена. Для формирования унифицированного запроса к БД, возвращающего группу данных для анализа документов (артефактов), таких как, например, «Отчет о работе регистратуры» или «Счет к оплате больному», применяется единственная операция *выборка из отношения*. Причем, все соединения, присутствующие в отношениях, моделирующих связи сущностей-объектов, сформированы не по факту выполнения запроса пользователя к БД, а по факту внесения текущих оперативных данных [210, 223].

На рис. 7.9. приведены графики, иллюстрирующие рост времени доступа к данным при получении одной записи из увеличивающихся групп записей при запросе на соединение 14-ти (7.11., а) и 16-ти (7.11., б) 3НФ-отношений (кривая 2). И заметно меньший прирост времени на индексную

выборку этой же записи из одного многоарного каркасного отношения (кривая 1) при таком же увеличении числа записей. Кривые совпади с результатами предыдущих разделов и работы [209]. Это подтверждает подобие схем БД разных ПрО еще и при исследовании скорости доступа к данным



а) 14 отношений б) 16 отношений

Рис. 7.9.

В приведенной схеме БД синтезировано 125 каркасных отношения. Скорость доступа к данным по типовым запросам повысилась на несколько порядков по сравнению со схемой, разработанной в соответствии с алгоритмом нормализации Кодда [361].

7.8. ПрО невычислительного характера – способ автоматизированной цифровой многопрограммной мультисигнальной коммутации

«Выявление порядков позволяет сравнивать в БД прежде несравнимые объекты. И, в конечном счете, выявлять закономерности, ранее, быть может, неизвестные» [164]. «В 1996 году специалистом по базам данных М. Леоновым для биологического факультета Московского университета была построена БД по теме «Зонтичные»… В результате не биологом был совершен ряд биологических открытий — около пятидесяти ранее неизвестных видов и два рода в семействе, в котором, укажем для сравнения, всего около трех тысяч видов. А каждое из подобных открытий может увековечить имя открывателя. В общем случае, наведение порядка в

сложной (пусть даже и изученной) предметной области, т.е. выявление упорядочивающих признаков, может стать научной проблемой». Эта поучительная история стала мотивацией исследований, представленных ниже.

Предложенный в [212] метод анализа ПрО был хорошо апробирован при разработке промышленных приложений БД [207, 210] и при решении задач автоматизации бизнес-процессов. Однако этот подход может успешно применяться и для исследования ПрО невычислительного характера. Для проверки универсальности предложенной методики проведем каркасный анализ существующей технологии (и серия соответствующих устройств) коммутации аналоговых или цифровых сигналов с пакетной, т.е. периодически-дискретной структурой [162] (в дальнейшем – просто сигналов). Такой тип сигналов используется в области связи, в телевизионных и видео сетях, в системах видеонаблюдения и компьютерных сетях. Как известно [87, 162], существующая технология коммутации отличается значительной трудоемкостью и высокой себестоимость.

Современные потребности пользователей медийных пространств ставят перед организаторами телевизионного производства [87] задачу значительного увеличения числа одновременно обрабатываемых источников сигналов. Сегодня прямые трансляции событий, обслуживаемые значительным числом видеокамер [289], являются признаком не только телеканалов, но и интернет-сервисов. Поэтому вопрос снижения себестоимости таких технологий является очень актуальным.

Каркасный анализ позволил обнаружить и предложить новые способы [231] автоматизированной цифровой многопрограммной мультисигнальной коммутации. Способы [231] обеспечивают групповое синхронизированное

переключение аналоговых или цифровых сигналов от значительного числа источников (50, 100, 1000 и т.д.). Рассматривается ситуация, когда возможность предварительной синхронизации сигналов отсутствует. Это обусловлено потребностью одновременной работы разнообразных источников сигналов от различных производителей.

7.8.1. Постановка задачи

Рассмотрим входящие сигналы, в каждом из которых моменты начала движения пакетов, имеющих постоянные характеристики, происходят по случайному принципу. Т.е., начало существования этих сигналов не синхронизировано. Тогда во время переключения с одного входящего сигнала на иной критической является целостность пакетов, так как при несинхронизированном переключении разрушаются первые пакеты сигнала, включаемого в выходящий.

Типичными примерами является сигналы, выходящие из видеокамер, VGA-сигнал компьютера или сигнал телевизионной трансляции, в которых пакетом является видеокадр. Разрушение видеокадров во время переключения таких сигналов является явлением недопустимым.

Известно [87, 162], что для устранения этого дефекта коммутации наибольшее распространение получили три технических решения – полная предварительная синхронизация всех коммутируемых сигналов на уровне источников, полная синхронизация на уровне коммутации, а также синхронизация лишь выбранных сигналов на уровне коммутации. Наиболее распространенным является первый подход. Однако, именно этот способ является самыми избыточным.

Второй подход заключается в одновременной цифровой буферизации пакетов всех входящих сигналов. На упомянутом принципе построены

устройства для коммутации телевизионных сигналов - широко известные коммутирующие видеопульты.

Третий подход заключается в цифровой буферизации пакетов только *выбранных* входящих сигналов. Наименее избыточным является именно он. Однако третий способ коммутации в настоящее время имеет ограниченное распространение. Причина в том, что в нем использована ручная регистрация выбранного входящего сигнала для его дальнейшей синхронизации.

Таким образом, целью этого исследования является выявление и устранение всех типов технологической избыточности в методике коммутации аналоговых или цифровых сигналов.

7.8.2. Каркасный анализ и схема ПрО

Как и в [210], под каркасным анализом ПрО будем понимать формализацию описания ПрО так, что каждой сущности-объекту ставиться в соответствие актуальная ячейка реляционного каркаса [210]. Тогда неформально каркасный анализ - это формирование каркасной схемы реляционной БД исследуемой ПрО таким образом, чтобы приложение БД, синтезируемое на данной схеме, моделировало бы функционирование исследуемой ПрО.

Такой подход может применяться и в том случае, когда разработка самого приложения БД не планируется. А каркасная схема БД используется как формальный прототип ПрО для выявления в ней тех или иных аномалий или противоречий. Исследовав связи между сущностями-объектами, можно сделать вывод о месте и причинах избыточности существующих технических решений.

Следуя идеям [207, 210], выпишем из описания ПрО [87] все сущности-объекты, используемые в принятых методах и устройствах коммутации

аналоговых или цифровых сигналов с дискретно-периодической структурой. Как указывалось в [210], особенностью каркасного анализа является непротиворечивость модификаций искомой схемы БД: если в результирующей схеме будут временно упущены некоторые сущности-объекты, ее дальнейшие модификации не приведут к противоречиям.

Имеем следующие отношения (многоместные предикаты [232]):

ВХОДНЫЕ СИГНАЛЫ (*КодСигнала*, …) – атомарная сущность-объект,

СИНХРОНИЗАТОРЫ (*КодСинхрон,…*) – атомарная сущность-объект,

УСТРОЙСТВА ЦАП-АЦП (*КодЦП,* …) – атомарная сущность-объект,

УСТРОЙСТВА КОДИРОВАНИЯ-ДЕКОДИРОВАНИЯ (*КодКодераДек*, …) – атомарная сущность-объект,

СПЛИТТЕРЫ СИГНАЛОВ (*КодСплит,* …) – атомарная сущность-объект

ПОЛЬЗОВАТЕЛИ-РЕЖИССЕРЫ (*КодПользов*, …) – атомарная сущность-объект,

УПРАВЛЯЮЩИЕ СИГНАЛЫ (*КодПользов*, *КодУпрСигн,* …) – слабая сущность-объект,

ВЫХОДНЫЕ ПРОГРАММЫ (*КодСигнала*, *КодСинхрон, КодЦП, КодКодера, КодУпрСигн, КодПользов*, ...) – составная сущность-объект.

На рис. 7.10. показана каркасная схема БД исследуемой ПрО.

Рис. 7.10.

В [210] доказано условие, при котором каркасное отношение может быть отнесено к безаномальной форме [378] – отсутствие в нем ограничений, которые не следуют из особых ограничений [210]. Однако технологическое ограничение исследуемой ПрО, заключающееся в обязательной предварительной синхронизации каждого источника коммутируемого сигнала, вносит дополнительную функциональную зависимость (ФЗ) в ключевой атрибут отношения *ВЫХОДНЫЕ ПРОГРАММЫ*.

Эта ФЗ $КодСигнала \leftrightarrow КодУправлСиг н$ не следует из ограничений на домены и ключи отношения. Более того, такая зависимость будет еще и взаимной [306] (ВФЗ). Эта транзитивность приводит к снижению нормальности формы отношения, что означает, что данная ФЗ является избыточной. В теории БД такое отношение должно быть декомпозировано на 2 отношения, что исключит из него один из зависимых атрибутов.

Неформально это означает, что в классической технологии коммутации не учтено, что входящие и управляющие сигналы – это взаимно-независимые сущности-объекты со степенью связи между ними «многие ко многим». И то, что разработчики технологии внесли дополнительное ограничение, привело к значительной избыточности устройств.

Однако, как указывалось в [210], такая избыточность может быть исключена внесением изменений не в модель, а в саму ПрО. При этом результат каркасного анализа воспринимается как рекомендации к реинжинирингу ПрО.

Аналогичный недостаток присущий и второму типу синхронизации – буферизации всех предварительно несинхронизированных сигналов на уровне коммутации. В этом подходе существует искусственная ВФЗ между входящим сигналом и буфером памяти в коммутирующем устройстве: $КодСинхрон \leftrightarrow КодСигнала$.

Этот недостаток приводит к существенному увеличению стоимости устройств коммутации, что особенно критично, если пользователь вынужден обслуживать значительное число входящих сигналов, в том числе сигналов от смешанных типов источников, формировать многопользовательский режим, а также транспортировать на значительные расстояния некоторую часть высоко-потовых 3G-SDI [162] сигналов.

Таким образом, наименее избыточной является только синхронизация выбранных сигналов на уровне коммутации.

Прототипом предложенного в [231] нового способа коммутации, основанного на автоматизированной выборочной синхронизации, является [455]. Здесь также получен новый технический результат, который заключается в автоматизированной синхронизированной коммутации неограниченного числа предварительно несинхронизированных источников. Но это – совершенно иной технический результат, так как синхронизация осуществляется благодаря потере одного-двух пакетов сигнала. Сдвиг промежутка времени, на который отличаются входящие сигналы один относительно другого, в прототипе компенсируется благодаря замене нескольких несинхронизированных пакетов инородными - например, черными полями. Но это приводит к потере целостности и входящего, и результирующего сигналов.

В случае, например, видеосигнала такой способ является неприемлемым для использования в профессиональных ПТС для монтажа программ в режиме прямой трансляции. Описанное техническое решение [455] условно применимо лишь в бытовой сфере, например, в пультах переключения телеканалами приемников телевизионных программ.

7.8.3. Новое техническое решение

Отличие нового метода заключаются в том, что коммутация сигналов осуществляется благодаря выборочной *автоматизированной* оцифровке

входящих аналоговых сигналов и выборочной автоматизированной буферизации цифровых сигналов. Именно процесс автоматизированной буферизации выбранного входящего сигнала и предоставляет возможность управлять моментом начала считывания пакета из буфера, синхронизируя его со считыванием пакета из другого входящего сигнала. Причем, в каждый момент на одном тракте, т.е. во время коммутации одной группы отрезков сигналов – одной программы, происходит буферизация лишь двух входящих сигналов - только что заказанного и предварительно заказанного. Здесь под программой, по аналогии с телевизионным вещанием, понимается вся последовательность коммутированных временных отрезков входящих сигналов, которые избирались пользователем на протяжении конкретного времени.

Этот процесс существенно отличается от синхронизации, описанной в прототипе [455]. Другое отличие от известных решений – достаточность единственной активации выбора того или иного входящего сигнала благодаря его номеру, который отражается на клавиатуре. Такой процесс отличается от способа ручного перенаправления заказанного сигнала на свободный буфер памяти, который используется во многих упомянутых микшерских пультах.

Такой подход поддерживает неограниченную последовательность синхронизированных переключений сигналов, а общее число входящих сигналов теоретически не ограничивается: от минимально двух до произвольного количества - 100, 1000, 100 000 и так далее, в зависимости от технологии реализации указанного метода.

7.8.4. Многопользовательский режим

На базе описанного построен и многопользовательский способ коммутации. Он отличается тем, что благодаря использованию принципа автоматизированной синхронизации каждый входящий аналоговый или

цифровой сигнал сплиттируется (размножается) не предварительно, а лишь *согласно запросу* пользователя. Причем, в каждый момент общее число одновременно размноженных входящих сигналов равняется лишь числу заказов пользователей этого промежутка времени. Такое решение может быть реализовано исключительно благодаря наличию в схеме системы автоматизированного управления процессом коммутации.

При многопользовательской автоматизированной схеме коммутации посредством выборочной буферизации сигналов в пределах одного тракта процедура синхронизации осуществляется благодаря освободившемуся от сигнала во время предыдущего переключения буферу памяти, который, ради оптимизации загрузки, освобождаясь от входящего сигнала, становится общим для всех трактов. Причем автоматизированной системой управления коммутацией *отслеживается* последовательность запросов пользователей. Отслеживается также и *очередь запросов*, что исключает заклинивание, если случайно возникает ситуация параллельного и почти одновременного заказа, когда промежуток времени между заказами при параллельном запросе пользователей становится *меньшим*, чем промежуток времени синхронизации.

На рис. 7.11. приведена схема, обобщенно иллюстрирующая описанный подход.

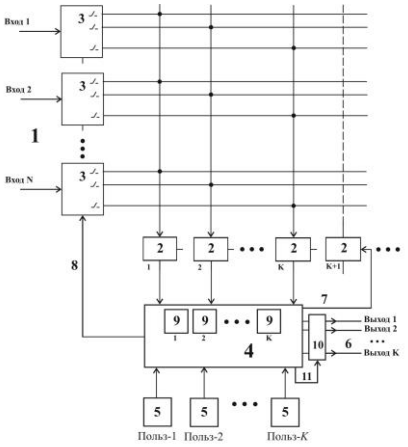


Рис. 7.11.

Здесь показана последовательность процессов и расположение устройств, которые используются в алгоритме метода.

1 - *N* входящих сигналов,

2 - буферы памяти от 1-го до *K+1*, каждый из которых имеет вход и выход для сигнала, который синхронизируется, а также вход для управляющего сигнала, причем на рисунке показана ситуация, когда можно ожидать следующий заказ одного из пользователей, поэтому текущий *(K+1)*-й буфер памяти пуст, а свободный от входящего сигнала канал для синхронизации здесь условно показан пунктирной линией; возможность дальнейшей установки новых буферов памяти показана многоточием,

3 - процесс входящего предварительного или лишь заказанного размножения *N* входящих сигналов,

4 - объединено система управления коммутацией и процесс цифровой коммутации входящих сигналов,

5 - клавиатуры пользователей,

6 - выходящие программы, количество которых равняется *K*,

7 - управляющий сигнал, который обеспечивает автоматизированную подачу выбранного входящего сигнала на вход свободного буфера памяти для синхронизации, здесь для снижения загрузки рисунку стрелка условно показана только к свободному от входящего сигнала буферу памяти,

8 - управляющий сигнал, который обеспечивает автоматизированное управление процессом размножения входящих сигналов,

9 - специализированные носители памяти для фиксации последовательности пар «номер входящего сигнала - продолжительность выбранного входящего сигнала в программе», количество которых равняется

количеству пользователей *K* (на рисунке показано в составе системы управления коммутацией),

10 - процесс агрегации выходящих сигналов,

11 - управляющий сигнал, который обеспечивает автоматизированное управление процессом агрегации выходящих сигналов.

7.8.5. Результаты тестовых испытаний

Описанный подход позволил собрать и неоднократно протестировать на значительном числе съемочных площадок, подобных [304, 305], специализированное устройство многокамерной многопрограммной коммутации видеосигналов.

Апробация проводилась на 12-16 профессиональных видеокамерах. Тестирование показало высокую надежность устройства. Эксплуатация подтвердила серию новых конкурентных преимуществ.

В течение многих лет объектом пристального внимания диссертанта и его соавторов является процесс обучения телеоператорскому искусству [133, 228, 304, 305]. При этом основные особенности многокамерной съемки [289] существенно влияют на скорость обучения. И позволяют использовать передвижную телевизионную студию как эффективный ПТС-тренажер, который также апробирован и всесторонне тестирован. В процессе многолетней практики [228, 304] по формированию профессиональных навыков у более чем 50 телеоператоров-стажеров и режиссеров-стажеров, система показала высокую надежность и эргономичность, что для процесса обучения очень важно.

7.8.6. Выводы к подразделу

Математические подходы к семантическому анализу тех ПрО, которые традиционно принято относить к «нематематическим» (техническим или технологическим), позволяет выявить в них противоречия и аномалии. И

подготовить новые решения, основанные на математических моделях и вычислительных алгоритмах.

Результаты каркасного анализа способов коммутации дискретно-периодических сигналов позволили предложить решение, поддерживающее не только автоматизированный, но многопользовательский режим коммутации. Такое обобщение позволяет нескольким пользователям одновременно работать в реальном времени на единой системе источников. В случае видеосигналов построение такой многопользовательской ПТС эффективно решает техническую задачу многопрограммной агрегации значительного числа входящих видеосигналов.

Предложено также эффективное решение классической задачи телевещательного объединения множества признаков: сигналов телеканалов, инородных отрезков телепрограмм, времен удаленных стартов каждого из отрезков, территорий вещания, заказчиков врезок-коммутаций, владельцев прав на телеканалы и т.п. При этом если пользователь в процессе эксплуатации обнаружит новую совокупность множеств признаков, не учтенную до эксплуатации, способ подразумевает естественную интеграцию этих новых признаков.

Описанная концепция была апробирована при организации значительного числа прямых телевизионных трансляций, телемостов и моментального многокамерного видеообслуживания событий, видеоотчеты о которых опубликованы на том же ресурсе и под теми же рубриками, что и [304, 305]. Полученные результаты позволяют предлагать методику к широкому внедрению.

7.9. Выводы к седьмому разделу

1. Как отмечалось в [216], схема каркасной БД отличается от диаграммы Чена [359] отсутствием дуг – их применение, как правило, нецелесообразно.

2. Если же дуги применяются, то в каркасной диаграмме они играют роль не связей между сущностями-объектами, а указателей на основные взаимообусловленности доменов и ключей. А значит и маршруты отслеживания целостности для СУБД. Причем, очевидно, что соответствие имен ключевых атрибутов и означает их взаимность. Впервые аналогичные диаграммы были предложены в [222, 223].

3. В каркасной диаграмме нецелесообразно также и отмечать обязательность, вхождение и степень каждой связи сущности-объекта, как предложено Р. Баркером [346, 347].

Тут под обязательностью [122] понимается характеристика связи, показывающая гарантию хотя бы одного вхождения в связь экземпляра каждой сущности. А под вхождением в связь [122, 377], например, ключевого атрибута, понимается число разных значений этого атрибута при постоянном значении конкатенации всех оставшихся. Вхождение принимает одно из значений множества {0,1,…, n}. Этот показатель в разных концептуальных моделях и разных работах называют разными терминами. Например, в UML-моделировании [303] этот показатель назван «кратностью». А в серии работ авторов [36] исследован более обобщенный показатель «степень участия» - полная (все вхождения в связь не равны 0) и частичная (имеется хотя бы одно вхождение, равное 0).

Под степенью связи сущностей-объектов понимается традиционный дискретный показатель типа «множественности» связи (отображения) экземпляра одной сущности-объекта с экземплярами других сущностей-объектов – от *1:1* до $G:H$. Также имеет разные наименования у разных авторов. В [36], например, это свойство названо «кардинальностью» связи.

В каркасной схеме БД указанные характеристики связей сущностей-объектов и их ключевых атрибутов не влияют на иные свойства отношений. Этот важный результат совпадает с аналогичным выводом работы [294].

4. Из приведенных в разделе рисунков фрагмента схемы БД видно, что чем сложнее ПрО, тем большее число многоарных связей должно присутствовать в схеме БД. Тем более интенсионалы каркасных отношений напоминают высказывания – обычные словесные предложения. Это означает, что реляционные каркасные отношения обладают одновременно и безаномальностью [206], и семантичностью.

5. Этот вывод не совпадает с тезисом о несемантичности «нерасширенной» РМД [362]. Фрагмент схемы БД показывает, что подавляющее большинство отношений моделирует связи. Впервые атомарную или слабую сущность-объект представлено как тривиальную связь в [210]. Таким образом, вся совокупность каркасных отношений (вся БД) есть не что иное, как совокупность связей в ПрО. Поэтому, по аналогии с термином ER-модель, концептуальная КМД может быть названа R-моделью. Различия в понятиях сущности-объекты и связи между ними становятся условными.

ВЫВОДЫ

Благодаря РК получена возможность для произвольной ПрО разработать алгоритм автоматизированной декомпозиции и синтеза схемы реляционной БД в форме, относящейся к максимально высокой степени нормальности. При этом каркасная схема БД получает дополнительные конкурентные преимущества, важные для проектировщиков-практиков. К ним относятся (по убыванию значимости).

- Безаномальность в смысле ДКНФ,

- Высокая скорость доступа к данным благодаря значительному снижению операций соединения в запросах,

- Динамическая модифицируемость схемы БД, что позволяет корректно вносить изменения в эксплуатируемое приложение, а также гибко модифицировать схемы не только пользовательских, но и управляющих БД (метабаз).

- Подобие схем БД для разных ПрО несмотря на значительные отличия семантик, что обеспечивает интероперабельность и кроссплатформность приложений.

Для исследователей также важны следующие выводы:

- ДКНФ-схема БД – это еще и схема семантики ПрО - формируя ДКНФ-схему БД, пользователь исследует семантику ПрО, и тем самым имеет возможность оптимизировать саму ПрО.

- Основная классическая концепция «все связи должны быть сначала декомпозированы к бинарным, а затем на уровне подготовки ответа на запрос сервер все соединит» устарела. Более актуальна концепция «все актуальные связи поддерживаются в режиме он-лайн».

- Каждое отношение в ДКНФ-схеме – это одно предложение, соответствующее одной «теме» по Кренке, то есть имеют одно подлежащее и одно сказуемое, без внутренней структуры – то есть без сложных подчинений и витиеватостей.

- С точки зрения взаимосвязи ограничений ПрО и ограничений отношения некорректно говорить о доменно-ключевой нормальной форме отношения. Но в смысле взаимозависимости доменов от ключей и ключей от доменов, а также возможности моделирования любого изменения состояния ПрО изменениями только доменов и ключей схемы БД, можно говорить лишь о схеме БД в целом. Одно отношение не может моделировать произвольное состояние ПрО. А совокупность отношений, построенная на каркасе, моделирует замкнутую систему. Эта совокупность и моделирует произвольное состояние ПрО. Таким образом, корректнее говорить о ДКНФ-схеме.

- Любой текст – от одного атомарного предложения и до произвольного их количества, представляющий пользователю информацию, является фрагментом каркаса. Это означает, что каркасный подход позволяет синтезировать разборщик смыслов.

Полученные результаты также опровергают гипотезу о том, что эффективность работы БД можно получить лишь путем денормализации. Если в некоторых локальных ситуациях эффективность денормализованных схем выше, этот результат не может претендовать на универсальный подход, потому что он также плохо поддается алгоритмизации и типизации, как и сама нормализация. Эффективность же ДКНФ является следствием строгой обоснованности и прогнозируемости.

Все перечисленное дает возможность решить проблему унификации, типизации и минимизации СУБД для произвольной ПрО. То есть, по сути, разработать инструментальную оболочку, которая масштабируется метаданными. И на этом основании создать малогабаритный, но мощный CASE-генератор каркасных метаданных, управляющих универсальной каркасной оболочкой. А также снабдить этот гибкий CASE-инструмент библиотекой типовых функций навигации каркасной реляционной БД, используемых в унифицированных запросах.

Такой подход практически полностью исключит потребность в ресурсоемких операциях соединения в большинстве запросов к БД. И существенно упростит настройку CASE-оболочки на произвольную ПрО в прикладной разработке.

Предложенный метод синтеза схемы БД может быть применен для любой ПрО. Особенно важен для той ПрО, где ограничения подвергаются частым изменениям. Такая схема БД позволяет минимизировать затраты на разработку и сопровождение приложений.

В результате проведенного диссертационного исследования:

- в РМД решена проблема, считавшаяся неразрешимой – получен и строго обоснован алгоритм синтеза безаномальной доменно-ключевой нормальной формы схемы БД;

- решена «проблема Кодда» о противоречии между нормализацией и оптимальным числом отношений в схеме БД – построена схема БД, в которой число отношений – фиксировано;

- предложено семантическое дополнение к РМД – КМД;

- обнаружена возможность существенно упростить алгоритмы пользовательских запросов приложений путем частичного или полного отказа от операций соединений в РМД;

- заложен фундамент нового подхода к семантическому анализу произвольных ПрО с возможностью обобщения его на моделирование поведения сложных саморазвивающихся систем;

- предложен новый взгляд на темпоральность в БД, на OLAP-потребности пользователя, на проблему объединения объектно-ориентированного и реляционного проектирования и схем БД, и приложений;

- на базе КМД разработан, спроектирован и апробирован на значительном числе промышленных внедрений уникальный CASE-генератор метаданных, управляющих приложениями – инструментальная система SWS 1.0;

- для некоторых специализированных ПрО предложены уникальные решения: предложен способ мультиканальной многопользовательской коммутации несинхронизированных периодически-дискретных сигналов (видео), на базе которого разработана, собрана и экспериментально апробирована малогабаритная малобюджетная ПТС (перемещаемая телевизионная студия); разработан подход к проектированию CASE-оболочки кластерного решения систем интегральных уравнений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Агапов В.П. Метод конечных элементов в статике, динамике и устойчивости пространственных тонкостенных подкрепленных конструкций / М: Изд. АСВ, 2000, – 152 с.

2. Алтайбек А.А. Проектирование баз данных на основе доменно-ключевой нормальной формы // Вестник КазНУ. – № 1. – Алматы, 2009. – С.111-118.

3. Айвазян С.А., Енюков И.С., Мешалкин Л.Д. Прикладная статистика. Исследование зависимостей / М.: Финансы и статистика, - 1985. – 487 с.

4. Андерсон Д.А. Дискретная математика и комбинаторика / М.: «Вильямс», 2003. – 960 с.

5. Андон Ф.И., Резниченко В.А., Язык запросов SQL / СПб: «Питер», -2006. - 417 с.

6. Артемьева И.Л., Гаврилова Т.Л., Клещев А.С., Логические модели второго порядка для предметных областей // Научно-техническая информация, сер. 2.- 1997.- № 6. С.14-30.

7. Аткинсон М. и др. Манифест систем объектно-ориентированных баз данных // СУБД, № 4, 1995, с.142-155

8. Атре Ш. Структурный подход к организации баз данных / М: Финансы и статистика, 1983. - 320 с.

9. Ахмадеев И.А. Проектирование баз данных. Учебное пособие / Набережные Челны.: «КамПИ», 1998. – 94 с.

10. Ахо А., Хопкрофт Д., Ульман Д. Структуры данных и алгоритмы / М.: Вильямс, 2006. - 400 с.

11. Ахо А., Сети Р., Ульман Д. Компиляторы. Принципы, технологии, инструменты / М.: Вильямс, 2001. – 1185 с.

12. Бабанов А.М. Теория семантически значимых отображений // Вестник ТГУ. 2003. № 280. С. 239-248.

13. Бабанов А.М. Применение теории семантически значимых отображений для проектирования реляционных баз данных // Вестник ТГУ. 2003. № 280. С. 249-257.

14. Бабенко Л.П., Лавріщева К.М. Основи програмної інженерії / К: Знання, 2001. – 269 с.

15. Бадмаева К.В. Алгоритм оценки релевантности представлений для материализации в специализированном хранилище данных // Вестник Сибирского государственного аэрокосмического университета, Выпуск 1 (22). В 2 частях. Ч.2. - 2009. — С. 60-64.

16. Бакулева М.А. Тензорная модель работы реляционной СУБД // Информационные технологии в проектировании. Межвуз. сб. научн. трудов. Рязань: РГРТА, 2004. - С.39-43

17. Бакулева М.А. Математическая модель построения многомерной БД // Информационные технологии и телекоммуникации в образовании и науке. Межвуз. сб. научн. трудов. Рязань: РГРТА, 2005. — С.23—26

18. Баранчиков П.А., Конфликт ключей отношения и методы его решения // Рязань.: Вестник РГРТУ, № 4, Вып. 26, - 2008. - С. 66-69

19. Барвайс Дж. Введение в логику первого порядка // Справочная книга по математической логике, Ч.1. Теория моделей, М.: Наука, 1982. 392 с, С. 12-59

20. Бардзокас Д.И., Кудрявцев Б.А., Сеник Н.А. Распространение волн в электромагнитоупругих средах / М.: «Едиториал УРСС», - 2003 г. – 336 с.

21. Баркер С.Ф. Создание приложений баз данных в среде Microsoft Visual Basic .Net и ADO.Net : советы, рекомендации, примеры / М.: Вильямс, 2003. - 560 с.

22. Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И. Методы и модели анализа данных: OLAP и Data Mining / БХВ-Петербург, - 2004, - 336 с.

23. Беккенбах Э. Прикладная комбинаторная математика, сборник статей / М.: Мир, 1968-360 с.

24. Белов В.С. Информационно-аналитические системы. Основы проектирования и применения / М. МЭСИ, 2005. — 111 с.

25. Белоногов Г.Г., Кузнецов Б.А. Языковые средства автоматизированных информационных систем / М.: Наука, 1983. – 289 с.

26. Бергер А.Б., Горбач И.В., Меломед Э.Л. и др Microsoft SQL Server 2005 Analysis Services. OLAP и многомерный анализ данных // СПб.: БХВ-Петербург, 2007. - 928 с.: ил.

27. Бир С. Кибернетика и управление производством / М.: Наука, 1965. - 392 с.

28. Бир С. Кибернетика и менеджмент / М.: URSS, 2011. – 280 с.

29. Бойко В.В., Савинков В.М. Проектирование баз данных информационных систем / М.: «Финансы и статистика», 1989. – 351 с.

30. Бородаев В.А., Кустов В.Я. Банки и базы данных / СПб.: «Вики», 1989. - 224 с.

31. Борщев В.Б. Естественный язык – наивная математика для описания наивной картины мира // Московский лингвистический альманах. 1996. № 1. С. 203-225.

32. Боярский К.К., Каневский Е.А., Лезин Г.В. Концептуальные модели в базах знаний // Информационные, вычислительные и управляющие системы. - СПб.: СПб ГИТМО (ТУ), 2002. - Вып. 6. - С. 57-62.

33. Брукс Ф.П. Как проектируются программные комплексы / М.: Наука, 1979. - 151 с.

34. Брукшир Дж. Г. Введение в компьютерные науки / М.: Вильямс, 2001. - 688 с.

35. Брюхов Д.О. Интероперабельные информационные системы: архитектуры и технологии. / Брюхов Д.О., Задорожный В.И., Калиниченко Л.А., Курошев М.Ю., Шумилов С.С. // СУБД, № 4, 1995. С.86-113

36. Буй Д.Б., Сильвеструк Л.М. Модель «сутність-зв’язок»: формалізація сутностей та зв’язків // Вісник Київського університету. Серія: фіз.-мат. науки. – 2006. – Вип. 3. – С. 143-152.

37. Булос Дж., Джеффри Р. Вычислимость и логика / М.: Мир, 1994. 396 с.

38. Буч Г., Объектно-ориентированный анализ и проектирование с примерами приложений на C++ / М.: БИНОМ, - 2001,- 560 с.

39. Буч Г., Джекобсон А., Максимчук Р.А. и др. Объектно-ориентированный анализ и проектирование с примерами приложений / М.: Вильямс, 2008, - 721 с.

40. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя // М.: ДМК, 2000, 432 с.

41. Бутенко И.В., Метасистема как инструмент для построения аналитических отчетов в информационных системах // Научно-технические ведомости СПбГПУ. - СПб.: СПбГПУ, 2011, № 2 (120) - с. 32-38.

42. Васильев В. Объектно-ориентированная БД: взгляд изнутри // Компьютеры плюс программы. – 1997. – № 36. – С. 45–49.

43. Варламов О.О. Эволюционные базы данных и знании для адаптивного синтеза интеллектуальных систем. Миварное информационное пространство / М.: Радио и связь, 2002. - 282 с.

44. Васкевич Д. Стратегии клиент-сервер / Киев: Диалектика;- 1996 г.- 384 с.

45. Вендров А.М. CASE-технологии: Современные методы и средства проектирования информационных систем / М.: Финансы и статистика, 1998 . – 176 с.

46. Вертгейм И.И., Терпугов В.Н. Параллельные технологии вычислений в механике сплошных сред и МДТТ / Пермь: ПГУ, 2007. – 84 с.

47. Выхованец В.С., Иосенкин В.Я. Понятийный анализ и контекстная технология программирования // Проблемы управления. 2004. №.4. С. 3-25.

48. Волков А.А., Шведенко В.Н., Модель формирования параллельных структур в объектно-ориентированных СУБД // Программные продукты и системы. № 3, 2011- С. 14-17

49. Волкова В.Н., Денисов А.А. Основы теории систем и системного анализа / СПб.: СГТУ, 2001 г. – 512 с

50. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем / СПб: «Питер», 2001, 382 с

51. Галахов И.В. Проектирование корпоративной информационно-аналитической системы // Открытые системы, №4, 2003. С.27-32.

52. Гайдамакин Н.А. Автоматизированные информационные системы, базы и банки данных: Вводный курс / М.: «Гелиос» 2002. – 368 с.

53. Ганти Венкатеш, Герке Йоханнес, Рамакришнан Раджу, Добыча данных в сверхбольших базах данных // Открытые системы. - 1999. - № 9-10. С. 45−53.

54. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных / М: «Вильямс», 2003, 1084 с.

55. Гаскаров Д.В. Интеллектуальные информационные системы / М.: Высш. шк., 2003, 432 с.

56. Гасанов Э.Э., Кудрявцев В.Б. Теория хранения и поиска информации / М.: «Физматлит», - 2002. - 288 с.

57. Гвардейцев М.И., Кузнецов П.Г., Розенберг В.Я. Математическое обеспечение управления. Меры развития общества / М. «Радио и связь», 1996, 176 с.

58. Гейн К., Сарсон Т. Структурный системный анализ: средства и методы / М.: Эйтекс, 1993. – Кн. 1. – 187 с.; Кн. 2. – 214 с.

59. Гери М., Джонсон Д. Вычислительные машины и трудно-решаемые задачи / М.; Мир, 1982. - 416 с.

60. Глибовец Н.Н., Глибовец А.Н., Шабинский, Применение онтологий и методов анализа текстов при создании интеллектуальных поисковых систем // Проблемы управления и информатики – К., 2011. - № 6. С. 95-102

61. Глушаков С.В., Ломотько Д.В. Базы данных. Учебный курс / Харьков: «Фолио», 2000, 257 с.

62. Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки. Программирование / К.: Наук. думка, 1989. – 376 с.

63. Голованов М. Иерархические структуры данных в реляционных БД // RSDN Magazine. - 2002. №1.- С.34-40

64. Голосов А.О., Аномалии в реляционных базах данных // СУБД. – № 3. – Москва, 1996, с. 23-38

65. Горев А., Ахаян Р., Макашаринов С. Эффективная работа с СУБД / СПб.: «Питер», 1997. -700 с.

66. Грабер М. Введение в SQL / М.: «Лори», 2000. – 382 с.

67. Гребенщиков Н.Н., Метод реляционного моделирования иерархий. // Информатика и проблемы телекоммуникаций. – Новосибирск: СибГУТИ, 2006. – С. 27-31.

68. Грей П. Логика, алгебра и базы данных / М.: Машиностроение, 1989. – 359 с.

69. Грекул В.И., Денищенко Г.П., Коровкина Н.Л., Проектирование информационных систем / Бином: Лаборатория знаний, 2008. - 304 с.

70. Гречко В.О., Темперанский В.А., Некоторые вопросы организации данных в системном анализе сложных объектов / Сб. "Автоматизация проектирования информационных систем", ИК АН УССР, 1976

71. Григорьев Ю.А., Ревунков Г.И. Банки данных. Учебное пособие. – М., МГТУ, 2002.- 319 с.

72. Григорьев Е., Представления сложных идентифицируемых объектов в реляционной базе данных // «Открытые системы», №1-2, 2000, С. 79-81

73. Грищенко А., Макаренко И. Системы на основе метаописаний. // Открытые системы. - 2001. № 10, С. 42-45

74. Гришенков Е.А. Построение концептуальных моделей баз данных при помощи шаблона модели единого корпоративного пространства данных // Системы управления и информационные технологии, 3(41), 2010. - С. 81-87

75. Грофф Дж. Р., Вайнберг П.Н., Энциклопедия SQL / СПб.: Питер, - 2004, - 896 с.

76. Гузь А.Н., Кубенко В.Д., Черевко М.А. Дифракция упругих волн / К.: «Наук. думка», 1978. – 307 с.

77. Гузь А.Н., Немиш Ю.Н. Метод возмущения формы границы в механике сплошных сред / К. «Наукова думка», – 1989. – 352 с.

78. Гуков Л.И., Ломако Е. И., Морозова А. Б. и др. Макетирование, проектирование и реализация диалоговых информационных систем / М.: Финансы и статистика, 1993. - 320 с.

79. Гуляев А.И. Временные ряды в динамических базах данных / М.: Радио и связь. 1989 г. – 128 с.

80. Гуляев Ю.В., Кравченко В.Ф., Рвачев В.Л., Сизова Н.Д. Исследование дифракции упругих волн на пластинах, ослабленных двумя отверстиями произвольной формы // ДАН. Математическая физика. – 1996. – 349, № 2. – С. 175 - 179.

81. Дал У., Дейкстра Э., Хоор К. Структурное программирование / М.: Мир, 1975. - 247 с.

82. Девитт Д., Грэй Д. Параллельные системы баз данных: будущее высоко эффективных систем баз данных. // «СУБД», «Открытые системы». 1995. - № 2. С. 8-31

83. Дейв Э., Иен С. Oracle. Проектирование баз данных / К.: BHV, 2000. - 528 с.

84. Дейниченко Р.А., Московская мебельная фабрика «Интерьер» и украинская SWS / Компьютер-Х, - 1998. - вып. 137, видеодокумент, Ютуб, http://www.youtube.com/watch?v=zVZfmLDt5GY

85. Дейт К.Дж. Введение в системы баз данных, - М.: Вильямс, 2005. – 1327 с.

86. Дейтел Г. Введение в операционные системы. В 2–х томах / М.: Мир, 1987. 359 с., 398 с.

87. Джакония В.Е. и др., Телевидение. Учебник для ВУЗов / М: «Горячая линия – Телеком» - 2007 - 616 с.

88. Джексон Г. Проектирование реляционных баз данных для использования с микроЭВМ – М.: «Мир», 1991, - 252 с.

89. Диго С.М., Базы данных. Проектирование и создание / М.: ЕАОИ, 2008, – 171 с.

90. Дорошенко А.Е., Захария Л.М., Цейтлин Г.Е. Алгебраическое проектирование программ: алгоритмы, объекты, инструменты // Проблемы программирования, 2007, № 2, с.

91. Дрибас В.П. Реляционные модели данных / М.: Мир, 1992. 192 с.

92. Дрождин В.В., Зинченко Р.Е., Масленников А.А., Отображение концептуальной модели предметной области в модель базы данных // Проблемы информатики в образовании, управлении, экономике и технике: сб. ст. VIII Всерос. науч.-техн. конф. - Пенза, 2008. - С. 208-210.

93. Елманова Н., Федоров А., Введение в OLAP / М.: Диалог-МИФИ, 2002, 268 с.

94. Замулин А.В. Системы программирования баз данных и знаний / Новосибирск: «Наука», Сиб. отд-ние, 1990, 352 с.

95. Замулин А.В. Алгебраическая семантика императивного языка программирования // Программирование. - 2003. - № 6. - С. 51-64.

96. Замулин А.В. Абстрактная модель компилятора как результат алгебраической семантики языка программирования. // Программирование. 2004. - № 5. - С. 69-80.

97. Зильбершатц А., Стоунбрейкер М., Ульман Дж. Базы данных: достижения и перспективы на пороге 21-го столетия // СУБД № 3, 1996, с. 103-117.

98. Зинченко Р.Е. Системно-изоморфное динамическое соответствие концептуальной модели предметной области и схемы базы данных // Программные продукты и системы. - 2010. - № 1, - С. 71-75.

99. Зыкин С.В. Построение отображения реляционной базы данных в списковую модель данных // Управляющие Системы и Машины. - 2001. - № 3. - С. 42-63.

100. Зыкин С.В. Метод формирования представлений данных для работы с информационными ресурсами // Омский научный вестник. - 2006. - № 3 (36). - С. 124 – 126

101. Иванов А.Н., Терехов А.Н., Романовский К.Ю., Кознов Д.В., Долгов П.С. Real: Методология и CASE-средство для разработки систем реального

времени и информационных систем // Программирование — 1999. — № 5. — С. 44-51.

102. Иванов Ю.Н. Теория информационных объектов и системы управления базами данных / М.: Наука, 1988. 232 с.

103. Ивахненко А.Г. Моделирование сложных систем: Информационный подход / К.: Вища школа, 1987, - 67 с.

104. Ильин А.А. Автоматизированная технология проектирования модели данных при построении информационно-аналитической системы // Вестн. Тамб. ун-та. Сер. Естеетв. и техн. науки. XIII Державинские чтения. - Тамбов, 2008. - Т. 13, вып. 1. -С. 89-90.

105. Капитонова Ю.В., Летичевский А.А. Математическая теория проектирования вычислительных систем / М.: Наука, - 1988. – 296 с.

106. Калиниченко Л.А. Методы и средства интеграции неоднородных баз данных / М.: Наука, - 1983. - 424 с.

107. Калиниченко Л.А., Брюхов Д.О., Задорожный В.И., Курошев М.Ю., Шумилов С.С. Интероперабельные информационные системы: архитектуры и технологии // СУБД. 1995. № 4. с. 35-43

108. Калянов Г.Н. CASE. Структурный системный анализ (автоматизация и применение) / М.: «Лори», 1996. - 360 с.

109. Калянов Г.Н. CASE-технологии: Консалтинг при автоматизации бизнес-процессов / М.: Горячая линия - Телеком, 2000 - 172 с.

110. Калянов Г.Н. Моделирование, анализ, реорганизация и автоматизация бизнес-процессов / М.: Финансы и статистика. 2006. - 240 с.

111. Карпова Т.С., Базы данных: модели, разработка, реализация / Питер, 2001, 304 с.

112. Карпунина М.Е., Бабкин Э.А. Об одном методе синтеза структуры распределенной базы данных, основанном на использовании искусственных нейронных сетей Хопфилда // Известия Академии инженерных наук им. А.М. Прохорова. Бизнес-информатика, Москва − Н.Новгород: ТАЛАМ, 2005. − Т.12. − С. 37-46

113. Карпуша В.Д. Моделирование и проектирование реляционных баз данных. Учебное пособие / В.Д. Карпуша, Б.Е. Панченко, Сумы: Изд. СумДУ. – 2010, - 385 с.

114. Кендалл С. Унифицированный процесс. Основные концепции / М.: Вильямс, 2002. - 160 с.

115. Ким В. Три основных недостатка современных хранилищ данных // Открытые Системы. — 2003. — № 2, С. 69-73.

116. Клещев А.С., Артемьева И.Л. Математические модели онтологий предметных областей. Часть 2. Компоненты модели. // Научно–техническая информация, сер. 2. – 2001. - № 3. - С.19-29.

117. Кнут Д. Семантика контекстно-свободных языков // В сб.: Семантика языков программирования. М.: Мир, 1980 – 395 с.

118. Кнут Д. Искусство программирования. Том 1. Основные алгоритмы / М.: «Вильямс», 2006. - 720 с.

119. Когаловский М.Р. Технология баз данных на персональных ЭВМ. – М.: «Финансы и статистика», 1992. – 224 с.

120. Когаловский Р.М. Расширение реляционной модели для баз данных временных рядов // Управляющие системы и машины. – 1994, -№ 6, - С. 24-30

121. Когаловский М. Р. Энциклопедия технологий баз данных. М.: Финансы и статистика, 2002. 800 с.

122. Козленко Л.А. Проектирование информационных систем // КомпьютерПресс, № 9, 2001, С. 10-11

123. Кокорева Л.В., Перевозчикова О.Л., Ющенко Е.Л. Диалоговые системы и представление знаний / К.: Наук. думка, 1993. – 448 с.

124. Комар Ф.В. Разработка метода описания семантики атрибутов реляционных баз данных // Успехи современного естествознания. - 2008. № 3. с. 56-59

125. Кондратьев В.В., Мисевич П.В. Объединенная система автоматизированного проектирования систем управления // Системы

управления и обработки информации: межвуз.сб. науч. тр../ Нижегород. политехн. ин-т. - Н.Новгород, 1991. - С.41-46.

126. Коннолли Т., Бегг К., Страчан А., Базы данных: проектирование, реализация и сопровождение / М.: «Вильямс», 2003, 1440 с.

127. Копейкин А.М., Методы построения концептуальных моделей баз данных // Труды учебных заведений связи. - СПБ.: СПБГУТ, 2007. - № 176, с. 166-178.

128. Корнеев В.В. Параллельные вычислительные системы / М.: Нолидж. 1999, 320 с.

129. Корнеев В.В., Гариев А.Ф., Васютин С.В., Райх В.В. Базы данных. Интеллектуальная обработка информации / М.: «Нолидж», 2000.- 352 с.

130. Костюк А.И., Базы данных и знаний: Курс лекций / Таганрог: Изд-во ТРТУ, 1999 г. - 175 с.

131. Кренке Д.М. Теория и практика построения баз данных / СПб.: Питер, 2003 г. – 800 с.

132. Крейг С.М. Администрирование баз данных. Полное справочное руководство по методам и процедурам / М.: Кудиц-образ. - 2003. – 752 с.

133. Кривомаз Л.С. Многокамерная прямая трансляция как эффективный тренажер для формирования профессиональных навыков телеоператоров и телережиссеров / Инновационное развитие общества в условиях кросс-культурных взаимодействий. Тезисы докладов 3-й междунар. конф., 26-29 апреля 2010, - Сумы. – 2010, Том 2, С.169 – 171

134. Кузнецов О.П., Адельсон-Вельский Г. М. Дискретная математика для инженера / М.: Энергоатомиздат, 1988.

135. Кузнецов С.Д. Базы данных: модели и языки. Учебник / М.: 2008. – 720с.

136. Кузьменко В.Г. VBA 2000 / М.: «Бином», 2008, 624 с.

137. Кукс С.В. Аксиоматизация эволюции схемы XML-баз данных // Программирование - 2003. - № 3 - с. 1-9

138. Кульба В.В., Ковалевский С.С. и др., Теоретические основы проектирования оптимальных структур распределенных баз данных // В кн. Информатизация России на пороге XXI века / М.: СИНТЕГ, 1999, 660 с.

139. Кунгурцев А.Б., Зиноватная С.Л., Анализ целесообразности реструктуризации базы данных методом введения нисходящей денормализации // Тр. Одес. политехн. ун-та. — Одесса: ОНПУ, 2006. — 1 (25). — С. 104 — 108.

140. Кунгурцев А.Б., Зиноватная С.Л. Модель реструктуризации реляционной базы данных путем денормализации схемы отношений // Тр. Одес. политехн. ун-та. – Одесса: ОНПУ, 2006. – № 2(26). – С. 105 – 111.

141. Кунгурцев А.Б., Зиноватная С.Л. Иерархическая модель объектов для исследования запросов к базе данных // Труды Одесского политехнического университета. – 2008, - вып. 2 (30), - С. 130 – 134

142. Курганов В.Ю., Блынский Л.Г. Организация хранилища данных для передачи информации между уровнями АСУТП и АСУСП // Автоматизация в промышленности. 2003. № 9. С. 56-59.

143. Курганов В.Ю. Современные тенденции развития и исследований в области реляционных систем управления базами данных // Математическое и программное обеспечение вычислительных систем: Межвуз. сб. науч. тр. / Рязань: РГРТА, 2002. С. 118-124.

144. Курош А.Г. Лекции по общей алгебре / М: «Наука», 1973 г. – 400 с.

145. Кюркчан А.Г., Скородумова Е.А. Решение трехмерной задачи дифракции волн на группе объектов //Акустический журнал. – 2007. – Т. 53. – №1. – С. 5 – 14.

146. Лаврищева Е.М., Петрухин В.А., Методы и средства инженерии программного обеспечения / М: МФТИ, - 2006, - 304 с.

147. Ладыженский Г.М. Архитектура корпоративных информационных систем. // Системы управления базами данных. 1997. № 5-6. с 5-11

148. Лейбниц Г.В. Сочинения: В 4-х т. Т. 3 / М.: Мысль, 1984. 734 с.

149. Летичевский А.А., Капитонова Ю.В., Летичевский А.А. (мл.) и др, Спецификация систем с помощью базовых протоколов // КИСА, 2005, № 4, С. 3-21

150. Летичевский А.А., Годлевский А.Б. и др., Свойства предикатного трансформера системы VRS // КИСА, 2010, № 4, С. 3-17

151. Львов В. Создание систем поддержки принятия решений на основе хранилищ данных / В. Львов // СУБД 1997 № 3, с. 30 – 40.

152. Ложкін О.М., Назаренко О.М. Дифракція пружних хвиль на періодичних системах циліндричних порожнин та жорстких включень // Акустичний вісник. – 2006. – Т. 9 – № 4 – С. 35-42

153. Ломако Е.И. Макетирование, проектирование и реализация диалоговых информационных систем / М.: Финансы и статистика, 1993, 320 с.

154. Лингер Р., Миллс Х., Уитт Б. Структурное программирование: Теория и практика / М.: Мир, 1982, 406 с.

155. Майерс Г. Архитектура современных ЭВМ: В 2-х книгах. Кн. 1. / М.: Мир, 1985.

156. Мальцев А.И. Алгебраические системы / М.: «Наука», 1970, - 392 с.

157. Маликов А.В. Проектирование реляционных баз данных на основе операций выборки и соединения. Исследование их свойств / Ставрополь: СевКавГТУ, 2002, 245 с.

158. Маликов А.В. К вопросу нормализации реляционных баз данных / On the problem of relational database normalization. Сборник «Кибернетика и технологии XXI века» по материалам IV Международной научно-технической конференции. Воронеж: ВГУ, 2003. с.79-86

159. Малыхина М.П. Базы данных: основы, проектирование, использование / СПб.: 2006, – 528 с.

160. Мамаев Е. Microsoft SQL Server 2000 в подлиннике / СПб.: BHV, 2005, 1277 с.

161. Мамиконов А.Г. и др, Оптимизация структур распределенных баз данных в АСУ / М.: Наука, 1990. – 240 с.

162. Мамчев Г.В. Особенности радиосвязи и телевидения. Учебное пособие для ВУЗов / М.: «Горячая линия – Телеком» - 2007 - 416 с.

163. Марка Д.А., МакГоуэн К., Методология структурного анализа и проектирования SADT / М.: МетаТехнология, 1993. — 239 с.

164. Марков А.С., Лисовский К.Ю. Базы данных: введение в теорию и методологию / М.: Финансы и статистика, 2006, 512 с.

165. Мартин Дж., Организация баз данных в вычислительных системах / М.: Мир, 1980. – 664 с.

166. Масленников А.А., Линьков В.М., Породников Е.А., Представление табличной информации в виде семанти ческих доменно-ориентированных структур данных // Информационный листок № 54-158-03, серия Р 20.53.19. - Пенза, 2003.

167. Масленников В.А., Левков А.А., Проблемы организации структуры данных в сверхбольших базах данных // Системы управления и информационные технологии, 2007, № 3.1 (29), С. 169-176.

168. Мацяшек Л.А. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML / М.: Вильямс, 2002. - 432 с.

169. Мейер Д. Теория реляционных баз данных / М.: 1987. – 608 с.

170. Мендельсон Э. Введение в математическую логику / М.: «Наука», 1976. – 320 с.

171. Мещеряков Б., Зинченко В. Большой психологический словарь / М.: Олма-пресс. 2004, 672 с.

172. Минский М. Фреймы для представления знаний / М: Энергия, 1979. - 151 с.

173. Мисевич П.В. Применение рабочей технологической модели для проектирования и сопровождения автоматизированных систем // Системы управления и информационные технологии. 2007. № 1.2 (27). - С. 248-253.

174. Михновский С.Д. Автоматизация проектирования баз данных. Общий анализ проблемы // Управляющие системы и машины. - 1981. - № 4. - С. 35-44.

175. Мюллер Р.Д. Базы данных и UML: проектирование / М.: «Лори», 2002, 432 с.

176. Нагао М., Катаяма Т., Уэмура С. Структура и базы данных / М.: «Мир», 1986, 197 с.

177. Назаренко А.М. Об одном подходе к исследованию волновых полей в пластинах с трещинами и включениями в рамках уточнённых теорий // Динамика и прочность машин. – 1989. – Вып. 50, С. 49–55.

178. Назаренко А.М. Дифракция волн сдвига на цилиндрических включениях и полостях в упругом полупространстве // Проблемы прочности. – 1990. – № 11. – С. 90 – 94.

179. Назаренко А.М. Дифракция гармонических волн на цилиндрическом упругом включении в условиях плоской деформации // Динамические системы. – 2005. – № 19. – С. 54–60.

180. Назаренко О.М., Ложкин О.М. Дифракция упругих гармонических волн на периолической системе криволинейных трещин в условиях плоской деформации // Прикладні проблеми механіки і математики. – 2006. – Вип. 4 , С. 162-169

181. Назаренко А.М., Острик В.И. Вынужденные колебания прямоугольной пластинки с тонким криволинейным включением // Изв. АН СССР. Механика твёрдого тела. – 1990. – № 4. – С. 93–98.

182. Назаренко А.М., Панченко Б.Е. Дифракция волн сдвига на цилиндрических неоднородностях произвольного поперечного сечения // Динамика и прочность машин. – 1991. – Вып. 52. – С. 38 – 45.

183. Назаренко А.М., Панченко Б.Е. Динамическая напряженность полосы с неоднородностями произвольной формы (антиплоская деформация) // Динамика и прочность машин. – Харьков, 1991. – Вып. 52. – С. 68–74.

184. Назаренко А.М., Панченко Б.Е. Взаимодействие волн сдвига с периодической системой цилиндрических неоднородностей произвольного поперечного сечения // Проблемы машиностроения. – 1992. – Вып. 38. – С. 48–52.

185. Назаренко А.М., Панченко Б.Е. Схема параллельных вычислений в задачах дифракции волн сдвига на системе отверстий в бесконечной упругой среде // Проблемы программирования, Киев. - 2010. – № 2-3, С. 604-610

186. Назаренко А.М., Панченко Б.Е., Ложкин А.М. Метод сингулярных интегральных уравнений в задачах дифракции упругих волн на цилиндрических включениях // Вісник СумДУ. Сер. Фізика, математика, механіка. – 2004. - № 8. - С. 144 – 150

187. Назаренко А.М., Панченко Б.Е., Ложкин А.М. Взаимодействие упругих волн с цилиндрической полостью в условиях плоской деформации // Вісник НТУ «ХПІ». Тематичний випуск: Динаміка і міцність машин. – 2005. - № 47 - С. 112- 117

188. Назаренко А.М., Панченко Б.Е., Ложкин А.М., Дифракция упругих волн на жестком цилиндрическом включении произвольного поперечного сечения // Вісник Донецького університету, Серія А: Природничі науки. – 2006. - № 3, С. 143-147

189. Назаренко А.М., Фильштинский Л.А. Взаимодействие упругих волн с криволинейной трещиной в полуплоскости // Теоретическая и прикладная механика. – 1988. – Вып. 19. – С. 77–82.

190. Нейбург Э., Максимчук Р. Проектирование баз данных с помощью UML / М.: Вильяме, 2002.-288 с.

191. Неклюдова Е.А., Цаленко М.Ш. Синтез логической схемы реляционных баз данных // Программирование. 1979, № 6. - с. 58 -68.

192. Непейвода Н.Н. Прикладная логика. Учебное пособие / Новосибирск, Изд. Новосиб универс. 2000. 521 с.

193. Новоженов Ю. В. Объектно-ориентированные технологии разработки сложных программных систем / М.: «Аргуссофт компании», 1998, 342 с.

194. Озкарахан Э. Машины баз данных и управление базами данных / М: «Мир», - 1989, -695 с.

195. Ойхман Е.Г., Попов Э.В. Реинжиниринг бизненса: Реинжиниринг организаций и информационные технологии. — М.: Финансы и статистика, 1997.- 332 с.

196. Оптнер С.Л. Системный анализ для решения деловых и промышленных проблем / М.: Сов. радио, 1969

197. Орешков В.И., Паклин Н.Б., Бизнес-аналитика: от данных к знаниям / СПб.: «Питер», - 2009, - 624 с.

198. Палагин А.В., Опанасенко В.Н. Реконфигурируемые вычислительные системы: Основы и приложения / К.: Просвита, 2006. – 280 с.

199. Палагин А.В., Петренко Н.Г. К вопросу системно-онтологической интеграции знаний предметной области // Математичні машини і системи. – 2007. – № 3-4. – С. 63–75.

200. Палагин А.В., Крывый С.Л., Петренко Н.Г. Онтологические методы и средства обработки предметных знаний / Луганск : "ВНУ", 2012, - 324 с.

201. Палей Д., Моделирование квази-структурированных данных // «Открытые системы», № 9, 2002, С.57-64

202. Панасюк В.В., Саврук М.П., Назарчук З.Т. Метод сингулярных интегральных уравнений в двумерных задачах дифракции / К.: «Наук. думка», 1984. – 344 с.

203. Панченко Б.Е., Стандартизация проектирования и генерации БД как единый путь решения задачи информатизации Украины // Глобальное информационное пространство: ресурсы, технологии, инновации. Тезисы докладов 5-й междунар. конф., 22-23 октября 1998, УкрИНТЭИ. – Киев, 1998, с. 58

204. Панченко Б.Е. Пат. Украины № 63036, Способ расположения данных в компьютерном хранилище, обеспечивающий модифицируемость его структуры // Промислова власність, - 2004. – № 1. – С. 3.134. – (дата заявки – 15.11.2001)

205. Панченко Б.Е. О шунтировании многозначной зависимости в реляционной модели данных // Проблемы программирования, – К., 2010. – № 2-3, – с. 428–423

206. Панченко Б.Е. Пат. Украины № 92248, Способ обобщенного размещения данных с учетом модифицируемости структуры хранилища // Промислова власність, - 2010. – № 19. – С. 3.131-3.132 (дата заявки – 02.03.2009)

207. Панченко Б.Е. Об алгоритме синтеза реляционного каркаса. Постановка задачи и формализация // Компьютерная математика – К., 2012. – № 1. – С. 84-93

206. Панченко Б.Е. Каркасное проектирование доменно-ключевой схемы реляционной базы данных // Кибернетика и системный анализ. – 2012. – № 3. – С. 174 – 187.

209. Панченко Б.Е. К вопросу о модифицируемости и безаномальности схемы реляционной базы данных // Проблемы программирования - К., 2012. – № 2-3. – С. 281-288

210. Панченко Б.Е. Исследования доменно-ключевой схемы реляционной базы данных // Кибернетика и системный анализ – К., 2012. – № 6. – С. 157–172

211. Панченко Б.Е. Численный анализ каркасной схемы реляционной базы данных // Компьютерная математика, – Киев. – 2012, - № 2, - С. 116-126

212. Панченко Б.Е., Пат. Украины № 99921, Способ предварительной каркасной сепарации данных перед их модифицируемым размещением в хранилище или процессом дальнейшей обработки // Промислова власність, - 2012 - № 20 - С. 3.30

213. Панченко Б.Е. Высокоточное кластерное решение задачи дифракции волн сдвига на системе отверстий в полубесконечной изотропной среде с защемленной границей / Проблемы программирования, Киев. - 2012. – № 1, С. 121-131

214. Панченко Б.Е., Поведение системы некруговых отверстий в полупространстве со свободной границей от воздействия стационарных SH-волн // Проблемы управления и информатики – К., 2012. - № 4. С. 84-93

215. Панченко Б.Е. Информационная модель поведения системы упругих волокон некруговой формы в полупространстве со свободной границей под воздействием стационарных SH-волн // Вестник СумДУ, 2013, – № 2, С. 31-42

216. Панченко Б.Е. Алгоритм синтеза реляционного каркаса. Неформальное описание // Проблемы управления и информатики, 2013, № 1, С. 83-103

217. Панченко Б.Е. Рекурсивные связи и темпоральность в реляционном каркасе - маски сущностей-объектов // Проблемы управления и информатики, 2013, № 2, С. 92-104

218. Панченко Б.Е. Хранилища данных на реляционном каркасе // Управляющие системы и машины, 2013, № 1, С. 71-84

219. Панченко Б.Е. Высокоточное параллельное решение задачи о дифракции волн сдвига на системе упругих включений в полупространстве с защемленной границей // Проблемы программирования, 2013, № 1, С. 116-124

220. Панченко Б.Е. Система упругих волокон некруговой формы в полупространстве со свободной границей под воздействием стационарных SH-волн // Проблемы управления и информатики, 2013, № 6, С. 112-122

221. Панченко Б.Е., Гайдабрус В.Н., Реляционный каркас и модель CASE-оболочки нового типа // Кибернетика и системный анализ, 2013, № 3, С. 172-186

222. Панченко Б.Е., Гайдабрус В.Н., Церковицкий С.Л. Создание сетевых информационных комплексов // Компьютеры плюс программы. – 1993. – № 5 (6). – С. 46 – 50.

223. Панченко Б.Е., Гайдабрус В.Н., Церковицкий С.Л., Сетевые вычислительные комплексы // Компьютеры плюс программы, Киев, 1994. - спец. вып, с. 30-37

224. Панченко Б.Е., Гайдабрус В.Н., Церковицкий С.Л., Инструментальная система SWS: CASE-генерация сетевых АСУ // Компьютерные технологи в промышленности. Тезисы докладов междунар. конф., 4-6 октября 1994, Общество «Знание», Киев, 1994, с. 59

225. Панченко Б.Е., Гайдабрус В.Н., Церковицкий С.Л., CASE-генератор прикладных сетевых информационных комплексов – инструментальная система SWS 1.0 // Свидетельство об официальной регистрации программы для ЭВМ № 940165, - М.: РосААП, - 1994, с. 1-2

226. Панченко Б.Е., Гуреев Д.С. Численное сравнение параметров доступа к n-арным и бинарным отношениям // Информатика, математика, механика. Тезисы докладов IV межвуз. конф., 21-24 апреля 2009, Сумы. – 2009, с. 145

227. Панченко Б.Е., Денисенко В.И., Сидорец И.И., Графическая система MonoCAD // «Технология программирования 90-х». Тезисы докладов 2-й междунар. конф., Киев, 1992., с. 63

228. Панченко Б.Е., Кривомаз Л.С., Многокамерная прямая трансляция как метод эффективного дистанционного обучения / Инновационное развитие общества в условиях кросс-культурных взаимодействий. Тезисы докладов 2-й междунар. конф., 27-30 апреля 2009, - Сумы. – 2009, Том 3, с. 18

229. Панченко Б.Е., Крючко Е.В., Исследование элементов OLAP–решений на реляционном каркасе // Компьютерная математика, 2013, № 1, С. 123-130

230. Панченко Б.Е., Назаренко А.М., Каркасный анализ предметной области: стационарные динамические задачи теории упругости для изотропных сред с произвольными неоднородностями // Кибернетика и системный анализ, - 2013, - № 1, С. 172-187.

231. Панченко Б.Е., Печенюк Д.А., Каркасный анализ способов коммутации видеосигналов // Управляющие системы и машины, 2013, № 5, С. 53-64

232. Панченко Б.Е., Писанко И.Н., Свойства реляционного каркаса на множестве семантически атомарных предикатов // Кибернетика и системный анализ, - Киев, 2009. – № 6. – С. 120-129

233. Панченко Б.Е., Писанко И.Н., О полноте и единственности универсального каркаса в реляционной модели данных // Системные исследования и информационные технологии, Киев, 2010, № 3, С. 25-35

234. Панченко Б.Е., Писанко И.Н., О топологии путей нормализации в реляционном каркасе // Системные исследования и информационные технологии, Киев, - 2011, - № 3. - с 102-107

235. Панченко Б.Е., Шаповалов В.Н., OLAP в реальном времени: параллельные вычисления на каркасной модели // Инновационное развитие общества в условиях кросс-культурных взаимодействий. Тезисы докладов 3-й междунар. конф., 26-29 апреля 2010, - Сумы. – 2010, том. 2, с. 253-255

236. Пасичник В.В., Шаховская Н.Б. Хранилища данных. Учебное пособие / Львов: 2006 г. - 492с.

237. Педерсен Т.Б., Йенсен К.С., Технология многомерных баз данных // Открытые системы. - 2002. - № 1. - С. 45-50.

238. Пергаменцев Ю.А., Проектирование БД на основе универсальной модели данных // Материалы седьмой технической конференции «Корпоративные базы данных'2002», 16-17 апреля 2002 года, М.: 2002 http://www.citforum.ru/seminars/cbd2002/

239. Перевозчикова О.Л. Основы системного анализа объектов и процессов компьютеризации. Учебник / К.: «КМ Академия», 2003. - 247 с.

240. Перевозчикова О.Л. Информационные системы и структуры данных. Учебное пособие / К.: «КМ Академия», - 2007. - 287 с.

241. Перевозчикова О.Л., Тульчинский В.Г., Панченко Б.Е., Коломиец А.В. и др., Высокопродуктивные методы анализа и спецификации пространств атрибутов предметной области для организации вычислений // Отчет о НИР № 0107U000800 ВФ.145.09.11. - Киев, 2011, - 378 с.

242. Петренко С.И. SWS в «Аргументах и фактах» // Украинские телевизионные новости, НТКУ, 1994. видеодокумент, Ютуб, http://www.youtube.com/watch?v=42ZQTjteAoE

243. Петренко С.И., Каркасная модель данных и SWS для г. Измаил / Компьютер-Х, 1995, вып. 6, видеодокумент, Ютуб,

http://www.youtube.com/watch?v=PcXBzLkPczo

244. Петренко С.И. Успех украинской SWS в «Останкино»// Украинские телевизионные новости, НТКУ, 1994. – видеодокумент, Ютуб,

http://www.youtube.com/watch?v=1ePq6eQP2qc

245. Плоткин Б.И. Универсальная алгебра, алгебраическая логика и базы данных / М.: Наука, 1991. – 448 с.

246. Погодаев А.К., Анисимова Т.А., Батищев Р.В. Объектно-реляционный подход в распределенных базах данных // Теория и практика производства проката, Сб. научн. трудов международной научн.-техн. конф., посвященной памяти С.Л. Коцаря. - Липецк: ЛГТУ, 2001. С.397-401.

247. Погодаев А.К., Бурцев В.Д., Объектно-реляционная модель сложного производства // Современные проблемы информатизации в технике и технологиях: Труды VI Международной открытой научн. конф. - Воронеж: ВЭПИ, 2001. С.22-23.

248. Полищук Ю.М. Автоматизированные банки информации / М.: «Мир», 1989.

249. Полухин А.Л. Операторы темпорального расширения реляционной модели данных. // Вестник С.-Петерб. ун-та. Сер. 10, 2006. Вып. 1, - с. 123- 132

250. Попов С.Г. Методика построения оптимального репозитория схем реляционных баз данных // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Серия «Информатика. Телекоммуникации. Управление.» - СПб.: СПбГПУ, 2010. - № 5 (108). - С. 91-98.

251. Поспелов Д.А. Ситуационное управление: теория и практика / М.: Наука, 1986. – 288 с.

252. Поспелов Д.А. Где исчезают виртуальные миры? // Новости искусственного интеллекта. 2003. № 3. С. 9-25.

253. Пратт Т., Зелковиц М. Языки программирования: разработка и реализация / СПб.: Питер, 2002. – 688 с.

254. Пржиялковский В.В. Абстракции в проектировании баз данных // СУБД. - 1998. - №1-2. - С. 90-97

255. Пржиялковский В.В. Что объектам здорово и что реляциям смерть, и наоборот, и еще пол-оборота // PC Week/RE (146) 22, 1998, с. 7

256. Прилуцкий М.Х., Раппопорт И.А. Ситдиков А.Х. Распределение ресурсов в иерархических системах древовидной структуры // Вестник ВГАВТ. Моделирование и оптимизация сложных систем: межвуз. сб. тр., ВГАВТ. - Н.Новгород, 2002. Вып 1. С.37 - 39.

257. Прохоров А. Использование объектно-реляционных СУБД для хранения и анализа временных рядов // КомпьютерПресс. - 2001. - № 6, с. 7-9

258. Пуле М. Денормализация: как нарушить правила и избежать последствий // SQL Server Magazine, 2000, September, InstantDoc ID #9785.

259. Райордан Р., Основы реляционных баз данных / Русская редакция, 2001, 352 с.

260. Рассел С., Норвиг П., Искусственный интеллект: современный подход / М.: Вильямс, 2006. - 1408 с.

261. Ревунков Г.И., Самохвалов Э.Н., Чистов В.В. Базы и банки данных и знаний / М.: Высшая школа, 1992, - 368 с.

262. Редреев П.Г. Построение табличных приложений со списочными компонентами // Информационные технологии. - 2009. - № 5. - С. 7-12.

263. Редько В.Н., Брона Ю.Й., Буй Д.Б., Поляков С.А., Реляційні бзи даних: таблична алгебра та SQL-подібні мови / К.: «Академперіодика», 2001. – 198 с.

264. Рейнгольд Э., Нивергельт Ю., Део Н. Комбинаторные алгоритмы. Теория и практика / М.: Мир, 1980. - 476 с.

265. Роб П., Коронел К., Системы баз данных: проектирование, реализация и управление / БХВ-Петербург, 2004, 1024 с.

266. Роджерс Х, Теория рекурсивных функций и эффективная вычислимость / М.: «Мир», 1972, 624 с.

267. Ролланд Ф.Д., Основные концепции баз данных / Вильямс, 2002, 256 с.

268. Саймон А.Р. Стратегические технологии баз данных: менеджмент на 2000 год / М.: «Финансы и статистика», 1999. – 479 с.

269. Сахаров Л.Л. Принципы проектирования и использования многомерных баз данных (на примере Oracle Express Server) // СУБД. - 1996. - № 3. - С. 44-59.
270. Сахаров Л.Л. Концепция построения и реализации информационных систем, ориентированных на анализ данных // СУБД. 1996. - № 4. - С. 55-70.
271. Селезов И.Т., Кривонос Ю.Г., Яковлев В.В. Рассеяние волн локальными неоднородностями в сплошных средах / К.: Наук. думка, 1985. – 136 с.

272. Сергиенко И.В., Кривый С.Л., Провотар О.И. Алгебраические аспекты информационных технологий / К.: Наукова думка. - 2012. - 400 с.
273. Ситник Н.В. Проектирование баз и хранилищ данных. Учебное пособие / К.: «КНЕУ», 2004. – 348 с.
274. Смирнова Е.Д. Формализованные языки и проблемы логической семантики / М., 1982, 181 с.
275. Смирнов Г.Л., Сорокин А.А., Тельнов Ю.Ф., Проектирование экономических информационных систем / М.: Финансы и статистика. 2001. - 502 с.
276. Смит Дж., Смит Д. Принципы концептуального проектирования баз данных // В сб.: Требования и спецификации в разработке программ, под ред. В.Н. Агафонова. М.: Мир, 1984. С. 165-198.
277. Соколов А. Г. Монтаж: телевидение, кино, видео / М.: Изд. «625», - 2001 – 207 с
278. Соколинский Л.Б. Параллельные машины баз данных // Природа. Естественно-научный журнал Российской академии наук. -2001. - № 8. - С. 10-17
279. Соловьев В.Д., Добров Б.В., Иванов В.В., Лукашевич Н., Онтологии и тезаурусы. Учебное пособие / М.: МГУ, 2006. – 157 с.
280. Солтон Дж. Динамические библиотечно–информационные системы / М.: Мир, 1979. – 557 с.

281. Соммервилл И. Инженерия программного обеспечения / М.: Вильямс, 2002. — 624 с.

282. Спирли Э., Корпоративные хранилища данных. Планирование, разработка, реализация / М.: Вильямс, 2001. - 400 с.

283. Степанов А.И. Число и культура / М.: Мир, 2004 г. – 832 с.

284. Стулов А.С. Особенности построения информационных хранилищ // Открытые системы, М., 2003, № 04, с. 76-79.

285. Ступников С.А., Калиниченко Л.А. Формирование базы метаинформации на основе спецификаций канонической модели // Материалы третьей международной конференции по программированию. УкрПРОГ’2002. Проблемы программирования, № 1-2, 2002. С. 301-308.

286. Тамер Оззу М., Валдуриз П. Распределенные и параллельные системы баз данных // СУБД, 1996 - № 4. С. 4-26.

287. Тарасов И.А. Целостность данных, аномалии модификации данных и нормальные формы таблиц реляционных баз данных // Проектирование телекоммуникационных и информационных средств и систем. М.: МИЭМ, 2007, с. 195.

288. Тарасов И.А., Метод проектирования логической структуры реляционной БД для веб-приложений без нормализации таблиц: дисс. к.т.н. / М: МИЭМ, - 2009 г. – 112 с.

289. Тетерин В.С. Особенности режиссуры телевидения при многокамерном методе съемок / М.: ВГИК, - 1971. – 105 с.

290. Таха Х.А. Введение в исследование операций / М.: Вильямс, 2007. – 912 с.

291. Тиори Т., Фрай Дж. Проектирование структур баз данных / М.: Мир, 1985. – Кн. 1. – 287 с.; Кн. 2. – 320 с.

292. Ту Дж., Гонсалес Р. Принципы распознавания образов / М.: Мир. -1978. - 411 с.

293. Тудер И.Ю. Коллективный анализ предметной области // Банковские технологии, М. «Бизнес и компьютер», № 5, май, 2001, с. 32-38.

294. Тукеев У.А., Алтайбек А.А. Концептуальная, логическая модели и алгоритм проектирования баз данных в доменно-ключевой нормальной форме // Труды 13-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» (RCDL'2011) - Воронеж, 2011. - с. 119-125

295. Тульчинский В.Г., Тульчинский П.Г. Графовый прототип приложения // Проблемы программирования. – 2002. – №1-2. – С.489-498

296. Туо Дж. Каждому пользователю - свое представление данных / Дж. Туо // ComputerWeek-Москва. — 1996. — № 38. — С. 32—33

297. Туо Дж., Инструменты для анализа информации на настольных ПК // ComputerWeek-Москва. - 1996. - № 38. - С. 34-35, 46.

298. Тюхтин В.С., Урманцев Ю.А., Система. Симметрия. Гармония / М.: Мысль, 1988.-315 с.

299. Уилсон П., Борер Т., Дэвид Т., Семейство систем цифрового сжатия DIRAC расширяется / Журнал «625», - Москва, - № 3, 2007, с. 63-67

300. Ульман Д.Д., Уидом Д. Основы реляционных баз данных / М.:2006 г. – 374 с.

301. Урманцев Ю.А. Эволюционика или общая теория развития систем природы, общества и мышления / М.: Либроком, 2009. - 240 с.

302. Успенский В.А. Теорема Геделя о неполноте / М.: Наука, 1982, 112 с.

303. Фаулер М., Скотт К. UML. Основы / М: Символ-Плюс, 2002. - 192 с.

304. Федоришин В.И., Кривомаз Л.С., Панченко Б.Е. и др., Микола Колесса – син століття, видеоверсия концерта / Ютуб, 2007, http://www.youtube.com/watch?v=8kv_4y-qDT4

305. Федоришин В.И., Кривомаз Л.С., Панченко Б.Е. и др., Украинская музыка в мировой культуре, видеоверсия концерта / Ютуб, 2009, http://www.youtube.com/watch?v=BPoTin46UBQ

306. Филиппович А.Ю. Принципы взаимных функциональных зависимостей // Интеллектуальные технологии и системы. Сборник статей. Вып. 4. - М.: Изд-во МГУП, - 2002, - с. 222-241

307. Фильштинский Л.А. Дифракция упругих волн на трещинах, отверстиях, включениях в изотропной среде // Изв. АН СССР. Механика твердого тела. – 1991. – №4. – С. 119 –127.

308. Франклин М., Хэлеви Э., Майер Д. От баз данных к пространствам данных: новая абстракция управления информацией // SIGMOD Record, Vol. 34, № 4, Dec. 2005, p. 27-33

309. Хаббард Дж. Автоматизированное проектирование баз данных / М.: «Мир», 1984. – 296 с.

310. Хансен Г., Хансен Д. Базы данных: разработка и управление / М.: «Бином», 1999. – 704 с.

311. Харрингтон Д.Л. Проектирование объектно-ориентированных баз данных / М: ДМК, 2000. - 272 с.

312. Харрингтон Д.Л. Проектирование реляционных баз данных / М.: Лори, 2006, 231с.

313. Хернандес М.Д., Вьескас Д.Л. SQL-запросы для простых смертных / М.: Лори, 2003, 460 с.

314. Химич А.Н., Молчанов И.Н., Попов А.В. Численное программное обеспечение интеллектуального MIMD-компьютера «Инпарком» / К.: Наук. думка. – 2007. – 220 с.

315. Химич А.М., Полянко В.В. Эффективность двумерных блочно-цикличных параллельных алгоритмов // Проблемы программирования. – 2008. – №3. – С. 145–149.

316. Хоббс Л., Хилсон С., Лоуренд Ш., Разработка и эксплуатация хранилищ данных (Oracle 9iR2) / М.: Кудиц-образ, 2004, 586 с.

317. Хомоненко А.Д. Базы данных / А.Д. Хомоненко, В.М. Цыганков, М.Г. Мальцев. - СПб: Корона-Принт, 2004. - 736 с.

318. Хопкрофт Дж., Мотвани Р., Ульман Дж. Введение в теорию автоматов, языков и вычислений / М.: Вильямс, 2007. - 528 с.

319. Цаленко М.Ш. Семантические и математические модели баз данных // Итоги науки и техники. Информатика. Том 9 / М: ВИНИТИ – 1985. - 207 с.

320. Цветков В.Я. Геоинформационные системы и технологии / М.: Финансы и статистика, 1998, 288 с.

321. Цейтлин Г.Е. Алгебраическая алгоритмика: теория и приложения // Кибернетика и системный анализ. – 2003. – № 1. – С. 8 – 18.

322. Цикритзис Д. Модели данных / Д. Цикритзис, Ф. Лоховски. - М.: Финансы и статистика, 1985. – 344 с.

323. Чарнецки К., Айзнекер У. Порождающее программирование: методы средства и приложения / СПб.: Питер, 2005. – 736 с.

324. Чаудхари С. Методы оптимизации запросов в реляционных системах // СУБД. - 1998. - № 3. - С.22-36

325. Чаудхари С., Дайал У., Гаити В., Технология баз данных в системах поддержки принятия решений // Открытые системы. - 2002. - № 1. - С. 37-44.

326. Чемберлин Д., Анатомия объектно-реляционных баз данных // СУБД, - Москва, 1998. - № 1-2. – С. 65-76

327. Чери С., Готлоб Г., Танка Л., Логическое программирование и базы данных / Мир, - 1992, - 352 с.

328. Шаллоуей А., Тротт Дж.Р. Шаблоны проектирования. Новый подход к объектно-ориентированному анализу и проектированию / М.: Вильямс, 2002. - 288 с.

329. Швецов В.И., Визгунов А.Н., Мееров И.Б. Базы данных / Н.Новг.: ННГУ, - 2004. 271 с.

330. Шлеер С., Меллор С. Объектно–ориентированный анализ: моделирование мира в состояниях / К.: Диалектика, 1993.– 240 с.

331. Шумаков П.В Delphi 3 и создание приложений баз данных / М.: «Нолидж», 1998, 704 с.

332. Щавелев Л.В. Способы аналитической обработки данных для поддержки принятия решений // СУБД. - 1998. - № 4-5.- С. 51-60.

333. Эдельман С.Л. Математическая логика / М., Наука, 1975. - 176 с.

334. Эмблер С.В., Садаладж П.Дж., Рефакторинг баз данных: эволюционное проектирование / М.: Вильяме, 2007. – 368 с.

335. Яловец А.Л. Представление и обработка знаний с точки зрения математического моделирования. Проблемы и решения / К: Наук. думка. - 2011. - 360 с.

336. Янов Ю.И. Математика, метаматематика и истина / М.: Институт прикладной математики им. М.В. Келдыша, 2006. 32 с.

337. Abiteboul S., Beeri C. On the Power of Languages for the Manipulation of Complex Values. Technical report / Cadex (France): INRIA, 1995. – 80 p.

338. Adelsberger H., Komer F. Data Modeling with IDEFiX: // Lect. Notes Comput. Sci. —1995. — Vol. 973 — P. 355-391.

339. Allen S., Terry E. Beginning relational data modelling / Apress, 2005, 632 p

340. Ambler S.W., Sadalage P.J. Refactoring databases: evolutionary database design / Addison-Wesley Professional, - 2006, - 384 p.

341. Andany J., Leonard M., Palisser C. Management of schema evolution in databases // in Proc. Conf. Very Large Databases, Barcelona, Spain — 1991. — P. 161-170.

342. Ariav G. Temporally oriented data definitions: managing schema evolution in temporally oriented databases // Data Knowledge Eng.— 1991.— 6 (6) — P.451-467.

343. Avison D.E., Fitzgerald G. Information Systems Development: Methodologies, Techniques and Tools / McGraw-Hill, 2002, 505 p.

344. Bagui S., Earp R. Database design using entity-relationship diagrams / Auerbach Publications, - 2003, - 242 p.

345. Baralis E., Paraboschi S., Teniente E., Materialized views selection in a multidimensional database // Proc. of the 23rd Inter. Conf. on Very Large Data Bases, Eds. Very Large Data Bases. Morgan Kaufmann Publishers, San Francisco, CA. - 1997. - P. 156-165

346. Barker R. CASE-Method. Entity-Relationship Modelling. Copyright Oracle Corporation UK Limited, Addison-Wesley Publishing Co., 1990. – 310 p.

347. Barker R. CASE-Method. Function and Process Modeling / Copyright Oracle Corporation UK Limited, Addison-Wesley Publishing Co, 1990. – 280 p.

348. Barwise J., Etchemendy J., Language, proof and logic / Seven Bridges Press, 1999, 567 p.
349. Bernstein P., Swenson J., Thichritzis D.A. Unified Approach to Functional Dependencies and Relations // Proc. International Conference on the Management of Data. – ACM SIGMOD, 1975. – P.237–245.
350. Bernstein P.A. Synthesizing third normal form relation from functional dependencies // ACM Transactions on Database Systems V. 1, № 4, 1976. P. 277-298.
351. Barwise J., Etchemendy J. Language, proof and logic / NY: Seven Bridges Press, 2000. – 387 p.
352. Bettini C., De Sibi R. Symbolic Representation of User-defined Time Granularities // Annals of Mathematics and Artificial Intelligence, 2000, Vol. 30, № 1-4, p. 53-92
353. Beynon-Davies P. Database systems / Palgrave Macmillan, 2004, 601 p.
354. Blakeley J.A., Larson P.-A., Tompa F.W., Efficiently updating materialized views // Proceedings of the 1986 ACM SIGMOD international conference on Management of data, May 28-30, Sigmod Record, 1986, p. 61-71
355. Brachman R.J., Levesque H.J. Knowledge representation and reasoning, Morgan Kaufmann, 2004, 381 p.
356. Brown W., Wallnau K. The Current State of CBSE // IEEE Software, Volume 15, Issue 5, 1998, p. 37-46
357. Cabibbo L., Torlone R., A logical approach to multidimensional databases // Proc. of the 6th Inter. Conf. on Extending Database Technology: Advances in Database Technology, Springer-Verlag, London. — 1998. -Vol. 377.-P. 183-197
358. Carver A., Halpin T. Atomicity and normalization // Proceedings of the 13th International Workshop on Exploring Modelling Methods for Systems Analysis and Design (EMMSAD08). - Montpellier, France, 2008. - P. 40-54.
359. Chen P.P. The Entity-Relationship Model: toward a unified view of data // ACM Trans. on Data base systems, v. 1, № 1, 1976. P. 9 – 36.

360. Churcher C. Beginning database design: from novice to professional / Apress, 2007, 240 p.
361. Codd E.F. The Relational Model For Database Management, Version 2, Reading Mass. – New York: Addison-Wesley Publishing Co, 1990. – 538 p.
362. Codd E.F. Extending the database relational model to capture more meaning // ACM Transactions on Database Systems, Vol. 4, № 4, 1979. - P. 397- 434.
363. Cohen J., Northrop L.M., Object-Oriented Technology and Domain Analysis // Proceedings of the Fifth International Conference on Software Reuse. - IEEE Computer Society Press, 1998. - P. 86-93.
364. Cohen J., Dolan B., Dunlap M. and as, MAD Skills: New Analysis Practices for Big Data. Proceedings of the VLDB'09 Conference, Lyon, France, August 24-28, 2009
365. Cohen W. Integration of heterogeneous databases without common domains using queries based textual similarity // In proceedings of the ACM SIGMOD International Conference on Management of Data, 1998. P. 201-212.
366. Darwen H., Date C.J. The third manifesto // ACM SIGMOD Records, Vol. 24, № 1, 1995. - P. 39-49.
367. Date C.J., Fagin R. Simple Conditions for Guaranteeing Higher Normal Forms in Relational Databases // ACM Transactions on Database Systems, Vol. 17, No. 3, September 1992, P. 465-476
368. Date C.J. Date on database: writings 2000-2006 / New York: Apress, 2006. – 539 p.
369. Date C.J., Darwen H., Lorentzos N.A., Temporal data and the relational model, Morgan Kaufmann, 2003, 425 p.
370. Delobel C., Caseyr R.G. Decomposition of a data base and the theory of Boolean switching functions // IBM J. Res. and Dev., 1973. - V. 17,- № 5.-p. 374-386.
371. DeWitt D.J., Gray J. Parallel Database Systems: The Future of High-Performance Database Systems // Communications of the ACM. -1992. -Vol. 35, № 6. - P. 85-98.

372. Duzi M., Jaakkola H., Kiyoki Y., Kangassalo H. Information modelling and knowledge bases, vol. XVIIIIOS Press 2007, 322 p.

373. Drewek K. Data Warehousing: Our Great Debate Wraps Up // 2005, http://www.b-eye-network.com/view/766

374. Drewek K. Data Warehousing: Similarities and Differences of Inmon and Kimball / 2005, http://citforum.ru/database/articles/mad_skills/

375. Ehlmann B.K. Object relationship notation (ORN) for database applications: enhancing the modeling and implementation of associations / Springer, 2009, 246 p

376. Elmagarmid A.K., Pu C. Introduction to the Special Issue on Heterogeneous Databases // ACM Computing Surveys, 22, 1990, p. 175-178

377. Fagin R. Multivalued dependencies and a new normal form for relational databases // ACM Transactions on Database Systems, 1977, Vol. 2, № 3, P. 262 - 278.

378. Fagin R. A Normal Form for Relational Databases That Is Based on Domains and Keys// ACM Transactions on Database Systems, 1981, Vol. 6, № 3, P. 387-415.

379. Fankam Ch., Jean S., Bellatreche L., Ait-Ameur Y., Extending the ANSI/SPARC Architecture Database with Explicit Data Semantics: An Ontology-Based Approach // Software Architecture. Second European Conference, ECSA 2008 Paphos, Cyprus, September 29 - October 1, 2008. - Springer. - 2008.- P. 318 -321

380. Ferragine V.E., Doorn J.H., Rivero L.C. Handbook of research on innovations in database technologies and applications: current and future trends Information, Science Reference 2009, 1051 p.

381. Fisher A.S. CASE: Using Software Development Tools. N.Y.: J.Wiley & Sons Inc., 1988. – 322 p.

382. Flouris G., Monakenates D., Kondylakis H. et al, Ontology Chenge: classification and survey // Knowl. Eng. Rev., 2008, 23, No 2, C. 117-152

383. Florescu D., Kossmann D. Rethinking Cost and Performance of Database Systems // SIGMOD Record, Vol. 38, № 1, March 2009, p. 43-48
384. Fotache M. Database Designers and Normalization: Anatomy of a Divorce // Proceedings of the 8th International Conference on Business Information Systems, Poznan, Poland, 2005, p. 330-341
385. Fotache M. Why normalization failed to become the ultimate guide for database designers // Social Science Research Network, Information Technology & Systems, Working Paper Series, Vol. 2, № 9, June 2006, p. 3-41
386. Galindo J. Handbook of research on fuzzy information processing in databases / Information Science Reference, 2008, 875 p.
387. Gardarin G., Valduriez P. Relational databases and knowledge bases / Addison-Wesley Publishing Company, Reading, Massachusetts, 1990, 448 p.
388. Garcia H.-M., Chen S.L., Olivia R., An entity-relationship-based methodology for distributed database design: An integrated approach towards combined logical and distribution designs // Lecture Notes in Computer Science, V. 645, 1992. – P. 178-193.
389. Garmany J., Walker J., Clark T. Logical database design principles, Auerbach Publications, 2005, 180 p.
390. Gomez-Perez A., Fernandez-Lopez M., Corcho O. Ontological engineering / Springer, 2003, 403 p.
391. Golfarelli M., Maio D., Rizzi S. The dimensional fact model: a conceptual model for data warehouses // Inter. Journal of Cooperative Information Systems. - 1998. - № 7. - P. 215-247
392. Golfarelli M., Rizzi S., A methodological framework for data warehouse design // Proc. of the 1st Inter. Workshop on Data Warehousing and OLAP, Maryland, USA. - 1998. - P. 3-9
393. Golfarelli M., Rizzi S., A methodological framework for data warehouse design // Proc. of the 1st Inter. Workshop on Data Warehousing and OLAP, Maryland, USA. - 1998. - P. 3-9

394. Grechko V., Tulchinsky V. Building standard user interface: DBMS "MicroPoisk" approach // Informatica. - 1995. - № 4. - P. 446-456.
395. Gray J., Bosworth A., Layman A., and Pirahesh H. Data cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-total // In Proc. 12th Int. Conf. on Data Engineering, 1996, P. 152–159.
396. Gruber T.R. A translation approach to portable ontologies // Knowledge Acquisition, 1993, V. 5, № 2, P. 199-220.
397. Gruber, T.R., Toward Principles for the Design of Ontologies Used for Knowledge Sharing // International Journal Human-Computer Studies, Vol 43, № 5-6, 1995. - p. 907-928
398. Gruber T.R., Ling Liu, Tamer Özsu M., Ontology // Encyclopedia of Database Systems / Springer-Verlag, 2009, p. 1963-1965
399. Guarino N. Formal Ontology and Information Systems // Proceedings of the 1st International Conference on Formal Ontologies in Information Systems, FOIS'98, Trento, Italy, IOS Press, June 1998, P. 3-15.
400. Guarino N. Some Ontological Principles for Designing Upper Level Lexical Resources // Proceedings of First International Conference on Language Resources and Evaluation, 1998. p. 527–534
401. Haan L., Koppelaars T. Applied mathematics for database professionals / Apress, 2007, - 376 p.
402. Halpin T., Morgan T. Information modelling and relational databases / Morgan Kaufmann, 2008, 976 p.
403. Hanson E.N. A performance analysis of view materialization strategies // Proc. of the ACM SIGMOD Conf. on Management of Data, - 1987, - P. 440-453
404. Harinarayan V., Anand R., Ullman J.D. Implementing data cubes efficiently // Proc. of the 1996 ACM SIGMOD Inter. Conf. on Management of Data, Quebec. 1996. - P. 205-216.
405. Harrington J.L. Relational database design: clearly explained / Morgan Kaufmann, 2002, 393 p.

406. Helbig H. Knowledge representation and the semantics of natural language / Springer, 2006, 646 p.
407. Hellerstein J.M., Stonebracker M., Hamilton J. Architecture of a database system / Now Publishers Inc, 2007, 123 p.
408. Hernandez M.J. Database design for mere mortals / Addison-Wesley, 2003, 440 p.
409. Hoare C.A. An axiomatic basis for computer programming // Communications ACM, 1969, Vol. 12, № 10. - P. 576-583
410. Hoberman S. Data Modeling Made Simple: A Practical Guide for Business and IT Professionals / Take IT With You, 2009. - 360 p.
411. Hoffer J.A. Prescott M.B., McFadden F.R. Modern database management / Prentice Hall, 2007, 622 p.
412. Hull R., Zhou G. A framework for supporting data integration using the materialized and virtual approaches // In proceedings of the ACM SIGMOD International Conference on Management of Data, 1996. p. 481-492.
413. Imhoff C. Galemmo N., Geiger J.G. Mastering Data Warehouse design: relational and dimensional techniques / John Wiley & Sons, 2003, 438 p.
414. Inmon W. H. Building The Data Warehouse / W. H. Inmon. - NY: John Wiley & Sons. – 1993, 298 p.
415. Inmon W.H. Building the Data Warehouse / John Willey & Sons, 2002, 412 p.
416. International Standart 9075, Database Language SQL, AMENDMENT 1: On–Line Analytical Processing (SQL/OLAP), ISO/IEC, 2001
417. Isloor S.S. An algorithm with logical simplicity for designing third normal form relational database schema from functional dependencies // Proc. of Int. Conf. on DBMSs (ICMOD 78), Fast Milan, Italy. - 1978. - p. 31 - 50.
418. Jackson D. Software abstractions: logic, language, and analysis / The MIT Press, 2006. – 350 p.
419. Jacobs A. The Pathologies of Big Data / ACM Queue, Vol. 7, № 6, July 2009, p. 10-19

420. Jacobson I., Christerson, M., Jonsson, P., and Overgaard, G. Object-Oriented Software Engineering: A Use Case Driven Approach / Addison-Wesley, Reading, Massachusetts, 1992, 273 p.
421. Jensen C.S. et al. A Consensus Glossary of Temporal Database Concepts // ACM SIGMOD Record — 1994. — Vol. 23, № 1 — P. 52-64.
422. Khosrow-Pour M. Cases on database technologies and applications / Idea Group, 2006, 292 p.
423. Kimball R. The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses / John Willey & Sons, New York, 1996, 491 p.
424. Kimball R. Meta Meta Data Data // DBMS magazine. March 1998, p. 21-31
425. Kimball R., Ross M., The data warehouse toolkit: the complete guide to dimensional modeling / John Wiley & Sons, Inc. New York, 2002. - 416 p.
426. Kim W., Ballou N., Chou H.-T., Garza J.F., Woelk D. Features of the Orion object-oriented database system // in Object-oriented Concepts, Databases and Applications. W. Kim and F. Lochovsky (eds.), ACM Press, New York — 1989. — P.251-282.
427. Kiyoki Y. Henno J., Jaakkola H., Kangassalo H. Information modelling and knowledge bases XVII / IOS Press Amsterdam, 2006, 341 p.
428. Klas W. Schrefl M. Metaclasses and their application: data model tailoring and database integration / Springer, 1995, 203 p.
429. Knobbe A.J. Multi-relational data mining / IOS Press, 2006, 118 p.
430. Krogstie J. Halpin T., Siau K., Information modeling methods and methodologies / Idea Group, 2005, 356 p.
431. Krogstie J. Opdahl A.L., Brinkkemper S., Conceptual modelling in information systems enginnering / Springer, 2007, 342 p.
432. Lahdenmaki T., Leach M. Relational database index design and the optimizers / John Wiley & Sons, 2005, 310 p.
433. Lacroix Z., Delobel C., Brèche Ph., Object Views and Database Restucturing // Database Programming Languages, Lecture Notes in Computer Science, 6th

International Workshop, DBPL-6 Estes Park, Colorado, USA, August 18–20, 1997 Proceedings, Volume 1369, 1998, p. 180-201

434. Lenzerini M. Data Integration: A Theoretical Perspective // Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (Proceedings of ACM PODS), 2002. P. 233–246.

435. Lerner B.S., Habermann A.N. Beyond schema evolution to database reorganization // SIGPLAN Notices — 1990. — Vol. 25, № 5 — P.67-76.

436. Lightstone S., Teorey T., Nadeau T. Physical database design / Morgan Kaufmann, 2007, 425 p.

437. Lientz B.P. Issues in software maintenance // ACM Computing Surveys — 1983. — 15(3) — P. 271-278.

438. Ling T.W., Lee M.L., Dobbie G. Semistructured database design / Springer, 2005, 177 p.

439. Linger R., Mills H., Witt B. Structured Programming: Theory and Practice. Reading / MA: Addison-Wesley, 1979 - ?

440. Litwin W., Mark L., Roussopoulos N. Interoperability of multiple autonomous databases // ACM Computing Surveys, 22, 1990, p. 267-293

441. Liu L., Ozsu M.T. Encyclopedia of database systems / Springer, 2009, 3749 p.

442. Lu G. Multimedia Database Management System / Artech House, 1999, 374 p.

443. Ma Z. Database modeling for industrial data management: emerging technologies and applications / Idea Group, 2006, 374 p.

444. Ma Z. Intelligent databases: technologies and applications / Idea Group, 2007, 320 p.

445. Maier D. Why isn’t there an object-oriented data model? // Technical Report No. CSIE-89-002, April 23, 1989, published in Proceedings IFIP 11th World Computer Conference, San Francisco, California, August-September, 1989, 18 p.

446. Mata-Toledo R.A. Cushman P.K. Fundamentals of relational databases / McGraw-Hill, 2000, 249 p.

447. McGuinness D.L., Wright J. Conceptual Modeling for Configuration: A Description Logic-based Approach // Artificial Intelligence for Engineering Design, Analysis, and Manufacturing - special issue on Configuration, 1998, V. 12, № 4, p. 333-344.

448. McKenzie L.E., Snodgrass R.T. Schema evolution and the relational algebra // Information Systems — 1990. — V. 15. — № 2. — P. 207-232.

449. Mendelzon A., Mihaila G., Milo T. Querying the WWW // International Journal on Digital Libraries. 1997. — V. 1. — № 1. — P. 54-67.

450. Miller L., Nila S., Data Warehouse Modeler: A CASE Tool for Warehouse Design // Thirty-First Annual Hawaii International Conference on System Sciences. - 1998. - V. 6. – P. 42-48.

451. Morales-Luna G. Simple epistemic logic for relational database // Lecture Notes in Artificial Intelligence. MICAI. Springer, – 2002. – LNAI. 2313. - P. 234-241.

452. Mullins C.S. Denormalization Guidelines // The data administration newsletter, June 1, 1997, p. 34-41

453. Mullins C.S. Database Administration: The Complete Guide to Practices and Procedures / Addison-Wesley, 2002, 736 p.

454. Naiburg E.J., Maksimchuk R.A. UML for database design / Addison-Wesley, 2001, 320 p.

455. Murakami N. Video switcher and video switching method / Patent US 2009/0109334 A1, Apr. 30.2009.

456. Nicola A.De., Missikoff M., Navigli R. A Software Engineering Approach to Ontology Building // Information Systems. Elsevier. – 2009. – Vol. 34. – № 2. – P. 258–275.

457. Noy N.F. Semantic integration: a survey of ontology-based approaches / SIGMOD Record, Vol. 33, № 4, 2004, p. 65-70

458. Noy N.F., Chugh, A., Liu, W., Musen, M.A. A Framework for Ontology Evolution in Collaborative Environments // International Semantic Web Conference (ISWC), 2006, p. 544-558

459. Noy N.F., Klein M. Ontology evolution: Not the same as schema evolution // Knowledge and Information Systems, Vol. 6, № 4, 2004, p. 428-440

460. Noy N.F., McGuinness D.L. Ontology Development 101: A Guide to Creating Your First Ontology / Stanford Knowledge Systems Laboratory Technical Report KSL-01-5 and Stanford Medical Informatics Technical Report SMI-2001-0880, March, 2001, 25 p.

461. Olive A. Conceptual modeling of information systems / Springer, 2007, 455 p.

462. Oppel A.J. Databases: a beginner's guide / McGraw-Hill, 2009, 478 p.

463. Olson J., Data Quality Accuracy Dimension / Morgan Kauffmann Publishers, 2003. - 293 p.

464. Paepcke A., Chang C., Garcia-Molina H., Winograd T. Interoperability for Digital Libraries Worldwide // Communications of the ACM. 1998. № 4, P. 33-43

465. Paradaens J., Janssens D. Decomposition of Relations – a Comprehensive Approach // Advances in Database Theory, 1979, p.73-100

466. Paradaens J. The interaction of integrity Constraints in an Information System // J. Com. And Syst. Sci., 1980, Vol. 20, № 3, p. 310-329

467. Paulsell K. Sybase SQL Server Performance and Tuning Guide: Database Design and Denormalizing for Performance, Chapter 2. // Sybase Report № 32645-01-1100-03, Sybase, 1996, p. 2.1 – 2.18

468. Pavlo A., Paulson E., Rasin A. and as. A Comparison of Approaches to Large-Scale Data Analysis // Proceedings of the 35th SIGMOD International Conference on Management of Data, 2009, p. 165-178.

469. Pedersen T.B., Jensen C.S., Multidimensional data modeling for complex data // Proc. of 15th Inter. Conf. on Data Engineering, IEEE Computer Society. – 1999, - P. 336-345

470. Pedersen T.B., Jensen C.S., Dyreson C.E. A foundation for capturing and querying complex multidimensional data // Inf. Syst. - 2001. - V. 26. - № 5. - P. 383-423.

471. Penney D.J., Stein J. Class Modification in the GemStone object-oriented DBMS // in Proc. of the Int'l Conf. on Object-Oriented Programming Systems, Languages and Applications (OOP-SLA), Orlando, FL – 1987 - P. 111-117.
472. Peters R.J., Ozsu M.T. An Axiomatic Model of Dynamic Schema Evolution in Objectbase Systems // ACM Transactions on Database Systems – 1997 – V. 22, № 1, — P. 75-114.
473. Piattini M., Diaz O., Advanced database technology and design / Artech House, 2000, 535 p.
474. Ponniah P. Data modeling fundamentals: a practical guide for IT professionals / John Wiley & Sons, 2007, 436 p.
475. Poole J., Chang D., Tolbert D., Mellor D. Common warehouse metamodel: an introduction to the standard for data warehouse integration / John Wiley & Sons, 2002, 269 p.
476. Rafanelli M. Multidimensional databases: problems and solutions / Idea Group, 2003, 443 p.
477. Ramakrishnan R., Gehrke J. Database management systems / McGraw-Hill, 2003, 1065 p.
478. Raymond D., Tompa F., Wood D. From data representation to data models // Computer standards and interfaces. 1996. № 1. P. 25-36.
479. Rivero L.C., Doorn J.H., Ferraggine V.E. Encyclopedia of database technologies and applications / Idea Group, 2006, 744 p.
480. Rob P., Coronel C. Database systems: design, implementation, and management, Thomson Course Technology, 2004, 795 p.
481. Roddik J.F. Schema Evolution in Database Systems. An Updated Bibliography // SIGMOD Rec. — Revised, May 1994. —V. 21, № 4, — P. 35-40.
482. Roddik J.F. A Survey of Schema Versioning Issues for Database Systems. // Information and Software Technology. — 1995. —V. 37, № 7, — P. 383-393.
483. Rumbaugh J., Jacobson I., Booch G. The Unified Modeling Language Reference Manual / Addison-Wesley, 1999. - 550 p.

484. Sapia C., Blaschka M., Heolling G., Dinter B., Extending the E/R model for the multidimensional paradigm // Proc. ER Workshop on Data Warehousing and Data Mining. - 1998. - P. 105-116

485. SAS Institute, Base SAS 9.2 Procedures Guide // Third Edition, SAS Publishing, 2010, 488 p.

486. Shapiro S.C., Cables, paths and “subconscious” reasoning in propositional semantic networks // Principles of semantic networks, Morgan Kaufman, San Mateo, CA, 1991, P. 137-156

487. Shapiro S.C., Rapapart W.J., SNePS considered as a fully intensional propositional semantic networks // The Knowledge Frontier: Essays in the Representation of Knowledge, Springer-Verlag, 1987, P. 262-315

488. Shasha D., Bonnett P. Database tuning: principles, experiments, and troubleshooting techniques / Morgan Kaufmann, 2003, 416 p.

489. Sheth A.P., Larson J.A., Federated database systems for managing distributed, heterogeneous and autonomous databases. // ACM Computing Surveys, V. 22, № 3, 1990, p. 183-236

490. Siau K. Advanced topics in database research / Idea Group, vol. 5, 2006, 460 p.

491. Silberschatz A, Korth H.F. Sudarshan S. Database system concepts // PRC edition, McGraw-Hill, - 2002, 1064 p.

492. Simsion G.C., Witt G.C. Data modeling essentials / Morgan Kaufmann Publishers, - 2005, 532 p.

493. Silverston L., Inmon W.H., Graziano K. The Data Model / Resource Book, - 1997, - 400 p.

494. Sjøberg Dag. Quantifying schema evolution Information and Software Technology — 1993. — V. 35, № 1, — P. 35-44.

495. Smith J., Smith D. Database Abstractions: Aggregation and Generalization // ACM Transactions on Database Systems, Vol. 2, № 2, June 1977, p. 105-133

496. Sowa J.F., Conceptual graphs for a database interface // IBM. J. Res and Develop, - 1976 – V. 20, № 4, – P. 336-357

497. Soylu A., Causmaecker P.De, Merging model driven and ontology driven system development approaches pervasive computing perspective // The 24th International Symposium on Computer and Information Sciences, ISCIS 2009, 14-16 September 2009. - North Cyprus. IEEE, 2009. - P. 730-735

498. Spaccapietra S. Journal on data semantics / Springer, Vol. 12, 2009, 179 p.

499. Sumathi S., Esakkirajan S. Fundamentals of relational database management systems / Springer, 2007, 776 p.

500. Stonebraker M. Madden S., Abadi D.J. et al, The End of an Architectural Era (It's Time for a Complete Rewrite) // Proceedings of VLDB, Vienna, September 23-28, 2007, p. 1150-1160

501. Stonebraker M. The NoSQL Discussion has Nothing to Do With SQL // BLOG@CACM, - № 4 (November), - 2009

http://cacm.acm.org/blogs/blog-cacm/50678-the-nosql-discussion-has-nothing-to-do-with-sql/fulltext

502. Stoy J.E. Denotational semantics: the Scott-Strachey Approach to Programming Language Theory / MIT Press. 1977 – 414 p.

503. Tarski A. Logic, Semantics, Metamathematics / Oxford, 1956, 471 p.

504. Teorey T. et al, Database design: know it all / Morgan Kaufmann, 2009, 349 p

505. Theodoratos D., Bouzeghoub M., A general framework for the view selection problem for data warehouse design and evolution // Proc. of the 3rd ACM Inter. Workshop on Data Warehousing and OLAP (DOLAP 2000), - 2000. - P. 1-8

506. Thompson S. Type Theory and Functional Programming / Addison-Wesley, 1991, 387 p.

507. Thomsen E. OLAP solutions: building multidimensional information systems / John Wiley & Sons, 2002, 661 p.

508. Tryfona N., Busborg F., Christiansen J., StarER: A conceptual model for data warehouse design // Proc. of the ACM 2nd Intl. Workshop on Data Warehousing and OLAP (DOLAP 1999), - 1999. - P. 3-8

509. Van H.D., George C., Janowski T., Moore R. Specification Case Studies in RAISE (FACIT) / Springer, 2002. - 405 p.

510. Vrbsky S.V., Tomic S. Satisfying temporal consistency constraints of real-time databases // J. Syst. and Software, V. 45, № 1,- 1999. - p. 45-60.

511. Wang R.Y., Ziad M., Lee Y.W. Data quality / Kluwer Academic Publishers, 2002, 167 p.

512. Wegner P. Vienna definition language // ACM Computer Surveys, 1972. - Vol. 4., № 1, - P. 5-63.

513. Wells A.J. Grid Database Design / Taylor & Francis Group, 2005, 288 p.

514. Wijngaarden V.A., Mailloux B.J., Peck J.E., Koster C.H. Report on the algorithmic language Algol-68 / Amsterdam: Mathematisch Centrum, 1969. 265 p.

515. Wrembel R., Koncilia C. Data warehouses and OLAP: concepts, architectures, and solutions / IRM Press, 2007, 332 p.

516. Yourdon E. Modern Structured Analysis / Yourdon Press, Prentice-Hall, 1989. - 672 p.

517. Ziegler P., Dittrich K.P. Three Decades of Data Integration - All Problems Solved? // IFIP Congress Topical Sessions, 18th World Computer Congress (WCC), Toulouse, 22-27 August 2004, p. 3–12

Приложение 1.

Схема сущностей-объектов ПрО «Горгаз»

ГОСУДАРСТВА (*КодГос*, Наименов) – атомарная сущность-объект, *АДМИНИСТРАТИВНЫЕ ОБЛАСТИ* (*КодГос*, *КодОбл*, Наименов) – слабая сущность-объект, *АДМИНИСТРАТИВНЫЕ РАЙОНЫ* (*КодГос*, *КодОбл*, *КодРай*, Наименов) – слабая сущность-объект, *НАИМЕНОВАНИЯ НАСЕЛЕННЫХ ПУНКТОВ* (*КодНаименовНП*, *Литера*, Наименов) – атомарная сущность-объект, *НАСЕЛЕННЫЕ ПУНКТЫ В ОБЛАСТЯХ-РАЙОНАХ* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *Литера*, *КодНаименовНП*, Примечание) – составная сущность-объект, *КАТЕГОРИИ НАСЕЛЕННЫХ ПУНКТОВ* (*КодТипаНП*, Наименов) – атомарная сущность-объект, *РАЙОНЫ В НАСЕЛЕННЫХ ПУНКТАХ* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодРайГор*, Наименов) – слабая сущность-объект, *СПИСОК ФАМИЛИЙ КАК СЛОВ* (*КодФам*, *ЛитераФам*, Фамилия) – атомарная сущность-объект, *СПИСОК ИМЕН КАК СЛОВ* (*КодИмени*, *ЛитераИм*, Имя) - – атомарная сущность-объект, *СПИСОК ОТЧЕСТВ КАК СЛОВ* (*КодОтчества*, *ЛитераОтч*, Отчество) – атомарная сущность-объект, *СПИСОК АБОНЕНТОВ* (*КодАбонента*, *ЛитераФам*, *КодФам*, *ЛитераИм*, *КодИмени*, *ЛитераОтч*, *КодОтч*, ДатаРождения, Примечание) – атомарная сущность-объект, *КАТЕГОРИИ АБОНЕНТОВ* (*КодКатегАбонен*, Наименов) – атомарная сущность-объект, *АБОНЕНТЫ ПО КАТЕГОРИЯМ* (*КодАбонента*, *КодКатегАбонен*, Примечание) – составная сущность-объект, *НАИМЕНОВАНИЯ АДРЕСНЫХ КАТЕГОРИЙ («УЛИЦ») КАК СЛОВ* (*КодНаимАдрКатег*, *Литера*, Наименов) – атомарная сущность-объект, *ТИПЫ АДРЕСНЫХ КАТЕГОРИЙ* (*КодТипаАдрКатегор*, Наименов) – атомарная сущность-объект, *СПИСОК ВАРИАНТОВ АДРЕСНЫХ КАТЕГОРИЙ* (*КодАдрКатег*, *КодТипаАдрКатегор*, *Литера*, *КодНаимАдрКатег*, Примечание) – слабая сущность-объект, *СПИСОК АДРЕСНЫХ КАТЕГОРИЙ В НАСЕЛЕННЫХ ПУНКТАХ* (*КодГос*, *КодОбл*,

КодРай, *КодТипаНП*, *КодНаселПункта*, *КодТипаАдрКатегор*, *КодАдрКатег*, Примечание) – слабая сущность-объект, *РЕЕСТР ЗДАНИЙ И СООРУЖЕНИЙ ПО НАСЕЛЕННЫМ ПУНКТАМ НА «УЛИЦАХ»* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодТипаАдрКатегор*, *КодАдрКатег*, *НомерЗдания*, *ЛитераЗдания*, *НомерКорпуса*, *ЛитраКорпуса*, НаименовЗдания) – слабая сущность-объект, *АДРЕСА АБОНЕНТОВ* (*КодАбонента*, *КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодТипаАдрКатегор*, *КодАдрКатег*, *НомерЗдания*, *ЛитераЗдания*, *НомерКорпуса*, *ЛитраКорпуса*, НомерКвартиры, ЛитераКвартиры, Примечание) – слабая сущность-объект, *СПИСОК СОТРУДНИКОВ* (*КодСотр-ТабНом*, *КодКатегСотр*, *КодФам*, *Литера*, Примечан) – атомарная сущность-объект, *БРИГАДЫ СОТРУДНИКОВ* (*КодБриг*, *КодКатегСотр, КодСотр-ТабНом,* Наименов) – слабая сущность-объект, *УЧАСТКИ ОБХОДА* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодТипаАдрКатегор*, *КодАдрКатег*, *КодУчастка*, Наименов) – атомарная сущность-объект, *РЕЕСТР ЗДАНИЙ И СООРУЖЕНИЙ ПО УЧАСТКАМ В НАСЕЛЕННЫХ ПУНКТАХ* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодТипаАдрКатегор*, *КодАдрКатег*, *НомерЗдания*, *ЛитераЗдания*, *НомерКорпуса*, *ЛитраКорпуса*, *КодУчастка, КодЗданНаУчастке,* Примечан) – слабая сущность-объект, *УЧАСТКИ ОБХОДА КОНТРОЛЕРОВ* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодТипаАдрКатегор*, *КодАдрКатег*, *КодУчастка*, *КодКатегСотр*, *КодСотр*, Примечан) – слабая сущность-объект, *КАТЕГОРИИ УСЛУГ* (*КодКатегУслуг*, *КодНаличОтопл*, Наименов) – атомарная сущность-объект, *НАЛИЧИЕ ОТОПЛЕНИЯ* (*КодНаличОтопл,* Наименов) – атомарная сущность-объект, *СПОСОБЫ ОТПУСКА АБОНЕНТОВ* (*КодСпособОтпуска*, Наименов) – атомарная сущность-объект, *СПОСОБЫ РАСЧЕТА АБОНЕНТОВ* (*КодСпособРасчета*, Наименов) – атомарная сущность-объект, *СПОСОБЫ ОТКРЫТИЯ ЛИЦЕВОГО СЧЕТА* (*КодСпособОткрытЛС*, Наименов) – атомарная сущность-объект,

СПРАВОЧНИК ЖЕКОВ (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодУчастка*, *КодЖЕКа*, Ниименов) – слабая сущность-объект, *СПРАВОЧНИК ГРП* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодУчастка*, *КодГРП*, Ниименов) – слабая сущность-объект, *СПРАВОЧНИК ГРС* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодУчастка*, *КодГРС*, Ниименов) – слабая сущность-объект, *ЦЕНОВЫЕ ГРУППЫ ПО ПН* (*КодКатегорПН*, Наименов) – атомарная сущность-объект, *ЦЕНОВЫЕ ГРУППЫ ПО ПК* (*КодКатегорПК*, Наименов) – атомарная сущность-объект, *ЖИЛИЩНЫЕ СЕКТОРЫ* (*КодГос*, *КодОбл*, *КодРай*, *КодТипаНП*, *КодНаселПункта*, *КодУчастка*, *КодЖилСектора*, Наименовм) – слабая сущность-объект, *ТИП КВАРТИР АБОНЕНТОВ* (*КодТипаКвартир*, Наименов) – атомарная сущность-объект, *ОТЧЕТНЫЙ ПЕРИОД* (*Год*, *Месяц*, *КодОтчПер*, Наименов) – слабая сущность-объект, *ТИПЫ СЕМЕЙ АБОНЕНТОВ* (*КодТипаСемьи*, Наименов) – атомарная сущность-объект, *РЕЕСТР АБОНЕНТОВ-ПОКУПАТЕЛЕЙ УСЛУГ* (*КодАбонен*, *КотОтчетнПериода*, *КодГРП, КодГРС*, *КодТипаКвартир*, *КодЖилСектора, КодКатегУслуг*, *КодНаличОтопл*, *КодСпособОтпуска*, *КодСпособОткрытЛС*, *КодСпособРасчета*, *КодКатегорПН*, *КодКатегорПК*, *КодТипаСемьи,* ПоказанияЗаОтчПериод, Примечание) – составная сущность-объект.

Приложение 2.

Схема сущностей-объектов ПрО
«Учет отгрузки угля на шахте»

ПОКАЗАТЕЛИ КАЧЕСТВА ПАРТИИ (*КодПоказателя,* Наименование) – атомарная сущность-объект;

ПОКАЗАТЕЛИ ПАРТИИ СКИДКИ-НАДБАВКИ (*КодПоказателя, КодСкидкиНадбавки, КодДиапазонПоказателя,* СуммаСкидкиНадбавки) – составная сущность-объект;

ПОТРЕБИТЕЛИ (*КодПотребителя*, Наименование) – атомарная сущность-объект;

АДРЕСА ПОТРЕБИТЕЛЕЙ (*КодГос, КодОбл, КодГор, КодАдреснКатегор, НомерЗдания, НомерОфиса, КодПокупателя,* Примечание) – составная сущность-объект;

ТЕЛЕФОНЫ ПОТРЕБИТЕЛЕЙ (*КодПотребителя, КодТипаТЕлефона, НомерТелефона*) – слабая сущность-объект;

КОНТАКТЫ ПОТРЕБИТЕЛЕЙ (*КодПотребителя, КодТипаКониакта, КодКонтакта,* Контакт) – составная сущность-объект;

ПОТРЕБИТЕЛИ КАК ПЛАТЕЛЬЩИКИ (*КодПотребителя, КодКатегорииПлательщика,* Примечание) – составная сущность-объект;

ПАРТИЯ ПРОДУКТА (*КодКатегПродукта, НомГода, КодМесяца, КодШахты, КодУчастка, КодПартии,* Объем партии, Примечание) – составная сущность-объект;

ЦЕНА ПРОДУКТА (*КодКатегПродукта, НомГода, КодМесяца, ЦенаЕдиницыПродукта,* Примечание) – составная сущность-объект;

ПАРТИЯ ПРОДУКТА ПОТРЕБИТЕЛЮ (*КодКатегПродукта, НомГода, КодМесяца, КодШахты, КодУчастка, КодПартии, КодПотребителя,* Объем продукта, Дата, Примечание) – составная сущность-объект;

УДОСТОВЕРЕНИЕ ПАРТИИ (*КодКатегПродукта, НомГода, КодМесяца, КодШахты, КодУчастка, КодПартии, НомерУдостовер, КодЗольности,*

КодВлажности, КодТеплотыСгоран, КодПроцСеры, Дата, Примечание) – артефакт, заменяется одноименной составной сущностью-объектом;
ТРАНСПОРТНАЯ НАКЛАДНАЯ ПРТИИ ПОТРЕБИТЕЛЮ (*КодКатегПродукта, НомГода, КодМесяца, КодПартии, КодПотребителя, НомВагона*, Объем партии, Дата, Примечание) – артефакт, заменяется одноименной составной сущностью-объектом;
ЦЕНОВАЯ КОМПЕНСАЦИЯ-НАДБАВКА ПОЛУЧАТЕЛЮ ПАРТИИ (*КодПоказателя, КодСкидкиНадбавки*, *КодДиапазонПоказателя, КодКатегПродукта, НомГода, КодМесяца, КодШахты, КодУчастка, КодПартии, КодПотребителя,* Сумма скидки-надбавки, Примечание) – составная сущность-объект;
СЧЕТ К ОПЛАТЕ ПОТРЕБИТЕЛЮ (*КодКатегПродукта, НомГода, КодМесяца, КодШахты, КодУчастка, КодПартии, КодПоказателя, КодСкидкиНадбавки, КодДиапазонПоказателя, КодПотребителя,* Объем продукта, Цена объема продукта, СуммаСкидкиНадбавки, Итого к оплате, Дата выписки, Номер счета) – артефакт, заменяется одноименной составной сущностью-объектом;
ТЕРРИТОРИЯ (*КодОбъединения, КодТерритории, КодОбл*, Наименование, Примечание) – слабая сущность-объект;
ШАХТА (*КодОбъединения, КодТерритории, КодШахты,* Наименование, Примечание) – слабая сущность-объект;
УЧАСТОК ШАХТЫ (*КодОбъединения, КодТерритории, КодШахты, КодУчастка*, Наименование, Примечание) – слабая сущность-объект;
СМЕНА ШАХТЫ (*КодОбъединения, КодТерритории, КодШахты, КодУчастка, НомерСмены,* Наименование, Примечание) – слабая сущность-объект;
БРИГАДА ШАХТЫ (*КодОбъединения, КодТерритории, КодШахты, КодУчастка, КодБригады,* Наименование, Примечание) – слабая сущность-объект;

СМЕНА БРИГАДЫ (*КодОбъединения, КодТерритории, КодШахты, КодУчастка, КодБригады, НомерСмены,* Дата, Примечание) – составная сущность-объект;

ГОД (*НомГода,* Примечание) – атомарная сущность-объект;

МЕСЯЦ (*НомГода, НомМесяца,* Наименование, Примечание) – слабая сущность-объект;

ДОБЫЧА ШАХТЫ ДЕНЬ (*НомГода, НомМесяца, НомДня, КодОбъединения, КодТерритории, КодШахты,* Объем, Примечание) – составная сущность-объект;

ДОБЫЧА УЧАСТКА ШАХТЫ ДЕНЬ (*НомГода, НомМесяца, НомДня, КодОбъединения, КодТерритории, КодШахты, КодУчастка,* Объем, Примечание) – составная сущность-объект;

ДОБЫЧА БРИГАДЫ ШАХТЫ ДЕНЬ (*НомГода, НомМесяца, НомДня, КодОбъединения, КодТерритории, КодШахты, КодУчастка, НомБригады,* Объем, Примечание) – составная сущность-объект.

Приложение 3.

Схема сущностей-объектов ПрО
«Городская поликлиника»

1. **Реестры физических лиц по городам государств** (территориально-распределенная система справочников)

ГОСУДАРСТВА (*КодГосуд*, Наименов, СокрНаимен, …) – слабая сущность-объект.

СЛОВАРЬ НАИМЕНОВАНИЙ ОБЛАСТЕЙ (*КодНаименОбл*, Наименов, СокрНаимен, …) – атомарная сущность-объект;

СЛОВАРЬ НАИМЕНОВАНИЙ ГОРОДОВ (*КодНаименГор,* Наименов, СокрНаимен, …) – атомарная сущность-объект;

СЛОВАРЬ НАИМЕНОВАНИЙ РАЙОНОВ (*КодНаименРай,* Наименов, СокрНаимен, …) – атомарная сущность-объект;

ОБЛАСТИ В ГОСУДАРСТВАХ (*КодГосуд, КодОбл, КодНаименОбл*, Примечание, …) – слабая сущность-объект;

РАЙОНЫ В ОБЛАСТЯХ В ГОСУДАРСТВЕ (*КодГосуд, КодОбл, КодРайона, КодНаименРай,* Примечание) – слабая сущность-объект;

ГОРОДА В ГОСУДАРСТВАХ (*КодГосуд, КодОбл, КодРайона, КодГорода, КодНаименГор,* Примечание) – слабая сущность-объект;

МИКРОРАЙОНЫ ГОРОДОВ (*КодГосуд, КодОбл, КодРайона, КодГорода, КодМикроР*, *КодНаимМикроР*, Примечание) – слабая сущность-объект;

ТИПЫ АДРЕСНЫХ КАТЕГОРИЙ (*КодТипаАдрКатег*, Наименов) – атомарная сущность-объект;

СЛОВАРЬ НАИМЕНОВАНИЙ АДРЕСНЫХ КАТЕГОРИЙ (*КодНаименАдрКатег*, *КодТипаАдрКатег,* Наименов) – слабая сущность-объект;

АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ (*КодГосуд, КодОбл, КодРайона, КодГорода, КодТипаАдрКатег*, *КодАктуальнНаимен, КодНаименАдрКатег,* Примечание) – составная сущность-объект – «справочник улиц городов» с учетом истории изменений собственных наименований;

СЛОВАРЬ ФАМИЛИЙ (*КодФам*, Фамилия) – атомарная сущность-объект;

СЛОВАРЬ ИМЕН (*КодИмени*, Имя) – атомарная сущность-объект;

СЛОВАРЬ ОТЧЕСТВ (*КодОтч*, Отчество) – атомарная сущность-объект;

ФИЗИЧЕСКИЕ ЛИЦА (*ИдентНомерФизЛица*, *КодГос, КодФам*, *КодИмени*, *КодОтч*, *ДатаРожд, КодГородаРожден*, Пол, …, Примечание) – атомарная сущность-объект – справочник «люди в государстве»;

АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГос, КодОбл, КодРайона, КодГорода, КодТипаАдрКатег*, *КодНаименАдрКатег, НомЗдания, Литера, СубНомЗдания, СубЛитера, НомПомещен, Литера*) – слабая сущность-объект;

ПЕРСОНАЛЬНЫЕ ТЕЛЕФОНЫ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГос, НомТелефона, КодТипаТелеф,* Примечание) – слабая сущность-объект;

ПЕРСОНАЛЬНЫЕ ЭЛЕКТРОННЫЕ АДРЕСА ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГос, НомерЭлектрАдр,* ИмяЭлектрПочты) – слабая сущность-объект;

ПЕРСОНАЛЬНЫЕ БЛОГИ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГос, НомерБлога,* Имя блога) – слабая сущность-объект;

КРАТКИЙ СПРАВИЧНИК ПРОФЕСИЙ (*КодПроф*, Наименов) – атомарная сущность-объект;

СПРАВИЧНИК СПЕЦИАЛЬНОСТЕЙ (*КодПроф*, *КодСпец,* Наименов) – слабая сущность-объект;

СПРАВИЧНИК СПЕЦИАЛИЗАЦИЙ В СПЕЦИАЛЬНОСТЯХ (*КодПроф*, *КодСпец, КодСпециализ*, Наименов) – слабая сущность-объект;

СПОСОБЫ ПРИОБРЕТЕНИЯ СПАЦИАЛЬНОСТИ (*КодТипаПриобретения,* Наименов) – атомарная сущность-объект;

СПЕЦИАЛЬНОСТИ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодПроф*, *КодСпец, КодТипаПриобретения,* Примечан) – слабая сущность-объект;

ОФИЦИАЛЬНЫЕ СПЕЦИАЛЬНОСТИ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодПроф*, *КодСпец*, НомДокумента, КодГородаВыдачи, ДатаВыдачи) – слабая сущность-объект;

ПРОФЕССИИ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодПроф*, Примечан) – слабая сущность-объект;

СПЕЦИАЛИЗАЦИИ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодПроф*, *КодСпец, КодСпециализ*, Примечан) – слабая сущность-объект;

ТИПЫ СОБСТВЕННСОТИ (*КодТипаСобств,* Наименов) – атомарная сущность-объект;

ОРГАНИЗАЦИОННО-ПРАВОВЫЕ ФОРМЫ (*КодТипаСобств, КодОргПравФормы,* Наименов) – слабая сущность-объект;

НАИМЕНОВАНИЕ ДОЛЖНОСТЕЙ (*КодДолжн*, Наименов) – атомарная сущность-объект;

ОТРАСЛИ (*КодОтр*, *КодПодотр,* Наименов) – слабая сущность-объект;

РЕКОМЕНДОВАНЫЕ ДОЛЖНОСТИ В ОТРАСЛЯХ (*КодОтр*, *КодПодотр, КодОргПравФормы, КодДолжн*, Примечан) – слабая сущность-объект;

ОРГАНИЗАЦИИ ГОРОДА (*КодГос, КодОтр*, *КодПодотр, КодПредпр*, *КодТипаСобств, КодОргПравФормы,* Наименов) – слабая сущность-объект;

РАБОЧИЕ МЕСТА ФИЗЛИЦА (*КодГос, ИдентНомерФизЛица*, *КодПредпр*, *НомГодаЗачисл, НомМесЗачисл, ДеньЗачисл, КодОтр*, *КодПодотр, КодДолжн,* Примечан) – составная сущность-объект;

2. **Штатный реестр в городской поликлинике в одном из городов**

УЧАСТКИ В ГОРОДЕ (*КодУчастка, КодГосуд, КодОбл, КодРайона, КодГорода,* Наименование) – слабая сущность-объект;

ЗДАНИЯ И ПОМЕЩЕНИЯ НА УЧАСТКАХ В ГОРОДЕ (*КодУчастка, КодГосуд, КодОбл, КодРайона, КодГорода, КодТипаАдрКатег*, *КодНаименАдрКатег, НомЗдания, НомПомещения*) – слабая сущность-объект;

СОТРУДНИК ПОЛИКЛИНИКИ (*ТабНомСотрПоликл, КодТипаЗанятости,* Примечание, *ИдентНомерФизЛица, КодГос*) - маска на отношение ФИЗИЧЕСКОЕ ЛИЦО – фильтр по принадлежности к данной поликлинике (в штате, по совместительству или по договору), тут *ИдентНомерФизЛица,*

КодГос – вторичные ключи, фильтрующий отношение, далее аналогичный прием;

СПИСОК ТИПОВ ЗАНЯТОСТИ (*КодТипаЗанятости,* Наименование) – атомарная сущность-объект;

СПИСОК ДОЛЖНОСТЕЙ В ПОЛИКЛИНИКЕ (*КодДолжнПоликл,* Наименование) – маска на отношение *ОБЩЕГОСУДАРСТВЕННЫЙ РЕЕСТР ДОЛЖНОСТЕЙ* – как атомарная сущность-объект;

ШТАТНОЕ РАСПИСАНИЕ ПОЛИКЛИНИКИ (*КодДолжнПоликл, НомГода, Вакантность, НомМес,* Примечание) – составная сущность-объект - текущее состояние списка должностей в поликлинике;

ДОЛЖНОСТЬ СОТРУДНИКА ПОЛИКЛИНИКИ (*ТабНомСотрПоликл, КодДолжнПоликл, НомГодаЗачисл, НомМесЗаичл, НомДняЗаичсл,* Примечание) – составная сущность-объект - текущее состояние должностей сотрудников в поликлинике;

ВРАЧ (*КодВрача,* Примечание, *ТабНомСотрПоликл*) – маска на отношение-связь *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на отношение *ФИЗИЧЕСКИЕ ЛИЦА*) – фильтр по признаку «*врач»*;

СПСОК КАБИНЕТОВ ПРИЕМА ВРАЧЕЙ В ПОЛИКЛИНИКЕ (*КодВрача, НомКорпуса, НомКабин,* Примечание) – слабая сущность-объект;

УЧАСТКОВЫЙ ВРАЧ (*КодВрача,* Примечание, *ТабНомСотрПоликл,*) – маска на отношение *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на маску ВРАЧ) - фильтр по признаку «*тип работы»*;

ПРИНИМАЮЩИЙ ВРАЧ (*КодВрача,* Примечание, *ТабНомСотрПоликл,*) – маска на отношение *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на маску ВРАЧ) – фильтр по признаку «*тип работы»*;

ЛЕЧАЩИЙ ВРАЧ (*КодВрача,* Примечание, *ТабНомСотрПоликл,*) – маска на отношение *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на маску ВРАЧ) – фильтр по признаку «*тип работы»*;

ТИПЫ РАБОТЫ ВРАЧЕЙ (*КодТипаРаботы*, Наименов, …) – атомарная сущность-объект (возможно со ссылкой на метаданные);

СПАЦИАЛИЗАЦИЯ ВРАЧА (*КодСпециалВрача,* Наименов,) – атомарная сущность-объект (возможно со ссылкой на метаданные);

СПАЦИАЛИЗАЦИИ ВРАЧЕЙ В ПОЛИКЛИНИКЕ (*КодВрача, КодСпециалВрача,* Примечание, `КодТаблВрачей`) – составная сущность-объект, тут `КодТаблВрачей` – ссылка на метаданные;

СПРАВОЧНИК ДИАГНОЗОВ (*КодСпециалВрача, КодДиагноза,* Наименов) – слабая сущность-объект;

СПРАВОЧНИК ИССЛЕДОВАНИЙ (*КодСпециалВрача, КодИсследов*, Наименов) – слабая сущность-объект;

СПРАВОЧНИК ИССЛЕДОВАНИЙ ПО ДИАГНОЗАМ (*КодСпециалВрача, КодДиагноза, КодИсследов,* Наименов) – слабая сущность-объект;

ПРЕЙСКУРАНТ УСЛУГ (*КодТипаВизита, КодВрача, КодСпециалВрача, КодУслуги, НомГода, НомМес,* Наименов, Цена,…) – слабая сущность-объект;

3. **Прием в городской поликлинике в одном из городов**

ПАЦИЕНТ (*КодПациента,* Примечание, *ИдентНомерФизЛица, КодГос*) – маска на отношение *ФИЗИЧЕСКОЕ ЛИЦО* – фильтр по потребности «*посещать поликлинику*»;

ЗАПИСЬ К СПЕЦИАЛИСТУ (*КодПациента, КодТипаВизита, КодВрача, КодСпециалВрача, КодУслуги, НомГода, НомМес, НомДня, НомКорпуса, НомКабин, НомОчереди,* Время, Примечание, СуммаКОплате, ...) - составная сущность-объект, связь маски и сущности-объекта – запись или к врачу на прием, или к врачу на исследование, или к медперсоналу на процедуры, или к медперсоналу на манипуляции, и т.д.;

ЗАКАЗ СПЕЦИАЛИСТА НА ДОМ (*КодУчастка, КодПациента, КодВрача, КодСпециалВрача, КодУслуги, НомГода, НомМес, НомДня, НомОчереди,* Время, Примечание, СуммаКОплате, …) - составная сущность-объект, связь маски и сущности-объекта – заказ на дом или врача, или лаборанта, или медсестры для процедуры, или медсестры для манипуляции, и т.д.;

ВИЗИТ К ВРАЧУ СТОМАТОЛОГУ (*КодПациента, КодТипаВизита, КодВрача, НомГода, НомМес, НомДня, НомОчереди, КодПредвДиагноза,* Время, Примечание, СуммаКОплате,…) – составная сущность-объект;

НАЗНАЧЕНИЯ ИССЛЕДОВАНИЙ СТОМАТОЛОГОМ (*КодПациента, КодВрача, НомГода, НомМес, НомДня, КодПредвДиагноза, КодИсследов,* Примечание, СуммаКОплате,…) – составная сущность-объект;

РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЙ ДЛЯ СТОМАТОЛОГА (*КодПациента, КодВрача, НомГода, НомМес, НомДня, КодПредвДиагноза, КодИсследов, НомЗуба, КодОкончДиагноза,* Примечание) – составная сущность-объект;

НАЗНАЧЕНИЯ КУРСА ЛЕЧЕНИЯ СТОМАТОЛОГОМ (*КодПациента, КодВрача, НомГода, НомМес, НомДня, КодОкончДиагноза, КодПроцедуры,* Примечание, СуммаКОплате,…) – составная сущность-объект;

РЕЦЕПТ СТОМАТОЛОГА (*КодПациента, КодВрача, НомГода, НомМес, НомДня, КодОкончДиагноза, КодЛекарства,* Примечание) – составная сущность-объект;

ЗАПИСИ В ЛИЧНОЕ ДЕЛО ПАЦИЕНТА (*КодУчастка, КодПациента, КодВрача, КодСпециалВрача, НомГода, НомМес, НомДня, КодОкончДиагноза, КодРезульт,* Примечание) – составная сущность-объект;

4. **Маски как составные сущности-объекты** (для хранения статистических данных, сформированных фоновыми процедурами по факту ввода данных в оперативные отношения в режиме реального времени – аналог отношений-фактов в схеме ХД «снежинка»).

РАБОТА ВРАЧЕЙ ЗА ДЕНЬ (*КодТипаРаботы*, *КодВрача, НомГода, НомМес, НомДня,* СуммЧислоЧасов, …);

РАБОТА ВРАЧЕЙ ЗА МЕСЯЦ (*КодТипаРаботы*, *КодВрача, НомГода, НомМес,* СуммЧислоЧасов, …);

РАБОТА ВРАЧЕЙ ЗА ГОД (*КодТипаРаботы*, *КодВрача, НомГода,* СуммЧислоЧасов, …); *РАБОТА ЛЕЧАЩИХ ВРАЧЕЙ ЗА ДЕНЬ* (*КодВрача, НомГода, НомМес, НомДня,* СуммЧислоЧасов, …);

РАБОТА ЛЕЧАЩИХ ВРАЧЕЙ ЗА МЕСЯЦ (*КодВрача, НомГода, НомМес,* СуммЧислоЧасов, …);

РАБОТА УЧАСТКОВЫХ ВРАЧЕЙ ЗА ДЕНЬ (*КодВрача, НомГода, НомМес, НомДня,* СуммЧислоЧасов, …);

РАБОТА УЧАСТКОВЫХ ВРАЧЕЙ ЗА МЕСЯЦ (*КодВрача, НомГода, НомМес,* СуммЧислоЧасов, …);

ПРИЕМЫ ВРАЧЕЙ ЗА ДЕНЬ (*КодВрача, НомГода, НомМес, НомДня,* СуммЧислоЧасов, …);

ПРИЕМЫ ВРАЧЕЙ ЗА МЕСЯЦ (*КодВрача, НомГода, НомМес,* СуммЧислоЧасов, ОбщЧислоПац,…);

ПОЛИКЛНИКА ЗА МЕСЯЦ (*НомГода, НомМес,* СуммЧислоЧасов, ОбщЧислоПац, ОбщСуммаРеализации, …);

РЕГИСТРАТУРА ЗА МЕСЯЦ (*НомГода, НомМес,* ОбщЧислоПац, …);

РЕНТГЕНОГРАФИЯ ЗА МЕСЯЦ (*НомГода, НомМес, КодРенгеногр,* ЧислоПац, ОбщСуммаРеализации,…);

ФЛЮОРОГРАФИЯ ЗА МЕСЯЦ (*НомГода, НомМес, КодФлюорогр,* ЧислоПац, ОбщСуммаРеализации,…);

УЗИ ЗА МЕСЯЦ (*НомГода, НомМес, КодУЗИ,* ЧислоПац, ОбщСуммаРеализации,…);

МАНИПУЛЯЦИОННАЯ ЗА МЕСЯЦ (*НомГода, НомМес, КодМанипул,* ЧислоПац, ОбщСуммаРеализации,…);

ДОВРАЧЕБНЫЙ КАБИНЕТ ЗА МЕСЯЦ (*НомГода, НомМес, КодДоврОбслед,* ЧислоПац, ОбщСуммаРеализации,…);

ТЕРАПИЯ ЗА МЕСЯЦ (*НомГода, НомМес,* ЧислоПац, ОбщСуммаРеализации,…);

Отношения, моделирующие итоговые данные за месяц по всем иным специализациям (*СТОМАТОЛОГИЯ*, *ГАСТРОЭНТЕРОЛОГИЯ*, *ПЕДИАТРИЯ*, *КАРДИОЛОГИЯ*, *УРОЛОГИЯ*, *ЭНДОКРИНОЛОГИЯ*, *ПУЛЬМАНОЛОГИЯ*, *ОФТАЛЬМОЛОГИЯ*, *ОТОЛАРИНГОЛОГИЯ*, *ГЕНИКОЛОГИЯ*, *ОРТОПЕДИЯ* и т.п.) будут аналогично сформированы по

схеме: *СПЕЦИАЛИЗАЦИЯ ЗА МЕСЯЦ* (*НомГода, НомМес, КодСпециализирУслуги,* ЧислоПац, ОбщСуммаРеализации,…).

5. **Оперативные и аналитические отчеты за день, месяц, год и более**

«Номерки очереди», «счета к оплате», «рецепты», «направления на исследования», «направления на процедуры», «список разноски личных дел пациентов по кабинетам врачей», «поврачебные списки адресов пациентов для посещения на дому», «наряд на обход», «накладная на получение лекарств на складе», «накладная на получение материалов на складе», «статистические отчеты по отделениям о посещении», «отчеты по расходам материалов и лекарств», «отчет об уровне заболеваемости», «отчет о прогнозах сезонной заболеваемости», «отчет о прогнозах сезонных эпидемий» и т.п.

РОССИЙСКАЯ ФЕДЕРАЦИЯ

РОССИЙСКОЕ АГЕНТСТВО ПО ПРАВОВОЙ
ОХРАНЕ ПРОГРАММ ДЛЯ ЭВМ, БАЗ ДАННЫХ И
ТОПОЛОГИЙ ИНТЕГРАЛЬНЫХ МИКРОСХЕМ
(РосАПО)

СВИДЕТЕЛЬСТВО

ОБ ОФИЦИАЛЬНОЙ РЕГИСТРАЦИИ ПРОГРАММЫ ДЛЯ ЭВМ

№ 940165

ПРОГРАММА ДЛЯ ЭВМ: CASE-генератор прикладных сетевых информационных комплексов - инструментальная система SWS 1.0 (CASE-генератор SWS 1.0)

ПРАВООБЛАДАТЕЛЬ: Гайдабрус Виталий Николаевич
Панченко Борис Евгеньевич
Церковицкий Сергей Леонидович

СТРАНА: Украина

АВТОР (АВТОРЫ): Панченко Б.Е.
Гайдабрус В.Н.
Церковицкий С.Л.

Заявка № 940052

Зарегистрировано в Реестре программ для ЭВМ

Дата регистрации 14 **число** 04 **месяц** 1994 **год**

Генеральный директор РосАПО _____ **М.П.**

Приложение 5.

УКРАЇНА

(19) **(UA)**

(11) **63036**

(51) 7 **G06F17/00,**
G06F17/30

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ ДЕПАРТАМЕНТ ІНТЕЛЕКТУАЛЬНОЇ ВЛАСНОСТІ

ПАТЕНТ
на винахід

видано відповідно до Закону України
"Про охорону прав на винаходи і корисні моделі"

Голова Державного департаменту інтелектуальної власності М. Паладій

(21) 2001128530
(22) 11.12.2001
(24) 15.01.2004
(46) 15.01.2004. Бюл.№ 1

(72) Панченко Борис Євгенійович
(73) Панченко Борис Євгенійович

(54) СПОСІБ РОЗМІЩЕННЯ ДАНИХ У КОМП'ЮТЕРНОМУ СХОВИЩІ ІЗ ЗАБЕЗПЕЧЕННЯМ МОДИФІКАЦІЙНОСТІ ЙОГО СТРУКТУРИ

Приложение 6.

УКРАЇНА UKRAINE

ПАТЕНТ

НА ВИНАХІД

№ 92248

СПОСІБ УЗАГАЛЬНЕНОГО РОЗМІЩЕННЯ ДАНИХ З УРАХУВАННЯМ МОДИФІКАЦІЙНОСТІ СТРУКТУРИ СХОВИЩА

Видано відповідно до Закону України "Про охорону прав на винаходи і корисні моделі".

Зареєстровано в Державному реєстрі патентів України на винаходи **11.10.2010.**

Голова Державного департаменту інтелектуальної власності М.В. Паладій

Міністерство освіти і науки України Державний департамент інтелектуальної власності № 1

ПАТЕНТ

НА ВИНАХІД

№ 99921

СПОСІБ ПОПЕРЕДНЬОЇ КАРКАСНОЇ СЕПАРАЦІЇ ДАНИХ ПЕРЕД ЇХ МОДИФІКАЦІЙНО ЗДАТНИМ РОЗМІЩЕННЯМ У СХОВИЩЕ АБО ПРОЦЕСОМ ПОДАЛЬШОЇ ОБРОБКИ

Видано відповідно до Закону України "Про охорону прав на винаходи і корисні моделі".

Зареєстровано в Державному реєстрі патентів України на винаходи **25.10.2012.**

Перший заступник Голови
Державної служби
інтелектуальної власності України О.В. Янов

Державна служба інтелектуальної власності України
Ідентифікаційний код 37552556
№1

Приложение 8.

«Затверджую»
Головний інженер ПАО «Сумигаз»

_____ О.В. Деменко

15.10.2012, № 22/1

АКТ ВПРОВАДЖЕННЯ
результатів дисертаційної роботи

Цей акт складено в тому, що результати дисертаційної науково-дослідної роботи «Каркасна модель даних та її застосування для розробки і впровадження CASE-засобів та інформаційних систем» в частині глав 2, 3, 4 та 5, виконаної к.ф.-м.н, снс Панченко Б.Є., впроваджено у виробництві. Матеріали цієї роботи стали частиною системи «Ділянки обходу абонентів».

Начальник відділу
інформаційних технологій С.І. Гавришенко

Акт
внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения CASE-генератора прикладных многопользовательских информационных комплексов – инструментальной системы SWS.

г. Кировское 3 декабря 2009 года

В соответствии с договором о сотрудничестве между Малым научно-производственным внедренческим предприятием «Монолог» и ОАО «Шахта Комсомолец Донбасса» в период с 1994 по 1997 год были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения CASE-генератора прикладных многопользовательских информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов инструментального средства на момент 1997 года были созданы и внедрены в промышленную эксплуатацию следующие прикладные программные системы:

1. *Учет отгрузки угля по потребителям;*
2. *Реализация угля потребителям;*
3. *Анализ заболеваемости;*

Все прикладные системы поддерживались в актуальном состоянии до 2000 года. В настоящий момент в актуальном состоянии поддерживается система «*Учет отгрузки угля по потребителям*».

Главный инженер
ОАО "Шахта"Комсомолец Донбасса" В.В. Раскидкин

Документ подготовил:
Руководитель департамента по информационным технологиям
Лыга А.С.

УТВЕРЖДАЮ
Гл. инженер ТТЦ
А.В. Соколов

Акт

применения при разработке прикладного программного обеспечения задач ТТЦ методического и инструментально-технологического программного обеспечения CASE- генератора прикладных сетевых информационных комплексов - инструментальной системы SWS

г. Москва *9 апреля 1998 г.*

В соответствии с договором о сотрудничестве между **Малым научно производственным внедренческим предприятием "Монолог"** и **Федеральным государственным предприятием "Телевизионный технический центр"** (ТТЦ) в период *1993 по 1995 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения CASE генератора прикладных сетевых информационных комплексов - инструментальной системы SWS (в дальнейшем - "инструментальное средство SWS).

На базе инструментального средства SWS специалистами ТТЦ при консультативном участии авторов этого инструментального средства на настоящий момент созданы и внедрены в промышленную эксплуатацию прикладные программные продукты: задача *АРМ "Картотека Первого отдела"* и первая очередь задачи *"Автоматизированная система подготовки и обработки документов" (АСПОД-1).*

В настоящий момент названные прикладные программные продукты поддерживаются в актуальном состоянии.

От предприятия "Монолог" Б.Е. Панченко

От ТТЦ Н.Н. Воробьев

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Москва *25 марта 1998 г.*

В соответствии с договором о сотрудничестве между **Малым научно-производственным внедренческим предприятием "Монолог"** и **ЗАО "Интерьер" (Московская мебельная фабрика)** в период с *1993* по *1998 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов этого инструментального средства в настоящий момент созданы и внедрены в промышленную эксплуатацию следующие прикладные программные системы:

1. *Сбыт готовой продукции:*
   - *1.1. Склад готовой продукции,*
   - *1.2. Учет взаиморасчетов с покупателями,*
   - *1.3. Документопроизводство (счета и накладные) и бухгалтерские расчеты по сбыту;*
2. *Планирование:*
   - *2.1. План производства,*
   - *2.2. Расчет себестоимости,*
   - *2.3. Расчет рентабельности и ценообразования;*
3. *Снабжение:*
   - *3.1. Учет материалов на материальном складе,*
   - *3.2. Учет взаиморасчетов с поставщиками,*
   - *3.3. Документопроизводство и бухгалтерские расчеты по снабжению;*
4. *Учет основных средств;*
5. *Расчет заработной платы (сдельной и повременной);*
6. *Финансы:*
   - *6.1. Касса,*
   - *6.2. Банк,*
   - *6.3. Учет подотчетных лиц,*
   - *6.4. Учет расчетов с дебиторами и кредиторами,*
   - *6.5. Валютные операции;*
7. *Производство:*
   - *7.1. Рабочее место технолога,*
   - *7.2. Учет движение сырья, материалов и деталей в производстве,*
   - *7.3. Производственный отчет.*

Все прикладные системы в настоящий момент поддерживаются в актуальном состоянии. На базе инструментального средства SWS планируются новые разработки.

Генеральный директор *Г.В. Анашкин*

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Киев *17 декабря 1997 г.*

В соответствии с договором о сотрудничестве между **Малым научно-производственным внедренческим предприятием "Монолог"** и **Главным управлением военно-строительных частей Министерства Обороны Украины** (в/ч № А0200) в период с *1994* по *1997 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов этого инструментального средства на настоящий момент создана и внедрена в промышленную эксплуатацию прикладная программная система *"Отчет финансово-экономической деятельности подразделений строительных органов"*.

В настоящий момент названная прикладная система поддерживается в актуальном состоянии. На базе инструментального средства SWS планируются новые разработки.

Начальник управления,
Генерал-майор *А.Н. Полтавцев*

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Хмельницкий *13 ноября 1997 г.*

В соответствии с договором о сотрудничестве между **Малым научно-производственным внедренческим предприятием "Монолог"** и **Войсковой частью № А3779** в период с *1994* по *1997 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов этого инструментального средства на настоящий момент созданы и внедрены в промышленную эксплуатацию следующие прикладные программные системы:

1. *Начисление денежного довольствия офицеров и прапорщиков;*
2. *Отчет финансово-экономической деятельности подразделений строительных органов.*

Все прикладные системы в настоящий момент поддерживаются в актуальном состоянии. На базе инструментального средства SWS планируются новые разработки.

Заместитель командира части,
Полковник *С.А. Игнатенко*

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Киев *13 октября 1997 г.*

В соответствии с договором о сотрудничестве между **Малым научно-производственным внедренческим предприятием "Монолог"** и **Государственным коммунальным предприятием "Киевжилтеплокоммунэнерго"** в период с *1994* по *1997 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов этого инструментального средства на настоящий момент созданы и внедрены в промышленную эксплуатацию следующие прикладные программные системы:

1. *Учет выработки тепла по котельным города Киева;*
2. *База данных Района эксплуатации котельных №5 (Ленинградский район г. Киева):*
   - *2.1. Сотрудники,*
   - *2.2. Оборудование,*
   - *2.3. Теплотрассы,*
   - *2.4. Экономические показатели;*
3. *Формирование форм бухгалтерской отчетности аппарата управления.*

Все прикладные системы в настоящий момент поддерживаются в актуальном состоянии. На базе инструментального средства SWS планируются новые разработки.

Директор *А.П. Матвиенко*

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Хмельницкий *18 ноября 1997 г.*

В соответствии с договором о сотрудничестве между **Малым научно-производственным внедренческим предприятием "Монолог"** и **Хмельницкой городской поликлиникой №3** в период с *1994* по *1997 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов этого инструментального средства на настоящий момент созданы и внедрены в промышленную эксплуатацию следующие прикладные программные системы:

1. *Учет прививок против дифтерии;*
2. *Учет платных услуг;*
3. *Регистрация талонов амбулаторных больных.*

Все прикладные системы в настоящий момент поддерживаются в актуальном состоянии. На базе инструментального средства SWS планируются новые разработки.

Главный врач  ***В.Л. Розгонюк***

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Сумы *"20" сентября 1997 г.*

В соответствии с договором о сотрудничестве № **38** от *17.05.96* между **Малым научно-производственным внедренческим предприятием "Монолог"** и **Представительством фонда Государственного имущества Украины в городе Сумы** за период с *1995* по *1997 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS под руководством авторов этого инструментального средства была создана и внедрена в промышленную эксплуатацию прикладная программная система ***"Приватизация"***. В настоящий момент названная прикладная программная система поддерживается в актуальном состоянии.

Директор Представительства
Фонда Государственного имущества
Украины в городе Сумы ***В. В. Панченко***

Акт

внедрения прикладного программного обеспечения, разработанного на базе методического и инструментально-технологического программного обеспечения: CASE – генератора прикладных сетевых информационных комплексов – инструментальной системы SWS.

г. Москва *"23" марта 1997 г.*

В соответствии с договором о сотрудничестве между **Малым научно-производственным внедренческим предприятием "Монолог"** и **Федеральным дорожно-строительном Управлением при Министерстве обороны Российской Федерации** в период с *1993* по *1997 год* были проведены работы по промышленной эксплуатации методического и инструментально-технологического программного обеспечения: CASE–генератора прикладных сетевых информационных комплексов – инструментальной системы SWS (в дальнейшем – "инструментальное средство SWS").

На базе инструментального средства SWS специалистами нашей организации при консультативном участии авторов этого инструментального средства были созданы и внедрены в промышленную эксплуатацию следующие прикладные программные системы:

1. *Учет штатной численности кадрового состава;*
2. *Учет работ ремонтных предприятий;*
3. *Учет дорожно-строительной техники;*
4. *Учет травм и происшествий.*

Все прикладные системы в настоящий момент поддерживаются в актуальном состоянии. На базе инструментального средства SWS планируются новые разработки.

ВРИО Начальника штаба **ФДСО** при Министерстве Обороны Российской Федерации, полковник *Ю. Гилинец*

Приложение 9.

Начало промышленной эксплуатации КМД посредством SWS: рекламные материалы 1992 года

«АиФ»-РЕКЛАМА

Программирование
без программиста -
система SWS.

Уникальный CASE-генератор EXE-модулей позволяет создавать без написания исходных текстов ЛЮБЫЕ виды сетевых АСУ единым модульным методом. Пользователь не зависит от разработчика-прикладника. Имеется возможность самостоятельно модифицировать такую АСУ, использовать идеологию Распределенных Баз Данных и режим Реального Времени. Комплекс АСУ можно дополнить графическим редактором MonoCAD (САПР-пакет, полный СИ-интерфейс, открытые СИ-библиотеки, полная параметризация).

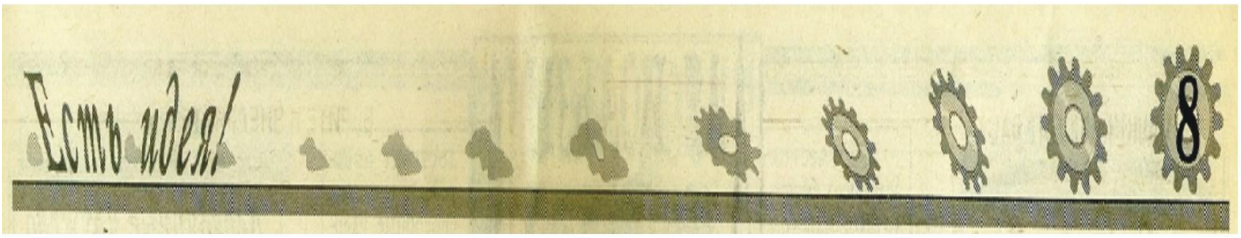
Заказы по тел. в Москве:
925-92-03, для МП "Монолог".

Программирование без программиста

CASE-генератор SWS позволяет создавать как открытые сетевые АСУ РВ с распр. БД, так и отдельные EXE-прогр. единым модульным безлистинговым методом. MonoCAD - САПР-пакет (СИ-интерфейс, СИ-библиотеки, полная параметризация; X-Windows, MS-Windows).

Презентация - на выставке "Софтул-93" с 5 по 9 октября (Москва, ВВЦ, стенд № 6).

МП "Монолог", Украина, г. Сумы, (0542) 27-51-83.



РАБОТАЕМ ПО-КРУПНОМУ

CASE-технология программных продуктов - не имеющая аналогов на мировом рынке!

В это направление западные бизнесмены вкладывают миллионы долларов!

Продается готовый программный инструмент SWS-генератор для обслуживания предприятий и организаций, включающий в себя ВСЕ ВИДЫ ДЕЯТЕЛЬНОСТИ ЛЮБОЙ ОРГАНИЗАЦИИ: бухгалтерию, склад, сбыт, любые виды банковской деятельности, неограниченное число АРМ-ов, объединенных сетью. Их создание и настройка осуществляются в самой системе.

После приобретения системы и ее настройки владелец становится АБСОЛЮТНО НЕЗАВИСИМЫМ от разработчика, благодаря специализированной CASE-технологии. При этом тексты программ разработчика не нужны. Кроме того, система может принимать в себя любые программы и подгружать их в основной выполняемый файл, так называемый EXE-модуль (интерфейс пользователя). Такой генератор EXE-модулей отсутствует на мировом рынке!

Дополнительно к системе можно приобрести оригинальный графический редактор, позволяющий самостоятельно решать проблемы САПР любой конфигурации в комплексе с АСУ - MonoCAD.

При участии инвеститоров, готовых вложить в эту перспективнейшую систему свои средства, система может конкурировать на мировом рынке программных средств.

Приложение 10.

Отзывы в профессиональной Украинской и Российской прессе о ходе промышленной эксплуатации SWS в 1994-1995 гг.

КОМПЬЮ ТЕРРА
КОМПЬЮТЕРНЫЙ ЕЖЕНЕДЕЛЬНИК
ПОДПИСНОЙ ИНДЕКС – 32197
17 ОКТЯБРЯ 1994 #35 (66)

SOFTWARE

Программисты больше не нужны?

Выставка SofTool поразила обилием бухгалтерских программ, которых, видимо, становится все больше. Это не совсем понятно - мне лично кажется, что их время ушло. Точнее, рынок универсальных бухгалтерий давно уже поделен и пытаться сейчас конкурировать, скажем, с "1С" практически невозможно. К тому же развитие индустрии ПО (я имею ввиду совершенствование инструментальных средств и накопление разработчиками практического опыта) привело к тому, что любое достаточно крупное предприятие может создать собственными силами или заказать свой вариант бухгалтерской программы либо систему управления своими специфическими данными. При этом, с одной стороны, резко сократились время и стоимость разработки, а с другой — такой продукт изначально производится под конкретного заказчика и наилучшим образом отвечает его требованиям (конечно, если тот четко их сформулирует, а исполнитель точно будет им следовать).

Алексей Дементьев

В контексте вышесказанного меня весьма заинтересовал инструментальный пакет фирмы "Монолог" из города Сумы. Он позволяет автоматизировать создание информационных систем и, в частности, финансовых программ. Замечу, кстати, что компания "Монолог" была на выставке единственным представителем Украины. Кроме того, фирма "Монолог" в этом году участвовала в выставке CeBit. И там она оказалась единственной украинской фирмой, выступавшей в классе инструментального программного обеспечения.

Собственно, первоначально мое внимание привлекла аббревиатура "CASE", упоминавшаяся среди характеристик пакета. Понятие "CASE" расшифровывается как средства, автоматизирующие труд программиста. Применяя эту технологию, проектировщик той или иной системы оперирует понятиями, соответствующими моделируемому процессу. При этом исключаются такие чисто программистские термины, как "чтение файла", "сравнение переменных" и так далее.

Соответственно уменьшается и вероятность технических ошибок, которых только в двух вышеупомянутых низкоуровневых операциях бывает несколько десятков. При традиционном подходе к разработке информационных систем на языках высокого уровня, таких как C++, dBASE или Clipper, около 90% времени приходится на исправление именно таких ошибок, значительная часть которых, в свою очередь, приходится на интерфейсные программы. И лишь оставшиеся 10% продуктивно расходуются на конструирование логики приложений.

CASE позволяет избежать многих трудностей за счет того, что программа строится из готовых и отлаженных блоков. В итоге высвобождается время на интеллектуальную работу. В идеале посредством CASE-технологии готовые продукты будут выпускать не программисты, а специалисты в данной прикладной области. При этом на экране они будут оперировать привычными для них диаграммами, таблицами, осуществлять с помощью мыши логические связи между функциональными элементами и так далее.

Некоторые составляющие этой технологии реализованы в пакетах корпораций Oracle, Gupta, Ingres и других. Все это работает на мощных и весьма дорогих компьютерах. К примеру, минимальные требования для CASE-системы компании Symantec (Enterprise Developer), работающей под Windows — 8Mb ОЗУ (реально — 12-16Mb) и процессор не ниже 486DX33. Иначе придется часто и подолгу перекуривать. Еще более жесткие условия предъявляют к аппаратным средствам инструментальные средства Oracle.

Поэтому и удивила CASE-система, работающая в DOS и предъявляющая требований не больше, чем Clipper, на котором и написана. В SWS (Server-Workstation System), разработанной фирмой "Монолог", нет традиционных E-R-диаграмм (Entity-Relation), но тем не менее она представляет собой настоящую CASE-систему. Иными словами, разработчик при проектировании задает логические взаимосвязи между файлами БД путем редактирования признаков файлов и их полей. Таким же образом описывается протокол извлечения, обработки и обновления данных. При этом создаваемым операциям может быть присвоено имя. Затем оно ставится в соответствие элементу ниспадающего меню, который и инициирует выполнение соответствующего действия.

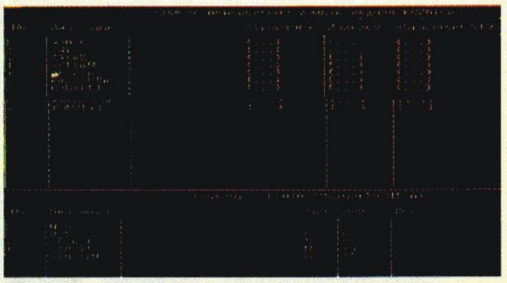
Так реализуется безлистинговая технология создания ПО, исключающая традиционные программистские ошибки в готовом продукте. Остается только правильно сформулировать задачу, то есть спроектировать реляционно-иерархическую логику построения базы данных. Разработчику достаточно минимального знания синтаксиса Clipper, из которого применяются практически лишь элементарные математические и логические операции сравнения. Однако здесь есть возможность и явного использования кода, написанного непосредственно на Clipper.

Приведу примерную технологию проектирования корректной базы данных. В первую очередь формируют полный список актуальных признаков изучаемой системы. На этом этапе определяют информационное наполнение БД и разбивку всех признаков на несколько логических уровней. В результате получают список таблиц, каждой из которых ставится в соответствие свой файл. На рисунке 2 приведены две копии экрана. В верхней половине редактируется список таблиц, в нижней определяется структура таблицы "OPLATA". В интерактивном режиме на стадии проектирования можно когда угодно изменять количество и тип полей любой таблицы.

Далее для каждого файла определяются индексы или процедуры их заменяющие. Они позволяют ускорить поиск данных по выбранным ключам.

Наиболее сложный и ответственный этап — проектирование всех информационных потоков. Оно представляет собой описание всех процедур, вызываемых при вводе данных в определенные смысловые либо ключевые поля и определяющих взаимную целостность и соответствие данных в логически связанных таблицах. Этот процесс получает имя — алиас, используемое затем в меню.

К примеру, требуется осуществлять в режиме реального времени проводку предоплаты определенного товара. Это значит, что синхронно со вводом всех данных об оплате корректируется информация о долге данного заказчика. Операция инициируется нажатием клавиши "Enter", подтверждающим завершение редактирования данных о проводке. После этого в таблицу, хранящую информацию об оплате, поступит соответствующая информация, а в таблице, хранящей счета организаций-должников, будет скорректирована сумма долга. Далее формируется систе-



[illegible] впрочем, и завершает процесс разработки продукта.

SWS позволяет создавать информационные системы с распределенными базами данных. При этом часть файлов может находиться в общем пользовании на диске сервера, а остальные, применяемые для чернового ввода и редактирования, — на локальном винчестере или закрытом для остальных диске. Таким образом, в данной системе доступно написание продуктов для групповой работы в сети.

Создаваемый программой код представляет собой некий макроязык, скрытый от пользователя. В процессе работы он интерпретируется run-time модулем. Такой подход, обеспечивая, с одной стороны, достаточно высокую производительность, дает возможность динамично дорабатывать продукт без утомительного перекомпилирования.

Интересно, что в последнее время за рубежом проводится большая работа с целью стандартизировать технологию разработки информационных систем. Такой стандарт принят в Англии (SSADM), нечто подобное предложила компания Siemens. Фирма Монолог, как заявил нам ее директор Борис Панченко, предлагает свое решение этой проблемы в качестве всеукраинского стандарта. Возможно в ближайшем будущем она будет принимать участие в конкурсе на разработку такого стандарта.

В заключение отметим, что одна из систем, созданных посредством SWS, внедрена в Агентстве по распространению изданий "АиФ" (АО Аргументы и Факты).

Новые компьютеры — старая политика

4 октября в отеле Балчуг состоялась презентация новых продуктов корпорации Compaq. Для этого в Россию приехало все европейское руководство Compaq. Лейтмотивом всех выступлений прошла мысль, что Compaq должен стать первой компанией во всех секторах рынка компьютерных систем на базе процессоров Intel.

КОМПЬЮ ТЕРРА
КОМПЬЮТЕРНЫЙ ЕЖЕНЕДЕЛЬНИК
ПОДПИСНОЙ ИНДЕКС—32197
21 НОЯБРЯ 1994 #42-43 (73-74)

Программистам нужен стандарт?

24-29 октября этого года в Киеве проходила первая на Украине международная компьютерная выставка "Компьютерэкспо-94". В ней приняли участие 45 фирм. Шесть из них представляли Москву, остальные - Украину.

А.Федорина, Н.Мартынич

Если оценивать экспозицию по торговым маркам, красовавшимся на фризах стендов, то на первых порах создавалось впечатление, будто на крохотном пятачке Торгово-промышленной палаты Украины, где проходила эта тусовка, собралась вся мировая компьютерная элита. Symantec, Digital, HP, IBM, Nowell - просто дух захватывало. Но уже первые два-три вопроса к представителям фирм все поставили на свои места. С какой-то особой гордостью и нарочитой амбициозностью вчерашние вполне советские парнишки на весьма среднерусском диалекте, приправленном британским апломбом, вовсю расхваливали товар, и без того признанный во всем мире. Авторизированные (и не очень) дилеры, реселлеры и дистрибьюторы, сотрудники всевозможных СП и филиалов... Словом, гудит компьютерное купечество. Торгует "забугорной" оргтехникой, периферией, софтом, окучивает покупателя.

А где же мы? Где тот всемирно известный интеллект, которым по праву гордилась "одна шестая суши"? Который на Западе всегда высоко ценили, но куда как дешево покупали? И который в силу определенных и всем известных причин не был, да, видно, и не мог быть реализован в собственной стране?. Казалось бы, времена изменились, но...

Как показала выставка, компьютерный рынок Украины на 90% процентов сформирован из предложений мировых лидеров оргтехники и программного обеспечения, причем в пропорции один к двадцати отнюдь не в пользу последнего. Прогуляешься по такому рынку, послушаешь купцов-зазывал, рассчитывающих на покупателя-непрофессионала, и вопрошаешь недоуменно: то ли иссяк украинский "мозговой потенциал (во что, ежели честно, мало верится), то ли опять ушел в подполье до лучших времен? А почему? И если он все-таки жив, то должен хоть как-то заявлять о себе, показываться на люди, получать, разумеется, свои шишки и тумаки, но вместе с тем крепнуть и развиваться.

Рынок не прощает раскачки и промедления. К нему трудно примериться, еще труднее предлагать ему свои продукты, на которые он смотрит сегодня если не с пренебрежением, то с недоверием - наверняка. И все же это единственный путь, на который пора выходить уже сейчас, ибо завтра будет поздно.

Среди тех, кто это понял и пытается нынче заявить о себе всерьез - ряд украинских фирм, уже накопивших некоторый опыт в неравном соперничестве с мировыми компьютерными гигантами и их не всегда профессиональными представителями. Так, в выставке приняла участие одесская компания Виста. Она выпускает компьютеры под уже известной одноименной маркой, а также контрольно-кассовые аппараты "ЭККА" и POS с оригинальным программным и аппаратным обеспечением. Свои IBM-совместимые машины "NOVA" продемонстрировала фирма Volyn Information Technologies из города Ровно.

Внимание посетителей привлекла внешне скромно представленная сумская фирма Монолог. Ее программные продукты (а это CASE-система SWS и графический редактор MonoCAD) вызвали не только массу заинтересованных вопросов, но и живейшее удивление: неужели и на украинской земле может произрастать нечто столь разумное, простое и универсальное?

Трудно согласиться с мыслью, что на Украине рынок ПО отмирает - а именно эта мысль неоднократно звучала в высказываниях руководителей процветающих фирм, занимающихся торговлей компьютерами и программным обеспечением от мировых лидеров. Трудно по той простой причине, что именно качество и степень внедрения ПО - один из важных признаков уровня цивилизованности общества и государства. Поэтому мы можем констатировать, что спрос на качественный софт (и особенно инструментарий) на Украине огромен, что еще раз убедительно подтвердила выставка "Компьютерэкспо-94". Другое дело, что мы переживаем кризис в этом направлении, в немалой степени обусловленный нашим дилетантизмом. И вот тому яркий пример.

В свое время, когда аббревиатура АСУ еще не набила оскомину, и на волне скорее своеобразной моды, нежели глубокого понимания, появился спрос на автоматизированные бухгалтерии, отделы кадров и тому подобное, эту нишу заполнил товар низкого качества. Появилось много фирм прокомпьютерного толка, в программисты пошли самоучки и недоучки. Как следствие, был получен и само-недо-продукт. Все это не могло не сказаться на рынке - родилось недоверие, упал спрос, сама идея была дискредитирована.

Справедливости ради следует отметить, что такая хворь постигла не только Украину. Однако в то время как Москва, к примеру, уже переболела, мы все еще находимся в стадии реабилитации. А поскольку, как говорится, свято место пусто не бывает, за время этой болезни рынок заполнила зарубежная продукция и, как показала выставка, превалирует на нем до сих пор.

Жаль, не понимают бойкие торговцы заокеанским товаром, что действительно прекрасные изделия зарубежных фирм прекрасны "там". В нашу ситуацию, в здешние условия работы эти продукты вписываются далеко не всегда. Их гениальные западные мозги зачастую не в силах переварить наше удивительное бытие - тут явно нужны своеобразные.

И что же мы имеем в классе программного обеспечения? Опыт прошедшей выставки свидетельствует, что софтовые предложения ограничиваются прикладными бухгалтерскими разработками. Бесспорным лидером здесь выступила днепропетровская фирма Баланс-Центр. Ее программа "Село для бухгалтера с компьютером" заняла первое место на Втором украинском конкурсе программного обеспечения по бухгалтерскому учету и аудиту.

Кроме того, выставка выявила, что многие фирмы, помимо оказания дилерских и прочих услуг, предлагают и свои собственные пакеты, но об их качестве говорит даже цена. К примеру, систему автоматизации деятельности трастовых компаний предлагали всего за 50$. К счастью, наш покупатель начинает понемногу усваивать западное правило: мы не так богаты, чтобы покупать дешевое. Нам давно пора самим производить высококачественное инструментальное ПО, но, к сожалению, ни производитель, ни покупатель пока этого не понимают. Более того, беседуя со специалистами вышеупомя-

Защ
свои
проб

МНЕНИЕ

«Монолог» представляет:

Инструментальные программные средства европейского качества!
— CASE-технология SWS:
программирование без программиста!
— CAD-систему MonoCAD:
мир графической параметризации!

На компьютерный рынок Украины провинциальная фирма «Монолог» из Сум пришла уже известной в Москве и на более динамично развивающемся в то время российском рынке. Известной не только своим участием в престижных софтовых выставках, но, прежде всего, снискав поддержку и уважение именитых пользователей производимого ею интеллектуального продукта. Пришла как раз в то время, когда на доселе почти безветренном украинском компьютерном рынке начало активизироваться «броуновское движение». Предложенный ею инструментальный программный товар оказался настолько спросовым, что вызвал немало удивления и восхищения потенциальных и реальных пользователей. Сегодня мы предоставляем слово пользователям системы.

г. Москва, ВГРТК «Останкино», тел. 215-10-24
Исаева Татьяна Юрьевна

Достоинства:
– Сокращает затраты по созданию программных модулей, т. к. не требует написания листингов программ.
– Программирование сводится к корректному построению БД, т. е. совокупности .dbf-файлов, установлению реляции между ними. Фактически, задача считается решенной, когда создана БД.
– Написание пунктов меню и внешнее оформление проводится самой SWS.
– Корректно обрабатывает БД, т. е. отслеживает транзакции после окончания работы с файлами. Обеспечивает резервное копирование, что защищает данные при сбое или аварийной ситуации на сервере, и корректную работу с памятью, в том числе оперативной.
– SWS обеспечивает совместимость с БД типа dBf (типа, широко использующегося на практике), т. е. позволяет включать в себя эти БД и производить операции с ними, что не приводит к потере ранее накопленной информации.
– SWS – открытая система, позволяющая вносить изменения и дополнения, совместима с Clipper, что дает дополнительные возможности для творчества программиста, т. к. при необходимости допускает включение внутрь .exe-модуля функций, которые не предусматриваются SWS.
– SWS – новый инструмент, позволяющий не писать программы, а формировать их из готовых объектов и блоков, т. е. процесс программирования сводится к конструированию программы.
– Допускается работа на ПК начиная с 286-й модели (и выше).
– Невысокая стоимость программы и поддержка фирмы-производителя — «Монолога».

Недостатки:
– Необходимо устранить некоторые недостатки документации.
– Пользовательский экранный интерфейс может быть более разнообразным, но, в принципе, устраивает.
Мы пользуемся системой в течение почти трех лет. В настоящий момент разрабатываем прикладную систему обработки документации для мелкоцентра. SWS сократила время на разработку и предоставила в именно том, что является стандартом кодирования прикладного информационного ПО. Мое мнение, что это самая перспективная разработка СНГ за последние годы.

г. Москва, АО «Аргументы и факты», служба распространения, тел. 923-68-59
Желтов Олег Владимирович

Системой пользуется 2 года.
Достоинства:
– Система быстро совершенствуется, т. е. любые сложные доработки можно быстро внедрить.
– Удобный для простого пользователя интерфейс.
– Система очень доступна и гибка, девиз: «Программирование без программиста» соответствует самой системе.
– Быстрая корректировка, внесение изменений.
– Обучение и поддержка «Монолога».
– Нет зависимости от разработчиков.
Помогал решить очень много нестандартных информационных задач, жизненно необходимых редакции и успешно внедренных.

г. Москва, АО «Артифекс», тел. 252-48-81
Видерман Михаил Феликсович

Достоинства:
– Достаточно легкий в освоении продукт.
– Идеология SWS определяет структурированный подход к созданию автоматизированных систем управления. Заставляет правильно работать.
– Быстрота написания приложений. Программа по зарплате на комбинате была написана за 2 месяца и перекрыла весь спектр обсчета. АСУ была организована за 1,5 года.
– У разработчика не возникает проблем при работе в сети.
– SWS не сравнима по качеству создания приложений с Oracle. В достаточной степени надежна.
– Упрощенный интерфейс оберпулся достоинством, т. к. не создает проблем для непрофессионалов, а разработчика этот вопрос не волнует вообще.
Недостатки:
– В используемом .dbf-стандарте файлов баз данных фирмой Nanticket не реализована идеология клиент/сервер.
– Недостаточно современный уровень сервиса при генерации среды выполнения.
С помощью АО «Артифекс» система внедрена в АО «Новые технологии», АОЗТ «Инженер» и на фирме «К-8». Все они находятся в Москве. У системы очень большое будущее.

г. Москва, Федеральное дорожно-строительное управление при Министерстве Обороны Российской Федерации, тел. 296-37-56, 915-41-20
Аршенов Виталий Кондратьевич

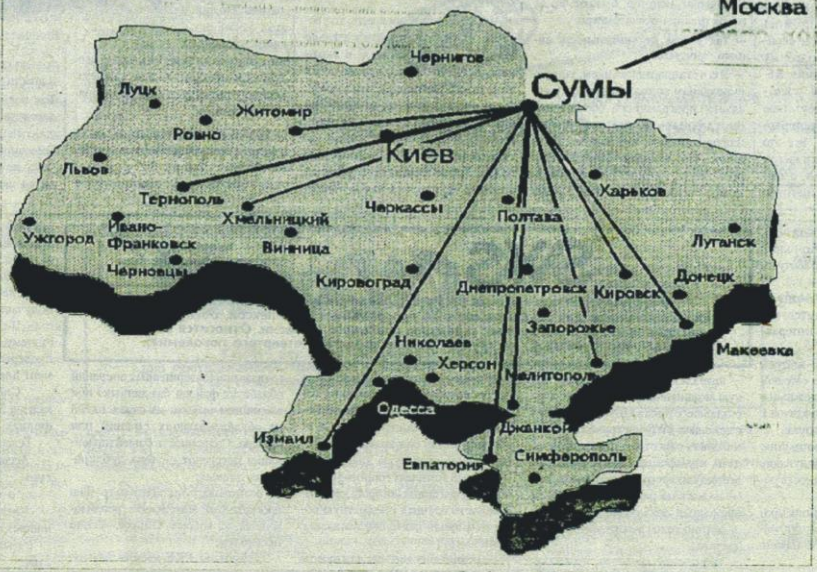
С системой работает уже год. За 4 месяца с помощью SWS один «оживил» локальную сеть всего Управления.
Достоинства:
– Быстродействие.
– Быстрое решение конкретной задачи.
– Минимум времени для создания программных продуктов.
– Создание программ в сетевом варианте.
– Сервисное обслуживание со стороны «Монолога».
Недостатки:
– Редакторский сервис по генерации отчетов для печати:
Знаком со многими продуктами, но предпочтение отдаю именно этой системе, т. к. она удовлетворяет всем моим требованиям к инструментарию. Считаю, что у системы большое будущее, и она может завоевать рынок СНГ.

г. Москва, Ассоциация промбанков «Россия», тел. 281-87-97
Шмелев Геннадий Валентинович

С системой работает 2 года. На ней написано более 10 продуктов, в том числе «Справочник по банкам России».
Достоинства:
– Несложная система.
– Позволяет работать с ней непрофессионалу в области кодирования на языках программирования, это подтверждает девиз – «Программирование без программиста».
– Уверенно и очень надежно работает в сети.
Недостатки:
– Стандартизованность экранных форм и наборов меню.
– Техническая зависимость от платформы DOS.
У системы есть будущее, она продается на рынке СНГ и может выступать как готовый продукт. По сравнению с другими системами, такими, как Oracle или FoxPro, она гораздо дешевле при почти аналогичных свойствах, оригинальна и очень проста в освоении. Надо отдать должное «Монологу» за поддержку своего продукта и работу с пользователями. Для СНГ это почти чудо.

г. Москва, частный пользователь, тел. 287-69-51
Медунов Александр Львович

Достоинства:
– Правильная идеология построения БД, ее структуры.
– Этап программирования (кодирования – в привычном смысле слова) отсутствует.
– Легкость работы в сети: она вообще автоматически обеспечивает работу в сети, проектировщик об этом не заботится.
Недостатки:
– Стандартизация экранного пользовательского интерфейса – неизменность формы представления информации.
– Нет возможности автоматического подключения к информации БД не .dbf-стандарта и выхода ...



➤ стр. 14

КОМПЬЮ ТЕРРА
КОМПЬЮТЕРНЫЙ ЕЖЕНЕДЕЛЬНИК
ПОДПИСНОЙ ИНДЕКС — 32197
13 ФЕВРАЛЯ 1995 • #6 (86)

ИНТЕРВЬЮ

CASE: АСУ для программиста

Ныне уже, наверное, все знают, что означает аббревиатура CASE. Однако в большинстве случаев она связывается с западными и весьма дорогими программными продуктами. К примеру, CASE-средства входят в стандартную поставку инструментальной оболочки SQL-сревера Oracle. Сегодня мы беседуем с Борисом Панченко, директором фирмы "Монолог", о проблемах использования этих технологий для автоматизации производства программного обеспечения.

Михаил Френкель

— Насколько мне известно, в СССР еще до перестройки проводилось широкомасштабное внедрение АСУ в производство. Где же все это сейчас?

— По-видимому, производственной, да к тому же "отечественной" АСУ, увы, не существует. Какие-то локальные разработки, решение чисто торговых, складских, сбытовых или банковских задач — этого хоть отбавляй. А к тому, что давным-давно внедрено в современном западном производстве, программисты СНГ только подбираются, да и то лишь на уровне постановок.

Дело еще осложняется тем, что на нашем "совдеповском" производстве с его общепринятым подходом практически невозможно развернуть полный комплекс информационной автоматизации. Главное препятствие — это страшная нестабильность во всем, начиная с законодательства. Я думаю, не стоит объяснять, какой хаос это вносит в любую отработанную программную среду. Ведь любая даже незначительная корректировка готовой крупной системе отнимает у программиста массу полезного времени.

Вот и подумайте, как можно нормально внедрить хотя бы часть АСУ, если чуть ли не каждую неделю необходимо что-то менять. Помимо прочего руководителя-производственника терзает и другая немаловажная проблема — нестабильность и текучесть кадров, то есть высококлассных программистов. Четыре года назад мы поняли, что невозможно создать промышленное прикладного программного обеспечения больших сетевых систем стандартными методами программирования.

Сталкиваясь с проблемами разработок больших информационных систем (таких, как сетевые АСУ), которые призваны эффективно управлять сложным производством, наши разработчики за эти годы создали свою собственную концепцию, даже, если хотите, свое мировоззрение на сетевые системы. Так появился на свет наш инструмент — система SWS.

Это сейчас мы знаем, что реализованная нами методология имеет специализированное название на международном компьютерном рынке. А тогда ни о каких CASE-технологиях, конечно, и не думалось. Первые же опыты дали потрясающие результаты. Теперь уже никто, наверное, не сомневается, что создавать большие информационные системы можно только на таких CASE-генераторах (или инструментах 4-го поколения, как их еще называют). Идея, между тем, проста — прежде, чем брать на себя ответственность что-либо автоматизировать, автоматизируй свой собственный труд.

В общем, назрел вполне конкретный вывод — разрабатывать информационные системы должен не программист, а специалист в области информационных систем (как принято говорить — инженер-архитектор Баз Данных).

— В нашей публицистике модная в 70-80-х годах аббревиатура АСУ была настолько замызгана, что, без преувеличения, набила у производственников оскомину. Действительно, реальных результатов производство в то время так и не ощутило. Вы не боитесь остаться не у дел из-за порочности самой отечественной идеи автоматизации?

— Не боимся. На наш взгляд, идея комплексной автоматизации вполне жизнеспособна. Просто нам, как всегда, не хватает профессионализма. Судите сами: с точки зрения информатики, любое производство — это всего лишь взаимозависимые и хорошо структурированные информационные потоки, которыми необходимо четко оперировать. Это может быть анализ информации, внесение каких-то корректив или что-то еще. Когда производство моделируется верно, информационные срезы в любой момент времени дают объективную и оперативную картину состояния производства. А именно это и нужно современному руководителю, изможденному информационными нагрузками сегодняшнего дня. Мы даже сформулировали специализированный тест для определения качества информационной автоматизации производства — "операционный день фирмы". Если на предприятии можно говорить о появлении такого понятия, как операционный день фирмы, то оно автоматизировано достаточно успешно, по крайней мере, по этой статье. Смысл этого понятия — возможность ежедневно, скажем, к вечеру (а в идеале — в любую минуту) снимать интересующую руководителя информацию прямо с экрана своей станции.

— Применительно к нам это воспринимается как фантастика...

— На самом деле — ничего сверхъестественного. Если бы наши предприятия были в состоянии платить западным коллегам, все это давно бы стало реальностью. Но на Западе эти системы, к сожалению, стоят отнюдь не дешево.

— Как я понимаю, все то, чем вы занимаетесь, — это в какой-то степени наука, идущая от практики. Но как на это смотрит наука фундаментальная? Не изобретаете ли вы велосипед?

— Что касается велосипеда, то уверены, что нет. А вот относительно науки — вопрос злободневный. Не секрет, что в СНГ почти не осталось высококлассных творителей организации баз данных. Старые школы разрушены временем: кто подался в бизнес, кто за границу, кто куда... Разумеется, отсутствие фундаментальной основы приводит к стихийному развитию прикладной науки: производство нуждается в развитии технологии проектирования баз данных, а наука на этот вопрос не успевает ответить.

Вот совсем свежий пример. Недавно в банках Украины исключительно административным методом было введено требование помимо МФО и номера расчетного счета указывать в платежке еще и коды ОКПО. Мы, естественно, понимаем, что это послужит повышению точности платежей в межбанковской компьютерной системе. Однако любой эксперт заметит, что введение избыточного количества уникальных ключей для отождествления клиента засорит базу данных.

— На чем же основывается ваша технология?

— На практическом опыте и знаниях, почерпнутых из западных источников. Дело в том, что "Америки мы не открываем". На Западе роль науки поняли давно, и результаты очевидны. При желании знания получить можно и у нас. Но если западный коллега получает специализированное высококлассное образование за пару лет, причем академически, то нам приходится годами барахтаться в литературе, самостоятельно выискивая, анализируя и систематизируя крохи знаний. Вузы наши пока еще довольно консервативны, а институты повышения квалификации — пустая формальность.

— Не слишком ли резкая оценка?

— Думаю, нет. Недавно к нам приезжали из Тернопольской академии народного хозяйства. Они заинтересовались нашей системой. А попробовав решить с ее помощью некоторые вузовские информационные проблемы, приняли решение не только стать ее пользователями, но и ввести в курс обучения студентов.

Совершенно очевидно, что отечественной науке развиваться практически не на чем. Классные специалисты, как известно, на голом месте не родятся. Вот и приходится преподавателям кафедры "Технологий обработки экономической информации" ездить по стране и по кругицам выискивать все те зернышки научных мыслей, которые рождаются то там, то здесь.

Кстати, мы ведь по себе чувствуем, как тает отечественный научный потенциал. Ведь, согласитесь, то, что на крупнейших компьютерных выставках России и Украины мы единственные представляли инструментальный программный товар собственного производства, — далеко не радостный факт.

— А может нам свое и не надо?

— Сторонники подобных мнений глубоко заблуждаются. Во-первых, западное ПО стоит огромных денег: от десятков до сотен тысяч долларов. В СНГ по карману это далеко не многим. В большей степени к Oracle, Informix и им подобным привыкли в России, где производство все-таки находится в более выгодном положении. Но у нас на Украине до этого еще очень далеко. Во-вторых, здесь совершенно другая инфраструктура пользователя. Мы надеемся, что преимущество SWS в том, что она здесь родилась и с момента появления на свет, вдыхая воздух отечественной экономики, приобрела к ней иммунитет. К тому же, она, по-моему, более проста в использовании и требует значительно меньше ресурсов.

— Посмотрим на вопрос с другой стороны. Внедрение средств автоматизации может породить ряд организационных и психологических трудностей. Мне кажется, что чем дальше от центра России, где давно никого не требуется убеждать в необходимости использования средств автоматизации труда (будь то бухгалтер или программист), тем большее сопротивление должны оказывать консервативные структуры. Ведь они моментально потеряют свою управляющую сущность и немедленно перейдут в разряд вспомогательных с успешным вводом систем в эксплуатацию?

- Действительно, все так и обстоит — эта тема для нас больная. Недавно кто-то подсчитал, что

Фото Олега Сердечникова

Борис Панченко, директор "Монолога"

даже в России бухгалтерскими пакетами пользуется менее 10% специалистов. Да и эти пакеты разрабатываются до сих пор в основном прямым кодированием. Какая уж тут автоматизация...

Системы автоматизации производства не внедряются в широком масштабе по причине нескрываемого саботажа со стороны бухгалтерского руководящего состава. И продолжаться это будет до тех пор, пока настоящие хозяева производства не осознают политического значения внедрения таких информационных комплексов для развития производства. Когда это понимаешь, то нелепым кажется вопрос об экономическом эффекте от внедрения АСУ. И при этом так же нелепо звучит фраза, что основной эффект от автоматизации заключается в сокращении двух-трех ни в чем неповинных и (к тому же) очень "дешевых" для производства девочек-расчетчиц ..

По сути, смысл всей нашей работы заключается в тиражировании технической части всех этих мыслей. А мешает нам неуверенность в будущем, отсутствие какой бы то ни было государственной программы относительно средств автоматизации, невыносимые налоги и прочее.

Контактные телефоны (095) 252.48.81., в Киеве — 244.05.63