УДК 004.652

Б.Е. Панченко

Хранилища данных на реляционном каркасе

Предложен новый поход к моделированию темпоральних баз данных. Показано, что классическая реляционная схема хранилища данных «снежинка» – частный случай реляционного каркаса. Введено понятие корректной модификации данных и метаданных.

A new approach to designing temporal data is suggested. It is shown that the classic relation scheme of a data warehouse «snowflake» is a special case of a relational framework. A new definition of correct modification of data and metadata is introduced.

Запропоновано новий підхід до моделювання темпоральних баз даних. Показано, що класична реляційна схема сховища даних «сніжинка» є окремим випадком реляційного каркасу. Введено поняття коректної модифікації даних та метаданих.

Введение. Потребность в моделировании темпоральности данных обусловлена динамическим характером функционирования любой системы. Исторически это было важной причиной разделения всех приложений баз данных (БД) на два типа – оперативные, в которых моделирование времени может быть минимизировано, и исторические, где время имеет принципиальное значение. Эти БД получили соответствующие названия – «оперативные» (или «транзакционные») и «хранилища» (или «аналитические»).

Принцип хранилищ данных (ХД) и их определение предложил У. Инмон [1]. В его подходе ХД – это «предметно-ориентированная, интегрированная, содержащая исторические данные, целостная совокупность данных, предназначенная для поддержки принятия управленческих решений». В [1] дана и классификация ХД. Из описанных типов, нас более всего будет интересовать виртуальное хранилище.

Виртуальное ХД [1] – 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 [2].

Главные достоинства такого подхода – простота и низкая стоимость реализации, единая платформа с источником информации и отсутствие сетевых соединений между источником информации и ХД. При этом хранилище, организованное в соответствии с каркасной схемой БД, избавлено от большинства недостатков, описанных в [3]. Схема такого ХД и приложения, его обслуживающие, могут быть динамически модифицируемы. А высокая производительность такой системы обеспечивается минимизацией операций соединения [4].

Следовательно, и модифицируемость схем БД, и интеграция данных с другими источниками, и отслеживание исторических измерений, и подобие схем БД [5], и гарантии чистоты данных [4], обеспечиваемые ограничениями на домены и ключи, позволяют сделать вывод о том, что объединение свойств оперативной (*OLTP*) и архивной (*OLAP*) БД в одной каркасной схеме возможно.

Постановка задачи

В [6] отмечено, что использование традиционной многомерной модели данных [1, 2] в многомерном ХД (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, поддерживаемый 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

Ключевые слова: схема реляционной базы данных, реляционный каркас, темпоральные данные, хранилища данных, целостность данных, корректность модификаций.

представление данных, целесообразно только в тех случаях, когда объем используемых данных сравнительно невелик, а сама многомерная модель имеет стабильный набор измерений.

Реляционный каркас как шаблон для схем и *OLTP*-БД, и *OLAP*-ХД позволяет иначе решить эти вопросы. Проведем анализ возможности организации ХД с использованием реляционного каркаса.

Причины появления ХД

Авторы [6] считают доказанным утверждение о том, что «… реляционная модель не есть оптимальной с учетом задач анализа, поскольку предполагает высокую степень нормализации, в результате чего снижается скорость выполнения запросов». Однако, как показано в [4, 7], 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 предметной области (ПрО) только посредством запросов к БД. Использование нормализованных по методике Кодда [8] отношений в произвольных запросах действительно требует значительного числа операций соединения, что и приводит к значительным временным затратам.

Высоконормализованные каркасные отношения (актуальные ячейки каркаса [4, 7]), избавляющие пользователя от проектирования большинства оперативных запросов на соединение, доказывают обратное утверждение.

Проанализируем основные мотивы [9] разделения на оперативные БД и аналитические ХД. А также сопоставим их со свойствами каркасной БД.

- «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 в сравнении с объемом таблиц оперативных данных.

- «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 реляционном каркасе, 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.

- «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 [7]) не дает возможности использовать данные от сущностей–объектов в виде омонимов и синонимов.

- «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 [10] 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 каркасная модель основана на булеане связей, проектировщик имеет возможность

* НФ – нормальная форма.

динамически интегрировать в хранимые отношения данные для любого вновь возникшего запроса.

Несложно показать, что даже при появлении между агрегированными шунтирующими атрибутами отношений дополнительных, необусловленных ограничениями на домены и ключи отношения, функциональными зависимостями (ФЗ), нормальность формы отношений снижается лишь до 5НФ, если изначально она была доменно-ключевой (ДКНФ) [4].

Для монотонного агрегирования докажем это утверждение в виде *леммы*. 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 [4], не является детерминантом отношения R.

Доказательство проведем от противного. Пусть существует некоторая функция F такая, что $S = F(R_i(X_i, A_{ij}))$ – также детерминант 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 детерминант отношения $R_i(X_i, A_{ij})$, так как монотонная функция не устраняет иных ФЗ атрибутов–аргументов, в том числе и $A_{ij} \to X_i$. Но такой ФЗ нет в явном виде в особых ограничениях на домены и ключи каркасного отношения [4] доминант. Такая ФЗ должна оговариваться отдельно. 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 каркасных 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.

Интуитивно понятно, что при использовании для агрегации более сложных вычислимых функций [12], нежели монотонные, также не возникнет неконтролируемых детерминантов. Но такое утверждение требует более глубокого исследования, которое выходит за рамки статьи. Однако заметим, что вероятность появления у пользователя потребности в усложненных алгоритмах агрегации крайне мала. Поэтому монотонных арифметических функций достаточно для моделирования большинства *OLAP*-потребностей пользователя.

• «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 учеб-

ников по БД, и семантически-ориентированных схем ХД типа многомерных кубов [1] или реляционной *звезды*/*снежинки* [2]. Более того, потребность пользователей в таком объединении общеизвестна. В [11] эта концепция описана в разделах «Исчезновение отдельных хранилищ данных» и «Хранилища данных в режиме реального времени». А ранее аналогичная потребность была исследована и предложена в [13]. Дальнейшие внедрения этого подхода разными пользователями подтвердили такой вывод.

Реляционный каркас и ХД

Рассмотрим представленную на рис. 1 классическую [2] схему ХД *снежинка*, заимствованную из [6]. Представим ее в предикатной форме. При этом для атомарных сущностей–объектов учтем рекомендации большинства учебников по БД о присвоении не более чем унарного ключа [14, 15]. Однако заметим, 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.

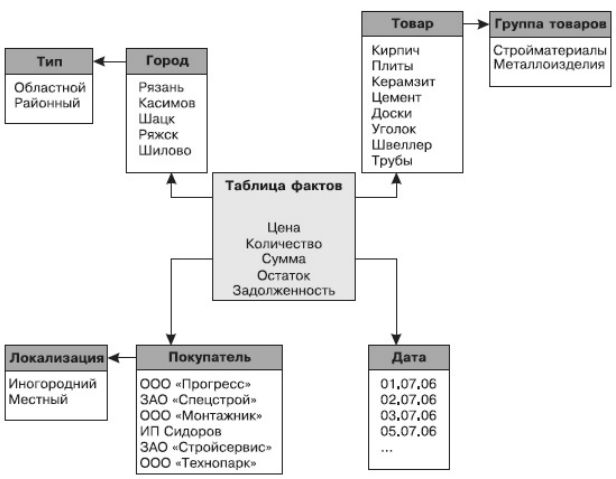

Рис. 1. Схема ХД «снежинка»

Имеем: *ЛОКАЛИЗАЦИЯ* (*КодЛокал*, Наименов), *ПОКУПАТЕЛЬ* (*КодПокуп*, Наименов), *ДАТА* (*НомГода*, *НомМес*, *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 ЗА ГРУППУ МЕСЯЦЕВ НЕКОТОРОГО ГОДА* (*НомГода*, *НомМес*, *КодТипаГор, КодЛокал, КодГор*, *КодТовара*, *КодГрупТов*, Количество, НаСумму), *ПОСТАВКА ТОВАРОВ НЕКОТОРОМУ ПОКУПАТЕЛЮ ЗА ГРУППУ МЕСЯЦЕВ НЕКОТОРОГО ГОДА* (*НомГода*, *НомМес, КодПокупат*, *КодТовара*, *КодГрупТов*, Количество, НаСумму) и др.

Тогда в общем виде отчет произвольной вложенности будет подобен отношению

$$REPORT(X_1 + X_2 + \ldots + X_k, A_1 + A_2 + \ldots + A_n).$$

Таким образом, на базе произвольной совокупности отношений–измерений классической схемы реляционного ХД *снежинка* всегда может быть получено произвольное отношение–факт, подобное актуальной ячейке реляционного каркаса [4].

Основным отличием снежинки от реляционного каркаса есть полнота совокупности отношений. Это дает возможность поддерживать косвенные связи между отношениями по неполному соответствию ключей, моделировать рекурсивные связи сущностей–объектов [10], формировать в схеме БД не одну, а несколько веток иерархических совокупностей отношений, а также поддерживать режим реального времени. Дальше будет показано, что схема ХД, построенная на реляционном каркасе, свободна и от недостатка низкой модифицируемости, присущем МХД [6].

При этом семантически неактуальные кортежи будут удалены. Это очень важное совпадение позволяет сделать следующие выводы.

- Схема реляционного ХД типа *снежинка* – частный случай схемы БД, полученной на реляционном каркасе.
- Схема ХД может быть подобной в смысле [5] схеме оперативной БД, т.е. построена на реляционном каркасе – и оперативные, и аналитические данные могут храниться в БД с

единой схемой и управляться приложениями, синтезированными единым подходом.

• В современных условиях хранить данные в компактном виде нецелесообразно, так как скорость получения отчетов – более важный фактор, нежели цена хранения.

• Причина разделения БД на *OLTP* и *OLAP* заключается не в неудобстве использования, а в неприспособленности нормализованных в соответствии с алгоритмом [8] отношений к *OLAP*-запросам.

Решето, отбраковывающее изоморфные объекты

Для реляционного каркаса принципиально важным есть его совпадение с методом рекурсивного решета. В [16] показано, что это метод комбинаторного программирования, рассматривающий конечное множество и исключающий все элементы этого множества, не представляющие интерес. Метод рекурсивного решета 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. Таким образом, алгоритм решета, отбраковывающего изоморфные объекты, подобен алгоритму исключения неактуальных ячеек каркаса [4, 7] из полной каркасной совокупности.

Сходство метода рекурсивного решета и реляционного каркаса позволяет утверждать, что алгоритм поиска необходимой таблицы по каркасной совокупности будет вычислимым [12].

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* понимаем 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 есть расширение поля для атрибута, которое не меняет тип и значения существующих в нем данных. Однако если модификация атрибута затрагивает значения данных, она будет некорректной.

Добавление нового шунтирующего атрибута, не являющегося 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 этих отношения и атрибута.

Для размещения даты актуализации метаданных и данных проектировщик использует отдельные отношения–маски [10], 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.

Более глубокие вопросы модификации метаданных (схем БД) в статье не рассматриваются. Однако необходимо отметить, что некорректная модификация метаданных (редактирование схем *задним числом*) вызвана или плохим предварительным анализом ПрО, или национальными традициями ее функционирования.

Под *корректной модификацией данных* понимаем изменение значений одного или более атрибутов, не вызывающее потери целостности БД.

Поскольку и приложение, и БД, разработанные на основе каркасной модели, обладают универсальными признаками – и оперативной сис-

темы, и ХД, примером некорректной модификации данных есть операция каскадного удаления. Она вызвана, как правило, плохим анализом ПрО, случайными ошибками проектировщиков или некорректной эксплуатацией системы. Эта операция разрешена только некоторому числу уполномоченных пользователей. 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, подверженных изменениям в ПрО, есть словесные наименования адресных категорий (улиц, переулков, площадей и др.), 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*, Наименов, СокрНаимен, Код АктуальнТипаАдрКатег, 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*, Наименов, СокрНаимен, Код АктНаименАдрКатег, ДатаАктНаименАдрКатег) – 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, Примечан) – составная сущность–объект – «справочник улиц в городах государств» с учетом истории изменений собственных наименований.

Для понимания более полной схемы этого фрагмента БД далее приведем 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 др.

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 описанные недостатки: избыточность данных, многозначность форматов, разночтение терминов [9] и др. 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 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 редактирования значения ключа *КодНаименАдрКатег* в отношении *АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ*() (замены старого значения новым) есть 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 КодАктАдрКатег 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 КодАктАдрКатег 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 [10]. 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*, ДатаДеАктНаименАдрКатег), *ДЕ-*

АКТУАЛИЗИРОВАННЫЕ АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ (*КодГосуд, КодОбл*, *КодРайона, КодГорода, КодТипаАдрКатег*, *КодНаименАдрКатег*, ДатаДеАктАдрКатег), *ДЕАКТУАЛИЗИРОВАННЫЕ АДРЕСА ПРОЖИВАНИЯ 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 описанные отношения–справочники, в фоновом режиме все записи, ссылающихся на старое значение 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*() есть то, что у каждого адресата может быть множество адресов проживания. И некоторая часть из них с определенного момента может быть уже не актуальной. Однако это не означает, что такие записи должны быть удалены из отношения. Принцип ХД подразумевает бессрочное хранение всех исторических фактов в целостном виде. Поэтому описанное 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, соответствующие дополнительные записи в это отношение вносятся вручную.

Таким образом, по факту выполнения подобного редактирования – замены в отношении *АДРЕСНЫЕ КАТЕГОРИИ ГОРОДОВ ГОСУДАРСТВ*() старого значения атрибута *КодНаименАдрКатег* на новое – процедуры каскадных обновлений связанных с ним отношений, как и *АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ*(), могут выполняться автоматически в режиме «квази-реального времени». Под термином «квази» 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, что современные вычислительные системы позволяют пренебрегать временем задержки на выполнение описанных процедур, если такая задержка не является критичной и это не оговорено.

Еще одной важной особенностью такого обновления отношения *АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ*() есть 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 есть определяющим.

Отметим, что иные каркасные отношения–справочники от атомарных и слабых сущностей–объектов фрагмента этой ПрО имеют вид: *ПЕРЕЧЕНЬ ГОСУДАРСТВ* (*КодГосуд*, Наименов, СокрНаимен, КодАктНаименГос,

ДатаАктНаименГос), *СЛОВАРЬ НАИМЕНОВАНИЙ ОБЛАСТЕЙ* (*КодНаименОбл*, Наименов, СокрНаимен, КодАктНаименОбл, ДатаАктНаименОбл), *СЛОВАРЬ НАИМЕНОВАНИЙ ГОРОДОВ* (*КодНаименГор,* Наименов, СокрНаимен, КодАктНаименГор, ДатаАктНаименГор), *СЛОВАРЬ НАИМЕНОВАНИЙ РАЙОНОВ* (*КодНаименРай,* Наименов, СокрНаимен, КодАктНаименРай, ДатаАктНаименРай), *ОБЛАСТИ В ГОСУДАРСТВАХ* (*КодГосуд*, *КодОбл*, *КодНаименОбл*, КодАктОблГос, ДатаАктОблГос, Примечан), *РАЙОНЫ В ОБЛАСТЯХ В ГОСУДАРСТВЕ* (*КодГосуд*, *КодОбл*, *КодРай*, *КодНаименРай*, КодАктРайГос, ДатаАктРайГос, Примечан), *ГОРОДА В ГОСУДАРСТВАХ* (*КодГосуд*, *КодОбл*, *КодРай*, *КодГорода*, *КодНаименГор*, КодАктГроГос, ДатаАктГорГос, Примечан).

Результаты экспериментальных исследований

В работе над статьей проведен эксперимент, основная характеристика которого – время доступа к любому атрибуту отношения–маски исследуемой БД. Как известно, самой ресурсоемкой операцией реляционной алгебры есть соединения отношений [17]. Эта операция применяется при выполнении запросов к БД без использования отношений–масок и более низкой «нормальности» – не выше 3НФ. Особенность каркасной схемы – большинство исследованных связей моделируется в БД 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 выбрана ПрО «Городская поликлиника». Описываемое приложение БД разработано в г. Хмельницком в одной из городских поликлиник с помощью *CASE*-оболочки *SWS* [18] сторонними пользователями этой инструментальной системы. Имеется соответствующий акт внедрения. Как показано в [13, 19], 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.

Предметная область «Городская поликлиника»

На основании алгоритмов нормализации [8] и каркасного синтеза схемы БД [4] были исследованы:

- семантика ПрО;
- каркасная схема БД;
- характеристики доступа к большим объемам данных.

Рассмотрим фрагмент ПрО. Для этого приведем список всех сущностей–объектов с указанием их классификации и схемы. Курсивом выделяются ключевые атрибуты. Подчеркиванием выделены вторичные ключи. Для примера в некоторых отношениях приведены ключи–ссылки на метаданные (выделены иным шрифтом). Ценность столь подробного разделения групп отношений на маски [10] особенно проявляется при поддержке приложения режима реального времени [13, 19], когда превалирующее большинство запросов пользователей проектируется заранее. Тогда на моменте ввода любого данного в отношения, моделирующие оперативное 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 большого объема листинговзапросов, так или иначе зависящих от семантики данных, значительно снижается. С целью экономии места в большинстве схем отношений атрибуты, отвечающие за актуальность записей, не приведены.

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*, Примечан, …) – слабая сущность–объект;

РАЙОНЫ В ОБЛАСТЯХ В ГОСУДАРСТВЕ (*КодГосуд*, *КодОбл*, *КодРайона*, *КодНаименРай*, Примечан) – слабая сущность–объект;

ГОРОДА В ГОСУДАРСТВАХ (*КодГосуд, КодОбл*, *КодРайона*, *КодГорода*, *КодНаименГор*, Примечан) – слабая сущность–объект;

МИКРОРАЙОНЫ ГОРОДОВ (*КодГосуд*, *КодОбл*, *КодРайона*, *КодГорода*, *КодМикроР*, *КодНаименМикроР*, Примечан) – 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*, Примечан) – 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*, *ДатаРожден*, *КодГородаРожден*, Пол, …, Примечан) – атомарная сущность–объект – справочник «люди в государстве»;

АДРЕСА ПРОЖИВАНИЯ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГос*, *КодОбл*, *КодРай*, *КодГорода*, *КодТипаАдрКатег*, *КодНаименАдрКатег*, *НомЗдания*, *Литера*, *СубНомЗдания*, *СубЛитера*, *НомПомещен*, *Литера*) – слабая сущность–объект;

ПЕРСОНАЛЬНЫЕ ТЕЛЕФОНЫ ФИЗЛИЦ (*ИдентНомерФизЛица*, *КодГос*, *НомТелефона, КодТипаТелеф*, Примечан) – 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 СПРАВОЧНИК ПРОФЕССИЙ (*КодПроф*, Наименов) – атомарная сущность–объект;

СПРАВОЧНИК СПЕЦИАЛЬНОСТЕЙ (*КодПроф*, *КодСпец*, Наименов) – слабая сущность–объект;

СПРАВОЧНИК СПЕЦИАЛИЗАЦИЙ В СПЕЦИАЛЬНОСТЯХ (*КодПроф*, *КодСпец*, *КодСпециализ*, Наименов) – слабая сущность–объект;

СПОСОБЫ ПРИОБРЕТЕНИЯ СПЕЦИАЛЬНОСТИ (*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 СОБСТВЕННОСТИ (*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;

РЕКОМЕНДОВАННЫЕ 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.

Штатный реестр в городской поликлинике в одном из городов

УЧАСТКИ В ГОРОДЕ (*КодУчастка*, *КодГос*, *КодОбл*, *КодРайона*, *КодГорода*, Наименов) – 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*, Примечан, *ИдентНомерФизЛица*, *КодГос*) – маска на отношение ФИЗИЧЕСКОЕ ЛИЦО – фильтр по принадлежности к данной поликлинике (в штате, по совместительству или по договору), где *ИдентНомерФизЛица*, *КодГос* – вторичные ключи, фильтрующие отношения, далее аналогичный прием;

СПИСОК ТИПОВ ЗАНЯТОСТИ (*КодТипаЗанятости*, Наименов) – атомарная сущность–объект;

СПИСОК ДОЛЖНОСТЕЙ В ПОЛИКЛИНИКЕ (*КодДолжнПоликл*, Наименов) – маска на отношение *ОБЩЕГОСУДАРСТВЕННЫЙ РЕЕСТР ДОЛЖНОСТЕЙ* – как атомарная сущность–объект;

ШТАТНОЕ РАСПИСАНИЕ ПОЛИКЛИНИКИ (*КодДолжнПоликл*, *НомГода*, *Вакантность*, *НомМес*, Примечан) – составная сущность–объект – текущее состояние списка должностей в поликлинике;

ДОЛЖНОСТЬ СОТРУДНИКА ПОЛИКЛИНИКИ (*ТабНомСотрПоликл*, *КодДолжнПоликл*, *НомГодаЗачисл*, *НомМесЗачисл*, *НомДняЗачисл*, Примечан) – составная сущность–объект – текущее состояние должностей сотрудников в поликлинике;

ВРАЧ (*КодВрача*, Примечан, *ТабНомСотрПоликл*) – маска на отношение–связь *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на отношение *ФИЗИЧЕСКИЕ ЛИЦА*) – фильтр по признаку «*врач*»;

СПИСОК КАБИНЕТОВ ПРИЕМА ВРАЧЕЙ В ПОЛИКЛИНИКЕ (*КодВрача*, *НомКорпуса, НомКабин*, Примечан) – слабая сущность–объект;

УЧАСТКОВЫЙ ВРАЧ (*КодВрача*, Примечан, *ТабНомСотрПоликл*) – маска на отношение *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на маску ВРАЧ) – фильтр по признаку «*тип работы*»;

ПРИНИМАЮЩИЙ ВРАЧ (*КодВрача*, Примечан, *ТабНомСотрПоликл*) – маска на отношение *СОТРУДНИК ПОЛИКЛИНИКИ* (суб-маска на маску ВРАЧ) – фильтр по признаку «*тип работы*»;

ЛЕЧАЩИЙ ВРАЧ (*КодВрача*, Примечан, *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);

СПЕЦИАЛИЗАЦИЯ ВРАЧА (*КодСпециалВрача,* Наименов) – атомарная сущность–объект (возможно со ссылкой на метаданные);

СПЕЦИАЛИЗАЦИИ ВРАЧЕЙ В ПОЛИКЛИНИКЕ (*КодВрача*, *КодСпециалВрача*, Примечан, *КодТаблВрачей*) – составная сущность–объект, где *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;

Прием в городской поликлинике в одном из городов

ПАЦИЕНТ (*КодПациента*, Примечан, *Идент-НомерФизЛица*, *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, Примечан, СуммаКОплате, ...) – составная сущность–объект, связь маски и сущности–объекта – запись или к врачу на прием или на исследование, или к медперсоналу на процедуры или манипуляции и т.д.;

ЗАКАЗ СПЕЦИАЛИСТА НА ДОМ (*КодУчастка*, *КодПациента*, *КодВрача*, *КодСпециалВрача*, *КодУслуги*, *НомГода*, *НомМес*, *НомДня*, *НомОчереди*, Время, Примечан, СуммаКОплате, …) – составная сущность–объект, связь маски и сущности–объекта – заказ на дом или врача, или лаборанта, или медсестры для процедуры или манипуляции и т.д.;

ВИЗИТ К ВРАЧУ СТОМАТОЛОГУ (*КодПациента*, *КодТипаВизита*, *КодВрача*, *НомГода*, *НомМес*, *НомДня*, *НомОчереди*, *КодПредвДиагноза*, Время, Примечан, СуммаКОплате, …) – составная сущность–объект;

НАЗНАЧЕНИЯ ИССЛЕДОВАНИЙ СТОМАТОЛОГОМ (*КодПациента*, *КодВрача*, *НомГода*, *НомМес*, *НомДня*, *КодПредвДиагноза*, *КодИсследов*, Примечан, СуммаКОплате, …) – составная сущность–объект;

РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЙ ДЛЯ СТОМАТОЛОГА (*КодПациента*, *КодВрача*, *НомГода*, *НомМес*, *НомДня*, *КодПредвДиагноза*, *КодИсследов*, *НомЗуба*, *КодОкончДиагноза*, Примечан) – составная сущность–объект;

НАЗНАЧЕНИЯ КУРСА ЛЕЧЕНИЯ СТОМАТОЛОГОМ (*КодПациента*, *КодВрача*, *НомГода*, *НомМес*, *НомДня*, *КодОкончДиагноза*, *КодПроцедуры*, Примечан, СуммаКОплате, …) – составная сущность–объект;

РЕЦЕПТ СТОМАТОЛОГА (*КодПациента*, *КодВрача*, *НомГода*, *НомМес*, *НомДня*, *КодОкончДиагноза*, *КодЛекарства*, Примечан) – составная сущность–объект;

ЗАПИСИ В ЛИЧНОЕ ДЕЛО ПАЦИЕНТА (*КодУчастка*, *КодПациента*, *КодВрача*, *КодСпециалВрача*, *НомГода*, *НомМес*, *НомДня*, *КодОкончДиагноза*, *КодРезульт*, Примечан) – составная сущность–объект.

Маски как составные сущности–обекты (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, …);

ПОЛИКЛИНИКА ЗА МЕСЯЦ (*НомГода, НомМес*, СуммЧислоЧасов, ОбщЧислоПац, ОбщСуммаРеализации, …);

РЕГИСТРАТУРА ЗА МЕСЯЦ (*НомГода*, *НомМес*, ОбщЧислоПац, …);

РЕНТГЕНОГРАФИЯ ЗА МЕСЯЦ (*НомГода*, *НомМес*, *КодРентгеногр*, ЧислоПац, ОбщСуммаРеализации, …);

ФЛЮРОГРАФИЯ ЗА МЕСЯЦ (*НомГода*, *НомМес*, *КодФлюрогр*, ЧислоПац, ОбщСуммаРеализации, …);

УЗИ ЗА МЕСЯЦ (*НомГода*, *НомМес*, *Код УЗИ*, 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 др.), будут аналогично сформированы по схеме: *СПЕЦИАЛИЗАЦИЯ ЗА МЕСЯЦ* (*НомГода*, *НомМес*, *КодСпециализирУслуги*, ЧислоПац, ОбщСуммаРеализации, …).

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 др.

В приведенной схеме БД синтезировано 125 каркасных отношений. Скорость доступа к данным по типовым запросам повысилась на несколько порядков в сравнении со схемой, разработанной в соответствии с алгоритмом нормализации Кодда [8].

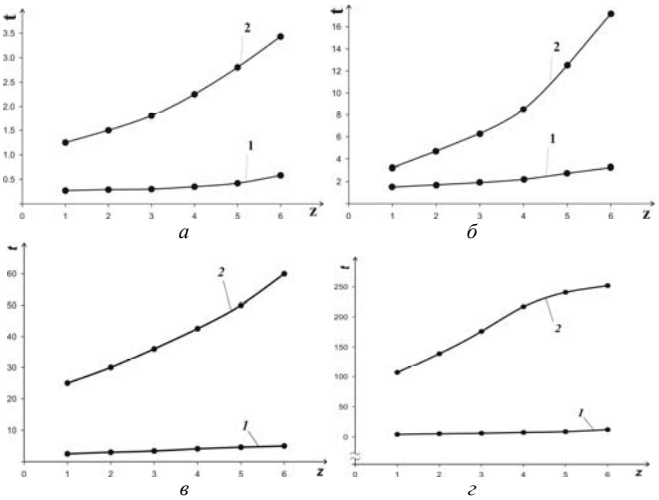

Рис. 2. Изменение времени доступа к данным при: *а* – четырех отношениях; *б* – восьми отношениях; *в* – 12 отношениях; г – 16 отношениях

На рис. 2,*а* показаны графики, иллюстрирующие увеличение времени (млсек) доступа к данным при получении одной записи из увеличивающихся пачек записей при запросе на соединение четырех 3НФ-отношений (кривая 2). И заметно меньше времени на индексную выборку этой же записи из одного квартарного каркасного отношения (кривая 1) при таком же увеличении числа записей. Кривые совпали с результатами работы [7]. Это подтверждает подобие схем БД разных ПрО еще и при исследовании скорости доступа к данным.

Рис. 2,*б* иллюстрирует еще меньший рост времени доступа к декарному каркасному отношению (кривая 1) с увеличением числа обрабатываемых записей. И значительный рост времени выполнения этого же запроса при соединении 8-ми отношений в 3НФ (кривая 2).

Рис. 2,*в* и 2,*г* иллюстрируют значительные отличия в росте времени при 12-ти и 16-ти арных отношениях. Для более выразительной иллюстрации ось абсцисс рис. 2,*г* несколько смещена в отрицательном направлении. Номера кривых имеют тот же смысл. При этом все ре-

зультирующие отношения аналогично [7] формировались в среднем до 150–200 кортежей.

Заметим, что вид каркасной диаграммы описанной ПрО полностью соответствует аналогичной диаграмме из работы [7]. Поэтому с целью экономии места диаграмму не приводим. 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 [18, 19].

Заключение. Таким образом, каркасная модель данных позволила обнаружить совпадение описанного подхода с классическим результатом модели ХД *снежинка* [2]. Однако все перечисленное дает возможность более общего построения схемы БД, обладающей как *OLTP*-свойствами, так и возможностями ХД. Унификация и типизация каркасной схемы БД позволяет также минимизировать потребность в ресурсоемких операциях соединения в большинстве запросов к БД, чем существенно упрощает настройку приложения. Приводятся результаты численного эксперимента доступа к БД.

1. *Inmon W.H.* Building the Data Warehouse. – New York: John Willey & Sons, 2002. – 412 p.
2. *Kimball R.* The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses / Ibid., 1996. – 491 p.
3. *Стулов А.С.* Особенности построения информационных хранилищ. – М.: Открытые системы, 2003. – № 4. – С. 82–89. – *http://www.osp.ru/os*/2003/ 04/182942/
4. *Панченко Б.Е.* Каркасное проектирование доменно-ключевой схемы реляционной базы данных // Кибернетика и системный анализ. – 2012. – № 3. – С. 174–187.
5. *Панченко Б.Е.* Об алгоритме синтеза реляционного каркаса. Постановка задачи и формализация // Компьютерная математика. – 2012. – № 1. – С. 84–93.
6. *Орешков В.И.*, *Паклин Н.Б.* Бизнес-аналитика: от данных к знаниям. – СПб.: Питер, 2009. – 624 с.
7. *Панченко Б.Е.* К вопросу о модифицируемости и безаномальности схемы реляционной базы данных // Проблемы программирования. – 2012. – № 1. – С. 281–288.
8. *Codd E.F.* The Relational Model For Database Management, V. 2, Reading Mass. – New York: Addison-Wesley Publ. Co, 1990. – 538 p.
9. *Кузнецов С.Д.*, *Артемьев В.А.* Обзор возможностей применения ведущих СУБД для построения хранилищ данных (*DataWarehouse*). – // *http://citforum.ck.ua/database/kbd*98/*glava*15.*shtml*
10. *Панченко Б.Е.* Рекурсивне связи и темпоральность в реляционном каркасе – маски сущностей–объектов // Проблемы управления и информатики. – 2013. – № 2. – С. 92–104.
11. *Хоббс Л.*, *Хилсон С.*, *Лоуренд Ш.* Разработка и эксплуатация хранилищ данных (*Oracle* 9*iR*2). – М.: Кудиц-образ, 2004. – 586 с.
12. *Роджерс Х.* Теория рекурсивных функций и эффективная вычислимость. – М.: Мир, 1972. – 624 с.
13. *Панченко Б.Е.*, *Гайдабрус В.Н.*, *Церковицкий С.Л.* Сетевые вычислительные комплексы // Компьютеры плюс программы. – 1994. – спец. вып. – С. 30–37.
14. *Харрингтон Д.Л.* Проектирование реляционных баз данных. – М.: Лори, 2006. – 231 с.
15. *Хернандес М.Д.*, *Вьескас Д.Л. SQL*-запросы для простых смертных. Там же. – 2003. – 460 с.
16. *Рейнгольд Э.*, *Нивергельт Ю.*, *Део Н.* Комбинаторные алгоритмы. Теория и практика. – М.: Мир, 1980. – 476 с.
17. *Ульман Д.*, *Уидом Д.* Основы реляционных баз данных. – М.: Лори, 2006. – 374 с.
18. *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.: РосАПП, 1994. – 2 с.
19. *Панченко Б.Е.* Исследования доменно-ключевой схемы реляционной базы данных // Кибернетика и системный анализ. – 2012. – № 6. – С. 157–172.

Поступила 12.11.2012
Тел. для справок: +38 044 526-3603, 285-3346,
+38 067 449-3970 (Киев)
E-mail: *pr-bob@ukr.net*