УДК 004.657+004.422.632

ПРИМЕНЕНИЕ МНОГОМЕРНЫХ ДИНАМИЧЕСКИХ МАССИВОВ ПРИ ОБРАБОТКЕ ПРОИЗВОДСТВЕННО-ЭКОНОМИЧЕСКОЙ ИНФОРМАЦИИ

Поляков А.И.
Донецкий национальный технический университет, кафедра АСУ
e-mail: pai@kita.dgtu.donetsk.ua

Abstract. Polyakov A.I. *In article is made short analysis OLTP and OLAP-systems, is determined circle of the problems, solved these system. Efficiency of the use multivariate dynamic array is shown on example at decision of the problems of the shaping the reports to different difficulty with provision for in them given changeable at time. Maded comparison proposed design method of the reports with methods, using in its base SQL -a requests.*

При рассмотрении текущего состояния информационно-программных комплексов, используемых на различных крупных промышленных предприятиях Донецкой области, четко прослеживается тенденция на слияние так называемых ***OLTP-приложений*** (On-Line Transaction Processing (OLTP) - оперативная обработка транзакций) с ***OLAP-приложениями*** (On-Line Analitical Processing (OLAP) - оперативная аналитическая обработка данных).

Основной чертой ***OLTP-приложений*** является максимально допустимая нормализация данных и как следствие большое количество нормализированных отношений. Основная функция подобных приложений заключается в выполнении большого количества коротких транзакций. Практически все запросы к базе данных в OLTP-приложениях состоят из команд вставки, обновления, удаления. Запросы на выборку в основном предназначены для предоставления пользователям возможности выбора из различных справочников. Большая часть запросов, таким образом, известна заранее еще на этапе проектирования системы. Таким образом, критическим для OLTP-приложений является скорость и надежность выполнения коротких операций обновления данных. Чем выше уровень нормализации данных в OLTP-приложении, тем оно, как правило, быстрее и надежнее. Отступления от этого правила могут происходить тогда, когда уже на этапе разработки известны некоторые часто возникающие запросы, требующие соединения отношений и от скорости выполнения которых существенно зависит работа приложений. В этом случае можно пожертвовать нормализацией для ускорения выполнения подобных запросов.

Исходя из академической трактовки ***OLAP-приложения*** представляют собой совокупность систем поддержки принятия решений (Decision Support System - DSS), хранилищ данных (Data Warehouse), систем интеллектуального анализа данных (Data Mining). В целом такие приложения предназначены для нахождения зависимостей между данными или определением изменения показателей объекта с течением времени или расчета этих показателей на заданное время с учетом того, что оно могло уже пройти или еще и не наступить. OLAP-приложения оперируют с большими массивами данных, уже накопленными в OLTP-приложениях, взятыми их электронных таблиц или из других источников данных. Ранее такие системы характеризовались следующими признаками [1]:

- добавление в систему новых данных происходит относительно редко крупными блоками;
- данные, добавленные в систему OLAP-приложения, обычно никогда не удаляются;
- перед загрузкой данные проходят различные процедуры "очистки", связанные с тем, что в одну систему могут поступать данные из многих источников, имеющих

различные форматы представления для одних и тех же понятий, данные могут быть некорректны, ошибочны;
- запросы к системе являются нерегламентированными и, как правило, достаточно сложными. Очень часто новый запрос формулируется аналитиком для уточнения результата, полученного в результате предыдущего запроса;
- скорость выполнения запросов важна, но не критична.

Многие разработчики рекомендуют представлять данные OLAP-приложений в виде одного или нескольких гиперкубов, измерения которого представляют собой справочные данные, а в ячейках самого гиперкуба хранятся собственно данные.

Физически гиперкуб может быть построен на основе специальной ***многомерной модели данных*** (***MOLAP - Multidimensional OLAP***) или построен средствами реляционной модели данных (***ROLAP - Relational OLAP***).

Если рассмотреть проблему нормализации данных при построении информационных моделей систем обработки производственно-экономической информации, можно сказать, что в системах OLAP, использующих реляционную модель данных (ROLAP), данные целесообразно хранить в виде слабо нормализованных отношений. Большая избыточность и связанные с ней проблемы тут не страшны, т.к. обновление происходит только в момент загрузки новой порции данных. При этом основным критерием работоспособности и адекватности данной модели является соответствие ее компонент основному требованию теоремы Хеза: корректность процедуры нормализации - декомпозиция без потерь.

Рассматривая на современном этапе задачи автоматизированной обработки производственно-экономической информации можно сделать вывод, что так как средства вычислительной техники предоставляют возможности хранения больших объемов информации и их скоростной обработки, а данные из OLTP-приложений необходимы для продуктивной работы соответствующих АРМов специалистов за длительный период (за год или более, а в некоторых случаях и задачах даже десятилетия), то необходимость в специальном разделении и расслоении приложений на OLTP и OLAP-приложения в среднесрочной и долгосрочной перспективе отпадает. При этом вместо представления данных в виде многомерной модели данных MOLAP (в виде гиперкубов) можно использовать соответствующие функции или процедуры обработки данных реляционной модели OLTP–приложения, возвращающих информацию в виде многомерных динамических массивов, структура которых соответствует заданным формам и отчетам. В этом случае также необходимо учитывать время как один из основополагающих параметров. То есть информация об объекте или субъекте учета всегда изменяется во времени и, если осуществляется ее отображение, то оно объективно и адекватно определяет состояние объекта только для какого-то заданного временного среза, поэтому для отображения информации за длительный период необходимо учитывать то, что отображается свойство объекта, которое за заданный временной интервал могло сколько угодно раз поменяться. Исходя их этого следует отметить, что всегда существует вероятностная ошибка отображения достоверности информации, которая напрямую зависит от вида отчета и продолжительности интервала отчета или его подинтервальных разбиений.

Рассмотрим для примера функцию формирования отчета о динамике изменений численности работников по профессиям по участкам предприятия, представленного в таблице 1.

Для того чтобы сформировать данный документ необходимо использовать данные как минимум двух рабочих таблиц (таблицы основных данных личной карточки работника и таблицы данных о перемещениях работников), двух таблиц нормативно-справочной информации (справочника подразделений и справочника профессий) и одной таблицы с соответствующими данными интересующих дат, на которые собственно и будет

производится расчет. Обобщенно схема данных данной задачи будет иметь вид, представленный на рисунке 1.

Таблица 1. Вид формируемого отчета.

Участок					
Шифр	Наименование				
Профессия		Дата			
Шифр	Наименование	01.01.xxxx	01.02.xxxx	…	31.12.xxxx
Код уч.1	Участок 1				
Проф.1					
Проф.2					
Проф.N					
Итого по участку 1					
Код уч.К	Участок К				
Проф.1					
Проф.2					
Проф.М					
Итого по участку К					
Итого по предприятию					
в разрезе профессий:					
Проф.1					
Проф.2					
Проф.Р					

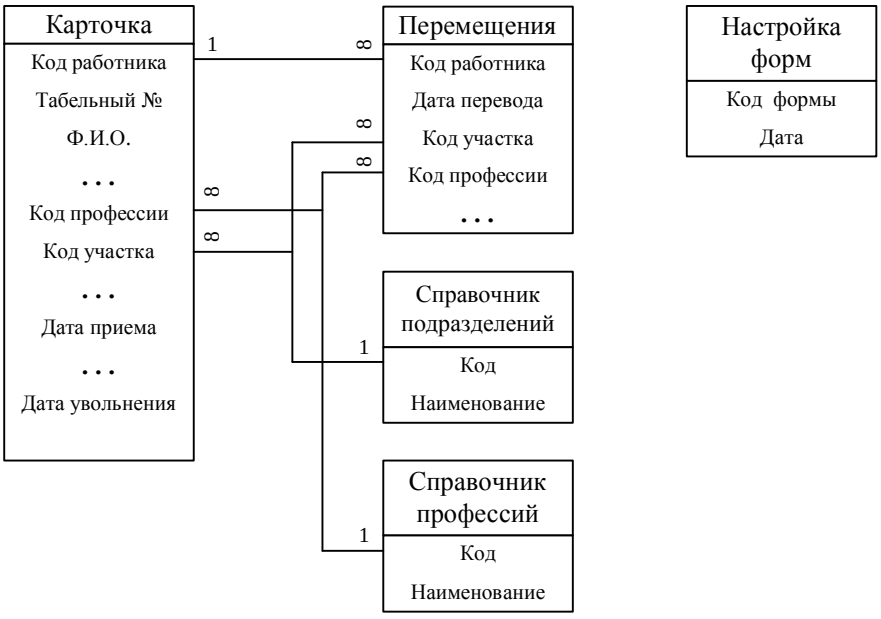

Рисунок 1. Схема данных для рассматриваемого примера

С применением многомерных динамических массивов данная задача решается в один проход по таблице карточек работников с подстановкой в сформированный динамический массив или его подмассивы информации, соответствующей состоянию свойств (доменов: профессия-участок) рассматриваемого объекта (работника) заданным временным интервалам. Если рассматривать этот процесс в рамках предприятия, то с объектами учета за

заданный период могут происходить (но не обязательно происходят) только следующие виды изменений: прием; увольнение; перемещения. Данные события и отображаются в рассматриваемом примере.

Для этого вначале с соответствии с заданным (определенным по текущим установкам) количеством расчетных интервалом определяется структура шаблонного одномерного массива, определяющего свойства одного объекта, будь-то участка или профессии по участку, или профессии в целом по предприятию вида:

$$ar\_sh:=\{ d_1, d_2, \ldots, d_n \}, \quad (1)$$

где d_1, d_2, …, d_n – количественные показатели рассчитываемого параметра, первоначально равные нулю.

Общий массив данных по участкам и профессиям из рассматриваемого примера будет выглядеть следующим образом:

$$ar:= \{ uch, ar\_sh, \{ prof, ar\_sh \} \}, \quad (2)$$

где uch – код участка, ar_sh – сформированный массив-шаблон, prof – код профессии.

Массив данных по профессии в целом по предприятию имеет вид:

$$ar\_prof:= \{ prof, ar\_sh \}, \quad (3)$$

При работе с данными массивами необходимо чтобы используемый язык программирования помимо элементарных функций работы с динамическими массивами поддерживал и функции необходимые для обработки данных массивов такие как: сортировка, поиск элемента в массиве по заданному отношению.

Так же для определения показателя «дата-участок-профессия» необходимо использовать соответствующую триггерную процедуру таблицы перемещений, возвращающую на заданную дату по заданному коду работника (или табельному номеру работника) код участка и код подразделения, на котором работник находился в тот момент времени. При этом, чтобы не повторять в цикле для одного работника несколько запросов с разными датами данную процедуру можно преобразовать и передавая в нее массив соответствующих дат, получать из нее набор информации, соответствующий местонахождению работника на предприятии в соответствующие моменты времени [2].

Выдача информации из сформированных массивов в виде таблиц отчета с подстановкой вместо кодов участков и профессий их наименований из таблиц справочников с указанием необходимого направления сортировки реализована в большинстве специализированных языков программирования, относящихся к категории СУБД.

Однако реализация такого, казалось бы не сложного примера, на языках SQL или с помощью SQL-запросов затруднительна. Это происходит потому, что набор инструкций SQL по формированию выборок данных реализуема для таблиц, находящихся в нормализированных формах высокого порядка, так как для слабой нормализации отношений построение произвольных расчетных выборок, структура которых зависит от временного фактора не представляется возможным. То есть вначале получается выборка или набор выборок данных, соответствующий различным временным градациям (изменениям) контролируемых свойств, а затем с помощью аппарата других программных продуктов или языков программирования необходимо еще и получить адекватную заданному отчету выборку расчетных данных.

В рассматриваемом примере вместо классического представления информации в виде многомерной модели данных MOLAP (совокупности так называемых гиперкубов) и дальнейшей ее сложной обработки, данные действия реализованы с использованием простейших многомерных массивов. Это многократно упрощает принципы понимания работы системы, увеличивает ее производительность и уменьшает временные затраты на

составления программного кода, делает программный код доступным и понятным, по существу приближая его логику, к примитивным общеобразовательным примерам.

Рассмотрим пример из задачи оперативного автоматизированного складского учета.

Необходимо сформировать за период оборотно-сальдовую ведомость движения материалов на складах предприятия с формированием сводных проводок по корреспондирующим балансовым счетам. Примерный вид данная ведомости представлен в таблице 2.

Таблица 2. Вид типовой оборотной ведомости движения материалов на складах предприятия.

Склад	Наименование склада				
Цена	Наименование материала		Единица измерения	Номенклатурный номер	
Движение	Дата	№ документа	Поставщик / Подразделение	Количество	Сумма
Итого по складу:					Сумма
в разрезе бухгалтерских проводок:					
проводка (дебит – кредит)					Сумма
в разрезе основных счетов аналитического учета:					
Счет	Остаток на начало	Приход	Расход	Остаток на конец	
в разрезе поставщиков:					
Поставщик					Сумма
в разрезе подразделений и участков:					
Участок					Сумма
Итого по предприятию:					Сумма
в разрезе бухгалтерских проводок:					
проводка (дебит – кредит)					Сумма
в разрезе основных счетов аналитического учета:					
Счет	Остаток на начало	Приход	Расход	Остаток на конец	
в разрезе поставщиков:					
Поставщик					Сумма
в разрезе подразделений и участков:					
Участок					Сумма

Для формирования данной выборки используются данные следующих таблиц:

Рабочие таблицы:

- таблица карточек складского учета материалов;
- таблица остатков материалов (на начало и конец периода).

Справочные данные:

- справочник сырья и материалов;
- справочник видов операций;
- справочник подразделений;
- справочник поставщиков (потребителей);
- справочник единиц измерения.

Схема данных для заданного примера представлена на рисунке 2.

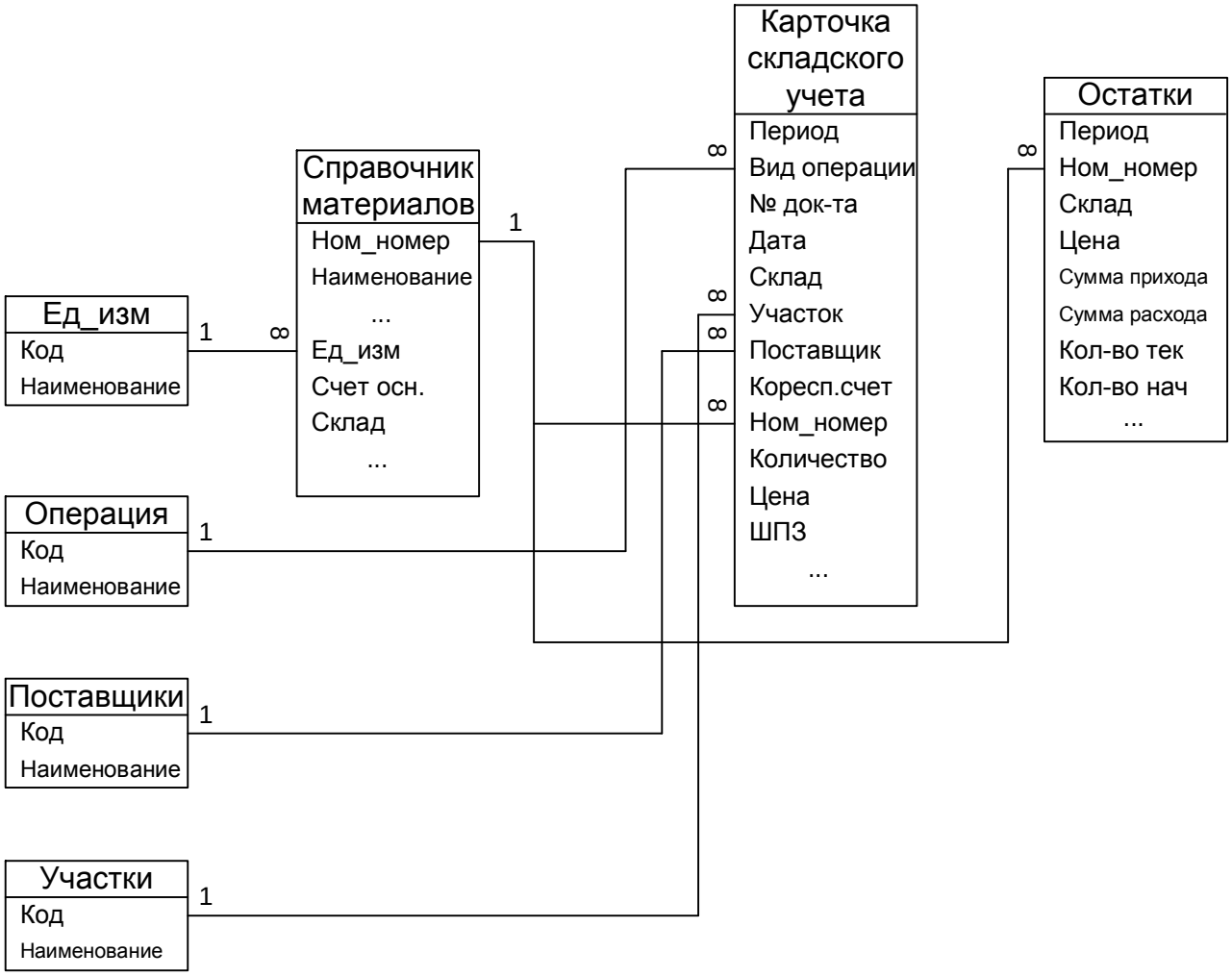

Рисунок 2. Примерная схема данных для формирования оборотно-сальдовой ведомости

Использование многомерных динамических массивов для формирования оборотно сальдовой ведомости движения материалов на складах предприятия заключается в следующем:

1. Формирование массива начинается из таблицы «Остатки» для заданного периода. Для каждого склада открывается новый элемент массива.
2. В элементе массива склад для каждого номенклатурного номера формируются из таблицы «Карточка» массивы содержащие информацию по элементам прихода и расхода материала за период, формируются сводные массивы проводок и распределения сумм по элементам затрат и объектам затрат (участкам или заказам, или по обоим критериям вместе – по заказам в разрезе участком и наоборот).
3. Вид получаемого в данном случае массива следующий:

ar_oborot:={ skl, { ksm, cena, kol_n, kol_k, { dat_prih, kol_prih, k_post }, { dat_ras, kol_ras, k_uch } }, {prov, sum_prov}, {schet, ost_n, s_prih, s_ras, ost_k }, { k_post, s_prih1 }, { k_uch, s_prih2}, {k_uch, s_rash} }, (4)

где skl – код склада, ksm – номенклатурный номер материала, cena – цена материала, kol_n – количество материала на складе по данной цене на начало периода, kol_k – количество материала на складе по данной цене на конец периода, dat_prih – дата

прихода материала на склад, kol_prih – количество прихода, k_post – код поставщика, dat_ras – дата списания материала со склада, kol_ras – количество расхода, k_uch – код участка, на который производится отпуск материала, prov – бухгалтерская проводка, sum_prov – общая сумма бухгалтерской проводки по складу, schet – основной счет, ost_n – сумма остатков материалов на счете на начало отчетного периода, s_prih – сумма прихода материалов на счет за отчетный период, s_ras – сумма расхода материалов со счета за отчетный период, ost_k – сумма остатков материалов на счете на конец отчетного периода, s_prih1 – сумма прихода по поставщику, s_prih2 – сумма возврата материалов на склад с участков, s_rash – сумма расхода материалов на участков.

4. Итоговые суммы по предприятию можно либо объединить в один массив итоговых сумм, либо использовать для каждой категории свои расчетные массивы аналогичные приведенным в пункте 3 (за исключением разбиений на склады):

ar_pred:={ {prov, sum_prov}, {schet, ost_n, s_prih, s_ras, ost_k },
{ k_post, s_prih1 }, { k_uch, s_prih2}, {k_uch, s_rash} }, (5)

Как и в предыдущем примере у приведенной реализации по сравнению с обработкой данных при помощи SQL-запросов многократно (больше чем на порядок) повышается скорость выполнения при снижении требований к необходимым объемам оперативной памяти. Отпадает необходимость построения сложно подчиненных реляционных отношений, улучшается восприятие алгоритма реализации.

Выводы:

1. На основании проведенных обследований некоторых крупных промышленных предприятий, разработки и внедрения для них автоматизированных систем обработки производственно-экономической информации можно утверждать, что в настоящее время происходит интенсивное слияние OLTP и OLAP-систем обработки данных.
2. Проведенные исследования показали, что использование языка SQL-запросов не упрощает работу систем, а наоборот повышает требования к аппаратной части комплекса обработки данных и помимо языка SQL требует наличие и использование какого-либо из языков программирования для построения не только интерфейсной части системы, но и для более глубокой обработки данных при построении элементарных расчетных форм.
3. Анализ структур действующих автоматизированных систем обработки производственно-экономической информации позволяет сделать вывод, что для эффективной работы OLAP-систем требуется слабая нормализация данных информационной модели, которая позволяет уменьшать требования к объемам хранения информации, повысить качество ее обработки и уменьшить зависимость скорости обработки информации от качества построения исходной информационной модели.

Литература:

1. Пушников А.Ю. Введение в системы управления базами данных. Изд-е Башкирского ун-та. - Уфа, 1999.
2. Свідоцтво про реєстрацію авторського права на твір № 18651. “Комп’ютерна програма “АРМ працівника відділу кадрів. АРМ керівника”. 14.11.2006г.