КОМПЬЮТЕРНЫЕ НАУКИ

УДК 681.3016

ОБ ОДНОМ ВОПРОСЕ ПОСТРОЕНИЯ СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ НА ОСНОВЕ РАСПРЕДЕЛЕННЫХ СИСТЕМ ОБРАБОТКИ ДАННЫХ

ТАНЯНСКИЙ С.С., РУДЕНКО Д.А., ЯКОВЛЕВА Е.С.

Описываются технология и средства реализации системы поддержки принятия решений с применением методологии крупномасштабных проектов по созданию корпоративных баз данных. Рассматривается ряд вопросов, связанных с оптимизацией выполнения запросов в системах управления реляционными базами данных.

1. Введение

Одним из основных факторов успеха в управлении и в повседневной жизни является скорость и качество принимаемых решений. Неудивительно, что попытки формализовать или автоматизировать процесс принятия решений начались практически сразу с появлением вычислительных машин и продолжаются до сих пор.

В области информационных технологий всегда существовали два взаимодополняющих друг друга направления развития [1]:

– системы обработки данных (СОД), ориентированные на операционную обработку данных;

– системы поддержки принятия решений (СППР), ориентированные на анализ данных.

До недавнего прошлого, когда говорилось о росте числа реализаций информационных систем, прежде всего имелись в виду системы, ориентированные исключительно на операционную обработку данных. И такое опережающее развитие одного из направлений вполне объяснимо.

На первых этапах автоматизации требуется навести порядок именно в процессах повседневной рутинной обработки данных, на что и ориентированы традиционные СОД. Более того, системы СППР являются в определенном смысле вторичными по отношению к ним, так как любая продукция, прежде чем попасть к потребителю, должна быть сначала произведена. И прежде чем заниматься анализом данных, необходимо их иметь (произвести). А именно это и является одной из функций СОД.

Сегодня СОД, реализованные на самой различной основе, исправно работают, порождают и пополняют многочисленные многотомные электронные архивы.

В результате огромные архивные массивы, накопленные за годы эксплуатации СОД и содержащие самую разнообразную жизненно важную для организации информацию, без предварительной доработки и согласования не могут быть непосредственно использованы в задачах анализа.

После того как традиционная СОД реализована и начинает функционировать, она становится таким же самостоятельным объектом реального мира, как и любой производственный процесс. А данные, которые являются одним из конечных продуктов такого производства, обладают ровно теми же свойствами и характеристиками, что и любой промышленный продукт: сроком годности, местом складирования (хранения), совместимостью с данными из других СОД, рыночной стоимостью, транспортабельностью, комплектностью и т. д.

СППР в зависимости от данных, с которыми они работают, можно разделить на оперативные, предназначенные для немедленного реагирования на текущую ситуацию, и стратегические – основанные на анализе большого количества информации из разных источников с привлечением сведений, содержащихся в системах, аккумулирующих опыт решения проблем.

Чем больше информации вовлечено в процесс принятия решений, тем обоснованнее решение. Информация, на основе которой принимается решение, должна быть достоверной, полной, непротиворечивой и адекватной. Поэтому при проектировании СППР возникает вопрос о том, на основе каких данных эти системы будут работать. Качество оперативных решений обеспечивается тем, что данные выбираются непосредственно из информационной системы управления предприятием (или из базы данных предприятия), которая адекватно отражает его состояние на данный момент времени. Ранние версии СППР в качестве исходных использовали относительно небольшой объем агрегированных данных, легко поддающихся проверке на достоверность, полноту, непротиворечивость и адекватность [2].

По мере развития и совершенствования алгоритмов принятия решений СППР сталкиваются с проблемами, вызванными замедлением процессов построения отчетов на основе соответствующих решений в связи с накоплением больших объемов информации. Кроме того, с развитием межкорпоративных связей требуется вовлекать в процесс анализа данные из внешних источников, не связанных напрямую с производственными процессами и потому не входящих в систему управления предприятием.

2. Технология и средства реализации СППР

Применяемая в СППР традиционная технология подготовки интегрированной информации на основе запросов и отчетов стала неэффективной из-за резкого увеличения количества и разнообразия исход-

ных данных. Кроме того, постепенное накопление в базе данных (БД) предприятия информации для принятия решений и последующий их анализ стали отрицательно сказываться на оперативной работе с данными.

Решение таких задач требует выполнения функций предварительной подготовки и хранения данных для СППР на основе информации из системы управления БД предприятия, а также информации из посторонних источников, которые в достаточном количестве имеются на рынке информации. В свою очередь это требует новых технологических решений при разработки и внедрении специализированных структур данных и создания СППР на основе распределенных систем управления базой данных (СУБД).

Уровень сложности систем управления респеределенными БД часто измеряется степенью независимости поведения пользователя от требований, выдвигаемых распределенной архитектурой. В идеальном случае пользователь вообще не должен ощущать распределенности данных: все функции по распределению операций доступа к БД в различных абонентских пунктах возлагаются на систему. Однако способ физического распределения данных влияет на общую производительность вычислительной системы [2].

Самым критичным из ресурсов СППР является время, и если не определить кто, когда, зачем и как будет принимать решения, какое влияние то или иное решение оказывает на результат, какие решения отнести к оперативным, а какие к стратегическим и т.д., то предприятие обрекает себя на неизбежное отставание в конкурентной борьбе.

Основное назначение модели предприятия - определение и формализация данных, необходимых в процессе принятия решения [1]. В этом случае модель представления данных является организационно-функциональным срезом модели системы, а при ее разработке необходимо учитывать:

– распределение пользователей системы: географическое, организационное, функциональное;

– доступ к данным: объем данных, необходимый для анализа; уровень агрегированности данных; их источники (внешние или внутренние), описание информации, совместно используемой различными функциональными группами предприятия;

– аналитические характеристики системы: измерения данных, основные отчеты, последовательность преобразования аналитической информации, степень предопределенности анализа, существующие или находящиеся в стадии разработки средства анализа.

Для СППР на основе распределенных СОД нормальным считается итерационный, а иногда и параллельный характер моделирования, при котором возврат на предыдущую стадию - обычное явление. Это связано с необходимостью выделения всех требуемых данных для произвольных запросов, в связи с чем следует составить исчерпывающий перечень необходимых данных и построить схему их связей. Следующий этап построения СППР связан с пониманием того, в каком виде и на каких аппаратных и программных платформах размещать структуру данных [2].

В самом простом варианте для СОД используется та модель данных, которая лежит в основе транзакционной системы. Если, как это часто бывает, эта система функционирует на основе реляционной СУБД, самой сложной задачей становится выполнение произвольных запросов, поскольку невозможно заранее оптимизировать структуру БД так, чтобы все запросы работали эффективно.

3. Распределенные системы управления данными

Сегодня распределенные реляционные СУБД (РСУБД) стали доминирующим промышленным решением при реализации самых разнообразных СОД. Они обеспечивают приемлемое время отклика при произвольной выборке отдельных записей и их небольших групп. А реальные объемы БД, с которыми они могут работать, превышают сотни гигабайт. Однако исходно ориентированные на реализацию систем операционной обработки данных, распределенные РСУБД оказались менее эффективными в задачах аналитической обработки.

Прежде всего это связано с наличием достаточно жестких ограничений, накладываемых существующей реализацией языка SQL. Аналитические запросы получаются весьма громоздкими, а многие функции обработки приходится выносить в заранее написанные пользовательские приложения [3]. И если вопрос о том, что громоздкость конструкций – это серьезный недостаток, достаточно спорный (сегодня практически никто не пишет непосредственно на SQL, а соответствующие конструкции автоматически генерируются средствами клиентского инструментария), то ограничения SQL реально существуют, и их так просто не обойти.

Примером такого реально существующего ограничения является предположение о том, что данные в реляционной базе не упорядочены (или, более точно, упорядочены случайным образом) [4]. Но выполнение большинства аналитических функций (например построение прогноза), наоборот, невозможно без предположения об упорядоченности данных. Естественно, при использовании реляционной БД имеется возможность после выборки данных из базы выполнить их сортировку и затем строить прогноз. Но это потребует дополнительных затрат времени на сортировку, которая должна будет проводиться каждый раз при обращении к этой функции, и, самое главное, такая функция может быть определена и применена только в пользовательском приложении, но не может быть встроенной функцией языка SQL.

Кроме того, для обеспечения приемлемого времени ответа, при использовании распределенной РСУБД, нужно уже на этапе проектирования знать обо всех возможных типах запросов, необходимых срезах и уровнях агрегации данных.

Основой традиционного реляционного подхода является нормализация (декомпозиция) таблиц БД,

подразумевающая устранение избыточности в первичных ключах и транзитивных зависимостей между реквизитами, образующими таблицу. Это не только минимизирует суммарный объем данных в БД, но и решает проблемы, связанные с различного рода аномалиями, возникающими при удалении и модификации данных в ненормализованных таблицах. И хотя в процессе нормализации утрачиваются семантические связи, существующие между реквизитами, это не особенно критично для традиционных СОД. Те немногие связи, которые необходимы для реализации конкретного приложения, известны заранее и легко реализуются с помощью механизма внешних ключей. Более критична эта проблема для аналитических систем. Здесь обычно даже нельзя заранее определить, какие связи между различными реквизитами будут применяться более часто, а какие не будут использоваться вообще.

Недостаток традиционных РСУБД заключался в том, что в качестве основного и часто единственного механизма, обеспечивающего быстрый поиск и выборку отдельных строк в таблице (или в связанных через внешние ключи таблицах), обычно используются различные модификации индексов, основанных на B-деревьях [4]. Но такое решение эффективно только при обработке небольших групп записей и высокой интенсивности модификации данных в БД. В аналитических системах ввод и выборка данных осуществляется большими порциями. А данные, после того как они попадают в БД, остаются неизменными в течение длительного периода времени. И здесь более эффективным оказывается хранение данных в форме частично денормализованных таблиц, в которых для увеличения производительности могут храниться не только детализированные, но и предварительно вычисленные агрегированные значения, а для навигации и выборки – использоваться специализированные, основанные на предположении о малоизменчивости и малоподвижности данных в БД, методы адресации и индексации.

И все же реляционные БД остаются наиболее подходящими в технологии реализации информационных систем уровня предприятия.

4. Выбор оптимального плана соединений в реляционных СУБД

Реляционные языки запросов обеспечивают высокоуровневый, декларативный интерфейс для доступа к данным, хранимым в реляционных БД. Подсистема выполнения запросов реализует набор физических операций. На вход каждой операции поступают один или несколько потоков данных, а на выходе формируется один общий поток.

Абстрактным представлением такого выполнения является дерево физических операций, в котором дуги представляют потоки данных между операциями. Будем использовать термины "дерево физических операций" и "план выполнения запроса" (или просто план) в одном и том же смысле [3].

При выборе эффективного плана выполнения запроса необходимо исходить из возможного пространства таких планов. Задача оптимизации нетривиальна, потому что для заданного SQL-запроса может существовать большое число возможных деревьев операций:

– алгебраическое представление запроса может быть преобразовано во многие другие логически эквивалентные алгебраические представления, например, Join (Join (A,B),C) = Join (Join (B,C),A);

– для алгебраического представления может существовать много планов выполнения запроса, реализующих алгебраическое выражение; например, в системе БД обычно поддерживается несколько алгоритмов соединения.

Кроме того, пропускная способность, или время ответа системы при выполнении этих планов может весьма различаться. Поэтому выбор плана выполнения имеет критическое значение. Таким образом, к оптимизации запросов можно относиться как к сложной поисковой проблеме. Для того чтобы ее решить, необходимо определить:

– пространство планов (пространство поиска);

– метод оценки стоимости, чтобы можно было оценить каждый план в каждом пространстве поиска;

– алгоритм перебора, который может осуществлять поиск в пространстве планов выполнения.

Каждая из этих задач нетривиальна, из-за чего выбор оптимального плана является достаточно сложным.

Пространство поиска оптимального плана может состоять из деревьев операций, которые соответствуют линейной и попарной последовательности операций соединения; например, последовательности Join (Join (Join (A,B),C),D) и Join (Join (A,B),Join (C,D)) проиллюстрированы на рис. 1.

Такие последовательности логически эквивалентны, поскольку соединения обладают свойствами ассоциативности и коммутативности.

Присвоим оценочную стоимость любому частичному или полному плану в пространстве поиска и определим оценочный размер потока данных для вывода каждой операции плана. Такие оценки могут базироваться на следующих характеристиках:

– набор статистик, поддерживаемых для отношений и индексов, например, число страниц данных в отношении, число страниц в индексе, число различных значений в столбце;

– формулы для оценки прогнозирования размера выходного потока данных. Например, размер вывода

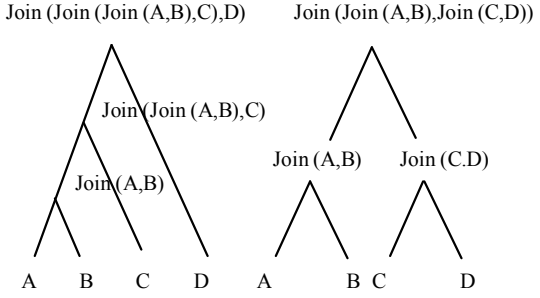

Рис. 1. Пример линейного и попарного соединений

соединения оценивается путем перемножения размеров отношений-операндов;

– формулы для оценки стоимости расходов центрального процессора при выполнении запроса для каждой операции. В этих формулах принимаются во внимание статистические свойства входных потоков данных операции, существующие методы доступа к данным входных потоков и т. д.

Для алгоритма перебора планов соединения будем использовать метод динамического программирования [3]. Суть подхода такого программирования основывается на предположении, что оценочная модель удовлетворяет принципам оптимальности. Более точно – предполагается, что для получения оптимального плана запроса Q, состоящего из k соединений, достаточно рассматривать только оптимальные планы для подзапросов Q, которые состоят из (k-1) соединений, и расширять эти планы дополнительным соединением. Другими словами, для определения оптимального плана выполнения Q не требуется рассматривать не самые оптимальные планы для подзапросов Q с (k-1) соединениями. Соответственно, основанный на динамическом программировании алгоритм перебора представляет запрос Q как множество соединяемых отношений $\{R_1,...,R_n\}$.Алгоритм перебора работает снизу вверх. В конце j-го шага алгоритм производит оптимальные планы для всех подзапросов размера j.

При получении оптимального плана для подзапроса, включающего (j+1) соединение, рассматриваются все возможные способы его построения путем расширения планов, полученных на j-м шаге. Например, оптимальный план для $\{R_1, R_2, R_3, R_4\}$ получается его выбором с наименьшей стоимостью из оптимальных планов для:

1) Join $\{\{R_1, R_2, R_3\}, R_4\}$;

2) Join $\{\{R_1, R_2, R_4\}, R_3\}$;

3) Join $\{\{R_1, R_3, R_4\}, R_2\}$;

4) Join $\{\{R_2, R_3, R_4\}, R_1\}$.

Остальные планы для $\{R_1, R_2, R_3, R_4\}$ можно отбросить. Подход динамического программирования работает быстрее, чем простой перебор, поскольку требуется перебрать O(n2n-1) планов вместо O(n!).

При традиционном выполнении запроса с группировкой вычисление его компоненты предшествует группировке. Эти преобразования применимы к SQL-запросам с SELECT DISTINCT. Выполнение операции GROUP BY потенциально может привести к значительному сокращению числа кортежей, поскольку для каждого раздела отношения, выделяемого операцией группировки, она генерирует только один кортеж. Поэтому в некоторых случаях при выполнении сначала группировки стоимость соединения может быть существенно уменьшена. Более того, при наличии подходящего индекса операция группирования может быть выполнена недорого.

Рассмотрим дерево запроса на рис. 2, а.

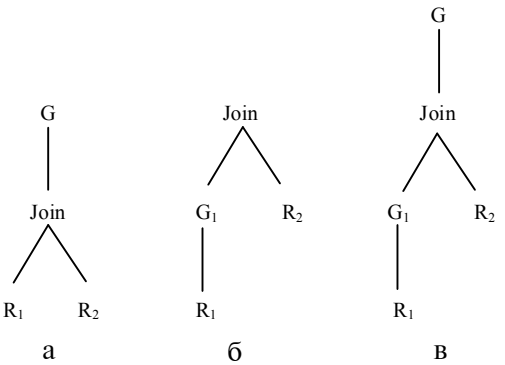

Рис. 2. Группировка и соединение

Пусть отношения R_1 и R_2 соединяются по внешнему ключу и все столбцы агрегирования G взяты из R_1, а в состав набора столбцов группировки входит внешний ключ R_2. Для такого запроса рассмотрим соответствующее дерево операций на рис. 2,б, где G_1=G. В этом дереве завершающее соединение может сократить набор потенциальных разделов R_1, созданных G_1, но не может повлиять на разделы и агрегаты, вычисляемые G_1 на этих разделах, поскольку каждый кортеж R_1 соединяется не более чем с одним кортежем R_2. Следовательно, можно опустить группировку вниз, как показано на рис. 2,б, и сохранить эквивалентность для произвольных агрегатных функций без побочного эффекта. На рис. 2,в показан пример, где операция группировки выполняется поэтапно.

Например, предположим, что в запросе, дерево операций которого изображено на рис. 2,а, агрегатные функции вычисляются только на столбцах R_1. В этих случаях введенный оператор группировки G_1 разделяет отношение по столбцам R_1 и вычисляет агрегатные функции на этих разделах. Однако на рис. 2,а могут потребоваться истинные разделы, чтобы объединить несколько разделов, образованных G_1, в один (отображение много-к-одному). Это обеспечивает оператор группирования G.

Такое поэтапное вычисление может быть полезным для уменьшения стоимости соединения по причине сокращения объема данных операцией группировки G_1. Для возможности такой поэтапной агрегации требуется, чтобы агрегатные функции обладали тем свойством, что Agg ($R_1 U R_2$) можно вычислить на основе Agg (R_1) и Agg (R_2).

Например, чтобы вычислить общий объем продаж для всех продуктов каждого отдела, можно использовать преобразование с рис. 2,в для выполнения ранней агрегации и получения общего объема продаж для каждого продукта. Затем потребуется еще одна группировка, чтобы сложить объемы продаж всех продуктов, относящихся к одному отделу.

5. Заключение

В статье рассмотрены только некоторые фундаментальные вопросы выбора оптимального плана соединения при создании запросов в реляционных БД. Одним из интересных направлений является то, в котором допускается генерация полных планов при условии доступности информации о времени выпол-

нения. Кроме того, открытой остается проблема учета других ресурсов (в особенности памяти) при определении планов выполнения запросов. Технология оптимизации в объектно-ориентированных системах также является важной областью, заслуживающей отдельного обсуждения. Кроме того, когда БД стали использоваться СППР, появилось интересное направление работы в связи с нечеткими (неточными) запросами. Существующее повышенное внимание к СППР побудило также проведение работ в области расширений SQL.

Разработка эффективных и корректных преобразований SQL-запросов является трудной задачей из-за сложности отыскания надежных метрик оценок. Таким образом, несмотря на многие годы работы, существенные проблемы остаются открытыми. Однако для того, чтобы внести вклад в области выбора оптимального плана запросов, необходимо понимание существующих подходов.

Эффект от правильной организации, стратегического и оперативного планирования развития производства трудно заранее оценить в цифрах, но очевидно, что он может превзойти затраты на реализацию СППР. Однако эффект обеспечивает не сама система, а люди, которые с ней работают. Современные аналитические системы не являются системами искусственного интеллекта, их цель – своевременно обеспечить сотрудника всей информацией, необходимой для принятия решений. А какая информация будет затребована и какое решение будет принято на ее основе, зависит только от конкретного человека.

Литература: 1. *Сахаров А.А.* Концепции построения и реализации информационных систем, ориентированных на анализ данных // Системы управления базами данных. 1996. №4. С.55-70. 2. *Львов В.* Создание систем поддержки принятия решений на основе хранилищ данных // Системы управления базами данных. 1997. №3. С.30-40. 3. *Чаудхари С.* Методы оптимизации запросов в реляционных системах // Системы управления базами данных. 1998. №3. С.22-46. 4. *Мейер Д.* Теория реляционных баз данных. М.: Мир, 1987. 608 с.

Поступила в редколлегию 09.10.99

Рецензент: д-р техн. наук Зацеркляный Н.М.

Танянский Сергей Станиславович, канд. техн. наук, доцент кафедры информационных систем и технологий Университета внутренних дел МВД Украины. Научные интересы: проектирование и поддержка баз данных с неопределенными значениями. Адрес: Украина, 61170, Харьков, ул. Блюхера, 22, кв. 159, тел. 65-47-52, 50-36-31, e-mail: k_infsis@adm.univd.kharkov.ua.

Руденко Диана Александровна, канд. техн. наук, ассистент кафедры применения ЭВМ ХТУРЭ. Научные интересы: распределенные структуры данных, теория нечетких множеств. Адрес: Украина, 61148, Харьков, ул. Родниковая, 3, кв. 181, тел 16-86-66.

Яковлева Елена Сергеевна, курсант факультета управления и информатики Университета внутренних дел МВД Украины. Научные интересы: теория принятия решений в условиях неопределенности. Адрес: Украина, 61058, Харьков, ул. Ромена Роллана, 7, кв. 40, тел. 43-75-14.

УДК 007.001.362; 681.327.12.001.362

СИНТЕЗ НОРМАЛИЗАТОРОВ АФФИННЫХ ПРЕОБРАЗОВАНИЙ НА БАЗЕ ОДНОМЕРНЫХ КОРРЕЛЯЦИЙ

ПУТЯТИН Е.П., ЯКОВЛЕВА Е.В., ЛУЦИВ В.В.

Разрабатывается алгоритм нормализации плоских изображений путем применения методов одномерной нормализации. Для этого исследуется вопрос о формировании разложения матрицы центроаффинного преобразования в суперпозицию матриц поворота, неоднородного изменения масштаба и еще одного поворота, возможно в сочетании с зеркальным отражением. Изучаются свойства преобразования неоднородного изменения масштаба.

Под нормализацией изображения будем понимать процедуру компенсации геометрических искажений, связывающих эталонные и реальные изображения, т.е. нахождение параметров того геометрического преобразования, которое позволит привести реальное изображение к эталонному виду [1].

Рассмотрим общий случай, когда заранее не известен конкретный вид геометрического преобразования, формирующего разницу между эталонным и реальным изображениями. При этом будем считать, что реальное изображение находится под воздействием достаточно общей аффинной группы геометрических преобразований G_a. Решив проблему центрирования обрабатываемого изображения, можем полагать, что на реальное изображение воздействует центроаффинная группа G^c_a, которая описывается вещественной квадратной матрицей A с определителем, не равным нулю.

Для проведения нормализации необходимо определить параметры нормализующей матрицы, которой является обратная матрица A^{-1}.

Известно [2], что преобразования с матрицей A можно представить в виде произведения ортогональной $A_{орт}$ и симметричной A_c матриц:

$$A=A_{орт} * A_c. \qquad (1)$$

Покажем, как формируется симметричная и ортогональная матрица. Будем рассматривать случай преобразования на плоскости. Пусть A – любая невырожденная матрица, описывающая невырожденное линейное преобразование в некотором ортонормированном базисе $\{e_1,e_2\}$. Тогда преобразование, описываемое матрицей A^*A, где A^* – транспонированная матрица, будет симметричным.

Известно [3, 4], что симметричное преобразование всегда приводится к диагональному виду путем