УДК 004.047:681.3.01

Л.А. Пономаренко, В.А. Филатов, С.С. Танянский

Минимизация вычислительных затрат при доступе к данным в распределенных экономических системах поддержки принятия решений

Предложена методика выбора оптимального плана соединений при выполнении запросов в экономических системах принятия решений, позволяющая повысить эффективность использования информации путем уменьшения количества переборов

The method for selecting an optimal plan compounds to execute queries in economic decision support systems is proposed. This method enables more efficient use of information by reducing the amount of searches.

Запропоновано методику вибору оптимального плану з'єднання при виконанні запитів в економічних системах прийняття рішень, яка дозволяє підвищити ефективність використання інформації шляхом зменшення кількості переборів.

Введение. Развитие информационных систем, а также совершенствование алгоритмов принятия решений на основе агрегированных данных, систем принятия решений столкнулись с проблемами, вызванными необходимостью обеспечить растущие информационные потребности. Традиционная технология подготовки интегрированной информации на основе запросов и отчетов стала неэффективной из-за резкого увеличения количества и разнообразия исходных данных. В настоящее время необходимо существенно сократить время выполнения информационных запросов в распределенных экономических системах. Решение такой задачи может быть связано с уменьшением количества переборов при выборе оптимального плана соединений элементов экономической системы.

Состояние проблемы

Экономическая информационная система (ЭИС) представляет собой совокупность организационных, технических, программных и информационных средств, объединенных в единую систему с целью сбора, хранения, обработки информации, предназначенной для выполнения функций управления. Экономическая информационная система связывает объект и систему управления между собой и с внешней средой через информационные потоки:

– из внешней среды в систему управления;

– из системы управления во внешнюю среду;

– из системы управления на объект управления;

– от объекта управления в систему управления [1].

Кроме того, ЭИС накапливает и перерабатывает поступающую учетную информацию и имеющиеся нормативы в аналитическую информацию, служащую основой для прогнозирования развития экономической системы.

К обработке информации в ЭИС предъявляются следующие требования:

– полнота и достаточность информации для реализации функций управления;

– своевременность предоставления информации;

– обеспечение необходимой степени достоверности информации в зависимости от уровня управления;

– экономичность обработки информации: затраты на обработку данных не должны превышать получаемый эффект;

– адаптивность к изменяющимся потребностям пользователей.

В соответствии с характером обработки информации в ЭИС на различных уровнях управления экономической системой (оперативном, тактическом и стратегическом) выделяются следующие типы информационных систем:

– системы обработки данных;

– информационные системы управления;

– системы поддержки принятия решений.

Системы обработки данных предназначены для учета и оперативного регулирования хозяйственных операций, подготовки стандарт-

ных документов для внешней среды (счетов, накладных, платежных поручений и другого). Эти задачи имеют итеративный, регулярный характер, выполняются непосредственными исполнителями хозяйственных процессов и связаны с оформлением и пересылкой документов в соответствии с четко определенными алгоритмами. Результаты выполнения хозяйственных операций вводятся в базу данных.

Информационные системы управления ориентированы на тактический уровень управления: среднесрочное планирование; анализ и организацию работ в течение некоторого времени; составление производственных программ. Для данного класса задач характерны регламентированность (периодическая повторяемость) формирования результатных документов и четко определенный алгоритм решения задач. Решение подобных задач предназначено для руководителей различных служб предприятий (отделов материально-технического снабжения, сбыта, цехов и т.д.). Задачи решаются на основе накопленной базы оперативных данных [2].

Системы поддержки принятия решений (СППР) используются в основном на верхнем уровне управления (руководства фирм, предприятий, организаций), имеющих стратегическое долгосрочное значение в течение года или нескольких лет. К таким задачам относятся формирование стратегических целей, планирование привлечения ресурсов, источников финансирования, выбор места размещения предприятий и т.д. Реже задачи класса СППР решаются на тактическом уровне, например, при выборе поставщиков или заключении контрактов с клиентами. Задачи СППР имеют, как правило, нерегулярный характер [3].

Для задач СППР свойственны недостаточность имеющейся информации, ее противоречивость и нечеткость, преобладание качественных оценок целей и ограничений, слабая формализованность алгоритмов решения.

Чем больше информации вовлечено в процесс принятия решений, тем более обоснованным может стать принятое 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 ЭИС (или из базы данных ЭИС), адекватно отражающей состояние предприятия на данный момент времени.

Исследовать достоинства и недостатки совместной обработки данных, используя консолидированный подход, включая онтологии и семантические сети, можно по работам Клещева А.С., Артемьева И.Л. [4] и Цикритзиса Д., Лоховски Ф. [5].

Среди работ по проектированию экономических информационных систем можно выделить труды Смирнова Г.Н., Тельнова Ю.Ф., Вендрова А.М. [6], где подробно рассматриваются методы и технологии построения информационных систем с учетом их применения в области экономического управления предприятием.

Среди авторов, исследовавших вопросы доступа к данным с использованием современных методов и средств, можно выделить работу Зельцера М. [7], в которой рассмотрены задачи создания единого информационного пространства на основе универсальной базы данных (БД) со средствами обработки данных, применяющими единый метод описания и манипулирования данными.

Постановка задачи

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 [3].

Таким образом, для эффективного решения экономических задач необходимо уделить особое внимание построению системы анализа информации на внутреннем уровне, а именно, на

уровне обработки данных. В связи с этим, в статье рассматриваются методы корректной интеграции распределенных данных с учетом минимизации данных в промежуточных результатах.

Технология и средства реализации систем поддержки принятия решений

Применяемая в СППР традиционная технология подготовки интегрированной информации на основе запросов и отчетов стала неэффективной из-за резкого увеличения количества и разнообразия исходных данных. Кроме того, постепенное накопление в БД предприятия информации для принятия решений и последующего их анализа 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 сторонних источников, которые в достаточном количестве доступны на рынке информации. В свою очередь, это требует новых технологических решений при разработке и внедрении специализированных структур данных и создания СППР на основе распределенных систем управления базой данных (СУБД) [8].

Уровень сложности систем управления распределенными 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.

Самым критичным из ресурсов СППР есть 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 так далее, то предприятие обрекает себя на неизбежное отставание в конкурентной борьбе.

Основное назначение модели предприятия – определение и формализация данных, необходимых в процессе принятия решения. 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 агрегирования данных, источники данных (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 систем обработки данных (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.

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

Распределенные системы управления данными

Сегодня распределенные РСУБД стали доминирующим промышленным решением при реа-

лизации самых разнообразных СОД. Они обеспечивают приемлемое время отклика при произвольной выборке отдельных записей и небольших групп записей. Однако распределенные РСУБД, исходно ориентированные на реализацию систем операционной обработки данных, оказались менее эффективными в задачах аналитической обработки.

Прежде всего это связано с наличием достаточно жестких ограничений, накладываемых существующей реализацией языка *SQL*, при которой аналитические запросы получаются весьма громоздкими, а многие функции обработки приходится выносить в заранее написанные пользовательские приложения. И если вопрос о том, что громоздкость конструкций – 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 есть предположение о том, что данные в реляционной базе не упорядочены (или, более точно, упорядочены случайным образом). 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 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*.

В то же время для обеспечения приемлемого времени ответа при использовании распределенной РСУБД, необходимо уже на этапе проектирования знать обо всех возможных типах запросов, необходимых срезах и уровнях агрегации данных.

Основа традиционного реляционного подхода – нормализация (декомпозиция) таблиц БД, подразумевающая устранение избыточности в первичных ключах и устранение транзитивных зависимостей между реквизитами, образующими таблицу. Это позволяет не только минимизировать 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, необходимые для реализации конкретного приложения, известны заранее и легко реализуются с помощью механизма внешних ключей. Более критична эта проблема для аналитических систем. Здесь 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. Но такое решение оказывается эффективным 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. 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.

Итак, реляционные БД остаются наиболее подходящей технологией для реализации информационных систем уровня предприятия.

Оптимальный план соединений в РСУБД

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 есть 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.

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 [9]:

- 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 баз данных 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 решить эту проблему, 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 задача выбора оптимального плана достаточно сложна.

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.

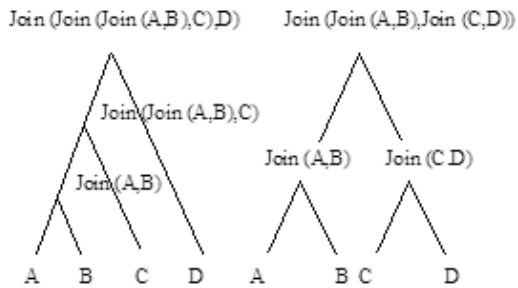

Рис. 1. Пример линейного и попарного соединений

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 [10]:

- 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 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 используем метод динамического программирования. Суть подхода динамического 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, состоящих из $(k-1)$ соединений, и расширять эти планы дополнительным соединением. Другими словами, для определения оптимального плана выполнения Q не требуется рассматривать заведомо не 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 находит оптимальные планы для всех подзапросов размера j.

Для получения оптимального плана для подзапроса, включающего $(j+1)$ соединение, рассматриваются все возможные способы построения плана путем расширения планов, полученных на j-м шаге. Например, оптимальный план для $\{R_1, R_2, R_3, R_4\}$ получается выбором плана 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 существенно быстрее, чем простой перебор, поскольку требуется перебрать $O(n2n-1)$ планов вместо $O(n!)$.

При традиционном выполнении запроса с группировкой вычисление компоненты запроса 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 группировки эффективность операции соединения может быть существенно повышена, особенно при наличии подходящего индекса.

Рассмотрим структуру дерева запроса, представленную на рис. 2.

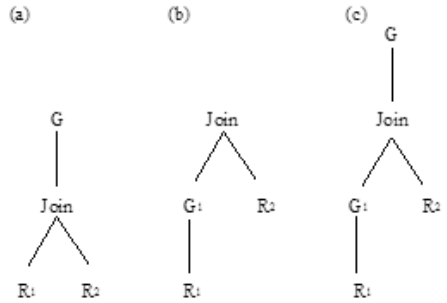

Рис. 2. Группировка и соединение

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,*b*, 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,*b*, и сохранить эквивалентность для произвольных агрегатных функций без побочного эффекта. На рис. 2,*c* показан пример, где операция группировки выполняется поэтапно.

Например, предположим, что в запросе, дерево операций которого показано на рис. 2,*a*, 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,*a* могут потребоваться истинные разделы, чтобы объединить несколько разделов образованных G_1, в один раздел (отображение *много–к–одному*). Это обеспечивает оператор группировки G.

Такое поэтапное вычисление может быть полезным для уменьшения вычислительной сложности соединения по причине сокращения объема данных операцией группировки G_1. Для возможности такой поэтапной агрегации требуется, чтобы агрегатные функции обладали тем свойством, что $Agg(R_1UR_2)$ 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 (рис. 2,*c*) 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.

Заключение. В статье рассмотрены только некоторые фундаментальные вопросы выбора оптимального плана соединения при создании запросов в реляционных БД. Одним из интересных направлений есть то, в котором допускается генерация полных планов при условии доступности информации о времени выполнения. Кроме того, открытой остается проблема учета других ресурсов (в особенности памяти) при определении планов выполнения запросов. Технология оптимизации в объектно-ориентированных системах также важная область, заслуживающая 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 представляется трудной задачей в связи со сложностью 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 есть системами искусственного интеллекта, их цель своевременно обеспечить сотрудника всей информацией, необходимой для принятия решений. А какая информация будет запрошена и какое решение будет принято на ее основе, зависит только от конкретного человека.

1. *Смирнова Г.Н.*, *Тельнов Ю.Ф.* Проектирование экономических информационных систем. Ч. 1. – М.: МЭСИ, 2004. – 223 с.
2. *Пономаренко Л.А.*, *Филатов В.А.*, *Цыбульник Е.Е.* Агентные технологии в задачах поиска информации и принятия решений // УСиМ. – 2003. – № 1. – С. 36–41.
3. *Львов В.* Создание систем поддержки принятия решений на основе хранилищ данных // Системы управления базами данных. – 1997. – № 3. – С. 30–40.
4. *Клещев А.С.*, *Артемьева И.Л.* Математические модели онтологий предметных областей. Ч. 1. Существующие подходы к определению понятия «онтология» // Научно-техническая информация, Сер. 2. – 2001. – № 2. – С. 20–27.
5. *Цикритзис Д.*, *Лоховски Ф.* Модели данных // М.: Финансы и статистика, 1985. – 343 с.
6. *Вендров А.М.* Проектирование программного обеспечения экономических информационных систем // Там же, 2002. – 352 с.
7. *Зельцер М.* За пределами реляционных баз данных: доступ к базам данных не ограничивается возможностями *SQL* // Data engineering. – 2005. – **3**, N 3 – P. 21–29.
8. *Танянский С.С.*, *Филатов В.А.*, *Чапланова Е.Б.* Модель «сущность–связь» в задачах представления объектно-реляционных свойств предметной области // УСиМ. – 2011. – № 3. – С. 73–78.
9. *Чаудхари С.* Методы оптимизации запросов в реляционных системах // Системы управления базами данных. – 1998. – № 3. – С. 22–46.
10. *Мейер Д.* Теория реляционных баз данных // М.: Мир, 1987. – 608 с.

Поступила 18.02.2014
Тел. для справок: +38 044 229-2195, 050 198-1695 (Киев)
E-mail: *laponomarenko@ukr.net*