Введение

В наше время без систем управления базами данных не обходится практически ни одна организация, особенно среди тех, которые традиционно ориентированы на взаимодействие с клиентами. Банки, страховые компании, авиа- и прочие транспортные компании, сети супермаркетов, телекоммуникационные и маркетинговые фирмы, организации, занятые в сфере услуг и другие – все они собирают и хранят в своих базах гигабайты данных о клиентах, продуктах и сервисах. Ценность подобных сведений несомненна. Такие базы данных называют операционными или транзакционными, поскольку они характеризуются огромным количеством небольших транзакций, или операций записи-чтения. Компьютерные системы, осуществляющие учет операций и собственно доступ к базам транзакций, принято называть системами оперативной обработки транзакций (OLTP – On-Line Transactional Processing) или учетными системами.

Учетные системы настраиваются и оптимизируются для выполнения максимального количества транзакций за короткие промежутки времени. Обычно отдельные операции очень малы и не связаны друг с другом. Однако каждую запись данных, характеризующую взаимодействие с клиентом (звонок в службу поддержки, кассовую операцию, заказ по каталогу, посещение Web-сайта компании и т.п.) можно использовать для получения качественно новой информации, а именно для создания отчетов и анализа деятельности фирмы.

Набор аналитических функций в учетных системах обычно весьма ограничен. Схемы, используемые в OLTP-приложениях, осложняют создание даже простых отчетов, так как данные чаще всего распределены по множеству таблиц, и для их агрегирования необходимо выполнять сложные операции объединения. Как правило, попытки создания комплексных отчетов требуют больших вычислительных мощностей и приводят к потере производительности.

Кроме того, в учетных системах хранятся постоянно изменяющиеся данные. По мере сбора транзакций суммарные значения меняются очень быстро, поэтому два анализа, проведенные с интервалом в несколько минут, могут дать разные результаты. Чаще всего, анализ выполнятся по окончании отчетного периода,

иначе картина может оказаться искаженной. Кроме того, необходимые для анализа данные могут храниться в нескольких системах.

Некоторые виды анализа требуют таких структурных изменений, которые недопустимы в текущей оперативной среде. Например, нужно выяснить, что произойдет, если у компании появятся новые продукты. На живой базе такое исследование провести нельзя. Следовательно, эффективный анализ редко удается выполнить непосредственно в учетной системе.

Системы поддержки принятия решений обычно обладают средствами предоставления пользователю агрегатных данных для различных выборок из исходного набора в удобном для восприятия и анализа виде. Как правило, такие агрегатные функции образуют многомерный (и, следовательно, нереляционный) набор данных (нередко называемый гиперкубом или метакубом), оси которого содержат параметры, а ячейки — зависящие от них агрегатные данные – причем храниться такие данные могут и в реляционных таблицах. Вдоль каждой оси данные могут быть организованы в виде иерархии, представляющей различные уровни их детализации. Благодаря такой модели данных пользователи могут формулировать сложные запросы, генерировать отчеты, получать подмножества данных.

Именно это и обусловило интерес к системам поддержки принятия решений, ставших основной сферой применения OLAP (On-Line Analytical Processing, оперативная аналитическая обработка, оперативный анализ данных), превращающей “руду” OLTP-систем в готовое “изделие”, которое руководители и аналитики могут непосредственно использовать. Этот метод позволяет аналитикам, менеджерам и руководителям “проникнуть в суть” накопленных данных за счет быстрого и согласованного доступа к широкому спектру представлений информации.

Трудно найти в компьютерном мире человека, который хотя бы на интуитивном уровне не понимал, что такое базы данных и зачем они нужны. В отличие от традиционных реляционных СУБД, концепция OLAP не так широко известна, хотя загадочный термин “кубы OLAP” слышали, наверное, почти все. Что же такое Online Analytical Processing?

OLAP – это не отдельно взятый программный продукт, не язык программирования и даже не конкретная технология. Если постараться охватить OLAP во всех его проявлениях, то это совокупность концепций, принципов и требований, лежащих в

основе программных продуктов, облегчающих аналитикам доступ к данным.

Для начала мы выясним, зачем аналитикам надо как-то специально облегчать доступ к данным. Дело в том, что аналитики – это особые потребители корпоративной информации. Задача аналитика – находить закономерности в больших массивах данных. Поэтому аналитик не будет обращать внимания на отдельно взятый факт, что в четверг четвертого числа контрагенту Чернову была продана партия черных чернил – ему нужна информация о сотнях и тысячах подобных событий. Одиночные факты в базе данных могут заинтересовать, к примеру, бухгалтера или начальника отдела продаж, в компетенции которого находится сделка. Аналитику одной записи мало – ему, к примеру, могут понадобиться все сделки данного филиала или представительства за месяц, год. Заодно аналитик отбрасывает ненужные ему подробности вроде ИНН покупателя, его точного адреса и номера телефона, индекса контракта и тому подобного. В то же время данные, которые требуются аналитику для работы, обязательно содержат числовые значения – это обусловлено самой сущностью его деятельности.

Централизация и удобное структурирование – это далеко не все, что нужно аналитику. Ему ведь еще требуется инструмент для просмотра, визуализации информации. Традиционные отчеты, даже построенные на основе единого хранилища, лишены одного – гибкости. Их нельзя “покрутить”, “развернуть” или “свернуть”, чтобы получить желаемое представление данных. Конечно, можно вызвать программиста, и он сделает новый отчет достаточно быстро – скажем, в течение часа. Получается, что аналитик может проверить за день не более двух идей. А ему (если он хороший аналитик) таких идей может приходить в голову по нескольку в час. И чем больше “срезов” и “разрезов” данных аналитик видит, тем больше у него идей, которые, в свою очередь, для проверки требуют все новых и новых “срезов”. Вот бы ему такой инструмент, который позволил бы разворачивать и сворачивать данные просто и удобно! В качестве такого инструмента и выступает OLAP.

Хотя OLAP и не представляет собой необходимый атрибут хранилища данных, он все чаще и чаще применяется для анализа накопленных в этом хранилище сведений.

Оперативные данные собираются из различных источников, очищаются, интегрируются и складываются в реляционное хранилище. При этом они уже доступны для анализа при помощи различных средств построения отчетов. Затем

данные (полностью или частично) подготавливаются для OLAP-анализа. Они могут быть загружены в специальную БД OLAP или оставлены в реляционном хранилище. Важнейшим его элементом являются метаданные, т. е. информация о структуре, размещении и трансформации данных. Благодаря ним обеспечивается эффективное взаимодействие различных компонентов хранилища.

Подытоживая, можно определить OLAP как совокупность средств многомерного анализа данных, накопленных в хранилище. Теоретически средства OLAP можно применять и непосредственно к оперативным данным или их точным копиям (чтобы не мешать оперативным пользователям). Но мы тем самым рискуем наступить на уже описанные выше грабли, то есть начать анализировать оперативные данные, которые напрямую для анализа непригодны.
Основная часть

Глава 1. Хранилище данных

1.1. Что такое хранилище данных?

Информационные системы масштаба предприятия, как правило, содержат приложения, предназначенные для комплексного многомерного анализа данных, их динамики, тенденций и т.п. Такой анализ в конечном итоге призван содействовать принятию решений. Нередко эти системы так и называются — системы поддержки принятия решений.

Принять любое управленческое решение невозможно не обладая необходимой для этого информацией, обычно количественной. Для этого необходимо создание хранилищ данных (Data warehouses), то есть процесс сбора, отсеивания и предварительной обработки данных с целью предоставления результирующей информации пользователям для статистического анализа (а нередко и создания аналитических отчетов).

Ральф Кимбалл (Ralph Kimball), один из авторов концепции хранилищ данных, описывал хранилище данных как “место, где люди могут получить доступ к своим данным” (см., например, Ralph Kimball, “The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses”, John Wiley & Sons, 1996 и “The Data Web house Toolkit: Building the Web-Enabled Data Warehouse”, John Wiley & Sons, 2000). Он же сформулировал и основные требования к хранилищам данных:

- поддержка высокой скорости получения данных из хранилища;
- поддержка внутренней непротиворечивости данных;

- возможность получения и сравнения так называемых срезов данных (slice and dice);
- наличие удобных утилит просмотра данных в хранилище;
- полнота и достоверность хранимых данных;
- поддержка качественного процесса пополнения данных.

Удовлетворять всем перечисленным требованиям в рамках одного и того же продукта зачастую не удается. Поэтому для реализации хранилищ данных обычно используется несколько продуктов, одни их которых представляют собой собственно средства хранения данных, другие — средства их извлечения и просмотра, третьи — средства их пополнения и т.д.

Типичное хранилище данных, как правило, отличается от обычной реляционной базы данных. Во-первых, обычные базы данных предназначены для того, чтобы помочь пользователям выполнять повседневную работу, тогда как хранилища данных предназначены для принятия решений. Например, продажа товара и выписка счета производятся с использованием базы данных, предназначенной для обработки транзакций, а анализ динамики продаж за несколько лет, позволяющий спланировать работу с поставщиками, — с помощью хранилища данных.

Во-вторых, обычные базы данных подвержены постоянным изменениям в процессе работы пользователей, а хранилище данных относительно стабильно: данные в нем обычно обновляются согласно расписанию (например, еженедельно, ежедневно или ежечасно — в зависимости от потребностей). В идеале процесс пополнения представляет собой просто добавление новых данных за определенный период времени без изменения прежней информации, уже находящейся в хранилище.

И, в-третьих, обычные базы данных чаще всего являются источником данных, попадающих в хранилище. Кроме того, хранилище может пополняться за счет внешних источников, например статистических отчетов.

1.2ю Типичная структура хранилищ данных

Как мы уже знаем, конечной целью использования OLAP является анализ данных и представление результатов этого анализа в виде, удобном для восприятия и принятия решений. Основная идея OLAP заключается в построении многомерных кубов, которые будут доступны для пользовательских запросов. Однако исходные данные для построения OLAP-кубов обычно хранятся в реляционных базах данных. Нередко это специализированные реляционные базы данных, называемые также хранилищами данных (Data Warehouse). В отличие от так называемых оперативных

баз данных, с которыми работают приложения, модифицирующие данные, хранилища данных предназначены исключительно для обработки и анализа информации, поэтому проектируются они таким образом, чтобы время выполнения запросов к ним было минимальным. Обычно данные копируются в хранилище из оперативных баз данных согласно определенному расписанию.

Типичная структура хранилища данных существенно отличается от структуры обычной реляционной СУБД. Как правило, эта структура денормализована (это позволяет повысить скорость выполнения запросов), поэтому может допускать избыточность данных.

Для дальнейших примеров мы снова воспользуемся базой данных Northwind, входящей в комплекты поставки Microsoft SQL Server и Microsoft Access.

Основными составляющими структуры хранилищ данных являются таблица фактов (fact table) и таблицы измерений (dimension tables).

1.2.1. Таблица фактов

Таблица фактов является основной таблицей хранилища данных. Как правило, она содержит сведения об объектах или событиях, совокупность которых будет в дальнейшем анализироваться. Обычно говорят о четырех наиболее часто встречающихся типах фактов. К ним относятся:

- факты, связанные с транзакциями (Transaction facts). Они основаны на отдельных событиях (типичными примерами которых являются телефонный звонок или снятие денег со счета с помощью банкомата);
- факты, связанные с «моментальными снимками» (Snapshot facts). Основаны на состоянии объекта (например, банковского счета) в определенные моменты времени, например на конец дня, или месяца. Типичными примерами таких фактов являются объем продаж за день или дневная выручка;
- факты, связанные с элементами документа (Line-item facts). Основаны на том или ином документе (например, счете за товар или услуги) и содержат подробную информацию об элементах этого документа (например, количестве, цене, проценте скидки);
- факты, связанные с событиями или состоянием объекта (Event or state facts). Представляют возникновение события без подробностей о нем (например, просто факт продажи или факт отсутствия таковой без иных подробностей).

Для примера рассмотрим факты, связанные с элементами документа (в данном случае счета, выставленного за товар).

Таблица фактов, как правило, содержит уникальный составной ключ, объединяющий первичные ключи таблиц измерений. Чаще всего это целочисленные значения либо значения типа «дата/время» — ведь таблица фактов может содержать сотни тысяч или даже миллионы записей, и хранить в ней повторяющиеся текстовые описания, как правило, невыгодно — лучше поместить их в меньшие по объему таблицы измерений. При этом как ключевые, так и некоторые неключевые поля должны соответствовать будущим измерениям OLAP-куба. Помимо этого таблица фактов содержит одно или несколько числовых полей, на основании которых в дальнейшем будут получены агрегатные данные.

В данном примере измерениям будущего куба соответствуют первые шесть полей, а агрегатным данным — последние четыре.

Отметим, что для многомерного анализа пригодны таблицы фактов, содержащие как можно более подробные данные (то есть соответствующие членам нижних уровней иерархии соответствующих измерений). В данном случае предпочтительнее взять за основу факты продажи товаров отдельным заказчикам, а не суммы продаж для разных стран — последние все равно будут вычислены OLAP-средством. Исключение можно сделать, пожалуй, только для клиентских OLAP-средств (о них мы поговорим чуть позже), поскольку в силу ряда ограничений они не могут манипулировать большими объемами данных.

Отметим, что в таблице фактов нет никаких сведений о том, как группировать записи при вычислении агрегатных данных. Например, в ней есть идентификаторы продуктов или клиентов, но отсутствует информация о том, к какой категории относится данный продукт или в каком городе находится данный клиент. Эти сведения, в дальнейшем используемые для построения иерархий в измерениях куба, содержатся в таблицах измерений.

1.2.2. Таблицы измерений

Таблицы измерений содержат неизменяемые либо редко изменяемые данные. В подавляющем большинстве случаев эти данные представляют собой по одной записи для каждого члена нижнего уровня иерархии в измерении. Таблицы измерений также содержат как минимум одно описательное поле (обычно с именем члена измерения) и, как правило, целочисленное ключевое поле (обычно это суррогатный ключ) для однозначной идентификации члена измерения. Если

будущее измерение, основанное на данной таблице измерений, содержит иерархию, то таблица измерений также может содержать поля, указывающие на «родителя» данного члена в этой иерархии. Нередко (но не всегда) таблица измерений может содержать и поля, указывающие на «прародителей», и иных «предков» в данной иерархии (это обычно характерно для сбалансированных иерархий), а также дополнительные атрибуты членов измерений, содержавшиеся в исходной оперативной базе данных (например, адреса и телефоны клиентов).

Каждая таблица измерений должна находиться в отношении «один ко многим» с таблицей фактов.

Отметим, что скорость роста таблиц измерений должна быть незначительной по сравнению со скоростью роста таблицы фактов; например, добавление новой записи в таблицу измерений, характеризующую товары, производится только при появлении нового товара, не продававшегося ранее.

Одно измерение куба может содержаться как в одной таблице (в том числе и при наличии нескольких уровней иерархии), так и в нескольких связанных таблицах, соответствующих различным уровням иерархии в измерении. Если каждое измерение содержится в одной таблице, такая схема хранилища данных носит название «звезда» (star schema).

Если же хотя бы одно измерение содержится в нескольких связанных таблицах, такая схема хранилища данных носит название «снежинка» (snowflake schema). Дополнительные таблицы измерений в такой схеме, обычно соответствующие верхним уровням иерархии измерения и находящиеся в соотношении «один ко многим» в главной таблице измерений, соответствующей нижнему уровню иерархии, иногда называют консольными таблицами (outrigger table).

Отметим, что даже при наличии иерархических измерений с целью повышения скорости выполнения запросов к хранилищу данных нередко предпочтение отдается схеме «звезда».

Однако не все хранилища данных проектируются по двум приведенным выше схемам. Так, довольно часто вместо ключевого поля для измерения, содержащего данные типа «дата», и соответствующей таблицы измерений сама таблица фактов может содержать ключевое поле типа «дата». В этом случае соответствующая таблица измерений просто отсутствует.

В случае несбалансированной иерархии (например, такой, которая может быть основана на таблице Employees базы данных Northwind, имеющей поле Employee ID, которое одновременно является и первичным, и внешним ключом и отражает подчиненность одних сотрудников другим (см. рис. 6) в схему «снежинка» также следует вносить коррективы. В этом случае обычно в таблице измерений присутствует связь, аналогичная соответствующей связи в оперативной базе данных.

Еще один пример отступления от правил — наличие нескольких разных иерархий для одного и того же измерения. Типичные примеры таких иерархий — иерархии для календарного и финансового года (при условии, что финансовый год начинается не с 1 января), или с различными способами группировки членов измерения (например, группировать товары можно по категориям, а можно и по компаниям-поставщикам). В этом случае таблица измерений содержит поля для всех возможных иерархий с одними и теми же членами нижнего уровня, но с разными членами верхних уровней (пример такой таблицы приведен на рис. 3).

Как мы уже отмечали выше, таблица измерений может содержать поля, не имеющие отношения к иерархиям и представляющие собой просто дополнительные атрибуты членов измерений (member properties). Иногда такие атрибуты могут быть использованы при анализе данных.

Следует сказать, что для создания реляционных хранилищ данных нередко применяются специализированные СУБД, хранение данных в которых оптимизировано с точки зрения скорости выполнения запросов. Примером такого продукта является Sybase Adaptive Server IQ, реализующий нетрадиционный способ хранения данных в таблицах (не по строкам, а по столбцам). Однако создавать хранилища можно и в обычных реляционных СУБД.

Глава 2. Основные сведения об OLAP

2.1 Оперативная аналитическая обработка данных

В основе концепции OLAP лежит принцип многомерного представления данных. В 1993 году термин OLAP ввел Эдгар Кодд. Рассмотрев недостатки реляционной модели, он в первую очередь указал на невозможность «объединять, просматривать и анализировать данные с точки зрения множественности измерений, то есть самым понятным для корпоративных аналитиков способом», и определил общие требования к системам OLAP, расширяющим функциональность реляционных СУБД и включающим многомерный анализ как одну из своих

характеристик.

В большом числе публикаций аббревиатурой OLAP обозначается не только многомерный взгляд на данные, но и хранение самих данных в многомерной БД. Вообще говоря, это неверно, поскольку сам Кодд отмечает, что “Реляционные БД были, есть и будут наиболее подходящей технологией для хранения корпоративных данных. Необходимость существует не в новой технологии БД, а, скорее, в средствах анализа, дополняющих функции существующих СУБД и достаточно гибких, чтобы предусмотреть и автоматизировать разные виды интеллектуального анализа, присущие OLAP”. Такая путаница приводит к противопоставлениям наподобие “OLAP или ROLAP”, что не совсем корректно, поскольку ROLAP (реляционный OLAP) на концептуальном уровне поддерживает всю определенную термином OLAP функциональность. Более предпочтительным кажется использование для OLAP на основе многомерных СУБД специального термина MOLAP. По Кодду, многомерное концептуальное представление (multi-dimensional conceptual view) представляет собой множественную перспективу, состоящую из нескольких независимых измерений, вдоль которых могут быть проанализированы определенные совокупности данных. Одновременный анализ по нескольким измерениям определяется как многомерный анализ. Каждое измерение включает направления консолидации данных, состоящие из серии последовательных уровней обобщения, где каждый вышестоящий уровень соответствует большей степени агрегации данных по соответствующему измерению.

Исполнитель может определяться направлением консолидации, состоящим из уровней обобщения “предприятие – подразделение – отдел – служащий”. Измерение «Время» может даже включать два направления консолидации – “год – квартал – месяц – день” и “неделя – день”, поскольку счет времени по месяцам и по неделям несовместим. В этом случае становится возможным произвольный выбор желаемого уровня детализации информации по каждому из измерений. Операция спуска (drilling down) соответствует движению от высших ступеней консолидации к низшим; напротив, операция подъема (rolling up) означает движение от низших уровней к высшим.

2.2. Требования к средствам оперативной аналитической обработки

Многомерное концептуальное представление данных (Multi Dimensional Conceptual View). Концептуальное представление модели данных в продукте OLAP должно быть многомерным по своей природе, то есть позволять аналитикам выполнять

интуитивные операции “анализа вдоль и поперек” (“slice and dice”), вращения (rotate) и размещения (pivot) направлений консолидации. Прозрачность (Transparency). Пользователь не должен знать о том, какие конкретные средства используются для хранения и обработки данных, как данные организованы и откуда берутся.

Доступность (Accessibility). Аналитик должен иметь возможность выполнять анализ в рамках общей концептуальной схемы, но при этом данные могут оставаться под управлением оставшихся от старого наследства СУБД, будучи при этом привязанными к общей аналитической модели. То есть инструментарий OLAP должен накладывать свою логическую схему на физические массивы данных, выполняя все преобразования, требующиеся для обеспечения единого, согласованного и целостного взгляда пользователя на информацию.

Устойчивая производительность (Consistent Reporting Performance). С увеличением числа измерений и размеров базы данных аналитики не должны столкнуться с каким бы то ни было уменьшением производительности. Устойчивая производительность необходима для поддержания простоты использования и свободы от усложнений, которые требуются для доведения OLAP до конечного пользователя.

Клиент – серверная архитектура (Client-Server Architecture). Большая часть данных, требующих оперативной аналитической обработки, хранится в мэйнфреймовых системах, а извлекается с персональных компьютеров. Поэтому одним из требований является способность продуктов OLAP работать в среде клиент-сервер. Главной идеей здесь является то, что серверный компонент инструмента OLAP должен быть достаточно интеллектуальным и обладать способностью строить общую концептуальную схему на основе обобщения и консолидации различных логических и физических схем корпоративных баз данных для обеспечения эффекта прозрачности.

Равноправие измерений (Generic Dimensionality). Все измерения данных должны быть равноправны. Дополнительные характеристики могут быть предоставлены отдельным измерениям, но поскольку все они симметричны, данная дополнительная функциональность может быть предоставлена любому измерению. Базовая структура данных, формулы и форматы отчетов не должны опираться на какое-то одно измерение.

Динамическая обработка разреженных матриц (Dynamic Sparse Matrix Handling). Инструмент OLAP должен обеспечивать оптимальную обработку разреженных матриц. Скорость доступа должна сохраняться вне зависимости от расположения ячеек данных и быть постоянной величиной для моделей, имеющих разное число измерений и различную разреженность данных.

Поддержка многопользовательского режима (Multi-User Support). Зачастую несколько аналитиков имеют необходимость работать одновременно с одной аналитической моделью или создавать различные модели на основе одних корпоративных данных. Инструмент OLAP должен предоставлять им конкурентный доступ, обеспечивать целостность и защиту данных.

Неограниченная поддержка кроссмерных операций (Unrestricted Cross-dimensional Operations). Вычисления и манипуляция данными по любому числу измерений не должны запрещать или ограничивать любые отношения между ячейками данных. Преобразования, требующие произвольного определения, должны задаваться на функционально полном формульном языке.

Интуитивное манипулирование данными (Intuitive Data Manipulation). Переориентация направлений консолидации, детализация данных в колонках и строках, агрегация и другие манипуляции, свойственные структуре иерархии направлений консолидации, должны выполняться в максимально удобном, естественном и комфортном пользовательском интерфейсе[3].

Гибкий механизм генерации отчетов (Flexible Reporting). Должны поддерживаться различные способы визуализации данных, то есть отчеты должны представляться в любой возможной ориентации.

Неограниченное количество измерений и уровней агрегации (Unlimited Dimensions and Aggregation Levels). Настоятельно рекомендуется допущение в каждом серьезном OLAP инструменте как минимум пятнадцати, а лучше двадцати, измерений в аналитической модели.

Глава 3. Компоненты OLAP-систем

3.1. Сервер. Клиент. Интернет

OLAP позволяет выполнять быстрый и эффективный анализ над большими объемами данных. Данные хранятся в многомерном виде, что наиболее близко отражает естественное состояние реальных бизнес-данных. Кроме того, OLAP

предоставляет пользователям возможность быстрее и проще получать сводные данные. С его помощью они могут при необходимости углубляться (drill down) в содержимое этих данных для получения более детализированной информации Елманова Н.В., Федоров А.А. Введение в OLAP-технологии Microsoft. – М.:Диалог-МИФИ, 2004. – 312 с..

OLAP-система состоит из множества компонент. На самом высоком уровне представления система включает в себя источник данных, OLAP-сервер и клиента. Источник данных представляет собой источник, из которого берутся данные для анализа. Данные из источника переносятся или копируются на OLAP-сервер, где они систематизируются и подготавливаются для более быстрого впоследствии формирования ответов на запросы. Клиент – это пользовательский интерфейс к OLAP-серверу. В этом разделе статьи описываются функции каждой компоненты и значение всей системы в целом. Источники. Источником в OLAP-системах является сервер, поставляющий данные для анализа. В зависимости от области использования OLAP-продукта источником может служить Хранилище данных, наследуемая база данных, содержащая общие данные, набор таблиц, объединяющих финансовые данные или любая комбинация перечисленного. Способность OLAP-продукта работать с данными из различных источников очень важна. Требование единого формата или единой базы, в которых бы хранились все исходные данные, не подходит администраторам баз данных. Кроме того, такой подход уменьшает гибкость и мощность OLAP-продукта. Администраторы и пользователи полагают, что OLAP-продукты, обеспечивающие извлечение данных не только из различных, но и из множества источников, оказываются более гибкими и полезными, чем те, что имеют более жесткие требования.

Сервер. Прикладной частью OLAP-системы является OLAP-сервер. Эта составляющая выполняет всю работу (в зависимости от модели системы), и хранит в себе всю информацию, к которой обеспечивается активный доступ. Архитектурой сервера управляют различные концепции. В частности, основной функциональной характеристикой OLAP-продукта является использование для хранения данных многомерной (ММБД, MDDB) либо реляционной (РДБ, RDB) базы данных. Агрегированные/Предварительно агрегированные данные

Быстрая реализация запросов является императивом для OLAP. Это один из базовых принципов OLAP – способность интуитивно манипулировать данными требует быстрого извлечения информации. В целом, чем больше вычислений необходимо произвести, чтобы получить фрагмент информации, тем медленнее происходит отклик. Поэтому, чтобы сохранить маленькое время реализации

запросов, фрагменты информации, обращение к которым обычно происходит наиболее часто, но которые при этом требуют вычисления, подвергаются предварительной агрегации. То есть они подсчитываются и затем хранятся в базе данных в качестве новых данных. В качестве примера типа данных, который допустимо рассчитать заранее, можно привести сводные данные – например, показатели продаж по месяцам, кварталам или годам, для которых действительно введенными данными являются ежедневные показатели Дейт К. Введение в системы баз данных. – М.: Наука, 2005 г. – 246 с..

Различные поставщики придерживаются различных методов отбора параметров, требующих предварительной агрегации и числа предварительно вычисляемых величин. Подход к агрегации влияет одновременно и на базу данных и на время реализации запросов. Если вычисляется больше величин, вероятность того, что пользователь запросит уже вычисленную величину, возрастает, и поэтому время отклика сократиться, так как не придется запрашивать изначальную величину для вычисления. Однако, если вычислить все возможные величины – это не лучшее решение – в таком случае существенно возрастает размер базы данных, что сделает ее неуправляемой, да и время агрегации будет слишком большим. К тому же, когда в базу данных добавляются числовые значения, или если они изменяются, информация эта должна отражаться на предварительно вычисленных величинах, зависящих от новых данных. Таким образом, и обновление базы может также занять много времени в случае большого числа предварительно вычисляемых величин. Поскольку обычно во время агрегации база данных работает автономно, желательно, чтобы время агрегации было не слишком длительным.

Клиент. Клиент – это как раз то, что используется для представления и манипуляций с данными в базе данных. Клиент может быть и достаточно несложным – в виде таблицы, включающей в себя такие возможности OLAP, как, например, вращение данных (пивотинг) и углубление в данные (дриллинг), и представлять собой специализированное, но такое же простое средство просмотра отчетов или быть таким же мощным инструментом, как созданное на заказ приложение, спроектированное для сложных манипуляций с данными. Интернет является новой формой клиента. Кроме того, он несет на себе печать новых технологий; множество интернет-решений существенно отличаются по своим возможностям в целом и в качестве OLAP-решения – в частности. В данном разделе обсуждаются различные функциональные свойства каждого типа клиентов.

Несмотря на то, что сервер – это как бы “хребет” OLAP-решения, клиент не менее важен. Сервер может обеспечить прочный фундамент для облегчения

манипуляций с данными, но если клиент сложен или малофункционален, пользователь не сможет воспользоваться всеми преимуществами мощного сервера. Клиент настолько важен, что множество поставщиков сосредотачивают свои усилия исключительно на разработке клиента. Все, что включается в состав этих приложений, представляет собой стандартный взгляд на интерфейс, заранее определенные функции и структуру, а также быстрые решения для более или менее стандартных ситуаций. Например, популярны финансовые пакеты. Заранее созданные финансовые приложения позволят специалистам использовать привычные финансовые инструменты без необходимости проектировать структуру базы данных или общепринятые формы и отчеты. Инструмент запросов/Генератор отчетов. Инструмент запросов или генератор отчетов предлагает простой доступ к OLAP-данным. Они имеют простой в использовании графический интерфейс и позволяют пользователям создавать отчеты перемещением объектов в отчет методом “drag and drop”. Тогда как традиционный генератор отчетов предоставляет пользователю возможность быстро выпускать форматированные отчеты, генераторы отчетов, поддерживающие OLAP, формируют актуальные отчеты. Конечный продукт представляет собой отчет, имеющий возможности углубления в данные до уровня подробностей, вращения (пивотинг) отчетов, поддержки иерархий и др.. Add-Ins (дополнения) электронных таблиц.

Сегодня во многих направлениях бизнеса с помощью электронных таблиц производятся различные формы анализа корпоративных данных. В каком-то смысле это идеальное средство создания отчетов и просмотра данных. Аналитик может создавать макросы, работающие с данными в выбранном направлении, а шаблон может быть спроектирован таким образом, что, когда происходит ввод данных, формулы рассчитывают правильные величины, исключая необходимость неоднократного ввода простых расчетов.

Тем не менее, все это дает в результате “плоский” отчет, что означает, что как только он создан, трудно рассматривать его в различных аспектах. Например, диаграмма отображает информацию за некоторый временной период, – скажем, за месяц. И если некто желает увидеть показатели за день (в противоположность данным за месяц), необходимо будет создать абсолютно новую диаграмму. Предстоит определить новые наборы данных, добавить в диаграмму новые метки и внести множество других простых, но трудоемких изменений. Кроме того, существует ряд областей, в которых могут быть допущены ошибки, что в целом уменьшает надежность. Когда к таблице добавляется OLAP, появляется возможность создавать единственную диаграмму, а затем подвергать ее

различным манипуляциям с целью предоставления пользователю необходимой информации, не обременяя себя созданием всех возможных представлений. Интернет в роли клиента. Новым членом семейства OLAP-клиентов является Интернет. Существует масса преимуществ в формировании OLAP-отчетов через Интернет. Наиболее существенным представляется отсутствие необходимости в специализированном программном обеспечении для доступа к информации. Это экономит предприятию кучу времени и денег.

Каждый Интернет-продукт специфичен. Некоторые упрощают создание Web-страниц, но имеют меньшую гибкость. Другие позволяют создавать представления данных, а затем сохранять их как статические HTML-файлы. Все это дает возможность просматривать данные через Интернет, но не более того. Активно манипулировать данными с их помощью невозможно.

Существует и другой тип продуктов – интерактивный и динамический, превращающий такие продукты в полнофункциональные инструменты. Пользователи могут осуществлять углубление в данные, пивотинг, ограничение измерений, и др. Прежде, чем выбрать средство реализации Интернет, важно понять, какие функциональные возможности требуются от Web-решения, а затем определить, какой продукт наилучшим образом воплотит эту функциональность Голицина О.Л., Максимов Н.В., Попов И.И. Базы данных: Учебное пособие. – М.: ФОРУМ: ИНФРА-М, 2003. – 352с..

Приложения. Приложения – это тип клиента, использующий базы данных OLAP. Они идентичны инструментам запросов и генераторам отчетов, описанным выше, но, кроме того, они вносят в продукт более широкие функциональные возможности. Приложение, как правило, обладает большей мощностью, чем инструмент запроса.

Разработка. Обычно поставщики OLAP обеспечивают среду разработки для создания пользователями собственных настроенных приложений. Среда разработки в целом представляет собой графический интерфейс, поддерживающий объектно-ориентированную разработку приложений. К тому же, большинство поставщиков обеспечивают API, который может использоваться для интеграции баз данных OLAP с другими приложениями.

3.2. OLAP – клиенты

OLAP-клиенты со встроенной OLAP-машиной устанавливаются на ПК пользователей. Они не требуют сервера для вычислений, и им присуще нулевое

администрирование. Такие клиенты позволяют пользователю настроиться на существующие у него базы данных; как правило, при этом создается словарь, скрывающий физическую структуру данных за ее предметным описанием, понятным специалисту. После этого OLAP-клиент выполняет произвольные запросы и результаты их отображает в OLAP-таблице. В этой таблице, в свою очередь, пользователь может манипулировать данными и получать на экране или на бумаге сотни различных отчетов. OLAP-клиенты, предназначенные для работы с РСУБД, позволяют анализировать уже имеющиеся в корпорации данные, например хранящиеся в БД OLTP Сахаров А. А. Концепция построения и реализации информационных систем, ориентированных на анализ данных // СУБД. – 2004. – № 4. – 55-70 с.. Однако вторым их назначением может быть быстрое и дешевое создание хранилищ или витрин данных — в этом случае программистам организации нужно лишь создать совокупности таблиц типа “звезда” в реляционных БД и процедуры загрузки данных. Наиболее трудоемкая часть работы — написание интерфейсов с многочисленными вариантами пользовательских запросов и отчетов — реализуется в OLAP-клиенте буквально за несколько часов. Конечному же пользователю для освоения такой программы требуется порядка 30 минут. OLAP-клиенты поставляются самими разработчиками баз данных, как многомерных, так и реляционных. Это SAS Corporate Reporter, являющийся почти эталонным по удобству и красоте продуктом, Oracle Discoverer, комплекс программ MS Pivot Services и Pivot Table и др. Многие программы, предназначенные для работы с MS OLAP Services, поставляются в рамках кампании “OLAP в массы”, которую проводит корпорация Microsoft. Как правило, они являются улучшенными вариантами Pivot Table и рассчитаны на использование в MS Office или Web-браузере. Это продукты фирм Matryx, Knosys и т. д., благодаря простоте, дешевизне и эффективности приобретшие огромную популярность на Западе.

Глава 4. Классификация продуктов OLAP

4.1. Многомерный OLAP

В настоящее время на рынке присутствует большое количество продуктов, которые в той или иной степени обеспечивают функциональность OLAP. Обеспечивая многомерное концептуальное представление со стороны пользовательского интерфейса к исходной базе данных, все продукты OLAP делятся на три класса по типу исходной БД.

1. Самые первые системы оперативной аналитической обработки (например, Essbase компании Arbor Software, Oracle Express Server компании Oracle)

относились к классу MOLAP, то есть могли работать только со своими собственными многомерными базами данных. Они основываются на патентованных технологиях для многомерных СУБД и являются наиболее дорогими. Эти системы обеспечивают полный цикл OLAP-обработки. Они либо включают в себя, помимо серверного компонента, собственный интегрированный клиентский интерфейс, либо используют для связи с пользователем внешние программы работы с электронными таблицами. Для обслуживания таких систем требуется специальный штат сотрудников, занимающихся установкой, сопровождением системы, формированием представлений данных для конечных пользователей.

2. Системы оперативной аналитической обработки реляционных данных (ROLAP) позволяют представлять данные, хранимые в реляционной базе, в многомерной форме, обеспечивая преобразование информации в многомерную модель через промежуточный слой метаданных. К этому классу относятся DSS Suite компании MicroStrategy, MetaCube компании Informix, DecisionSuite компании Information Advantage и другие. Программный комплекс ИнфоВизор, разработанный в России, в Ивановском государственном энергетическом университете, также является системой этого класса. ROLAP-системы хорошо приспособлены для работы с крупными хранилищами. Подобно системам MOLAP, они требуют значительных затрат на обслуживание специалистами по информационным технологиям и предусматривают многопользовательский режим работы.
3. Наконец, гибридные системы (Hybrid OLAP, HOLAP) разработаны с целью совмещения достоинств и минимизации недостатков, присущих предыдущим классам. К этому классу относится Media/MR компании Speedware. По утверждению разработчиков, он объединяет аналитическую гибкость и скорость ответа MOLAP с постоянным доступом к реальным данным, свойственным ROLAP.

Помимо перечисленных средств существует еще один класс – инструменты генерации запросов и отчетов для настольных ПК, дополненные функциями OLAP или интегрированные с внешними средствами, выполняющими такие функции. Эти хорошо развитые системы осуществляют выборку данных из исходных источников, преобразуют их и помещают в динамическую многомерную БД, функционирующую на клиентской станции конечного пользователя. Основными представителями этого класса являются BusinessObjects одноименной компании, BrioQuery компании Brio Technology и PowerPlay компании Cognos. Обзор некоторых продуктов OLAP приведен в приложении.

В специализированных СУБД, основанных на многомерном представлении данных, данные организованы не в форме реляционных таблиц, а в виде упорядоченных многомерных массивов:

1) гиперкубов (все хранимые в БД ячейки должны иметь одинаковую мерность, то есть находиться в максимально полном базисе измерений) или

2) поликубов (каждая переменная хранится с собственным набором измерений, и все связанные с этим сложности обработки перекладываются на внутренние механизмы системы).

Использование многомерных БД в системах оперативной аналитической обработки имеет следующие достоинства.

1. В случае использования многомерных СУБД поиск и выборка данных осуществляется значительно быстрее, чем при многомерном концептуальном взгляде на реляционную базу данных, так как многомерная база данных денормализована, содержит заранее агрегированные показатели и обеспечивает оптимизированный доступ к запрашиваемым ячейкам.
2. Многомерные СУБД легко справляются с задачами включения в информационную модель разнообразных встроенных функций, тогда как объективно существующие ограничения языка SQL делают выполнение этих задач на основе реляционных СУБД достаточно сложным, а иногда и невозможным.

С другой стороны, имеются существенные ограничения.

1. Многомерные СУБД не позволяют работать с большими базами данных. К тому же за счет денормализации и предварительно выполненной агрегации объем данных в многомерной базе, как правило, соответствует (по оценке Кодда) в 2.5-100 раз меньшему объему исходных детализированных данных.
2. Многомерные СУБД по сравнению с реляционными очень неэффективно используют внешнюю память. В подавляющем большинстве случаев информационный гиперкуб является сильно разреженным, а поскольку данные хранятся в упорядоченном виде, неопределенные значения удаётся удалить только за счет выбора оптимального порядка сортировки, позволяющего организовать данные в максимально большие непрерывные группы. Но даже в этом случае проблема решается только частично. Кроме того, оптимальный с точки зрения хранения разреженных данных порядок сортировки скорее всего не будет совпадать с порядком, который чаще всего

используется в запросах. Поэтому в реальных системах приходится искать компромисс между быстродействием и избыточностью дискового пространства, занятого базой данных.

Следовательно, использование многомерных СУБД оправдано только при следующих условиях.

1. Объем исходных данных для анализа не слишком велик (не более нескольких гигабайт), то есть уровень агрегации данных достаточно высок.
2. Набор информационных измерений стабилен (поскольку любое изменение в их структуре почти всегда требует полной перестройки гиперкуба).
3. Время ответа системы на нерегламентированные запросы является наиболее критичным параметром.
4. Требуется широкое использование сложных встроенных функций для выполнения кроссмерных вычислений над ячейками гиперкуба, в том числе возможность написания пользовательских функций.

4.2. Реляционный OLAP

Непосредственное использование реляционных БД в системах оперативной аналитической обработки имеет следующие достоинства.

1. В большинстве случаев корпоративные хранилища данных реализуются средствами реляционных СУБД, и инструменты ROLAP позволяют производить анализ непосредственно над ними. При этом размер хранилища не является таким критичным параметром, как в случае MOLAP.
2. В случае переменной размерности задачи, когда изменения в структуру измерений приходится вносить достаточно часто, ROLAP системы с динамическим представлением размерности являются оптимальным решением, так как в них такие модификации не требуют физической реорганизации БД.
3. Реляционные СУБД обеспечивают значительно более высокий уровень защиты данных и хорошие возможности разграничения прав доступа.

Главный недостаток ROLAP по сравнению с многомерными СУБД – меньшая производительность. Для обеспечения производительности, сравнимой с MOLAP, реляционные системы требуют тщательной проработки схемы базы данных и настройки индексов, то есть больших усилий со стороны администраторов БД. Только при использовании звездообразных схем производительность хорошо настроенных реляционных систем может быть приближена к производительности

систем на основе многомерных баз данных Пржиялковский В. В. Сложный анализ данных большого объема: новые перспективы компьютеризации // СУБД. – 2006. – № 4. – 71-83 с.

Описанию схемы звезды (star schema) и рекомендациям по ее применению полностью посвящены работы. Ее идея заключается в том, что имеются таблицы для каждого измерения, а все факты помещаются в одну таблицу, индексируемую множественным ключом, составленным из ключей отдельных измерений (Приложение А). Каждый луч схемы звезды задает, в терминологии Кодда, направление консолидации данных по соответствующему измерению.

В сложных задачах с многоуровневыми измерениями имеет смысл обратиться к расширениям схемы звезды – схеме созвездия (fact constellation schema) и схеме снежинки (snowflake schema). В этих случаях отдельные таблицы фактов создаются для возможных сочетаний уровней обобщения различных измерений (Приложение Б). Это позволяет добиться лучшей производительности, но часто приводит к избыточности данных и к значительным усложнениям в структуре базы данных, в которой оказывается огромное количество таблиц фактов.

Увеличение числа таблиц фактов в базе данных может проистекать не только из множественности уровней различных измерений, но и из того обстоятельства, что в общем случае факты имеют разные множества измерений. При абстрагировании от отдельных измерений пользователь должен получать проекцию максимально полного гиперкуба, причем далеко не всегда значения показателей в ней должны являться результатом элементарного суммирования. Таким образом, при большом числе независимых измерений необходимо поддерживать множество таблиц фактов, соответствующих каждому возможному сочетанию выбранных в запросе измерений, что также приводит к неэкономному использованию внешней памяти, увеличению времени загрузки данных в БД схемы звезды из внешних источников и сложностям администрирования.

Частично решают эту проблему расширения языка SQL (операторы GROUP BY CUBE”, “GROUP BY ROLLUP” и “GROUP BY GROUPING SETS”); кроме того, предлагается механизм поиска компромисса между избыточностью и быстродействием, рекомендуя создавать таблицы фактов не для всех возможных сочетаний измерений, а только для тех, значения ячеек которых не могут быть получены с помощью последующей агрегации более полных таблиц фактов (Приложение В).

В любом случае, если многомерная модель реализуется в виде реляционной базы данных, следует создавать длинные и “узкие” таблицы фактов и сравнительно небольшие и “широкие” таблицы измерений. Таблицы фактов содержат численные значения ячеек гиперкуба, а остальные таблицы определяют содержащий их многомерный базис измерений. Часть информации можно получать с помощью динамической агрегации данных, распределенных по незвездообразным нормализованным структурам, хотя при этом следует помнить, что включающие агрегацию запросы при высоконормализованной структуре БД могут выполняться довольно медленно.

Ориентация на представление многомерной информации с помощью звездообразных реляционных моделей позволяет избавиться от проблемы оптимизации хранения разреженных матриц, остро стоящей перед многомерными СУБД (где проблема разреженности решается специальным выбором схемы). Хотя для хранения каждой ячейки используется целая запись, которая помимо самих значений включает вторичные ключи – ссылки на таблицы измерений, несуществующие значения просто не включаются в таблицу фактов.

Заключение

Успешная компания сегодня должна принимать множество различных решений. И чем более обоснованные решения будут приняты, тем большего успеха и прибыли достигнет предприятие. Для многих лиц, играющих ключевую роль в принятии решений, способность быстрее и эффективнее конкурента анализировать бизнес-процессы означает принятие более правильных решений, достижение большей прибыльности и большего успеха. Оптимизация реляционной базы данных предоставила компаниям возможность продуктивно собирать данные о транзакциях, поставляя информации лицам, принимающим решения.

Рассмотрев вопросы работы и применения технологии OLAP перед компаниями возникают вопросы, ответы на которые позволят выбрать продукт наилучшим образом отвечающий потребностям пользователя.

Это следующие вопросы:

Откуда поступают данные? – Данные, подлежащие анализу, могут находиться в различных местах. Возможно, что база данных OLAP будет получать их из корпоративного Хранилища данных или из OLTP-системы. Если OLAP-продукт уже

имеет возможность получить доступ к какому-то источнику данных, процессы категоризации и очистки данных сокращаются.

Какие манипуляции пользователь производит над данными? –

Как только пользователь получил доступ к базе данных и начал выполнять анализ, важно, чтобы он был в состоянии оперировать данными соответствующим образом. В зависимости от потребностей пользователя, может оказаться, что необходим мощный генератор отчетов или возможность создавать и размещать динамические web-страницы. Вместе с тем, может быть пользователю предпочтительнее иметь в своем распоряжении средство простого и быстрого создания собственных приложений.

Каков общий объем данных? – Это наиболее важный фактор при определении базы данных OLAP. Реляционные OLAP-продукты способны оперировать большими объемами данных лучше, чем многомерные. Если объем данных не требует использования реляционной базы, многомерный продукт может использоваться с не меньшим успехом.

Кем является пользователь? – При определении клиента OLAP-системы важен уровень квалификации пользователя. Некоторым пользователям удобнее интегрировать OLAP с таблицей, тогда как другие предпочтут специализированное приложение. В зависимости от квалификации пользователя решается и вопрос о проведении обучения. Крупная компания может пожелать оплатить тренинги для пользователей, компания меньшего размера может отказаться от них. Клиент должен быть таким, чтобы пользователи чувствовали себя уверенно и могли эффективно его использовать.

Сегодня большинство мировых компаний перешли к использованию OLAP как базовой технологии для предоставления информации лицам, принимающим решениям. Поэтому принципиальный вопрос, которым необходимо задаться, не состоит в том, следует ли продолжать применять электронные таблицы в качестве основной платформы для подготовки отчетности, бюджетирования и прогнозирования. Компании должны спросить себя, готовы ли они терять конкурентные преимущества, используя неточную, неактуальную и неполную информацию, прежде чем они созреют и рассмотрят альтернативные технологии.

Так же, в заключение следует отметить, что аналитические возможности технологий OLAP повышают пользу данных, хранящихся в корпоративном хранилище информации, позволяя компании более эффективно взаимодействовать

со своими клиентами.

1. BI-инструменты

Инструменты и технологии, используемые для доступа к информации. Включают OLAP-технологии, data mining и сложный анализ; средства конечного пользователя и инструменты построения нерегламентированных запросов, инструментальные панели для мониторинга хозяйственной деятельности и генераторы корпоративной отчетности.

2. On-line Analitic Processing, OLAP (Оперативная аналитическая обработка)

Технология аналитической обработки информации в режиме реального времени, включающая составление и динамическую публикацию отчетов и документов.

3. Slice and Dice (Продольные и поперечные срезы, дословно – “нарезка на ломтики и кубики”)

Термин, использующийся для описания функции сложного анализа данных, обеспечиваемой средствами OLAP. Выборка данных из многомерного куба с заданными значениями и заданным взаимным расположением измерений.

4. Вращение (пивотинг) данных (Data Pivot)

Процесс вращения таблицы с данными, т. е. преобразования столбцов в строки и наоборот.

5. Вычисленный элемент (Calculated member)

Элемент измерения, чья величина определяется величинами других элементов (например, математическими или логическими приложениями). Вычисленный элемент может представлять собой часть OLAP сервера или быть описан пользователем в течение интерактивной сессии. Вычисленный элемент – это любой элемент, который не вводится, а вычисляется.

6. Глобальные бизнес-модели (Global Business Models)

Тип Хранилища данных, обеспечивающий доступ к информации, которая распределена по различным системам предприятия и находится под контролем различных подразделений или отделов с разными базами данных и моделями данных. Такой тип Хранилища данных труден для построения из-за необходимости объединения усилий пользователей различных подразделений для разработки

общей модели данных для Хранилища.

7. Добыча данных (Data Mining)

Технические приемы, использующие программные инструменты, предназначенные для такого пользователя, который, как правило, не может заранее сказать, что конкретно он ищет, а может указать лишь определенные образцы и направления поиска.

8. Клиент/Сервер (Client/Server)

Технологический подход, заключающийся в разделении процесса на отдельные функции. Сервер выполняет несколько функций – управление коммуникациями, обеспечение обслуживания базы данных и др. Клиент выполняет индивидуальные пользовательские функции – обеспечение соответствующих интерфейсов, выполнение межэкранной навигации, предоставление функций помощи (help) и др.

9. Многомерная база данных, СУМБД (Multi-dimensional Database, MDBS and MDBMS)

Мощная база данных, позволяющая пользователям анализировать большие объемы данных. База данных со специальной организацией хранения – кубами, обеспечивающая высокую скорость работы с данными, хранящимися как совокупность фактов, измерений и заранее вычисленных агрегатов.

10. Углубление в данные (Drill Down)

Метод изучения детальных данных, используемый при анализе суммарного уровня данных. Уровни “углубления” зависят от степени детализации данных в [ранилище.

11. Центральное Хранилище (Central Warehouse)

а) База данных, содержащая данные, собираемые из операционных систем организации. Имеет структуру, удобную для анализа данных. Предназначена для поддержки принятия решений и создания единого информационного пространства корпорации.

б) Способ автоматизации, охватывающий все информационные системы, управляемые из одного места.

Список использованных источников

1. Голицина О.Л., Максимов Н.В., Попов И.И. Базы данных: Учебное пособие. – М.: ФОРУМ: ИНФРА-М, 2003. – 352 с.

2. Дейт К. Введение в системы баз данных. – М.: Наука, 2005 г. – 246 с.
3. Елманова Н.В., Федоров А.А. Введение в OLAP-технологии Microsoft. – М.:Диалог-, 2004. – 312 с.
4. Карпова Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2006. – 304 с.
5. Коровкин С.Д., Левенец И.А., Ратманова И.Д., Старых В.А., Щавелёв Л.В. Решение проблемы комплексного оперативного анализа информации хранилищ данных / СУБД. – 2005. – № 5-6. – 47-51 с.
6. Кречетов Н.В., Иванов П.В. Продукты для интеллектуального анализа данных Computer Week-Москва. – 2003. – № 14-15. – 32-39 с.
7. Пржиялковский В.В. Сложный анализ данных большого объема: новые перспективы компьютеризации / СУБД. – 2006. – № 4. – 71-83 с.
8. Сахаров А.А. Концепция построения и реализации информационных систем, ориентированных на анализ данных / СУБД. – 2004. – № 4. – 55-70 с.
9. Ульман Дж. Основы систем баз данных. – М.: Финансы и статистика, 2003. – 312 с.
10. Хаббард Дж. Автоматизированное проектирование баз данных. – М.: Мир, 2007. – 294 с