Введение в OLAP и многомерные базы данных

Хранилища данных (место OLAP в информационной структуре предприятия)

Термин “OLAP” неразрывно связан с термином “хранилище данных” (Data Warehouse).

Приведем определение, сформулированное “отцом-основателем” хранилищ данных Биллом Инмоном: “Хранилище данных - это предметно-ориентированное, привязанное ко времени и неизменяемое собрание данных для поддержки процесса принятия управляющих решений”.

Данные в хранилище попадают из оперативных систем (OLTP-систем), которые предназначены для автоматизации бизнес-процессов. Кроме того, хранилище может пополняться за счет внешних источников, например статистических отчетов.

Зачем строить хранилища данных - ведь они содержат заведомо избыточную информацию, которая и так “живет” в базах или файлах оперативных систем? Ответить можно кратко: анализировать данные оперативных систем напрямую невозможно или очень затруднительно. Это объясняется различными причинами, в том числе разрозненностью данных, хранением их в форматах различных СУБД и в разных “уголках” корпоративной сети. Но даже если на предприятии все данные хранятся на центральном сервере БД (что бывает крайне редко), аналитик почти наверняка не разберется в их сложных, подчас запутанных структурах. Автор имеет достаточно печальный опыт попыток “накормить” голодных аналитиков “сырыми” данными из оперативных систем - им это оказалось “не по зубам”.

Таким образом, задача хранилища - предоставить “сырье” для анализа в одном месте и в простой, понятной структуре. Ральф Кимбалл в предисловии к своей книге “The Data Warehouse Toolkit” пишет, что если по прочтении всей книги читатель поймет только одну вещь, а именно: структура хранилища должна быть простой, - автор будет считать свою задачу выполненной.

Есть и еще одна причина, оправдывающая появление отдельного хранилища - сложные аналитические запросы к оперативной информации тормозят текущую работу компании, надолго блокируя таблицы и захватывая ресурсы сервера.

На мой взгляд, под хранилищем можно понимать не обязательно гигантское скопление данных - главное, чтобы оно было удобно для анализа. Вообще говоря, для маленьких хранилищ предназначается отдельный термин - Data Marts (киоски данных), но в нашей российской практике его не часто услышишь.

OLAP - удобный инструмент анализа

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 (если он захочет придти), и он (если не занят) сделает новый отчет достаточно быстро - скажем, в течение часа (пишу и сам не верю - так быстро в жизни не бывает; давайте дадим ему часа три). 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.

Компоненты, входящие в типичное хранилище, представлены на рис. 1.

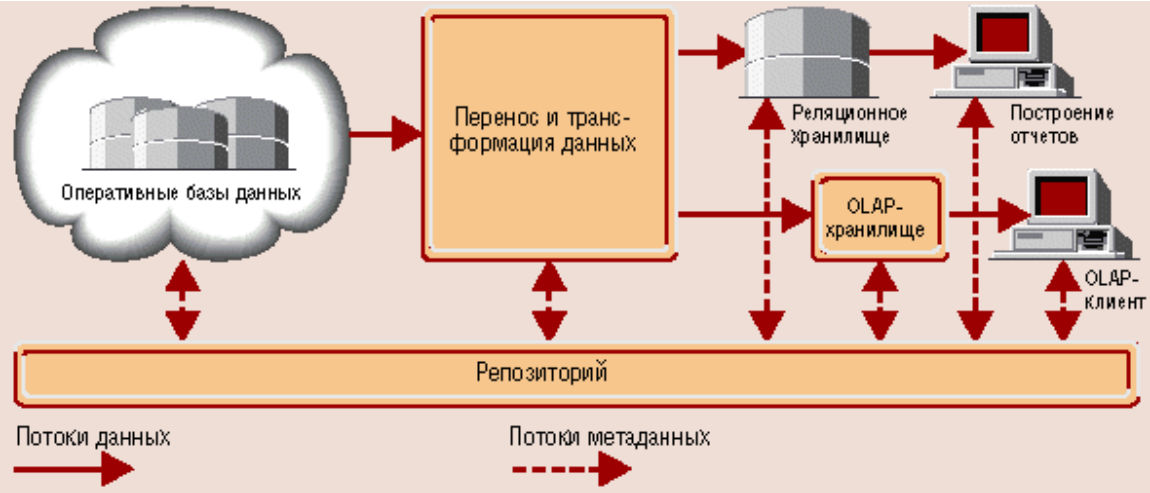

Рис. 1. Структура хранилища данных

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 им 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, т. е. начать анализировать оперативные данные, которые напрямую для анализа непригодны.

Определение и основные понятия OLAP

Для начала расшифруем: OLAP - это Online Analytical Processing, т. е. оперативный анализ данных. 12 определяющих принципов OLAP сформулировал в 1993 г. Е. Ф. Кодд - “изобретатель” реляционных БД. Позже его определение было переработано в так называемый тест FASMI, требующий, чтобы OLAP-приложение предоставляло возможности быстрого анализа разделяемой многомерной информации (см. http://www.olapreport.com/fasmi.htm).

Тест FASMI

- Fast (Быстрый) - анализ должен производиться одинаково быстро по всем аспектам информации. Приемлемое время отклика - 5 с или менее.

- Analysis (Анализ) - должна быть возможность осуществлять основные типы числового и статистического анализа, предопределенного разработчиком приложения или произвольно определяемого пользователем.

- Shared (Разделяемой) - множество пользователей должно иметь доступ к данным, при этом необходимо контролировать доступ к конфиденциальной информации.

- Multidimensional (Многомерной) - это основная, наиболее существенная характеристика OLAP.

- Information (Информации) - приложение должно иметь возможность обращаться к любой нужной информации, независимо от ее объема и места хранения.

OLAP = многомерное представление = Куб

OLAP предоставляет удобные быстродействующие средства доступа, просмотра и анализа деловой

информации. Пользователь получает естественную, интуитивно понятную модель данных, организуя их в виде многомерных кубов (Cubes). Осями многомерной системы координат служат основные атрибуты анализируемого бизнес-процесса. Например, для продаж это могут быть товар, регион, тип покупателя. В качестве одного из измерений используется время. На пересечениях осей - измерений (Dimensions) - находятся данные, количественно характеризующие процесс - меры (Measures). Это могут быть объемы продаж в штуках или в денежном выражении, остатки на складе, издержки и т. п. Пользователь, анализирующий информацию, может “разрезать” куб по разным направлениям, получать сводные (например, по годам) или, наоборот, детальные (по неделям) сведения и осуществлять прочие манипуляции, которые ему придут в голову в процессе анализа.

В качестве мер в трехмерном кубе, изображенном на рис. 2, использованы суммы продаж, а в качестве измерений - время, товар и магазин. Измерения представлены на определенных уровнях группировки: товары группируются по категориям, магазины - по странам, а данные о времени совершения операций - по месяцам. Чуть позже мы рассмотрим уровни группировки (иерархии) подробнее.

Рис. 2. Пример куба

“Разрезание” куба

Даже трехмерный куб сложно отобразить на экране компьютера так, чтобы были видны значения интересующих мер. Что уж говорить о кубах с количеством измерений, большим трех? Для визуализации данных, хранящихся в кубе, применяются, как правило, привычные двумерные, т. е. табличные, представления, имеющие сложные иерархические заголовки строк и столбцов.

Двумерное представление куба можно получить, “разрезав” его поперек одной или нескольких осей (измерений): мы фиксируем значения всех измерений, кроме двух, - и получаем обычную двумерную таблицу. В горизонтальной оси таблицы (заголовки столбцов) представлено одно измерение, в вертикальной (заголовки строк) - другое, а в ячейках таблицы - значения мер. При этом набор мер фактически рассматривается как одно из измерений - мы либо выбираем для показа одну меру (и тогда можем разместить в заголовках строк и столбцов два измерения), либо показываем несколько мер (и тогда одну из осей таблицы займут названия мер, а другую - значения единственного “неразрезанного” измерения).

Взгляните на рис. 3 - здесь изображен двумерный срез куба для одной меры - Unit Sales (продано штук) и двух “неразрезанных” измерений - Store (Магазин) и Время (Time).

	США	Канада	Мексика
Январь	20 000	4000	2000
Февраль	30 000	6000	3000
Март	50 000	10 000	5000

Рис. 3. Двумерный срез куба для одной меры

На рис. 4 представлено лишь одно “неразрезанное” измерение - Store, но зато здесь отображаются значения нескольких мер - Unit Sales (продано штук), Store Sales (сумма продажи) и Store Cost (расходы магазина).

	США	Канада	Мексика
Unit Sales	2000	400	200
Store Sales	30 000	6000	3000
Store Cost	10 000	2000	1000

Рис. 4. Двумерный срез куба для нескольких мер

Двумерное представление куба возможно и тогда, когда “неразрезанными” остаются и более двух измерений. При этом на осях среза (строках и столбцах) будут размещены два или более измерений “разрезаемого” куба - см. рис. 5.

	Январь			Февраль		
	США	Канада	Мексика	США	Канада	Мексика
Unit Sales	500	100	50	500	100	50
Store Sales	7500	1500	750	7500	1500	750
Store Cost	2500	500	250	2500	500	250

Рис. 5. Двумерный срез куба с несколькими измерениями на одной оси

Метки

Значения, “откладываемые” вдоль измерений, называются членами или метками (members). Метки используются как для “разрезания” куба, так и для ограничения (фильтрации) выбираемых данных - когда в измерении, остающемся “неразрезанным”, нас интересуют не все значения, а их подмножество, например три города из нескольких десятков. Значения меток отображаются в двумерном представлении куба как заголовки строк и столбцов.

Иерархии и уровни

Метки могут объединяться в иерархии, состоящие из одного или нескольких уровней (levels). Например, метки измерения “Магазин” (Store) естественно объединяются в иерархию с уровнями:

All (Мир)

Country (Страна)

State (Штат)

City (Город)

Store (Магазин).

В соответствии с уровнями иерархии вычисляются агрегатные значения, например объем продаж для USA (уровень “Country”) или для штата California (уровень “State”). В одном измерении можно реализовать более одной иерархии - скажем, для времени: {Год, Квартал, Месяц, День} и {Год, Неделя, День}.

Архитектура OLAP-приложений

Все, что говорилось выше про OLAP, по сути, относилось к многомерному представлению данных. То, как данные хранятся, грубо говоря, не волнует ни конечного пользователя, ни разработчиков инструмента, которым клиент пользуется.

Многомерность в OLAP-приложениях может быть разделена на три уровня:

- Многомерное представление данных - средства конечного пользователя, обеспечивающие многомерную визуализацию и манипулирование данными; слой многомерного представления абстрагирован от физической структуры данных и воспринимает данные как многомерные.

- Многомерная обработка - средство (язык) формулирования многомерных запросов (традиционный реляционный язык SQL здесь оказывается непригодным) и процессор, умеющий обработать и выполнить такой запрос.

- Многомерное хранение - средства физической организации данных, обеспечивающие эффективное выполнение многомерных запросов.

Первые два уровня в обязательном порядке присутствуют во всех OLAP-средствах. Третий уровень, хотя и является широко распространенным, не обязателен, так как данные для многомерного представления могут извлекаться и из обычных реляционных структур; процессор многомерных запросов в этом случае транслирует многомерные запросы в SQL-запросы, которые выполняются реляционной СУБД.

Конкретные OLAP-продукты, как правило, представляют собой либо средство многомерного представления данных, OLAP-клиент (например, Pivot Tables в Excel 2000 фирмы Microsoft или ProClarity фирмы Knosys), либо многомерную серверную СУБД, OLAP-сервер (например, Oracle Express Server или Microsoft OLAP Services).

Слой многомерной обработки обычно бывает встроен в OLAP-клиент и/или в OLAP-сервер, но может быть выделен в чистом виде, как, например, компонент Pivot Table Service фирмы Microsoft.

Технические аспекты многомерного хранения данных

Как уже говорилось выше, средства OLAP-анализа могут извлекать данные и непосредственно из реляционных систем. Такой подход был более привлекательным в те времена, когда OLAP-серверы отсутствовали в прайс-листах ведущих производителей СУБД. Но сегодня и Oracle, и Informix, и Microsoft предлагают полноценные OLAP-серверы, и даже те IT-менеджеры, которые не любят разводить в своих сетях “зоопарк” из ПО разных производителей, могут купить (точнее, обратиться с соответствующей просьбой к руководству компании) OLAP-сервер той же марки, что и основной сервер баз данных.

OLAP-серверы, или серверы многомерных БД, могут хранить свои многомерные данные по-разному. Прежде чем рассмотреть эти способы, нам нужно поговорить о таком важном аспекте, как хранение агрегатов. Дело в том, что в любом хранилище данных - и в обычном, и в многомерном - наряду с детальными данными, извлекаемыми из оперативных систем, хранятся и суммарные показатели (агрегированные показатели, агрегаты), такие, как суммы объемов продаж по месяцам, по категориям товаров и т. п. Агрегаты хранятся в явном виде с единственной целью - ускорить выполнение запросов. Ведь, с одной стороны, в хранилище накапливается, как правило, очень большой объем данных, а с другой - аналитиков в большинстве случаев интересуют не детальные, а обобщенные показатели. И если каждый раз для вычисления суммы продаж за год пришлось бы суммировать миллионы индивидуальных продаж, скорость, скорее всего, была бы неприемлемой. Поэтому при загрузке данных в многомерную БД вычисляются и сохраняются все суммарные показатели или их часть.

Но, как известно, за все надо платить. И за скорость обработки запросов к суммарным данным приходится платить увеличением объемов данных и времени на их загрузку. Причем увеличение объема может стать буквально катастрофическим - в одном из опубликованных стандартных тестов полный подсчет агрегатов для 10 Мб исходных данных потребовал 2,4 Гб, т. е. данные выросли в 240 раз! Степень “разбухания” данных при вычислении агрегатов зависит от количества измерений куба и структуры этих измерений, т. е. соотношения количества “отцов” и “детей” на разных уровнях измерения. Для решения проблемы хранения агрегатов применяются подчас сложные схемы, позволяющие при вычислении далеко не всех возможных агрегатов достигать значительного повышения производительности выполнения запросов.

Теперь о различных вариантах хранения информации. Как детальные данные, так и агрегаты могут храниться либо в реляционных, либо в многомерных структурах. Многомерное хранение позволяет обращаться с данными как с многомерным массивом, благодаря чему обеспечиваются одинаково быстрые вычисления суммарных показателей и различные многомерные преобразования по любому из измерений. Некоторое время назад OLAP-продукты поддерживали либо реляционное, либо многомерное хранение. Сегодня, как правило, один и тот же продукт обеспечивает оба этих вида хранения, а также третий вид - смешанный. Применяются следующие термины:

- MOLAP (Multidimensional OLAP) - и детальные данные, и агрегаты хранятся в многомерной БД. В этом случае получается наибольшая избыточность, так как многомерные данные полностью содержат реляционные.

- ROLAP (Relational OLAP) - детальные данные остаются там, где они “жили” изначально - в реляционной БД; агрегаты хранятся в той же БД в специально созданных служебных таблицах.

- HOLAP (Hybrid OLAP) - детальные данные остаются на месте (в реляционной БД), а агрегаты хранятся в многомерной БД.

Каждый из этих способов имеет свои преимущества и недостатки и должен применяться в зависимости от условий - объема данных, мощности реляционной СУБД и т. д.

При хранении данных в многомерных структурах возникает потенциальная проблема “разбухания” за счет хранения пустых значений. Ведь если в многомерном массиве зарезервировано место под все возможные комбинации меток измерений, а реально заполнена лишь малая часть (например, ряд продуктов продается только в небольшом числе регионов), то бо/льшая часть куба будет пустовать, хотя место будет занято. Современные OLAP-продукты умеют справляться с этой проблемой.

Продолжение следует. В дальнейшем мы поговорим о конкретных OLAP-продуктах, выпускаемых ведущими производителями.

С автором статьи можно связаться по адресу: alperovich@digdes.com.