Попробуйте OLAP !

Бернард Люпен, Oracle Magazine/RE, #9/2000

Источник:http://perso.wanadoo.fr/bernard.lupin/english

Много лет реляционные базы данных главенствовали в сфере управления данными. Но в последнее время была разработана новая концепция в мире баз данных.

Возможно, Вы уже слышали о многомерных базах данных и OLAP-инструментарии. Возможно, Вы даже понимаете значение этого термина, который так легко катится с языка, на каком бы языке Вы не говорили (или во всяком случае более легко, чем слова RDBMS или SGBDR - французский эквивалент RDBMS).

Но, к сожалению, когда люди говорят с Вами об On Line Analytical Processing (оперативный анализ данных), Вы, даже если понимаете английский язык, вероятно, имеете не более чем смутное представление о возможностях баз данных OLAP.

Ниже, при помощи простого (в 8 этапов) примера я намерен шаг за шагом построить маленькую базу данных OLAP. Меньше чем за 5 минут, измерения (dimensions), меры (measures), другие аггрегации (aggregations) не будут больше составлять для Вас никакой тайны.

Конечно, в теории все это очень интересно, но как насчет практики? Я намерен построить базу данных OLAP для этого примера, используя Oracle Express. Если Вы знакомы с реляционными базами данных и языком SQL, то без труда заметите, что это абсолютно разные вещи! А если Вы уже знакомы с другими многомерными продуктами, то сможете и сравнить их между собой.

Если Вам понравится этот пример, то прочитайте и другие подготовленные для Вас страницы:

- я предлагаю Вам чуть-чуть более подробные сведения об OLAP .
- Вы также можете обратиться к словарю-глоссарию (glossary), в котором собраны некоторые термины, используемые в мире OLAP.

Наконец, Вы можете послать мне свои предложения, критические или ободряющие замечания, использовав для этого мойemail-адрес или специальную форму.

От редактора Russian Oracle Internet Magazine:

- Во-первых, поблагодарим автора, Bernard Lupin, любезно разрешившего воспользоваться его материалом, и сообщим адрес сайта http://perso.wanadoo.fr/bernard.lupin/english, где читатели могут познакомиться с оригинальными английской и французской авторскими версиями. Отметим, что Bernard Lupin открыл этот сайт 5 мая 1998 года.
- Во-вторых, извинимся перед автором и читателями в том, что в силу специфики нашего издания, нам пришлось несколько урезать оформление, сократить ссылки и отказаться от некоторых подразделов.
- В-третьих, нам пришлось перестроить Оглавление

О Г Л А В Л Е Н И Е

От переводчика

Известно, что разработчики приложений для традиционных СУБД, к числу которых ныне относится Oracle, несколько побаиваются многомерных систем, или, иначе – “OLAP-систем”. Уж больно там все необычно и непривычно. Учебников мало, а документация – не всегда хороший учебник. А ведь, несмотря на теоретическую неразличимость реляционной и многомерной модели, благодаря своему особому методу хранения и методу доступа многомерные системы весьма многообещающи. Они могут послужить хорошим дополнением к использованию традиционных СУБД в приложении, а иногда и играть самодостаточную роль.

Но получается, что их возможности на практике востребованы явно недостаточно. Помочь исправить такое положение дел, хотя бы частично, способно хорошее доступное введение в тематику многомерных баз. Представляется, что Бернар Люпэн, консультант из Франции, сделал весьма полезное дело, разместив и Internet именно такое введение, вполне подходящее для начала пути “от простого – к сложному”.

При этом важной особенностью текста Люпэна является то, что он разбит на две части: большую часть, в которой речь идет об основах построения многомерных баз, и меньшую, в которой решения, подготовленные в первой части, ложатся на практику применения конкретной системы – Oracle Express. Весьма педагогично, Люпэн, как бы, “подпускает к инструменту” в самый последний момент. Такое распределение материала очень важно – потому, что главную трудность обычно вызывает именно методология построения многомерной базы, – но, к сожалению, оно нетрадиционно.

Автор любезно согласился на безвозмездное размещение русского перевода своего текста в Internet, за что ему следует выразить благодарность. Если вы соберетесь написать ему о своих впечатлениях, он будет только рад, но учтите, что нашего языка он не знает.

И вот еще о чем нельзя не сказать: полупустая книжная полка разработчика многомерных баз на основе Oracle Express, все же, не абсолютно пуста. Те, кто с помощью Люпэна успешно сделал свои первые шаги в OLAP, могут обратиться для дальнейшего изучения этих систем к книге Сергея Архипенкова "Аналитические системы на базе Oracle Express OLAP" (рецензия по адресу http://www.oracle.ru/press/press2.htm), вышедшей на русском языке некоторое время назад. Поищите эту книгу в электронных или традиционных книжных магазинах !

Владимир Пржиялковский

Знакомимся с OLAP: шаг за шагом

Шаг 1: Свойства таблицы Sales

В рамках этого шага мы намереваемся поэтапно построить классическую базу данных, то есть базу данных с таблицами и столбцами. Мы увидим с вами, что несмотря на очевидный

интерес разработчиков к реляционным базам данных, этот подход не всегда оказывается наилучшим возможным.

Компания "Best Foot Forward" хочет построить свою базу данных, с помощью которой она могла бы отслеживать продажи обуви по категориям и месяцам. Хорошая затея ! Представим себе следующую таблицу продаж **Sales**:

Month	Style	Quantity	Total Value TE
January 2000	Ski boot	5	1 700 F
January 2000	Gumboot	300	65 000 F
.....			

Здесь разные типы обуви собраны в таблице Style, и все, что нам нужно -- это включить соответствующее поле для внешнего ключа в таблицу Sales.

Для анализа данных мы могли бы, например, соотнести месяцы строкам, а типы столбцам и сформировать таким образом две таблицы: количества продаж Quantity и общей стоимости без учета налогов Total Value Tax Exclusive. Эти таблицы позволят нам проводить моделирование и строить графики так, как мы пожелаем. Итак, у нас два простых отчета: один, показывающий число пар обуви, проданных за месяц, и второй -- общая стоимость.

Продажи пошли как по-маслу (а до этого, все же, были больше !), компания выросла и со временем открыла несколько магазинов. Таблицу продаж придется переработать так:

Month	Style	Outlet	Quantity	Total Value TE
April 2000	Gumboot	Lyon	10	1 500 F
April 2000	Sneaker	Paris Bastille	850	260 000 F
.....				

В нашей реляционной базе данных теперь появилась таблица outlet магазинов с такими полями, как название, поле ключа и прочими характеристиками наподобие адреса.

Теперь для анализа стало уже не хватать двумерного листка бумаги (или электронной таблицы). Если мы хотим посмотреть данные о работе магазина, нам нужно сначала их извлечь. А если нам нужно по очереди посмотреть работу всех магазинов в феврале, нам придется по очереди и извлекать данные.

Теперь мы имеем уже шесть разных возможных отчетов :

- Для каждого магазина или для всех магазинов каждый из двух ранее приведенных отчетов,
- Для каждого типа или по всем типам два отчета, где строка -- это месяц и столбец -- магазин,
- Для каждого месяца или для года отчеты с соответствием "строка -- магазин" и "столбец -- тип".

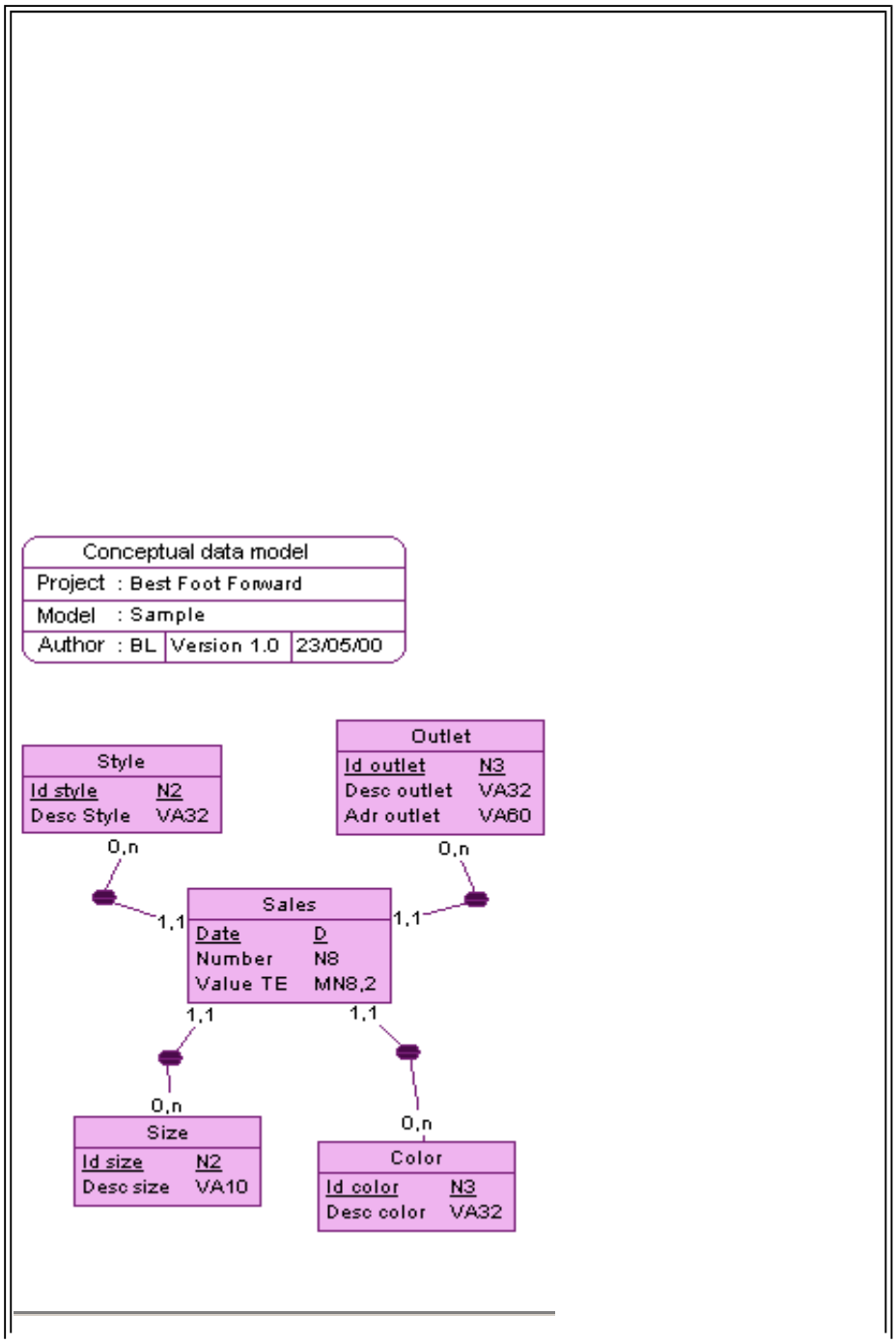
Если мы захотим в отчетах поменять строки со столбцами, то их окажется уже не шесть, а двенадцать.

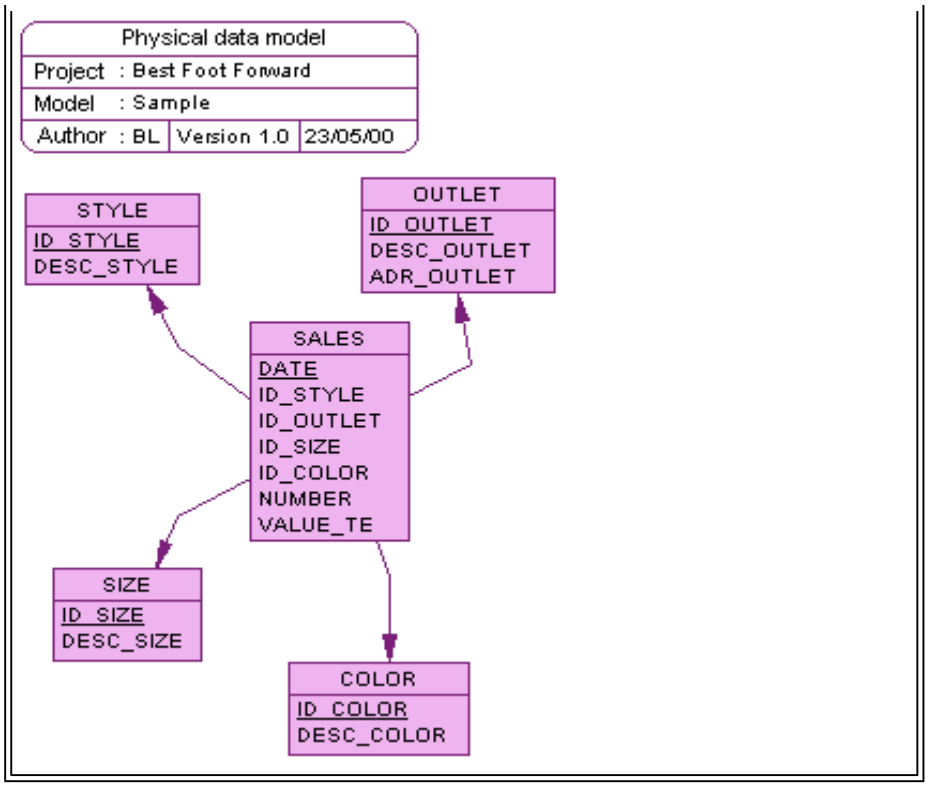
Еще мы пришли к заключению, что нужно анализировать продажи и по другим критериям,

таким как "род" Gender (мужская, женская, детская обувь), "размер" Size, а может быть и "цвет" Colour.

При таком повороте событий возникает проблема обеспечения каждого табочего места, с которого будут анализироваться продажи, монитором с разрешением 1280 X 1024, чтобы можно было все эти продажи разглядеть. Таблица продаж стала очень большой, как по числу строк, так и столбцов. При анализе данных будут возникать промежуточные суммы по группам обуви, что может сказаться на времени реакции системы. Нам придется заняться индексированием и прочими уловками повышения производительности.

Структура реляционной базы данных, которую мы только что построили, иногда называется "схема типа звезда" - название содержит намек на внешний вид соответствующей понятийной модели: в центре "звезды" таблица продаж, а "лучи" протянуты к таблицам магазинов, типа, цвета и так далее. Увидели по четыре таких "луча" на изображениях понятийной (концептуальной) и физической моделей ниже ?





Шаг 2 : Три измерения куба количество

Как мы уже знаем, компания "Best Foot Forward" намеревается достичь прогресса в показателях своих продаж, скажем, по месяцам, роду обуви и по магазинам. В терминологии OLAP эти критерии анализа носят название измерений, или иногда -- осей.

В нашем примере можно говорить об измерении Магазин. Оно может содержать несколько значений, наподобие "Paris Bastille". Эти значения являются положениями на шкале измерения Магазин.

В реляционной модели измерения соответствуют лучам схемы "звезда", которую мы видели на Шаге 1. Есть, однако, одно небольшое исключение: практика заведения таблицы с месяцами в реляционной модели отсутствует, потому что названия месяцев и их порядок присутствует в системе неявно. Здесь же нам абсолютно необходимо иметь явное измерение для месяцев, так чтобы положения на шкале этого измерения именовались как-нибудь наподобие "января 2000".

В общем случае одно измерение может содержать от двух до нескольких тысяч существующих в нем положений.

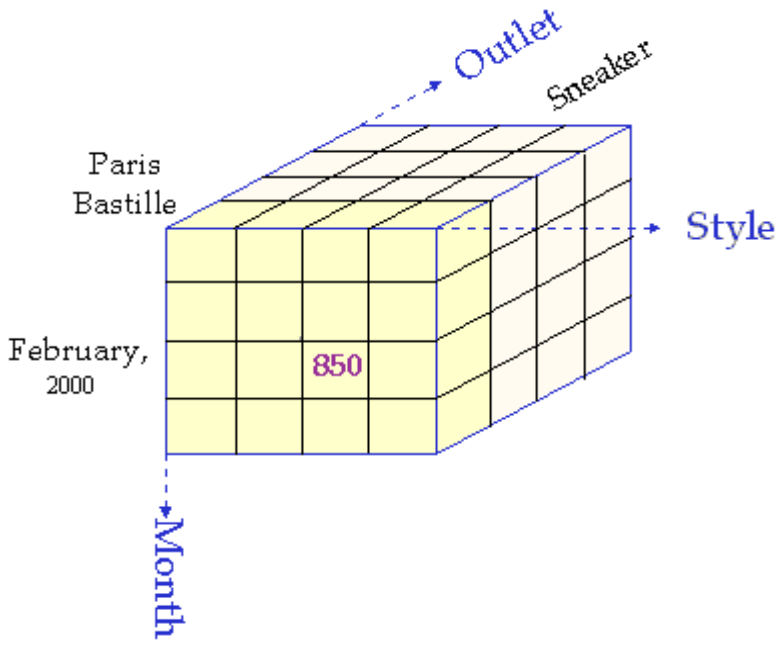
В нашем случае объектом внимания руководства фирмы служат два индикатора: количества проданной обуви и стоимость без учета налогов. Эти индикаторы носят названиемер, или, иногда, переменных.

Теперь мы можем сказать, что мера Количество имеет измерения Месяц, Род и Магазин.

Каждая мера может характеризоваться значениями числом от одного до нескольких миллионов. Все значения меры имеют одинаковый тип, например, целое или десятичное.

Куда нас занесло ! Теперь не остается никаких сомнений в том, что наша база данных стала

по-настоящему многомерной. К нашему счастью мера "Количество" определяется только тремя измерениями. Это действительно счастье, хотя при отображении на двумерный экран нам придется рисовать куб. 850 пар ботинок с предыдущей страницы будут при этом выглядеть примерно так:



Каждая ячейка куба представляет собой значение. Измерения обозначены вдоль ребер куба. Каждая грань соответствует набору значений, соответствующему положению на одной из трех шкал измерений. К примеру, передняя грань относится к магазину Paris Bastille. И, наконец, куб целиком представляет собой всю меру, в данном случае количество проданных пар.

Если мы добавим общую стоимость без учета налогов Total Value TE, то получим уже другой объект, состоящий из четырех шкал измерений: Месяц, Род, Магазин и Индикатор, позиции на которых задают существующие значения мер. Этот новый объект называется гиперкуб.

Шаг 3: Иерархии шкалы измерения time

Возможность помесячного изучения продаж, конечно, полезна, но, в тоже время, имеет свои ограничения. По этой причине мы переименуем измерение Месяц и назовем его, к примеру, Time. Положениями на шкале измерений Time могут быть, помимо месяцев, дни, периоды и годы. Таким образом, управляющий персонал сможет анализировать результаты работы разных магазинов как по более общим, так и более детальным отрезкам времени.
Чтобы нам самим было легче ориентироваться во всех этих позициях на шкале измерений, требуется создать иерархию. В нашем случае иерархия образована четырьмя уровнями, соответственно дню, месяцу, кварталу и году. Теперь мы имеем возможность вполне естественно соединить 18 мая 2000 года с маем 2000 года, и далее со вторым кварталом 2000 года и с самим 2000 годом. Для описания данных иерархии годятся выражения типа "прямой" или "дальний потомок", "родитель", "брат/сестра".

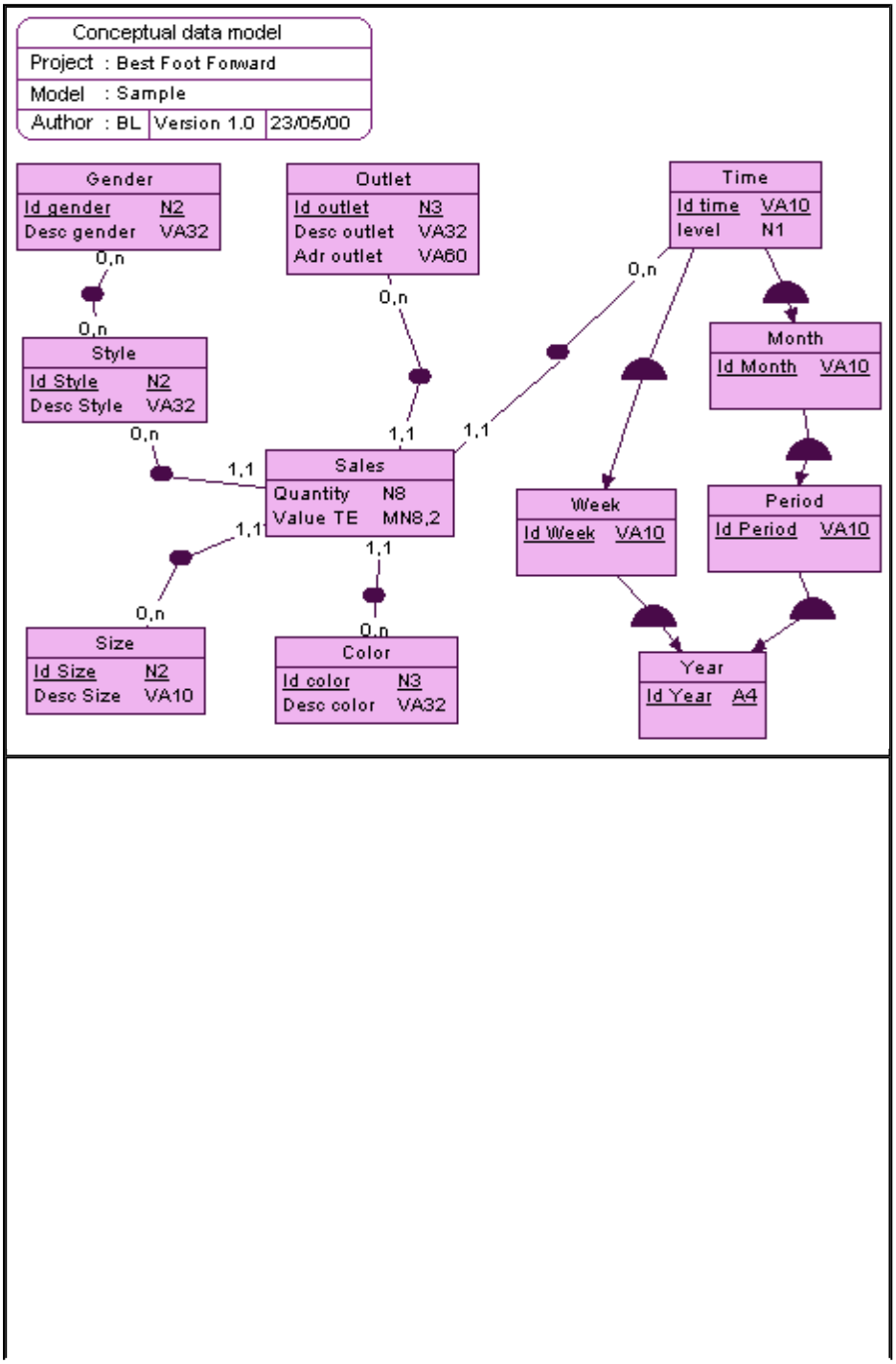
Точно таким же образом можно завести иерархию для шкалы измерений Магазин. К примеру, мы бы могли захотеть группировать магазины по району, области, стране. Обычно для одного измерения можно организовывать даже сразу несколько иерархий. Так, вторая иерархия для

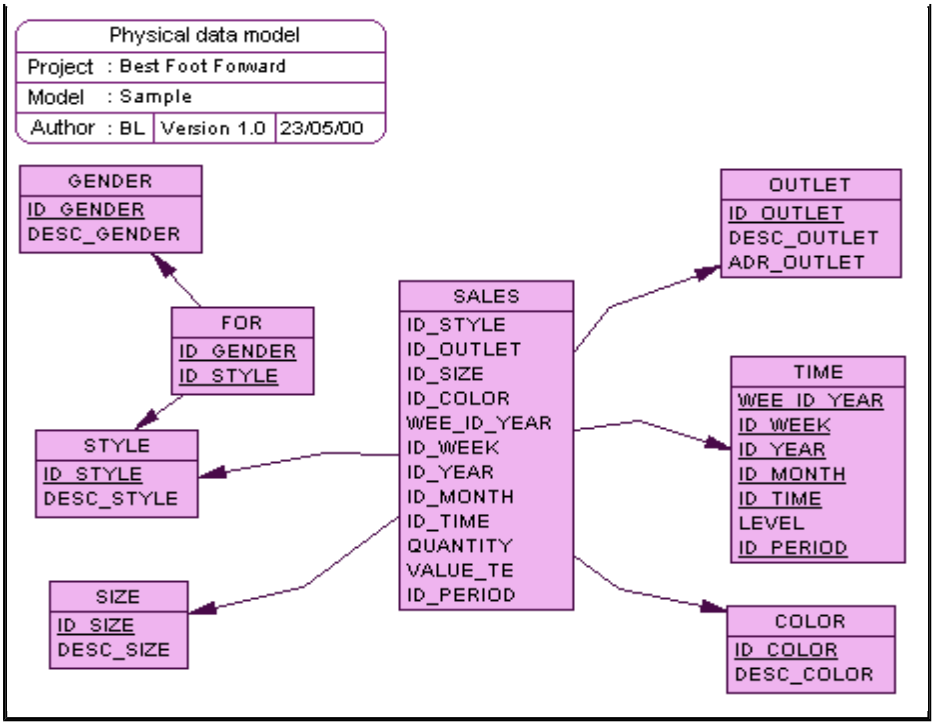
измерения времени могла бы иметь три уровня: день, неделя и год. В этом случае 18 мая 2000 года будет соединено с 19-й неделей 2000 года и с самим 2000 годом.

Единственным ограничением для создания иерархии служит то, что образована она должна быть каскадом отношений "один ко многим". То есть в любой иерархии каждая позиция на шкале должна иметь ровно одного родителя.

Предположим для примера, что мы захотели создать иерархию для магазинов, чтобы показывать с ее помощью, какие категории обуви (спортивная, прогулочная, детская) как продаются. Для этого нам потребуется ввести специализацию магазинов ! По этой причине лучшим решением будет расположить новую иерархию на шкале измерения стиля Style, так как стиль пары обуви принадлежит ровно одной категории.

Новая реляционная модель будет выглядеть так:





Теперь измерения количества Quantity и стоимости Total Value Tax Exclusive охарактеризованы измерениями Time, Style и Outlet. Каждый магазин предоставляет данные за день, а база OLAP отслеживает консолидацию данных в соответствии с разными иерархиями. Пока вся консолидация заключается в вычислении сумм, встроенные в систему функции обеспечивают этот процесс просто и незаметно для глаз.

Результаты осуществления такого агрегирования в соответствии с иерархиями хранятся в базе данных на более высоких уровнях иерархии.

Шаг 4 : Автоматические вычисления

Итак, наша база данных типа OLAP состоит сейчас из двух "кубов" данных (или же "мер", "замеров", "показателей"), именно данных о числе проданных пар обуви и их стоимости без учета налогов. Обе наши меры имеют в качестве шкал измерений время, род и магазин: Time, Style и Outlet.

Из этих двух показателей мы может получить и другие, но не путем хранения их в базе, а путем выполнения автоматического пересчета всякий раз, когда пользователям эти данные понадобятся. Такие показатели иногда называютформулами, имея в виду задающий их вычисления текст.

Для пользователя разницы между хранимым показателем и формулой нет. И те, и другие определяются шкалами измерений и типом. Так, формула Average Price, вычисляющая среднюю цену пары ботинок, имеет шкалы измерений Time, Style и Outlet и имеет десятичный тип. Это самая простая формула, и записывается она так:

```
Average Price = Total Value Tax Exclusive / Quantity,
```

то есть Средняя цена = Общая стоимость без учета налогов / Количество.

Всякий раз, когда пользователь запрашивает это значение, СУБД производит необходимое

деление и выдает нужные данные.

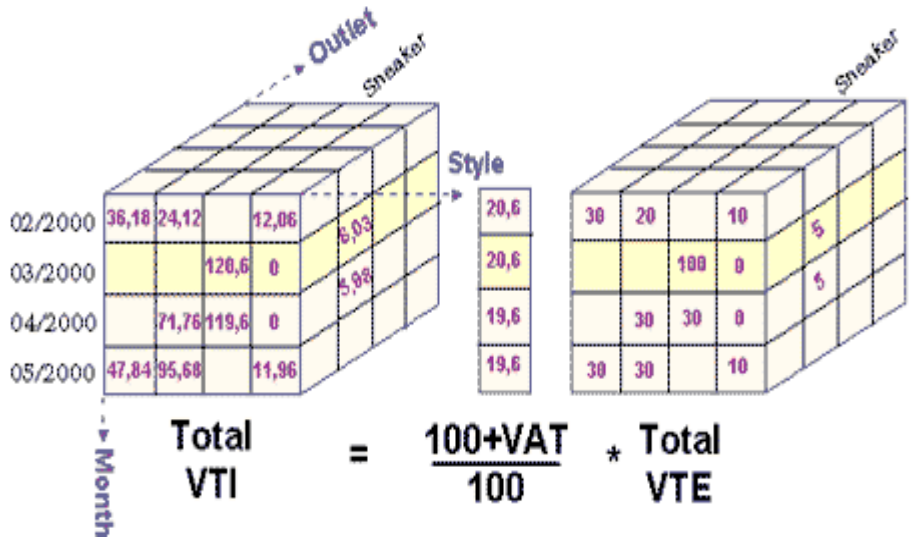
Для формулы вовсе не обязательно, чтобы все ее элементы измерялись одинаково. Мы бы могли, например, определить новую формулу так:

Total Value tax Inclusive = Total Value Tax Exclusive x 1.206.

Здесь каждая ячейка индивидуально помножается на 1.206.

Так как налог на добавленную стоимость (VAT) изменяется (чрезмерно) часто, можно задать новую переменную VAT, шкалой измерения которой будет только время. Эта переменная будет указывать на рост налога VAT в любой точке шкалы времени Time. Тем самым мы, весьма интуитивно, получим:

Total Value Tax Inclusive = Total Value Tax Exclusive x (100+VAT)/100



Таким образом получится, что для каждой ячейки, к которой применилась эта формула, будет правильно вычислена общая сумма.

Далее, если налог VAT изменяется по-разному для разных товаров, то шкалами измерения для этой переменной должны стать Time и Product. Графически значения переменной в этом случае образуют не линию, а плоскость.

Шаг 5: Начинаем использовать атрибуты

Хорошо, конечно, знать о том, сколько ботинок продано, но для процветания нашего объединения "Best Foot Forward" неплохо было бы знать к тому же, сколько именно ботинок какого сорта. Для этого каждый склад должен будет сообщать более детальную информацию и указывать, вдобавок, число проданных за день пар конкретной модели.

По это причине мы заменим шкалу Style шкалой Reference, артикул. Теперь мы знаем, что магазин "Paris Bastille" продал 18 августа 1998 года 7 пар ботинок артикула 215431324. Это весьма ценно для повышения качества управления заказами: нехватка или избыток товара на складе могут уйти в прошлое !

С другой стороны стало труднее понять состояние рынка. Невзирая на это, мы должны прямо сейчас определить атрибуты для каждого артикула ботинок. Значениями атрибутов могут стать Color (цвет: голубой, белый, красный), Substance (материал: кожа, ткань, синтетика), или Type (тип: мужская, женская, детская обувь). Если для каждого размера артикулы разные, то атрибутом следует также сделать и размер Size; иначе же Size станет новой шкалой измерения.

Теперь, когда мы предложили новое понятие артикула, администратору остается только занести его характеристики в систему. В OLAP-системе Color, Substance и Type составят новые шкалы измерений. А для определения атрибутов для каждого артикула состема строит отношения.

Имея эти отношения, мы можем построить новые формулы, шкалами размерности которым послужат какие-нибудь из этих атрибутов, а не шкала "артикул". Когда пользователь запросит данные, система запросит значения по артикулу и построит суммы по атрибутам.

Как вы, возможно, уже догадались, наша база данных OLAP теперь в состоянии отвечать на вопросы типа :

- Какой цвет ботинок лучше всего продавался в августе 1998 года ?
- Сколько кожаной женской обуви продал магазин "Paris Bastille" в 1998 году ?
- Каковы доли мужской, женской и детской обуви, продаваемой фирмой "Best Foot Forward" ?

Шаг 6 : Вычисления на оси времени

Теперь компания Best Foot Forward хочет прогнозировать свои действия на будущее. Предыдущие примеры в этом не помогут: нам нужно анализировать изменения в поступающих в базу данных и, по возможности, уметь готовиться к грядущим изменениям.

По части обработки запросов, связанных с временной шкалой, базы данных типа OLAP -- весьма мощный инструмент. Например, они дают возможность узнать, какой неделе года соответствует день 12 июня 2000 года; сколько дней имеется между 9 маем 1998 года и 15 июнем 2002 года; существует ли день 29 февраля 2000 года и так далее.

В некоторых системах для шкалы времени используются стандартные типы данных. Характеристикий для таких шкал служит понятие периодичности (день, неделя, семестр, ...), и существуют функции преобразования из одного типа периодичности в другой. OLAP-система хранит в себе все подобное описание календаря, так что разработчику нет нужды заботиться о сложных алгоритмах построения и поддержки иерархических структур описания времени.

Таким образом, для вопросов, приводимых ниже, нам достаточно простых и небольших программ :

- Какие изменения претерпела моя чистая прибыль по сравнению с тем же месяцем прошлого года.
- Какие изменения претерпели мои чистые показатели в сравнении с усредненным значением за последние три месяца.
- Какова будет тенденция рынка в ближайшие 12 месяцев.

В реляционных базах такие запросы потребуют очень ресурсоемких формулировок, так что ответа от системы пользователь рискует не получить никогда ...

Шаг 7 : Разреженность данных

И наконец, случилось ! Благодаря четкому управлению, фирма Best Foot Forward чувствует себя прекрасно, а предлагаемый ею выбор обуви безупречен. В продаже наименования более 12000 артикулов, и по всей стране открыто более 500 магазинов. Из каждого ежедневно в главную контору поступают цифры продаж. То есть, к концу года всего появляется 12000*500*365 ячеек с переменной типа 'Number', то есть более 2 мллиардов.

Но ведь не каждый магазин предлагает клиентам всю номенклатуру по каталогу. В среднем каждый магазин предлагает примерно 600 наименований, и при этом открыт для покупателей 250 дней в году. То есть реально будет заполнено только 600*500*250 ячеек, или 75 миллионов -- в 25 раз меньше максимального числа.

Это означает, что данные не распределены равномерно по нашей базе OLAP. Так, магазин 'Paris Bastille' никогда не продает ботинки с артикулом "23154". Значит для каждого дня в году показатель 'Number' для этого магазина будет пуст.

Из этого следует, что не все шкалы измерений имеют одинаковую важность для показателя . Для того, чтобы улучшить использование дискового пространства и доступ к данным, будет правильно сообщить OLAP-системе подобные свойства переменных.

В Oracle Express для этого требуется всего лишь указать, является ли шкала измерения плотной или разреженной. В нашем примере шкала времени плотна, а шкалы артикулов и магазинов разрежены Далее система сама автоматически соптимзирует управление данными. Так, в Oracle Express создается составная шкала измерения, в которой будут присутствовать все важнейшие пары "артикул -- магазин". Поддерживать эту специальную шкалу придется вручную. Такие шкалы носят название совмещенных (conjoint).

Для OLAP-системы теперь будет иметься всего две шкалы измерения, а для пользователя и разработчика эти детали останутся не видны.

Шаг 8: Узнаем же

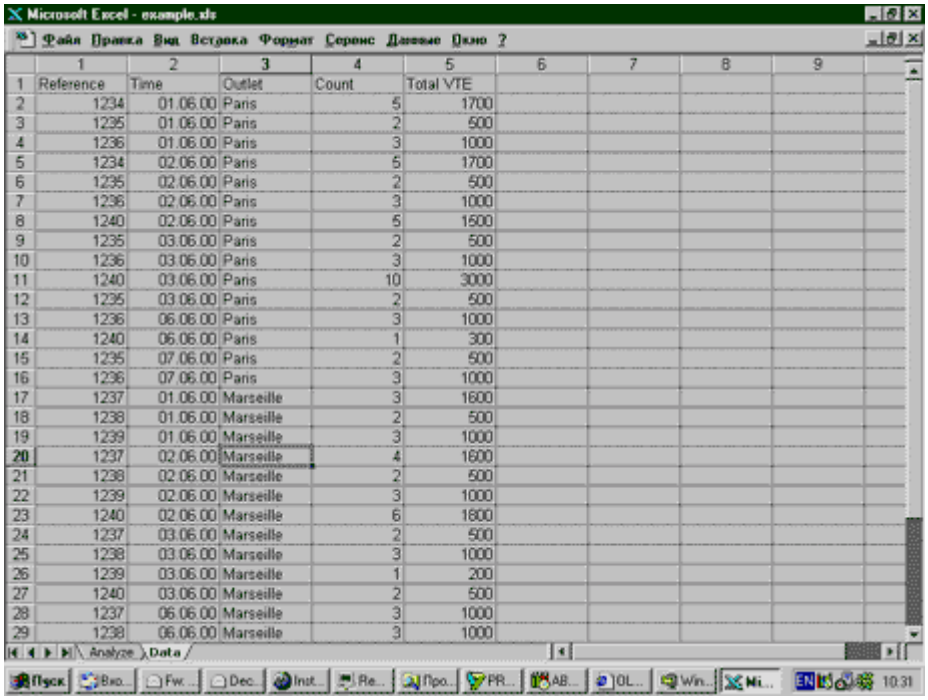
Последний в списке, но не по значимости (Last but not least). Я советую Вам реализовать на практике некоторые идеи, которые мы рассмотрели в этом примере.

"Таблица Pivot " в Microsoft Excel имеет некоторое сходство с OLAP-инструментарием.

Для того, чтобы убедить Вас в этом (если Вы не еще знакомы с этой функцией), пробуете создать маленькую таблицу с 5 колонками (время (time), стиль (style), магазин (shop), значение (value), количество (quantity), как в нашем примере). Загрузите какие-нибудь данные в таблицу Pivot и запустите (launch) ее. Разместите измерения, как Вы пожелаете в строке (Row), столбце (Column) и на странице (Page) и поместите меры (Measures) в область данных. Тем самым, мы имеем инструмент, который позволит нам визуализировать в двух измерениях любую грань куба в нашем примере.

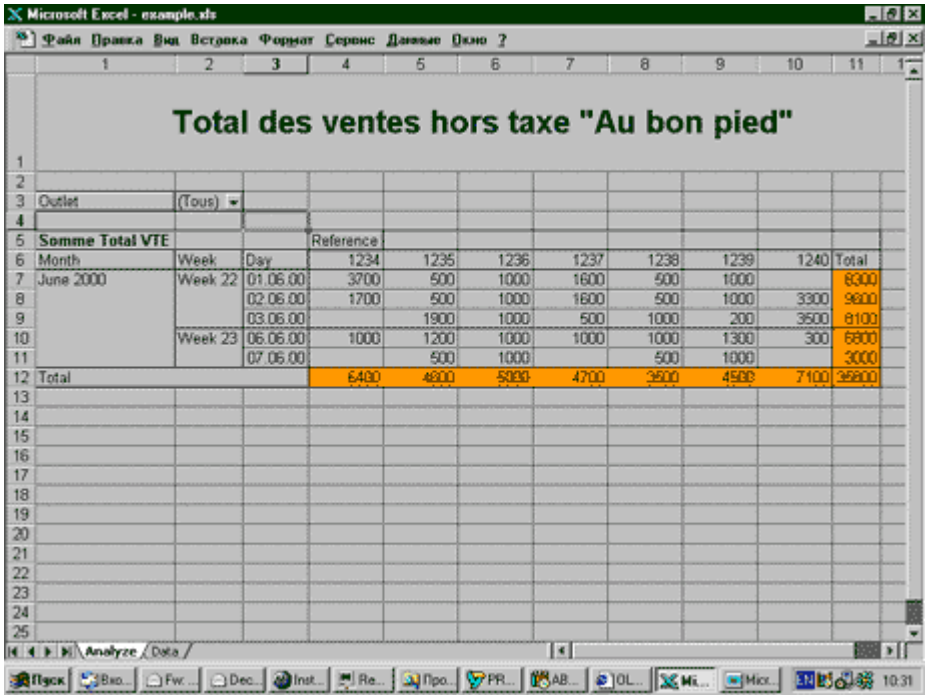
Если Вы не хотите ждать, пока получится результат, разгрузите себе этот файл: **example.xls**.

- Лист Data содержит коммерческие данные, представленные из Парижа, Лиона и Марселя.



	Reference	Time	Outlet	Count	Total VTE
2	1234	01.06.00	Paris	5	1700
3	1235	01.06.00	Paris	2	500
4	1236	01.06.00	Paris	3	1000
5	1234	02.06.00	Paris	5	1700
6	1235	02.06.00	Paris	2	500
7	1236	02.06.00	Paris	3	1000
8	1240	02.06.00	Paris	5	1500
9	1235	03.06.00	Paris	2	500
10	1236	03.06.00	Paris	3	1000
11	1240	03.06.00	Paris	10	3000
12	1235	03.06.00	Paris	2	500
13	1236	06.06.00	Paris	3	1000
14	1240	06.06.00	Paris	1	300
15	1235	07.06.00	Paris	2	500
16	1236	07.06.00	Paris	3	1000
17	1237	01.06.00	Marseille	3	1600
18	1238	01.06.00	Marseille	2	500
19	1239	01.06.00	Marseille	3	1000
20	1237	02.06.00	Marseille	4	1600
21	1238	02.06.00	Marseille	2	500
22	1239	02.06.00	Marseille	3	1000
23	1240	02.06.00	Marseille	6	1800
24	1237	03.06.00	Marseille	2	500
25	1238	03.06.00	Marseille	3	1000
26	1239	03.06.00	Marseille	1	200
27	1240	03.06.00	Marseille	2	500
28	1237	06.06.00	Marseille	3	1000
29	1238	06.06.00	Marseille	3	1000

- Лист Analyze содержит таблицу Pivot, что обеспечивает Вам просмотр всех данных.



Total des ventes hors taxe "Au bon pied"

3	Outlet	(Tous)									
5	Somme Total VTE			Reference							
6	Month	Week	Day	1234	1235	1236	1237	1238	1239	1240	Total
7	June 2000	Week 22	01.06.00	3700	500	1000	1600	500	1000		8300
8			02.06.00	1700	500	1000	1600	500	1000	3300	9600
9			03.06.00		1900	1000	500	1000	200	3500	8100
10		Week 23	06.06.00	1000	1200	1000	1000	1000	1300	300	6800
11			07.06.00		500	1000		500	1000		3000
12	Total			6400	4800	5000	4700	3500	4500	7100	35800

Тот поступит правильно, кто останется с нами и далее, а я рассчитываю на Вашу конструктивную критику.

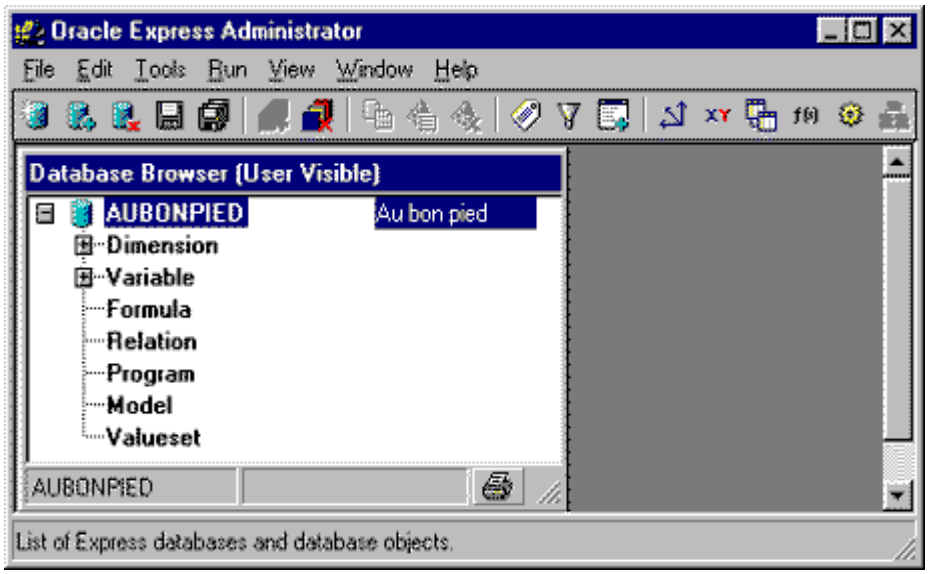
OLAP-практикум: осваиваем Oracle Express

Шаг 1 : Структура базы данных

До настоящего момента пример базы данных для "Best Foot Forward" из предыдущей части существовал лишь на бумаге. Теперь пора сделать его более "живым", воспользовавшись продуктами фирмы Oracle. Oracle Express я выбрал не случайно: я давно работаю с этой системой и знаю ее особенности.

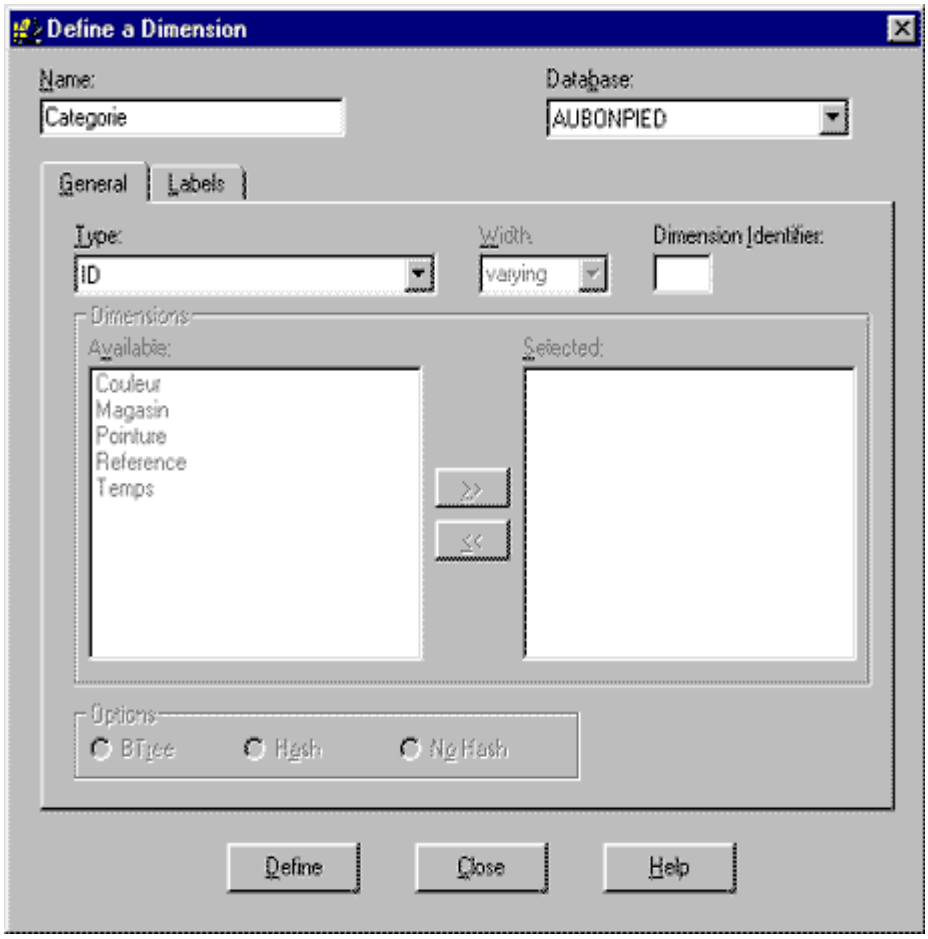
Весьма быстро базу данных OLAP можно построить, пользуясь средой Oracle Express Administrator. На первом этапе создания базы данных для "Best Foot Forward" мы так и поступим.

Подсоединимся к Oracle Express (возможно, с другой машины), и в первом же окошке обнаружим возможность создать базу данных. Введем имя и описание и получим следующее:



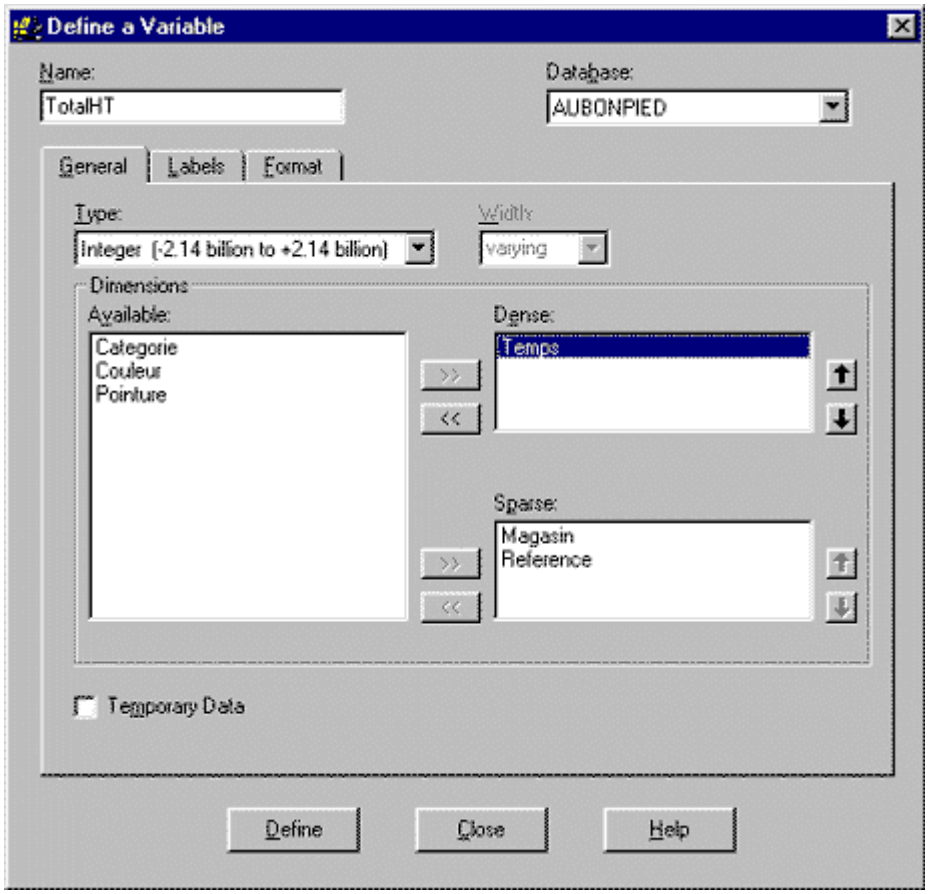
Это окошко не вызывает у нас никаких затруднений, поскольку все важнейшие понятия -- шкала измерения, переменная-показатель, формула и отношение -- нам уже известны. Если что-то забыли, смело обращайтесь к словарю.

Нажатием правой кнопки на надписи "Dimension" мы попадем в диалоговое окошко, откуда можем создавать все шкалы базы данных:



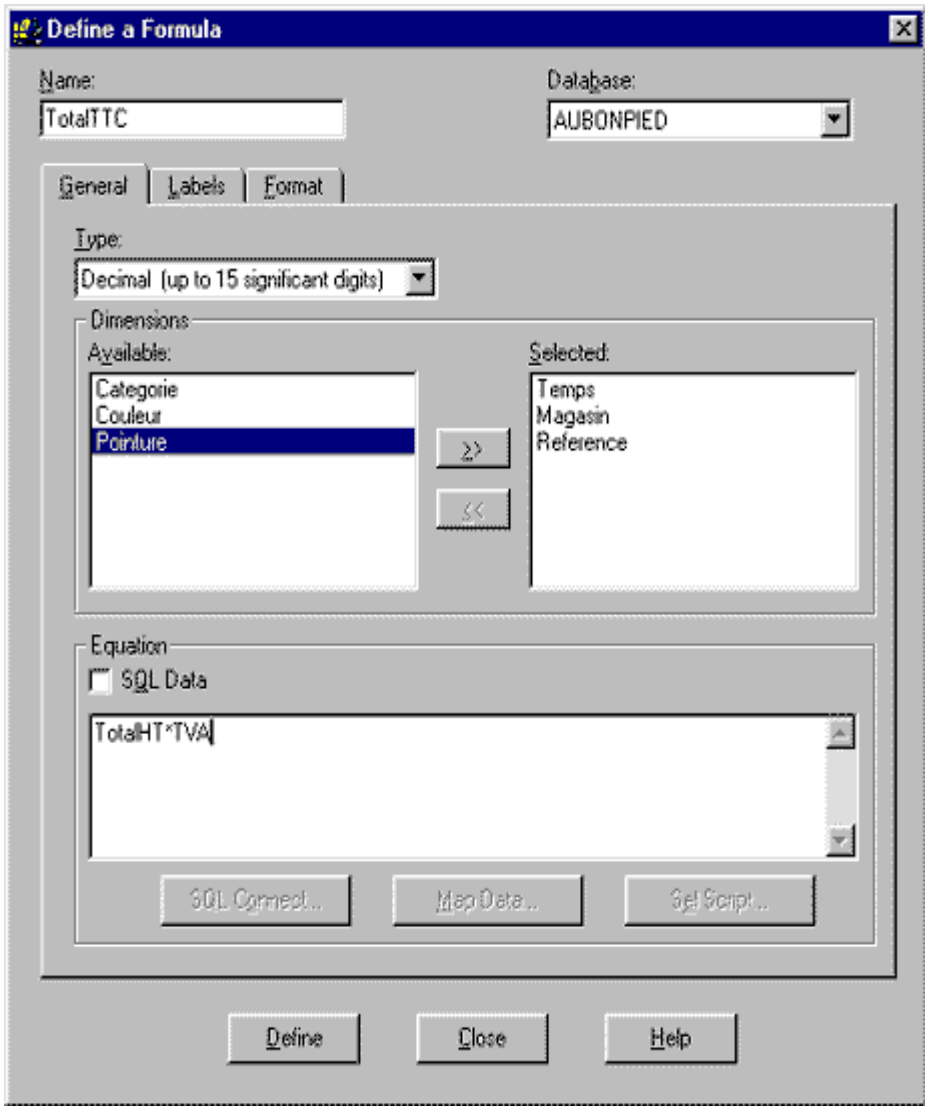
Закладка "General" дает возможность определить тип каждой шкалы (идентификатор, число, текст, ...). Закладка "Labels" дает возможность определить для каждой шкалы длинную и короткую метки -- их будет видеть конечный пользователь. Прочие возможности этого окна позволяют задать вручную или оптимизировать совмещенные шкалы.

Нажав правой кнопкой на позицию "Variable", мы сможем завести переменные Quantity, TotalVTE и VAT. Для каждой из них укажем вид шкалы -- плотный или разреженный.



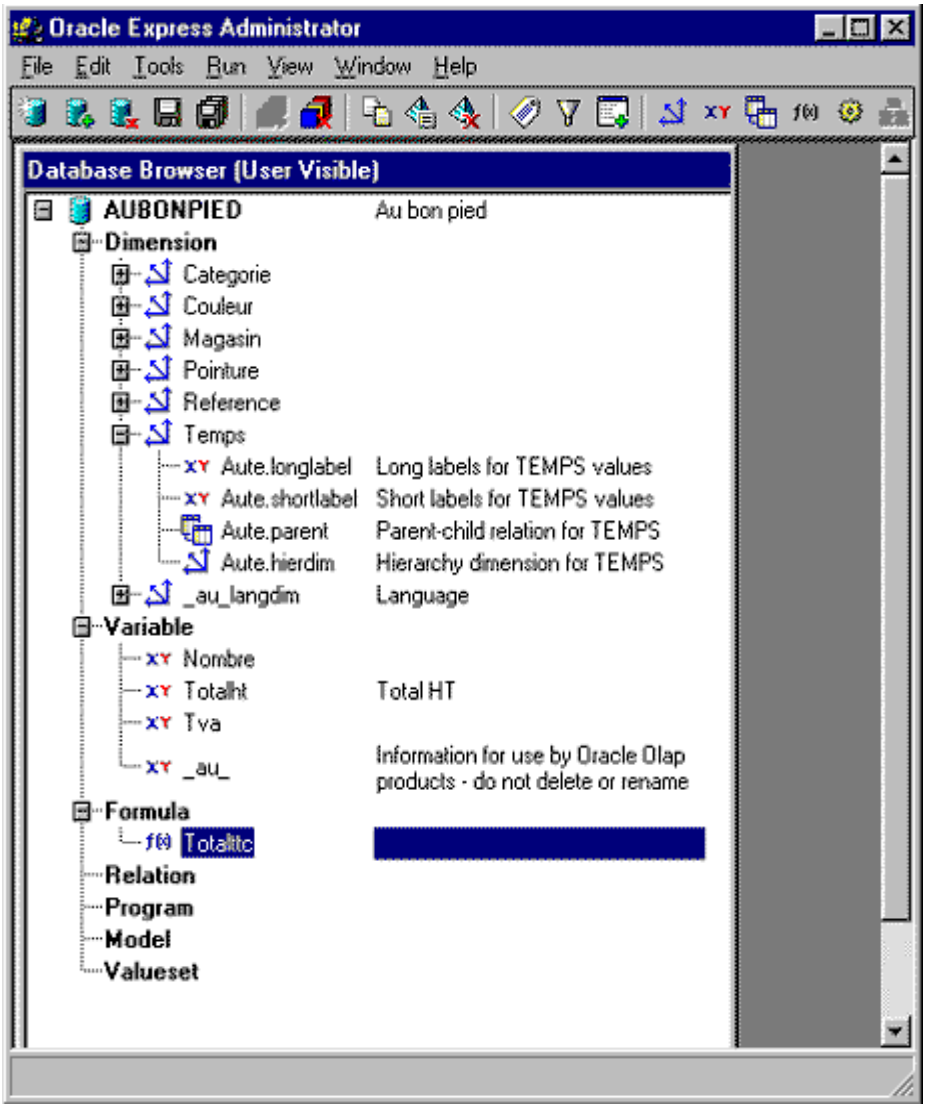
В этом примере для переменной TotalVTE будет указана плотная шкала Time, и разреженные шкалы Outlet и Reference (Магазин и Артикул).

Наконец, как и следовало ожидать, нажав правой кнопкой на "Formula", можно создавать формулы. Определи формулуTotalVTI следующим образом:



Сама формула (в нашем случае TotalVTE * TVA) записывается на специальном языке. За некоторыми примерами рекомендую обратиться к шагу 4.

Теперь структура базы данных построена. Общий вид базы данных представлен в главном окошке.



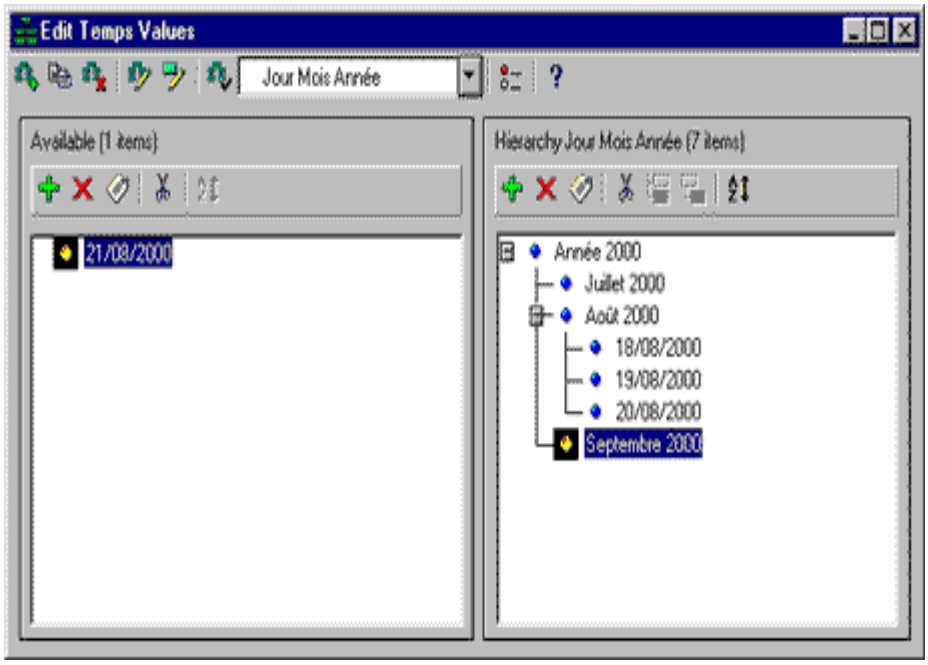
Теперь эту структуру нужно наполнить данными. Перейдем к следующему шагу !

Шаг 2 : Загружаем данные

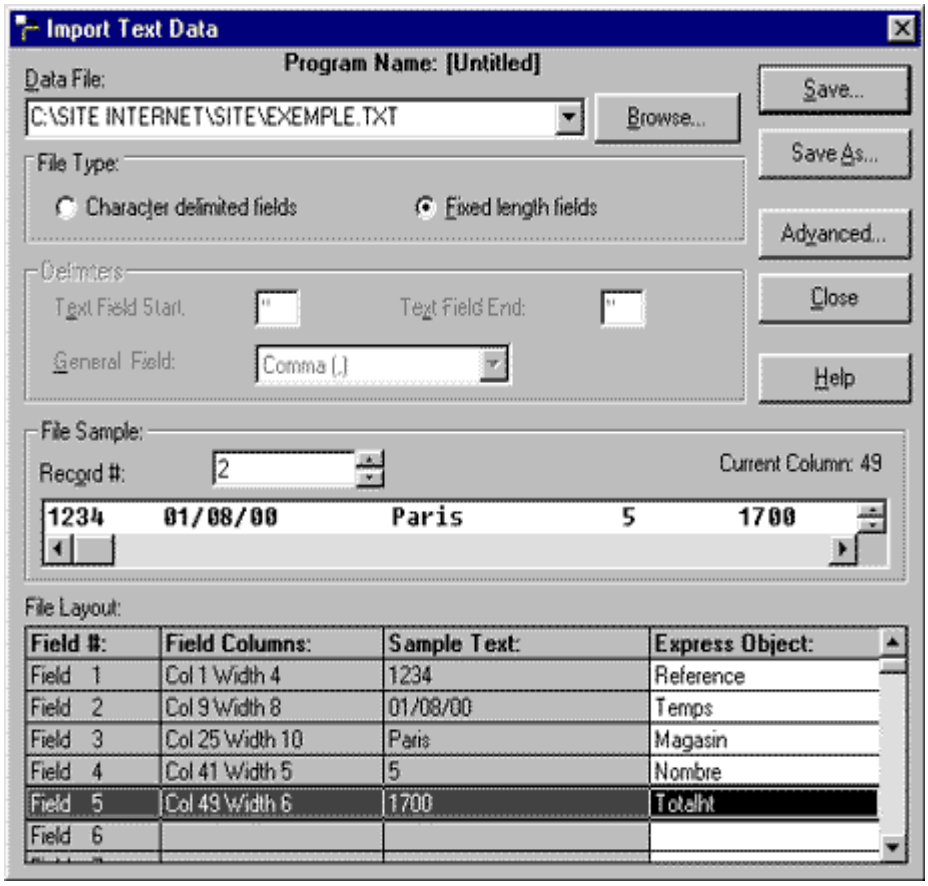
У нас имеется два вида данных:

- Начальные данные для приведения базы OLAP в рабочее состояние. Это позиции на шкале и метки. Для нашего случая опять-таки проще всего воспользоваться средой Express Administrator, хотя это и не обязательно.
- Цифровые данные, присылаемые ежедневно магазинами. Предположим, что эти данные поступают в виде плоских файлов, и тогда наша задача -- организовать загрузку этих файлов в базу OLAP.

Для того, чтобы в Express Administrator заполнить шкалу измерений позициями, нам, как обычно, нужно нажать правой кнопкой на метку измерения и выбрать "Edit Values". В открывшемся окошке можно занести в систему все позиции, и, кроме того, задать иерархии для этой шкалы. В нашем примере это будут две шкалы для шкалы Time.



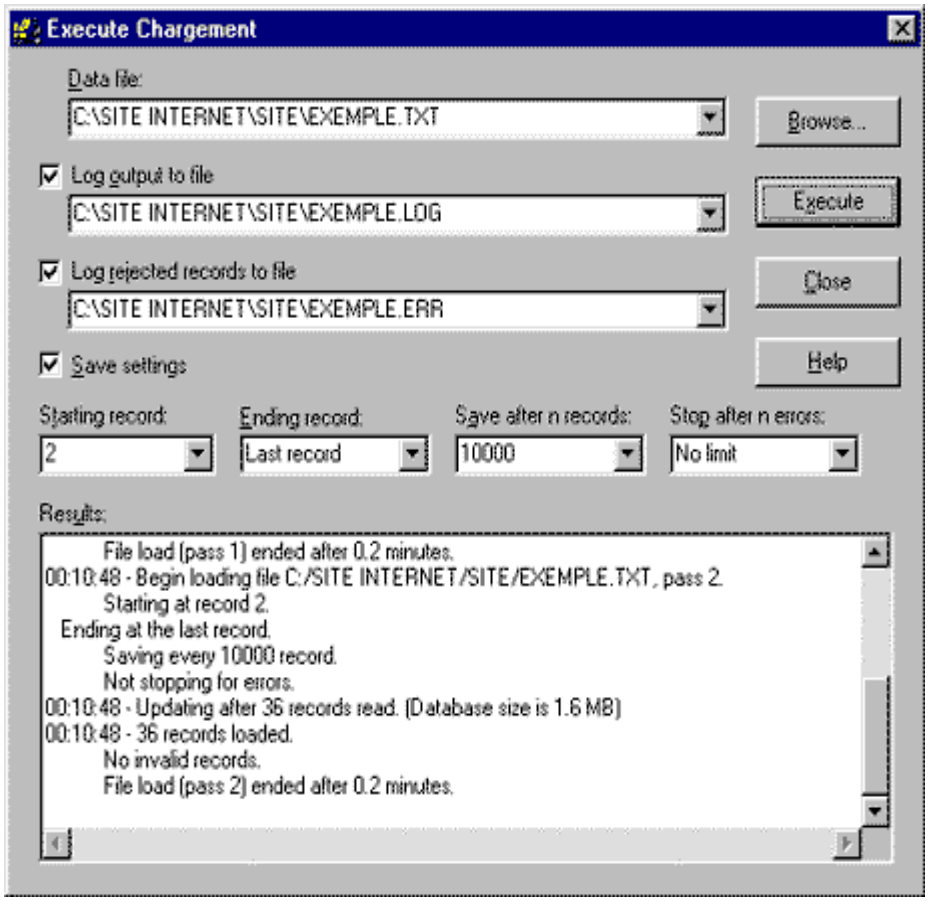
Далее нам нужно организовать ежедневную автоматическую загрузку плоских файлов. Для этого нам придется написать программу и сохранить эту программу в базе данных. Составить программу нам поможет помощник из среды Administrator. Действуя через него, нам понадобится описать формат плоского файла примерно в таком окошке :



Здесь сказано, что каждая строка файла содержит 5 полей. В зависимости от шкал перечисление полей завершается значением переменной Quantity или TotalVTE.

Теперь перейдем к средству построения соответствующей программы на 4GL. Начнем со

следующего окошка :



Теперь ежедневные данные попадают в базу данных. Нам осталось сагрегировать эти данные вдоль шкалы времени. Свертку поможет построить специальный помощник в среде Administrator. Все, что от нас требуется -- указать, какие переменные агрегировать, иерархии и позиции на шкале, которые будут участвовать в свертке.

Использование этого помощника настолько просто, что нам неинтересно на нем останавливаться. Увы, но программы, которые он строит, очень длинные и сложные ввиду использования большого числа параметров. В рамках шага 4 я приведу очень простые примеры использования языка Oracle Express.

Теперь наше приложение стало самодостаточным : каждый день выполняется загрузка данных, сразу вслед за которой пересчитываются агрегаты. Конечные пользователи получили возможность подсоединяться к базе и анализировать свои данные. Как реализовать эту возможность, станет темой для нашего следующего шага.

Шаг 3 : Анализируем данные

Для анализа данных из нашей базы для "Best Foot Forward" существует много продуктов :

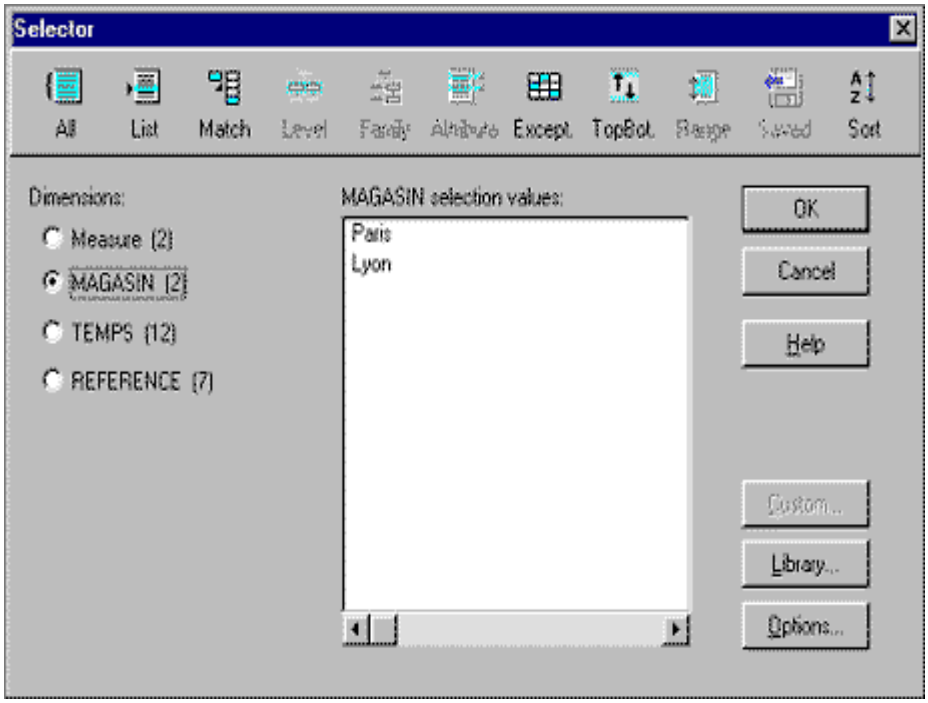
- Express Analyzer позволяет пользователям работать с данными из Oracle Express в режиме "клиент/сервер". Примеры мы увидим на этой странице ниже.
- Express Object позволяет разработчикам строить завершенное "клиент/серверное" приложение на основе данных из Oracle Express.
- Специальные вставки типа "Add-in" позволяют напрямую обращаться к данным Oracle

Express из Microsoft Excel.

- Такие продукты Oracle, как Web Agent, можно использовать для работы с данными Express через браузер.
- Можно использовать и некоторые другие продукты, к примеру, Business Objects, позволяющие осуществлять прямой доступ к данным из Oracle Express.

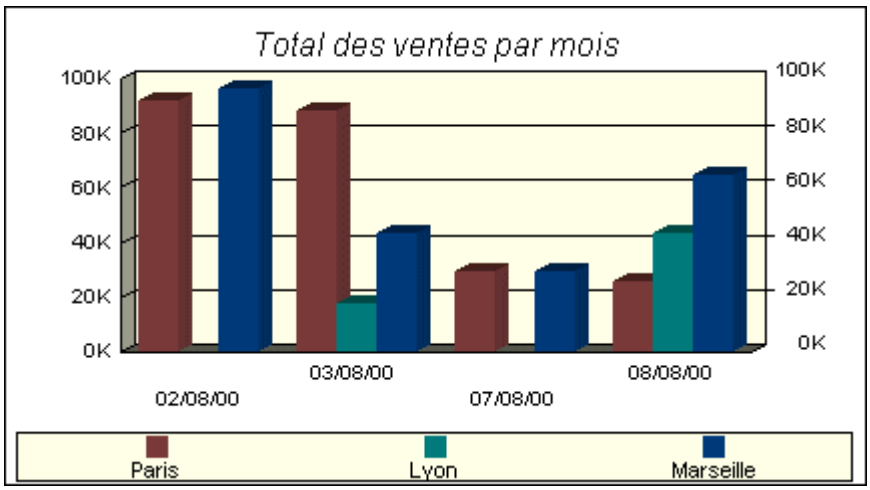
Для того, чтобы просматривать данные из Express с помощью Analyzer, нам требуется только указать нужные меры. Analyzer построит таблицу или график значений переменных и связанных с ними шкал. Выбранные меры -- это положения на специальной шкале под названием "Measure".

Для выбора нужных положений на каждой шкале используется средство Oracle, называемое Selector. С помощью этого средства можно делать достаточно сложный выбор: например, все месяцы в году; все артикулы обуви, проданной в сентябре в количестве не менее 10 пар; 5 наилучших магазинов. С помощью этого средства выборы можно сохранять и восстанавливать. Первое диалоговое окно средства для выбора Selector показано ниже.



Выбрав меры и отобрав позиции на шкале, можно посмотреть таблицу или график. Вот два примера для базы данных "Best Foot Forward" :

1235	NOMBRE		TOTALTTC	
	Paris	Lyon	Paris	Lyon
– Année 2000	8	3	39 200,00	41 160,00
– Août 2000	8	3	39 200,00	41 160,00
01/08/00	2		9 800,00	
02/08/00	2		9 800,00	
03/08/00	2	1	9 800,00	17 640,00
07/08/00	2		9 800,00	
08/08/00		2		23 520,00



Когда на экран выводится рисунок, мы можем мышкой подтаскивать ту или иную шкалу в область строки, столбца или страницы (верхний левый угол). Мы можем перемещаться вверх или вниз по иерархии, нажимая на значок + или -, агрегировать шкалы и так далее. Все это выполняется быстро и прото, так что пользователь может выбирать форму анализа сам.

Я надеюсь, что приведенные примеры, весьма конкретного характера, помогли вам понять возможности баз данных OLAP.

Теперь те, кто поотважнее, могут перейти к следующему шагу, где приводятся некоторые примеры 4GL-языка Oracle Express и даются к этим примерам комментарии.

Шаг 4 : Программирование на 4GL

В Oracle Express имеется специальный язык, с помощью которого в базе можно делать все, что угодно. Это язык типа 4GL, и позволяет он:

- Целиком определять многомерную базу данных, включая шкалы, переменные, формулы и так далее.
- Заносить в базу данные -- например, с помощью загрузки плоских файлов, или путем непосредственного подключения к реляционной БД.
- Агрегировать данные по иерархиям, или же задавать весьма сложные вычисления.
- Выбирать требуемые данные для построения плоских файлов или сложных отчетов.

Этот язык 4GL используют при взаимодействии с базой данных такие продукты, как Express Administrator или же Express Analyzer. На этом языке можно писать пакетные задания, для того, чтобы запускать их в ночное время, и так далее.

Я предлагаю вашему вниманию несколько примеров работы с базой данных Best Foot Forward. Примеры снабжены комментариями. Пожалуйста, обратите внимание на простоту примеров. Моею целью было продемонстрировать всего лишь некоторые особенности языка. Тем не менее, примеры нормально компилируются и исполняются в Oracle Express.

1. Создание базы данных для Best Foot Forward

Этот первый мой пример иллюстрирует возможность создания упрощенной версии базы данных Best Foot Forward. В ней имеются некоторые шкалы, переменные и формулы, но нет иерархии для шкалы времени Time

" Создаем базу данных
DATABASE CREATE 'BESTFOOTFORWARD.DB'

" Создаем размерности
" Указываем имена и типы данных
DEFINE CATEGORY DIMENSION ID
DEFINE COLOR DIMENSION ID
DEFINE OUTLET DIMENSION TEXT WIDTH 32
DEFINE SIZE DIMENSION ID
DEFINE REFERENCE DIMENSION INTEGER
DEFINE TIME DIMENSION ID

" Создаем переменные
" Указываем имена, типы данных и ассоциированные размерности
" Указываем плотные и разреженные размерности

DEFINE QUANTITY VARIABLE DECIMAL >
DEFINE TOTALVTE VARIABLE DECIMAL >
DEFINE VAT VARIABLE DECIMAL

" Создаем формулы
" Указываем имена, типы данных и ассоциированные размерности
" Пишем текст формулы
DEFINE TOTALVTI FORMULA DECIMAL
EQ TotalVTE*VAT

2. Загрузка данных из файла

Эта программа загрузит данные из файла "exemple.txt", о котором шла речь в **шаге 8** раздела "Знакомимся с OLAP: шаг за шагом". Этот сценарий создаст новый объект -- программу. Ее можно запустить на выполнение командой "call <имя программы>".

DEFINE LOAD_EXAMPLE PROGRAM
PROGRAM
variable _funit integer
variable _i integer

" Открываем файл данных
_funit = fileopen('exemple.txt' read)

" Первая строка - дескриптор, отбрасываем ее
fileread _funit stopafter 1

if not filenext(_funit)
" Ничего не делаем, если строка только одна
then goto DONE

" Для каждой строки файла
_i=0
while true
do
" Грузим данные в размерности и переменные
" Для файла известного типа, указываем размер и позицию каждого столбца
fileview _funit ruled -

```
_i = _i + 1 -
COL 1 WIDTH 4 STRIP APPEND REFERENCE -
COL 9 WIDTH 8 STRIP APPEND TIME -
COL 25 WIDTH 10 STRIP APPEND OUTLET -
COL 41 WIDTH 5 STRIP QUANTITY -
COL 49 WIDTH 6 STRIP TOTALVTE

" Здесь добавим немного специального кода для каждой строки файла

" Следующая запись
if not filenext(_funit)
then goto DONE
doend

DONE:
" Закрываем файл
fileclose _funit
update

" Выводим время дня и число записей
show joinchars (tod ' - ' _i ' records created')
END
```

3. Загрузка данных из реляционной базы

На этом примере показано, как можно загрузить номенклатуру артикулов из реляционной таблицы под названием "model". Многомерная база может обращаться к реляционной и динамически, в процессе работы приложения.

```
DEFINE LOAD_MODEL PROGRAM
PROGRAM
variable _i integer

" подключаемся к Oracle8i
sql.dbms='oracle'
sqlmessages=yes
sql connect 'admin' identified by 'pass'
if sqlcode ne 0
then signal err_connect joinchars('Connection problem ' sqlerrm)

" Объявляем курсор
sql declare C1 cursor for -
SELECT id_model, desc_model FROM model

if SQLCODE ne 0
then signal err_req1 joinchars('SQL error ' sqlerrm)

sql open C1
if SQLCODE ne 0
then signal err_req2 joinchars('SQL error ' sqlerrm)

" Для каждой строки в таблице MODEL
_i = 0
while SQLCODE eq 0
```

```
do
_i = _i + 1
sql FETCH C1 INTO APPEND modele, mod.longlabel
if sqlcode ne 0 and sqlcode ne 100
then signal err_req3 joinchars('SQL error ' sqlerrm)
doend
update

" Закрываем курсор и базу данных
sql close C1
sql disconnect
END
```

4. Агрегирование данных

Этот пример, последний, демонстрирует возможность выполнения свертки данных по имеющейся иерархии.

```
DEFINE ROLL PROGRAM
PROGRAM

arg vDay text " Параметр: дни для агрегации

" Перед rollup-ом, определяем все месяцы в году и
" все дни в месяце в размерности Time.
" Это делается командой limit для временной размерности и всех с ней связанных

limit time to vJours " Определяем дни в параметре
limit aute.hierdim to 'DMY' " Определяем иерархию Day-Month-Year

limit time to ancestors using aute.parent " Месяцы и годы дней
limit time to children using aute.parent " Дни месяцев и месяцы года

" Для других размерностей сохраняем все положения
limit outlet to all
limit reference to all

show joinchars(tod, ' - Rollup along time')
" аггрегируем
rollup quantity over time using aute.parent
rollup totalvte over time using aute.parent

END
```

Ну вот и все, для начала. Благодарю вас за то, что вытерпели мой английский. Хорошо, что вы оставались с нами все это время -- ведь я надеюсь на конструктивную критику от читателей. Написать мне можно либо по электронной почте, либо используя мою замечательную форму, которую я создал своими руками..

Чуть-чуть подробнее об OLAP

Когда появился OLAP?

Термин **OLAP** - On Line Analytical Processing (оперативная аналитическая обработка данных) возник в 1993. Его придумал E.F. Codd, отец реляционных баз данных (в статье E.F. Codd, S.B. Codd (его жена) и C.T. Salley “Providing OLAP to User-Analysts: An IT Mandate”.) С тех пор возникло множество новых акронимов: от ROLAP до MOLAP через DOLAP. Все эти термины приведены в глоссарии.

Правила OLAP

Первоначально было определено 12 правил OLAP, которые определяли эту технологию. Все OLAP-продукты должны были им следовать. Вот эти 12 правил:

- Multidimensional model (Многомерная модель)
  Многомерные концептуальные представления.
  Данные для пользователя должны быть представлены в многомерной парадигме.
- Transparency of the server (Прозрачность от сервера)
  ПРОЗРАЧНОСТЬ
  Пользователь не обязан знать, что он использует базу данных OLAP.
- Accessibility (Доступность)
  Для поддержки запросов программное средство должно выбирать самый лучший источник данных.
- Stable access performance (Постоянность характеристик производительности)
  Согласованная производительность отчетов
  Производительность должна быть одинаковой, независимо от числа используемых измерений
- Client server architecture (Архитектура клиент/сервер)
  Программные средства должны работать в архитектуре клиент/сервер.
- Generic Dimensionality (Общность измерений)
  Равноправность измерений
  Все измерения должны быть равноправными; не может быть”крена” в сторону какого-то одного измерения.
- Management of data sparsity (Управление разреженными данными)
  Динамическая обработка разреженных данных
  Нулевые (null) значения должны храниться эффективно.
- Multi-user (Наличие многих пользователей)
  Программные средства должны поддерживать более одного пользователя !
- Operation on dimension (Операции с измерениями)
  Отсутствие ограничений на операции с разными измерениями
  Правила агрегации единообразно и согласованно применяются ко всем измерениям.
- Intuitive manipulation of data (Интуитивное манипулирование данными)
  Пользовательские представления данных должны содержать все необходимое для того, что бы он не прибегал к использованию меню и других сложных элементов интерфейса.
- Flexible posting and editing (Гибкое позиционирование и отчетность)
  Гибкая система отчетности
  Пользователи должны иметь возможность представлять данные в любой удобной для них форме.
- Multiple dimensions and levels (Многожественность измерений и уровней)
  Неограниченное число измерений и уровней агрегации
  Модель не должна иметь ограничений на число измерений и уровней агрегации

В настоящее время эти 12 правил расширились до 18 главных правил, а всего их около 300!

Для чего применяется OLAP?

OLAP-технология может применяться в широком спектре областей:

- Коммерческий анализ и маркетинг: любимая область. Рассматриваемый в курсе пример часто является отправной точкой для приложений такого типа, когда Вам требуется изучить тома данных о продажах изделий по регионам во времени.
- Консолидация данных: непосредственное использование одной из возможностей OLAP.
- Поддержка принятия решений: попытка предсказания изменения доходов и расходов.
- Служба качественного анализа.

Кому нужен OLAP?

Все управленцы (менеджеры) должны быть заинтересованы в применении OLAP:

- Благодаря возможности получения и изучения объединенных, консолидированных данных, коммерческие менеджеры могут сравнивать фактическое положение дел с текущим и прошлым бюджетами по изделиям, клиентам и каналам сбыта.
- Благодаря большому объему сохраняемых данных, производственные менеджеры и инженеры могут изучать ежедневные сведения и коэффициенты о функционировании чего-либо и отыскивать проблемы, требующие разрешения.

Глоссарий OLAP

OLAP технология использует значительное число точных терминов, обозначающих элементы многомерных структур. Кроме того, находясь в мире OLAP, Вы столкнетесь кое с какими специфическими аббревиатурами. Поэтому нужен глоссарий.

Этот глоссарий предназначен только для того, чтобы осветить первые шаги, он не претендует на статус полного и строгого источника ссылок. Ваша (конструктивная J !) критика поможет ему в его совершенствовании.

От редактора

> Воспользуемся предложением автора и попросим прокомментировать эти термины Ольгу Горчинскую, технического консультанта Oracle СНГ по OLAP и хранилищам данных. Она скажет, как эти термины понимаются в русскоязычном OLAP. Я надеюсь, что все Вы присоединитесь с моей искренней благодарности Ольге. Ее замечания выделены особым цветом.

Aggregate (Агрегирование)

> Операция по вычислению значений, связанных с родительскими позициями в иерархических измерениях (**hierarchical dimensions**). Это объединение, консолидация может быть суммированием (addition), усреднением (averaging) или каким-либо другим более сложным действием для получения вторичного интересующего аналитика значения.
> Операция консолидации значений многомерного показателя по некоторой иерархии, определенной между значениями измерения. В качестве такой операции может

использоваться суммирование, вычисление среднего, определение наименьшего или наибольшего и т.п. Например, можно агрегировать значения объема продаж по измерению“Временные периоды", имея эти значения для каждого дня и суммируя их для вычисления агрегированной величины для месяцев.

Attribute (атрибут, признак)

Сведение, характеризующий каждую позицию **(position)** измерения.
Дополнительная характеристика значения измерения. Например, для измерения “Дни” может быть определен атрибут “Тип дня”, с помощью которого каждый день характеризуется как выходной или рабочий. Атрибуты используются для формирования запросов к многомерным данным, агрегирования и определения формул.

Axis (ось)

Синоним измерения **(dimension)**

Cell (Ячейка)

Часть данных, определенных ее позицией в каждом измерении. Ячейки гиперкуба **(hypercube)** могут быть пусты или полны. Когда значительное число ячеек не содержат данных, говорят, что он "разрежен" ("sparse").
Элемент многомерного массива (гиперкуба), в котором хранятся значения многомерного показателя. Ячейка гиперкуба может быть пуста или заполнена. Гиперкуб с большим количеством пустых ячеек называют разреженным.

Cube (Куб)

Наиболее часто используемый синоним гиперкуба **(hypercube)**
Обычно используется в качестве синонима гиперкуба.

Datamart (витрина данных)

Данные по одному из направлений деятельности компании. Несколько витрин данных формируют хранилище данных конкретной фирмы.
База данных, отвечающая тем же требованиям, что и хранилище данных, но функционально-ориентированная и, как правило, содержащая данные по одному из направлений деятельности организации,. В отличие от хранилища, нейтрального к приложениям, в витрине данных информация хранится специальным образом, оптимизировано с точки зрения решения конкретных аналитических задач или деятельности некоторого подразделения или группы аналитиков.

Datawarehouse (хранилище данных)

Склад (хранилище) данных. Термин обозначает все данные компании, сохраненные в форме, пригодной для обработки данных.
База данных, содержащая информацию для поддержки аналитической деятельности. Автором концепции хранилищ данных (Data Warehouse) является Б.Инмон, который определил хранилища данных, как предметно ориентированные, интегрированные, неизменчивые, поддерживающие хронологию наборы данных, организованные для целей поддержки управления, призванные выступать в роли “единого и единственного источника истины”, обеспечивающего руководителей и аналитиков достоверной информацией, необходимой для оперативного анализа и принятия решений.

Dimension (измерение)

Собрание (коллекция) данных одного и того же типа, что позволяет структурировать многомерную базу данных. Измерение иногда упоминается как ось. В мере **(measure)** каждая ячейка данных ассоциирована с одной единственной позицией (**position**) в каждом измерении. Время (time), местоположение (location) и изделие (product) представляют собой классические измерения.
Один из основных объектов многомерной модели данных. Измерение – это список значений, относящихся к одному и тому же типу данных с точки зрения пользователя. Например, все дни, месяцы, кварталы и годы с точки зрения пользователя относятся к одному и тому же типу “Временные периоды”; список городов, регионов и стран образуют измерение “География”. Измерения используются в качестве индексов для идентификации элементов многомерного массива (гиперкуба), в котором хранятся значения многомерных показателей.

DOLAP

Настольный OLAP. Маломощные (small) OLAP-продукты для локального многомерного анализа (Desktop OLAP). Для применения с мини многомерной базой данных, продуцированной Personal Express, или с извлечением из куба данных (datacube), продуцированным Business Objects.

Desktop OLAP (Настольный или персональный OLAP). OLAP-продукты для локального многомерного анализа, не поддерживающие многопользовательский режим. Примером персональной многомерной СУБД является Personal Express.

DSS - Decision Support System. (система поддержки принятия решений)

Система поиска и представления данных, специально используемая в процессах принятия решений. Французский эквивалент - SIAD, или " Systime d'Information d'Aide a la Decision".

EIS (исполнительная информационная система)

Английский термин, чаще обычно используется DSS.

EIS и DSS не одно и то же. EIS – информационная система для руководства, для принятия управленческих решений. Обычно такая система должна быть достаточно максимально простой в использовании и обеспечивать быстрый отклик, аналитические требования при этом довольно ограничены. DSS – система поддержки принятия решений в более широком смысле, для различных областей и технологических процессов .

FASMI - Fast Analysis of Shared Multidimensional Information (быстрый анализ разделяемой многомерной информации)

Все эти пять слов имеют их место в определении **OLAP**-технологии.

Определение понятия OLAP в виде пяти критериев (**Fast, Analysis, Shared, Multidimensional, Information)**, которым должны удовлетворять продукты, попадающие в эту категорию. Тест FASMI был разработан в качестве альтернативы известным 18 правилам Кодда, определяющим OLAP-систему.

Formula (формула)

Виртуальный гиперкуб. Значения, которые часто вычисляются "на лету" (on the fly) и не сохраняются в базе данных.

Hierarchy (иерархия)

Позиции измерения, организованные каскадом отношений (relationships) "один к многим". Этот способ организации данных сооьветствует логическому дереву, где каждый член только имеет одного родителя, но может иметь разное число потомков.

Множество значений измерения с заданным для них отношением “многие к одному”. Такая организация данных соответствует древовидной структуре, в которой каждый элемент имеет родно одного родителя и любое число потомков.

Hierarchy level (уровень иерархии)

В иерархии позиции классифицируются по уровням. Все позиции уровня соответствуют уникальной классификации. Например, в измерении "Время" ("Time") первый уровень составляют дни (days), второй - месяцы (months), третий - годы (years).

Множество элементов иерархии, находящихся на одном и том же расстоянии от корня иерархической структуры. Часто отдельный уровень иерархического измерения соответствует некоторому понятию предметной области. Например, для иерархического измерения “Временные периоды” (множество различных временных периодов) один из уровней может соответствовать понятию “Дни”, другой – понятию “Месяцы” и т.п.

HOLAP

Гибрид OLAP. Определяет многомерные инструменты анализа, которые прозрачным для пользователя способом сохраняют данные, или в реляционной, или в многомерный базе данных.

Гибридный OLAP. OLAP-технология, при которой часть анализируемых данных хранится в многомерной базе данных, а часть в реляционной базе данных. Инструментальные средства, поддерживающие эту технологию, должны обеспечивать требование прозрачности данных для пользователя: на логическом уровне пользователь всегда работает с многомерными данными.

Hypercube (гиперкуб)

Многомерная конструкция, сформированная соединением нескольких измерений. Каждая ячейка (**cell**) определена отдельным членом из каждого измерения (**dimension**).
Один из основных объектов многомерной модели данных. Многомерный массив, используемый для хранения значений многомерного показателя. В качестве индексов такого многомерного массива используются измерения (**dimension**). Отдельный элемент многомерного массива называется ячейкой гиперкуба (**cell**). Например, с помощью гиперкуба можно моделировать трехмерный показатель “Объем продаж” с измерениями “Временной период”, “Продукт”, “Регион”.

MDB(многомерная база данных)
Осуществляет хранение, манипуляцию и воспроизводство многомерных данных.
База данных, поддерживающая многомерную логическую модель данных. Основными объектами многомерной базы данных являются измерения и многомерные показатели (или гиперкубы).

Measure (мера, Показатель)
Гиперкуб. Наиболее часто это целый (integer) или десятичный (decimal) тип, структурированный измерениями. "Жалованье" (Salary), "Ценность" (Value) и "Количество" (Quantity) - классические меры.
Многомерный показатель. Иногда, например в Oracle Express, используется для обозначения любого многомерного показателя: как хранимого непосредственно (гиперкуб или переменная), так и вычисляемого “на лету” по заданному соотношению (Формула).

MOLAP
Многомерный OLAP. Этот термин определяет специфическое декартово произведение (cartesian stocking). Фактически MOLAP противопоставляется ROLAP. Для первого соединения (joints) уже установлены, что повышает производительность. Для второго соединения между таблицами устанавливаются в момент запроса.
Многомерный OLAP. OLAP-технология, основанная на хранении данных под управлением специализированных многомерных СУБД.

Multicube (Мультикуб)
Многомерная конструкция, сформированная из нескольких гиперкубов, имеющих общие (sharing - разделяемые) измерения

Multidimensional (Многомерный)
Структура данных, имеющая по крайней мере три независимых измерения.
Двухмерный массив тоже является многомерным!

OLAP (оперативная аналитическая обработка данных)
Термин определяет категорию приложений и технологий, которые с целью анализа обеспечивают собирание (collection - коллекционирование), хранение, манипулирование и воспроизводство многомерных данных. Другое определение содержится в акрониме **FASMI** (Fast Analysis of Shared Multidimensional Information - быстрый анализ разделяемой многомерной информации). OLAP-инструментарий должен соответствовать 12 правилам, которые **были приведены выше**.
Категория инструментальных средств и приложений, поддерживающих технологию динамического многомерного анализа данных, включая хранение, управление и манипулирование многомерными показателями, предоставление пользователям простых и удобных средств оперативного доступа к этим данным, динамического построения отчетов, графиков, формирования сложных запросов, агрегирования, прогнозирования и выполнения других аналитических действий. Для определения принадлежности продукта к категории OLAP можно использовать известные 18 правил Е.Кодда или альтернативный подход – тест FASMI.

Position (позиция)
Значение измерения (**dimension**).
Термин “Позиция” не очень удачный и нигде не используется, лучше говорить “значение измерения”

RDBMS - Relational database Management System (система управления реляционной базой данных)

Система обеспечивает хранение, манипулирование и воспроизводство данных, занесенных в реляционные таблицы. Французский эквивалент - **SGBDR**.
Система хранения, управления и манипулирования данными, поддерживающая на логическом уровне представления информации реляционную модель данных

Relation (Отношение)
Наличие отношения между позициями двух измерений позволяет делать вычисления на лету (on-the-fly) и таким образом легко создавать формулы.
Объект многомерной модели данных, с помощью которого моделируется отношение (обычно “многие к одному”) между значениями двух измерений. Отношения используются для формирования запросов, агрегирования данных, определения формул.

ROLAP
Реляционный OLAP. Определяет одну или несколько звездоподобных схем, хранимых в реляционных базах данных. Эта технология позволяет реализовать многомерный анализ с данными, находящимися в реляционных базах данных.
Реляционный OLAP. OLAP-технология, основанная на хранении многомерной информации в реляционных базах данных.

SGBDR
Французский эквивалент термина **RDBMS**

SIAD
Французский эквивалент термина **EIS**

Star schema (звездоподобная или звездная схема)
Соглашение по организации данных в реляционной базе данных. В центре (middle - середин) находится фактографическая таблица, столбцы которой являются многомерными мерами. Лучи (branch - ветка) звезды, которые исходят от фактографической таблицы, соответствуют измерениям. Концептуальная модель данных представляет собой звездоподобную схему.
Специальная организация реляционных таблиц, удобная для хранения многомерных показателей. Схема “звезда” состоит из нескольких реляционных таблиц с одной выделенной (фактографической) таблицей, в которой хранятся значения многомерных показателей. Остальные таблицы связаны с фактографической ограничением внешнего ключа и используются для хранения значений измерений.

Variable (переменная)
Обычно применяется как синоним меры (**measure**).
Обычно применяется как синоним гиперкуба.
