**Введение в MDX. Часть 1**

***Полина Трофимова, Алексей Шуленин***

[**Предисловие**](http://compress.ru/article.aspx?id=9512#01)

[**Область действия и основное назначение**](http://compress.ru/article.aspx?id=9512#02)

[**Основные понятия многомерной модели**](http://compress.ru/article.aspx?id=9512#03)

[**Пример многомерной модели (куб «Учет рейсов»)**](http://compress.ru/article.aspx?id=9512#04)

[**Кубы и ячейки**](http://compress.ru/article.aspx?id=9512#05)

[**Базовые термины языка MDX**](http://compress.ru/article.aspx?id=9512#06)

[**Пример запроса**](http://compress.ru/article.aspx?id=9512#07)

[**Описание осей результирующего набора**](http://compress.ru/article.aspx?id=9512#08)

[**Функции и выражения языка MDX**](http://compress.ru/article.aspx?id=9512#09)

[**Навигация по дереву измерений**](http://compress.ru/article.aspx?id=9512#10)

[**Выражения, не зависящие от уровня**](http://compress.ru/article.aspx?id=9512#11)

[**На том же уровне (CurrentMember; PrevMember)**](http://compress.ru/article.aspx?id=9512#12)

[**Уровень ниже, уровень выше**](http://compress.ru/article.aspx?id=9512#13)

[**Другие функции и выражения**](http://compress.ru/article.aspx?id=9512#14)

[**ORDER**](http://compress.ru/article.aspx?id=9512#15)

**[NonEmpty](http://compress.ru/article.aspx?id=9512" \l "16)**

[**Пример визуализации конструирования MDX-запроса в клиентских приложениях**](http://compress.ru/article.aspx?id=9512#17)

**Предисловие**

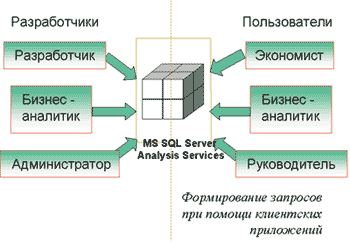
MDX (MultiDimensional eXpressions — язык запросов к многомерным данным) был впервые введен в рамках спецификации OLE DB for OLAP для работы с многомерными кубами. MDX — это расширение языка SQL (изначально ориентированного, как известно, на реляционную модель данных), предназначенное для манипуляции многомерным представлением информации, наиболее удобным для задач анализа. Действительно, их синтаксис во многом схож, и для тех читателей, кому довелось работать с SQL, освоение MDX не составит особого труда. Будучи открытым стандартом, MDX является основным инструментом программирования для Microsoft SQL Server 2000 Analysis Services. Учитывая актуальность аналитических приложений в современном бизнесе, следует отметить, что знание MDX позволяет значительно упростить их разработку на всех этапах производственного цикла начиная с постановки задачи, так как этот язык специально создавался для указанной предметной области и оперирует привычными для аналитика категориями.

Эта статья носит вводный характер. Не подменяя собой руководство по языку MDX, она знакомит читателя с описанием базовых синтаксических конструкций. Во второй части статьи мы продолжим систематизировать подход к решению практических задач программирования бизнес-логики, с которыми авторы столкнулись в реальных проектах.

|  |  |
| --- | --- |
|  |  |

**Область действия и основное назначение**

Существуют два основных аспекта использования MDX ([рис. 1](http://compress.ru/article.aspx?id=9512&part=11ext1)).



Первым и наиболее распространенным является построение запросов для получения наборов данных, находящихся в многомерной структуре. При построении запроса используется метаинформация, описывающая данные в терминах предметной области.

Вторым аспектом MDX, зачастую недооцениваемым на практике, служит формирование бизнес-модели, на основе которой возможно проведение наиболее полного и емкого анализа данных предприятия или организации. Для достижения этой цели необходимы не только базовые знания языка MDX, но и профессиональное владение предметной областью (будь то финансовый анализ банковской деятельности или управление авиапредприятием).

На графике ([рис. 2](http://compress.ru/article.aspx?id=9512&part=21ext1))



изображена зависимость возможностей разработчика MDX от изученной им функциональности языка. Следует отметить, что хотя владение базовыми конструкциями языка является необходимым этапом в освоении многомерной модели, основная сила MDX осознается при следующем шаге — овладении моделированием бизнес-логики. Именно здесь происходит качественный скачок в знаниях, осуществляется их систематизация и из набора разрозненных функций и выражений образуется стройная система организации бизнес-логики и формирования запросов. Дальнейшее совершенствование в основном касается оптимизации выполнения запросов. Это также является немаловажным аспектом при построении промышленных аналитических систем.

Благодаря грамотно построенной бизнес-логике значительно упрощается задача построения запросов. Цель разработчика MDX состоит в том, чтобы скрыть от пользователя громоздкую логику формирования расчетных величин — будь то обычный нарастающий итог или значение средневзвешенной продукции, предоставив ему возможность выбора этих величин как обычных мер и членов измерений.

Задачи языка можно определить следующим образом:

* MDX «понимает» многомерную модель устройства данных (куб, измерение, мера, ячейка);
* язык позволяет осуществлять навигацию по многомерному пространству и определенным над ним иерархиям;
* MDX нужен не только разработчикам и администраторам — он может быть полезен практически всем пользователям аналитических приложений.

|  |  |
| --- | --- |
|  |  |

**Основные понятия многомерной модели**

Поскольку перевод некоторых терминов, используемых в технологиях OLAP, на русский язык еще не утвердился окончательно, мы опишем необходимый набор ключевых слов, которые будут использованы в статье.

Хранилище данных, ХД (DataWarehouse, DW) — информация, получаемая из различных источников и накапливаемая в течение продолжительного промежутка времени и отражающая основные или наиболее значимые направления деятельности предприятия, а также средства хранения этой информации и обеспечения доступа к ней.

Сервер анализа (Analysis Server) — структура хранения многомерных данных и метаданных, в простейшем случае специальным образом организованная, содержащая описания создаваемых запросов и формул и другую аналитическую информацию плюс программные средства для поддержания ее целостности и выполнения запросов.

Куб (Cube) — часть данных сервера анализа, относящаяся к определенной предметной области, с которой осуществляется работа. Куб представлен набором мер и измерений.

Мера, или факт, или показатель (Measure) — параметры предметной области, служащие предметом анализа.

Измерение (Dimension) — понятия предметной области, на основании анализируются меры. Каждое измерение может иметь иерархическую структуру уровней. В частности, измерение Время может включать уровни Год, Квартал, Месяц, Число, а измерение География может включать уровни Весь мир, Страна, Город.

Член (Member) — единица описания на каждом уровне иерархии. Например, для измерения География. На уровне городов это — Москва, Санкт-Петербург, Киев, Россия, Украина и т.д.

Кортеж (Tuple)  — коллекция членов одного или нескольких разных измерений. Необходим для определения местоположения конкретной ячейки внутри многомерного куба или набора ячеек (Set), участвующих в запросе.

Множество (Set) — набор кортежей.

|  |  |
| --- | --- |
|  |  |

**Пример многомерной модели (куб «Учет рейсов»)**

В качестве примера многомерной модели данных рассмотрим куб «Учет рейсов», созданный на основе данных, которые поступят из транзакционных приложений, регистрирующих деятельность абстрактного аэропорта. Сами данные (архив многомерной базы «Авиаперевозки.CAB» и резервная копия транзакционной БД Flights.bak) находятся на прилагаемом к журналу компакт-диске. Если у вас установлен Microsoft SQL Server 2000, вы можете воспользоваться ими, чтобы практическая работа с примерами была более наглядной. Для работы с приведенными здесь примерами можно использовать модельное клиентское приложение MDX Sample Application в составе Analysis Services.

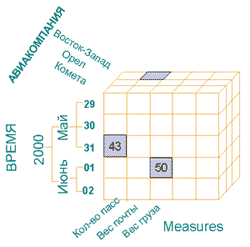
Итак, данные оперативных систем погружены в куб для решения задач анализа объемных (не финансовых) результатов работы аэропорта. В этом случае будут анализироваться пассажиро- и грузопотоки, поэтому в качестве мер (показателей) куба «Учет рейсов» будут фигурировать данные о количестве пассажиров и весе перевозимых авиалайнерами грузов. Теперь рассмотрим, в каких разрезах необходимо проводить анализ, то есть какие требуются измерения. Представим, что экономической службе аэропорта требуются сравнительные отчеты общих объемов ежемесячных и квартальных перевозок, а также возможность сопоставлять объемы перевозок различных авиакомпаний (для изменения условий контрактов с ними в области льгот и снижения тарифов оплаты по амортизационным взносам). Кроме того, для организации более логичного распределения грузоперевозок представляют интерес и географический анализ, и сравнительные данные различных типов самолетов. Таким образом, в куб включаются следующие измерения и меры:

* Измерения: Авиакомпания, Пункт назначения, Время, Тип самолета;
* Показатели: Колво пасс, Вес груза.

|  |  |
| --- | --- |
|  |  |

**Кубы и ячейки**

Для того чтобы продемонстрировать структуру многомерного куба и объяснить, каким образом организован доступ к ячейкам, обратимся к [рис. 3](http://compress.ru/article.aspx?id=9512&part=31ext1),



на котором изображен трехмерный куб, содержащий данные о количестве пассажиров, весе почты и груза, перевозимых через аэропорт под названием «Мягкой посадки», в разрезе времени отправки рейсов и принадлежности самолетов авиакомпаниям.

Модель куба представляет собой набор ячеек, каждая из которых содержит значение одного из показателей (мер) и привязана к одному из значений каждого измерения. Например, данные ячейки содержат следующую информацию:

**«43»: 31 мая а/компанией «Комета» было перевезено 43 пассажира.**

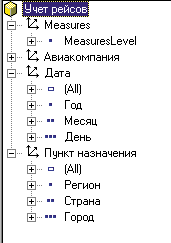
**«50»: 1 июня а/к «Орел» доставлено 50 тонн груза.**

|  |  |
| --- | --- |
|  |  |

**Базовые термины языка MDX**

Используя основные понятия и объекты многомерной модели, рассмотренные выше, обратимся к синтаксису языка.

Измерения и меры как частный случай измерений, например [Время], [Тип самолета], [Measures]. Analysis Services позволяет организовать несколько иерархий для одного измерения. Например, для различных отчетов могут потребоваться [Время].[недели от начала года] и [Время].[по месяцам]. В нашем примере каждое измерение имеет только одну иерархию. Остановимся более подробно на описании измерений в запросе ([рис. 4](http://compress.ru/article.aspx?id=9512&part=41ext1)).



Как правило, измерение содержит несколько уровней, по которым производится агрегация показателей.

Например, для измерения «Время» обычно имеются уровни: [All] (сумма всех загруженных в ХД дней), [Год] (сумма за год), [Месяц] (агрегация по конкретному дню), [День] (данные непосредственно за день).

Для аналитических систем регистрация временных величин ниже уровня [День] обычно не производится, так как системы поддержки и принятия решений основываются не на оперативном, а на временном, более глобальном, анализе. Измерение [Пункт назначения] разбито на уровни по географической принадлежности пунктов назначения рейсов: [Регион], [Страна], [Город].

Именование измерений производится следующим образом. Все заголовки, включающие специальные символы (пробелы, русские буквы и т.д.), заключаются в квадратные скобки. Для указания необходимого члена измерения можно прописать весь путь к нему, разделяя имена членов каждого уровня точками, например:

**31 мая 2000 г [Время].[Все время].[2000].[Май].[31].**

Членом измерения может быть значение на любом уровне измерения, в частности значение месяца, года или общее значение [Все время]. Необходимо отметить, что у каждого измерения существует член по умолчанию (default member), который используется в случае, если описание измерения в явном виде в запросе не участвует. Обычно в роли Default Member выступает единственный член специального уровня All, который добавляется автоматически при создании измерения и содержит совокупные результаты по всему данному измерению. Если этот уровень отсутствует (Advanced -> All Level = No), то в его роли выступает первый член следующего уровня. В Analysis Services (в отличие от OLAP Services версии 7.0) значением по умолчанию можно назначить любой член или MDX-выражение измерения.

Чтобы правильно сослаться на член измерения, необходимо описать полный путь к нему по иерархии измерения, начиная с самого верхнего уровня, например:

**[Пункт назначения].[Все пункты].[Россия].[РФ].[Москва]**

Однако если имя члена уникально в пределах измерения, то можно опустить наименования промежуточных уровней:

**[Пункт назначения].[Москва],**

а в случае уникальности среди измерений — и наименование самого измерения:

**[Москва],**

хотя мы бы не рекомендовали это делать.

Чтобы сослаться на ячейку «43» в примере трехмерного куба, необходимо описать следующий кортеж:

**([Авиакомпания].[Комета], [Время].[2000].[Май].[31], Measures.[Колво пасс]),**

Помните, что не вошедшие в кортеж измерения присутствуют в нем неявно за счет наличия членов по умолчанию. Таким образом, данный кортеж в действительности эквивалентен:

**([Авиакомпания].[Комета], [Время].[2000], [Пункт назначения].[Все пункты], [Тип самолета].[Все типы], [Measures].[Колво пасс])**

Порядок перечисления измерений и мер в кортеже не имеет существенного значения. Обратите внимание, что кортеж заключается в круглые скобки, в то время как множество — в фигурные:

**{([Время].[1997],[Measures].[Вес груза]), ([Время].[1998],[Measures].[Вес груза])}**

Множество можно не заключать в фигурные скобки, если оно является результатом функции, например

**[Пункт назначения].Members**

Набор всех членов в интервале задается при помощи двоеточия:

**{[Время].[1997]:[Время].[1999]}**

|  |  |
| --- | --- |
|  |  |

**Пример запроса**

Продолжим рассмотрение синтаксиса языка MDX, обратившись к простому запросу. Предположим, требуется получить сведения о ежемесячной перевозке пассажиров авиакомпанией «Орел» за январь и февраль 2000 года:

**SELECT**

**{[Авиакомпания].[Орел]} ON COLUMNS,**

**{[2000].[Январь],[2000].[Февраль]} ON ROWS**

**FROM [Учет рейсов]**

**WHERE**

**[Measures].[Колво пасс]**

Результатом выполнения этого запроса будет набор данных, представленный на [рис. 5](http://compress.ru/article.aspx?id=9512&part=51ext1).

http://compress.ru/archive/cp/2002/1/37/5.gif

В запросе указаны только измерения «Авиакомпания» и «Время». По измерениям «Тип самолета», «Пункт назначения» члены не были указаны, и было принято значение «По умолчанию» (уровень [All]). В этом случае по типам самолетов и городам назначения произведена агрегация значений. Запрос с учетом всех измерений имеет следующий вид:

**SELECT**

**{[Авиакомпания].[Орел]} ON COLUMNS,**

**{[Январь],[Февраль]} ON ROWS**

**FROM [Учет рейсов]**

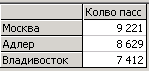
**WHERE**

**([Measures].[Колво пасс] ,**

**[Пункт назначения].[Все пункты],**

**[Тип самолета].[Все типы ])**

Характерным отличием языка MDX от SQL является то, что результатом запроса всегда являются значения одной или нескольких мер. Невозможно получить значения измерений, оторванные от мер (фактов). Например, в случае трех наиболее напряженных по пассажиропотоку маршрутов ([рис. 6](http://compress.ru/article.aspx?id=9512&part=61ext1)) запрос имеет следующий вид.



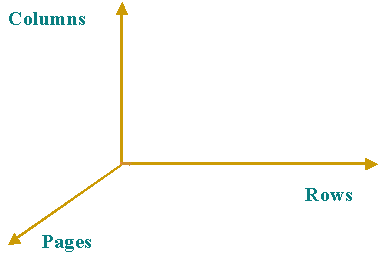
SELECT {[Measures].[Колво пасс]} on columns, TopCount(ORDER (FILTER([Пункт назначения].Members, [Пункт назначения].CurrentMember.Level.Name =”Город”), [Пункт назначения].CurrentMember.Name, BASC), 3, ([Measures].[Колво пасс])) ON rowS FROM [Учет рейсов]

В результирующем наборе помимо перечисленных городов отображается и столбец мер.

|  |  |
| --- | --- |
|  |  |

**Описание осей результирующего набора**

Оси, по которым мы располагаем результаты MDX-запроса, не обязательно должны совпадать с измерениями куба. Мы можем расположить несколько измерений по одной оси, получив тем самым их декартово произведение. Например, для каждой авиакомпании будут перечислены все типы самолетов, вошедших в результаты запроса. Следовательно, нам необходимо в тексте запроса осуществить привязку множеств к конкретным осям отчета. Несмотря на то что Analysis Services позволяет возвращать истинно многомерные наборы данных, для простоты визуализации требуются, как правило, лишь плоские представления. Обычно используются лишь оси столбцов и строк ([рис. 7](http://compress.ru/article.aspx?id=9512&part=71ext1)):



**select … on columns,**

**… on rows**

**from cube**

**where …**

В общем случае результирующий набор может быть многомерным (добавляются оси страниц, разделов, секций).

Каждой оси присваивается номер:

**x-axis=0; y-axis=1; z-axis=2…**

Для первых пяти осей используются именования и не допускается пропуск осей в запросе:

**Columns 0**

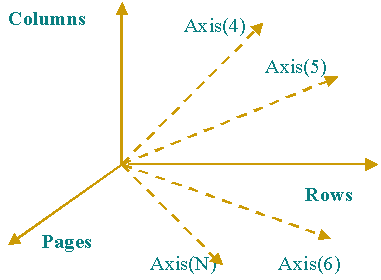
**Rows 1**

**Pages 2**

**Sections 3**

**Chapters 4**

Общий вид запроса и его графическое представление, иллюстрирующее размещение данных на нескольких осях, представлены на [рис. 8](http://compress.ru/article.aspx?id=9512&part=81ext1):



**select axis [,axis]**

**from cube**

**where slicer [,slicer]**

Нельзя располагать одно и то же измерение по разным осям отчета, поскольку такая операция лишена смысла.

|  |  |
| --- | --- |
|  |  |

**Функции и выражения языка MDX**

Язык MDX включает в себя значительное количество функций и выражений, позволяющих обрабатывать многомерные данные в соответствии с бизнес-логикой приложения, например для получения отчетов. Некоторые из них носят избыточный характер, то есть могут быть выражены через другие функции. Это сделано для упрощения конечных формул, поскольку облегчает составление запроса к кубу и делает текст запроса лучше читаемым.

|  |  |
| --- | --- |
|  |  |

***Навигация по дереву измерений***

Набор функций, позволяющих ссылаться на одни члены измерения относительно других, называют функциями навигации. В MDX навигация осуществляется по измерениям, а точкой отсчета является член измерения, обрабатываемый в данный момент Analysis Services (выражение CurrentMember), либо указанный в выражении член.

|  |  |
| --- | --- |
|  |  |

***Выражения, не зависящие от уровня***

Существуют два выражения, не зависящие от уровня точки отсчета.

Если требуется запросить все члены измерения независимо от уровня, на котором они расположены, используется функция .Members.

Например, требуется получить полную информацию о пассажиропотоке в разрезе географии пунктов следования пассажиров и авиакомпаний-перевозчиков за 2000 год. В этом случае можно сформировать следующий запрос:

**SELECT**

**{[Пункт назначения].Members} ON ROWS,**

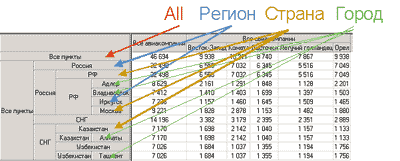
**{[Авиакомпания].Members} ON COLUMNS**

**FROM [Учет рейсов]**

**WHERE ([Время].[Все Время].[2000],**

**Measures.[Колво пасс])**

На [рис. 9](http://compress.ru/article.aspx?id=9512&part=91ext1) приведен результат выполнения запроса. Выражение .Members позволило получить все значения измерения [Пункт назначения].



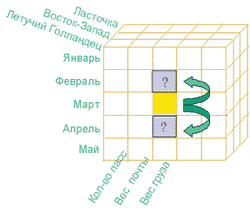
DefaultMember — выражение, позволяющее работать со значением измерения по умолчанию.

|  |  |
| --- | --- |
|  |  |

***На том же уровне (CurrentMember; PrevMember)***

Для того чтобы перемещаться в пределах уровня, на котором находится точка отсчета, используются следующие функции. Ссылку на соседние члены измерения без указания имени члена обеспечивают выражения .PrevMember и .NextMember.

Допустим, требуется создать отчет, отображающий сведения на интересующий пользователя месяц в сравнении с предыдущим. На одну из осей отчета помещаем измерение «Время», месяц «март» ([рис. 10](http://compress.ru/article.aspx?id=9512&part=101ext1)):



Чтобы сослаться в запросе на апрель, применим выражение:

**[Время].[2000].[Март].NextMember**

Чтобы сослаться на февраль:

**[Время].[2000].[Март].PrevMember**

Обратиться к члену январь можно, используя выражение:

**[Время].[2000].[Март].PrevMember.PrevMember,**

но удобнее применить более общую функцию:

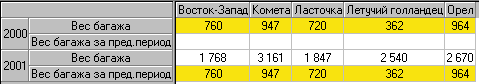
**[Время].[2000].[Март].Lag(2)**

или:

**[Время].[2000].[Март].Lead(-2)**

Отрицательное число в скобках меняет направление отсчета членов измерения.

А теперь вернемся к нашему примеру. Представим, что главному экономисту аэропорта требуется отчет об объеме грузоперевозок за текущий год по сравнению с предыдущим ([рис. 11](http://compress.ru/article.aspx?id=9512&part=111ext1)):



**WITH MEMBER [Measures].[Вес груза за пред период]**

**AS ‘([Measures].[Вес груза],[Время].PREVMEMBER)’**

**SELECT CROSSJOIN({[Время].[2000], [Время]. [2001]},**

**{[Measures].[Вес груза], [Measures].[Вес груза за пред период]}) ON ROWS, {[Авиакомпания].[Комета],**

**[Авиакомпания].[Ласточка]} ON COLUMNS**

**FROM [Учет рейсов]**

Существует еще несколько функций для обращения к членам измерения на том же уровне:

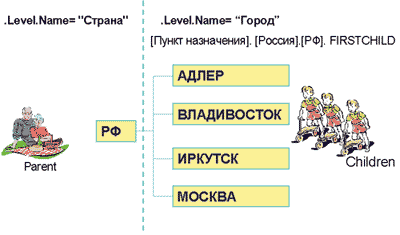
* определение первого и последнего члена в ряду FirstSibling и LastSibling;
* определение «кузенов», принадлежащих другому родителю — Cousin().

|  |  |
| --- | --- |
|  |  |

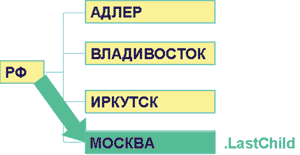
***Уровень ниже, уровень выше***

Предполагается, что в момент выполнения запроса известны имена членов родителей или потомков интересующего члена. В примере отображены 2-го уровня измерения [Пункт назначения] — Страна и Город. Функциональность выражений, используемых в этой группе, очевидна: .Children, .FirstChild, .LastChild — для известного родителя и .Parent — для известного потомка.

Допустим, требуется регулярно предоставлять отчет о перевозках пассажиров и груза в пределах Российской Федерации ([рис. 12](http://compress.ru/article.aspx?id=9512&part=121ext1)).

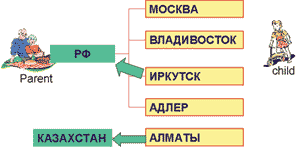


Для этого на ось строк поместим измерение [Пункт назначения]. Перечисление одного за другим всех городов России, имеющихся в измерении, могло бы решить поставленную задачу, но это не только приведет к удлинению запроса, но и заложит мину замедленного действия: спустя некоторое время может открыться рейс в город, в который ранее полеты не совершались, а потому его название не было включено в измерение. В результате подготовленный отчет станет неточен. Чтобы гарантировать пользователю актуальность созданных запросов, следует на оси строк расположить выражение ([рис. 13](http://compress.ru/article.aspx?id=9512&part=131ext1)):



**[Пункт назначения].[РФ].Children**

Выражения .FirstChild и .LastChild определяют первого и последнего потомка родителя ([рис. 14](http://compress.ru/article.aspx?id=9512&part=141ext1)).



Чтобы определить родителя известного члена измерения, используется функция .Parent ([рис. 15](http://compress.ru/article.aspx?id=9512&part=151ext1)).



Например, требуется узнать долю пассажиропотока интересующего пользователя города от общего количества перевезенных по стране пассажиров. Для этого применяем формулу:

**[Measures].[Колвопасс].[Пункт назначения]. CurrentMember/**

**[Measures].[Колво пасс].[Пункт назначения]. CurrentMember.Parent\*100**

Для того чтобы определить «дедушку» (родителя родителя), например регион для города Алма-Ата, используем выражение:

**[Алма-Ата].Parent.Parent – «СНГ».**

В общем случае используется функция Descendants(), позволяющая ссылаться на члены измерений различных уровней.

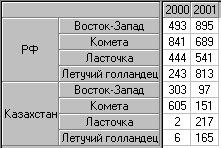
|  |  |
| --- | --- |
|  |  |

**Другие функции и выражения**

Как уже отмечалось ранее, на любой оси запроса возможно наличие сразу нескольких измерений. Сочетание членов одного измерения с другим дают пользователю возможность совмещать отчеты, ранее представленные ему раздельно, поскольку именно многомерная модель данных ориентирована на формирование подобных запросов.

Для получения декартова произведения членов измерений существует функция CrossJoin.

Допустим, экономической службе авиапредприятия требуется провести анализ перевозок авиакомпаний в разрезе регионов полетов за текущий и предыдущий годы. Чтобы построить отчет, разместим на оси строк [Пункт назначения].[Все пункты].Children и [Авиакомпания].[Все авиакомпании].Children, а на оси столбцов разместим измерение [Время] ([рис. 16](http://compress.ru/article.aspx?id=9512&part=161ext1)):



**SELECT CROSSJOIN({[Пункт назначения].[Все пункты].Children},**

**{[Авиакомпания].[Все авиакомпании].Children})**

**ON ROWS,**

**{[Время].[Все время].[2000], [Время].[Все время].[2001]} ON COLUMNS**

**FROM [Учет рейсов]**

**Filter, TopCount(), TopSum()**

Функция CrossJoin() позволяет увеличить результирующий набор строк за счет пересечения измерений, но нередко возникает необходимость уменьшить поток выходных данных, предъявляя определенные условия к мерам или измерениям. Задачи фильтрации данных занимают в многомерном анализе одно из первых мест по количеству запросов, которые используют функцию Filter и некоторые другие функции, служащие для отсечения ненужной информации.

В общем случае построения запроса недостаточно указать, какие измерения будут находиться на осях, а по каким будет произведен срез. Аналитик нуждается в деталях, уточнениях, конкретных данных. Эти возможности предоставляются функцией Filter. Ее синтаксис очень прост: необходимо указать набор данных и условное выражение, а возвращены будут данные исходного набора, для которых выполняется указанное в Filter() условие. Кроме основной функции фильтрации имеется набор вспомогательных функций: TopCount(),TopSum(), BottomCount(), BottomSum().

Сочетание этих функций дает аналитику возможность формировать такие типовые отчеты, как:

* Фильтрация данных по величине показателя

(Авиакомпании-перевозчики с объемом перевозок не менее…):

**SELECT FILTER({[Авиакомпания].[Все авиакомпа-нии].Children}, ([Measures].[Колво**

**пасс])>50) ON ROWS, {[Measures].[Колво пасс]} ON COLUMNS FROM [Учет рейсов]**

* Фильтрация данных по условию на измерение

(Города, содержащие в названиях подстроку):

**SELECT FILTER({[Пункт назначения].members}, InStr([Пункт назначения].CurrentMember.Name,**

**“ск”)<>0) ON ROWS, {[Measures].[Вес груза], [Measures].[Колво пасс]}**

**ON COLUMNS FROM [Учет рейсов]**

* Ведущие и последние в отсортированном списке

(Три авиакомпании-перевозчика с наибольшим пассажирооборотом):

**SELECT TopCount({[Авиакомпания].[Все авиакомпа-нии].Children}, 3, ([Measures].[Колво**

**пасс])) ON ROWS, {[Measures].[Колво пасс]} ON COLUMNS FROM [Учет рейсов]**

|  |  |
| --- | --- |
|  |  |

***ORDER***

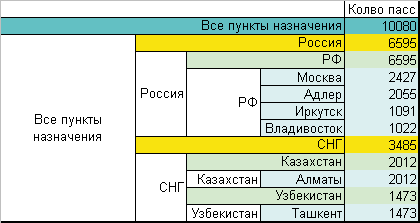
Итак, все необходимые для анализа данные получены, осталось лишь представить информацию в удобном для пользователя виде. Упорядочивание членов измерений на осях производится при помощи функции ORDER(). В обычной ситуации, когда нужно отсортировать члены одного уровня измерения на оси, ее действие аналогично предикату ORDER языка SQL. Если в запросе используются члены нескольких уровней измерения, упорядочивание членов можно производить как с сохранением иерархии уровней, так без нее (так называемый Break Hierarchy):

**ASC | DESC | BASC | BDESC,**

**BASC – B (Break Hierarchy) ASC**

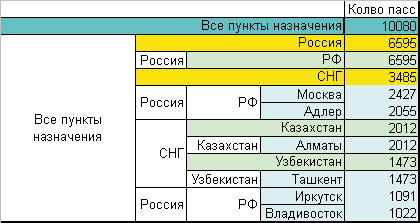
В ряде случаев необходимо анализировать данные без учета иерархии. Рассмотрим это на примерах.

Задача № 1: отсортировать данные о количестве отправленных пассажиров по направлениям полетов по убыванию с сохранением иерархии измерения «Пункт назначения» ([рис. 17](http://compress.ru/article.aspx?id=9512&part=171ext1)):



SELECT {[Measures].[Колво пасс]} ON COLUMNS, ORDER({[Пункт назначения].members}, ([Measures]. [Колво пасс]), DESC) ON ROWS FROM [Учет рейсов]

Задача № 2: предоставить сведения о количестве пассажиров по направлениям полетов, упорядоченных по убыванию, включая сведения по регионам, странам и городам, независимо от уровня — для выявления нерентабельных направлений ([рис. 18](http://compress.ru/article.aspx?id=9512&part=181ext1)):



Результаты анализа показывают, что полеты в Алма-Ату приносят большую прибыль, чем полеты во Владивосток, хотя полеты в регионе Россия значительно выгоднее полетов в Казахстан.

|  |  |
| --- | --- |
|  |  |

***NonEmpty***

Регулярные загрузки хранилища данных не предполагают, что при каждой загрузке будут предоставляться данные по всем измерениям и показателям. Например, в бархатный сезон рейсы на Адлер осуществляются всеми авиакомпаниями ежедневно, а в зимнее время — лишь два раза в неделю, то есть для измерения «Время» не каждый день января будет заполнен сведениями о полетах на Адлер. Это значит, что появятся «пустые» ячейки, которые являются неинформативными и вряд ли будут интересовать аналитика при работе с отчетом. Необходимо учитывать, что хотя пустоты физически в кубе не хранятся, многомерная модель требует их отображения. Можно исключить пустые строки и столбцы из отчета, применив функцию NonEmpty. Допустим, вас интересует напряженность полетов в конкретные дни января по направлениям полетов. Задействовав функцию NonEmpty, вы исключаете направления полетов, по которым не осуществлялись рейсы за этот период. Отчет сокращается, а результаты не искажаются.

|  |  |
| --- | --- |
|  |  |

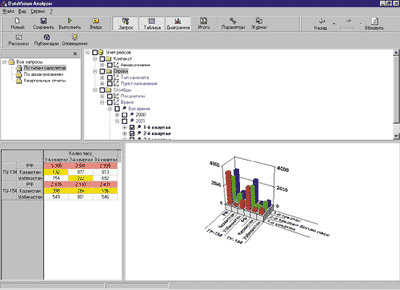
**Пример визуализации конструирования MDX-запроса в клиентских приложениях**

Можно упростить освоение MDX, воспользовавшись удобным и интуитивно понятным интерфейсом конструирования запросов. Аналитик строит отчеты, размышляя о проблеме лишь в предметных терминах, а все специальные функции и выражения скрыты от него за мастерами построения тех ли иных условий. Пользователь переносит мышью необходимые изменения на строки и столбцы формируемого отчета, выделяет нужные для анализа члены измерений с помощью мастеров, накладывает фильтрацию и устанавливает порядок сортировки.

Приложения, содержащие интерфейс для визуального конструирования запросов, доступны аналитику с первого же дня работы с многомерными данными, поскольку они легки в освоении, дают представление о многомерной модели, а также позволяют оценить возможности языка MDX, не изучая подробно весь его синтаксис.

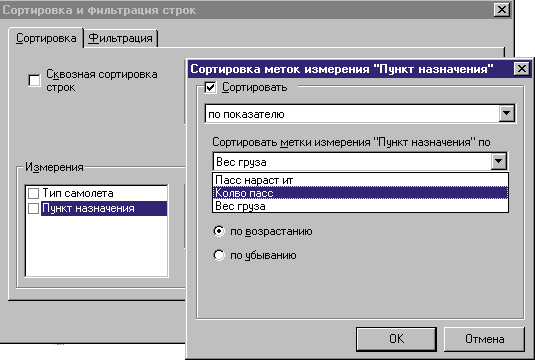
В качестве примера рассмотрим клиентское приложение DataVision Analyzer (разработка Digital Design: www.digdes.com), позволяющее строить отчеты на основе многомерного хранилища данных.

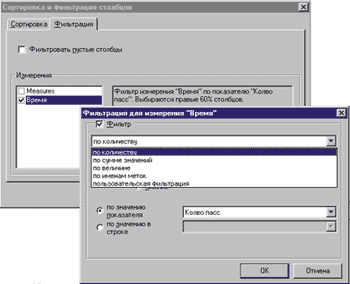
При работе DataVision Analyzer использует метаданные, содержащиеся в его репозитарии. DataVision Analyzer предоставляет пользователю возможность построить запрос к OLAP-серверу в терминах конкретной предметной области, указав не только интересующие его данные, но и их размещение по столбцам и строкам результирующей таблицы, а затем просмотреть полученный результат выполнения запроса в табличном и графическом виде ([рис. 19](http://compress.ru/article.aspx?id=9512&part=191ext1)).



Ранее построенные запросы доступны всем пользователям DV Analyzer и могут выполняться неограниченное количество раз. Одной из основных и самых удобных возможностей DV Analyzer является экспорт получаемой из OLAP-сервера информации (в виде таблиц или диаграмм) в Microsoft Excel, что позволяет использовать предоставляемые Excel инструменты оформления и обработки данных.

С точки зрения построения MDX-запросов DV Analyzer является прекрасным инструментом для начала работы с многомерными данными, помогающим пользователю не только разложить измерения куба на строки и столбцы запроса, но настроить различные сортировки, фильтрации и другие условия ограничения вывода данных в отчет, отображая все условия в конструкторе запросов для последующих изменений. На [рис. 20](http://compress.ru/article.aspx?id=9512&part=201ext1) и [21](http://compress.ru/article.aspx?id=9512&part=211ext1) представлены примеры настройки сортировки и фильтрации при помощи соответствующих мастеров.





Результатом работы с конструктором запросов и выполнения настроек сортировки и фильтрации при помощи мастеров является запрос, который выглядит следующим образом:

**SELECT CROSSJOIN({[Тип самолета].[Все типы].[ТУ-134], [Тип самолета].[Все типы].[ТУ-154]},**

**ORDER({[Пункт назначения].[Все пункты].[Россия].[РФ], [Пункт назначения].[Все пункты].[СНГ].[Казахстан],**

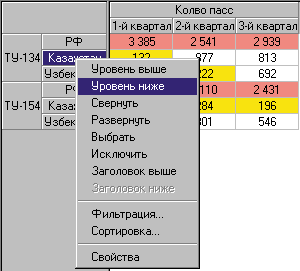
**[Пункт назначения].[Все пункты].[СНГ].[Узбекистан]},**

**([Measures].[Вес груза]), ASC)) ON ROWS, CROSSJOIN({[Measures].[Колво пасс]},**

**TopPer-cent({[Время].[Все время].[2001].[1-й квартал], [Время].[Все время].[2001].[2-й квартал], [Время].[Все время].[2001].[3-й квартал]}, 60,**

**([Measures].[Колво пасс]))) ON COLUMNS FROM [Учет рейсов]**

Построенные запросы доступны всем пользователям, обладающим соответствующими правами. После выполнения запроса пользователь имеет возможность более подробно проанализировать полученные данные при помощи стандартных OLAP-операций: «Уровень выше/Уровень ниже», «Свернуть/Развернуть», «Выбрать/Исключить», «Заголовок выше/Заголовок ниже» ([рис. 22](http://compress.ru/article.aspx?id=9512&part=221ext1)).



Кроме того, DV Analyzer позволяет пользователям cоздавать списки электронных рассылок, а также публиковать на Web-сервере выбранные отчеты по расписанию. Все это дает доступ к требуемой информации без необходимости запуска дополнительных приложений и делает данные доступными для удаленных пользователей.