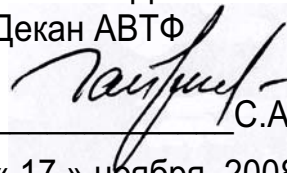


УТВЕРЖДАЮ

Декан АВТФ



С.А. Гайворонский

« 17 » ноября 2008 г.

А.В. Кудинов

ХРАНИЛИЩА ДАННЫХ
ЦИКЛ ЛАБОРАТОРНЫХ РАБОТ
Часть 1

Методические указания к циклу лабораторных работ по дисциплине
«Хранилища данных» для магистрантов, обучающихся
по магистерской программе «Компьютерный анализ
и интерпретация данных» направления 230100
«Информатика и вычислительная техника»

Издательство
Томского политехнического университета
2008

УДК 681.3.016(076.5)
ББК 32.973-018.2я73
К88

Кудинов А.В.

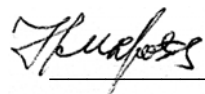
К88 Хранилища данных. Цикл лабораторных работ. Часть 1: методические указания к циклу лабораторных работ по дисциплине «Хранилища данных» для магистрантов, обучающихся по магистерской программе «Компьютерный анализ и интерпретация данных» направления 230100 «Информатика и вычислительная техника» / А.В. Кудинов. – Томск: Изд-во Томского политехнического университета, 2008. – 37 с.

ISBN 5-98298-306-3

УДК 681.3.016(076.5)
ББК 32.973-018.2я73

Методические указания рассмотрены и рекомендованы
к изданию методическим семинаром кафедры
вычислительной техники АВТФ
20 декабря 2007 г.

Зав. кафедрой ВТ
доктор технических наук

 Н.Г. Марков

Председатель учебно-методической
комиссии

 В.И. Рейзлин

Рецензент

Кандидат технических наук,
заместитель начальника управления ОАО «Востокгазпром»
А.В. Сарайкин

ISBN 5-98298-306-3

© Кудинов А.В., 2008
© Томский политехнический университет, 2008
© Оформление. Издательство Томского
политехнического университета, 2008

ВВЕДЕНИЕ

Предлагаемые методические указания предназначены для магистрантов, обучающихся по магистерской программе «Компьютерный анализ и интерпретация данных» направления «Информатика и вычислительная техника» факультета автоматизации и вычислительной техники. Используются при выполнении лабораторных работ и индивидуальных заданий по дисциплине «Хранилища данных».

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

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

Основные требования к хранилищам данных:

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

Типичное хранилище данных отличается от обычной реляционной базы данных.

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

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

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

В отличие от так называемых оперативных баз данных, с которыми работают приложения, модифицирующие данные, хранилища данных предназначены исключительно для обработки и анализа информации, поэтому проектируются они таким образом, чтобы время выполнения запросов к ним было минимальным. Данные копируются в хранилище из оперативных баз данных.

Данные методические указания к лабораторным работам охватывают первую часть цикла лабораторных работ. Курс ориентирован на использование платформы разработки/управления хранилищ данных на основе *Microsoft SQL Server 2000/2005 Analyzes Services* и предполагает последовательную разработку студентом хранилища данных согласно индивидуальному заданию: от проектирования, реализации и наполнения базы данных, являющейся источником данных для хранилища до реализации хранилища и проведения анализа его данных посредством использования *Microsoft Office Excel* или создания специализированного приложения. Методические указания к последним двум темам и список индивидуальных заданий планируется издать отдельно.

Методические указания к лабораторным работам отражают практические вопросы, предусмотренные образовательной программой магистерской подготовки, и соответствуют рабочей программе дисциплины.

1. ПРОЕКТИРОВАНИЕ, РЕАЛИЗАЦИЯ И НАПОЛНЕНИЕ БАЗЫ ДАННЫХ, ЯВЛЯЮЩЕЙСЯ ИСТОЧНИКОМ ДАННЫХ ДЛЯ ХРАНИЛИЩА

1.1. Основные понятия

Сущность – это некоторая абстракция реально существующего объекта (класса объектов), процесса или явления, о котором необходимо хранить информацию в БД.

Атрибут – это поименованная характеристика сущности, которая принимает значения из некоторого множества значений. В модели атрибут выступает в качестве средства, с помощью которого моделируются свойства сущностей.

Связи выступают в модели в качестве средства, с помощью которого представляются отношения между сущностями, имеющими место в предметной области. При анализе связей между сущностями могут встречаться бинарные (между двумя сущностями), тернарные (между тремя сущностями) и в общем случае – n-арные связи.

Потенциальный ключ – это подмножество атрибутов, обладающее свойствами уникальности и минимальности. В одном отношении может существовать несколько потенциальных ключей.

Первичный ключ – это любой из потенциальных ключей, выбираемый для использования в связях между отношениями. Если выбран первичный ключ, то все остальные потенциальные ключи называют альтернативными ключами.

Внешний ключ – это столбец или комбинация столбцов, значения которых необходимы для сопоставления с первичным ключом в другой таблице. Используется для объединения родственных таблиц.

1.2. Анализ предметной области

Рассмотрим пример. Необходимо спроектировать базу данных супермаркета. В этой базе данных должна храниться информация о товарах, поставщиках, заказах, доставках, клиентах, сотрудниках супермаркета. В ходе анализа предметной области необходимо учесть её специфику. В частности, будущая база данных должна учитывать следующие факты:

- в одном заказе может быть несколько товаров;
- один товар может фигурировать в нескольких заказах;
- отношения между сотрудниками супермаркета могут быть представлены в виде иерархии, то есть один сотрудник может быть и начальником и подчиненным одновременно.

С учетом изложенных фактов можно приступить к созданию концептуальной модели.

1.3. Концептуальное моделирование

Модель базы данных, отражающая взгляд пользователей на предметную область с точки зрения решаемых задач, называется *концептуальной моделью данных*. Она складывается под воздействием ряда концептуальных требований, определяющихся множеством задач, для решения которых проектируется конкретная БД. Концептуальные требования определяют набор объектов, описываемых в БД, набор их атрибутов, хранящихся в БД, а также связи между объектами.

При создании концептуальной модели сначала определяются сущности и их атрибуты, назначаются первичные ключи. После создания сущностей определяются связи между ними.

Для создания моделей баз данных используются специальные программные продукты, названные CASE-средствами. Одним из таких продуктов является программа *Power Designer*. Она разработана фирмой *Sybase* и предназначена для создания различных моделей данных, отчетов и репозиториев.

Для построения концептуальной модели, после запуска программы *Power Designer*, нужно в меню *File* выбрать пункт *New*. В появившемся окне в поле *Model type* нужно выбрать *Conceptual model* и нажать кнопку *OK*.

Чтобы создать сущности, нужно на панели *Palette* выбрать значок сущности, а затем щелкнуть мышкой на рабочем поле. Чтобы задать имя сущности и ее атрибуту, нужно дважды щелкнуть по ней мышкой. В появившемся окне на закладке *General* в соответствующем поле вводится имя сущности. Атрибуты, их типы, обязательность их заполнения в БД и наличие ключа задаются на закладке *Attributes* (в столбцах *Name*, *Data type*, *Mandatory* и *Primary Key*, соответственно).

Чтобы создать связи между сущностями, нужно на панели *Palette* выбрать значок связи и протянуть связь от одной сущности к другой. Чтобы изменить сущность, нужно дважды по ней щелкнуть. В появившемся окне на закладке *General* определяются общие свойства. Закладка *Details* позволяет определить направленность связи (*One – One*, *One – Many* и т. д.), мощность связи – (0, 1), (1, n) и т. д., зависимость одной сущности от другой.

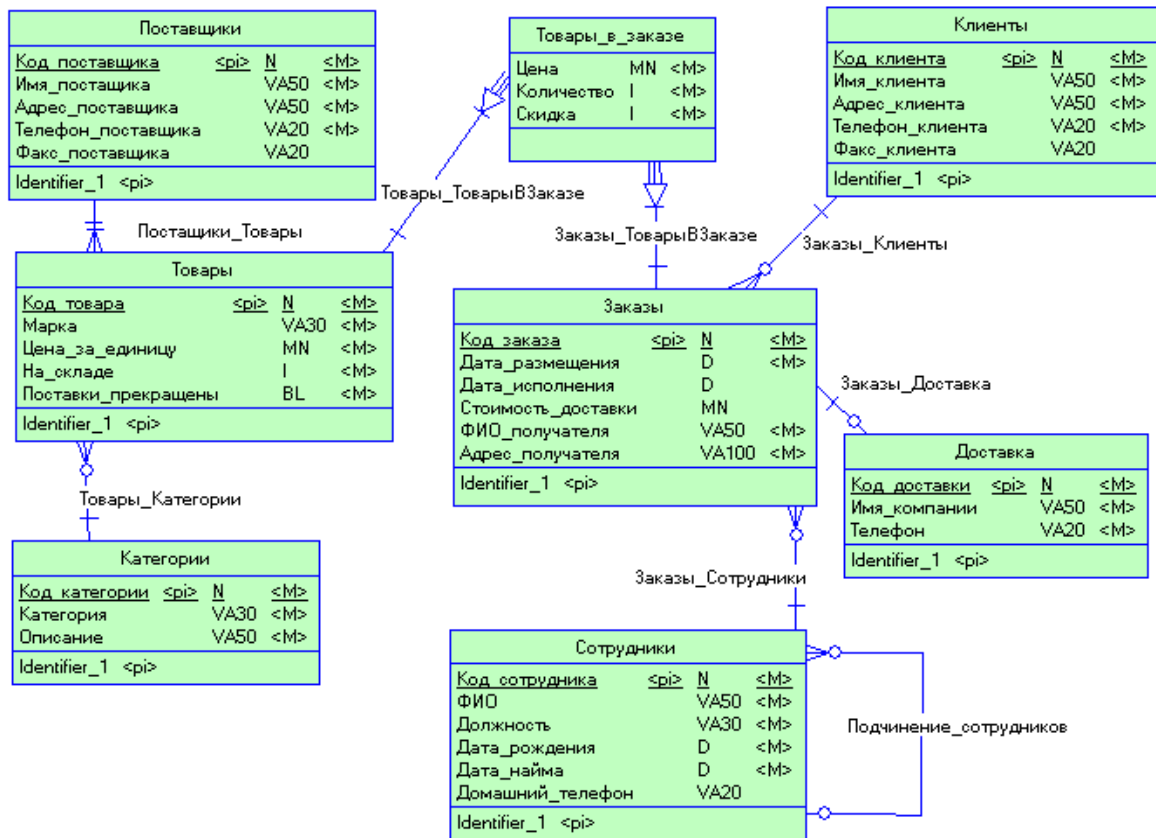


Рис. 1.1. Концептуальная модель

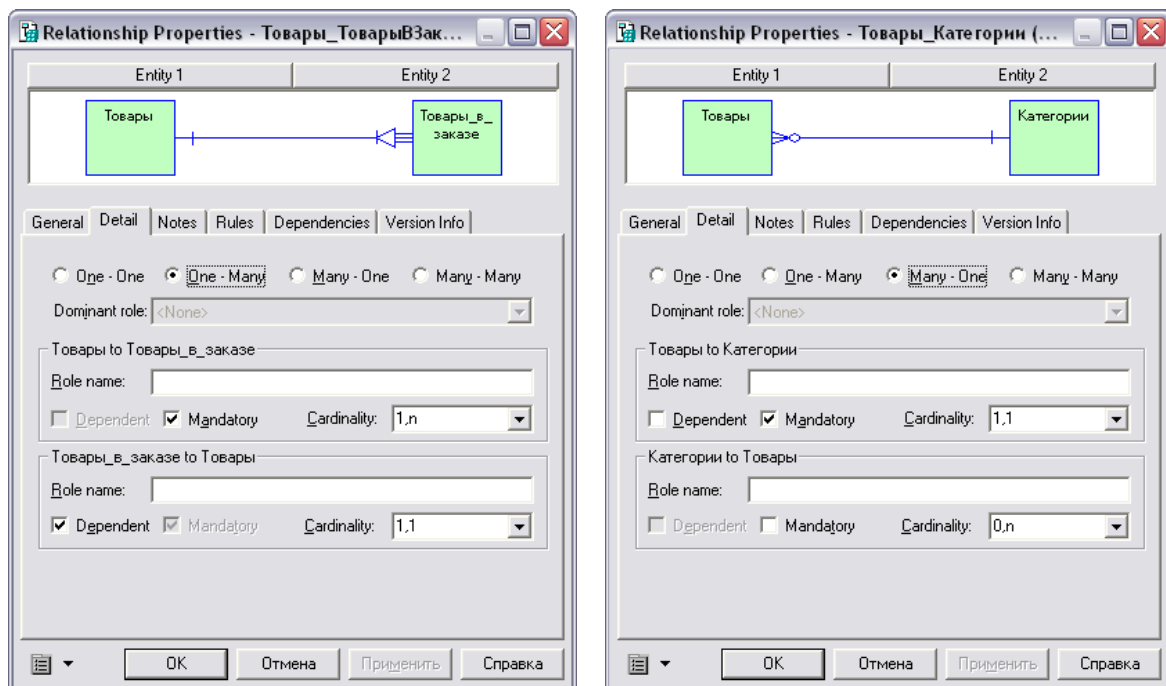


Рис. 1.2. Свойства связей

Свойства связей *Товары_Товары в заказе* и *Товары_Категории* представлены на рис. 1.2.

Если одна сущность зависима от другой, то при формировании физической модели в состав зависимой сущности в качестве ключевых атрибутов войдут ключи из связанной сущности.

Пример концептуальной модели, построенной в программном продукте *Power Designer*, представлен на рис. 1.1.

1.4. Физическое моделирование

Отображение внутренней модели данных на внешний носитель с учетом особенностей файловой системы и системы ввода-вывода ЭВМ называется физической БД или физической моделью данных.

При создании физической модели предоставляется возможность настройки модели: выбор сервера, имя сервера записываются ли изменения в физической модели поверх концептуальной и пр.

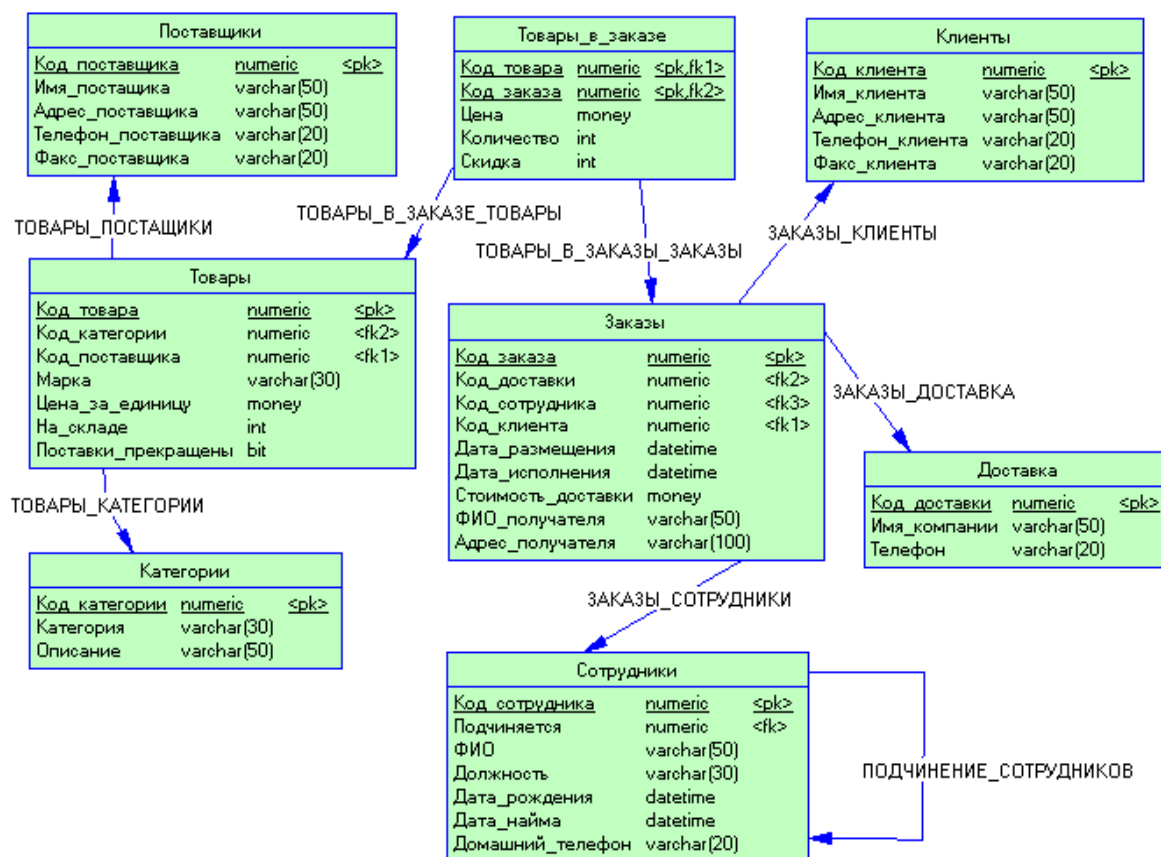


Рис. 1.3. Физическая модель

Отношения *многие ко многим* реализуются через создание специальных сущностей.

Чтобы создать физическую модель, нужно, прежде всего, открыть концептуальную модель. Затем в меню *Tools* нужно выбрать пункт *Generate Physical Data Model*. После этого физическая модель будет сгенерирована программой автоматически.

Пример физической модели приведен на рис. 1.3.

1.5. Создание базы данных

В соответствии с созданной физической моделью создается база данных. В программном продукте *Power Designer* предусмотрено средство генерации базы данных по существующей физической модели данных.

Чтобы сгенерировать базу данных, нужно в меню *Database* выбрать пункт *Generate Database*. В открывшемся диалоговом окне можно настроить параметры генерации базы данных. После этого нужно запустить программу *Microsoft SQLServer Enterprise Manager* и создать новую базу данных. Для этого нужно в дереве консоли открыть папку *Databases*, щелкнуть в любом месте окна правой кнопкой мыши и выбрать пункт контекстного меню *New Database*. После этого нужно запустить программу *Microsoft SQLServer Query Analyzer*, выбрать в меню *File* пункт *Open* и в появившемся окне выбрать файл, созданный программой *Power Designer* в процессе генерации базы данных. Затем нужно на панели управления выбрать базу данных, созданную в программе *Enterprise Manager* и нажать кнопку *Execute Query*, расположенную на панели управления. Программа *Query Analyzer* создаст в базе данных таблицы, прописанные в файле, сгенерированном программой *Power Designer*.

1.6. Заполнение базы данных

На этом этапе каждая из созданных таблиц заполняется информацией. Работа по заполнению базы данных информацией проводится в программе *Microsoft SQLServer Enterprise Manager*. Чтобы открыть базу данных, нужно выбрать ее в дереве консоли, расположенном в левой части рабочей области. Чтобы открыть таблицу в режиме заполнения, необходимо нажать правой кнопкой мыши по ее названию и в появившемся контекстном меню выбрать *Open Table → Return all rows*. В открывшемся окне вводятся значения атрибутов для каждой строки.

1.7. Контрольные вопросы

1. Что такое сущность, атрибут, связи?
2. Какие типы ключей Вы знаете?
3. Перечислите основные этапы проектирования реляционных баз данных.

2. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ ХРАНИЛИЩА ДАННЫХ

2.1. Определение и типовые архитектуры хранилищ данных

Определение понятия «хранилище данных» первым дал Уильям Инмон в своей монографии – это «предметно-ориентированная, интегрированная, содержащая исторические данные, неразрушаемая совокупность данных, предназначенная для поддержки принятия управленческих решений».

Данные из различных источников помещаются в хранилище, а их описания – в репозиторий метаданных. Конечный пользователь, используя различные инструменты (средства визуализации, построения отчетов, статистической обработки и т. д.) и содержимое репозитория, анализирует данные в хранилище. Результатом является информация в виде готовых отчетов, найденных скрытых закономерностей, каких-либо прогнозов. Так как средства работы конечного пользователя с хранилищем данных могут быть самыми разнообразными, то теоретически их выбор не должен влиять на структуру хранилища и функции его поддержания в актуальном состоянии. Физическая реализация данной концептуальной схемы может быть самой разнообразной.

Двухуровневая архитектура хранилища данных подразумевает построение витрин данных (*data mart*) без создания центрального хранилища, при этом информация поступает из регистрирующих систем и ограничена конкретной предметной областью. При построении витрин используются основные принципы построения хранилищ данных, поэтому их можно считать хранилищами данных в миниатюре. Плюсы: простота и малая стоимость реализации; высокая производительность за счет физического разделения регистрирующих и аналитических систем, выделения загрузки и трансформации данных в отдельный процесс, оптимизированный под анализ структурой хранения данных; поддержка истории; возможность добавления метаданных.

Построение полноценного корпоративного хранилища данных обычно выполняется в трехуровневой архитектуре. На первом уровне расположены разнообразные источники данных – внутренние регистрирующие системы, справочные системы, внешние источники (данные информационных агентств, макроэкономические показатели). Второй уровень содержит центральное хранилище, куда стекается информация от всех источников с первого уровня, и, возможно, оперативный склад данных, который не содержит исторических данных и выполняет две основные функции. Во-первых, он является источником аналитической

информации для оперативного управления, и, во-вторых, здесь подготавливаются данные для последующей загрузки в центральное хранилище. Под подготовкой данных понимают их преобразование и проведение определенных проверок. Наличие оперативного склада данных просто необходимо при различном регламенте поступления информации из источников. Третий уровень представляет собой набор предметно-ориентированных витрин данных, источником информации для которых является центральное хранилище данных. Именно с витринами данных и работает большинство конечных пользователей.

2.2. Виды моделей хранилища данных

Хранилища строятся на основе многомерной модели данных, подразумевающей выделение отдельных измерений (время, география, клиент, счет) и фактов (объем продаж, доход, количество товара) с их анализом по выбранным измерениям. Многомерная модель данных физически может быть реализована как в многомерных, так и в реляционных СУБД. В последнем случае она выполняется по схеме «звезда» или «снежинка». Каждая таблица фактов содержит детальные данные и внешние ключи на таблицы измерений.

Теория построения многомерной модели данных и ее воплощение в реляционной структуре известна, однако информации по проблеме представления иерархий очень мало. В качестве примера измерения, широко применяющегося при анализе деятельности предприятия и имеющего иерархическую структуру, можно привести справочник статей затрат (рис. 2.1).



Рис. 2.1. Модель иерархического справочника

Классически проблема представления иерархий решается с помощью рекурсивной связи, что позволяет помещать в одной таблице дерево любой глубины и размерности. В нашем случае рассматриваемые данные представлены в виде табл. 2.1.

Таблица 2.1

Представление иерархий с помощью рекурсивной связи

ID	NAME	PARENT ID
1	Предприятие	
2	Управление	1
3	Инфраструктура	1
4	Производство	1
5	Энергия	3
6	Сервисные услуги	3
7	Месторождение А	4
8	Месторождение Б	4

Однако в простоте этого решения скрывается и основной его недостаток: стандарт SQL не поддерживает рекурсивные указатели, поэтому для представления деревьев в хранилище данных используют другие методы.

Метод, предложенный Джо Селко, основан на теории множеств – все узлы дерева проходятся в прямом порядке и для каждого узла записывается два значения (рис. 2.2).



Рис. 2.2. Нумерация левой и правой границы узлов дерева

Сначала заполняется левая граница и лишь затем правая – при движении обратно от потомков к родителям. При такой нумерации узел каждый родитель содержит потомков, левая и правая граница которых лежит в интервале между левой и правой границей родителя. Аналогично все родители потомка имеют левую границу, которая меньше левой границы потомка, и правую, большую правой границы потомка. Следовательно, сумму затрат для конкретного места возникновения затрат и всех его составляющих можно получить одним запросом. Например,

для получения затрат по инфраструктуре можно выполнить следующий SQL-запрос:

```
select sum(fact_table.cost)
from fact_table, dimension_table D1,
dimension_table D2
where fact_table.dimension_id = D2.id
and D2.left >= D1.left
and D2.right <= D1.right
and D1.name = «Инфраструктура»
```

Для простоты работы с таким справочником кроме полей *left*, *right* стоит добавить еще два поля: «*Level*» – уровень узла в дереве, «*Is_leaf*» – флаг, показывающий, является ли узел листом в дереве или нет. Таким образом, мы получаем таблицу «*dimension_table*» (табл. 2.2), которая позволяет хранить дерево любой глубины вложенности и размерности и выбирать потомков и родителей с помощью одного запроса.

Таблица 2.2

Представление иерархий с помощью левой и правой границы

ID	NAME	LEFT	RIGHT	LEVEL	IS_LEAF
1	Предприятие	1	16	1	N
2	Управление	2	3	2	Y
3	Инфраструктура	4	9	2	N
4	Производство	10	15	2	N
5	Энергия	5	6	3	Y
6	Сервисные услуги	7	8	3	Y
7	Месторождение А	11	12	3	Y
8	Месторождение Б	13	14	3	Y

Другой способ, описанный Ральфом Кимбаллом, основан на введении вспомогательной таблицы («*helper_table*»), через которую осуществляется связь таблицы фактов с таблицей измерения. Эта вспомогательная таблица отражает иерархическую структуру измерения и подчиняется следующему закону: вспомогательная таблица содержит весь набор пар «родитель-потомок», причем потомок может не быть непосредственным потомком родителя. Структура такой таблицы и ее содержимое показано в табл. 2.3.

Таблица 2.3

Структура и содержание вспомогательной таблицы

PARENT ID	CHILD ID	DISTANCE	IS LEAF
1	1	0	N
1	2	1	N
1	3	1	N
1	4	1	N
1	5	2	N
1	6	2	N
1	7	2	N
1	8	2	N
2	2	0	Y
3	3	0	N
3	5	1	N
3	6	1	N
4	4	0	N
4	7	1	N
4	8	1	N
5	5	0	Y
6	6	0	Y
7	7	0	Y
8	8	0	Y

Связывая таблицу фактов (рис. 2.3) с идентификатором потомка во вспомогательной таблице, а таблицу измерений с идентификатором родителя, мы можем вычислять сумму затрат для каждого места возникновения затрат и всех его составляющих одним запросом, как и в предыдущем случае. При этом, добавляя ограничения на поля *Distance* и *Is Leaf*, мы можем легко считать затраты для любого уровня в иерархии.

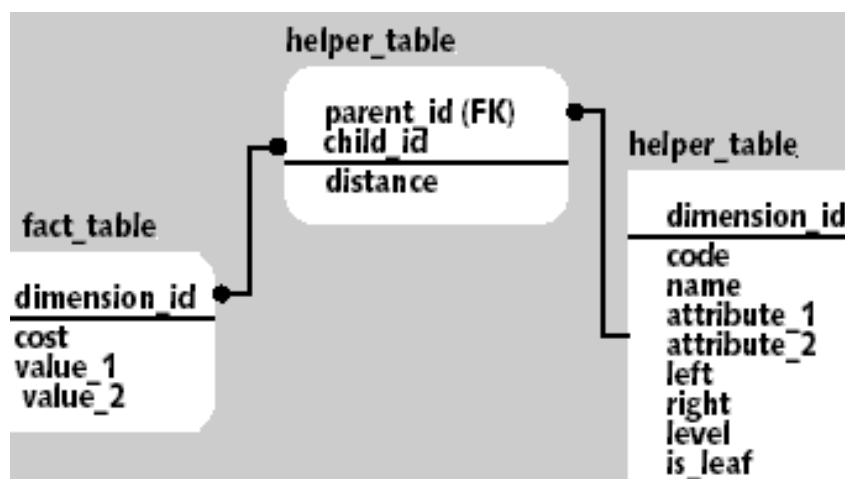


Рис. 2.3. Модель иерархического справочника со вспомогательной таблицей

Например, для того чтобы посчитать сумму затрат, возникающих в местах, находящихся по иерархии на один уровень ниже «Инфраструктуры», необходимо выполнить следующий SQL-запрос:

```
select sum(fact_table.cost)
from fact_table, dimension_table, helper_table
where fact_table.dimension_id =
helper_table.child_id
and dimension_table.dimension_id =
helper_table.parent_id
and dimension_table.name = «Инфраструктура»
and helper_table.distance = 1
```

Проблема проектирования иерархических справочников еще более усложняется, когда измерение может иметь несколько альтернативных иерархий, и становится совсем трудноразрешимой при необходимости поддерживать историю изменения таблицы измерения.

Важный момент, с которым часто приходится сталкиваться разработчику хранилища данных, связан с агрегатными значениями. Этот класс задач условно можно разделить на два подкласса. Первый рассматривает задачи создания и поддержания агрегатов по имеющимся детальным данным и широко освещен в литературе. Второй связан с тем, что источники данных для хранилища представляют собой не детальные значения, а уже некоторый набор агрегированных данных. Такая ситуация типична при создании хранилищ для управляющих компаний и государственных контролирующих органов, собирающих множество отчетных форм.

Крайним случаем такого подхода является модель, которую условно можно назвать «показатель-значение». Суть ее состоит в том, что собирается большой набор показателей, характеризующих финансово-хозяйственную деятельность предприятия. Эти показатели могут быть как связанными между собой функционально, так и нет, могут отражать одни и те же величины, но с разной степенью детализации, и т. д. При попытке представить такие данные в виде многомерной модели разработчик сталкивается со значительными проблемами и очень часто идет по пути создания не хранилища данных, а хранилища форм. Типичное хранилище форм строится на основе трех измерений: экономические показатели, время, отчетные формы; таблицы фактов – значения экономических показателей и вспомогательных таблиц, описывающих, как показатели и их значения расположены в отчетных формах. При анализе таких данных аналитик будет испытывать значительные трудности, связанные главным образом с тем, что показатели различных форм

нельзя сравнивать между собой. Единственное, что ему остается, – это отслеживание изменений показателей одной формы во времени.

2.3. Многомерное моделирование и звездообразные схемы

Выше мы вскользь упомянули о том, что хранилища данных можно моделировать методами многомерного моделирования с использованием звездообразной схемы. Повторим, что традиционные методы моделирования данных и проектирования баз данных, как правило, не позволяют получить структуры данных, оптимизированные для массовых запросов. Поэтому придется отбросить некоторые традиционные идеи о проектировании и выучить ряд новых приемов. В этом разделе частично затрагиваются вопросы анализа, потому что это важно для понимания того, как определяются эти модели.

Как и большинство понятий в компьютерном мире, звездообразная схема имеет несколько синонимов: *многомерная схема*, *куб данных* и *соединение по схеме «звезда»*. Причина, по которой эту схему называют звездообразной, заключается в том, что ее графическое представление напоминает звезду.

Звездообразная модель состоит из центральной *таблицы фактов (fact table)*, которая окружена несколькими *таблицами измерений (dimension table)*. Физически таблица фактов часто представляет собой несколько секционированных таблиц. Термин «секционирование» очень широко используется специалистами по базам данных и в общем случае означает формирование подмножеств данных.

Отношения между таблицей фактов и таблицами измерений должны быть простыми, чтобы существовал только один возможный путь соединения любых двух таблиц и чтобы смысл этого соединения был очевиден и хорошо понятен. Преимущество простоты отношений состоит в том, что это помогает повысить производительность.

Все, что аналитик должен сделать при многомерном моделировании, – это выявить факты и их измерения. Факты обычно представляют собой основные виды бизнес-деятельности организации и факторы, влияющие на данный бизнес или его сектор. Что можно сказать о таблицах измерений? В широком смысле слова – это элементы, которые могут оказывать определенное влияние или порождать различные тенденции в развитии фактов. Измерения, исходя из этого, можно разбить на следующие категории: *люди, места, вещи и время* (рис. 2.4).

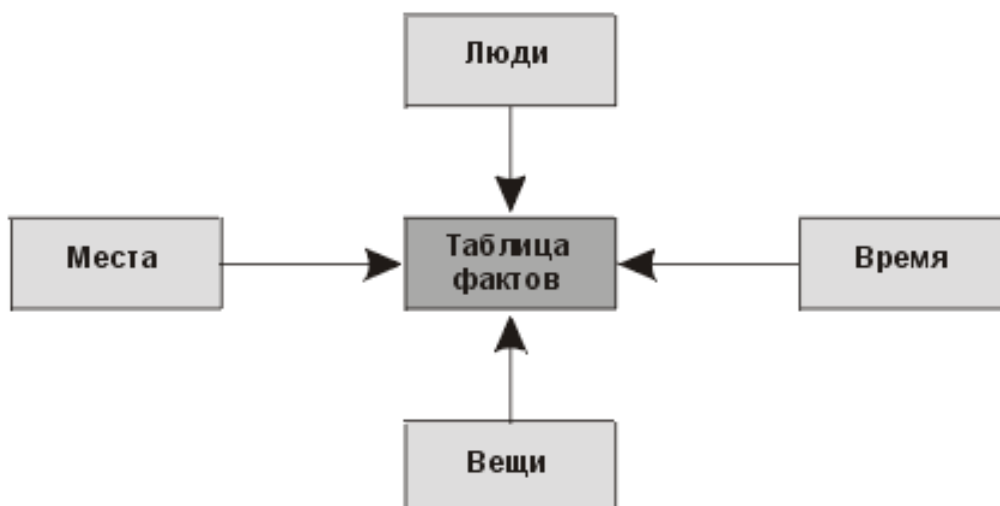


Рис. 2.4. Классификация измерений: люди, места, вещи и время

Рассматривайте таблицу фактов в свете реляционной теории: у нее есть внешний ключ к каждой соответствующей таблице измерения. Она объединяет измерения в нечто, представляющее собой значимое событие, и содержит качественную и количественную информацию об этом событии. В сценарии продаж таблица фактов может содержать данные о том, сколько товара реализовано и на какую сумму, а также внешние ключи к таблицам измерений, характеризующим операцию продажи (какой товар, когда он был продан, кто продал его и какой способ платежа был использован).

Временное измерение в звездообразной схеме обладает рядом интересных свойств. Факт, скорее всего, будет связан не с одним моментом времени, а с каким-то временным промежутком. Следовательно, временное измерение обычно является обобщающим. При выборе единицы изменения времени приходится выбирать между хранением огромного числа фактов в хранилище, которые в большинстве запросов необходимо будет обобщать, и отсутствием данных о сезонных тенденциях вследствие того, что при обобщении выбран слишком большой период времени. Временное измерение также часто выступает в качестве основы для секционирования таблицы фактов. Во многих случаях секционирование необходимо для уменьшения объема таблицы фактов и облегчения управления ею. Время же используется потому, что при этом, как правило, получают приблизительно равные по объему кванты и оно часто является наиболее логичным для секционирования измерением. Кроме того, время – наиболее вероятная основа для усечения таблицы фактов, если когда-нибудь удастся заставить предприятие согласиться на удаление данных из его хранилища.

Выбор степени детализации временного измерения (неделя, день, месяц и т. д.) может быть исключительно важным. Если, например, выбрать месяц, то средство анализа, возможно, сможет обобщить данные в еще большей степени и показать результаты по годам. Затем с помощью различных методов детализации можно представить данные по кварталам нужного года, а потом вновь по месяцам. Очевидно, что в этом случае нельзя выполнить более подробную временную детализацию, поскольку для этого нет данных. Выбор степени детализации по времени имеет большое значение, потому что позволяет уравновесить два фактора: необходимость в более детальной разбивке и объем хранилища данных. Таким образом, при проектировании хранилища данных необходимо уделять должное внимание временным измерениям и в каждом конкретном случае принимать решение о необходимой для обобщения степени детализации.

2.4. Пример «Система мониторинга подземных вод»

Рассмотрим процесс проектирования хранилища данных на примере системы мониторинга подземных вод.

Первым шагом проектирования структуры хранилища данных является выявление измерений и фактов. Количество таблиц фактов и измерений для этих фактов может варьироваться в зависимости от размера хранилища и его использования. В нашем примере это четыре измерения («Скважины», «Количество воды», «Цели отбора» и «Время») и один факт (сам отбор воды в определенное время с целью мониторинга состояния подземных вод).

Очень важным моментом является выбор степени детализации временных измерений, которая может составлять годы, месяцы, недели, дни и т. д. Эта характеристика выбирается исходя непосредственно из специфики проектируемого хранилища данных и его будущего использования. В данном примере целесообразно обобщать данные по неделям. На рис. 2.5 представлена звездообразная схема для системы мониторинга подземных вод.

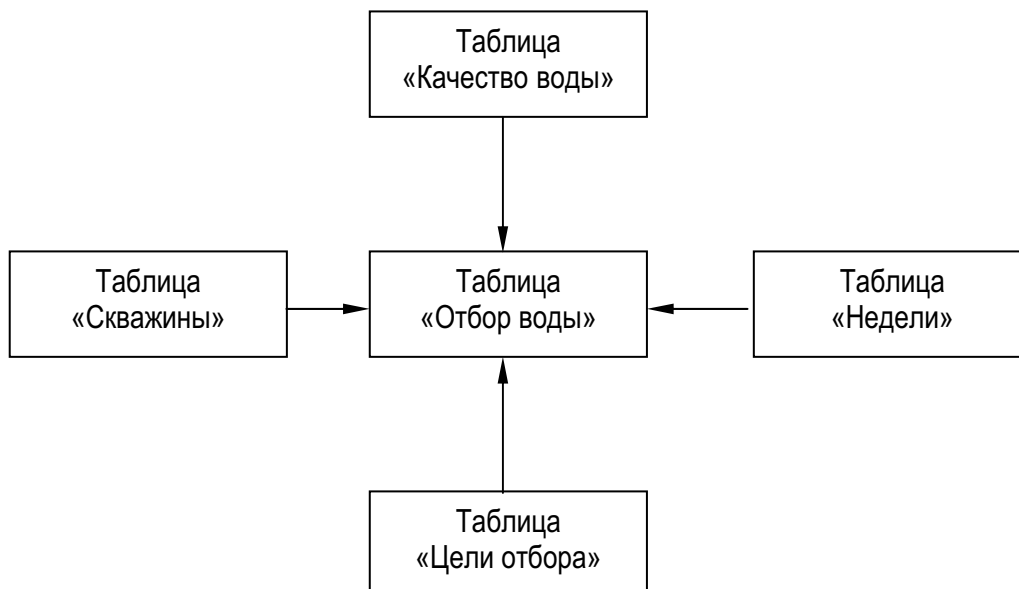


Рис. 2.5. Звездообразная схема системы мониторинга подземных вод

Далее следует определить таблицы измерений и фактов, т. е. наполнение их непосредственно атрибутами. На рис. 2.6 показаны определения таблиц, которые используются для реализации звездообразной схемы, изображенной на рис. 2.5.

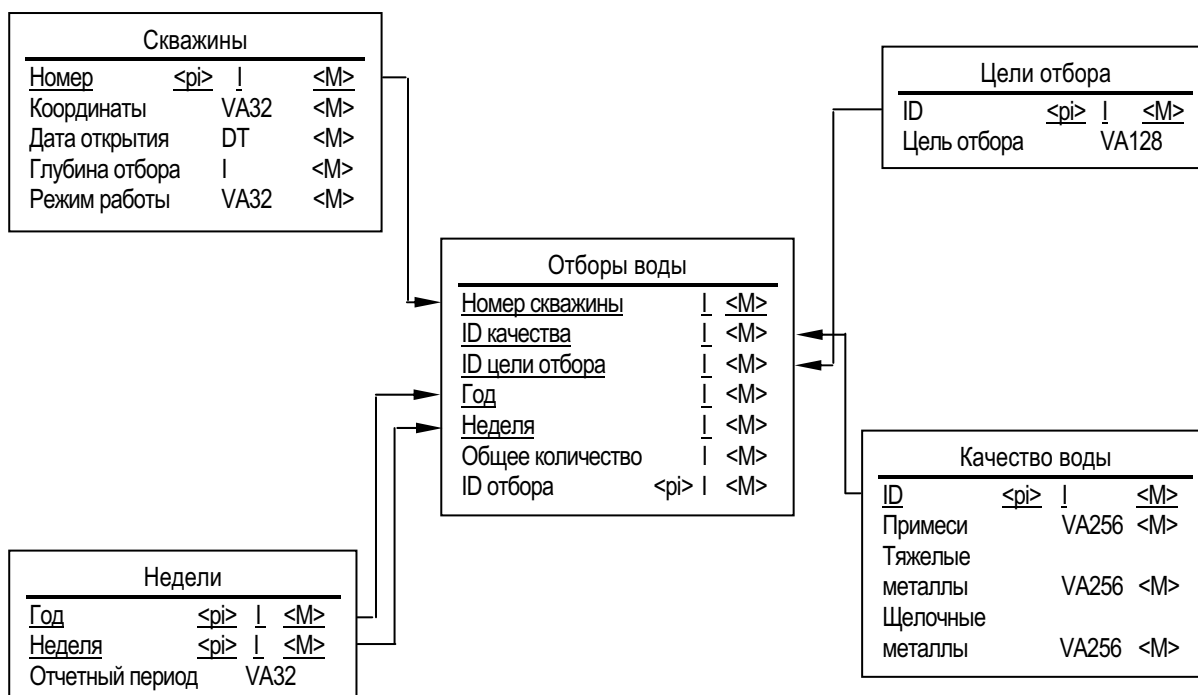


Рис. 2.6. Определение таблиц для реализации звездообразной схемы сбыта продукции

После определения таблиц необходимо проверить таблицы измерений на предмет множественного соединения между ними. В данном случае таблицы «Скважины» единственным образом связаны с таблицей «Цели отбора» через таблицу фактов. Справедливо то же самое и для остальных таблиц, что удовлетворяет основному требованию к таблицам измерений и фактов.

Для исключения возможности использования лишних измерений необходимо проверить, все ли из выбранных измерений влияют на таблицу фактов. Так, например, «Качество воды» влияет на таблицу фактов, т. к. при обобщении сведений за неделю может оказаться, что при неизменных остальных измерениях «Качество воды» будет иным, чем в предыдущий промежуток. Тем самым факт изменился. Если окажется так, что данное измерение лишнее, то его следует удалить.

2.5. Контрольные вопросы

1. Какие типовые архитектуры хранилищ данных Вам известны?
2. Чем обусловлен выбор модели хранилища данных?
3. В чем особенность звездообразных схем структуры хранилищ данных?

2.6. Задание

На основании БД, созданной в ходе предыдущей лабораторной работы, спроектируйте структуру хранилища данных. Для этого необходимо:

- 1) выделить измерения и факты хранилища данных;
- 2) определить структуру измерений и фактов хранилища данных;
- 3) проверить таблицы на предмет множественности связей между ними;
- 4) проверить, все ли измерения влияют на таблицу фактов.

3. РЕАЛИЗАЦИЯ ХРАНИЛИЩА ДАННЫХ ПОД УПРАВЛЕНИЕМ MICROSOFT SQL SERVER 2000 ANALYSIS SERVICES

3.1. Измерения (Dimension)

В службах *Microsoft SQL Server 2000 Analysis Services (SSAS)* измерения являются основными компонентами куба. В измерениях данные привязаны к некоторой предметной области, например заказчики, магазины или служащие. В службах *Analysis Services* измерения содержат атрибуты, которые соответствуют столбцам в таблицах измерения. Эти атрибуты выглядят в виде иерархий атрибутов и могут быть организованы в многоуровневые иерархические структуры. Эти иерархии применяются для организации мер, которые содержатся в кубе.

Все измерения основаны на таблицах или представлениях в терминах источника данных. Измерения существуют независимо от куба, могут использоваться в нескольких кубах или многократно в одном кубе. Измерение, существующее независимо от куба, называется измерением базы данных, а измерение, используемое в кубе, называется измерением куба.

Структура измерения в основном определяется структурой таблицы или таблиц базового измерения. Простейшая структура называется схемой «звезда», в которой каждое измерение основано на одной таблице измерения, которая непосредственно связана с таблицей фактов связью первичного и внешнего ключей.

Простейшим методом определения измерений базы данных и куба является использование мастера кубов для создания их одновременно с определением куба. Мастер кубов создаст измерения на основе таблиц измерений, найденных им или указанных пользователем из представления источника данных, используемого для куба. После этого мастер создает измерения базы данных и добавляет их к новому кубу, создавая измерения куба.

При создании куба также можно добавить любые измерения, которые уже существуют в базе данных. Эти измерения могли быть ранее определены для другого куба или с помощью мастера измерений. После создания измерения базы данных его можно отредактировать в конструкторе измерений.

3.2. Мера (Measure)

Мера представляет собой, как правило, столбец, содержащий количественные, статистически вычисляемые (обычно числовые) данные, соответствующие столбцу в таблице фактов. Выражение меры также может использоваться для определения значения меры на основе столбца в таблице фактов, измененной многомерным выражением. Выражение меры позволяет выполнить взвешивание значения меры, например для преобразования валют при взвешивании меры продаж по валютному курсу.

Для определения мер можно использовать столбцы атрибутов таблиц измерений, но такие меры являются, как правило, полуаддитивными или неаддитивными (в зависимости от режима статистического вычисления).

Кроме того, можно определить вычисляемую меру с помощью многомерных выражений, чтобы получить вычисляемое значение меры на основе других мер в кубе. Такие меры, называемые вычисляемыми, добавляют кубу в службах *Analysis Services* гибкие функциональные возможности анализа. В кубе меры сгруппированы по таблице фактов в группу мер. Группы мер используют для связи измерений с мерами, а также для меры с отдельным счетчиком в качестве режима статистического вычисления, что позволяет выполнять оптимальное статистическое вычисление.

Пересечение отдельной меры и отдельного элемента всех измерений, включенных в группу мер, представлено в службах *Analysis Services* в виде отдельной ячейки куба.

Меры и группы мер имеют свойства, позволяющие определять и контролировать, как меры и группы мер функционируют и отображаются для пользователей.

Свойства группы мер

Свойства группы мер определяют характеристики всей группы мер и устанавливают характеристики по умолчанию для определенных свойств мер в группе мер.

Таблица 3.1

Свойство	Определение
AggregationPrefix	Общий префикс, используемый для имен агрегатов
DataAggregation	Определяет, могут ли службы <i>Analysis Services</i> осуществлять статистическое вычисление сохраняемых данных или кешированных данных для группы мер. По умолчанию статистическое вычисление сохраняемых данных и кешированных данных разрешено
ErrorConfiguration	Настраиваемые параметры обработки ошибок для обработки дублирующихся ключей, неизвестных ключей, пределов ошибок, действий при обнаружении ошибки, файла журнала ошибок и ключа NULL
IgnoreUnrelatedDimensions	Определяет, будут ли несвязанные измерения принудительно перемещаться на их верхний уровень, когда элементы измерений, не связанных с группой мер, включаются в запрос. Значение по умолчанию равно <i>True</i>
ProactiveCaching	Настраиваемые параметры обработки ошибок для обработки дублирующихся ключей, неизвестных ключей, пределов ошибок, действий при обнаружении ошибки, файла журнала ошибок и ключа NULL
ProcessingMode	Указывает, должны ли статистическое вычисление и индексирование выполняться во время или после обработки (параметры – обычная или отложенная обработка)
ProcessingPriority	Определяет приоритет обработки куба во время фоновых операций, например отложенных статистических вычислений или индексирования. Значение по умолчанию равно 0
StorageLocation	Место хранения файловой системы для группы показателей. Если местоположение не указано, то оно наследуется из куба, содержащего эту группу мер
StorageMode	Режим хранения для группы мер, значения MOLAP, ROLAP или HOLAP
Type	Указывает тип группы мер

Свойства мер

Меры наследуют определенные свойства у группы мер, элементами которых они являются, если только эти свойства не переопределены на уровне мер. Свойства мер определяют статистическое вычисление мер, их тип данных, папку для отображения мер, строку форматирования, любые выражения мер, их понятные имена, базовые исходные столбцы и видимость для пользователей.

Таблица 3.2

Свойство	Определение
AggregateFunction	Определяет статистическое вычисление мер. Дополнительные сведения приведены ниже
DataType	Тип данных столбца базовой таблицы фактов, с которым связана мера
Description	Описание меры, которая может открываться в клиентских приложениях
DisplayFolder	Папка, в которой группа мер будет отображаться, когда пользователи подключаются к кубу, позволяющая распределять меры по категориям для повышения удобства просмотра пользователями куба, содержащего множество мер
FormatString	Определяет формат отображения. Более подробные сведения приведены ниже
ID	Уникальный идентификатор (ID) меры
MeasureExpression	Содержит многомерное выражение, определяющее меру
Name	Имя меры, понятное для пользователя
Source	Столбец в представлении источника данных, к которому привязана мера
Visible	Определяет, отображается мера или нет

Куб (*cube*)

В службах *Microsoft SQL Server 2000 Analysis Services* куб разрабатывается на основе таблиц и представлений, моделируемых в представлении источника данных. Куб представляет собой набор мер, которые являются фактами, и измерений, которые являются интересующими областями, такими как, например, время, продукт и заказчик. Куб дополняется вычислениями, ключевыми индикаторами производительности, действиями, секциями, перспективами и преобразованиями. Куб, в сущности, является синонимом унифицированной многомерной модели.

Меры куба основаны на столбцах из одной или нескольких таблиц фактов, а элементы измерений куба – на столбцах из одной или нескольких таблиц измерений. Куб также может быть разработан без базового реляционного источника данных. В этом случае может быть сформирована базовая реляционная структура для поддержки куба. Факты в кубе статистически вычисляются на основе иерархий измерений.

Свойства кубов

Кубы имеют ряд свойств, изменяя параметры которых можно влиять на поведение всего куба, и несколько неизменяемых свойств. Данные свойства перечислены в следующей табл. 3.3.

Таблица 3.3

Свойство	Определение
AggregationPrefix	Общий префикс, используемый для имен агрегатов
Collation	Идентификатор языка и флаг сравнения, отделенный подчеркиванием, например <i>Latin1_General_CI_AS</i>
DefaultMeasure	Многомерное выражение, определяющее меру по умолчанию для куба
Description	Описание куба, которое может открываться в клиентских приложениях
ErrorConfiguration	Настраиваемые параметры обработки ошибок для обработки дублирующихся ключей, неизвестных ключей, пределов ошибок, действий при обнаружении ошибки, файла журнала ошибок и ключа NULL
EstimatedRows	Количество предполагаемых строк в кубе
ID	Уникальный идентификатор (ID) куба
Language	Идентификатор языка куба
Name	Имя куба, понятное для пользователя
ProactiveCaching	Настройки упреждающего кеша для куба
ProcessingMode	Указывает, должны ли статистическое вычисление и индексирование выполняться во время или после обработки (параметры – обычная или отложенная обработка)
ProcessingPriority	Определяет приоритет обработки куба во время фоновых операций, например отложенных статистических вычислений или индексирования. Значение по умолчанию равно 0

Свойство	Определение
ScriptCacheProcessingMode	Указывает на необходимость создания кеша сценария во время или после обработки (параметры – обычная или отложенная обработка)
ScriptErrorHandlingMode	Определяет обработку ошибок, параметры – <i>IgnoreNone</i> или <i>IgnoreAll</i>
Source	Представление источника данных, используемое для куба
StorageLocation	Место хранения файловой системы для куба. Если не задано иное, то место хранения наследуется из базы данных, содержащей объект куба
StorageMode	Режим хранения для куба, значения – MOLAP, ROLAP или HOLAP
Visible	Определяет, отображается куб или нет

3.3. Реализация

Analysis Manager – программа, работающая под управлением *Microsoft Management Console* (MMC). Она представляет собой утилиту, входящую в состав *Analyses Services* и предназначенную главным образом для администраторов баз данных OLAP.

Как запускать *Analysis Manager*

Щелкните кнопку *Start*, пункт Программы, *Microsoft SQL Service*, *Analyses Services*, затем щелкните *Analysis Manager*.

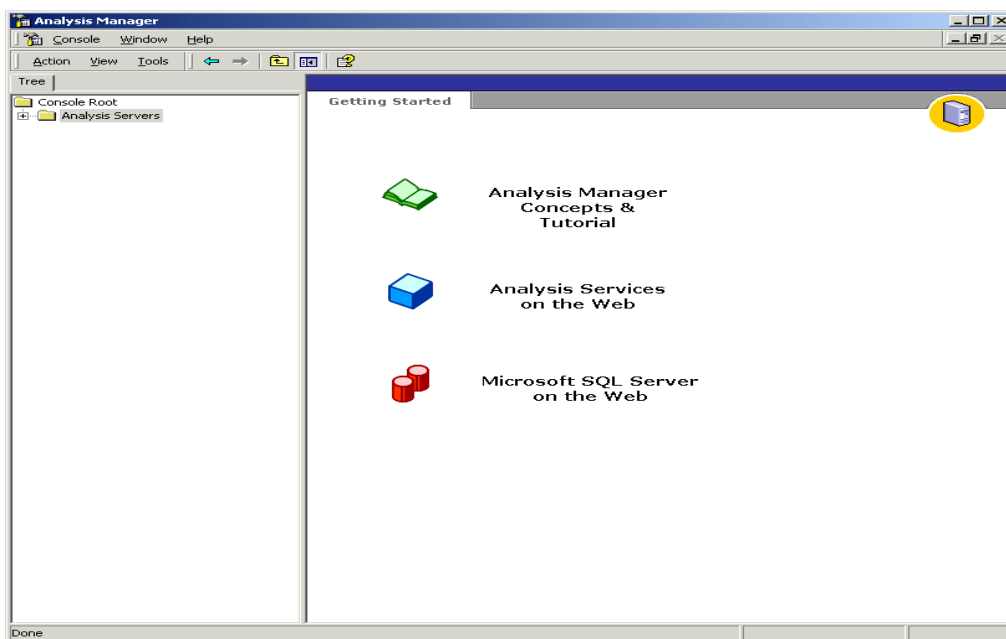


Рис. 3.1. Окно программы *Analysis Manager*

Как настраивать структуру базы данных

1. В *Analysis Manager* в дереве раскройте *Analyses Services*.
2. Щелкните название Вашего сервера. Подключение с сервером анализа будет установлено.
3. Щелкните правой кнопкой мыши на названии Вашего сервера, затем щелкните *New Database*.
4. В диалоговом окне *Database*, в блоке имени *Базы данных*, вводите *Tutorial*, затем щелкните ОК.
5. В области окна дерева *Analysis Manager* раскройте сервер, затем расширьте базу данных *Tutorial*, которую Вы только что создавали. Ваша новая база данных *Tutorial* содержит следующие элементы:
 - *Источники Данных*
 - *Кубы*
 - *Общедоступные Измерения*
 - *Добывающие Модели*
 - *Роли Базы данных*

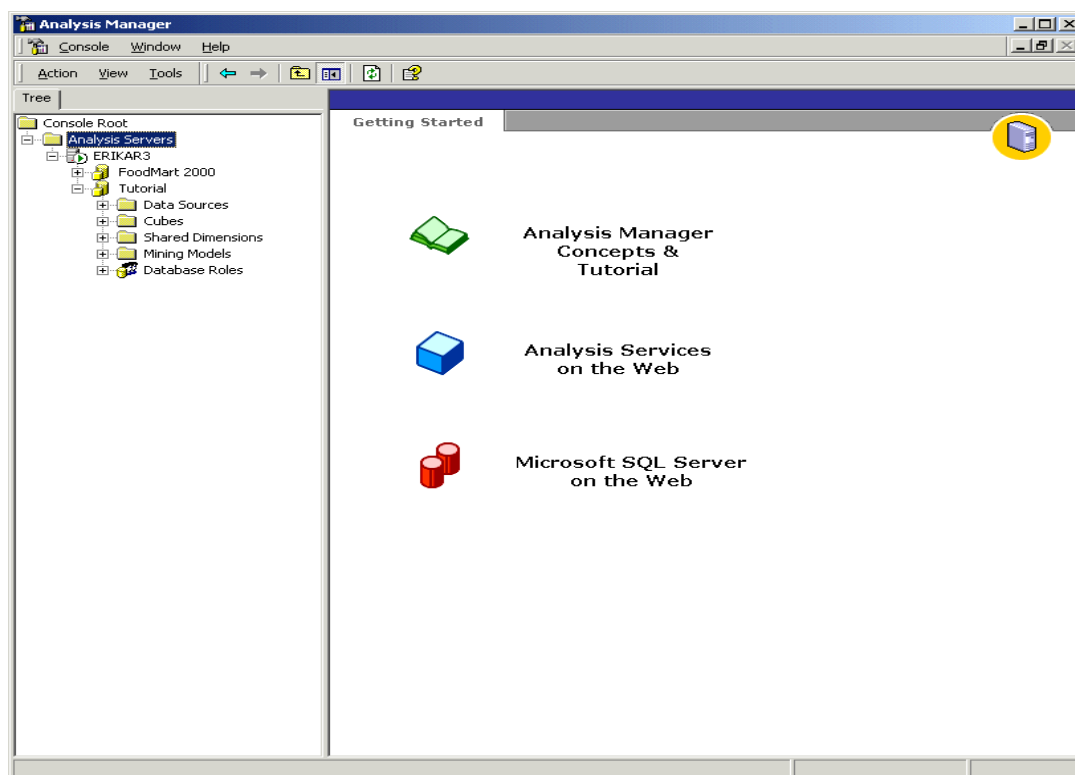


Рис. 3.2. База данных *Tutorial*

Затем установите подключение к типовым данным в источнике данных *Tutorial*. Этот пример будет использоваться для всех упражнений в этой обучающей программе.

Установка источника данных в *Менеджере Анализа* подключает Вашу базу данных к системному имени источника данных (DSN), которое устанавливается в ODBC – *Администраторе Источника Данных*.

Как настраивать источник данных

1. В *Analysis Manager* в области дерева щелкните правой кнопкой мыши на папке *Источники Данных* под базой данных *Tutorial*, затем щелкните *New Data Source*.
2. В диалоговом окне *Data Link Properties* щелкните вкладку *Provider*, затем щелкните *Microsoft OLE DB Provider for ODBC Drivers*.

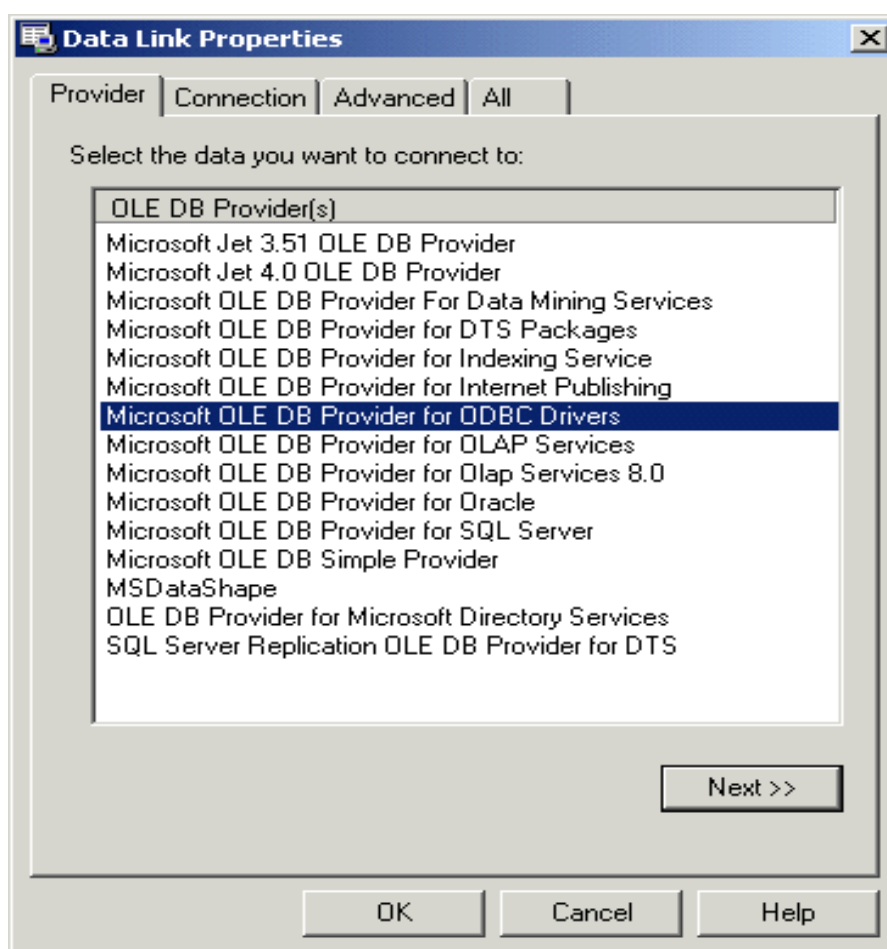


Рис. 3.3. Выбор провайдера

3. Щелкните вкладку *Connection*, затем в списке имен источников данных щелкните *Tutorial*.

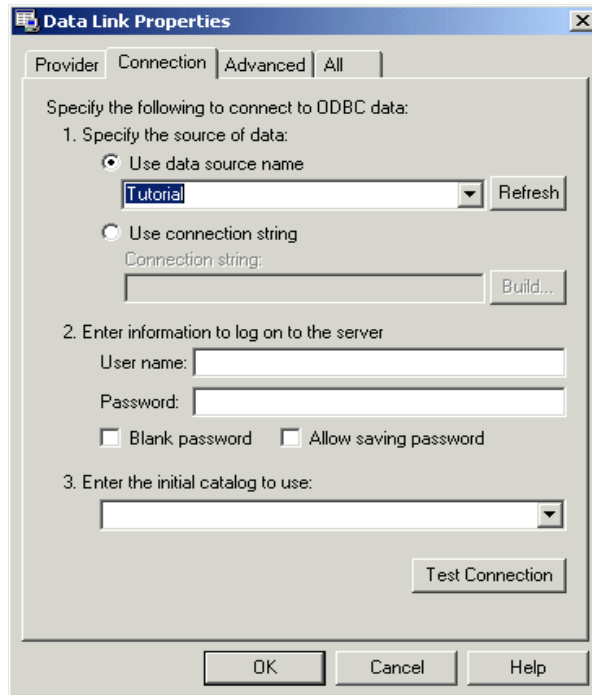


Рис. 3.4. Продолжение шага выбора провайдера

4. Щелкните *Test Connection*, чтобы убедиться, что все работает.
5. Щелкните *OK*, чтобы закрыть диалоговое окно *Data Link Properties*.

Построение куба

Куб – многомерная структура данных. Кубы определены набором измерений и мер.

Как открыть Мастер Куба

В *Менеджере Анализа* в области окна дерева, под базой данных *Tutorial*, щелкните правой кнопкой мыши на папке кубов, выберите новый куб и затем щелкните *Wizard*.

Как добавлять меры к кубу

Меры – количественные значения в базе данных, которую Вы хотите анализировать. Используемые обычно меры: коммерческие, стоимость и данные бюджета. Меры проанализированы против различных категорий измерения куба.

1. На шаге *Welcome Мастера Куба*, щелкните *Next*.
2. В выборе таблицы факта на шаге источника данных, раскройте источник данных *Tutorial*, затем щелкните *sales_fact_1998*.
3. Вы можете рассматривать данные в таблице *sales_fact_1998*, щелкая данные *Browse*.
4. Чтобы определить меры для Вашего куба, под таблицей *Факта*, дважды щелкните на столбце *store_sales*. Повторите эту процедуру для столбцов *store_cost* и *unit_sales*, затем щелкните *Next*.

Как построить измерение времени

1. На шаге *Select the dimensions for your cube* Мастера Куба щелкните *New Dimension*, чтобы вызвать Мастер Измерения.

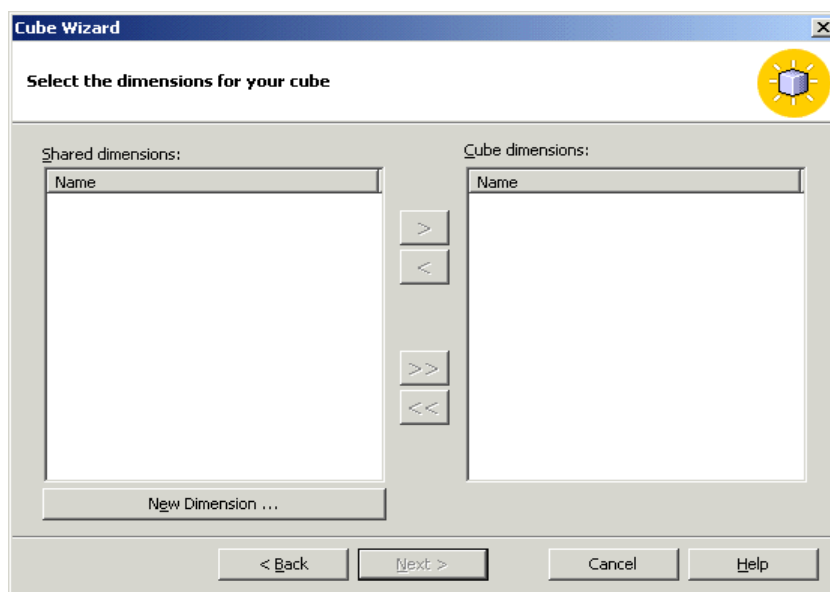


Рис. 3.5. Окно выбора измерения

2. На шаге *Welcome* щелкните *Next*.
3. Выберите *Star Schema* для одиночной таблицы измерения, затем щелкните *Next*.
4. На шаге выбора таблицы измерения щелкните *time_by_day*. Вы можете просматривать данные, содержащиеся в таблице *time_by_day*, щелкая *Browse Data*.

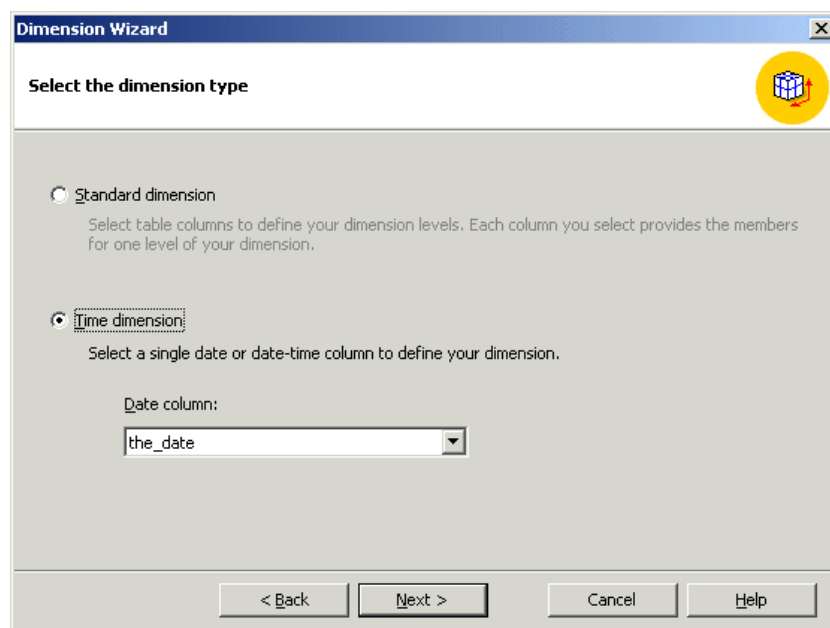


Рис. 3.6. Окно выбора типа измерения

5. На шаге выбора типа измерения выберите *Time Dimension*, затем щелкните *Next*.
6. Затем определите уровни для Вашего измерения. На следующем шаге щелкните *Select time level*, выберите *Year*, *Quarter*, *Month*, затем щелкните *Next*.
7. На шаге дополнительных параметров выбора щелкните *Next*.
8. На последнем шаге мастера введите имя Вашего нового измерения.
Обратите внимание: Вы можете определить, будет ли это измерение разделено или возможно только частное использование ресурсов этого измерения при помощи переключателя, который расположен в нижнем левом углу экрана.
9. Щелкните *Finish*, чтобы вернуться к *Мастеру Куба*.
10. В *Мастере Куба* теперь отражается измерение времени в *Списке Измерений Куба*.

Как построить измерение *product*

1. Щелкните *New Dimension* снова. На шаге *Welcome Мастера Измерения* щелкните *Next*.
2. Выберите *Snowflake Schema* и затем щелкните *Next*.
3. На шаге выбора таблиц измерения дважды щелкните *product* и *product_class*, чтобы добавить их к выбранным таблицам. Щелкните *Next*.
4. Эти две таблицы, которые Вы выбрали на предыдущем и текущем шаге, соединяются между собой. Щелкните *Next*.

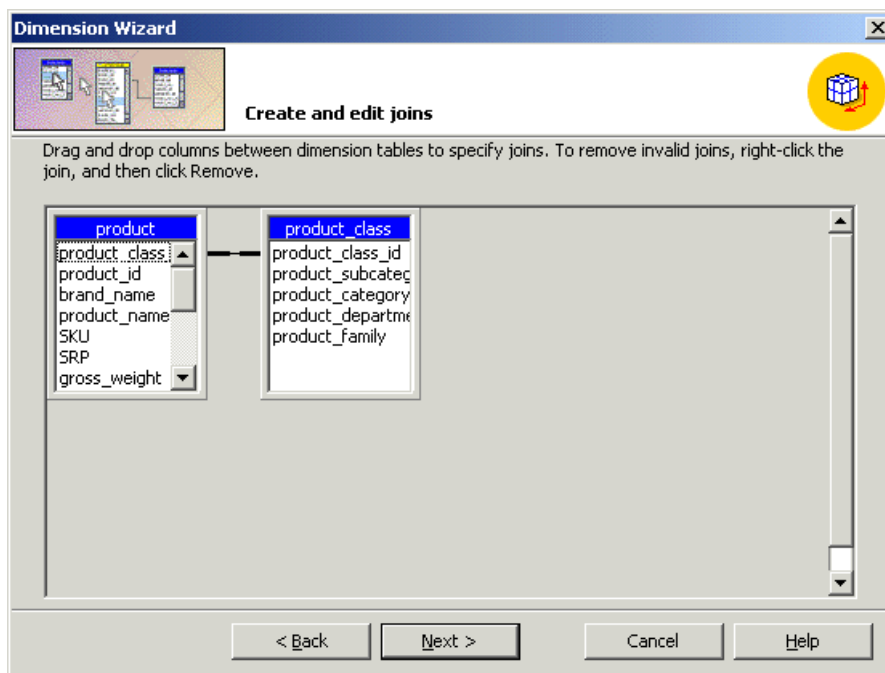


Рис. 3.7. Окно создания и редактирования соединения

5. Чтобы определить уровни для измерения, под столбцами *Available*, дважды щелкните столбцы *product_category*, *product_subcategory* и *brand_name*, в том же порядке. После того, как Вы дважды щелкните каждый столбец, его имя появляется под уровнями измерений. Щелкните *Next* после того, как выбрали все три столбца.
6. На шаге определения ключей столбцов щелкните *Next*.
7. На шаге дополнительных параметров выбора щелкните *Next*.
8. На последнем шаге мастера введите *product* в блок названия измерения. Щелкните *Finish*.
9. Вы должны увидеть измерение *product* в списке измерений куба.

Как построить измерение *Customer*

1. Щелкните *New dimension*.
2. На шаге *Welcome Мастера Измерения* щелкните *Next*.
3. Выберите *Star Schema* и затем щелкните *Next*.
4. На шаге выбора таблицы измерения щелкните *Customer*, затем щелкните *Next*.
5. На шаге выбора типа измерения щелкните *Next*.
6. Чтобы определить уровни для Вашего измерения, под столбцами *Available* дважды щелкните столбцы *Country*, *State_Province*, *City* и *Iname* в том же порядке. После того, как Вы дважды щелкаете каждый столбец, его имя появляется под уровнями измерений. После того, как Вы выбрали все четыре столбца, щелкните *Next*.
7. На шаге определения ключей столбцов щелкните *Next*.
8. На шаге дополнительных параметров выбора щелкните *Next*.
9. На последнем шаге мастера введите *Customer* в блок названия измерения. Щелкните *Finish*.
10. Вы должны увидеть измерение *Customer* в списке измерений куба.

Как построить измерение *store*

1. Щелкните *New dimension*.
2. На шаге *Welcome Мастера Измерения* щелкните *Next*.
3. Выберите *Star Schema* и затем щелкните *Next*.
4. На шаге выбора таблицы измерения щелкните *store*, затем щелкните *Next*.
5. На шаге выбора типа измерения щелкните *Next*.
6. Чтобы определить уровни для Вашего измерения, под столбцами *Available* дважды щелкните столбцы *store_country*, *store_state*, *store_city* и *store_name* в том же порядке. После того, как Вы дважды щелкните каждый столбец, его имя появляется под уровнями измерений. После того, как Вы выбрали все четыре столбца, щелкните *Next*.
7. На шаге определения ключей столбцов щелкните *Next*.

8. На шаге дополнительных параметров выбора щелкните *Next*.
9. На последнем шаге мастера введите *store* в блок названия измерения. Щелкните *Finish*.
10. Вы должны увидеть измерение *store* в списке измерений куба.

Как закончить построение куба

1. В *Мастере Куба* щелкните *Next*.
2. Щелкните *Yes* в окне сообщения.

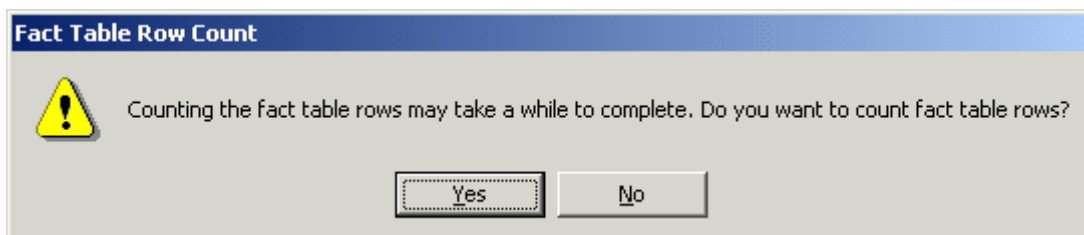


Рис. 3.8. Окно Fact Table Row Count

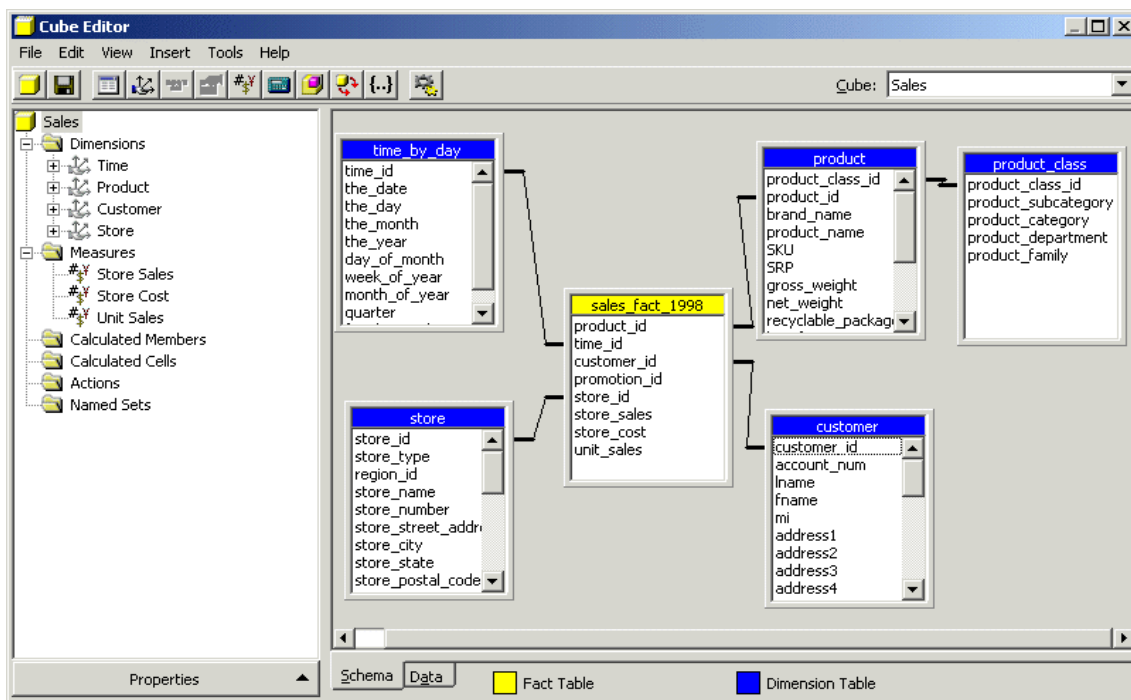


Рис. 3.9. Окно редактирования куба

3. На последнем шаге *Мастера Куба* назовите Ваш куб *Sales*, затем щелкните *Finish*.
4. Мастер закрывается и затем запускается *Редактор Куба*, который содержит новый куб. Нажимая по синим или желтым областям заголовка, упорядочьте таблицы так, чтобы они соответствовали следующей иллюстрации. **Обратите внимание:** Вы не должны закрывать *Редактор Куба*, т. к. будете редактировать куб в следующем разделе обучающей программы.

Редактирование куба

Вы можете использовать два метода для доступа к схеме *Редактор Куба*:

- в менеджере анализа в области дерева щелкните правой кнопкой мыши на существующем кубе, затем щелкните *Edit*;
- можно создать новый куб, используя *Редактор Куба* непосредственно. Этот метод не рекомендуется, если Вы не опытный пользователь.

Если Вы продолжаете работу от предыдущего раздела, Вы должны уже быть в редакторе куба.

В области окна схемы *Редактор Куба*, Вы можете видеть таблицу фактов (с желтой областью заголовка) с соединенными таблицами измерений (синие области заголовка). В области окна дерева *Редактор Куба* возможен предварительный просмотр структуры куба в иерархическом дереве. Вы можете редактировать свойства куба, щелкая кнопку *Properties* внизу левой области окна.

Хранение проекта и обработка куба

Вы можете проектировать опции хранения для данных в Вашем кубе. Прежде, чем Вы можете использовать или просматривать данные в Ваших кубах, Вы должны обработать их.

Как проектировать хранение, используя Мастер Проекта Хранения

1. В область дерева менеджера анализа раскройте папку кубов, щелкните правой кнопкой мыши на *Commerse Cube*, затем щелкните *Design Storage*.
2. На шаге *Welcome* щелкните *Next*.
3. Выберите *MOLAP* в качестве Вашего типа хранения данных, затем щелкните *Next*.
4. Под опциями набора нажмите *Performance gain reaches*. В блоке введите 40 %. Это настраивает *Analyses Services* на повышение производительности на 40 %, независимо от того, сколько дискового места это требует. Администраторы могут использовать эту способность настройки для сбалансирования потребности в производительности запросов и дискового пространства, требуемого для того, чтобы хранить данные агрегатов.
5. Щелкните *Start*.
6. Вы можете наблюдать выполнение в правой части мастера. По окончании работы щелкните *Next*.

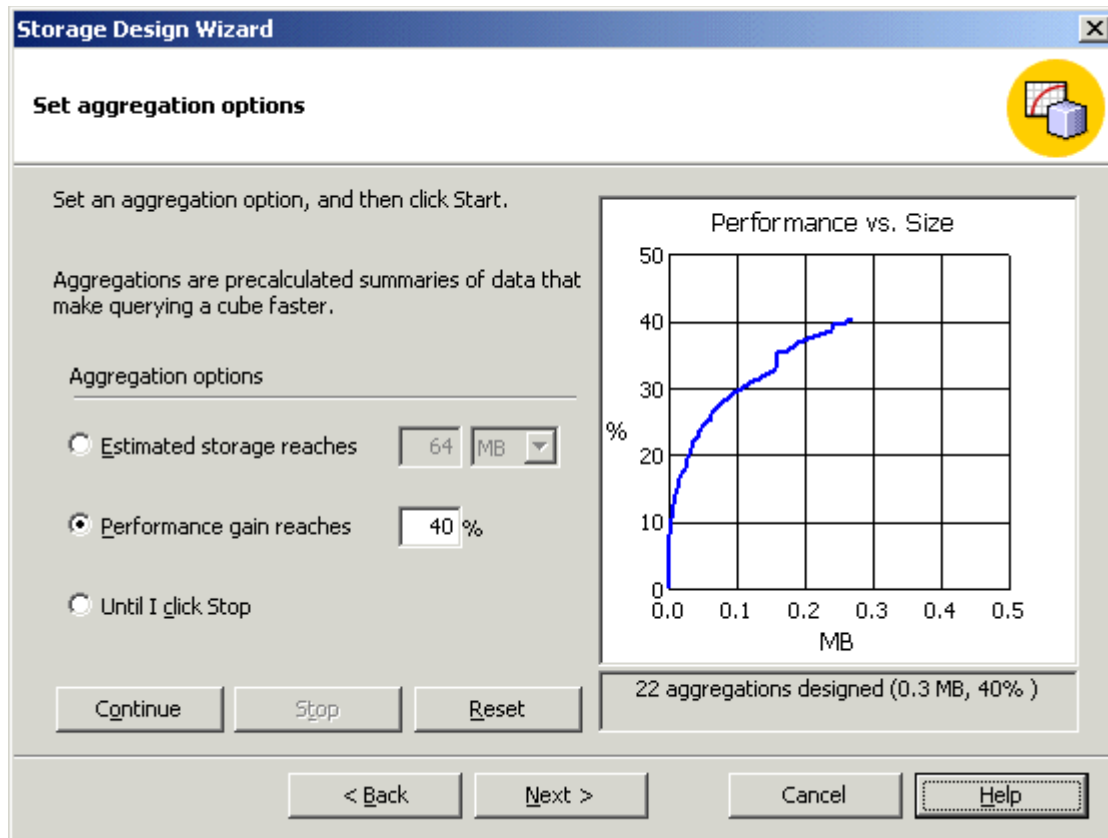


Рис. 3.10. Окно мастера проекта хранения

7. На следующем шаге выберите *Process* и затем щелкните *Finish*.
Обратите внимание: обработка агрегатов может занимать некоторое время.
8. В появившемся окне Вы можете наблюдать Ваш куб, в то время как происходит обработка. Когда обработка закончена, выдается сообщение, подтверждающая, что обработка была закончена успешно.
9. Необходимо щелкнуть *Close* для возвращения к менеджеру анализа. Теперь Вы готовы просмотреть данные в кубе.

3.4. Контрольные вопросы

1. В чем разница между измерением и мерой?
2. Каковы особенности реализации многомерных кубов данных в *Microsoft SQL Server 2000 Analysis Services*?
3. Опишите основные этапы реализации хранилищ данных в *Microsoft SQL Server 2000 Analysis Services*.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	3
1. ПРОЕКТИРОВАНИЕ, РЕАЛИЗАЦИЯ И НАПОЛНЕНИЕ БАЗЫ ДАННЫХ, ЯВЛЯЮЩЕЙСЯ ИСТОЧНИКОМ ДАННЫХ ДЛЯ ХРАНИЛИЩА	5
1.1. Основные понятия	5
1.2. Анализ предметной области	5
1.3. Концептуальное моделирование	6
1.4. Физическое моделирование	8
1.5. Создание базы данных	9
1.6. Заполнение базы данных	9
1.7. Контрольные вопросы	9
2. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ ХРАНИЛИЩА ДАННЫХ	10
2.1. Определение и типовые архитектуры хранилищ данных	10
2.2. Виды моделей хранилища данных	11
2.3. Многомерное моделирование и звездообразные схемы	16
2.4. Пример «Система мониторинга подземных вод»	18
2.5. Контрольные вопросы	20
2.6. Задание	20
3. РЕАЛИЗАЦИЯ ХРАНИЛИЩА ДАННЫХ ПОД УПРАВЛЕНИЕМ MICROSOFT SQL SERVER 2000 ANALYSIS SERVICES	21
3.1. Измерения (<i>Dimension</i>)	21
3.2. Мера (<i>Mesure</i>)	22
3.3. Реализация	26
3.4. Контрольные вопросы	35

Учебное издание

КУДИНОВ Антон Викторович

ХРАНИЛИЩА ДАННЫХ ЦИКЛ ЛАБОРАТОРНЫХ РАБОТ

Часть 1

Методические указания к циклу лабораторных работ по дисциплине
«Хранилища данных» для магистрантов, обучающихся
по магистерской программе «Компьютерный анализ
и интерпретация данных» направления 230100
«Информатика и вычислительная техника»

Научный редактор
доктор технических наук,
профессор

Н.Г. Марков

Редактор

М.В. Пересторонина

Верстка

В.П. Аршинова

Дизайн обложки

*О.Ю. Аршинова
О.А. Дмитриев*

Подписано к печати 17.11.2008. Формат 60x84/16. Бумага «Снегурочка».


Печать XEROX. Усл. печ. л. 2,09. Уч.-изд. л. 1,89.

Заказ 813. Тираж 100 экз.



Томский политехнический университет
Система менеджмента качества
Томского политехнического университета сертифицирована
NATIONAL QUALITY ASSURANCE по стандарту ISO 9001:2000



ИЗДАТЕЛЬСТВО  ТПУ. 634050, г. Томск, пр. Ленина, 30.