Tallinna Ülikool

Informaatika Instituut

Tatjana Aleksandrovitch

MS SQL Server 2008 uued võimalused

Seminaritöö

Juhendaja: Jaagup Kippar

Juhendaja…….....”…….”……2009. a.

Tallinn 2009

Оглавление

Вступление

MS SQL Server 2008, который создан на основе лучших качеств MS SQL Server 2005, гарантирует более защищённую платформу данных, предоставляя предприятиям возможность шифрования данных как в целой базе данных, файлах данных, так и в логфайлах без изменения приложений. Также MS SQL Server 2008 облегчает контроль безопасности данных, включая в себя более основательное проведение аудита данных.

Задачей данной семинарской работы является знакомство с новыми возможностями MS SQL Server 2008, такими как управление при помощи политик, новые возможности для создания отчётов и хранение любых видов данных.

Работа разделена на три части, которые вместе дают общую картину того, что новый MS SQL Server 2008 действительно имеет множество новых возможностей, функций и полезных свойств.

В первой части речь пойдёт о такой новой возможности, как управление при помощи политик. В этой части будет рассматриваться одна из важнейших частей управления при помощи политик – платформа Declarative Management Framework. Также в этой части будет продемонстрировано управление при помощи политики. Вторая часть посвящена новым возможностям при составлении отчётов. Третья часть расскажет о том, что MS SQL Server 2008 может хранить данные любых видов и типов. Для более полного раскрытия темы, в работе используются иллюстрации, схемы, а также примеры, которые показывают новые функций и возможности MS SQL Server 2008.

1.Управление при помощи политик

Одной из новых возможностей MS SQL Server 2008 является управление при помощи политик. Эта структура позволяет определять политики на множествах объектов, а потом или автоматически или вручную предотвращать изменения, основанные на вышеуказанных политиках. В SQL Server 2008 включена новая, основанная на политиках платформа управления ядром БД SQL Server — среда декларативного управления (Declarative Management Framework (DMF)). Использование DMF даёт гарантию соответствия конфигурации системы требованиям политик; а также DMF проводит мониторинг изменений в системе и их предотвращение путём разработки соответствующих политик. При этом сократилась совокупная стоимость владения за счёт упростившегося администрирования.

Главными преимуществами управления при помощи политик, являются следующие:

1) **Рост масштабов.**

Новое управление, которое создано на основе политик, легко масштабируется на несколько серверов, тем самым позволяет добиться выполнения единых политик конфигурирования по всему предприятию.

2) **Достижение желаемого результата.**

Благодаря инновационной инфраструктуре политик, администраторам даётся стройное представление о системной конфигурации, что позволяет им заранее устанавливать желаемые характеристики служб данных.

3) **Увеличение производительности за счёт мониторинга и оптимизации.**

В новом MS SQL Server 2008 работу служб данных можно отслеживать и оптимизировать.

4) **Расширенный контроль над состоянием системы.**

В состав SQL Server Management Studio входит ряд стандартных административных отчетов, содержащих информацию об экземплярах SQL Server, базах данных и других объектах.[1]

1.1.Declarative Management Framework

Среда Declarative Management Framework (*Рис.1*) представляет собой систему управления одним или несколькими экземплярами SQL Server 2008, основанную на

[1] Microsoft SQL Server; http://www.microsoft.com/sqlserver/2008/

политиках. С помощью DMF и SQL Server Management Studio можно создавать политики для управления объектами на сервере, такими как экземпляры SQL Server, базы данных и другие объекты. DMF обладает тремя компонентами: управление политиками, администраторы политик (то есть те, кто создаёт их) и явное администрирование.

Бывает также автоматическое администрирование. Администраторы политик задействуют их автоматические выполнение, используя один из перечисленных режимов:

- Enforce – для предотвращения нарушения политики, используются DDL-триггеры;
- Check on Changes – для проверки политики при внесении изменения, используется уведомление о событии;
- Check on Schedule – для периодической проверки политики, используется задание SQL Server Agent.

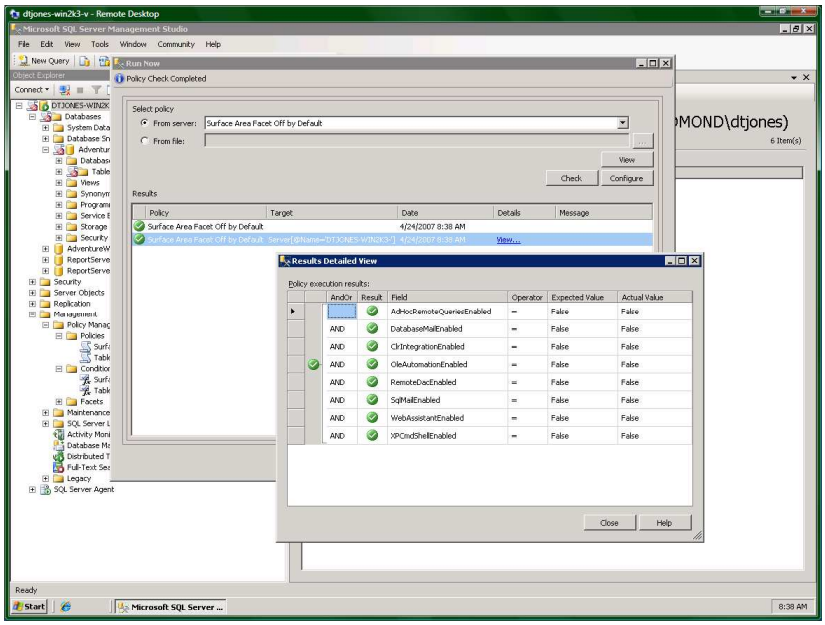

Рис.1 **Declarative Management Framework**

1.2.Пример управления при помощи политик.

Перед началом работы над примером, хочется отметить, что, к сожалению, управление при помощи политик в версиях MS SQL Server 2008 Express Edition не работает в полной мере. Показать это управление можно только вручную.

Допустим, что, являясь администратором сервера, мы не хотим, чтобы в базе данных была создана таблица с названием example, и все производные от этого названия, например,

example_1, example_table и тд. Для этого нужно создать политику, в которой будет указано, что все названия таблиц, начинающиеся на example, запрещены.

Чтобы создать свою политику для начала следует создать условие (*Рис.2*) для этой политики.

Для этого выбираем:

Управление => Управление политиками => Условия

Кликаем правой кнопкой мышки на Условия и выбираем - Создать условие…

Получаем такое окно:

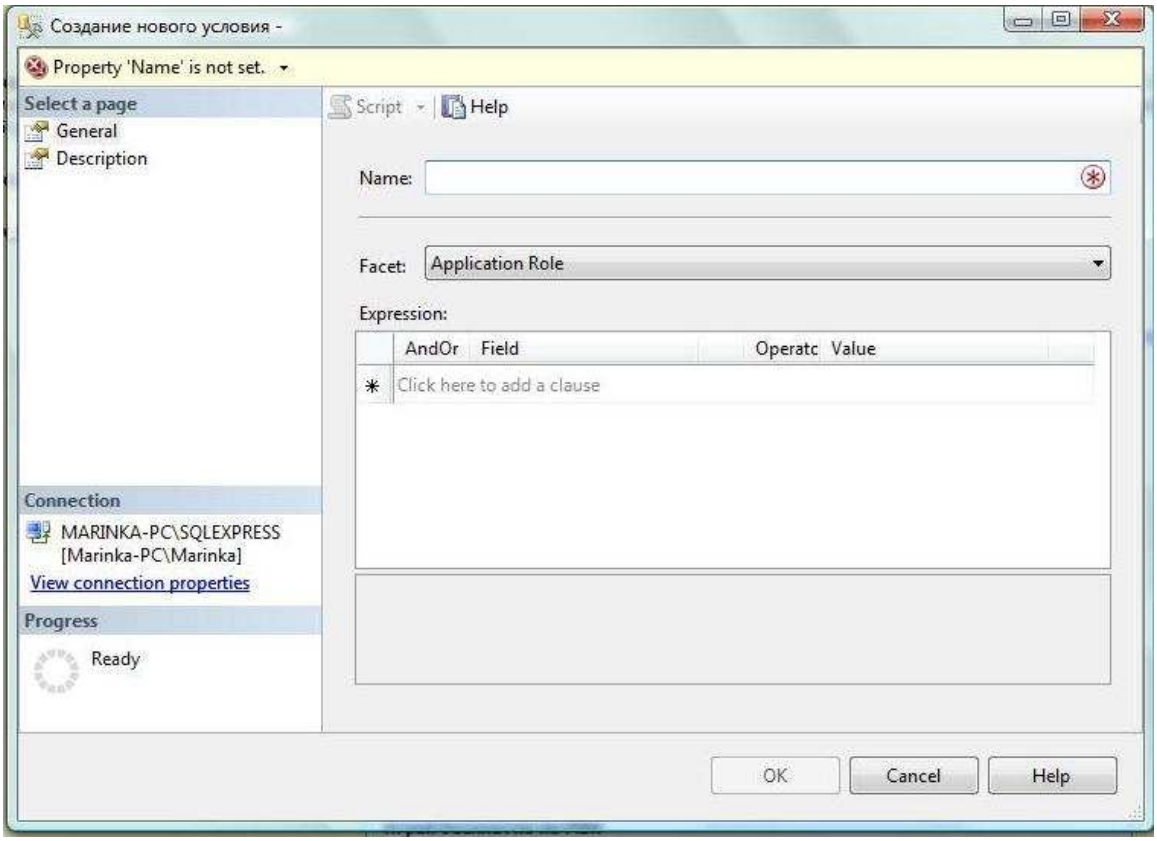

Рис.2 **Создание нового условия**

В поле Name пишем название нашего условия - Table Name 1. В Facet выбираем Multipart Name. В Expression Field выбираем @Name, Operation – Not Like, Value пишем 'example%' => OK.

Следующим шагом будет создание политики (*Рис.3*).

Управление => Управление политиками => Политики => Создание новой политики…

Называем нашу политику – Example Policy, выбираем созданное нами условие Table Name 1. В Against targets выбираем Every Table in Every Database, чтобы наша политика распространялась на каждую создаваемую таблицу в любой базе данных.

Evaluation mode выбираем On change: prevent это гарантирует то, что политика будет предотвращать таблицы с названием example => OK.

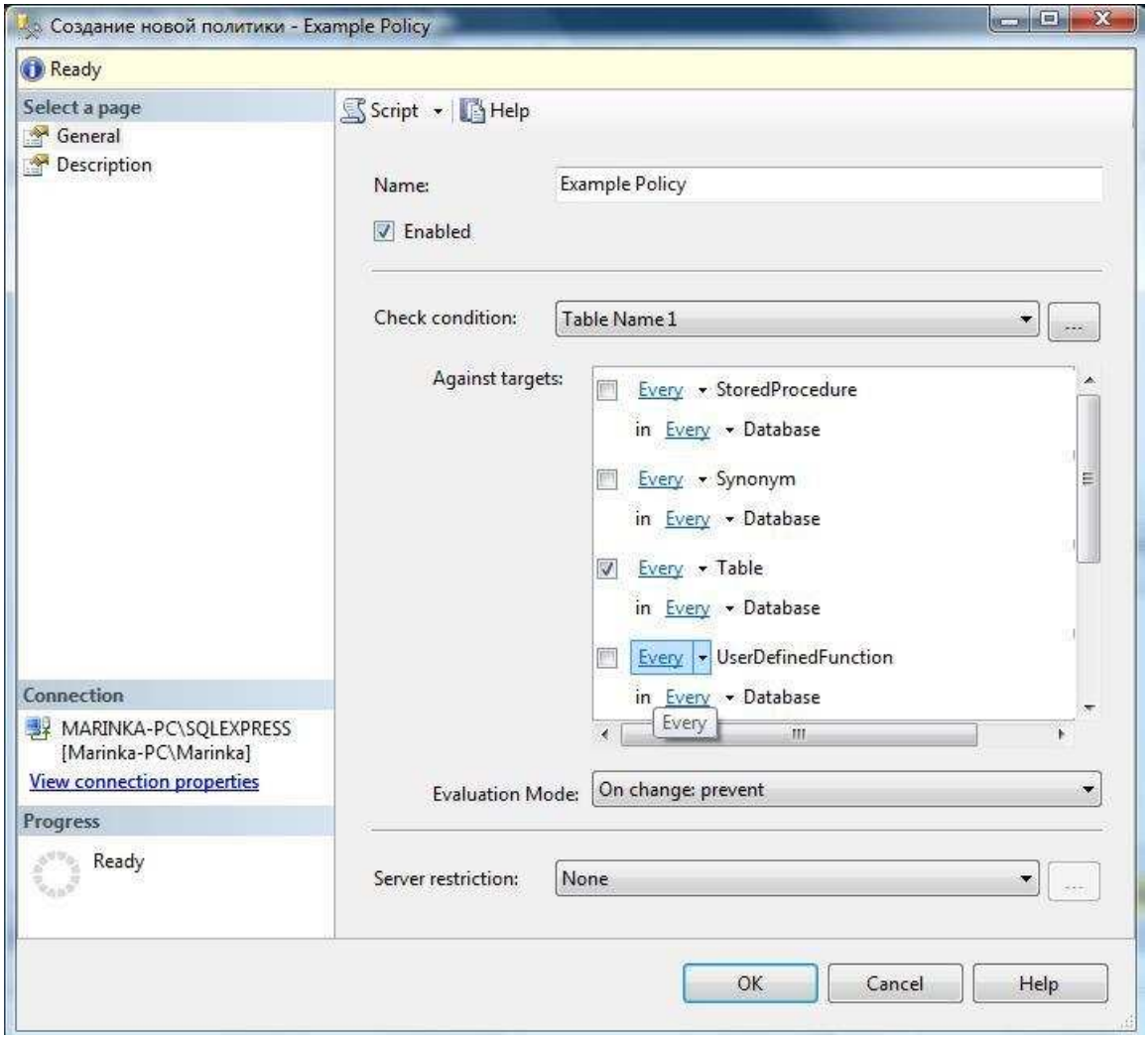

Рис. 3 **Создание новой политики**

Делаем проверку:

Используем функцию для создания таблиц CREATE TABLE. Создаём таблицы (*Рис.4*) с названием example, example_table и tbl_employee.

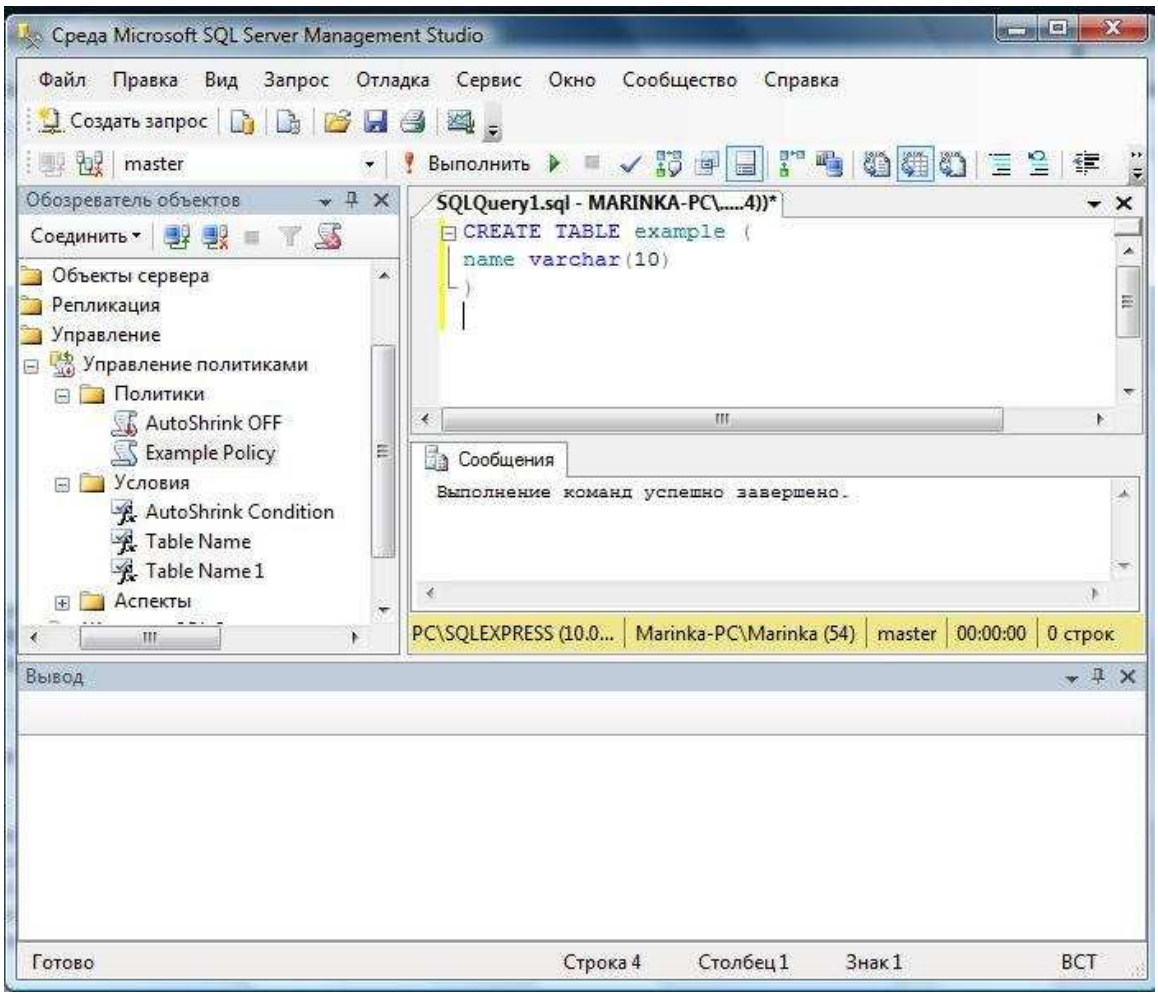

Рис. 4 **Создание таблицы**

Теперь, чтобы узнать, как работает политика, проделываем следующие шаги:

на политику Example Policy нажимаем правой кнопкой мышки – Вычислить.

Получаем, что только одна таблица соответствует условию политики, а две других не проходят, что и требовалось доказать. Это демонстрирует рисунок номер 5.

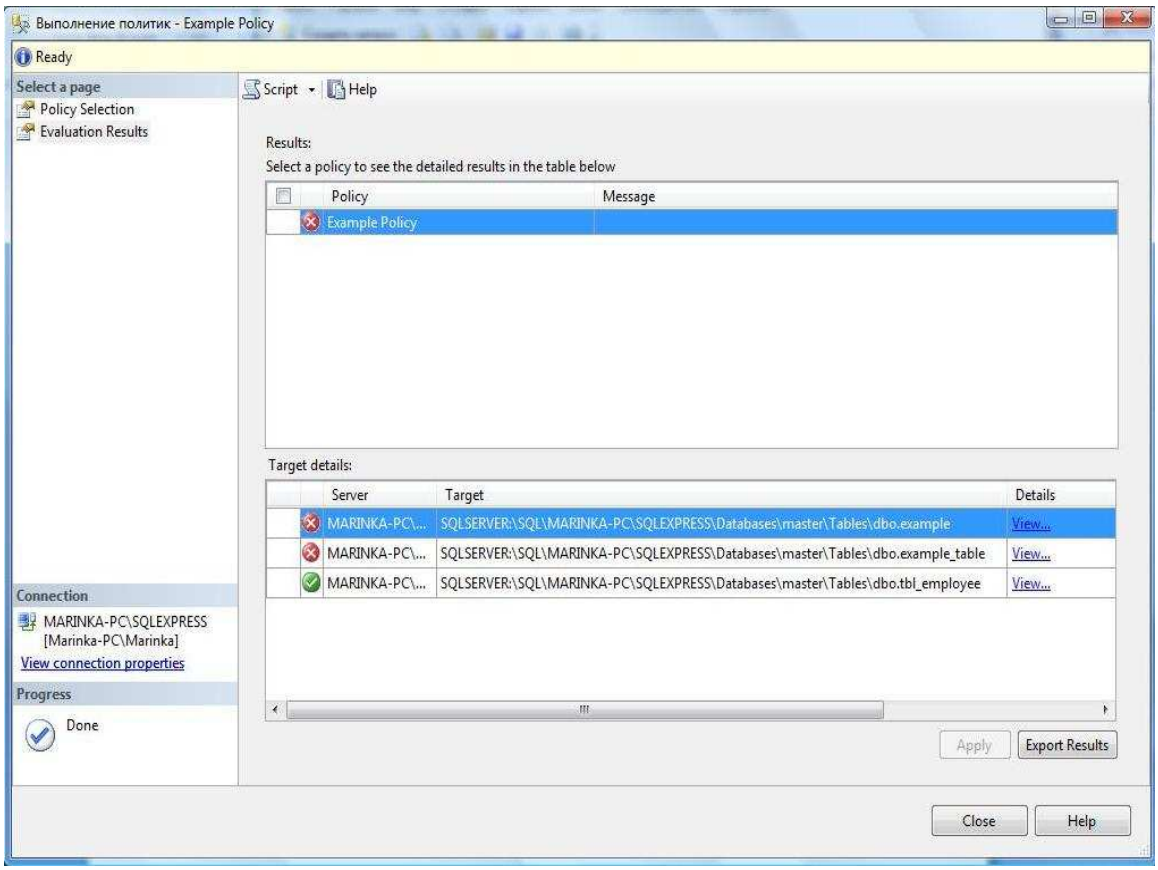

Рис. 5 **Окно выполнения политики**

В версиях MS SQL Server 2008 Enterprise и других, при создании таблицы, которая не соответствует заданной политики, сообщается об ошибки и эта таблица не создаётся. Благодаря этой новой возможности в MS SQL Server 2008 можно создавать различные правила и установки для работы сервера, так как существует достаточно большое количество аспектов для создания политик (*Рис.6*).

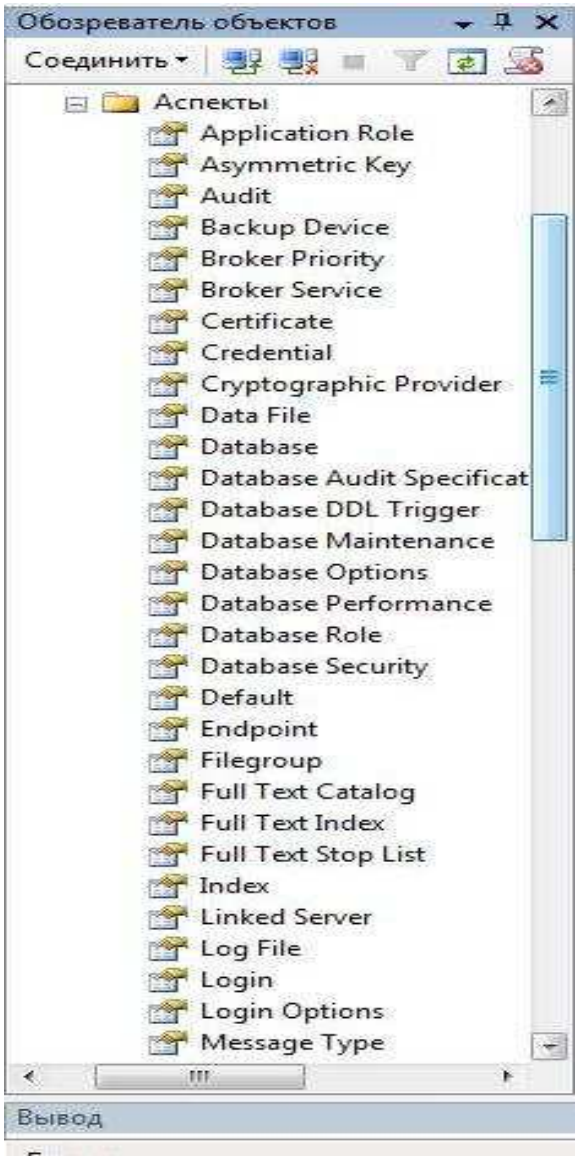

Рис. 6 **Аспекты для создания политик**

2.Расширенные возможности для создания отчётов

MS SQL Server 2008 богат средствами визуализации и уникальными гибкими возможностями проектирования, благодаря чему, при создании отчётов удовлетворяются любые потребности пользователя. Также можно отметить, что MS SQL Server 2008 теперь поддерживает формативный текст. Главными плюсами этой функции является то, что во время проектирования можно использовать смешанные форматы в текстовых блоках, а также эта функция поддерживает сценарии документов, с фокусом на тексте и стало более эффективно создание юридических, платежных документов, почтовых наклеек и тд. В этой части речь пойдёт о таких новых возможностях MS SQL Server 2008 как Tablix, средствах визуализации – диаграммы и «датчики».

2.1.Tablix

Tablix комбинирует в себе три типа визуализации данных – Table, Matrix и List – и заключает в себе свойства всех трёх названных типов. Эта функция позволяет работать как с фиксированными, так и динамическими колонками и строчками. Кроме того, эта функция даёт возможность выборочного отображения заголовков строк или колонок. Благодаря функции Tablix на каждом уровне могут быть множественные параллельные строки или колонки, и она позволяет совершать произвольную вложенность по всем осям. Область данных Tablix позволяет улучшить гибкость макета отчетов и обеспечивает более согласованный поход к его подготовке к просмотру. Она использует гибкий макет сетки, в котором группы упорядочены в иерархии строк и столбцов. Они могут быть вложенными, смежными или рекурсивными. Существующая в Tablix панель «Группирование» позволит с лёгкостью создавать группы строк и столбцов с добавлением промежуточных и общих итогов. В область данных Tablix входят строки и столбцы, содержащие данные групп и сводные данные, которые регулируются автоматически. Tablix открывает новые возможности для работы со сложными и статистическими данными, но позволяет также работать и с простыми таблицами и матрицами. Как было сказано выше, конструктор отчетов включает три типа данных: «Таблица», «Матрица» и «Список». Можно сделать вывод, что главным преимуществом соединения этих типов можно считать то, что каждый из этих типов можно использовать отдельно в качестве основы для создания отчета. Что в свою очередь позволяет разрабатывать сложные отчеты и интегрировать функции, относящиеся к разным типам данных.

2.2.«Датчики»

«Датчики» используются в том случае, когда пользователь хочет точно отобразить информацию реального времени в форме, наиболее удобной для восприятия. Область данных «Датчик» — это панель датчиков, в которой может содержаться один или несколько датчиков. Несколько «датчиков», располагаясь рядом на одной панели, могут отображать отдельные значения. “Бывают линейные и радиальные «датчики». Существует также специальный тип датчика — термометр, предназначенный для отображения температур“.[2] Благодаря новым возможностям к панели «датчиков» можно применить фильтрацию, сортировку, а также группирование. Применение «датчиков» довольно обширно, однако чаще всего «датчики» используются, когда требуется отобразить ключевые индикаторы производительности в одиночном радиальном или линейном датчике. А также, если «датчик» расположен внутри таблицы или матрицы, то его можно использовать для иллюстрации значений внутри каждой ячейки или, для сравнения данных между полями, если использовать несколько «датчиков» на одной панели.

Типы индикаторов «датчиков»:

- Круговые индикаторы
- Линейные индикаторы
- Численные индикаторы
- Индикаторы состояния

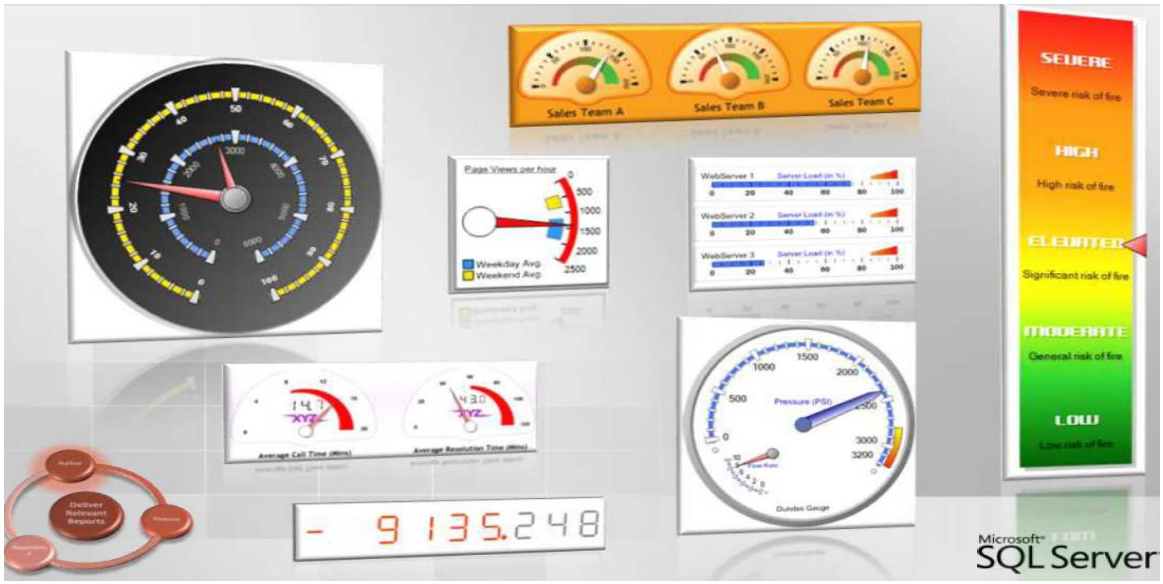

Рис. 7 **Виды «датчиков».**

[2] Microsoft TechNet. URL http://technet.microsoft.com/ru-ru/library/bb630399.aspx

2.3.Диаграммы

По сравнению с прошлыми версиями MS SQL Server, MS SQL Server 2008 имеет более широкий выбор диаграмм. Диаграммы используются, когда пользователь хочет как можно богаче представить аналитическую информацию. Основными возможностями диаграмм являются аннотации, собственные палитры и индикаторы ошибок. Также в диаграммах можно использовать множественные оси или же разделять оси, работая при этом с большими объёмами данных. Для каждой оси могут быть заданы параметры, определяющие разрывы шкалы, логарифмические шкалы, интервалы и пересекающиеся полосковые линии. Можно создавать объединенные диаграммы и альфа-смеси. При использовании области данных «Диаграмма» доступно автоматическое маркирование интервалов, благодаря чему не происходит наложения меток друг на друга. Настройка углов поворота, размер шрифта и особенности огибания областей текстом для вычисления меток осей также доступны в этой версии. Кроме того, поддерживает пересечения осей в настраиваемых точках, а также фоновые полосковые линии с регулярными или настраиваемыми интервалами. „Также в этой версии переработаны элементы управления диаграммами, которые предоставляют больший контроль над осями и формулами и снабжены улучшенным пользовательским интерфейсом, который позволяет существенно упростить создание диаграмм и работу с ними.“ [3]

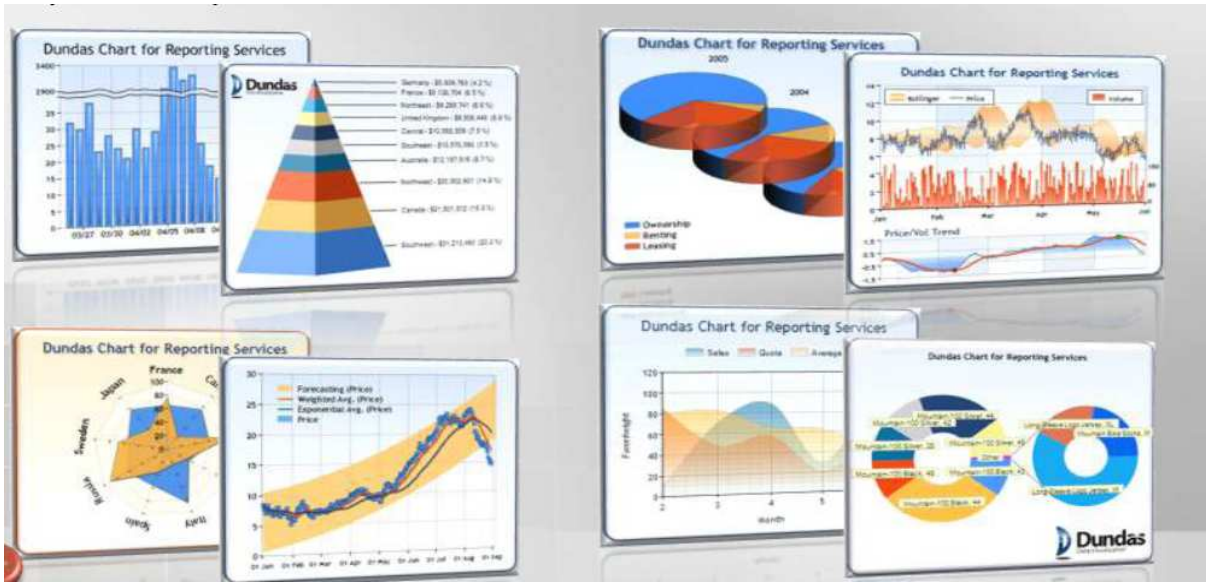

Рис. 8 **Виды диаграмм.**

На Рис.8 мы видим различные виды диаграмм. Можно сделать вывод, что при таком изобилии видов, пользователь при помощи диаграмм может изобразить всё, что

[3] SQL Server Developer Center. URL http://msdn.microsoft.com/ru-ru/library/bb630399.aspx

пожелает, то есть любые потребности будут удовлетворены. Далее представлены дополнительные виды диаграмм, которые используются в MS SQL Server 2008.

Дополнительные типы диаграмм MS SQL Server 2008
• Линии с нанесенным шагом • Круговые диаграммы • Поляры • Радары • Диаграммы Ганта • Ранжирование строк/колонок • Воронки • Пирамиды • Гистограммы • Кубы • Диаграмма «японские свечи»

Главное отличие между диаграммами и «датчиками» является то, что в диаграммах число точек ввода данных может быть неизвестно во время разработки, и результаты запросов определяют фактический номер. В случае с «датчиками», пользователь заблаговременно знает число точек ввода данных.

2.4. Создание диаграммы

Для создания tablix, диаграмм и датчиков используется приложение MS SQL Server 2008 - Report Builder 2.0. Итак, допустим, что требуется диаграмма, которая будет давать отчёт о выполнении норматива рабочими. Допустим, что в месяц рабочий должен выполнить 100 заказов. Начинаем создание диаграммы. Открываем Report Builder 2.0. И на вкладыше Insert выбираем Chart => Chart Wizard (*Рис.9*).

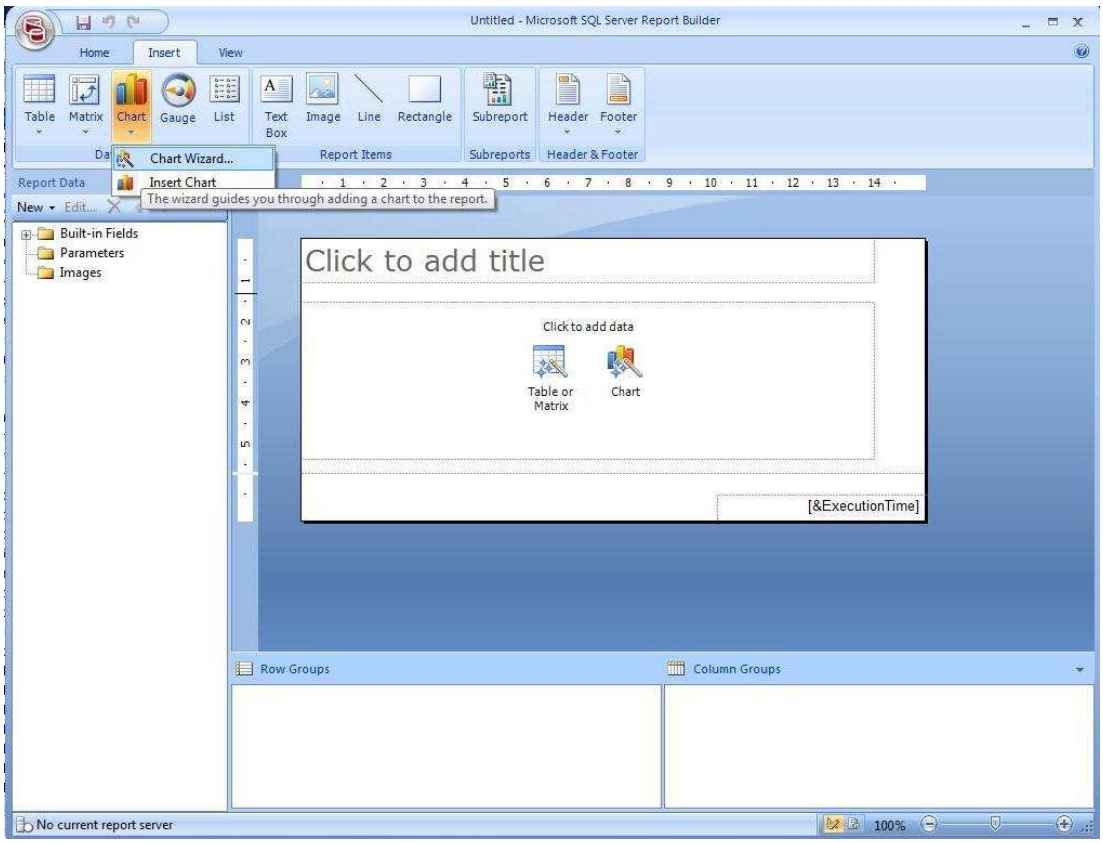

Рис. 9 **Report Builder**

Создаём новое свойство источника данных, рис. 10 иллюстрирует эти шаги.

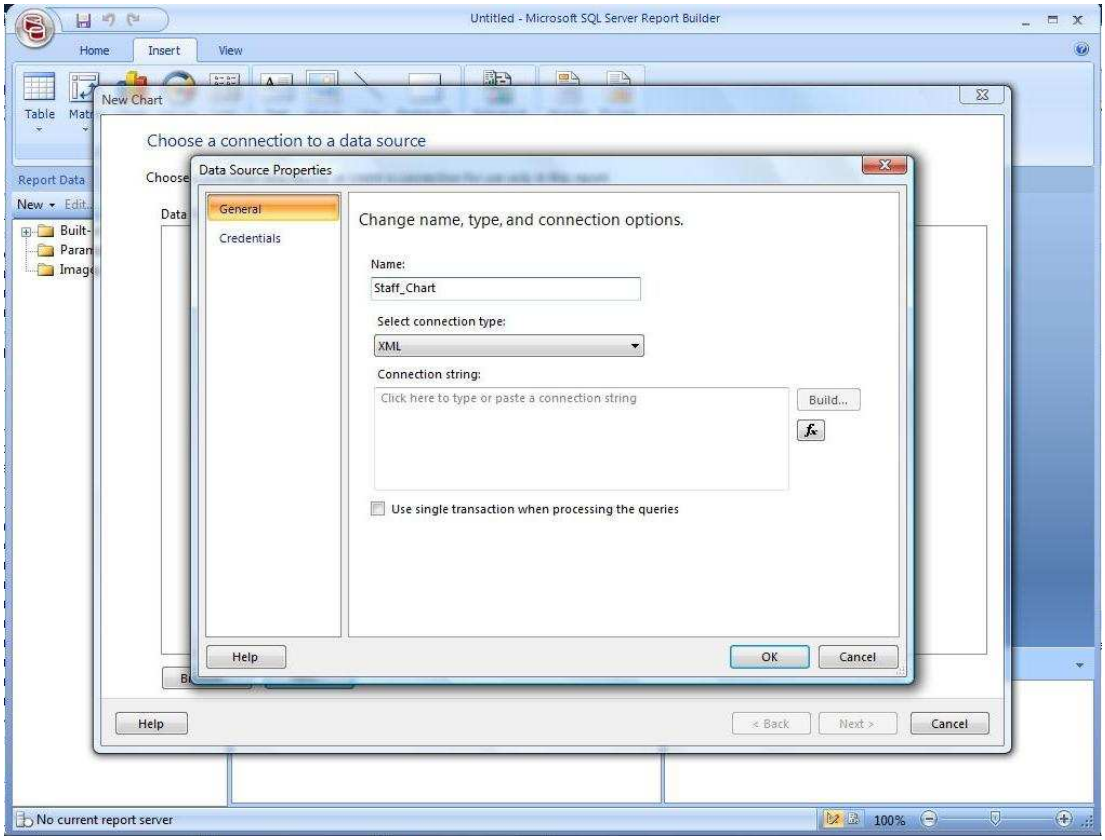

Рис. 10 **Свойства источника данных**

В Credentials выбираем Use current Windows user => OK.

На следующем этапе выбираем наш Staff_Chart и нажимаем Next. Появляется окно, в котором с помощью XML мы можем внести наши данные (*Рис.11*).

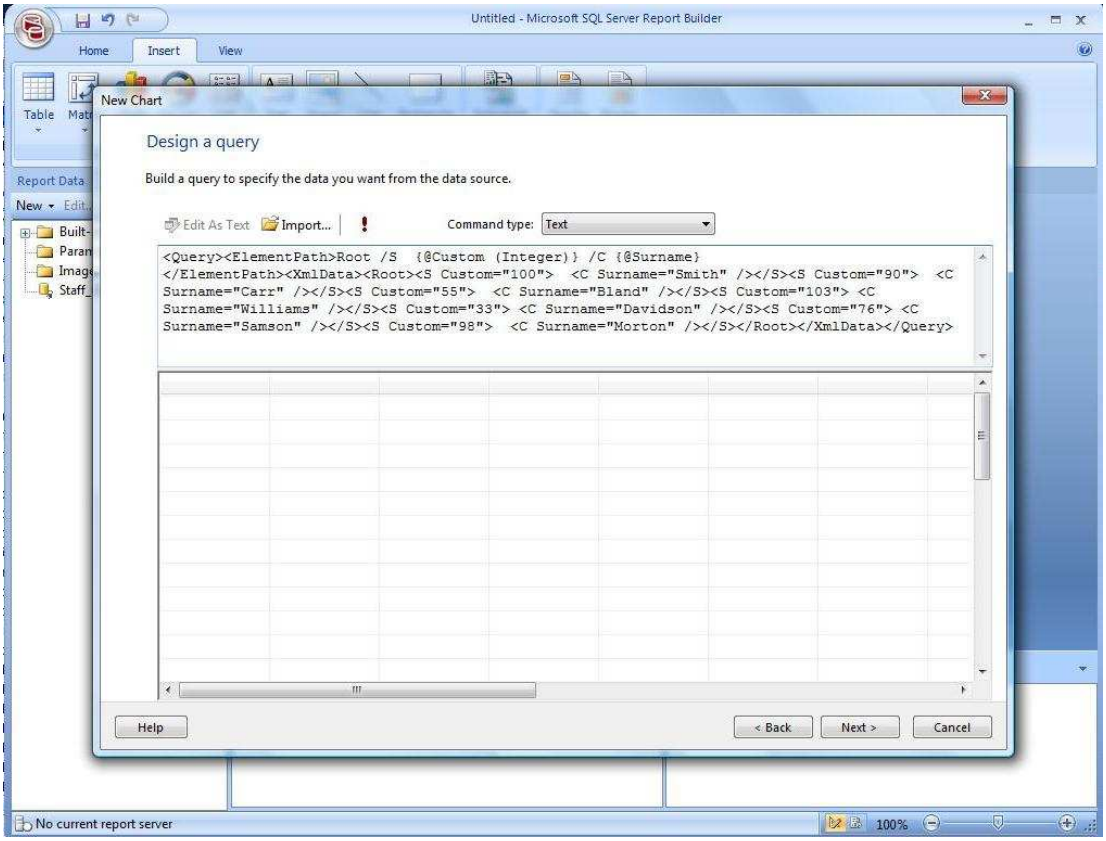

Рис. 11 **Создание данных для диаграммы**

Далее выбираем тип диаграммы, в нашем случае это Column. В следующем окне мы видим доступные нам аспекты, кликаем на каждый из них два раза. При этом Custom перемещается в Values , а Surname в Categories (*Рис.12*).

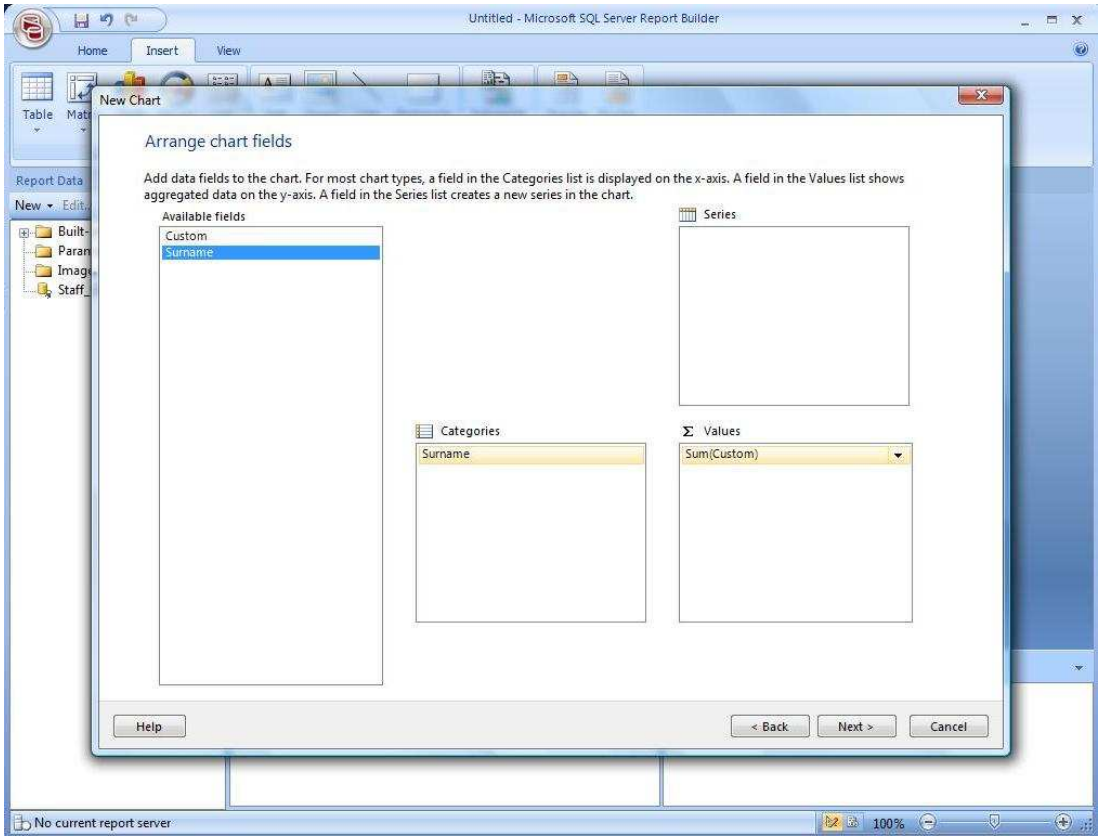

Рис. 12 **Распределение по полям**

Выбираем стиль и нажимаем Finish. Получаем такое окно – Рис. 13

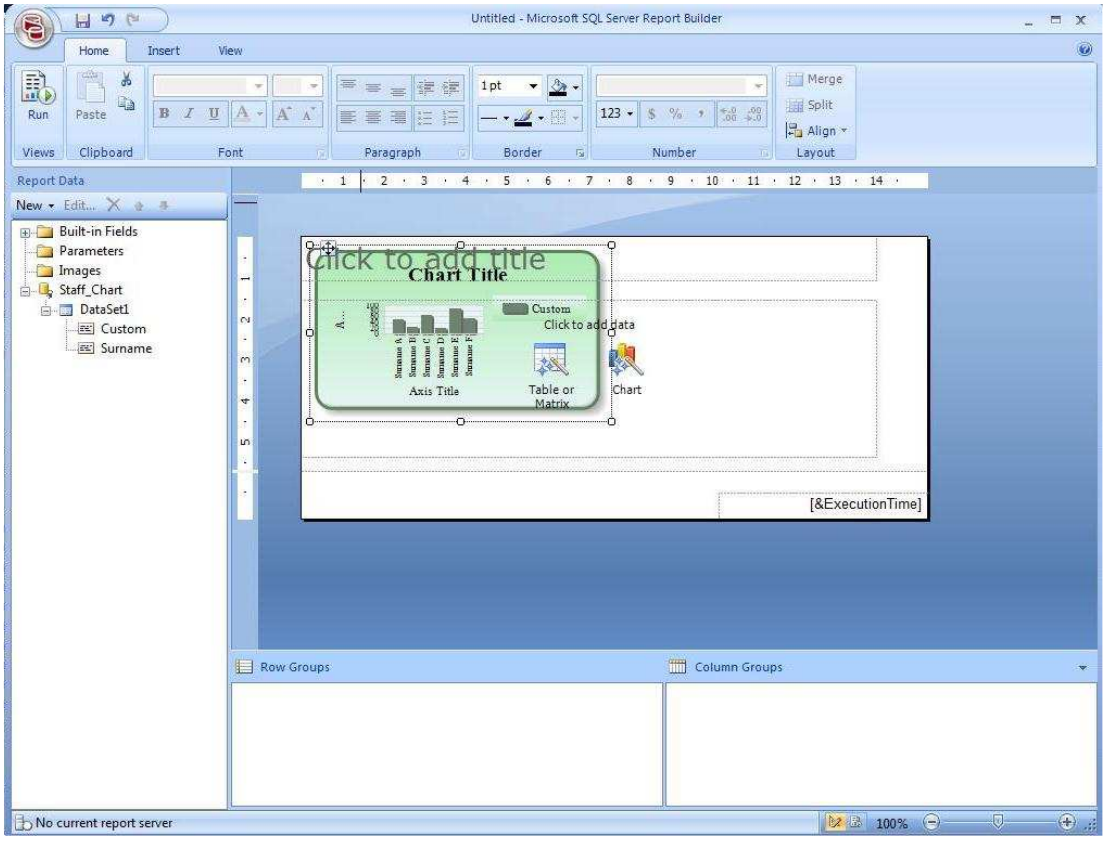

Рис. 13 **Вид диаграммы в окне конструктора**

Итак, в области конструктора появилась наша диаграмма, но это, как мы видим, стандартное представление. Для того, чтобы увидеть наши данные, нажимаем Run и получаем (*Рис.14*):

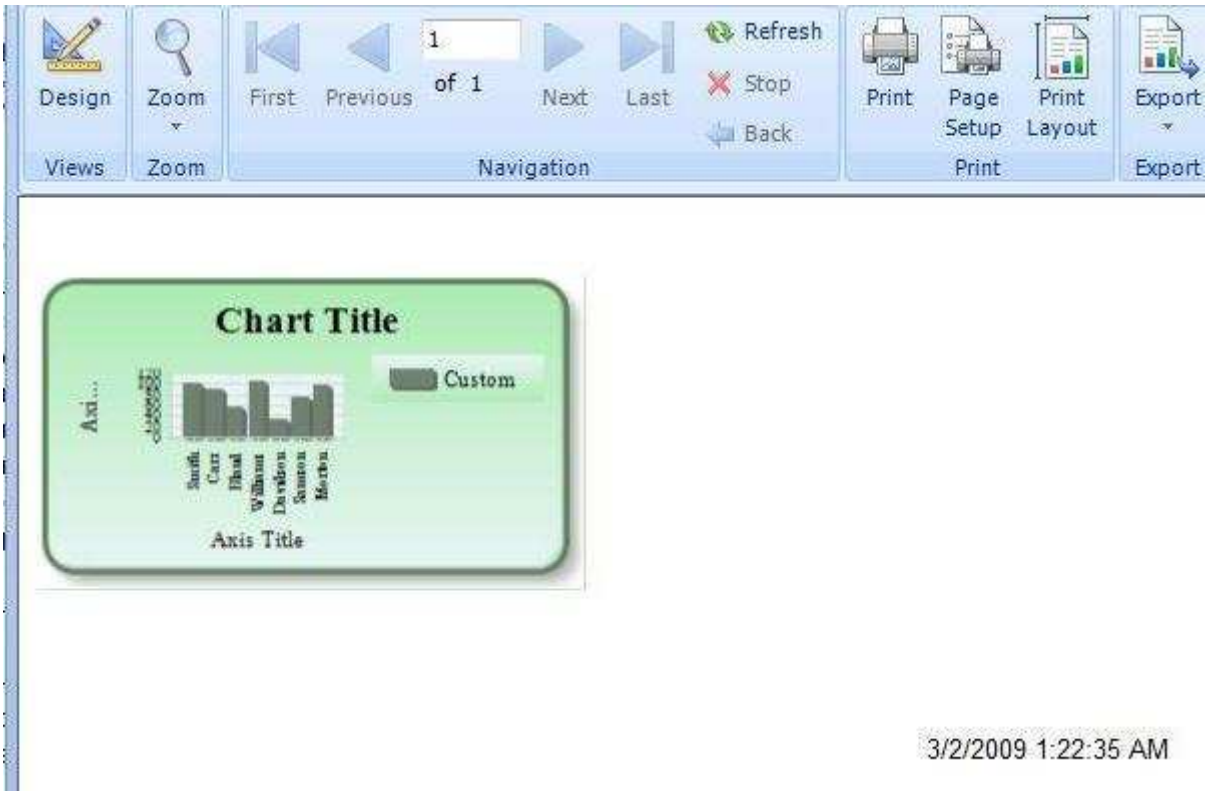

Рис. 14 **Диаграмма с заданными данными**

Как мы видим, данные не очень хорошо видны, поэтому возвращаемся в Design, чтобы откорректировать нашу диаграмму. Для начала увеличим её. Мы можем сделать диаграмму более объёмной. Для этого правой копкой мышки кликаем на диаграмму и выбираем 3D Effects (*Рис.15*).

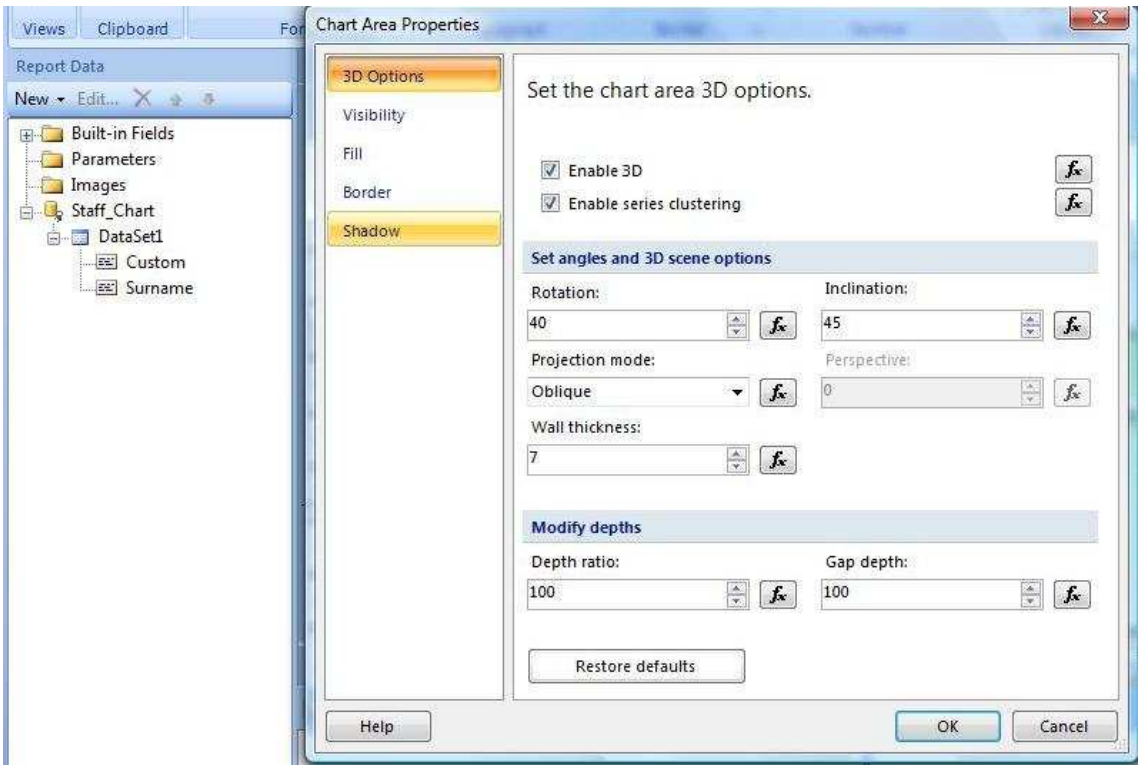

Рис. 15 **Свойства диаграммы**

Выбираем Enable 3D и Enable series clustering, а также меняем угол просмотра (по умолчанию стоит - 30, 30, ставим свои значения – 40, 45). Также назовём диаграмму – Staff. Нажимаем Run и видим, что теперь (*Рис.16*), наша диаграмма отличается от изначальной (*Рис.14*).

Рис. 16 **Диаграмма Staff**

Также можно изменять тип диаграммы, например, сделаем её с помощью Line (*Рис.17*) и Scatter (*Рис.18*).

Рис. 17 **Тип Line**

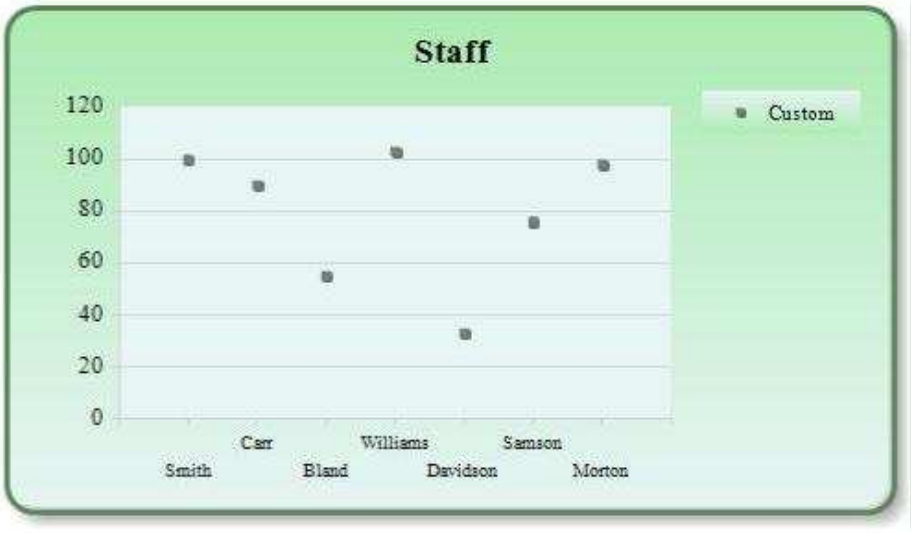

Рис. 18 **Тип Scatter**

3.Хранение данных любых типов

В SQL Server 2008 обеспечивается взаимодействие между реляционными и нереляционными данными, что позволяет пользователям обращаться к документам как к данным, кодировать сложные иерархии в XML и выполнять запросы как к реляционным, так и к текстовым данным.

3.1.Иерархия

Приложения для БД SQL Server 2008 способны моделировать древовидные структуры более эффективно, чем это возможно сейчас. Новый системный тип Hierarchyid предназначен для хранения значений, представляющих узлы иерархии.

Для примера сделаем таблицу иерархии персонала в некой фирме. Сначала создаётся обычная таблица, которая уже хранит в себе иерархию.

```
CREATE TABLE Staff(
      hid hierarchyid NOT NULL,
      userId int NOT NULL,
      userName nvarchar(50) NOT NULL,
CONSTRAINT PK_Table_1 PRIMARY KEY CLUSTERED
(
  [hid] ASC
))
```

Главными переменными в таблице являются: тип hierarchyid, id рабочего и его фамилия. В этой таблице мы хотим показать следующую иерархию:

1 Smith
    2 Carr
        7 Williams
            8 Davidson
    3 Ford
    4 Bland
        5 Samson
        6 Morton

Сначала требуется создать корень иерархии.

```
insert into Staff
values(hierarchyid::GetRoot(), 1, 'Smith')
```

`GetRoot()`является статистическим методом, который всегда возвращает идентификатор корня иерархии.

После создания корня, добавляем потомков.

```
declare @Id hierarchyid
select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = hierarchyid::GetRoot()

insert into Staff
values(hierarchyid::GetRoot().GetDescendant(@id, null), 2, 'Carr');

select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = hierarchyid::GetRoot()

insert into Staff
values(hierarchyid::GetRoot().GetDescendant(@id, null), 3, 'Ford');

select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = hierarchyid::GetRoot()

insert into Staff
values(hierarchyid::GetRoot().GetDescendant(@id, null), 4, 'Bland');
```

`hid.GetAncestor(1) = hierarchyid::GetRoot()` - выбирает все записи, предком (прямым) которых является корень;

`hierarchyid::GetRoot().GetDescendant(@id, null)` - выбирает первый свободный hierarchyid прямых потомков корня дерева.

Заполняем дерево оставшимся персоналом.

```
declare @phId hierarchyid
select @phId = (SELECT hid FROM Staff WHERE userId = 2);
```

```
select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = @phId

insert into Staff
values(@phId.GetDescendant(@id, null), 7, 'Williams');

select @phId = (SELECT hid FROM Staff WHERE userId = 4);

select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = @phId

insert into Staff
values(@phId.GetDescendant(@id, null), 5, 'Samson');

select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = @phId

insert into Staff
values(@phId.GetDescendant(@id, null), 6, 'Morton');

select @phId = (SELECT hid FROM Staff WHERE userId = 7);

select @Id = MAX(hid)
from Staff
where hid.GetAncestor(1) = @phId

insert into Staff
values(@phId.GetDescendant(@id, null), 8, 'Davidson');
```

Пишем команду:

```
select hid.ToString(), hid.GetLevel(), * from Staff
```

И, получаем:

```
/              0   0x                  1  Smith
```

```
/1/             1   0x58                2  Carr

/1/1/           2   0x5AC0              7  Williams

/1/1/1/         3   0x5AD6              8  Davidson

/2/             1   0x68                3  Ford

/3/             1   0x78                4  Bland

/3/1/           2   0x7AC0              5  Samson

/3/2/           2   0x7B40              6  Morton
```

Как мы видим, данные в точности схожи со структурой иерархии. При помощи типа hierarchyid и вспомогательных функций, программисту баз данных легко создавать иерархические структуры, используя минимум кода.

А теперь кратко о предназначениях вспомогательных функций, использованных в создании иерархии персонала, приведённом выше:

- GetAncestor - выдает hierarchyid предка, при его помощи можно указать уровень предка, например 1 выберет непосредственного предка;
- GetDescendant - выдает hierarchyid потомка, принимает два параметра, с помощью которых можно управлять тем, какого именно потомка необходимо получить на выходе;

- GetLevel - выдает уровень hierarchyid;
- GetRoot - выдает уровень корня;
- Parse - конвертирует строковое представление hierarchyid в собственно hierarchyid;
- Reparent - позволяет изменить текущего предка;
- ToString - конвертирует hierarchyid в строковое представление.

Для сравнения можно сказать о том, что в предыдущих версиях MS SQL Server, чтобы выполнить похожую задачу, требовалось использовать рекурсивные запросы. Минусом этого способа является то, что при рекурсии используется большой объём памяти, так как при каждом новом запросе прорабатывается полный объём данных. В MS SQL Server 2008 создание иерархических структур проходит проще и с меньшей затратой памяти.

3.2.Другие возможности

Дата и время.

В SQL Server 2008 включены новые типы данных для обозначения даты и времени:

- DATE – только дата;
- TIME – только время;
- DATETIMEOFFSET –дата и время с учетом часового пояса;
- DATETIME2 – тип для даты и времени с поддержкой большего диапазона долей секунд и лет, чем в существующем типе DATETIME.

Эти типы позволяют различать дату и время в приложениях, а также обеспечивают использование больших диапазонов или большей точности для временных показателей.

Усовершенствованный поиск.

В новой версии SQL Server 2008, благодаря встроенному полнотекстовому поиску, можно работать как текстовыми, так и с реляционными данными. Преимуществом является также то, что пользователь посредством текстовых индексов может эффективно выполнять поиск в больших текстовых полях. Эта функция доступна только в SQL Server 2008 Express with Advanced Services, если брать в учёт бесплатные версии.

Разреженные поля.

Благодаря этому компоненту значения NULL («пустые» значения) больше не занимают физическое пространство, что делает управление пустыми данными в высшей степени эффективным. В частности, разреженные поля позволяют создавать объектные модели с большим количеством значений NULL, при этом они не будут занимать много места на диске. В SQL Server 2008 также устранено 8-килобайтное ограничение для пользовательских типов, что значительно расширяет возможности пользователей.

Заключение

Задача, поставленная в начале работы, выполнена. В работе показаны новые возможности MS SQL Server 2008. Каждая часть рассказывает о том или ином нововведении. В течение написания этой работы, были изучены различные публикации, чтобы создать более полную картину о предлагаемой теме. Также было проведено ряд тестовых заданий для иллюстрации примеров работы с сервером.

Конечно, в этой работе показаны не все новые возможности MS SQL Server 2008, поэтому эта тема может быть исследована далее.

В заключении хочется отметить, что MS SQL Server 2008 это эффективная, надежная и интеллектуальная платформа. Она позволяет хранить в базах данных информацию, полученную из структурированных, полуструктурированных и неструктурированных источников. Благодаря SQL Server 2008 вы можете составлять запросы, выполнять поиск, проводить синхронизацию, делать отчеты, анализировать данные. Управление информацией благодаря инновационной инфраструктуре на основе политик стало намного проще. Все данные хранятся на основных серверах, входящих в состав центра обработки данных. К ним осуществляется доступ с настольных компьютеров и мобильных устройств. Таким образом, вы полностью контролируете данные независимо от того, где вы их сохранили.

Ülevaade

SQL Server 2005 parimate omaduste vundamendile rajatud SQL Server 2008 tagab turvalisema andmeplatvormi, võimadades ettevõtetel väärtuslikke andmeid krüptida nii terves andmebaasis, andmefailides kui ka logifailides ilma rakendust muutmata. Samuti hõlbustab SQL Server 2008 andmekaitse nõuete järgmise kontrolli, sisaldades põhjalikumat andmete auditeerimist.

Antud seminaritöö on koostatud eesmärgiga tutvustada MS SQL Serveri 2008 uusi võimalusi. Sealhulgas poliitikapõhine haldusraamistik, aruannete loomine ning erinevate andmetüüpi haldamine.

SQL Server 2008-s on kasutusele võetud uus poliitikapõhine haldusraamistik, mis vahetab ettevõtes skriptipõhise halduse reeglipõhise vastu. Andmebaasioperatsioonidele määratletakse ühine poliitikate kogum ning poliitikaid jõustatakse ja jälgitakse automaatselt. Samade reeglite järgi saab hõlpsalt hallata kogu ettevõte servereid – järjepidev lähenemine on turvaline ja hoiab aega kokku.

Uues MS SQL Serveris 2008 kõik andmetüübid on lubatud. SQL Server 2008 hoiab arendajate ja administraatorite aega kokku, sest talletada ja tarvitada on võimalik mitte ainult relatsioonilist tüüpi infot, vaid kõikvõimalikke andmeid XML-ist dokumentideni. Aruannete loomine muudab kergemaks ning kasutajal on palju võimalusi. Ilmusid visualisatsiooni võimalused, nagu Tablix, Chart Data Region ja Gauge.

Selle töö eesmärk on saavutanud. Töös on esitatud põhilised MS SQL Server 2008 uued võimalused. Kirjutamise käigul uurinti palju artikleid ja muud allikaid ning tehti oma rakenduseid.

Список литературы

1. DeBetta, P. (2007) Introducing MS SQL Server 2008. URL http://groups.google.com/group/freecomputerbooks/web/introducing-microsoft-sql-server-2008-2008-peter-debetta 3.01.2009
2. Дамлер, М. (2007). MS SQL Server 2008. Общие сведения о продукте. 3.01.2009
3. Microsoft TechNet. URL http://technet.microsoft.com/ru-ru/library/bb418491.aspx 29.10.2008
4. Microsoft SQL Server 2008. URL http://www.microsoft.com/sqlserver/2008/ru/ru/default.aspx 29.10.2008
5. SQL Server Developer Center. URL http://msdn.microsoft.com/ru-ru/library/bb630399.aspx 3.01.2009