АДМИНИСТРИРОВАНИЕ MICROSOFT SQL SERVER 2008

СУБД SQL Server 2008 выполняет функции надежной платформы данных, допускает динамическую разработку, обеспечивает всеобъемлющую бизнес-аналитику и выходит за пределы реляционных данных, становясь благодаря этому прочным фундаментом, на котором небольшие, средние й крупные организаций могут строить *ИТ*-инфраструктуру следующего поколения. На рис. 1 приведены данные о сравнительной популярности этой *СУБД*.

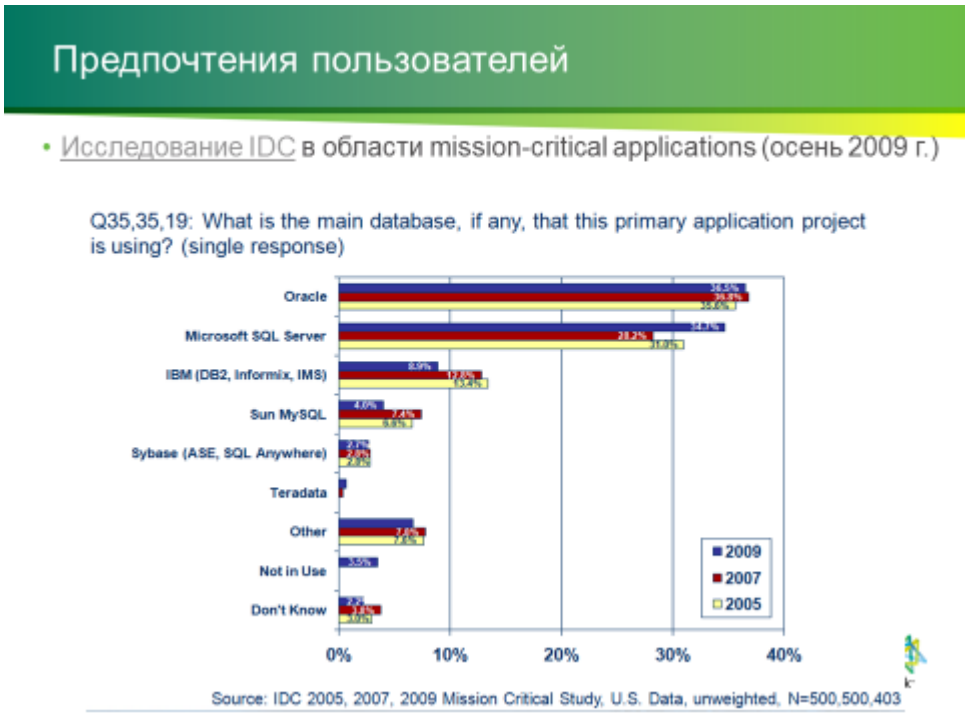

Рис. 1. Данные о популярности *SQL Server*

Важно понимать, что *СУБД SQL Server* не является единым монолитным приложением, а структурирован как ряд компонентов. Ниже перечислены основные компоненты *SQL Server 2008*.

Ядро SQL Server 2008 сформировано следующими службами:

- *Database Engine Services*. Основные компоненты для *БД*, уведомлений, репликации и полнотекстового поиска. В основе *SQL Server* находится ядро *СУБД - Database Engine*. Репликация повышает доступность данных, распределяя их по нескольким *БД* и позволяя разделить нагрузку между несколь-

кими выбранными серверами *БД*. Полнотекстовый поиск обеспечивает выполнение запросов к данным из таблиц *SQL Server* с учетом правил конкретного языка.

- *Analysis Services*. Обеспечивает функциональность *OLAP* (*Online Analytical Processing*) и интеллектуального анализа данных для приложений бизнес-аналитики. Службы *Analysis Services* позволяют собирать данные из нескольких источников, например, реляционных *БД*, и обрабатывать эти данные множеством разных способов.
- *Integration Services*. Решение масштаба предприятия для извлечения данных из нескольких источников, их преобразования, объединения и перемещения в один или несколько целевых источников данных. Позволяет объединять данные из разнородных источников, загружать данные в хранилища, киоски и пр.

Reporting Services. Серверная платформа для создания отчетов, управлеиия ими и их распространения, включающая *Диспетчер отчетов* (*Report Manager*) и *Сервер отчетов* (*Report Server*). Сервер отчетов построен на стандартной технологии *IIS* (*Internet Information Services*) и *.NET*, позволяя при обработке и размещении отчетов одновременно пользоваться преимуществами *SQL Server* и *IIS*.

- *Service Broker*. Часть *БД*, обеспечивающая организацию очередей и обмена сообщениями. Очереди используются для упорядочения задач, например, запросов, чтобы они выполнялись по мере высвобождения ресурсов. Обмен сообщениями обеспечивает передачу информации от одного приложения *БД* другому. Компонент *Database Engine* использует *Service Broker* для доставки уведомлений.
- *Sync Framework*. Обеспечивает синхронизацию данных для совместной и автономной работы. Разработчики могут применять *Sync Framework* для синхронизации *БД*, хранилищ других видов, а также файлов, папок и метаданных.

На рис. 2 и рис. 3 приведена иллюстрация основных элементов ядра *СУБД SQL Server 2008* и Ядро *SQL Server 2008 R2*.

Рис. 2. Ядро *SQL Server 2008*

ETL (от англ. Extract, Transform, Load - дословно извлечение, преобразование, загрузка) - один из основных процессов в управлении хранилищами данных, который включает в себя:

- извлечение данных из внешних источников;
- их трансформация и очистка, чтобы они соответствовали потребностям бизнес-модели;
- и загрузка их в хранилище данных.

С точки зрения процесса *ETL*, архитектуру хранилища данных можно представить в виде трёх компонентов:

- источник данных: содержит структурированные данные в виде таблиц, совокупности таблиц или просто файла (данные в котором разделены символами-разделителями);
- промежуточная область: содержит вспомогательные таблицы, создаваемые временно и исключительно для организации процесса выгрузки.
- получатель данных: хранилище данных или база данных, в которую должны быть помещены извлечённые данные.

Перемещение данных от источника к получателю называют потоком данных. Требования к организации потока данных описываются аналитиком. *ETL* следует рассматривать не только как процесс переноса данных из одного приложения в другое, но и как инструмент подготовки данных к анализу.

Рис. 3. Ядро *SQL Server 2008 R2*

Версии Microsoft SQL Server 2008

SQL Server 2008 распространяется в четырех основных версиях: *Workgroup, Standard, Enterprise* и *Developer*. Во всех этих версиях для установки предлагаются *компоненты экземпляра* и *общие компоненты*. К компонентам экземпляра (*instance feature*) относятся полная версия *SQL Server* и службы поддержки. К общим компонентам (*shared feature*) относятся клиентские инструменты, инструменты разработки и документация, необходимая для работы с *SQL Server* на рабочей станции, а также для работы с *Microsoft Sync Framework* и *Integration Services*.

Версия *Workgroup Edition* представляет собой решение начального уровня для работы с *БД*. Она идеально подходит для небольших отделов круп-

ных организаций, а также небольших компаний, которым необходима надежная система управления *БД*, но не нужны расширенные функции бизнес-аналитики из версий *Standard Edition* и *Enterprise Edition*. Основные функциональные возможности этой версии следующие.

• Работает с различными версиями *ОС Windows*, включая *Windows XP Professional, Windows Vista, Windows Server 2003* и *Windows Server 2008*.

• Поддерживает работу с *БД* любого размера, ограниченного только объемом оперативной памяти компьютера, использование двух процессоров, ограниченную публикацию репликации и полнотекстовый поиск.

• Допускает использование доставки журналов, что позволяет *SQL Server* пересылать журналы транзакций с одного сервера на другой. Используйте этот компонент для создания резервного сервера.

Наиболее популярной версией является *Standard Edition*. Она предназначена для организаций умеренного размера. Основные функциональные возможности этой версии следующие.

• Работает с различными версиями *ОС Windows*, включая, *Windows XP Professional, Windows Vista,Windows Server 2003* и *Windows Server 2008*.

• Поддерживает работу с *БД* любого размера; ограниченного только объемом оперативной памяти компьютера, использование четырех процессоров, полномасштабную публикацию репликации и полнотекстовый поиск.

• Поддерживает базовые функции *Analysis Services, Reporting Services* и *Integration Services*.

• Включает поддержку зеркального отображения *БД* (однопотокового), сжатия потока журналов и кластеров с двумя узлами.

• Ограниченно поддерживает функции шифрования и аудита.

Хотя версия *Standard Edition* уже является достаточно мощным решением для работы с *БД*, в крупных организациях имеет смысл использовать *Enterprise Edition*. Основные функциональные возможности ее следующие.

• Допускает неограниченное масштабирование и разбиение на разделы, что обеспечивает исключительную производительность и возможность использования *SQL Server* в *БД* любого объема. Разделение таблиц на разделы по горизонтали и их распределение по нескольким серверам позволяет совместно использовать группу серверов для поддержки большого веб-сайта или обработки корпоративных данных.

• Поддерживает расширенные возможности зеркального отображения данных, обеспечивающие полную параллельную обработку, и усовершенствованные инструменты для интеллектуального анализа данных и полномасштабного *OLAP*.

• Поддерживает работу кластера с 16 узлами, использование нескольких экземпляров, снимки *БД*, индексированные представления. Оперативное восстановление страниц и файлов, сжатие архивов, «горячую» замену памяти и процессоров.

• Располагает расширенными функциями шифрования и аудита, включая прозрачное шифрование данных (*Transparent Data Encryption, TDE*) и трассировку *C2*.

• Поддерживает, многосерверное администрирование, неограниченную виртуализацию, автоматическое управление группами серверов и зеркальное отображение архивных носителей.

SQL Server 2008 Enterprise Edition также работает в различных версиях Microsoft Windows. Версия *Developer Edition* включает все компоненты, *Enterprise Edition*, но лицензируется только для разработки и тестирования.

Имеются также и другие версии *SQL Server 2008 - Web Edition, Compact Edition, Express Edition* (включает распространяемое ядро *Database Engine*) и *Express Edition with Advanced Services*.

Версия *Compact Edition* позволяет применять *SQL Server* в качестве хранилища данных на мобильных устройствах, настольных компьютерах и веб-клиентах.

Версию *Express Edition* следует использовать, если требуется простое решение *БД* для настольного компьютера и небольших серверных приложений. Эта версия бесплатна и может, распространиться с приложениями независимых разработчиков. Обе версии поддерживают *БД* размером до 4 Гб, использование *ОЗУ* до 1 Гб и один процессор.

Если не считать *Express Edition* и *Compact Edition*, большая часть различий между версиями *SQL Server* скрыта от глаз пользователя и не отражается на интерфейсе. Как и следует ожидать, у версий *Express Edition* и *Compact Edition* управляющий интерфейс очень прост.

На рис. 4 приведена иллюстрация версий *SQL Server 2008 R2*.

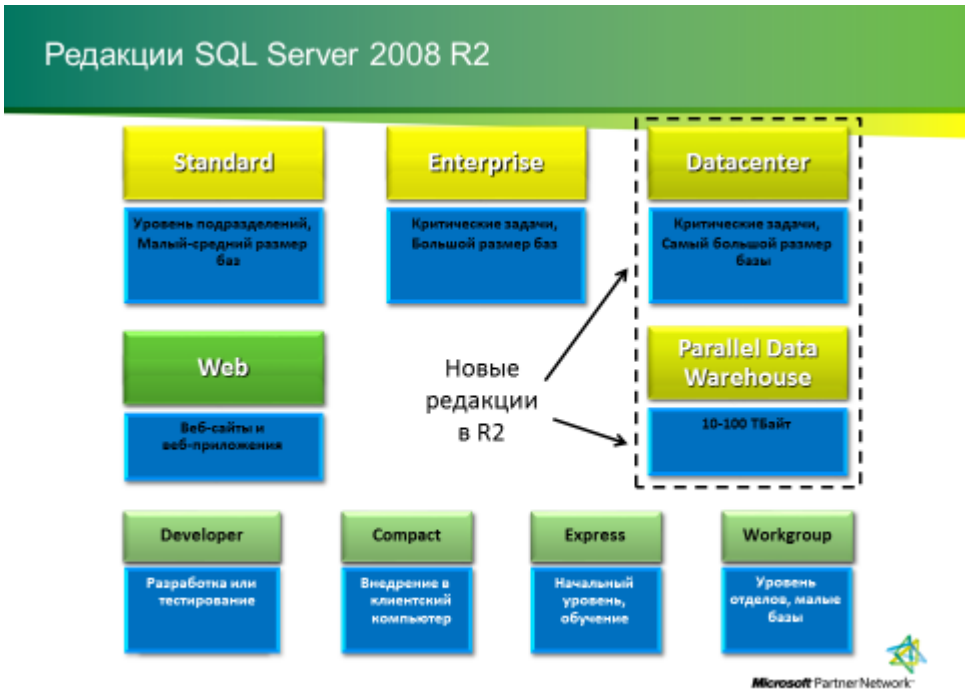

Рис. 4. Версии *SQL Server 2008*

Многие компоненты *SQL Server* можно установить более чем один раз в виде отдельных экземпляров (*instance*) сервера. Каждый экземпляр может настраиваться и управляться независимо. В следующих ситуациях полезно установить больше одной копии компонента *SQL* сервера на одном сервере.

- Можно управлять и защитить каждый экземпляр *SQL Server* отдельно. Поэтому имеет смысл для разных наборов баз данных иметь различных администраторов и/или различные среды безопасности.

- Каждый экземпляр *SQL Server* можно настроить самостоятельно (независимо друг от друга). Некоторым из приложений возможно потребуется конфигурация сервера, которая не соответствует или несовместима с требованиями других приложений.

- Экземпляры *SQL Server* можно использовать для разделения рабочих нагрузок с различными соглашениями об уровне обслуживания. Приложениям базы данных могут потребоваться различные уровни обслуживания, особенно в отношении доступности.

- Может потребоваться поддержка различных версий и выпусков *SQL Server*.

- Приложениям могут требоваться различные параметры сортировки на уровне сервера. Хотя каждая база данных может иметь разные параметры сортировки, приложение может зависеть от параметров сортировки базы данных *tempdb*, когда использует временные объекты.

Различные версии *SQL Server* также могут быть установлены с помощью нескольких экземпляров. Это может помочь при тестировании сценариев обновления или выполнения обновлений.

Графические инструменты администрирования

В *SQL Server 2008* имеется несколько видов административных инструментов. Чаще всего используют инструменты с графическим интерфейсом. Если установлены все *компоненты экземпляров* и *общие компоненты*, то в меню *Microsoft SQL Server 2008* будут доступны следующие команды и подкоманды.

- Импорт и экспорт данных (*Import and Export Data*).
- *SQL Server Business Intelligence Studio*.
- *SQL Server Management Studio*.
- *Analysis Services*.

- Мастер развертывания (*Deployment Wizard*).

- Средства настройки (*Configuration Tools*).

- Диспетчер конфигурации служб *Reporting Services* (*Reporting Services Configuration Manager*).

- Диспетчер конфигурации *SQL Server* (*SQL Server Configuration Manager*).

- Служба отчетов об ошибках и использовании *SQL Server* (*SQL Server Error and Usage Reporting*).

- Центр установки *SQL Server* (*SQL Server Installation Center*).

- Документация и учебные материалы (*Documentation and Tutorials*).

- Обзор образцов *Microsoft SQL Server* (*Microsoft SQL Server Samples Overview*).

- Электронная документация по *SQL Server* (*SQL Server Books Online*).

- Учебные материалы по *SQL Server* (*SQL Server Tutorials*).

- *Integration Services*.

- Средство просмотра профиля данных (*Data Profile Viewer*).

- Программа выполнения пакетов (*Execute Package Utility*).

- Средства обеспечения производительности (*Performance Tools*).

- Помощник по настройке ядра СУБД (*Database Engine Tuning Advisor*);

- *SQL Server Profiler*.

Результаты установки *Microsoft SQL Server 2008 Express Edition* приведены на рис. 5.

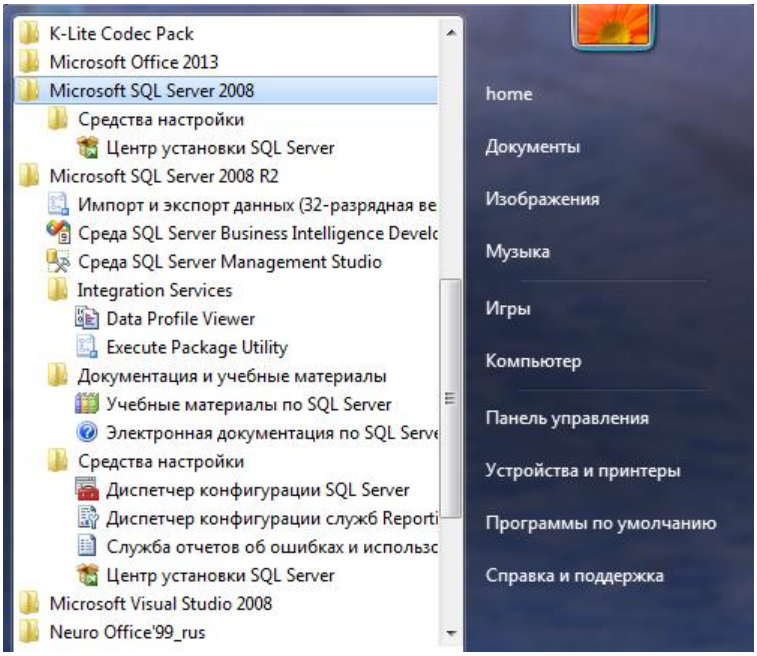

Рис. 5. Состав версии *SQL Server 2008 Express Edition*

Основные задачи администрирования *SQL Server* решаются при помощи консоли *SQL Server Management Studio* (*SSMS*). Она является основным инструментом управления базами данных для серверов баз данных *SQL Server* и представляет собой графический пользовательский интерфейс (*GUI*) и интерфейс сценариев *Transact-SQL* для управления компонентом ядра базы данных и базами данных. Кроме того, можно использовать *SSMS*, чтобы управлять экземплярами *SSAS*, *SSIS* и *SSRS*, а также базами данных на базе облака в *Microsoft Azure SQL Database*.

У *SSMS* есть несколько различных представлений, для управления которыми предназначено меню *Вид* (рис. 6).

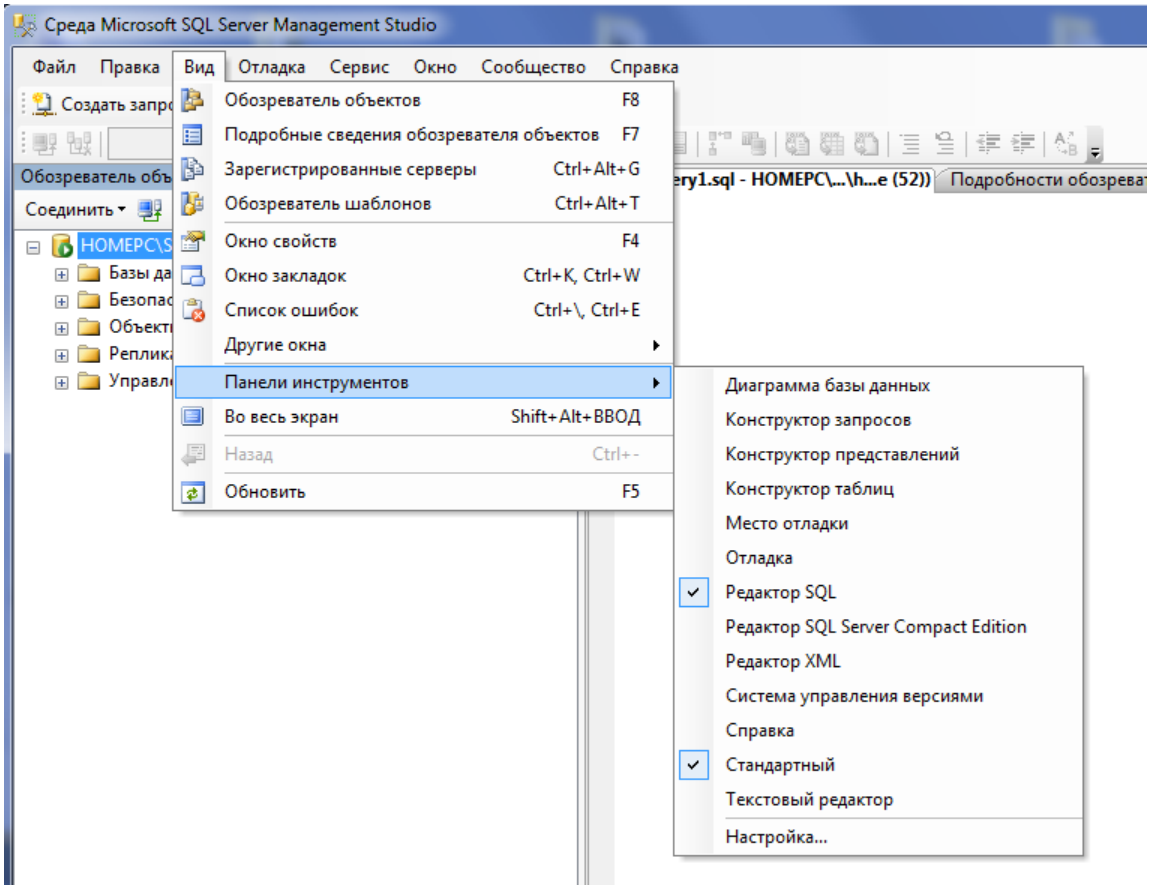

Рис. 6. *SQL Server Management Studio*

При первом запуске консоли она открывается в представлении *Обозреватель объектов*.

Инструменты командной строки

В графических инструментах есть практически все необходимое для работы *SQL Server*. Тем не менее, иногда удобнее бывает работать из командной строки. Данная технология используется при автоматизации установки, администрирования или обслуживание с помощью сценариев. Основным инструментом командной строки является утилита *SQLCMD.EXE*. При помощи этой утилиты из командной строки запускаются запросы на *T-SQL*, системные процедуры и файлы скриптов.

Программы командной строки - общедоступная версия, однако они выпускаются вместе с пакетом установщика для *SQL Server*. Ниже приводится полный формат команды *SQLCMD.EXE*, полученный по запросу `Sqlcmd -?`

```
синтаксис: Sqlcmd  [-U идентификатор_входа]  [-P пароль]
  [-S сервер]      [-H имя_узла]            [-E доверенное_соединение]
  [-N шифровать_соединение]  [-C доверять_сертификату_сервера]
  [-d использовать_имя_БД] [-l время_ожидания_входа]
  [-t время_ожидания запроса]
  [-h заголовки]  [-s разделитель_столбцов]  [-w ширина_экрана]
  [-a размер_пакета]  [-e отображать ввод]
  [-I включить идентификаторы в кавычках]
  [-c конец_команды]  [-L[c] вывести список серверов[очистить_вывод]
  [-q "запрос_командной_строки"]
  [-Q "запрос_командной_строки", выйти после выполнения]
  [-m уровень_ошибок]  [-V уровень_серьезности]
  [-W удалить конечные пробелы]
  [-u вывод в Юникоде]    [-r[0|1] вывод на stderr]
  [-i входной_файл]         [-o выходной_файл]        [-z новый_пароль]
  [-f <кодовая_страница> | i:<кодовая_страница>[,o:<кодовая_страница>]]
  [-Z создать пароль и выйти]
  [-k[1|2] удалить[заменить] управляющие символы]
  [-y ширина_экрана_переменного_типа]
  [-Y ширина_экрана_фиксированного_типа]
  [-p[1] печатать статистику[формат с двоеточиями]]
  [-R использовать региональные параметры клиента]
  [-b Завершение пакетного задания при ошибке]
  [-v var = "значение"...]
  [-Поддержка выделенного соединения с администраторскими полномочиями]
  [-X[1] отключить команды, запустить сценарий, переменные среды [и выход]]
  [-x отключить замену переменных]
  [-? показать сводку по синтаксису]
```

Пример подключения к серверу с выбором базы данных и просмотра в этой базе данных содержимого таблицы.

```
sqlcmd –S HOMEPC\SQLEXPRESS (имя сервера)
USE Моя_БД2
GO
SELECT * FROM dbo.Сотрудники_ТОГУ
GO
```

Кроме *SQLCMD.EXE*, также широко используется инструмент *BCP.EXE*. Это программа массового копирования (*bulk copy program*), которая применяется для импорта, экспорта и копирования данных между экземплярами *SQL Server 2008*. Основное преимущество *BCP* - быстродействие. Она работает значительно быстрее стандартных процедур импорта-экспорта *БД*, но ее недостатком является не слишком удобный синтаксис.

```
C:\Users\home>bcp /? (вывод синтаксиса программы)
Использование: bcp {dbtable | query} {in | out | queryout | format}
файл_данных
[-m макс.число_ошибок]       [-f формат_файла]       [-e файл_ошибок]
[-F первая_строка]           [-L последняя_строка]   [размер_пакета]
[-n собственный_тип]         [-c символьный_тип]
[-w широкосимвольный_тип]
[-N сохр.нетекстовый формат] [-V версия_формата]
[-q идентификатор_в_кавычках]
```

```
[-C ИД_кодовой_страницы]     [-t признак_конца_поля]
[-r признак_конца_строки]
[-i входной_файл]            [-o выходной_файл]      [-a размер_пакета]
[-S имя_сервера]             [-U имя_пользователя]   [-P пароль]
[-T доверенное_соединение]   [-v версия]
[-R вкл.региональные стандарты]
[-k сохранять значения NULL] [-E сохранить значения идентификаторов]
[-h "подсказки загрузки"]    [-x создать XML-файл форматирования]
```

Программа *bcp* выполняет следующие задачи:

- массовый экспорт данных из таблицы *SQL Server* в файл данных;
- массовый экспорт данных из запроса;
- массовый импорт данных из файла данных в таблицу *SQL Server*;
- создание файлов форматирования.

Пример экспорта результатов запроса в *.csv*-файл.

```
cmd> bcp "SELECT [FirstName],[LastName] FROM [MyDatabase].[dbo].[Users]
WHERE [Age]=18" queryout "C:\Result.csv" -c -t ; -S localhost -T
```

Работа с *SQL Server PowerShell*

По умолчанию при установке *Database Engine* программа установки *SQL Server* устанавливает *Windows PowerShell* и настраивает расширения *SQL Server 2008*. Если соответствующая версия *Windows PowerShell* на компьютере уже установлена, программа установки *SQL Server* добавит только необходимые расширения *SQL Server 2008*.

В *Windows PowerShell* впервые появилась концепция команд *cmdlet*. Их можно считать минимальными функциональными единицами *Windows PowerShell*. По сути, *cmdlet* - встроенная команда. Большинство *cmdlet*-команд очень просты и наделены очень ограниченным набором свойств. Использование *cmdlet*-команд не отличается от использования любых других команд и утилит. Имена *cmdlet*-команд можно набирать в любом регистре, в том числе, используя комбинаций прописных и строчных букв.

Чтобы имена *cmdlet*-команд проще запоминались, они составляются из пар глагол-существительное. Существительное конкретизирует объект, с которым работает *cmdlet*. Например, *cmdlet*-команда *get-variable* извлекает именованную переменную среды *Windows PowerShell* и возвращает ее значение.

Если не указать при помощи параметра, какая именно переменная нужна, то команда *get-variable* вернет список всех переменных *Windows PowerShell* и их значений.

Команды *cmdlet* можно выполнять как непосредственно из командной строки оболочки, так и из сценариев. Чтобы превратить любую простую команду или *cmdlet*-команду в сценарий, скопируйте текст в файл и сохраните его с расширением *.psl*. Затем сценарий запускается так же, как любая обычная команда или *cmdlet*. Однако при работе с *Windows PowerShell* текущая папка может отсутствовать в переменной *path*. Поэтому иногда перед именем сценария в текущей папке нужно набирать «./», например: *./runtasks*.

Чтобы запустить *cmdlet*-команду *Windows PowerShell* из командной строки *Windows* или пакетного файла, воспользуйтесь параметром *-command*. Как правило, следует также отменить вывод приветствия *Windows PowerShell* при помощи параметра *-nologo* и остановить выполнение профилей при помощи параметра *-noprofile*. Например, чтобы получить список всех работающих процессов из командной строки или сценария *.BAT*, введите команду

```
msh -nologo -noprofile -command get-process
```

В *Windows PowerShell* также включен мощный язык сценариев со всеми стандартными возможностями языка программирования - циклами, условиями и назначением переменных.

В *SQL Server 2008* имеется расширенная версия *Windows PowerShell - SQL Server PowerShell* (sqlps.exe). Она применяется для работы с экземплярами *SQL Server Database Engine* и объектами в этих экземплярах.

Чтобы вызвать *SQL Server PowerShell*, откройте *Окно командной строки* и введите команду *sqlps*. Чтобы выйти из *SQL Server PowerShell* и вернуться в командную строку, введите *exit*.

Одна из трудностей использования *PowerShell* заключается в том, что нужно знать, какие именно команды и параметры следует использовать. Изучение эффективного применения встроенных в *PowerShell* команд помощи является обязательным условием для использования различных команд. Во

встроенной системе помощи (*Help*) можно найти все необходимые инструкции вместе с примерами применения.

Чтобы просмотреть список всех *cmdlet*-команд, введите `get-command`. Ниже приведена лишь небольшая часть выдаваемой при этом информации.

```
Function        C:                                 Set-Location C:
Alias           cat                                Get-Content
Alias           cd                                 Set-Location
Function        cd..                               Set-Location
Function        cd\                                Set-Location \
Alias           chdir                              Set-Location
Alias           clc                                Clear-Content
Alias           clear                              Clear-Host
Cmdlet          Clear-Content  Clear-Content [-Path] <Strin...
Function        Clear-Host     $space = New-Object System.M...
Cmdlet          Clear-Item     Clear-Item [-Path] <String[]...
Cmdlet          Clear-ItemProperty                 Clear-ItemProperty [-Path]
Cmdlet          Clear-Variable  Clear-Variable [-Name] <Stri...
Alias           clhy                               Clear-History
Alias           cli                                Clear-Item
Alias           clp                                Clear-ItemProperty
Alias           cls                                Clear-Host
Alias           clv                                Clear-Variable
Alias           compare                            Compare-Object
Cmdlet          Compare-Object  Compare-Object [-ReferenceOb...
Cmdlet          ConvertFrom-SecureString           ConvertFrom-SecureString
Cmdlet          Convert-Path    Convert-Path [-Path] <String...
Cmdlet          ConvertTo-Html  ConvertTo-Html [[-Property] ...
Cmdlet          ConvertTo-SecureString             ConvertTo-SecureString
Cmdlet          Convert-UrnToPath     Convert-UrnToPath [-Urn] <St...
```

Чтобы получить подробную информацию о конкретной *cmdlet*-команде, введите `get-helpИмяКоманды -detailed`, где *ИмяКоманды* - имя нужной *cmdlet*-команды. Ниже приведен пример и часть результата выполнения команды.

```
PS SQLSERVER:\> get-help Clear-Variable -detailed

ИМЯ     Clear-Variable
ОПИСАНИЕ    Удаляет значение переменной.
СИНТАКСИС     Clear-Variable [-Name] <string[]> [-Exclude <string[]>] [-
Force] [-Include]    <string[]>] [-PassThru] [-Scope <string>] [-Confirm] [-
WhatIf] [<CommonPara>]    meters>]

ОПИСАНИЕ
    Командлет Clear-Variable удаляет данные, хранящиеся в переменной, но не
саму переменную. В результате переменная получает значение NULL (пустое).
Если переменная имеет указанный тип данных или объектный тип, командлет
Clear-V ariable оставляет тип объекта, хранимого в переменной, прежним.

ПАРАМЕТРЫ
    -Exclude <string[]>
Исключает указанные элементы. Значение данного параметра определяет значение
параметра Name. Введите часть имени или шаблон, например "s*". Подстановоч-
ные знаки разрешены.
```

```
    -Force [<SwitchParameter>] Позволяет командлету очистить переменную,
даже если она доступна только для чтения. Даже при использовании параметра
Force командлет не может очистить константу.
    -Include <string[]> Очищает только указанные элементы. Значение данного
параметра определяет значение параметра Name. Введите часть имени или шаб-
лон, например "s*". Подстановочные знаки разрешены.

    -------------------------- ПРИМЕР 1 --------------------------
    C:\PS>clear-variable my* -global

Эта команда удаляет значения глобальных переменных, имена которых начинаются
на "my".

    -------------------------- ПРИМЕР 2 --------------------------

    C:\PS>clear-variable -name processes

Эта команда удаляет значение переменной $processes. После этого переменная
$processes продолжает существовать, но имеет значение NULL.

ЗАМЕЧАНИЯ
    Для просмотра примеров введите: "get-help Clear-Variable -examples".
    Для получения дополнительных сведений введите: "get-help Clear-Variable
-detailed".
    Для получения технических сведений введите: "get-help Clear-Variable -
full".
```

Чтобы посмотреть только примеры использования команды следует задать соответствующий параметр (`get-help Clear-Variable –Examples`).

Чтобы получить подробную информацию о поставщике *SQL Server*, обеспечивающем функциональность *SQL Server в PowerS hell*, введите `get-help sqlserver | more`.

*Я*зык *T-SQL* и возможности *PowerShell* во многом эквивалентны. Однако многие специалисты в области администрирования *SQL Server* используют оболочку *PowerShell* в своей повседневной работе. Как обычно истина по середине - максимального эффекта можно добиться при совместном использовании *PowerShell* и *T-SQL*.

Использование *Transact-SQL* для выполнения задач администрирования. Можно выполнять большинство административных задач в среде *SSMS* с помощью графического пользовательского интерфейса. Однако некоторые задачи могут быть выполнены только с помощью инструкций *Transact-SQL*, и даже если задача может быть выполнена в графическом интерфейсе, целесообразно использовать код *Transact-SQL*, который может быть сохранен в виде

сценария и повторно выполняться (или запускаться автоматически с помощью планировщика).

Большинство графических интерфейсов в *SSMS* имеют кнопку *Script*, с помощью которой автоматически генерируется эквивалентный код *T-SQL*.

Команды *T-SQL*, которые можно использовать для выполнения задач управления, включают.

- Инструкции языка *DDL*. Например, можно использовать инструкции «*CREATE DATABASE*» или «*DROP DATABASE*» для создания БД или для удаления базы данных.

- Системные хранимые процедуры и функции. *SQL Server* предоставляет системные хранимые процедуры и функции, которые инкапсулируют общие задачи настройки и управления системы. Например, можно использовать системную хранимую процедуру *sp_configure* для задания параметров конфигурации экземпляра *SQL Server* (см. пример на рис. 7).

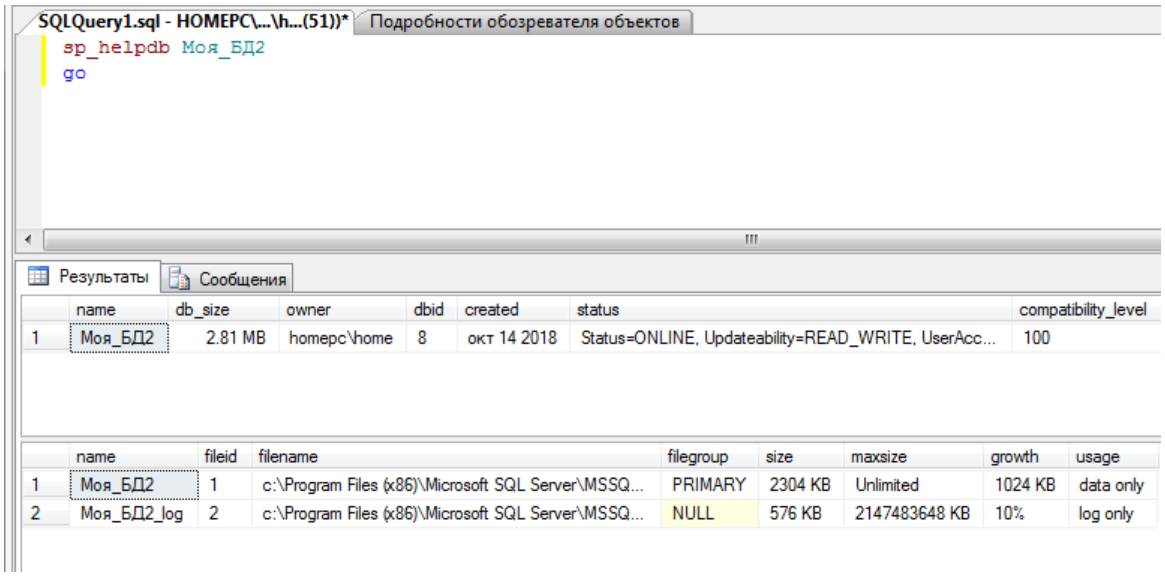

Рис. 7. Пример выполнения системной хранимой процедуры

- *DBCC (Database Console Commands)*. Команды *DBCC* используются для выполнения конкретных задач по настройке и обслуживанию, а также для выполнения проверок в базе данных. Например, можно использовать команду *DBCC CHECKDB* для проверки физической и логической целостности объектов в базе данных.