МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждения высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра информационной безопасности

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_____»_____2017 г.

АДМИНИСТРИРОВАНИЕ БАЗЫ ДАННЫХ

Методические указания по выполнению лабораторной работы №3
для студентов направления подготовки бакалавриата
10.03.01«Информационная безопасность»

Курск 2017

УДК 621.(076.1)

Составители: А.Г. Спеваков

Рецензент

Кандидат технических наук, доцент кафедры
«Информационная безопасность» И.В. Калуцкий

Администрирование базы данных [Текст]: методические указания по выполнению лабораторной работы / Юго-Зап. Гос. ун-т; сост.: А.Г. Спеваков. – Курск, 2017. – 37с.: ил. 26, табл. 1. – Библиогр.: с. 37.

Содержат сведения по вопросам проектирования баз данных. Указывается порядок выполнения лабораторной работы, правила содержание отчета.

Методические указания соответствуют требованиям программы, утвержденной учебно-методическим объединением по специальности.

Предназначены для студентов направления подготовки бакалавриата 10.03.01 «Информационная безопасность».

Текст печатается в авторской редакции

Подписано в печать . Формат 60х84 1/16.
Усл.печ. л. 2,15. Уч.-изд. л. 1,95. Тираж 100 экз. Заказ. Бесплатно.
Юго-Западный государственный университет.
305040, г.Курск, ул. 50 лет Октября, 94.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

При работе с базами данных очень важным моментом является разделение прав на управление базой данных. Одни пользователи могут вносить изменения, другие – имеют право только на чтение. Разделение прав – одна из важных функций политики безопасности. В данной работе в процессе администрирования рассмотрится добавление пользователей имеющих доступ к базе данных и процесс наделения пользователей правами.

ЦЕЛЬ РАБОТЫ

Научиться добавлять пользователей в SQLServer и наделять их различными правами на управление базой данных (admin и user). Научиться создавать план обслуживания базы данных.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Получить задание.
2. Изучить теоретическую часть.
3. Разработать политику безопасности.
4. Добавить пользователей.
5. Наделить их правами на управление базой.
6. Проверить в SQL Management Studio наличие пользователей.
7. Создать Back Up базы данных.

СОДЕРЖАНИЕ ОТЧЕТА

1. Индивидуальное задание.
2. Теоретические сведения.
3. Выполнение работы.
4. Создание/Добавление пользователей.
5. Создание Back Up
6. Вывод о проделанной работе

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

При определении прав пользователя SQL Server 2008 использует два уровня зашиты.

Первый уровень — проверка подлинности пользователя. Во время проверки определяется, имеется ли у пользователя право на подключения к данному SQL Server 2008.

Второй уровень системы безопасности — авторизация, часто называемая также проверкой прав доступа. При этом определяется, какие действия пользователь сможет выполнять с БД, после того как он пройдет проверку подлинности SQL Server 2008.

Чтобы установить соединение с SQL Server 2008, пользователь должен указать правильный идентификатор учетной записи (login identifier) пользователя, который определяет права доступа к SQL Server 2008. Каждый SQL Server из числа имеющихся в системе проверяет, действительно ли идентификатор учетной записи пользователя, введенный при установке соединения, обеспечивает пользователю право подключаться к этому SQL Server. Проверка идентификатора учетной записи пользователя называется проверкой подлинности (authentication). В SQL Server 2008 предусмотрены два вида проверки подлинности: средствами Windows и средствами SQL Server. Подключаясь к SQL Server 2008, пользователь указывает вид проверки подлинности для данного соединения.

Администратор БД может предоставить пользователям и группам пользователей Windows NT 4.0/2000 право устанавливать соединение с данным SQL Server 2008. Если при подключении указывается, что проверка подлинности выполняется средствами Windows, то для проверки того, зарегистрирован ли данный пользователь в этой сети, используются средства Windows. SQL Server идентифицирует пользователя по имени его учетной записи, существующей в сети, и разрешает или запрещает этому пользователю устанавливать соединение, не требуя его отдельной регистрации как пользователя SQL Server 2008. Такой тип соединения называется доверенным (trusted).

При проверке подлинности средствами Windows используются механизмы защиты Windows NT 4.0/2000, в том числе такие средства, как защищенный режим проверки прав

пользователя в сети и шифрование пароля, аудит, ограничение срока действия пароля, ограничение на минимальную длину пароля и блокирование учетной записи пользователя после нескольких неудачных попыток регистрации.

Администратор БД может создавать учетные записи пользователей SQL Server, вводя имя пользователя и назначая ему соответствующий пароль. Эти учетные записи никак не связаны с учетными записями пользователей и групп Windows NT 4.0/2000.

Если при подключении к серверу выбран режим проверки подлинности средствами SQL Server, то SQL Server 2008 сам проверяет подлинность пользователя, уточняя, имеется ли такая учетная запись с указанным именем и паролем на SQL Server 2008.

В SQL Server 2008 есть два режима проверки подлинности. По умолчанию используется режим проверки подлинности средствами Windows (Windows Authentication Mode).

При этом устанавливать соединение с SQL Server разрешается только зарегистрированным пользователям Windows NT 4.0/2000, прошедшим проверку подлинности Windows. SQL Server 2008 также может работать в смешанном режиме (Mixed Mode), при этом пользователь может подключиться к SQL Server 2008, если он прошел проверку подлинности Windows NT 4.0/2000 или указал правильное имя и пароль пользователя SQL Server.

После того как SQL Server 2008 проверит подлинность пользователя, SQL Server 2008 определяет права данного пользователя выполнять различные действия в размещенных на сервере БД. Сам по себе идентификатор учетной записи пользователя не дает зарегистрированному пользователю прав доступа к различным объектам БД. Он лишь позволяет перейти к следующему этапу — авторизации, или проверке прав пользователя. Такой механизм защиты гарантирует, что зарегистрированный пользователь не получит автоматически доступ ко всем БД на SQL Server 2008, с которым он установил соединение.

Как правило, администратор БД должен сопоставить идентификатор учетной записи пользователя идентификатору пользователя в БД, прежде чем пользователь, подключившийся с использованием этого идентификатора учетной записи, получит

доступ или сможет выполнить какие-либо действия в этой БД. Администратор БД определяет права доступа к объектам (таким как таблицы, представления и хранимые процедуры) в БД для всех учетных записей пользователей.

Если учетная запись пользователя на сервере, позволяющая пользователю подключиться к SQL Server 2008, не связана ни с одной учетной записью пользователя в БД, она автоматически связывается с идентификатором учетной записи пользователя guest в этой БД (если такой идентификатор существует). Если в БД присутствует учетная запись пользователя guest, права подключающегося пользователя ограничиваются правами пользователя guest. Если в БД отсутствует учетная запись пользователя guest, подключающийся пользователь не получит доступ к БД до тех пор, пока его учетная запись на сервере не будет связана с учетной записью БД. По умолчанию во всех вновь созданных пользовательских БД отсутствует учетная запись пользователя guest.

Роли позволяют администратору БД объединять пользователей в группы, для которых задаются определенные права.. Роли в SQL Server 2008 во многом аналогичны группам пользователей в Windows NT 4.0/2000. В SQL Server 2008 имеются встроенные роли, определенные на уровне сервера, и роли, определенные на уровне БД. Для этих ролей заранее установлены права на уровне всего сервера и на уровне БД. Кроме того, администратор БД может создавать новые роли на уровне БД. Каждый пользователь БД является участником роли БД public и, следовательно, обладает всеми правами, предоставленными роли public, если для него особо не определены какие-либо специальные права. Дополнительные права следует предоставлять пользователю или группе, к которой он принадлежит, в явном виде.

Фиксированные роли базы данных задаются на уровне базы данных и предусмотрены в каждой базе данных. Элементы ролей базы данных db_owner и db_securityadmin могут управлять членством в фиксированных ролях базы данных, однако только члены роли базы данных db_owner могут добавлять членов в фиксированную роль базы данных db_owner.

Имеются следующие фиксированные роли базы данных:

- db_accessadmin
- db_backupoperator
- db_datareader
- db_datawriter
- db_ddladmin
- db_denydatareader
- db_denydatawriter
- db_owner
- db_securityadmin

Каждый пользователь базы данных принадлежит к роли базы данных public. Если для пользователя не были заданы особые разрешения на защищаемый объект, то он наследует разрешения роли public на этот объект.

Члены фиксированной роли базы данных db_accessadmin могут добавлять или удалять права удаленного доступа для имен входа и групп Windows, а также имен входа SQL Server.

Члены фиксированной роли базы данных db_backupoperator могут выполнять ее резервирование.

Элементы фиксированной роли базы данных db_datareader могут считывать все данные из всех пользовательских таблиц.

Члены фиксированной роли базы данных db_datawriter могут добавлять, удалять или изменять данные во всех пользовательских таблицах.

Члены фиксированной роли базы данных db_ddladmin могут выполнять любые команды языка определения данных (DDL) в базе данных.

Члены фиксированной роли базы данных db_denydatareader не могут считывать данные из пользовательских таблиц базы данных.

Члены фиксированной роли базы данных db_denydatawriter не могут добавлять, изменять или удалять данные в пользовательских таблицах базы данных.

Члены фиксированной роли базы данных db_owner могут выполнять все действия по конфигурации и обслуживанию базы данных, а также удаление базы данных.

Элементы фиксированной роли базы данных db_securityadmin могут изменять членство в роли и управлять разрешениями.

Чтобы пользователь мог выполнять какие-либо действия в системе SQL Server 2008, он должен подключиться к SQL Server 2008, используя учетную запись, и к БД, используя регистрационную запись.

Шифрование - процесс кодирования конфиденциальных данных с использованием ключа или пароля. Шифрование надежно защищает данные и сокращает вероятность несанкционированного раскрытия конфиденциальной информации, так как без соответствующего ключа или пароля данные бесполезны. SQL Server располагает многими режимами шифрования, в том числе на уровне ячеек, базы данных, файлов через Windows и шифрования на транспортном уровне.

Шифрование SQL Server не решает проблему доступности инфраструктуры и баз данных SQL Server, но повышает защищенность данных на уровнях базы данных и операционной системы, даже если нарушена конфиденциальность инфраструктуры или баз данных SQL Server.

Модель шифрования SQL Server в основном предоставляет функции управления ключами шифрования, соответствующие стандарту ANSI X9.17. В этом стандарте определены несколько уровней ключей шифрования, использующихся для шифрования других ключей, которые в свою очередь применяются для шифрования собственно данных.

В таблице №1 перечислены уровни ключей шифрования SQL Server и ANSI X9.17.

Таблица 1 - уровни ключей шифрования SQL Server и ANSI X9.17

Таблица	Уровень шифрования ключей SQL Server и ANSI X9.17	
Уровень SQL Server	Уровень ANSI X9.17	Описание
SMK	Главный ключ	SMK – ключ верхнего уровня, используемый для шифрования DMK. SMK шифруется с применением DPAPI.
DMK	Ключ шифрования ключей	DMK – симметричный ключ, используемый для шифрования симметричного ключа, ассиметричного ключа и сертификата. Для каждой базы данных может быть определен только один DMK.
Симметричные ключи, ассиметричные ключи и сертификаты	Ключ данных	Симметричные ключи, ассиметричные ключи и сертификаты используются для шифрования данных.

Главный ключ службы Service master key(SMK) - ключ верхнего уровня и предок всех ключей в SQL Server. SMK - асимметричный ключ, шифруемый с использованием Windows Data Protection API (DPAPI). SMK автоматически создается, когда шифруется какой-нибудь объект, и привязан к учетной записи

службы SQL Server. SMK используется для шифрования главного ключа базы данных Database master key (DMK).

Второй уровень иерархии ключей шифрования - DMK. С его помощью шифруются симметричные ключи, асимметричные ключи и сертификаты. Каждая база данных располагает лишь одним DMK.

Следующий уровень содержит симметричные ключи, асимметричные ключи и сертификаты. Симметричные ключи - основное средство шифрования в базе данных. Microsoft рекомендует шифровать данные только с помощью симметричных ключей. Кроме того, в SQL Server 2008 и более новых версиях есть сертификаты уровня сервера и ключи шифрования базы данных для прозрачного шифрования данных. На рисунке №1 показана иерархия ключей шифрования для SQL Server 2008 и более новых версий.

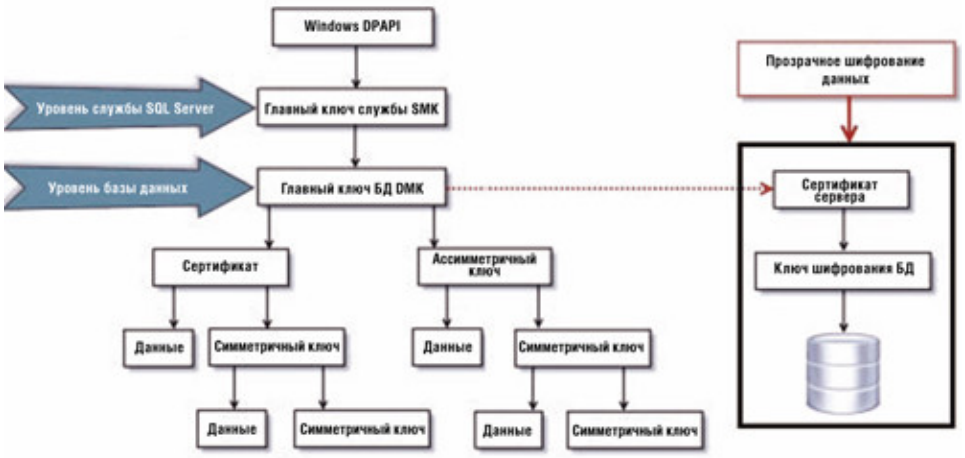

Рис. 1 - Иерархия ключей шифрования для SQL Server 2008.

После знакомства с иерархией ключей шифрования SQL Server мы рассмотрим способы реализации шифрования, доступные в SQL Server.

Начиная с SQL Server 2005, можно шифровать или расшифровывать данные на сервере. Делать это можно различными способами. Например, можно шифровать данные в базах данных одним из следующих методов.

- Пароль. Это наименее надежный способ, так как для шифрования и расшифровки данных используется одна и та же парольная фраза. Если хранимые процедуры и функции не

зашифрованы, то доступ к парольной фразе возможен через метаданные.

- Сертификат. Этот способ обеспечивает надежную защиту и высокое быстродействие. Сертификат можно связать с пользователем; подписать его необходимо с помощью DMK.
- Симметричный ключ. Достаточно надежен, удовлетворяет большинству требований к безопасности данных и обеспечивает достаточное быстродействие. Для шифрования и расшифровки данных используется один ключ.
- Асимметричный ключ. Обеспечивает надежную защиту, так как применяются различные ключи для шифрования и расшифровки данных. Однако это негативно влияет на быстродействие. Специалисты Microsoft не рекомендуют использовать его для шифрования крупных значений. Асимметричный ключ может быть подписан с использованием DMK или создан с помощью пароля.

SQL Server располагает встроенными функциями для шифрования и расшифровки на уровне ячеек. Функции шифрования:

* ENCRYPTBYKEY, использует симметричный ключ для шифрования данных;
* ENCRYPTBYCERT, использует открытый ключ сертификата для шифрования данных;
* ENCRYPTBYPASSPHRASE, использует парольную фразу для шифрования данных;
* ENCRYPTBYASYMKEY, использует асимметричный ключ для шифрования данных.

Функции расшифровки:

- DECRYPTBYKEY, использует симметричный ключ для расшифровки данных;
- DECRYPTBYCERT, использует открытый ключ сертификата для расшифровки данных;
- DECRYPTBYPASSPHRASE, использует парольную фразу для расшифровки данных;
- DECRYPTBYASYMKEY, использует асимметричный ключ для расшифровки данных;

- DECRYPTBYKEYAUTOASYMKEY, использует асимметричный ключ, который автоматически расшифровывает сертификат.

SQL Server располагает двумя системными представлениями, с помощью которых можно получить метаданные для всех симметричных и асимметричных ключей, существующих в экземпляре SQL Server. Как видно из названий, sys.symmetric_keys возвращает метаданные для симметричных, а sys.asymmetric_keys - для асимметричных ключей. Еще одно полезное представление - sys.openkeys. В этом представлении каталога содержится информация о ключах шифрования, открытых в текущем сеансе.

В SQL Server 2008 появилась возможность зашифровать всю базу данных с использованием прозрачного шифрования. При таком шифровании можно защитить базы данных без изменения существующих приложений, структур баз данных или процессов. Это лучший вариант для выполнения требований нормативных актов и правил корпоративной безопасности, поскольку шифруется вся база данных на жестком диске.

Прозрачное шифрование данных шифрует базы данных в реальном времени, по мере внесения записей в файлы (*.mdf) базы данных SQL Server и файлы (*.ldf) журнала транзакций. Записи также шифруются в реальном времени во время резервного копирования базы данных, а затем формируются моментальные снимки. Данные шифруются перед записью на диск и расшифровываются перед извлечением. Процесс полностью прозрачен для пользователя или приложения, поскольку выполняется на уровне SQL Server Service.

При прозрачном методе SQL Server шифрует базу данных с помощью ключа шифрования базы данных. Этот асимметричный ключ хранится в загрузочной записи базы данных и потому всегда доступен при восстановлении.

Как показано на рисунке 1, ключ шифрования базы данных шифруется с использованием сертификата сервера, который шифруется с помощью DMK базы данных master. DMK базы данных master шифруется с применением SMK. SMK - асимметричный ключ, зашифрованный с помощью Windows DPAPI. Ключ SMK автоматически формируется при первом

шифровании любого объекта и привязан к учетной записи SQL Server Service.

Приступим к резервному копированию (Backup). Начнем с того, что резервное копирование (англ. backup) — процесс создания копии данных на носителе (жёстком диске, дискете и т. д.), предназначенном для восстановления данных в оригинальном месте их расположения в случае их повреждения или разрушения, соответствующими программами — резервными дубликаторами данных.

Полный бэкап — это фундамент, на котором будет держаться вся ваша система резервирования. По большому счету, это копия базы данных, которая включает в себя вообще все — таблицы, данные, индексы, триггеры, статистику, хранимые процедуры и еще много разного добра. Более того — если вы выбираете вариант динамического резервного копирования (то есть во время резервирования данных пользователи могут полноценно работать с сохраняемой БД), то все изменения, которые пользователи сделают за то время, пока создается бэкап, тоже будут сохранены. Вы можете легко вернуть базу данных к тому состоянию, в котором она была в какой-то определенный момент, если у вас есть полный бэкап, сделанный в это время.

Бэкап лога отличается от полного бэкапа тем, что в него входят исключительно изменения базы данных (то есть операции INSERT, UPDATE и DELETE) с момента последнего бэкапа, будь то полный бэкап, дифференцированный или предыдущий бэкап лога. Поскольку объем сохраняемых данных крайне мал, такой тип резервирования намного быстрее, требует меньше ресурсов и занимает меньше места на диске. Недостатков, увы, тоже хватает. В первую очередь, бэкап лога бесполезен, если у вас нет хотя бы одного полного бэкапа. Объясняется это тем, что в таком логе не сохраняется никакая информация о таблицах, индексах, хранимых процедурах и так далее. Вторым существенным недостатком является то, что если с момента последнего полного бэкапа вы успели сделать сотню бэкапов лога, а потом случилась беда, то прежде, чем вы восстановите заветный сотый, вам потребуется восстановить не только полный бэкап, но и предыдущие девяносто девять бэкапов лога, да к тому же в правильном порядке.

Согласитесь, приятного в такой перспективе не очень много. Еще одна немаловажная особенность заключается в том, что бэкап лога доступен только для тех баз данных, у которых указан FULL или BULK LOGGED режим восстановления.

Дифференцированный бэкап — это что-то среднее между полным бэкапом и бэкапом лога. В нем сохраняются изменения, сделанные с момента последнего полного бэкапа по настоящее время. Главным его преимуществом по сравнению с бэкапом лога является то, что для полного восстановления базы данных вам достаточно восстановить только лишь полный и последний дифференцированный бэкапы. Недостатком же является то, что вы не можете восстановить промежуточное состояние базы данных на какой-то момент времени — история изменения БД в дифференцированном бэкапе не сохраняется. Как и бэкап лога, такой бэкап бесполезен, если у вас нет полной копии базы данных.

Важно:

•Используйте DBCC CHECKDB для проверки каждой базы данных перед копированием, это своевременно предупредит вас о надвигающихся проблемах.

•Используйте опцию BACKUP WITH CHECKSUM, чтобы убедиться, что все прошло хорошо. Недостатком такого решения является то, что для больших баз данных проверка контрольной суммы может серьезно загрузить систему.

•Если у вас проблемы с дисковым пространством, доступным для хранения бэкапов, используйте опцию BACKUP WITH COMPRESSION. Сжатие позволяет уменьшить итоговый размер файла с бэкапом в несколько раз, обычно в 3-5. Как и в случае с проверкой контрольной суммы, такая операция очень серьезно влияет на производительность, особенно для больших баз данных.

•Создавайте отдельный файл для каждого бэкапа, не храните все бэкапы в одном файле. В таком случае, если с этим файлом что-то случится, вы потеряете один бэкап, а не все.

•Естественно, не храните бэкап на том же диске, на котором хранится ваша БД.

•Если существует угроза, что кто-то посторонний будет иметь доступ к вашим бэкапам, используйте парольную защиту или шифрование.

•Автоматизируйте процесс создания бэкапов. Это просто, и вы будете уверены, что у вас всегда есть свежая копия базы данных.

•Бэкапы бесполезны, если вы не можете их использовать. Поэтому регулярно тренируйтесь в восстановлении баз данных, чтобы при необходимости вы могли в стрессовой ситуации сделать все быстро и безошибочно.

ВЫПОЛНЕНИЕ РАБОТЫ

1. Открываем "Безопасность" => Нажимаем правой кнопкой мыши по "Имена входа" => Выбираем "Создать имя входа...". (Рис. 2)

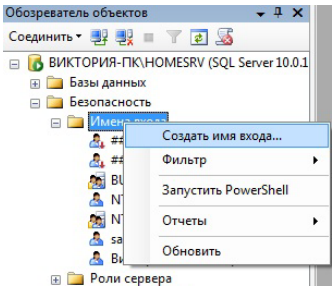

Рис. 2 – Порядок открытия окна «Создать имя входа»

2. Заполняем строку "Имя входа:". Выбираем "Проверка подлинности SQL Server". Заполняем строки "Пароль" и "Подтверждение пароля. Убираем галочку с "Требовать использование политики паролей". Выбираем нужную базу данных. Выбираем "Язык по умолчанию:" - "Russian". (Рис. 3)

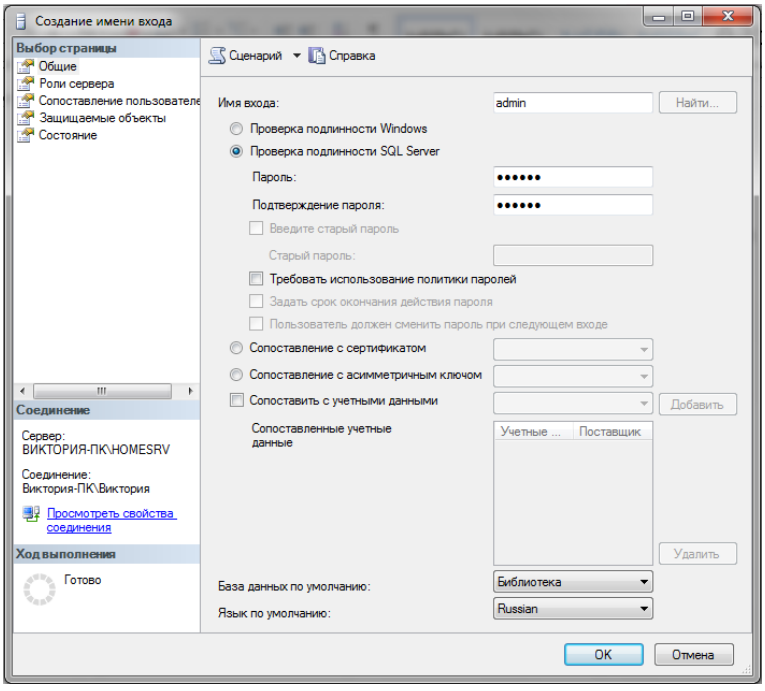

Рис. 3 – Общие параметры «Имя входа» - admin

3. Переходим во вкладку "Роли сервера:". Выделяем всё галочками. (Рис. 4)

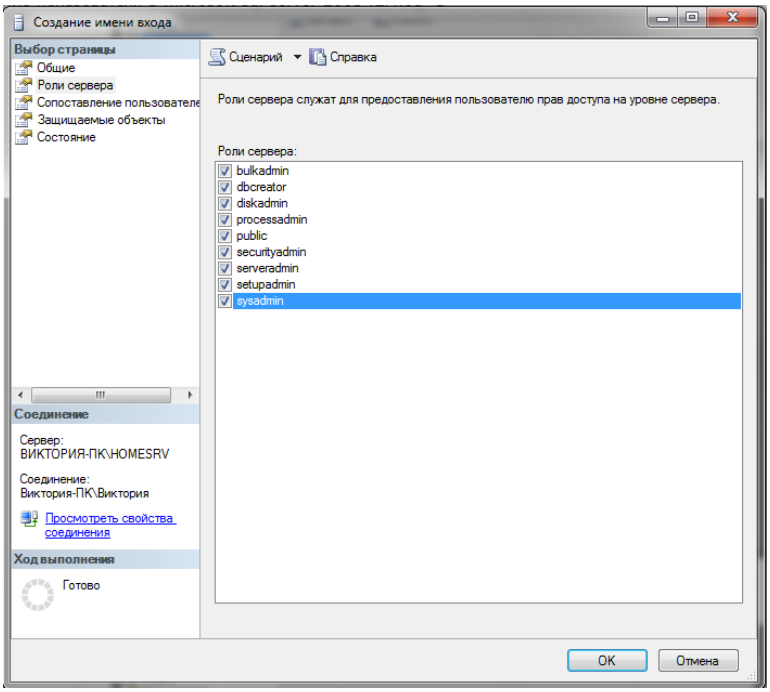

Рис. 4 – Вкладка «Роли сервера» - admin

4. Переходим во вкладку "Сопоставление пользователей". Выбираем БД `Библиотека` и назначаем роли сервера пользователю admin. (Рис. 5)

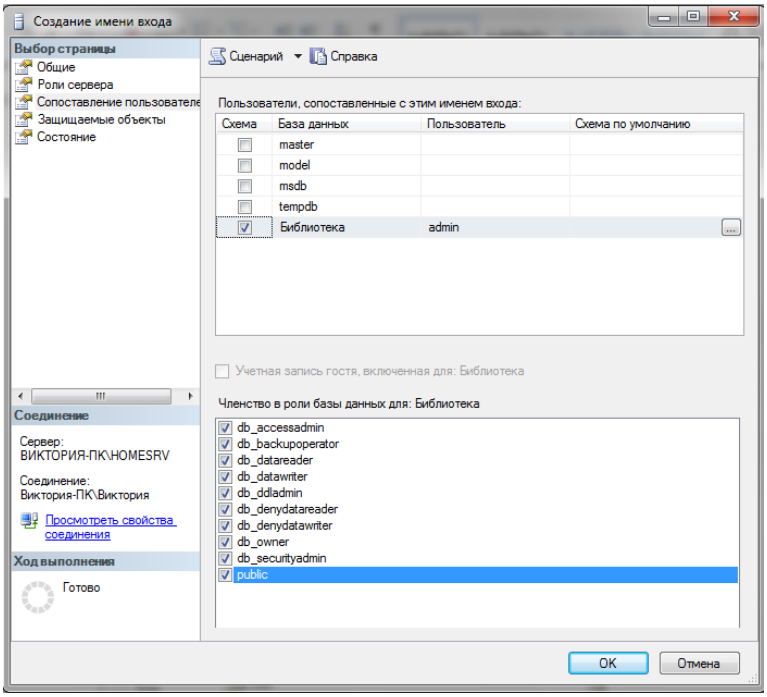

Рис. 5 – Вкладка «Сопоставление пользователей» - admin

5. Сохраняем изменения нажав на кнопку ОК, и переходим к созданию пользователя «user», аналогичным способом. (Рис. 6, 7)

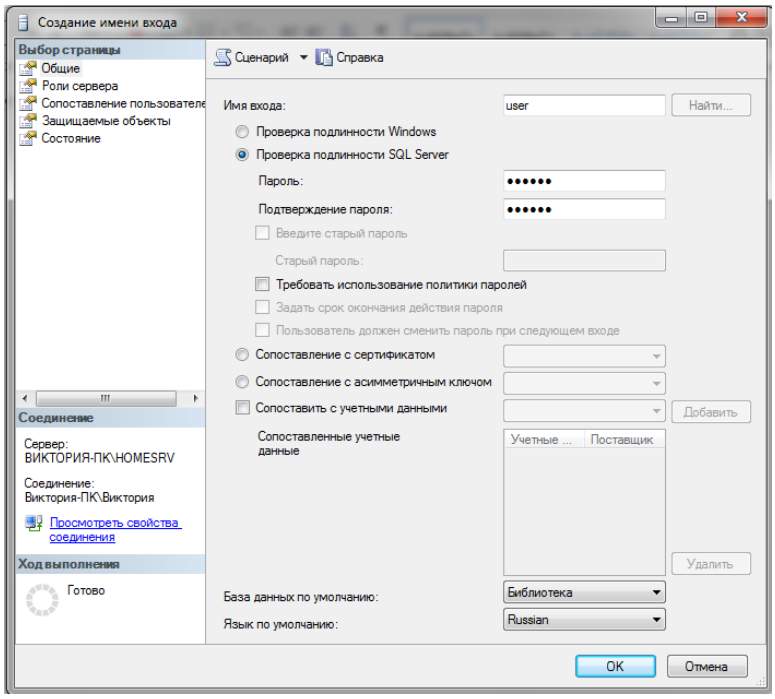

Рис. 6 – Вкладка «Общие» - user

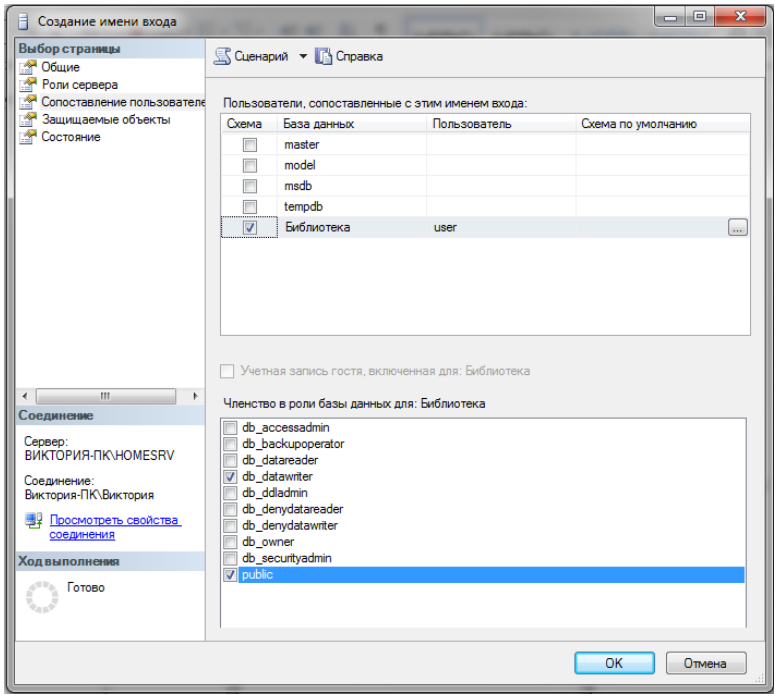

Рис. 7 – Вкладка «Сопоставление пользователей» - user

6. Пользователи user и admin успешно добавились в обозреватель объектов. (Рис. 8)

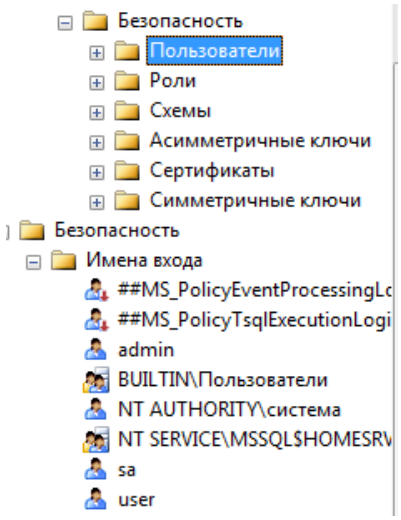

Рис. 8 – Отображение user и admin в обозревателе объектов

Теперь приступим к созданию BackUp для нашей базы данных. Первое что нам необходимо сделать, это убедиться, что Агент SQL Server установлен и работает. Для этого запустим оснастку «Службы» («Пуск» (Start) — «Администрирование» (Administrative Tools) — «Службы» (Services)) и в списке служб найдем службу «Агент SQL сервер». (Рис. 9)

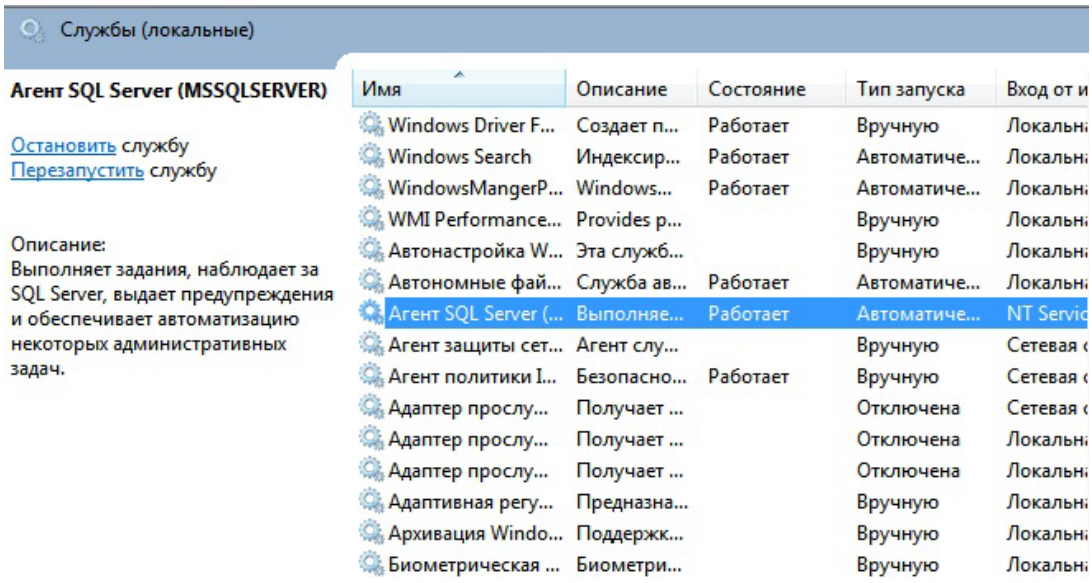

Рис. 9 – Окно «Службы»

Откроем свойства этой службы (кликнув по ней 2 раза) и убедимся, что:

1. Тип запуска стоит «Автоматически» (Startup type: Automatic);
2. Состояние «Работает» (Service status: Started. (Рис. 10)

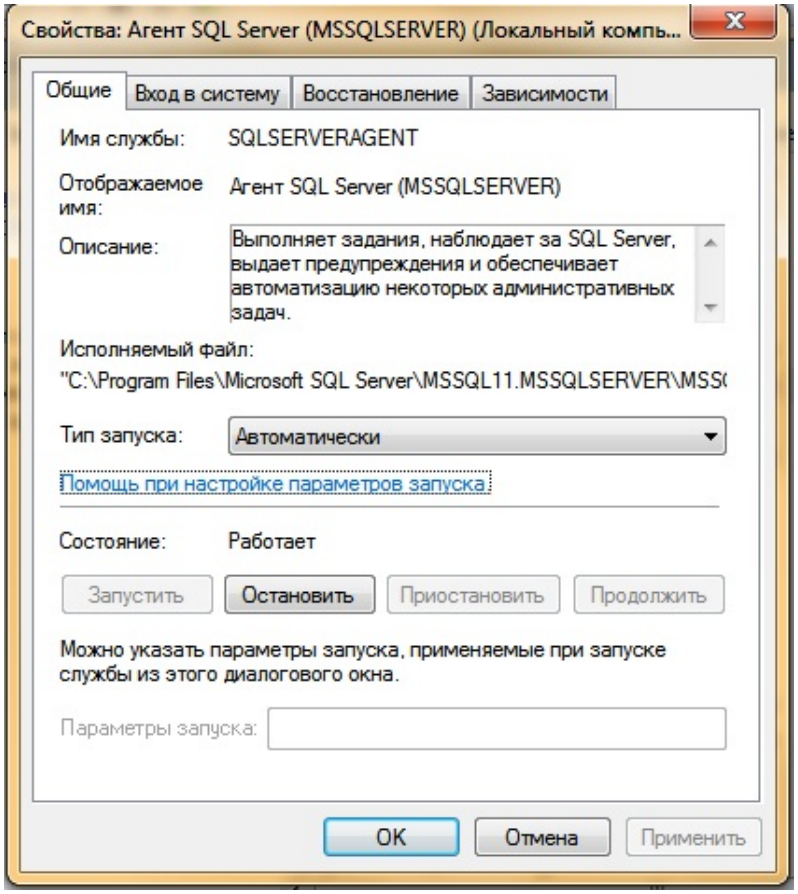

Рис. 10 – Выбор свойств Агент SQL Server

Теперь запустим SQL Sever Management Studio («Пуск» — «Все программы» — «Microsoft SQL Server 2008 R2» — «Средства SQL Server 2008 R2«) и введем данные для авторизации.

После чего, еще раз убедимся, что Агент SQL Server работает (в обозревателе объектов должна быть вкладка «Агент SQL Server» (SQL Server Agent) с зеленой иконкой слева. (Рис. 11)

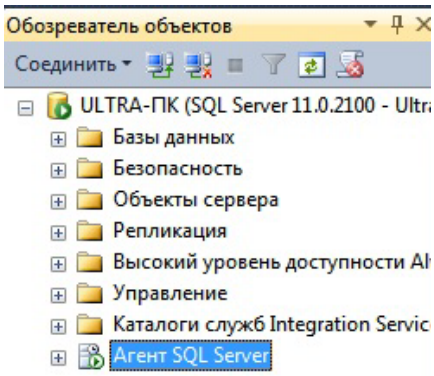

Рис. 11 – Отображение «Агент SQL Server» в обозревателе объектов

Теперь перейдем непосредственно к созданию плана обслуживания. В обозревателе объектов (Object Explorer) раскроем вкладку «Управление» (Management), кликнем правой кнопкой мыши по вкладке «Планы обслуживания» (Maintenance Plans) и в контекстном меню выберем «Мастер планов обслуживания» (Maintenance Plan Wizard). (Рис. 12)

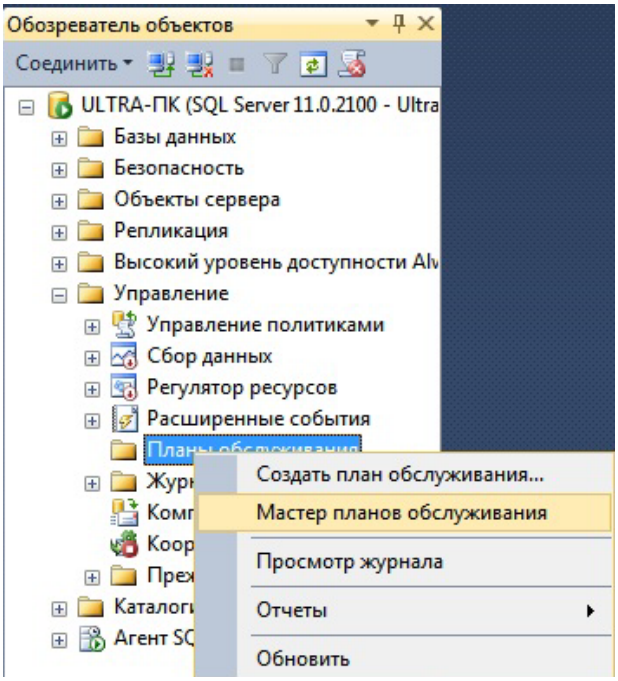

Рис. 12 – Порядок вызова «Мастер планов обслуживания»

В запустившемся мастере планов обслуживания на странице приветствия нажимаем «Далее» (Next) и в следующем окне вводим имя и описание нового плана. Затем необходимо определиться с расписанием, по которому будет выполняться данный план обслуживания. Для этого установим переключатель на «Единое расписание для всего плана или без расписания» (Single schedule for the entire plan ore no schedule) и нажмем «Изменить…» (Change…) для назначения расписания. (Рис. 13)

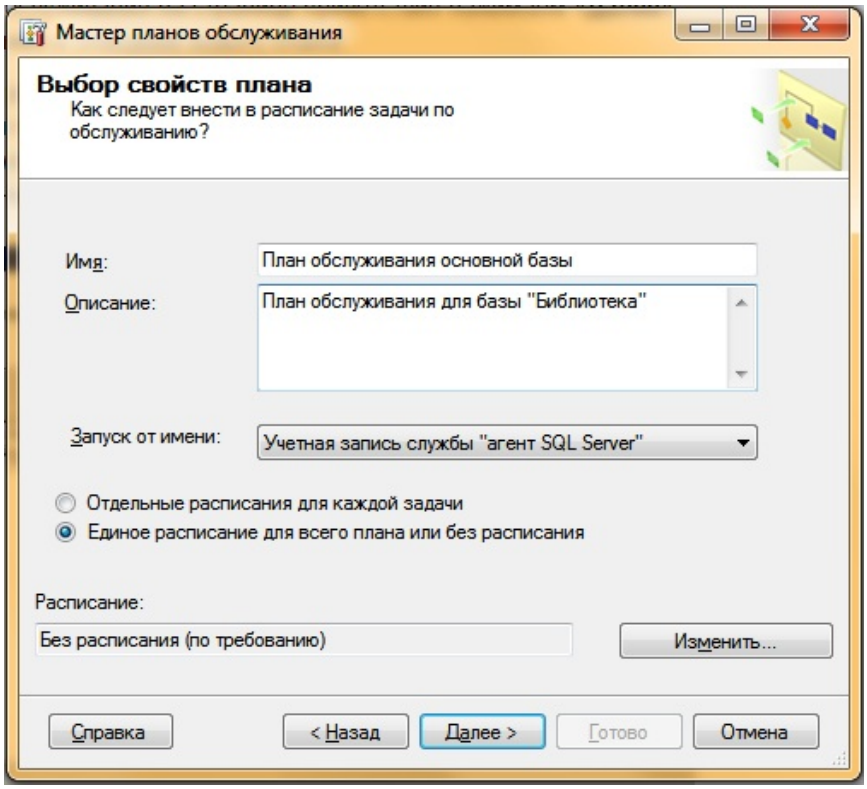

Рис. 13 – Окно «Мастер планов обслуживания»

Откроется окно «Создание расписания задания». Здесь зададим те параметры, согласно которым должен выполняться план обслуживания и нажмем «ОК». (Рис. 14)

В данном примере это:

1. Выполняется — «Еженедельно» (Occurs — Weekly);
2. Повторяется каждые — «1 нед.» в «Воскресенье» (Recurs every: 1 week(s) on Sunday);
3. Выполняться один раз в день в: — «2:00:00» (Occurs once at: «2:00:00»)

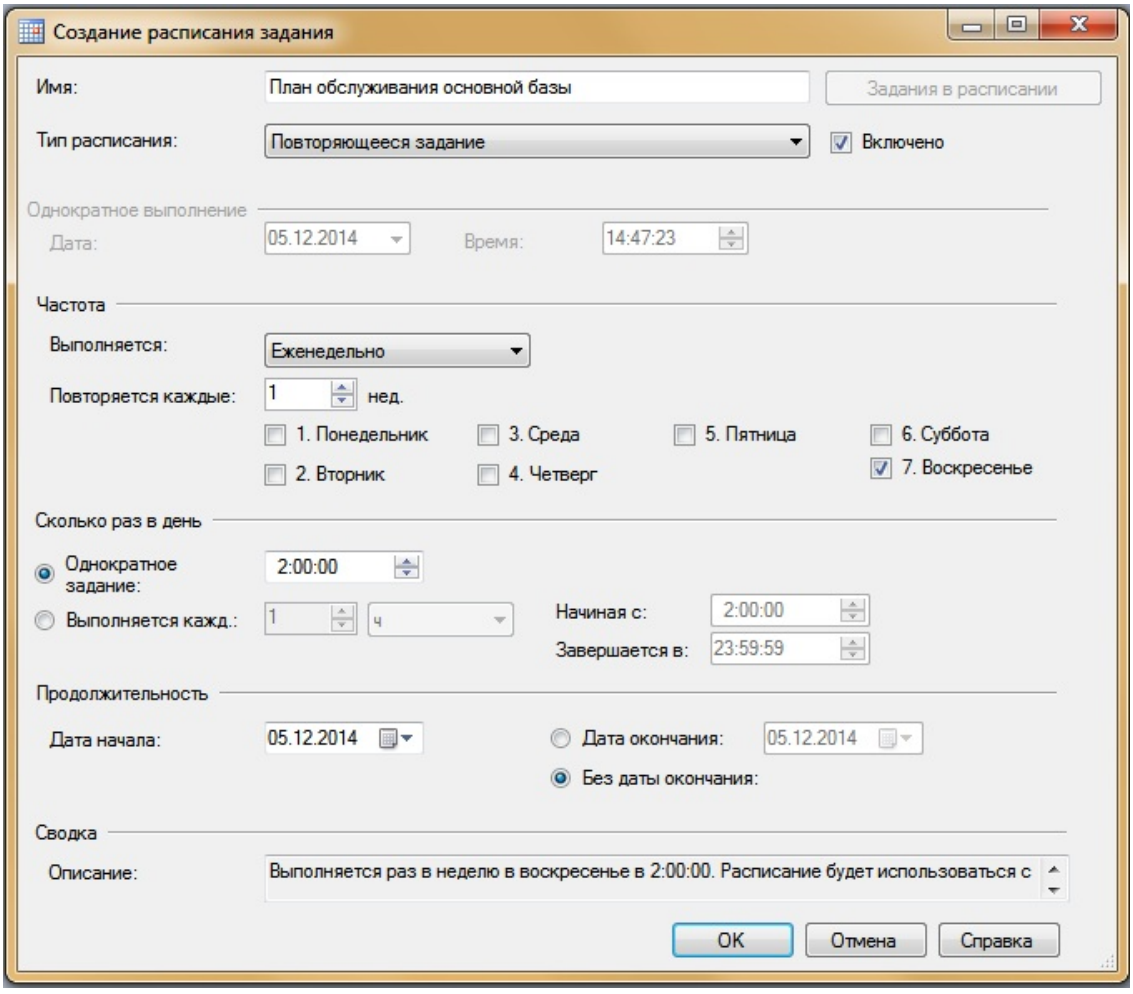

Рис. 14 – Окно «Создание расписания задания»

Еще раз убедимся, что расписание задано верно, и нажмем «Далее» (Next).

В следующем окне (Рис. 15) выберем те задачи, которые будет выполнять наш план обслуживания. В данном примере это:

1. Проверка целостности базы данных (Check Database Integrity);
2. Резервное копирование базы данных (полное) (The Back Up Database (Full));

Заметьте, что для каждой задачи приводится ее краткое описание в поле снизу. Выбрав необходимые задачи, жмем «Далее» (Next).

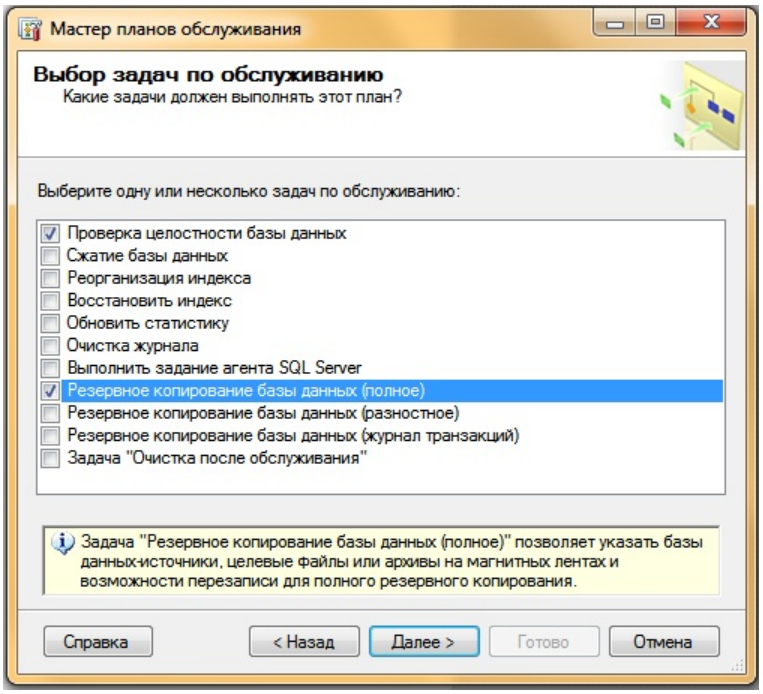

Рис. 15 – Выбор задач плана обслуживания

Теперь необходимо задать порядок выполнения задач, используя кнопки «Вверх…» (Move Up) и «Вниз…» (Move Down). Установив порядок, жмем «Далее» (Next). (Рис. 16)

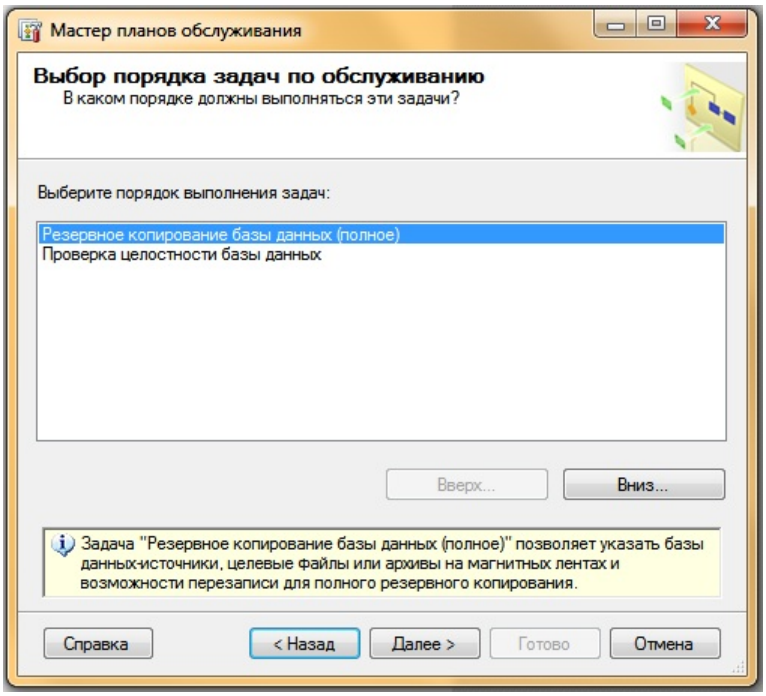

Рис. 16 – Задание порядка выполнения задач

В открывшемся окне требуется задать параметры для каждой задачи в плане. Первая задача в нашем списке — это «Копирование БД (полное)» (Back Up Database (Full)).

Прежде всего необходимо выбрать базы данных для резервного копирования, нажав на кнопку выбора списка «Определенные базы данных» (Select one ore more). Выбрав необходимые для резервного копирования базы данных, нажимаем «ОК». (Рис. 17)

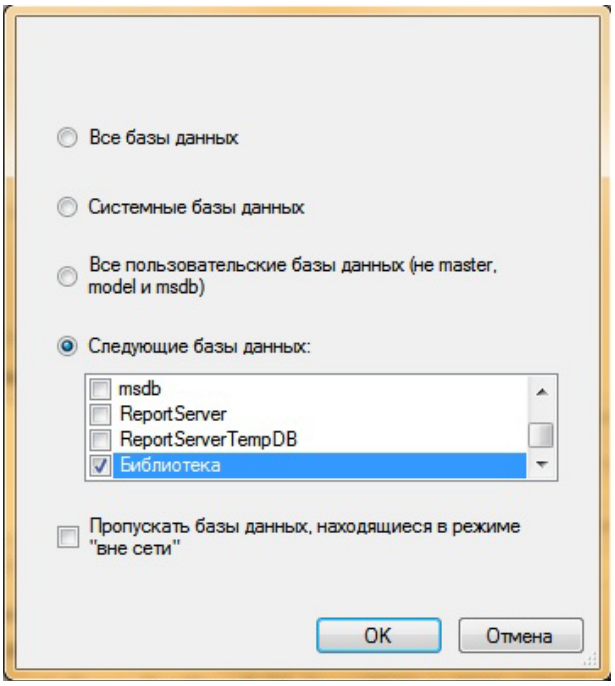

Рис. 17 – Выбор базы данных для резервного копирования

Ниже зададим размещение и срок хранения резервных копий, а также установим дополнительные параметры:

1. Если установить переключатель «Создать файл резервной копии для каждой базы данных» (Create a backup file for every database), то при выполнении задания в выбранной директории будет создаваться несколько файлов резервных копий с именами, соответствующими названиям баз данных. Ну а установка флага «Создавать вложенный каталог для каждой базы данных» (Create a sub-directory for each database) разложит файлы по отдельным папкам. Обратите внимание, что необходимо оставить заполненным расширение файла резервной копии. (Рис. 18)

2. Установка флага «Срок действия резервного набора данных истекает» (Backup set will expire) указывает SQL-серверу, когда этот набор может быть перезаписан без явного пропуска проверки на истечение срока.
3. Для наибольшей надежности, можно установить флаг «Проверять целостность резервной копии» (Verify backup integrity).
4. Также рекомендую выбрать режим «Сжимать резервные копии» (Compress backup) для экономии дискового пространства.

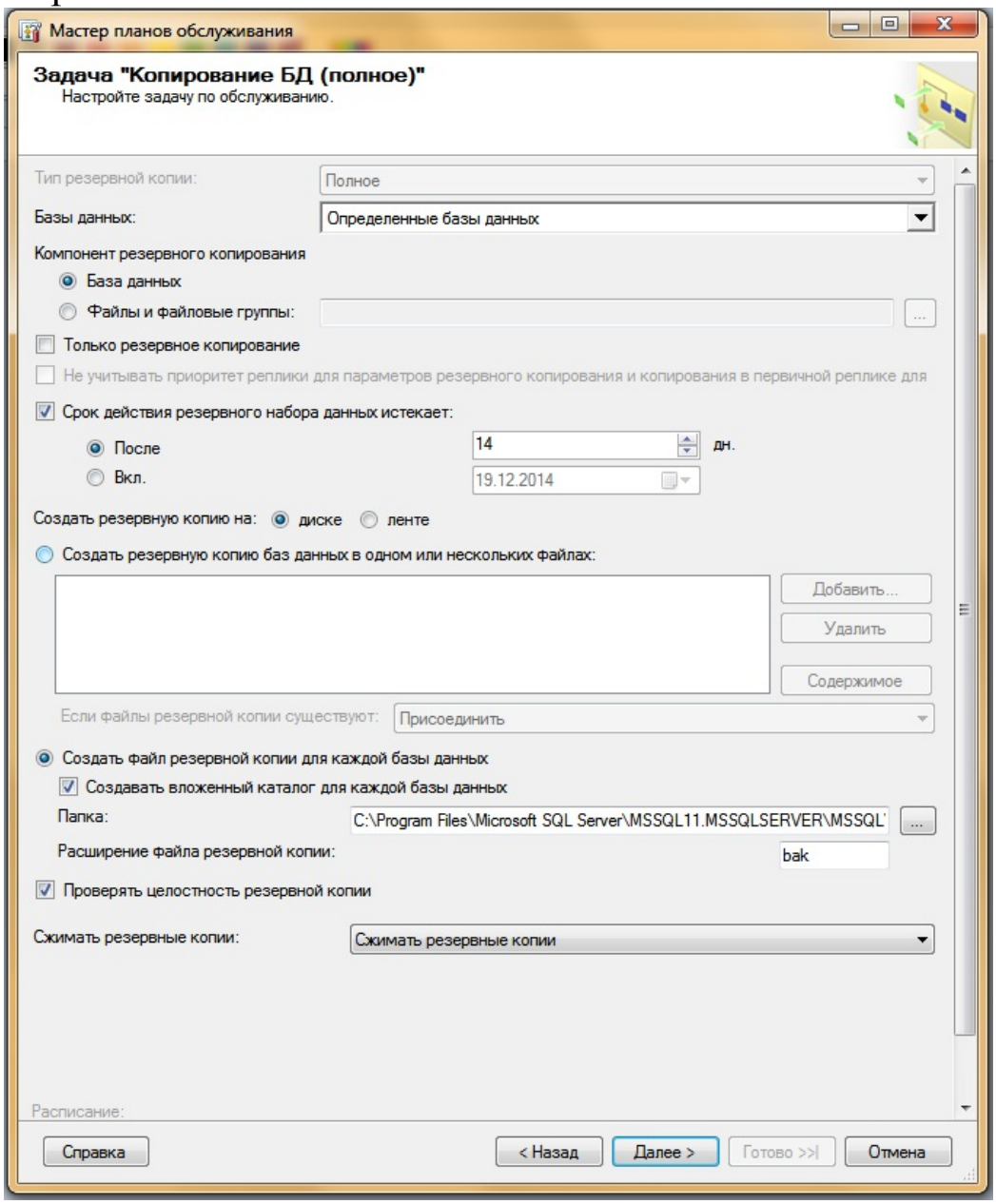

Рис. 18 – Установка параметров резервных копий

Если дисковое пространство ограничено, можно также выбрать один файл для хранения резервной копии, который будет перезаписываться после каждого выполнения плана обслуживания. Для этого установим соответствующий переключатель на «Создать резервную копию баз данных в одном или нескольких файлах» и укажем соответствующее имя файла (будьте внимательны, файл резервной копии следует задавать с расширением. bak), а также выберем режим «Перезаписать» в случае, если файлы резервной копии существуют. Определившись с настройками жмем «Далее» (Next). (Рис. 19)

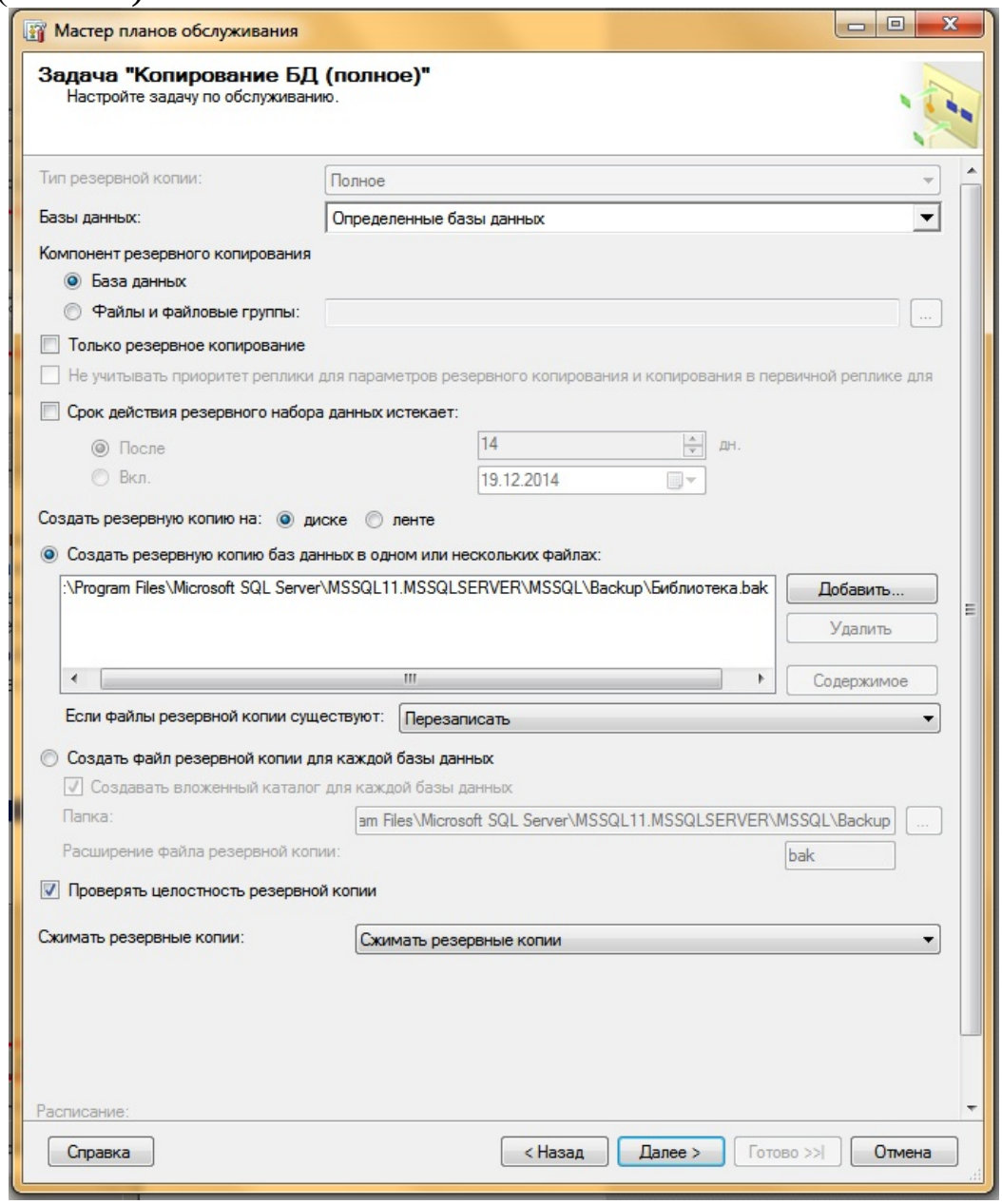

Рис. 19 – Установка параметров резервных копий

Теперь очередь задачи «Проверка целостности базы данных» (Database Check Integrity). Для нее всего лишь необходимо выбрать базу данных. В нашем примере это все та же база данных, что и на предыдущем шаге. Определившись с базами, жмем «Далее» (Next). (Рис. 20)

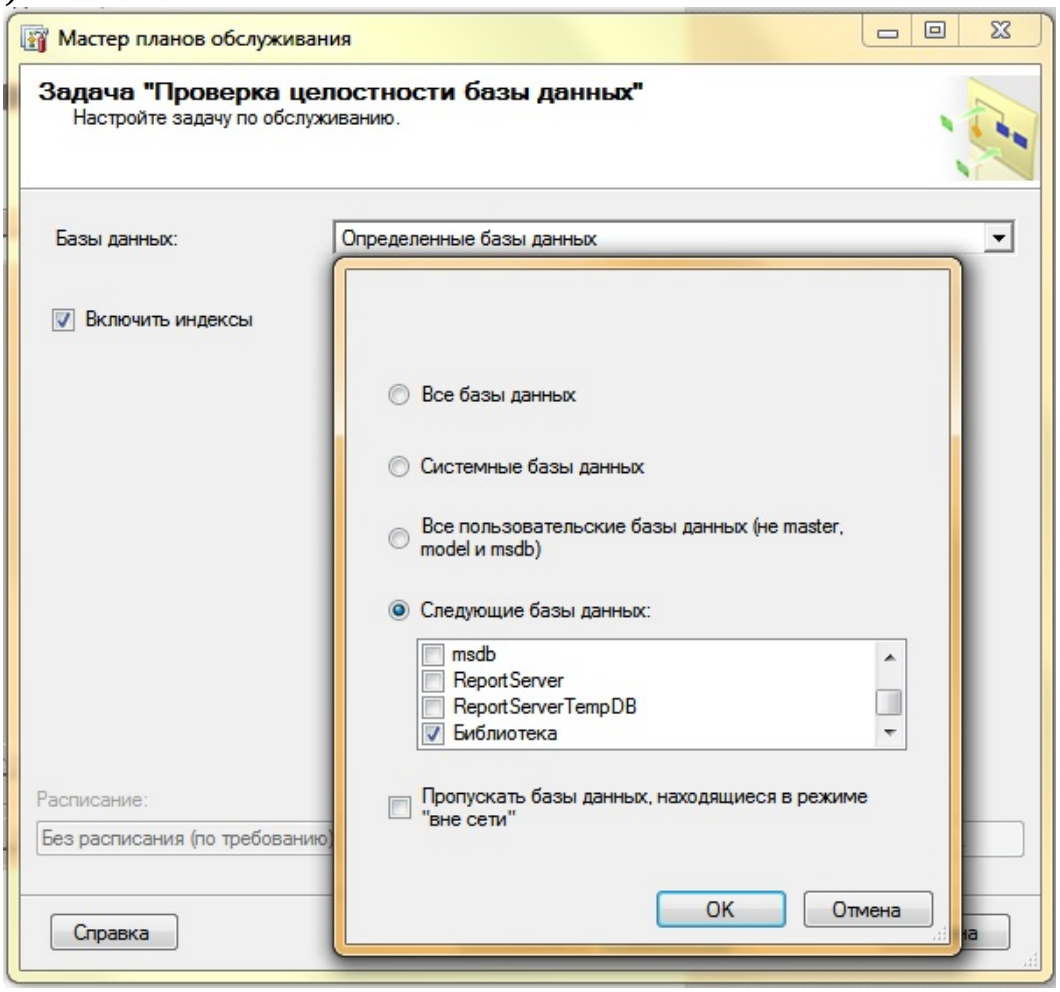

Рис. 20 – Проверка целостности базы данных

В следующем окне возможно выбрать директорию, куда будет сохраняться лог выполнения задания, а также указать оператора SQL Server для отправки отчета по электронной почте. Задав параметры, снова жмем «Далее». (Рис. 21)

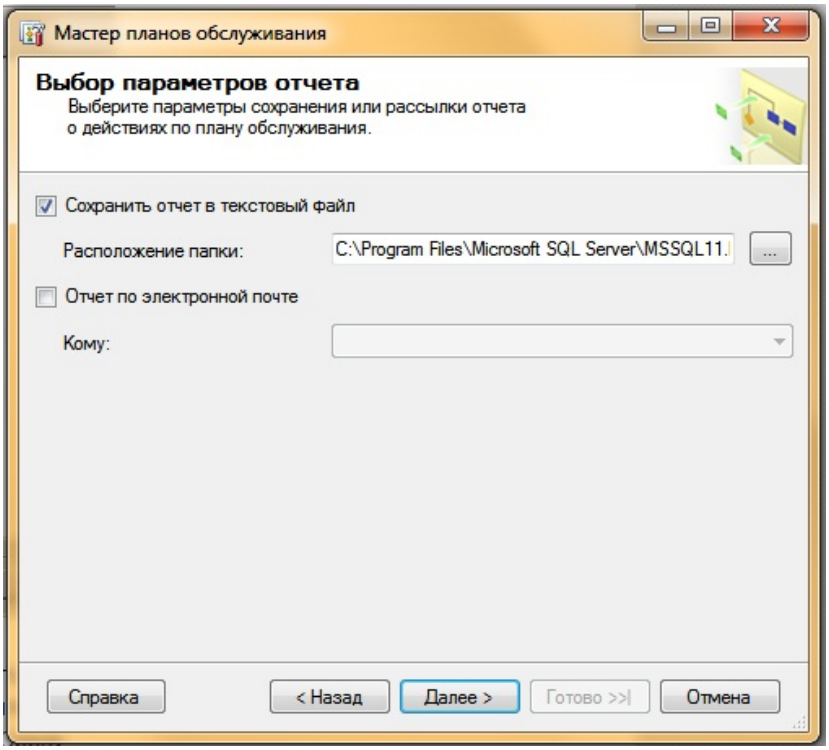

Рис. 21 – Возможность отправки отчета по электронной почте

Проверим еще раз все настройки плана обслуживания, и если все верно, нажимаем «Готово» (Finish). (Рис. 22)

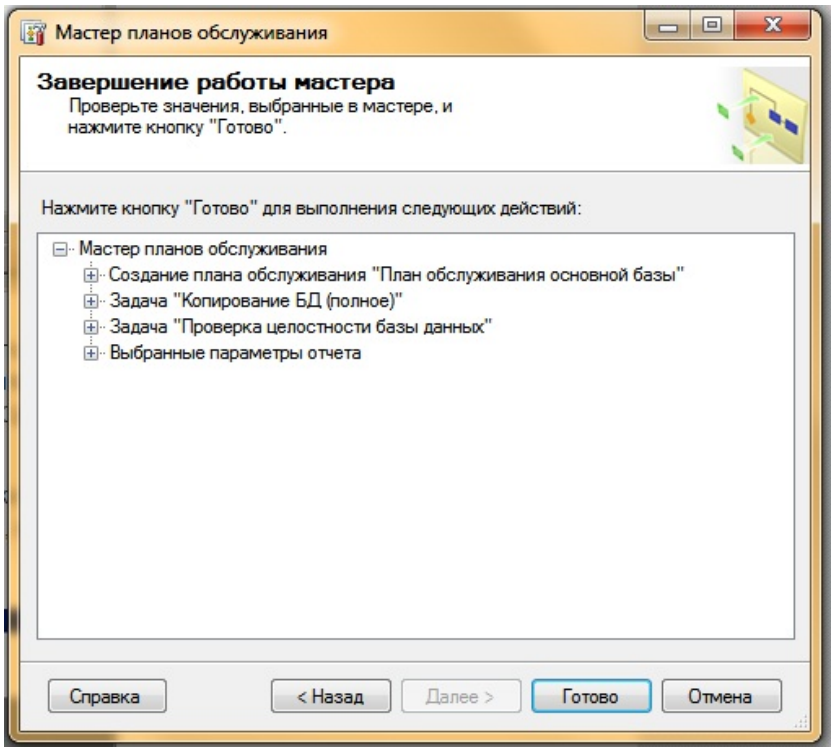

Рис. 22 – Проверка настроек плана обслуживания

Мастер начнет построение плана обслуживания. Если мастер не обнаружит ошибок, то увидим сообщение об успешном построении плана. В противном случае необходимо устранить ошибки и повторить процедуру снова. Закроем окно, нажав «Закрыть» (Close) и перейдем в Среду Microsoft SQL Server Management Studio.

Здесь, раскрыв вкладку «Планы обслуживания» (Maintenance Plans) увидим наш только что созданный план. Чтобы проверить его работу, кликнем по нему правой кнопкой мыши, и в контекстном меню выберем пункт «Выполнить» (Execute). (Рис. 23)

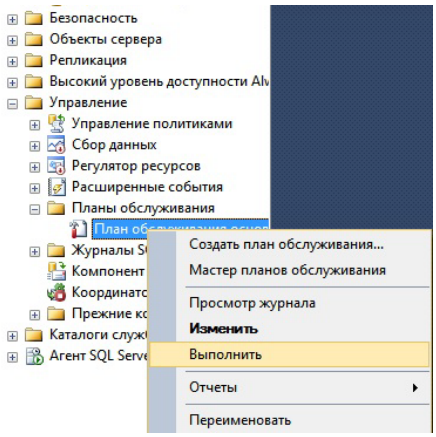

Рис. 23 – Отображение плана обслуживания в обозревателе объектов

После чего запустится окно выполнения плана обслуживания, в котором, спустя необходимое количество времени, должно появиться сообщение об успешном выполнении. (Рис. 24)

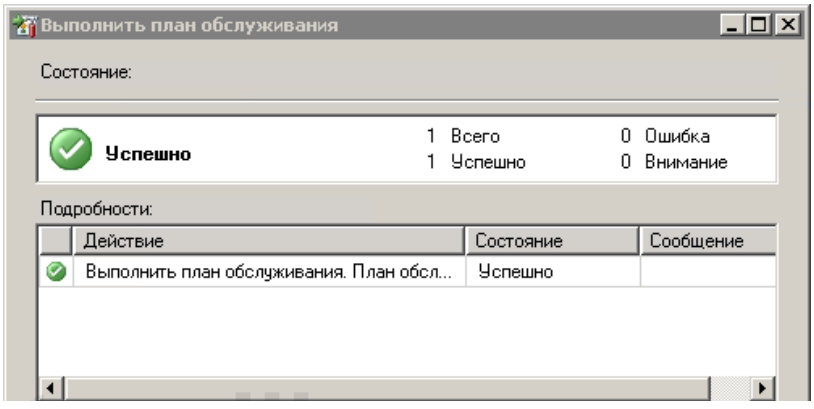

Рис. 24 – Успешное выполнение плана обслуживания.

А в соответствующих директориях должны появиться файл резервной копии и файл лога выполнения плана. (Рис. 25)

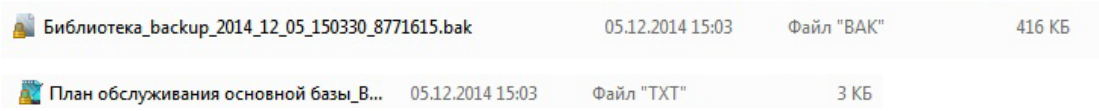

Рис. 25 – Файл резервной копии и файл лога

Открыв, файл лога, вы должны увидеть примерно следующее:

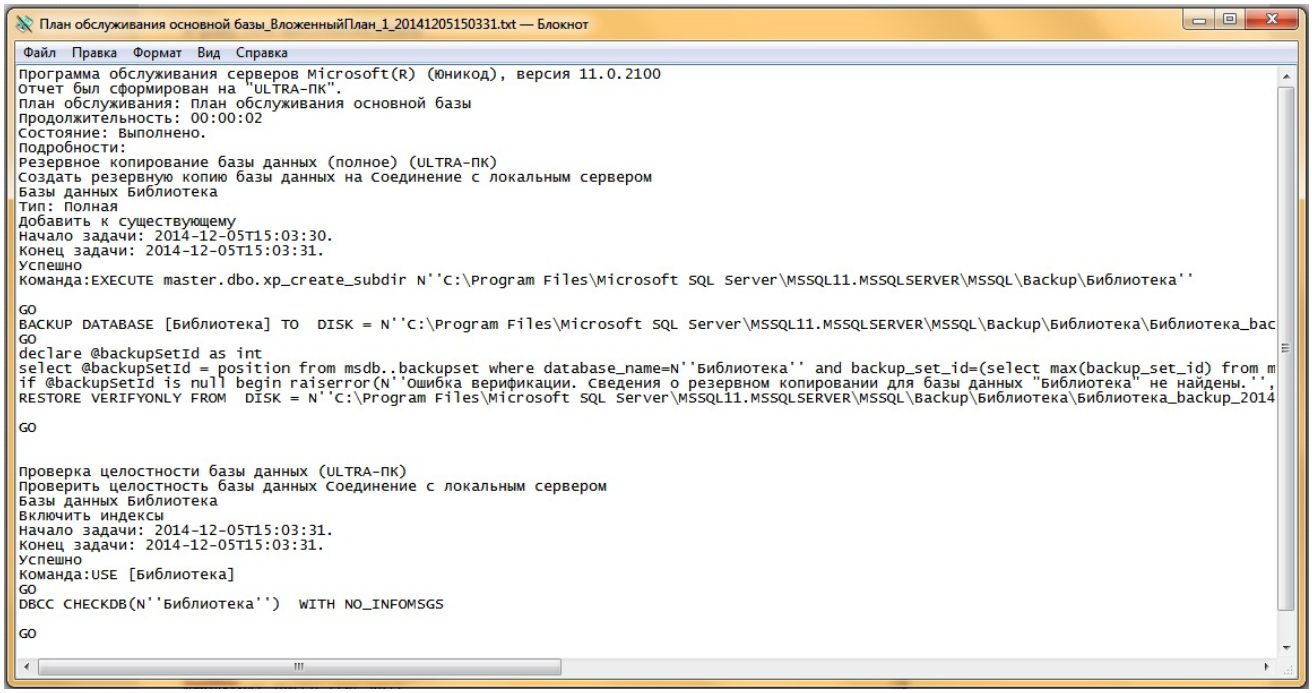

Рис. 26 – План обслуживания базы данных «Библиотека»

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Опишите архитектуру безопасности.
2. Что такое проверка подлинности?
3. Какие методы авторизации существуют в MSSQL сервере?
4. Что такое роли сервера баз данных?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Все о SQL и клиент/серверных технологиях [электронный ресурс] http://sql.ru
2. Администрирование баз данных [электронный ресурс] http://www.firebirdsql.org/manual/ru/migration-mssql-db-admin-ru.html
3. Прозрачное шифрование баз данных в Microsoft SQL Server 2008 http://rsdn.ru/article/db/liberman.xml
4. Шифрование в базах данных SQL Server http://www.itshop.ru/Shifrovanie-v-bazah-dannyh-SQL-Server/l9i36233