министерство НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
Российской федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Тюменский государственный университет»

институт математики и компьютерных наук

Кафедра программного обеспечения

Курсовая работа по направлению

«Математическое обеспечение и администрирование

информационных систем»

на тему «Разработка ПО для резервного копирования и восстановления баз данных MS SQL Server»

Выполнил: студент 3 курса

22 МОиАИС 184-2 группы

Поляков И.А.

Научный руководитель:

доцент кафедры ПО

Ялдыгин В.Б.

Тюмень 2021

Оглавление

[ВВЕДЕНИЕ 3](#_Toc86074912)

[ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 4](#_Toc86074913)

[1.1. Основные сведения о резервном копировании и восстановлении баз данных MS SQL Server 4](#_Toc86074914)

[1.2. Типы резервного копирования SQL Server 5](#_Toc86074915)

[1.2.1. Полное копирование базы данных 5](#_Toc86074916)

[1.2.2. Дифференцированное резервное копирование 8](#_Toc86074917)

[1.2.3. Резервное копирование протокола транзакции 8](#_Toc86074918)

[ГЛАВА 2. СОЗДАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕЗЕРВНОГО КОПИРОВАНИЕ И ВОССТАНОВЛЕНИЯ БАЗЫ ДАННЫХ 10](#_Toc86074919)

[ГЛАВА 3. ГРАФИЧЕСКОЕ ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 20](#_Toc86074920)

[ГЛАВА 4. ТЕСТИРОВАНИЕ РАБОТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 22](#_Toc86074921)

[ЗАКЛЮЧЕНИЕ 27](#_Toc86074922)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 28](#_Toc86074923)

# ВВЕДЕНИЕ

В настоящее время безопасное хранение данных представляет одну из основных проблем информационной безопасности. Согласно докладу о рисках Всемирного экономического форума 2021 года, утеря и утечка данных являются четвертым и пятым глобальными рисками, с которыми сталкивается каждая организация. По своей значимости эти риски приравниваются к экологическим проблемам. В официальном ежегодном отчете за 2019 год, опубликованном Cybersecurity Ventures, сообщается, что атаки хакеров во всём мире происходят каждые 11 секунд. Специалисты компании InfoWatch в конце года рассказали Известиям, что за прошлый год в сеть утекло более 14 млрд. конфиденциальных записей. Рост числа утечек во всём мире по сравнению с 2018 годом увеличился на 10%, в России – более чем на 40%.

Их этого можно сделать вывод, что хранение данных, должно обеспечивать их целостность. Большинство данных хранятся в базах, так как это является удобным способом взаимодействия с ними. Одной из основных технологий баз данных является Sql Server, который является системой управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка. Для сохранения целостности баз данных используют такие технологии как их резервное копирование и восстановление.

# ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

## 1.1. Основные сведения о резервном копировании и восстановлении баз данных MS SQL Server

Резервное копирование базы данных — это процесс загрузки данных (из базы данных, журнала транзакций или файла) на устройства резервного копирования, которые создаются и обслуживаются системой. Устройство резервного копирования может быть дисковым файлом или магнитной лентой. Компонент Database Engine обеспечивает статическое и динамическое резервное копирование.

Статическое резервное копирование означает, что во время процесса резервного копирования только один активный сеанс, поддерживаемый системой, является сеансом, создающим резервную копию. Другими словами, во время копирования никакие пользовательские процессы не допускаются.

Динамическое резервное копирование означает, что копирование базы данных может выполняться без остановки сервера базы данных, удаления пользователей и даже без закрытия файлов (пользователи даже не узнают, что идет процесс резервного копирования).

К одной из важнейших задач администратора базы данных можно отнести постоянную защиту целостности баз данных, а также поддержку возможности быстрого восстановления в случае сбоя. Именно из-за этого тактика резервного копирования и восстановления важны. Это поможет избежать чрезвычайных ситуаций.

Еще одна важная обязанность администратора – это доступность базы данных в тот момент, когда это необходимо. Ответственность за восстановление базы данных в случае, если она была удалена, повреждена или сброшена, лежит на администраторе базы данных.

Человек, отвечающий за базу данных, должен быть готов к её аварийному восстановлению. Это достигается путем постоянного тестирования резервного копирования и восстановления SQL Server. В таком случае можно избежать потери данных. Также к обязанностям администратора базы данных входит и её защита от различных сбоев.

## 1.2. Типы резервного копирования SQL Server

Компонент Database Engine предоставляет четыре различных метода резервного копирования:

1. Полное копирование базы данных;
2. Дифференцированное резервное копирование;
3. Резервное копирование протокола транзакции;
4. Резервное копирование файла или файловой группы. Частичная резервная копия.

## 1.2.1. Полное копирование базы данных

Полная копия базы данных фиксирует состояние базы данных, каким оно было при запуске копии. Во время полного копирования базы данных система копирует данные, а также схему всех таблиц в базе данных и соответствующие файловые структуры. Если полная копия базы данных выполняется динамически, система базы данных записывает все действия, происходящие во время процесса резервного копирования. Поэтому даже все неподтвержденные транзакции в журнале транзакций будут записаны на устройство резервного копирования.

Все типы резервных копий можно выполнять с помощью двух операторов Transact-SQL:

1. backup database;
2. backup log.

Компонент Database Engine позволяет копировать базы данных, журналы транзакций и файлы на устройства резервного копирования, такие как:

1. Диск;

2. Магнитная лента.

Дисковые файлы чаще всего используются для хранения резервных копий. Диск с копией может быть расположен на локальном жестком диске на сервере или на удаленном диске в общей сетевой папке. Компонент Database Engine позволяет добавлять новые копии в файл, который уже содержит копии той же или другой базы данных. Когда вы добавляете новый набор копий на существующий носитель, предыдущее содержимое носителя остается нетронутым, и новая копия записывается после последней копии на этом носителе. (Набор копий включает все сохраненные данные объекта, которые вы выбираете для копирования.) По умолчанию компонент Database Engine всегда добавляет новые копии в файлы на диске.

Магнитная лента обычно используется в качестве устройства хранения резервных копий так же, как и дисковый накопитель. Однако при резервном копировании на ленту ленточное устройство должно быть локально подключено к системе. Преимущество ленточных устройств перед дисковыми - простота администрирования и эксплуатации.

Сценарий восстановления — SQL Server — это процесс восстановления данных из одной или нескольких резервных копий и возврата базы данных в исходное состояние. Поддерживаемые сценарии восстановления зависят от модели восстановления базы данных и выпуска SQL Server.

Для большинства сценариев восстановления необходимо применить резервную копию журнала транзакций и позволить ядру СУБД SQL Server запустить процесс восстановления, чтобы перевести базу данных в оперативный режим. Восстановление — это процесс, используемый SQL Server для запуска каждой базы данных в транзакционно согласованном (чистом) состоянии.

В случае аварийного переключения или другого неясного завершения работы базы данных могут оставаться в состоянии, в котором некоторые изменения не были записаны из буферного кеша в файлы данных, и в файлах данных могут быть некоторые изменения, вызванные незавершенными транзакциями. Когда вы запускаете экземпляр SQL Server, выполняется трехэтапное восстановление каждой базы данных на основе самой последней контрольной точки в базе данных:

1. На этапе анализа анализируется журнал транзакций для определения последней контрольной точки и создается таблица грязных страниц (DTP) и таблица активных транзакций (TAT). TGS содержит записи о страницах, которые были грязными при выключении базы данных. TAT содержит записи транзакций, которые все еще были активны, когда база данных была аварийно завершена.
2. На этапе повтора повторяются все зарегистрированные изменения, которые, возможно, не были записаны в файлы данных, когда база данных была закрыта. Минимальный порядковый номер журнала (minLSN), необходимый для успешного восстановления всей базы данных, находится в TGS и отмечает начало операций восстановления, необходимых для всех грязных страниц. На этом этапе компонент SQL Server Database Engine записывает на диск все грязные страницы, принадлежащие зафиксированным транзакциям.
3. На этапе отката выполняется откат ожидающих транзакций, обнаруженных в TAT, чтобы гарантировать сохранение целостности базы данных. После отката база данных переводится в оперативный режим, и больше нельзя применять резервные копии журнала транзакций.

Цель полного восстановления – восстановить всю базу данных. База данных отключена во время периода восстановления. Перед тем, как какая-либо часть базы данных перейдет в оперативный режим, все данные восстанавливаются до точки согласованности, когда все части базы данных находятся в один и тот же момент времени и отсутствуют незавершенные транзакции.

В режиме полного восстановления после восстановления резервных копий данных необходимо восстановить все последующие резервные копии журналов транзакций, а затем и саму базу данных. База данных может быть восстановлена до определенной точки восстановления в одной из этих резервных копий журнала. Этой точкой восстановления может быть указанная дата и время, помеченная транзакция или порядковый номер журнала (LSN).

При восстановлении базы данных, особенно при использовании модели восстановления с полным или неполным протоколированием, используйте одну последовательность восстановления. Последовательность восстановления состоит из одной или нескольких операций восстановления, которые перемещают данные в одну или несколько фаз восстановления.

## 1.2.2. Дифференцированное резервное копирование

Данное копирование отвечает за копирование данных, которые появились с момента последней полной резервной копии.

Дифференцированное (разностное) резервное копирование используют вместе с полной резервной копией. Это объясняется тем, что для восстановления дифференциальной копии необходима полная резервная копия.

Чаще всего при использовании разностного резервного копирования придерживаются следующего сценария: полное резервное копирование раз в N дней, дифференциальное каждые M часов. Если же объем данных большой, то такой тип копирования может быть неудобен. Для примера: полная резервная копия весит 500 GB. Дифференциальная спустя час работы 5 GB. В итоге, спустя сутки такая копия будет весить 120 GB.

## 1.2.3. Резервное копирование протокола транзакции

Такой тип резервного копирования делает копии всех транзакций, которые были произведены с момента последнего резервного копирования. После этого журнал транзакций урезается для освобождения дискового пространства.

Такое резервное копирование является инкрементальным (связанным). Из-за этого для восстановления базы данных потребуется вся цепочка резервных копий. То есть, полная и все последующие инкрементальные копии журнала транзакций.

## 1.2.4. Резервное копирование файла или файловой группы. Частичная резервная копия

Partical backup (частичное резервное копирование) – это тип резервной копии, который используется для снятия копии с read-only файловых групп. Используется редко.

Резервное копирование файлов или файловых групп используется для создания резервных копий определенных файлов или файловых групп.

# ГЛАВА 2. СОЗДАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕЗЕРВНОГО КОПИРОВАНИЕ И ВОССТАНОВЛЕНИЯ БАЗЫ ДАННЫХ

## 2.1 Используемые технологии

При создании программного обеспечения для резервного копирования и восстановления базы данных SQL Server, использовалась среда разработки Microsoft Visual Studio 2022, язык программирования C# на Windows Forms.

## 2.2 Общие сведения об SMO

SQL ServerУправляющие объекты (SMO) — это объекты, предназначенные для программного управления Microsoft SQL Server . Объекты SMO можно использовать для создания специализированных приложений управления SQL Server . Хотя среда SQL Server Management Studio является мощным и универсальным приложением для управления SQL Server, иногда удобнее работать с приложением SMO.

К примеру, для удовлетворения потребностей новых пользователей, а также с целью сокращения издержек на подготовку персонала может возникнуть необходимость упрощения пользовательских приложений, выполняющих задачи управления SQL Server . Может возникнуть необходимость в создании специализированных баз данных SQL Server или в создании приложения для формирования индексов и наблюдения за их эффективностью. Кроме того, приложение SMO можно использовать для бесшовной интеграции аппаратных или программных компонентов от независимых поставщиков в приложение управления базами данных.

Поскольку модель объектов SMO совместима с SQL Server 2005 (9.x) и более поздними версиями, облегчается работа с продуктами различных версий.

## 2.3 Классы модели объектов SMO

Классы модели объектов SMO подразделяются на две категории: классы экземпляров и служебные классы.

**Классы экземпляров**

Классы экземпляров представляют объекты SQL Server, такие как серверы, базы данных, таблицы, триггеры и хранимые процедуры. Класс [ServerConnection](https://docs.microsoft.com/ru-ru/dotnet/api/microsoft.sqlserver.management.common.serverconnection) используется для установления соединения с экземпляром SQL Server и для управления режимом сбора направляемых ему команд.

Объекты экземпляра модели SMO образуют иерархию, которая представляет иерархию сервера баз данных. Вверху размещаются экземпляры SQL Server, под ними располагаются базы данных, а далее — таблицы, столбцы, триггеры и т. д. Если логика допускает возможность существования связи «один родитель ко многим потомкам», например в таблице с одним или несколькими столбцами, тогда потомок представляется коллекцией объектов. В других случаях потомок представляется одним объектом.

**Служебные классы**

Служебные классы — это группа объектов, созданных явным образом для выполнения конкретных задач. В соответствии со своими функциями они разделяются на различные иерархии объектов.

Класс Transfer. Используется для передачи другой базе данных схемы и данных.

Классы Backup и Restore. Используются для создания резервной копии и восстановления баз данных.

Класс Scripter. Используется для формирования файлов скриптов, предназначенных для повторного создания объектов и их зависимостей.

## 2.4 Функции SMO

**Оптимизированная производительность**

Архитектура объектов SMO эффективна с точки зрения памяти, поскольку объекты создаются только частично, а на сервере запрашиваются минимальные сведения о свойствах. Полное создание объектов откладывается до того времени, когда выполняется явная ссылка на объект. Объект полностью создается тогда, когда требуется свойство, не входящее в набор изначально полученных свойств, или когда вызывается метод, требующий такое свойство. Переход от частично полностью созданных объектов к полностью созданным незаметен для пользователя. Кроме того, некоторые свойства, использующие большие объемы памяти, так и не считываются — если только не выполняется явное обращение к подобному свойству. Примером сказанному может служить свойство [Size](https://docs.microsoft.com/ru-ru/dotnet/api/microsoft.sqlserver.management.smo.database.size) свойства объекта [Database](https://docs.microsoft.com/ru-ru/dotnet/api/microsoft.sqlserver.management.smo.database). Однако при частичном создании объектов возникает большее количество обменов данными через сеть, и потому с точки зрения производительности этот метод не обязательно будет наиболее эффективным для приложения.

Созданием экземпляров объектов можно управлять в соответствии с особенностями системной среды. При использовании метода отложенного создания экземпляров объем памяти, необходимый для работы приложения, сводится к минимуму, но на этапе обращения к свойствам может возникнуть большое число запросов к серверу.

Классы экземпляров (объекты, представляющие реальные объекты баз данных) могут существовать на трех уровнях создания. Эти уровни: минимальный экземпляр (лишь минимально необходимые свойства считываются в одном блоке), частично созданный экземпляр (все свойства, использующие относительно большие объемы памяти, считываются в одном блоке) и полностью созданный экземпляр. Традиционные состояния создания экземпляров — несозданный и полностью созданный. Состояние «частично созданный» повышает эффективность, так как частично созданный объект не содержит значений для полного набора свойств объекта. Состояние «частично созданный» применяется по умолчанию для объектов, на которые нет непосредственных ссылок. При обращении к одному из этих свойств возникает ошибка, которая инициирует создание полного экземпляра объекта.

**Выполнение после сбора**

Обычным методом выполнения является непосредственное выполнение. Инструкции передаются экземпляру SQL Server непосредственно по получении. Выполнение после сбора является альтернативным методом.

Выполнение после сбора дает возможность собирать пакеты Transact-SQL , которые обычно выполняются. В результате программист модели объектов SMO может отложить скрипт, сохранить его для выполнения в более поздний период или выполнить предварительный просмотр скрипта для конечного пользователя. Так, инструкции **create database**, **create table** и **create index** могут быть переданы в одном пакете и затем выполнены как три последовательных шага. Этой функцией управляет пользователь с помощью объекта [Server](https://docs.microsoft.com/ru-ru/dotnet/api/microsoft.sqlserver.management.smo.server.-ctor).

**Поставщик WMI**

Объекты поставщика WMI помещаются в объекты SMO. В результате программист модели SMO получает простую модель объектов, весьма напоминающую классы SMO. Однако в этом случае программист не должен понимать модель программирования, представленную пространством имен, и особенности организации поставщика WMI SQL Server. Поставщик WMI позволяет конфигурировать службы SQL Server, псевдонимы, а также сетевые библиотеки клиентов и серверов.

**Создание скриптов**

В модели объектов SMO средства для работы со скриптами были улучшены и переданы в класс **Scripter** . С помощью класса **Scripter** разработчик может находить зависимости, выявлять связи между объектами и осуществлять манипуляции с иерархией зависимостей. Главным объектом, обеспечивающим работу со скриптами, является объект **Scripter** . Существует несколько поддерживающих объектов, которые осуществляют обработку зависимостей, а также реагируют на события состояния и на события ошибок.

Объект **Scripter** поддерживает следующие дополнительные параметры создания скриптов:

простой однофазный метод (скрипт создается за один этап);

Продвинутые 3-фазные сценарии (создание скрипта выполняется в три этапа; обнаружение зависимостей, создание списка, создание скрипта)

двусторонний поиск зависимостей (позволяет выявлять зависимости, или зависимые элементы);

реагирование на события состояния;

реагирование на события ошибок.

**Уникальные имена ресурсов**

Ключевая концепция при использовании библиотеки объектов SMO — уникальные имена ресурсов (URN). Синтаксис URN напоминает синтаксис XPath. XPath — это путь иерархии, используемый для указания объекта, в котором каждый уровень имеет квалификаторы и функции. В модели объектов SMO URN имеет два элемента — путь, а также именование атрибутов, обладающее ограниченными функциональными возможностями. Путь используется для указания местоположения объекта, тогда как именование атрибутов дает возможность осуществлять частичную фильтрацию.

## 2.5 Реализованные классы и методы

При разработке приложения были реализованы следующие классы:  
Form1, Form2, dataBase.

Класс Form1 (Приложение 1):

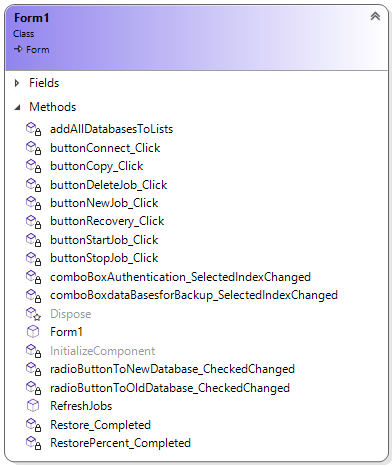


Диаграмма 1. Класс Form1

Методы класса Form1:

* public Form1() – Конструктор для инициализации form2 класса Form2 и заполнения списка серверов и методом аутентификации.
* void addAllDatabasesToLists() – Метод для заполнения списка баз данных сервера.
* private void buttonConnect\_Click(object sender, EventArgs e)- метод подключения к серверу.
* private void buttonCopy\_Click(object sender, EventArgs e)- метод создания бекапа базы данных.
* private void RestorePercent\_Completed(object sender, PercentCompleteEventArgs e) – метод для отображения прогресса восстановления.
* void Restore\_Completed(object sender, ServerMessageEventArgs e)- метод для отлова ошибки в случае некорректного восстановления.
* private void buttonRecovery\_Click(object sender, EventArgs e)- метод восстановления базы данных.
* private void comboBoxdataBasesforBackup\_SelectedIndexChanged(object sender, EventArgs e)- метод для вставки названия выбранной базы данных.
* private void radioButtonToOldDatabase\_CheckedChanged(object sender, EventArgs e)- метод контроля восстановления в новую или существующую базу данных.
* private void radioButtonToNewDatabase\_CheckedChanged(object sender, EventArgs e)- метод контроля восстановления в новую или существующую базу данных.
* private void comboBoxAuthentication\_SelectedIndexChanged(object sender, EventArgs e)-метод для контроля параметров аутентификации.
* public void RefreshJobs()- метод обновления существующих задач.
* private void buttonNewJob\_Click(object sender, EventArgs e)- метод вызова модального окна для создания новой задачи.
* private void buttonDeleteJob\_Click(object sender, EventArgs e)- метод удаления задачи
* private void buttonStartJob\_Click(object sender, EventArgs e)- метод старта задачи.
* private void buttonStopJob\_Click(object sender, EventArgs e)- метод остановки задачи.

Класс Form2 (Приложение 2):

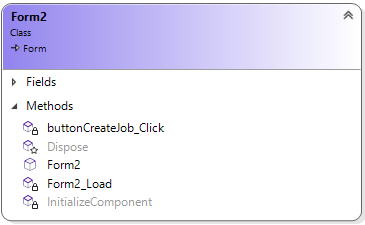


Диаграмма 2. Класс Form2

Методы класса Form1:

* public Form2()- конструктор для инициализации компонентов form2.
* private void Form2\_Load(object sender, EventArgs e)- метод загрузки form2, используется для заполнения списка базы данных при новом открытии.
* private void buttonCreateJob\_Click(object sender, EventArgs e)- метод создания задачи.

Класс DataBase (Приложение 3):

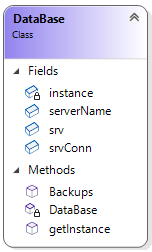


Диаграмма 3. Класс DataBase

Методы класса DataBase:

* private DataBase() { }- приватный конструктор.
* public static DataBase getInstance()- создание или возврат существующего объекта (шаблон одиночка).
* public string Backups(string NameBase, string Directory)- метод создания выражения команды бекапа.

# ГЛАВА 3. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ И ОПИСАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Для графической составляющей программного обеспечения использовался механизм Windows Forms. На Рисунке 2 представлено основное окно программного обеспечения.

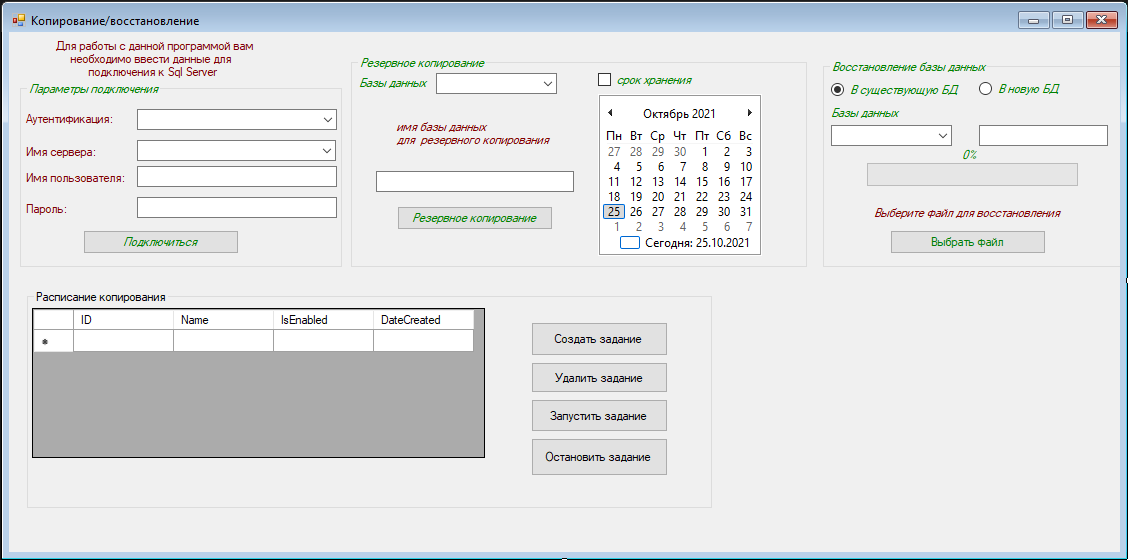


Рисунок 1. Основная форма программы

Данная форма делится на четыре блока:

1. Блок подключения к серверу;
2. Блок резервного копирования;
3. Блок восстановления базы данных;
4. Блок установки задач для бекапов Баз данных.

# ГЛАВА 4. ТЕСТИРОВАНИЕ РАБОТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

# ЗАКЛЮЧЕНИЕ

В ходе данной работы были рассмотрены основные факторы и алгоритмы, позволяющие провести резервное копирования и восстановление базы данных SQL Server. Построена структурная схема программы, по которой было реализовано программное обеспечение, позволяющее совершать резервное копирование и восстановление базы данных, не использую среду Microsoft SQL Server Management Studio.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ