Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Е. А. Блинова, Л. С. Мороз

БАЗЫ ДАННЫХ

**Учебно-методическое пособие
к выполнению курсовой работы для студентов
специальности 1-40 01 01 «Программное обеспечение
информационных технологий»**

Минск 2018

УДК 004.65(075.8)
ББК 32.973я73
Б69

Рассмотрено и рекомендовано к изданию редакционно-издательским советом Белорусского государственного технологического университета.

Р е ц е н з е н т ы :
кандидат технических наук, доцент кафедры информационных технологий автоматизированных систем УО «Белорусский государственный университет информатики и радиоэлектроники» *О. В. Герман*;
кандидат технических наук, доцент, доцент кафедры информатики и веб-дизайна УО «Белорусский государственный технологический университет» *А. А. Дятко*

Блинова, Е. А.

Б69 Базы данных : учеб.-метод. пособие к выполнению курсовой работы для студентов специальности 1-40 01 01 «Программное обеспечение информационных технологий» / Е. А. Блинова, Л. С. Мороз. – Минск : БГТУ, 2018. – 62 с.

Пособие предназначено для студентов, выполняющих курсовую работу по дисциплине «Базы данных», и содержит план работы, требования к курсовой работе, описание структуры пояснительной записки и демонстрационный пример. Результатом курсовой работы является разработанная студентом база данных с применением какой-либо технологии, используемой в современных СУБД, и приложение для демонстрации возможностей разработанной базы данных.

УДК 004.65(075.8)
ББК 32.973я73

ОГЛАВЛЕНИЕ

Введение

Целью курсовой работы является освоение навыков проектирования и администрирования базы данных. В процессе выполнения работы студент должен применить теоретические знания, полученные при изучении дисциплины «Базы данных», спроектировав базу данных, самостоятельно освоить новые технологии, применяемые в современных СУБД, и разработать приложение для демонстрации.

Задание предполагает:

– установку и настройку сервера СУБД;

– проектирование инфраструктуры базы данных;

– создание необходимых объектов;

– загрузку тестовых данных для демонстрации и проведения оптимизации;

– самостоятельное изучение и применение определенной технологии в созданной базе данных;

– самостоятельное изучение и применение резервного копирования и восстановления данных в созданной базе данных;

– разработку небольшого приложения для демонстрации.

Для успешной защиты курсовой работы студент должен:

– установить и настроить сервер СУБД;

– спроектировать инфраструктуру базы данных;

– создать необходимые объекты;

– импортировать или сгенерировать тестовые данные для демонстрации;

– исследовать и при необходимости оптимизировать структуру запросов к базе данных;

– самостоятельно изучить и применить определенную технологию в созданной базе данных;

– произвести резервное копирование и восстановление данных;

– разработать небольшое приложение для демонстрации;

– подготовить пояснительную записку к курсовому проекту.

Предлагаемое пособие содержит теоретический материал и требования к курсовому проекту (раздел 1), структуру и описание содержимого пояснительной записки (раздел 2), правила ее оформления (раздел 3), примерный список тематик областей применения баз данных и технологий (раздел 4) и демонстрационный пример (раздел 5).

1 Требования к курсовой работе

1.1 Этапы и сроки выполнения курсовой работы

Тематика курсовых работ представлена отдельным списком (см. раздел 4). Курсовая работа разрабатывается студентами в сроки, предусмотренные графиком учебного процесса. На выполнение курсовой работы в учебном плане предусмотрено 40 часов. Курсовая работа должна быть выполнена в течение 11 учебных недель со следующим распределением объемов выполнения задания (в процентах, с нарастающим итогом).

Неделя	1	2	3	4	5	6	7	8	9	10	11
%	5	15	25	35	45	55	65	75	85	95	100

Предварительно необходимо выбрать СУБД для решения задачи, область решения задачи и изучаемую технологию, для чего следует проконсультироваться с преподавателем. В задании должны быть указаны бизнес-задачи, для решения которых создается база данных. После согласования задания должен быть составлен подписанный студентом и преподавателем лист задания (в двух экземплярах) для подписи и утверждения у заведующего кафедрой.

Для консультаций по выполнению проекта установлено специальное время консультаций. В процессе консультаций у преподавателя могут возникнуть замечания, которые следует устранить до защиты работы. Во время проведения текущих аттестаций отмечается объем выполненного задания.

Законченная курсовая работа должна быть представлена для защиты на кафедру за месяц до начала сессии. При защите курсовой работы предоставляются пояснительная записка к курсовой работе и электронный носитель, содержащий пояснительную записку, SQL-скрипты, файлы базы данных и приложения. За месяц до сессии на кафедре вывешивается список дат защиты курсовых работ. Для защиты курсовой работы необходимо записаться на удобную дату.

Курсовые работы, претендующие на высокую оценку, направляются на открытую защиту, где могут присутствовать все желающие: студенты и преподаватели.

1.2 Минимальные требования к курсовой работе

Для получения положительной оценки за курсовую работу необходимо выполнить следующее:

– выбрать СУБД, область решения задачи и технологию;

– установить и настроить сервер СУБД;

– спроектировать инфраструктуру базы данных с учетом нескольких ролей пользователей;

– создать необходимые объекты базы данных, причем количество этих объектов регламентируется задачей;

– импортировать и/или сгенерировать тестовые данные для демонстрации;

– самостоятельно изучить и применить определенную технологию в созданной базе данных;

– произвести резервное копирование и восстановление данных;

– разработать небольшое приложение для демонстрации работы;

– подготовить пояснительную записку к курсовому проекту.

Требования к технологии:

– самостоятельно изучить и применить выбранную технологию в созданной базе данных;

– описать в пояснительной записке особенности применения технологии.

Требования по оптимизации:

– исследовать и при необходимости оптимизировать структуру запросов к базе данных;

– создать необходимые объекты для ускорения поиска в базе данных.

1.3 Дополнительные требования к курсовой работе

Для повышения оценки за курсовую работу необходимо:

– выбрать сложную бизнес-задачу (по согласованию с преподавателем);

– создать необходимые и достаточные объекты базы данных;

– создать план резервного копирования и восстановления данных;

– разработать полноценное приложение для демонстрации работы;

– подготовить пояснительную записку к курсовому проекту, оформленную на должном уровне.

Требования к технологии:

– самостоятельно изучить и применить выбранную технологию в созданной базе данных с использованием в приложении;

– описать в пояснительной записке особенности применения технологии в базе данных и в приложении.

2 Основные разделы пояснительной записки

Общий объем пояснительной записки курсовой работы должен составлять примерно 50–60 страниц текста, включая приложения.

2.1 Структура пояснительной записки

Пояснительная записка состоит из следующих элементов:
– титульного листа;
– задания на курсовой проект;
– содержания;
– введения;
– основной части пояснительной записки;
– заключения;
– списка использованных источников;
– графического материала;
– приложений.

2.2 Титульный лист

Титульный лист является первой страницей пояснительной записки. Номер страницы на титульном листе не указывается. Пример оформления приведен в приложении А.

2.3 Задание на курсовую работу

Задание на курсовую работу формулируется преподавателем при участии студента и включает:
– тему работы;
– исходные данные работы;
– срок сдачи работы;
– содержание пояснительной записки;
– перечень графического материала;
– календарный план выполнения работы.
Пример задания на курсовую работу приведен в приложении Б.

2.4 Содержание пояснительной записки

В содержании указываются все разделы и подразделы пояснительной записки и номера их начальных страниц.

2.5 Введение

Введение – это небольшой обзор курсовой работы. Во введении следует указать цель выполнения курсовой работы, сформулировать задачи для достижения цели, кратко описать содержание пояснительной записки со ссылкой на ее разделы.

2.6 Основная часть пояснительной записки

Основная часть пояснительной записки должна включать следующие разделы и подразделы.

2.6.1 Постановка задачи и анализ аналогичных решений

В настоящем разделе необходимо обосновать выбор области решения бизнес-задачи, СУБД и технологии, в ней применяющейся.

Примерные тематики областей баз данных приведены в разделе 4.

При формулировании задания должно быть подробно указано, какие именно бизнес-задачи сможет решить использование базы данных. Например, в случае автоматизации работы магазина необходимо указать в задании:

– процедуры поставки товара на склад магазина;

– процедуры возврата товара;

– процедуры оплаты товара, в том числе со скидкой;

– процедуры получения наличия остатков товара на складе;

– анализ свойств покупателей: частота покупок, наличие дорогостоящих покупок и т. д.;

– управление работой продавцов (смены, ответственность за определенные участки и др.)

Список решаемых бизнес-задач должен быть согласован с преподавателем. Чем больше бизнес-задач могут быть решены при помощи вашей базы данных, тем лучше и, соответственно, выше оценка. Если какая-то из указанных бизнес-задач не была решена, работа может быть признана невыполненной. Если какая-то из указанных задач была решена не полностью, работа может быть признана выполненной не полностью и оценка будет пропорционально снижена.

Предполагается, что будет использоваться СУБД Microsoft SQL Server версии от 2012 или Oracle 12c. Однако, по согласованию с преподавателем, можно использовать и другую реляционную СУБД, а в случае сравнения применения реляционного и нереляционного решения должны быть использованы оба типа СУБД.

При выборе СУБД следует руководствоваться в том числе и областью решения задачи, например, если ваша задача предполагает сравнение пространственных областей, то в выбранной СУБД должен быть удобный и достоверный механизм работы с пространственными данными.

При выборе технологии необходимо учитывать, целесообразно ли применение данной технологии при решении именно такой задачи. Например, в задачах, предполагающих создание общей базы данных для распределенного предприятия, может быть использована технология репликации. Примерный список технологий СУБД приведен в разделе 4.

Система управления базами данных должна предоставлять не только интерфейс, позволяющий пользователям создавать базы данных и выбирать или изменять данные, но также и такие компоненты для управления хранимыми данными, как:

– физическая независимость данных;
– логическая независимость данных;
– оптимизация запросов;
– целостность данных;
– управление параллельными запросами;
– безопасность данных;
– резервное копирование и восстановление данных.

Физическая независимость данных означает, что хранение данных не зависит от их физической структуры. Если данные были упорядочены одним образом, а затем другим образом, то изменение физических данных не должно влиять на описание базы данных, созданное языком определения данных.

Логическая структура базы данных может быть изменена без внесения изменений в файлы базы данных. Например, добавление столбца к уже существующей в базе данных таблице позволяет изменить только логическую структуру базы данных, а не менять структуру файла.

Большинство СУБД содержат так называемый оптимизатор, который рассматривает возможные стратегии исполнения запроса к данным и выбирает наиболее эффективную из них, называемую планом выполнения запроса. Оптимизатор принимает решение, исходя из набора факторов и собранной информации, такой как размер таблиц, индексы и операторы, используемые в предложении WHERE.

Очень важной является задача определить логически противоречивые данные и не допустить их помещения в базу данных. Кроме этого, для большинства реальных задач существуют ограничения для

обеспечения целостности данных. Обеспечение целостности данных может осуществляться в приложении или в СУБД, что является приоритетным.

СУБД должна обладать каким-либо механизмом, обеспечивающим управление одновременными попытками модифицировать данные несколькими приложениями, например механизмом обеспечения разного уровня изолированности транзакций.

Наиболее важными понятиями безопасности баз данных являются аутентификация и авторизация. Аутентификация – это процесс проверки подлинности учетных данных пользователя. Аутентификация обычно реализуется требованием ввода имени и пароля пользователя. Система проверяет достоверность этой информации и либо дает доступ к системе, либо нет. Авторизация – это процесс определения права на использование определенных ресурсов. Дополнительной опцией защиты данных может выступать шифрование.

СУБД должна быть оснащена системой для восстановления после ошибок. Например, если в процессе обновления 100 строк таблицы базы данных происходит сбой, то система восстановления должна выполнить откат всех выполненных обновлений, чтобы обеспечить непротиворечивость данных.

В пояснительной записке следует объяснить выбор СУБД и применяемой в ней технологии для решения выбранной бизнес-задачи. Необходимо также описать существующие аналогичные решения. Достаточно трех решений при их наличии. Ссылки на аналогичные решения необходимо указать в списке использованных источников.

2.6.2 Установка и настройка сервера СУБД

Необходимо установить и настроить сервер выбранной СУБД и среду выполнения приложения на университетском сервере (за деталями обратитесь к преподавателю), в облачной инфраструктуре или на личном ноутбуке (по выбору). В пояснительной записке требуется указать, какой сервер (сетевой, локальный, облачный) используется.

Если выбранная СУБД не СУБД Microsoft SQL Server версии от 2012 или Oracle 12c, то, по согласованию с преподавателем, необходимо указать краткие этапы установки и настройки сервера СУБД.

В настоящем пособии описывается порядок установки СУБД Microsoft SQL Server 2012.

Для установки СУБД Microsoft SQL Server 2012 требуется минимально 512 Мб оперативной памяти, 2 Гб дискового пространства, ОС Windows 64 разряда, доступ к сети Интернет.

СУБД Microsoft SQL Server 2012 может быть установлена в следующих редакциях:

– Express Edition;
– Workgroup Edition;
– Standard Edition;
– Web Edition;
– Enterprise Edition;
– Developer Edition;
– Datacenter Edition;
– Parallel Data Warehouse Edition.

Express Edition – это облегченная версия SQL Server, предназначена для разработчиков приложений, содержит базовый набор компонентов, поддерживает интеграцию общеязыковой среды выполнения CLR и поддержку языка XML.

Workgroup Edition – предназначена для малого бизнеса и для использования на уровне отделов предприятия. Отсутствуют средства бизнес-аналитики и возможности обеспечения высокого уровня доступности. Поддерживает системы с двумя процессорами и максимум 2 Гб оперативной памяти.

Standard Edition – версия предназначена для малого и среднего бизнеса, которая содержит весь диапазон возможностей бизнес-аналитики, включая службы Analysis Services, Reporting Services и Integration Services, но не подсистему аудита. Поддерживает до четырех процессоров и 2 Тб оперативной памяти.

Web Edition – версия предназначена для поставщиков вебхостинга, содержит службы отчетности Reporting Services, поддерживает до четырех процессоров и 2 Тб оперативной памяти.

Enterprise Edition – версия предназначена для приложений, критичных по времени и с большим количеством пользователей, поддерживает, кроме прочего, секционирование данных, возможность получения мгновенных снимков состояния базы данных и онлайновую поддержку баз данных.

Developer Edition – версия для разработки, содержит всю функциональность версии Enterprise Edition. Лицензия разрешает использование только для разработки, тестирования и демонстрации на необходимом количестве систем.

Datacenter Edition – версия предназначена для поддержки масштабирования наивысшего уровня. Нет ограничений по памяти, можно создавать до 25 экземпляров сервера, обеспечивается поддержка до 256 логических процессоров.

Parallel Data Warehouse Edition – версия специализирована для хранилищ данных и поддерживает базы данных хранилищ данных размером от 10 Тб до 1 Пб (петабайт, 1 Пб = 1024 Тб). Используется архитектура массово-параллельной обработки с возможностями высокопроизводительных вычислений HPC.

Для установки экземпляра Microsoft SQL Server 2012 необходимо запустить мастер установки (рисунок 2.1).

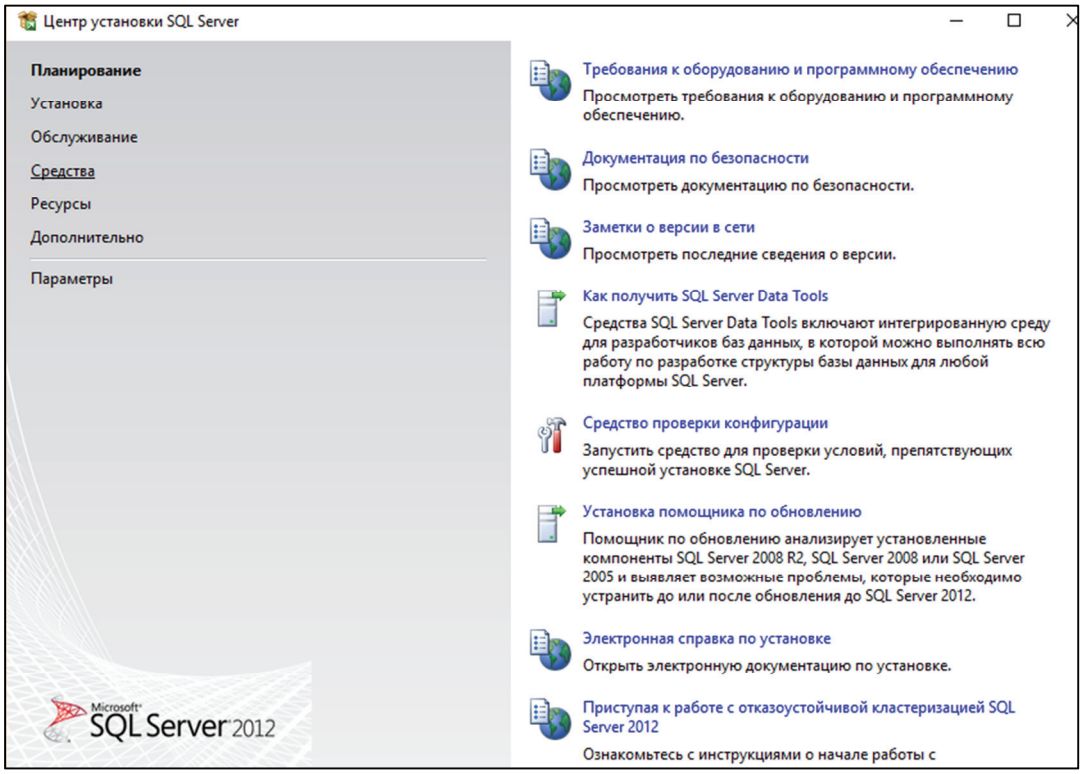

Рисунок 2.1 – Запуск мастера установки и конфигурации Microsoft SQL Server 2012

Мастер установки содержит 7 вкладок, перечисленных слева. Каждая из вкладок позволяет выполнить ряд административных задач, связанных с установкой, настройкой и обновлением Microsoft SQL Server.

Первая вкладка *Планирование* мастера установки предусматривает ссылки на документацию и проверку установленной конфигурации.

Вторая вкладка *Установка* предназначена для старта установки Microsoft SQL Server в режиме установки отдельного сервера, кластерного узла или обновления предыдущей версии.

Третья вкладка *Обслуживание* предназначена для изменения редакции в рамках одной версии Microsoft SQL Server, исправления

повреждений программного обеспечения и удаления кластерного узла и поиска обновлений.

Вкладка *Средства* предназначена для запуска различных утилит конфигурации Microsoft SQL Server и просмотра установленных компонентов.

Вкладка *Ресурсы* является перечнем электронных ресурсов по работе с Microsoft SQL Server.

Вкладка *Дополнительно* предназначена для установки и настройки отказоустойчивого кластера Microsoft SQL Server и создания файла конфигурации для множественных установок Microsoft SQL Server.

На вкладке *Параметры* указывается корневой каталог для поиска образа Microsoft SQL Server и архитектура 32 или 64 бита.

В окне мастера установки нужно перейти на вкладку *Установка* и выбрать *Новая установка изолированного экземпляра Microsoft SQL Server* (рисунок 2.2).

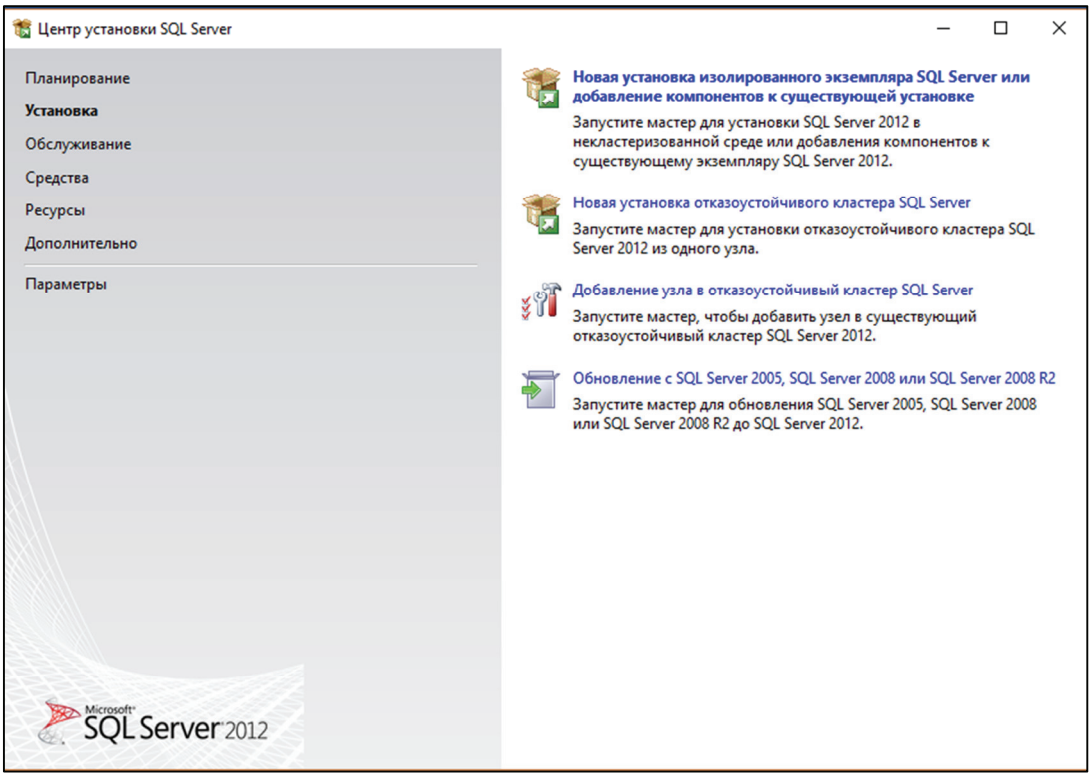

Рисунок 2.2 – Установка нового экземпляра Microsoft SQL Server 2012

Экземпляр компонента Database Engine представляет собой копию исполняемого файла sqlservr.exe, который работает как служба

операционной системы. Экземпляр компонента Database Engine работает как служба, которая обрабатывает все запросы приложений на работу с данными в любой из баз данных, которыми управляет этот экземпляр. На каждом компьютере могут работать несколько экземпляров компонентов Database Engine. Один экземпляр может быть экземпляром по умолчанию, он не имеет имени. Если в запросе на подключение указано только имя компьютера, соединение устанавливается с экземпляром по умолчанию. Экземпляр, которому при установке было задано имя, называется именованным экземпляром. Для подключения к такому экземпляру необходимо указать в запросе на подключение имя компьютера и имя экземпляра. Устанавливать экземпляр по умолчанию необязательно; все экземпляры могут быть именованными.

При установке вначале собирается список потенциальных проблем, которые могут помешать установке сервера экземпляра Microsoft SQL Server 2012 (рисунок 2.3). Если проблем не обнаружено, можно переходить к следующему шагу мастера, иначе требуется ликвидировать перечисленные мастером установки ошибки.

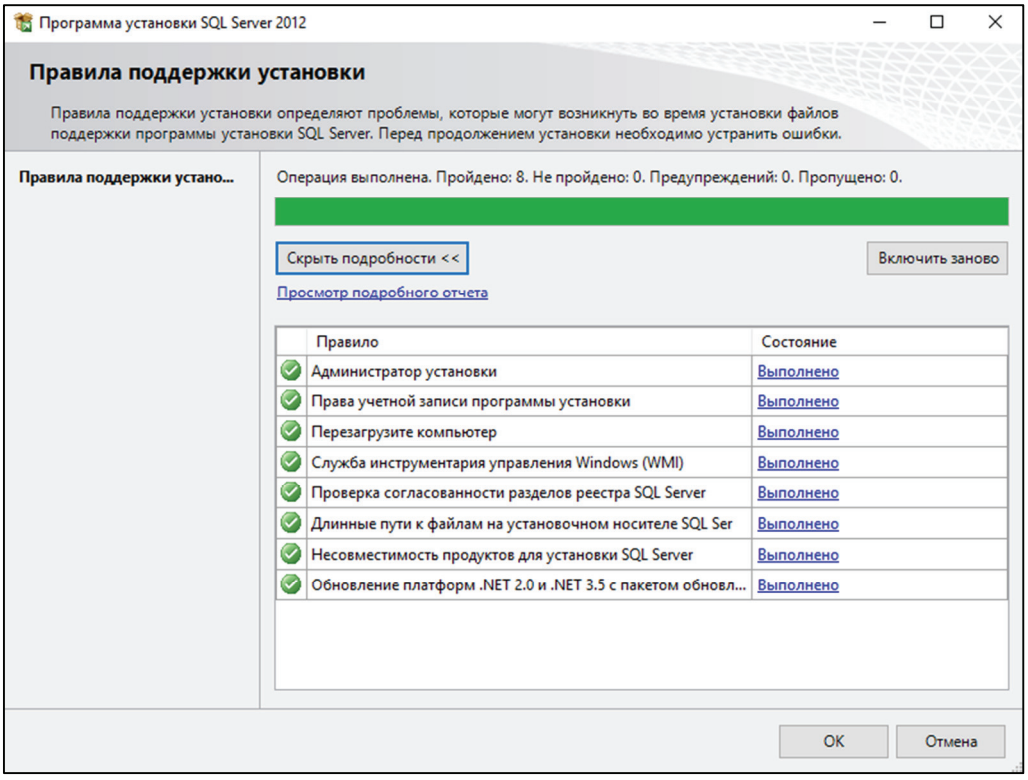

Рисунок 2.3 – Определение списка потенциальных проблем при установке Microsoft SQL Server 2012

Следующий экран мастера – проверка обновлений (рисунок 2.4).

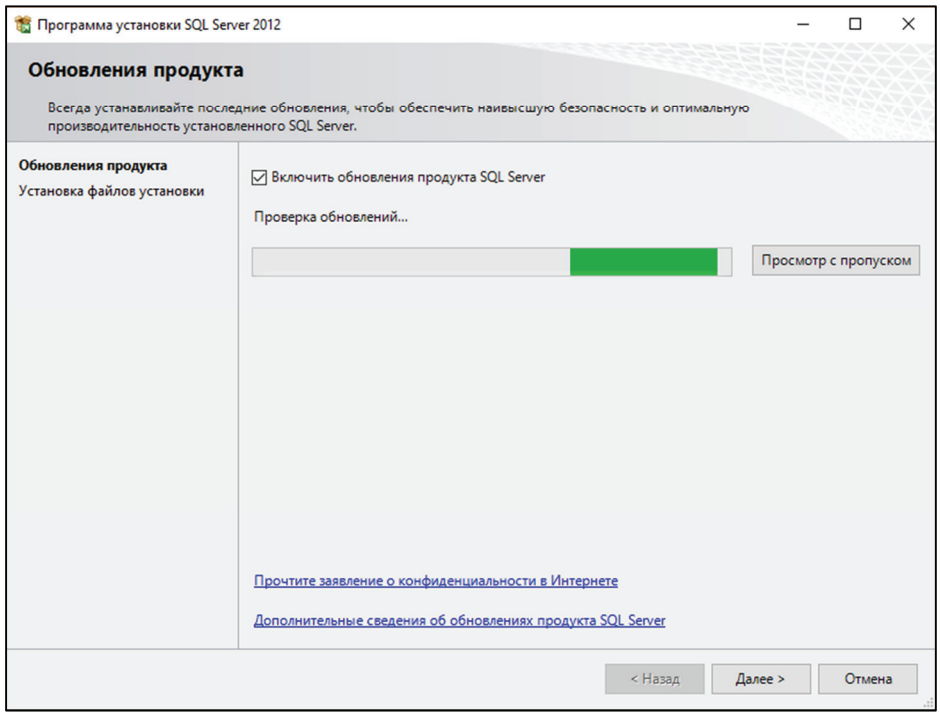

Рисунок 2.4 – Определение списка обновлений
при установке Microsoft SQL Server 2012

Мастер находит существующие обновления и устанавливает сервер вместе с обновлениями (рисунок 2.5).

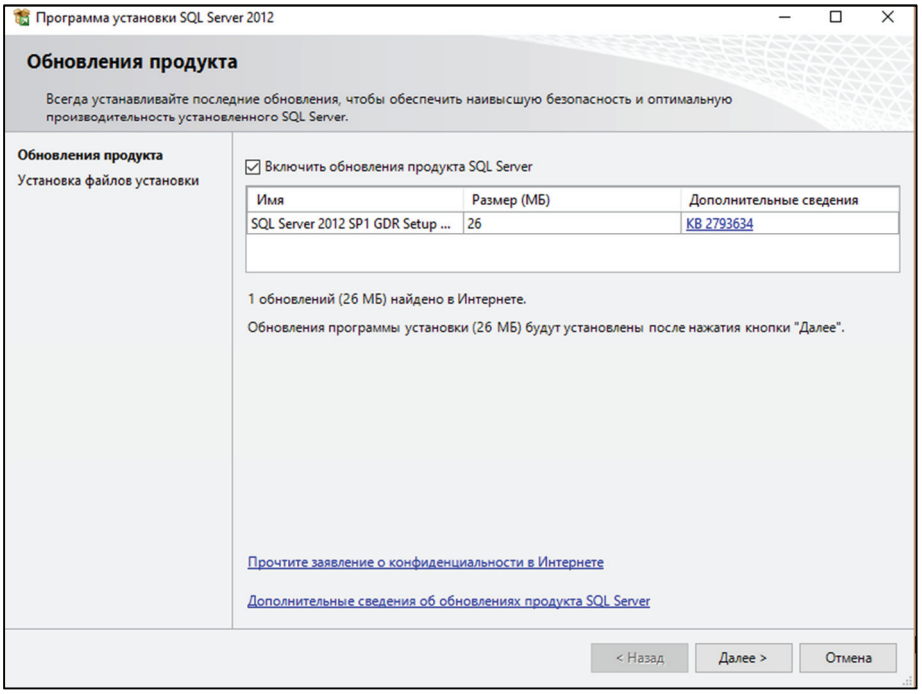

Рисунок 2.5 – Установка обновлений Microsoft SQL Server 2012

Мастер установки копирует свои файлы на диск для продолжения установки (рисунок 2.6). Затем оцениваются потенциальные проблемы установки сервера с учетом предыдущих установок Microsoft SQL Server 2012 (рисунок 2.7).

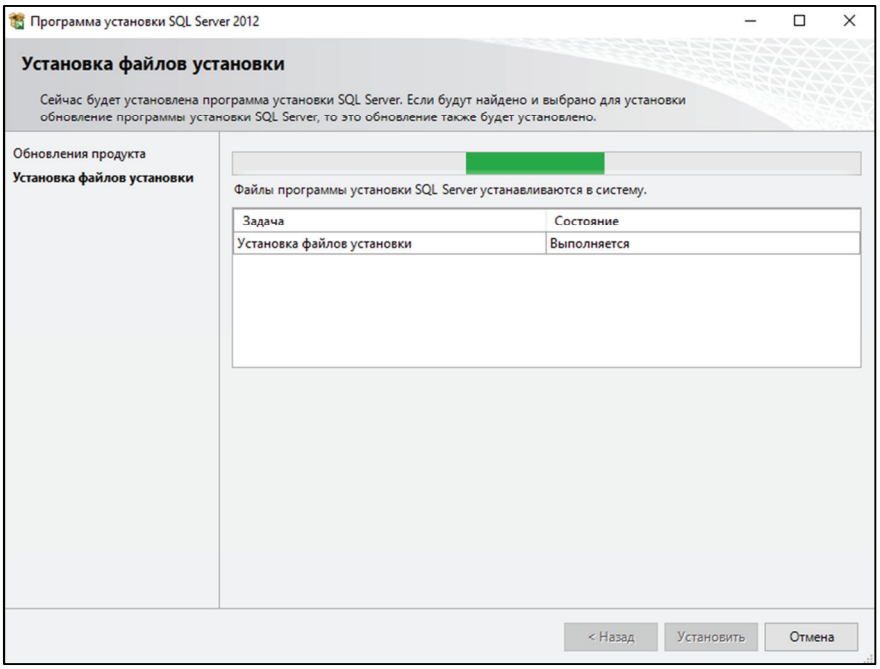

Рисунок 2.6 – Установка мастера установки Microsoft SQL Server 2012

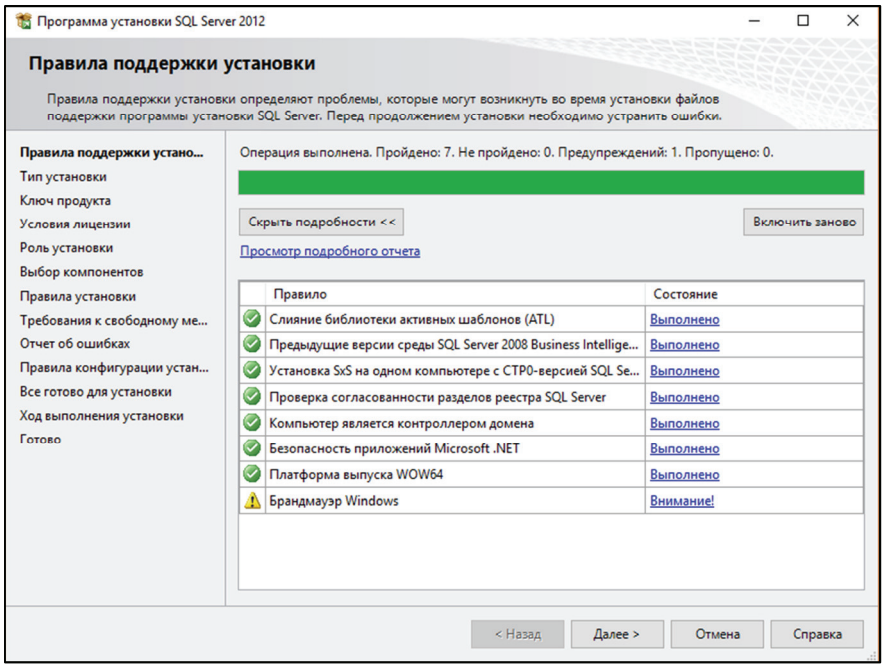

Рисунок 2.7 – Проверка установки Microsoft SQL Server 2012

Далее необходимо указать ключ для платной версии СУБД или доступную бесплатную версию и принять лицензионное соглашение (рисунок 2.8).

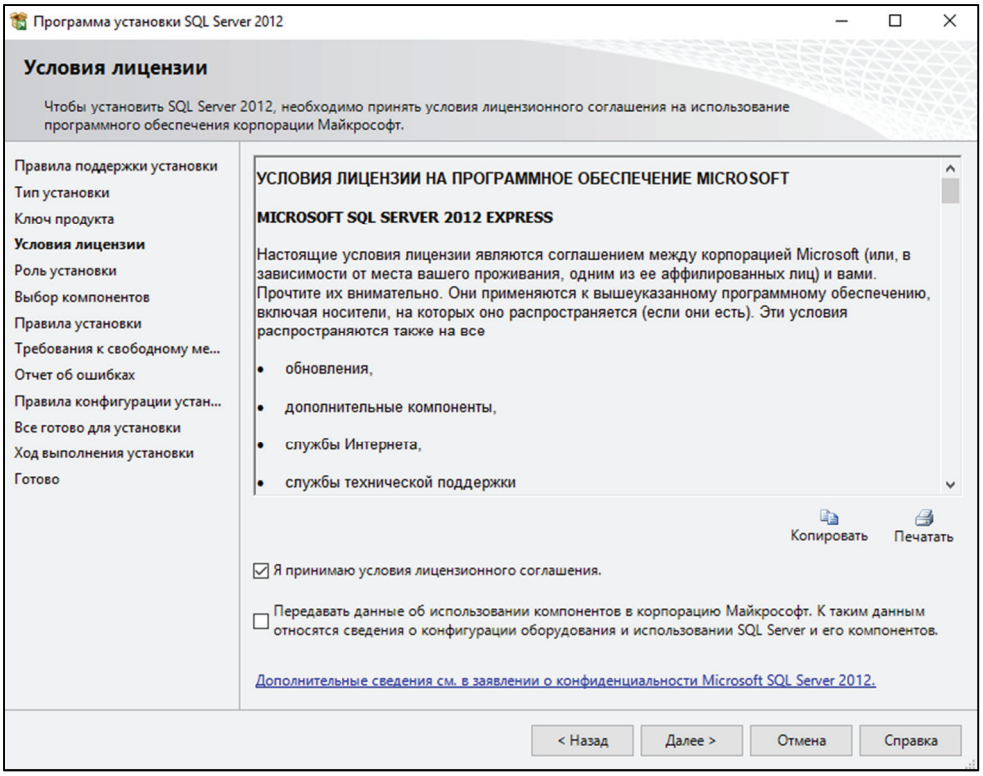

Рисунок 2.8 – Лицензионное соглашение Microsoft SQL Server 2012

На следующем экране (рисунок 2.9) следует выбрать вариант компонентов установки Microsoft SQL Server 2012. Выбирается *Установка компонентов SQL Server* для установки собственно сервера. Могут быть установлены надстройки для сервера SharePoint, связанные с обработкой отчетов, или выбрана установка всех компонентов.

На следующем экране (рисунок 2.10) необходимо выбрать список устанавливаемых компонентов. Для каждого компонента справа от дерева выбора указано краткое описание компонента. В любом случае обязательно требуется указать Database Engine – основной компонент, управляющий работой базы данных.

Службы Analysis Services – это подсистема аналитики данных, используемая в принятии решений и бизнес-аналитике. Она предоставляет семантические модели данных корпоративного уровня для бизнес-отчетов и клиентских приложений, например Excel, отчетов служб Reporting Services и других инструментов визуализации данных.

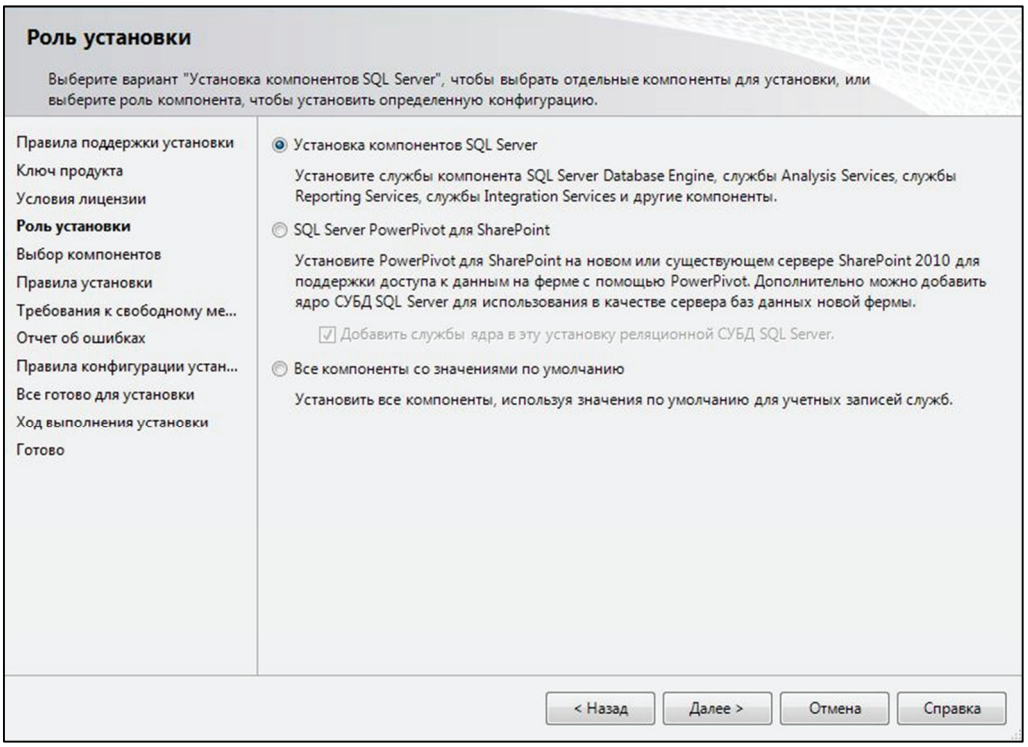

Рисунок 2.9 – Роль установки Microsoft SQL Server 2012

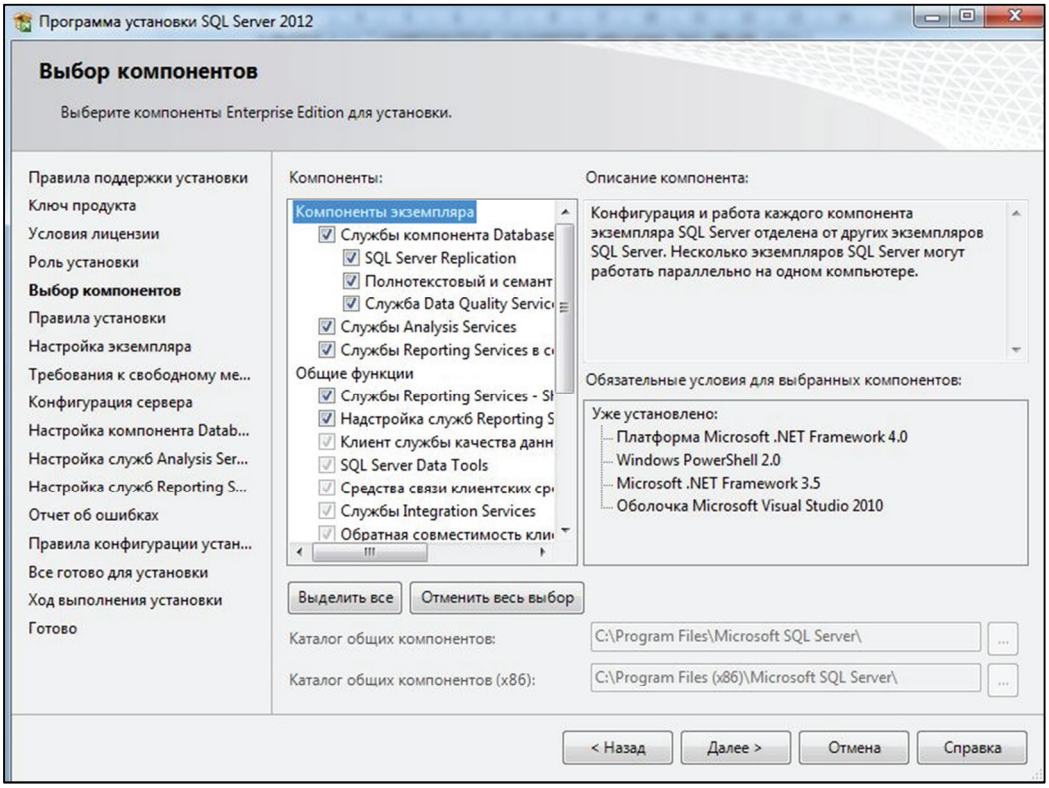

Рисунок 2.10 – Выбор компонентов установки Microsoft SQL Server 2012

Службы Reporting Services – подсистема создания отчетов, которая может быть использована для подготовки множества интерактивных и печатных отчетов.

Службы Integration Services – это платформа для построения решений по интеграции и преобразованию данных уровня предприятия. Службы Integration Services можно использовать для копирования и загрузки файлов, загрузки хранилищ данных, очистки и интеллектуального анализа данных, они могут извлекать и преобразовывать данные из различных источников, таких как XML-файлы, неструктурированные файлы и источники реляционных данных, и затем загружать эти данные в один или несколько реляционных объектов.

Full-Text Search – подсистема полнотекстового поиска в SQL Server, позволяет пользователям и приложениям выполнять полнотекстовые запросы к символьным данным в таблицах SQL Server. Полнотекстовые запросы выполняют лингвистический поиск в текстовых данных в полнотекстовых индексах путем обработки слов и фраз в соответствии с правилами конкретного языка, например английского или русского. Полнотекстовые запросы могут включать простые слова и фразы или несколько форм слова или фразы. Полнотекстовый запрос возвращает все документы, которые содержат как минимум одно совпадение и соответствуют всем условиям поиска, например указанному расстоянию между словами.

SQL Server Agent – это служба Microsoft Windows, выполняющая запланированные административные задания. SQL Server может выполнять задания по расписанию, в ответ на определенное событие или по требованию.

На следующем экране (рисунок 2.11) следует выбрать вариант установки экземпляра Microsoft SQL Server 2012. Выбирается новая установка, если до этого экземпляр Microsoft SQL Server не устанавливался на данный компьютер (тогда устанавливается новый экземпляр по умолчанию) или если требуется установить новый именованный экземпляр Microsoft SQL Server.

Если необходимо внести изменения в существующий экземпляр, например добавить или удалить какие-то компоненты, то выбирается пункт *Добавить компоненты в существующий экземпляр Microsoft SQL Server*. В таблице перечислены все установленные экземпляры, например, на компьютере на рисунке 2.11 установлен экземпляр по умолчанию Microsoft SQL Server и набор дополнительных компонентов.

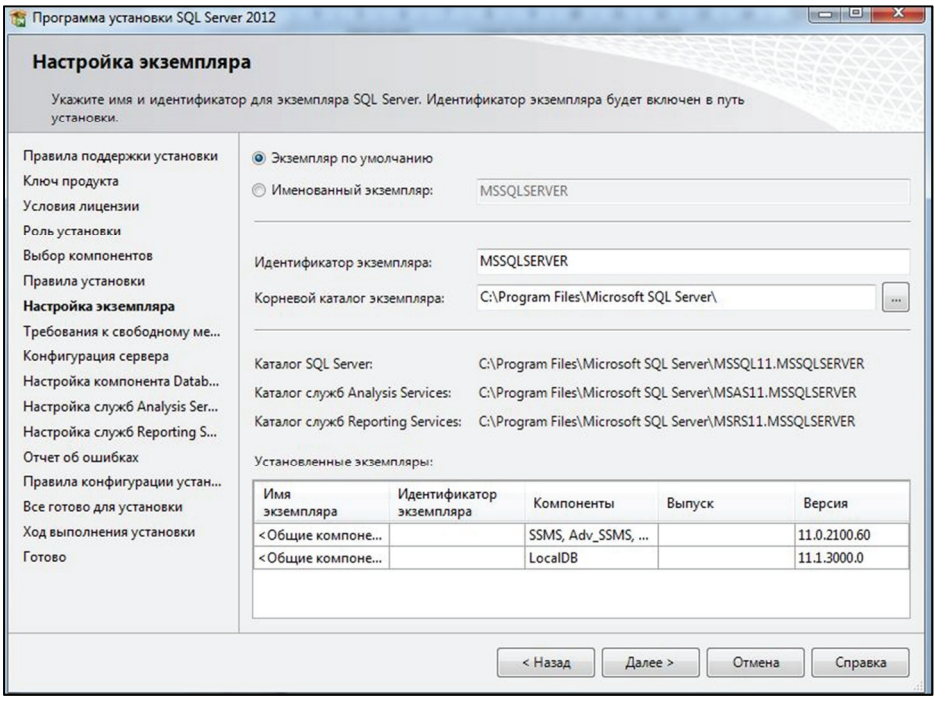

Рисунок 2.11– Выбор экземпляра установки Microsoft SQL Server 2012

Далее подсчитывается необходимое свободное место на диске для установки экземпляра (рисунок 2.12).

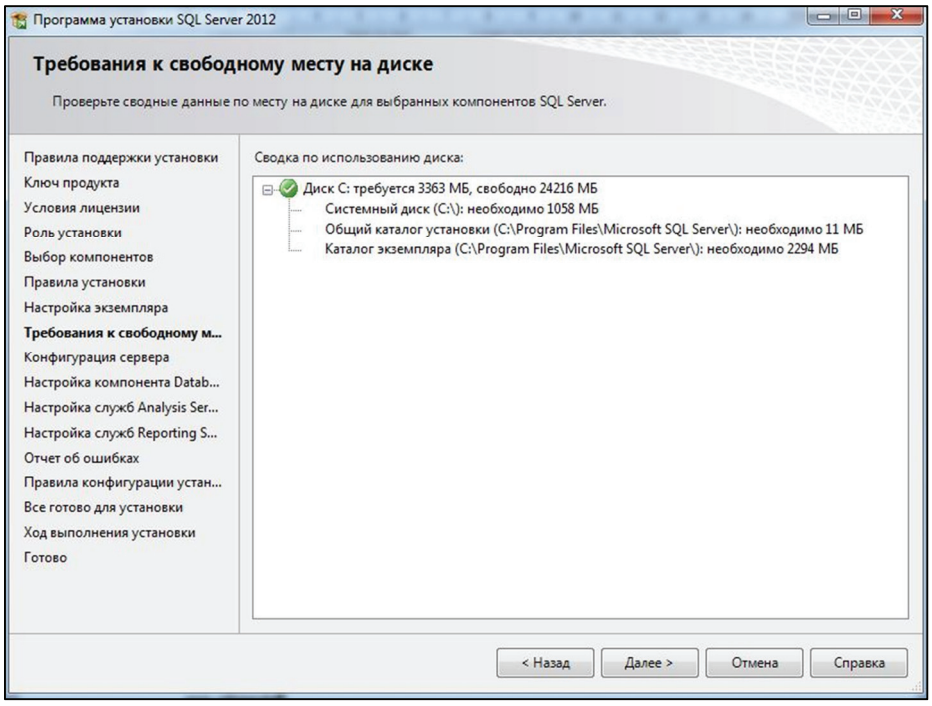

Рисунок 2.12 – Наличие свободного места при установке Microsoft SQL Server 2012

Затем идет процесс настройки параметров устанавливаемого экземпляра Microsoft SQL Server. На первой вкладке *Конфигурация сервера* (рисунок 2.13) требуется указать, какие из служб Microsoft SQL Server будут запускаться автоматически, и под какими учетными записями Windows это будет происходить.

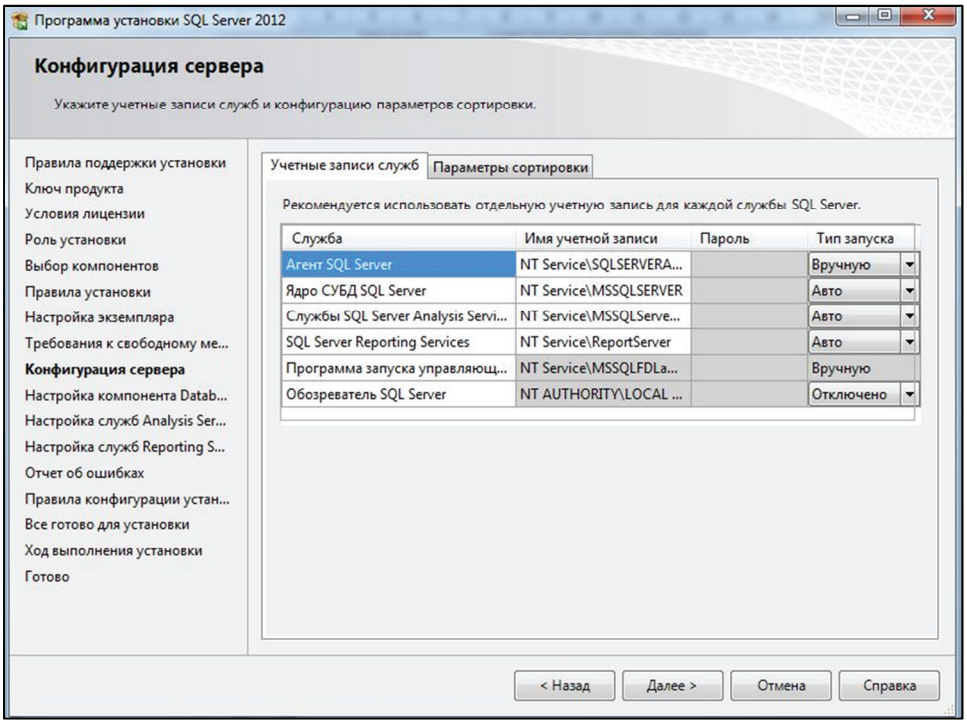

Рисунок 2.13 – Конфигурация служб экземпляра Microsoft SQL Server 2012

На второй вкладке *Конфигурация сервера* (рисунок 2.14) требуется указать, какие параметры сортировки использует данный экземпляр Microsoft SQL Server, это будет влиять на параметры сравнения строк и действия многих встроенных функций, работающих с датами.

На следующем этапе мастера *Настройка компонентов Database Engine* устанавливаются режим проверки подлинности, каталоги по умолчанию для хранения баз данных и журналов, а также поддержка файловых потоков FileStream. FileStream – хранилище данных, позволяющее связать Microsoft SQL Server и файловую систему, хранить данные, например изображения или документы, на диске и управлять ими из базы данных. Для работы с хранилищем FileStream необходимо включить его на уровне сервера и создать специальную файловую группу. Данные, хранящиеся в FileStream, доступны для просмотра и редактирования внешними программами.

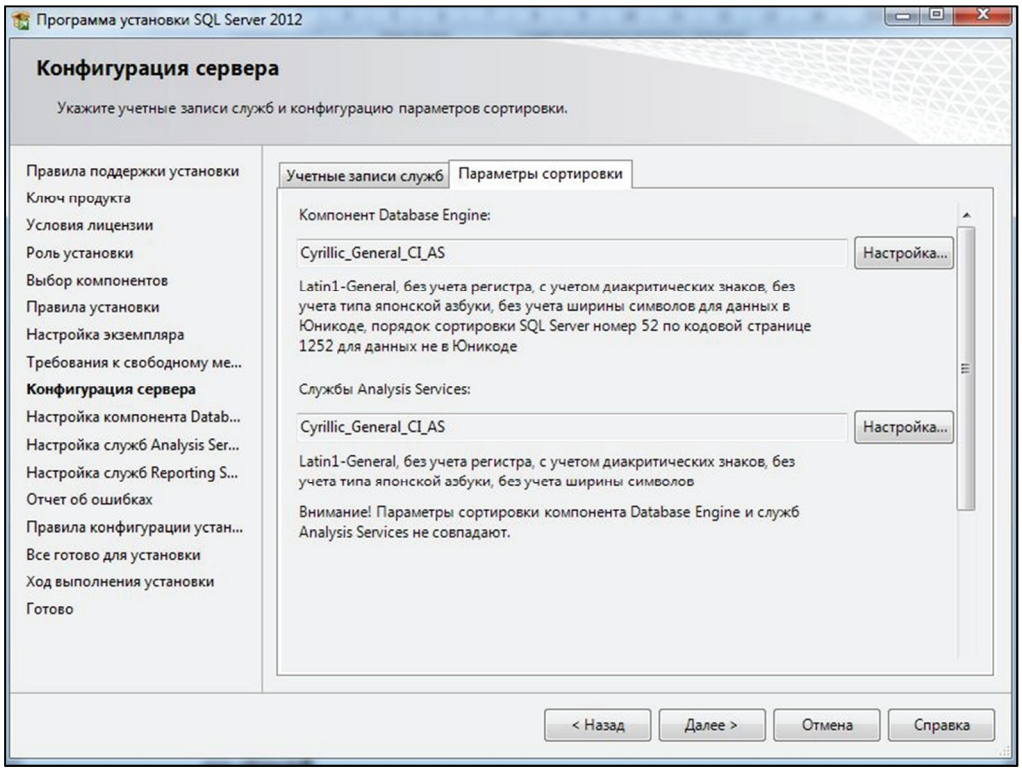

Рисунок 2.14 – Конфигурация параметров сортировки экземпляра Microsoft SQL Server 2012

На вкладке *Конфигурация сервера* требуется указать режим проверки подлинности и администраторов для компонента *Database Engine* (рисунок 2.15).

SQL Server поддерживает два режима проверки подлинности:

– режим проверки подлинности Windows;

– режим смешанной проверки подлинности.

Режим проверки подлинности Windows является режимом по умолчанию. Пользователи Windows, прошедшие проверку подлинности при входе, не должны предъявлять дополнительные учетные данные. Лучше использовать проверку подлинности Windows в следующих ситуациях:

– при наличии контроллера домена;

– приложение и база данных находятся на одном компьютере.

Режим смешанной проверки подлинности проверяет наличие пары «имя пользователя / пароль» в Microsoft SQL Server.

Лучше использовать смешанную проверку подлинности SQL Server в следующих ситуациях:

– пользователи будут подключаться из разных доменов, не имеющих связи с другими доменами;

– база данных используется веб-приложением, например ASP.NET.

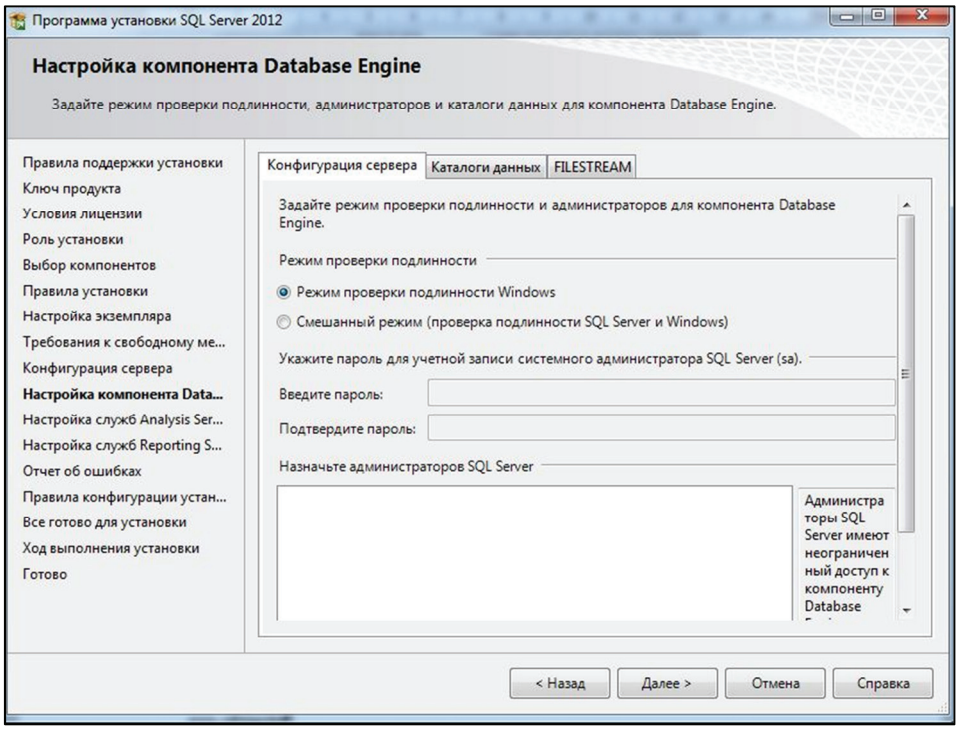

Рисунок 2.15 – Установка режима проверки подлинности экземпляра Microsoft SQL Server 2012

Обязательно убедитесь, что учетной записи системного администратора **sa** назначен надежный пароль и подключение через **sa** не используется в приложении. Microsoft SQL Server всегда устанавливается с именем входа **sa**, которое сопоставляется предопределенной роли сервера **sysadmin**, имеющей безотзывные административные привилегии для всего сервера. Все члены группы Windows BUILTIN\Administrators (группы локального администратора) по умолчанию являются членами роли **sysadmin**, но их можно удалить из этой роли. Если учетная запись **sa** будет оставаться отключенной, то требуется задать пользователей, являющихся администраторами сервера. Для целей курсового проекта их можно задать из числа пользователей компьютера.

Далее необходимо настроить необязательные компоненты Analysis Services и Reporting Services.

Для служб Analysis Services необходимо задать каталоги для хранения данных, журналов, файлов резервных копий и файлов временной обработки данных (рисунок 2.16). Также необходимо задать режим работы сервера:

– многомерный и интеллектуальный анализ данных;

– режим для работы с табличными моделями или моделями SharePoint.

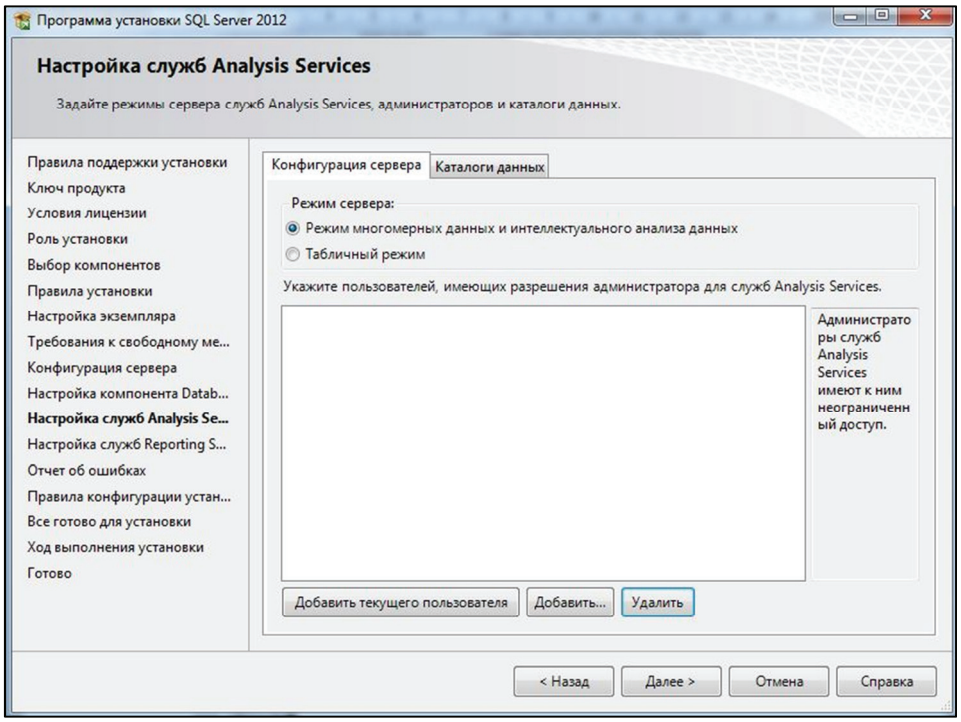

Рисунок 2.16 – Режим установки параметров служб Analysis Services экземпляра Microsoft SQL Server 2012

Службы Analysis Services могут иметь несколько экземпляров. Все установленные экземпляры работают в одном из режимов, определяемых во время установки, и для каждого режима потребуется отдельный экземпляр Analysis Services. Режим сервера – это свойство, которое определяет архитектуру организации хранилища и использования памяти для конкретного экземпляра. Сервер, работающий в многомерном режиме, использует архитектуру для баз данных многомерных кубов. В табличном режиме сервера используется подсистема аналитики в памяти VertiPaq и сжатие данных для статистической обработки данных при выполнении запросов.

Для служб Reporting Services необходимо задать параметры для работы в собственном режиме и в интеграции с SharePoint (рисунок 2.17). SharePoint – это набор программных компонентов, включающий в себя:

– набор веб-приложений для организации совместной работы;
– функциональность для создания веб-порталов;
– модуль поиска в документах и информационных системах;
– функциональность управления рабочими процессами предприятия;
– модуль создания форм для ввода информации;
– функциональность для бизнес-анализа.

На следующем этапе решается, будут ли передаваться сведения об ошибках, и проверяются возможные причины блокировки установки.

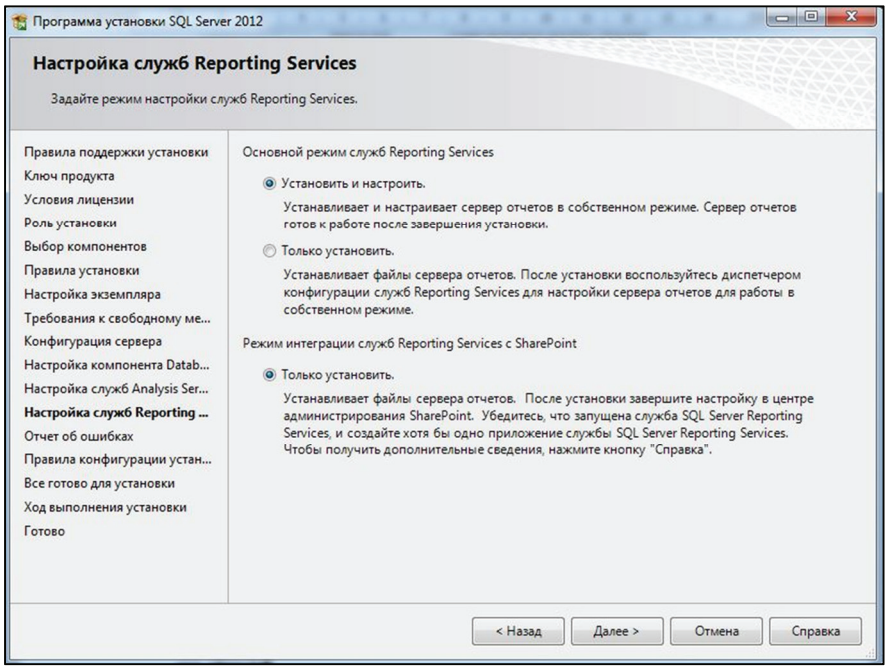

Рисунок 2.17 – Режим установки параметров служб Reporting Services экземпляра Microsoft SQL Server 2012

Если проверка пройдена успешно, то появляется окно со всеми компонентами, которые будут установлены (рисунок 2.18), и начинается процесс установки. После установки сервер сразу готов к работе.

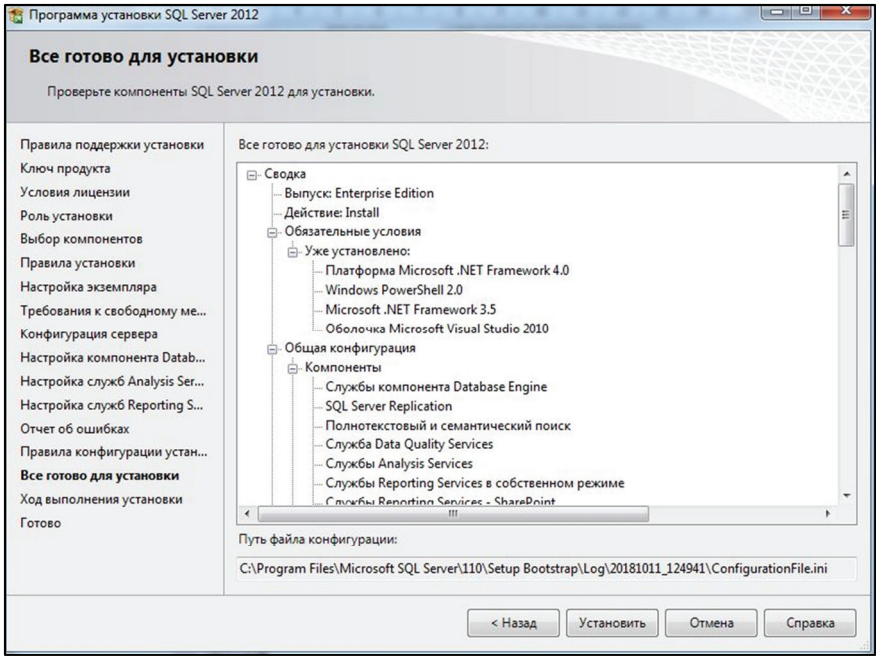

Рисунок 2.18 – Устанавливаемые компоненты экземпляра Microsoft SQL Server 2012

После установки сервера необходимо создать базу данных для выполнения курсового проекта. На рисунке 2.19 приведен скрипт создания базы данных.

```
CREATE database Orders
ON  PRIMARY
(name = 'Order_main_1', filename = 'D:\Database\Order_main_1.mdf',
size = 10240Kb, maxsize = UNLIMITED, filegrowth = 1024Kb),
(name = 'Order_main_2', filename = 'D:\Database\Order_main_2.ndf',
size = 10240KB, maxsize = 1Gb, filegrowth = 25%),

FILEGROUP FG1
(name = 'Order_Limits_1',   filename = 'D:\Database\Order_Limits_1.ndf',
size = 10240Kb, maxsize = 1Gb, filegrowth = 25%),
(name = 'Order_Limits_2',   filename = 'D:\Database\Order_Limits_2.ndf',
size = 10240Kb, maxsize = 1Gb, filegrowth = 25%)

LOG ON
(name = 'Order_log', filename = 'D:\Database\Order_log.ldf',
size = 10240Kb, maxsize = 2048Gb,   filegrowth = 10%);
```

Рисунок 2.19 – Скрипт для создания базы данных

При создании базы данных указывается:

– имя;

– имена и размеры файлов базы данных;

– файловые группы.

Кроме этого, для CREATE DATABASE существует большое число дополнительных параметров, с которыми можно ознакомиться в документации.

Имя базы данных должно соответствовать соглашению об именовании. В каждой базе данных имеется по крайней мере два файла (первичный файл и файл журнала транзакций) и по крайней мере одна файловая группа. При создании базы данных файлы данных следует делать как можно большего размера в соответствии с максимальным предполагаемым объемом данных в базе данных.

Для просмотра сведений об объектах в базе данных используется системный каталог. Представления каталога возвращают данные, используемые SQL Server Database Engine. Все доступные для пользователя метаданные предоставляются через представления каталога.

В пояснительной записке требуется указать размещение базы данных, ее файлов и файловых групп и пояснить, почему это размещение выполнено именно таким образом.

2.6.3 Проектирование инфраструктуры базы данных и порядок авторизации пользователей в базе данных

Проектирование физической реализации базы данных включает планирование файловых групп, использование секционирования для управления большими таблицами и возможное использование сжатия.

После аутентификации пользователя в базе данных ему разрешается выполнять в ней действия над данными. Не имея никаких полномочий, пользователь не может выполнять действия с базой данных, даже установить соединение с ней. С другой стороны, большинству пользователей не требуется создавать, изменять или удалять таблицы и другие объекты базы данных, поэтому привилегии на создание и удаление таблиц им тоже ни к чему.

Привилегия – это право выполнять конкретный тип предложений SQL, или право доступа к объекту другого пользователя. Привилегии назначаются оператором GRANT и отзываются оператором REVOKE.

Для поддержки доступа к объектам в Microsoft SQL Server дополнительно реализована поддержка оператора DENY, прямо запрещающего доступ к объекту.

Владелец объекта обладает всеми привилегиями на свой объект. Владелец может предоставить любую привилегию на любой объект схемы, которым он владеет, любому другому пользователю или любой роли.

Привилегии могут назначаться как по отдельности, так и набором. Роль – это именованный набор привилегий. Использование ролей существенно облегчает управление привилегиями. Вместо того чтобы предоставлять десятки полномочий каждому пользователю, можно создать несколько типичных ролей, наделить их необходимыми привилегиями и назначить пользователям их роли. При необходимости забрать одну роль проще, чем отбирать привилегии по списку.

Создаются, изменяются и удаляются роли стандартными операторами DDL – CREATE, ALTER и DROP. Роли предоставляются пользователям оператором GRANT, а отбираются оператором REVOKE.

Для доступа к серверу создается имя входа (login). Затем для базы данных создается пользователь, использующий это имя входа.

Пользователь базы данных является учетной записью, через которую можно подключиться к базе данных и получить доступ к данным при наличии соответствующих привилегий.

Администратор базы данных создает пользователя, и на момент создания пользователь не имеет никаких привилегий. Затем администратор базы данных может предоставить этому пользователю определенные привилегии. Этими привилегиями определяется, что именно пользователь может делать на уровне базы данных.

Каждый объект имеет определенный набор предоставляемых привилегий. В таблице 2.1 перечисляются привилегии для различных объектов.

Таблица 2.1 – Возможные привилегии

Привилегия	Описание
SELECT	Пользователь может выполнять запросы для таблицы
INSERT	Пользователь может добавлять строки в таблицу
DELETE	Пользователь может удалять строки из таблицы
UPDATE	Пользователь может изменять существующие значения в строках таблицы, причем можно ограничить множество столбцов таблицы
CREATE	Пользователь может создавать объекты сервера и базы данных
ALTER	Пользователь может изменять свойства (кроме владения) объекта, также получает права на выполнение ALTER, CREATE и DROP на любых защищаемых объектах в данной области
EXECUTE	Пользователь может выполнять хранимую процедуру или функцию
REFERENCES	Пользователь может определить внешний ключ, который использует один или несколько столбцов таблицы в качестве родительского ключа
CONTROL	Пользователь получает возможности владельца, имеет возможность предоставлять разрешения, неявно включает разрешение CONTROL для всех объектов в этой области видимости
TAKE OWNERSHIP	Пользователь получает возможность становиться владельцем объекта
VIEW DEFINITION	Пользователь получает возможность просматривать метаданные объекта
IMPERSONATE	Пользователь может выполнить запрос от имени другой учетной записи

Пользователь создается при помощи команды CREATE USER, параметры пользователя могут быть изменены при помощи команды ALTER USER, удаление пользователя производится командой DROP USER.

В курсовой работе необходимо определить порядок аутентификации и авторизации пользователей, в том числе:

– создать необходимые роли;

– предоставить ролям необходимые привилегии;

– создать пользователей и назначить им роли.

Обратите внимание, что пользователям должны быть предоставлены только необходимые привилегии, т. е. обычный пользователь базы данных не имеет доступа к системному каталогу и не может изменять структуру базы данных.

В пояснительной записке требуется указать все созданные роли и пользователей базы данных с указанием привилегий.

Кроме того, в пояснительной записке требуется указать логины, роли и пользователей сервера и базы данных, необходимые и достаточные привилегии.

2.6.4 Проектирование и реализация объектов базы данных

Для решения бизнес-задач, сформулированных в техническом задании, необходимо создать необходимые объекты базы данных.

В простейшем случае это:

– таблицы;

– представления;

– индексы;

– хранимые процедуры и функции, определенные пользователем;

– триггеры.

Логическая схема базы данных должна быть приведена в разделе графического материала.

2.6.4.1 Проектирование и реализация таблиц

В системе управления реляционными базами данных пользовательские и системные данные хранятся в таблицах. Каждая таблица состоит из набора строк, которые описывают сущности, и набора столбцов, которые содержат атрибуты объекта. Проектирование таблиц – одна из наиболее важных задач, которые выполняет разработчик базы данных, поскольку неправильный дизайн таблиц приводит к невозможности эффективного запроса данных.

Чтобы создать таблицу (рисунок 2.20), нужно указать имя таблицы, имена и типы данных для каждого столбца таблицы. Также рекомендуется

продумывать и указывать ограничения целостности для каждого из столбцов.

```
CREATE TABLE dbo.Orders (
    Order_Id int IDENTITY(1,1) NOT NULL PRIMARY KEY,
    Order_Date datetime NOT NULL,
    Customer int NOT NULL REFERENCES Customer(Customer_Id),
    Notes nvarchar(50) NOT NULL);
```

Рисунок 2.20 – Скрипт для создания таблицы

Большинство таблиц имеют первичный ключ, состоящий из одного или нескольких столбцов таблицы. Первичный ключ всегда уникален.

При установке SQL Server указывается параметр *Параметры сортировки сервера.* Он может быть установлен с учетом регистра и без учета регистра. Если этот параметр установлен с учетом регистра, имена объектов могут иметь одно и тоже имя в разном регистре. Например, таблица с именем Order будет отличаться от таблицы ORDER. Иначе эти два имени таблицы будут рассматриваться как одна таблица, т. е. имя в разных регистрах может быть использовано только один раз. Просмотреть установленные параметры сортировки можно в свойствах сервера (рисунок 2.21).

При выполнении условия WHERE сравнение строк (CHAR, VARCHAR, NCHAR и NVARCAR) выполняется в действующем порядке сортировки. При сравнении строк сравниваются соответствующие символы каждой строки. Старшинство символа определяется его позицией в кодовой таблице: символ, чей код стоит в таблице раньше, считается меньше. При сравнении строк разной длины более короткая строка дополняется в конце пробелами до длины более длинной строки.

Приоритет выполнения логических операторов:

– оператор NOT;

– оператор AND;

– оператор OR.

При создании таблиц может быть указан необязательный элемент, называемый схемой. Схема – это объект базы данных, к которому принадлежит таблица. Если текущий пользователь является администратором, то схемой по умолчанию будет схема **dbo**.

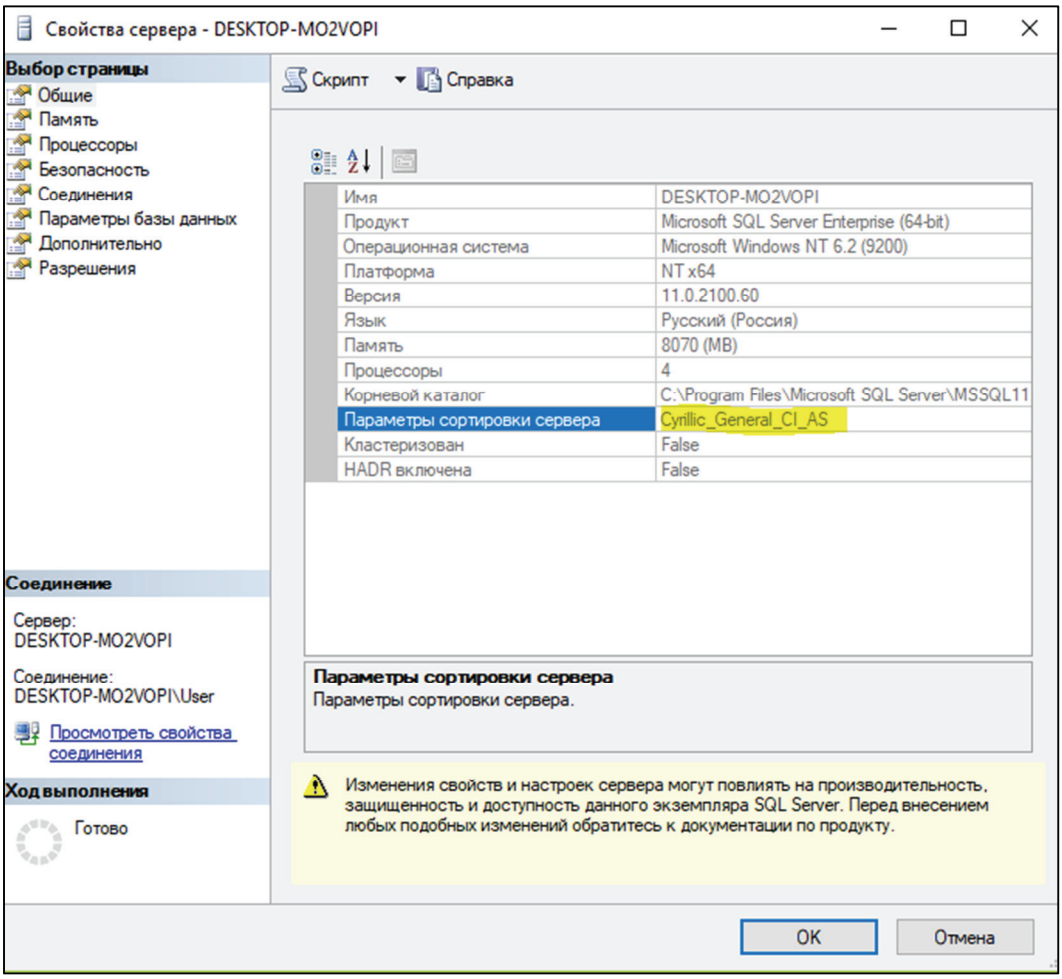

Рисунок 2.21 – Свойства сортировки

Тип данных для столбца должен выбираться в соответствии с требованиями бизнес-задачи. Типы данных делятся на следующие основные группы:

– числовые типы данных;
– типы данных, обеспечивающие хранение значений даты и времени;
– строковые типы данных;
– бинарные типы данных;
– остальные типы данных.

В таблице 2.2 перечислены основные числовые типы данных SQL Server 2012. В SQL Server нет логического типа (истина/ложь), вместо него используется тип bit. Тип decimal (numeric) хранит числа с фиксированной точностью. Параметрами этого типа являются точность и масштаб. Точность представляет собой количество десятичных знаков в числе. Масштаб представляет собой количество десятичных знаков справа от десятичного разделителя.

Таблица 2.2 – Числовые типы данных

Тип данных	Диапазон значений	Количество байт
tinyint	0–255	1
smallint	–32768–32768	2
int	-2^{31}–(2^{31} – 1)	4
bigint	-2^{63}–(2^{63} – 1)	8
bit	0 или 1	1
decimal (p, s), $1 \le p \le 38$, $0 \le s < p$	$(-10^{38} + 1)$–$(10^{38} + 1)$	5–17
smallmoney	–214 748,3648–214 748,3647	4
money	922 337 203 685 477,5808– 922 337 203 685 477,5807	8
float	(–1,79E + 308)–(1,79E + 308)	4–8
real	(–3,40E + 38)–(3,40E + 38)	4

В таблице 2.3 перечислены основные типы данных SQL Server 2012, представляющие дату и время.

Таблица 2.3 – Типы данных даты и времени

Тип данных	Диапазон	Количество байт
date	01.01.0001–31.12.9999	3
time(p)	00:00:00.0000000– 23:59:59.9999999	3–5
smalldatetime	01/01/1900–06/06/2079	4
datetime	01/01/1753–31/12/9999	8
datetime 2	01/01/0001 00:00:00.0000000– 31/12/9999 23:59:59.9999999	6–8
datetimeoffset	От 0001-01-01 до 9999-12-31, с точностью до 10 наносекунд	10

В таблице 2.4 перечислены основные типы строковых данных SQL Server 2012. Типы данных char и varchar хранят строки, выделяя на их хранение по 1 байту на каждый символ. Типы данных nchar и nvarchar хранят строки в кодировке Unicode, выделяя на их хранение по 2 байта на каждый символ. Типы данных char и nchar хранят строки фиксированной длины, т. е. при добавлении строки меньшей длины, чем размер столбца, строка будет дополнена пробелами.

Таблица 2.4 – Строковые типы данных

Тип данных	Размер в символах	Количество байт
char(n)	1–8000	n
varchar(n)	1–4000	Количество символов + 2
varchar(max)	$1–(2^{31} – 1)$	Количество символов + 2
nchar(n)	1–8000	2n
nvarchar(n)	1–4000	Удвоенное количество символов + 2
nvarchar(max)	$1–(2^{30} – 1)$	Удвоенное количество символов + 2

В таблице 2.5 перечислены основные типы бинарных данных SQL Server 2012. Бинарные типы данных служат для хранения изображений, аудио, видео и пр.

Таблица 2.5 – Бинарные типы данных

Тип данных	Размер в байтах	Описание
binary(n)	1–8000	Бинарные данные в виде последовательности постоянной длины
varbinary(n)	1–8000	Бинарные данные в виде последовательности переменной длины
varbinary(max)	$1–(2^{31} – 1)$	Бинарные данные в виде последовательности переменной длины

В таблице 2.6 перечислены остальные типы данных SQL Server 2012.

Таблица 2.6 – Прочие типы данных

Тип данных	Размер в байтах	Описание
uniqueidentifier	16	Уникальный идентификатор GUID
timestamp	8	Номер версии строки в таблице
hyerarchyid	–	Обеспечивает хранение иерархических данных
sql_variant	–	Может хранить данные любого другого типа данных T-SQL
XML	2^{30} (2 Гб)	Хранит документы XML или фрагменты документов XML
geography	–	Хранит географические данные широты и долготы
geometry	–	Хранит координаты на плоскости

Кроме указания типа данных, которое само по себе является ограничением целостности data type, при создании таблиц могут быть указаны следующие ограничения целостности:

– not null – запрет пустых значений NULL;

– default – устанавливает значение в столбце по умолчанию при выполнении операции INSERT;

– primary key – предотвращает появление в столбце или группе столбцов повторяющихся значений и пустого значения;

– foreign key – устанавливает связь между таблицей со столбцом, имеющим свойство foreign key (FK – внешний ключ), и таблицей, имеющей столбец со свойством primary key (PK – первичный ключ); предотвращает несогласованные операции между PK и FK;

– unique – устанавливает уникальные значения, значение NULL может быть только одно;

– check – предотвращает появление в столбце значения, не удовлетворяющего логическому условию, записанному после check.

Необходимо выполнить проектирование таблиц с указанием правильных типов данных и ограничений целостности, в том числе первичных и внешних ключей. В пояснительной записке должны быть перечислены все таблицы с указанием типов данных и ограничений целостности и кратко указано, данные о каких объектах находятся в этой таблице.

2.6.4.2 Проектирование и реализация представлений

Представление – это поименованный SELECT-запрос, который хранится и может использоваться в других запросах точно так же, как и таблица. Представления упрощают дизайн базы данных, формируя слой абстракции и скрывая сложность соединения таблиц. Представления также являются способом защиты данных, предоставляя пользователям разрешения использовать требуемые данные, при этом не позволяя манипулировать исходными объектами, содержащими весь набор данных.

В операторе CREATE после ключевого слова VIEW следует имя представления и далее после ключевого слова AS текст SELECT-запроса, лежащего в основе представления.

К SELECT-запросу представления предъявляются следующие требования:

– секцию ORDER BY можно использовать только совместно с опцией TOP;

– не допускается применение секции INTO;

– все столбцы результирующего набора, формируемого SELECT-запросом, должны быть поименованы.

Указание опции SCHEMABINDING в представлении запрещает операции с таблицами и представлениями, используемыми в SELECT-запросе, на котором основано представление.

При создании представлений, позволяющих выполнять операции INSERT, DELETE и UPDATE, базовый SELECT-запрос должен удовлетворять следующим правилам:

– запрос не должен содержать секцию группировки GROUP BY и агрегатные функции;

– запрос не должен использовать опции DISTINCT и TOP;

– запрос не должен использовать операторы UNION, INTERSECT и EXCEPT;

– в списке столбцов запроса не должно быть вычисляемых значений;

– в секции FROM запроса должна указываться только одна таблица.

Необходимо выполнить проектирование и реализацию представлений. В пояснительной записке должны быть перечислены все созданные представления и указано их назначение.

2.6.4.3 Проектирование и реализация индексов

Индекс является структурой на диске, которая связана с таблицей или представлением и ускоряет получение строк из таблицы или представления. Индекс содержит ключи, построенные из одного или нескольких столбцов в таблице или представлении. Эти ключи хранятся в виде структуры сбалансированного дерева, которая поддерживает быстрый поиск строк по их ключевым значениям в SQL Server.

Оптимизатор запросов в SQL Server обычно выбирает наилучший индекс в подавляющем большинстве случаев.

При проектировании индексов следует продумать следующее.

Необходимо определить наиболее часто используемые запросы.

Нужно выяснить характеристики столбцов, используемых в запросах, чтобы применить наиболее подходящий тип индекса, такой как индекс покрытия или фильтруемый индекс. Если столбец имеет точно определенные подмножества, например содержит различные диапазоны значений, то отфильтрованный индекс может увеличить скорость выполнения запроса, уменьшить стоимость обслуживания индекса и стоимость хранения.

Необходимо определить, какие параметры индексов могут повысить производительность при создании индекса или при его поддержке. Требуется проверить селективность запроса, т. е. отношение количества строк, удовлетворяющих условию, к общему количеству строк. Оптимальной считается селективность менее 5%.

Необходимо определить оптимальное расположение для хранения индекса. Некластеризованный индекс может храниться в той же или в другой файловой группе. Правильный выбор расположения для хранения индексов может повысить производительность запросов за счет повышения скорости дискового ввода-вывода.

Большое количество индексов в таблице снижает производительность инструкций INSERT, UPDATE, DELETE и MERGE, потому что при изменении данных в таблице все индексы должны быть соответствующим образом изменены, поэтому следует избегать использования чрезмерного количества индексов для интенсивно обновляемых таблиц и следить, чтобы индексы содержали как можно меньше столбцов.

Если таблица является справочником, т. е. практически не обновляется, но содержит большой объем данных, то большое число индексов может повысить производительность запросов, поскольку у оптимизатора запросов будет больший выбор при определении самого быстрого способа доступа.

Индексы для представлений могут дать улучшение производительности, если представление содержит агрегаты, соединения таблиц или сочетание того и другого.

Следует создавать некластеризованные индексы для столбцов, которые часто используются в предикатах и условиях соединения в запросах.

Индексы покрытия могут повысить производительность запросов, так как данные, необходимые для запроса, присутствуют в самом индексе и уменьшается общий объем операций дискового ввода-вывода.

Запросы следует составлять так, чтобы они вставляли или изменяли как можно больше строк одной инструкцией, вместо того чтобы использовать для обновления тех же строк несколько запросов.

Нужно следить, чтобы длина ключа для кластеризованных индексов была небольшой. Кроме того, кластеризованные индексы только выиграют при их создании на основе уникальных или не принимающих значения NULL столбцов. Столбец типа xml может быть ключевым столбцом только в XML-индексе.

Необходимо проверить, являются ли значения в столбце уникальными. Замена неуникального индекса уникальным для той же комбинации столбцов обеспечивает оптимизатору запросов дополнительные сведения, что делает индекс более полезным.

Необходимо проверить распределение данных в столбце. Часто длительное выполнение запроса обусловлено индексированием столбца, в котором мало уникальных значений.

Необходимо учитывать порядок столбцов, если индекс будет включать несколько из них. Столбец, использованный в предложении WHERE в условии поиска равных, больших, меньших или находящихся в интервале значений или участвующий в соединении, должен стоять первым. Дополнительные столбцы должны быть упорядочены по уровню различимости, т. е. от наиболее четкого к наименее четкому.

При определении индексов следует иметь в виду, что данные ключевых столбцов индекса сохраняются в порядке возрастания или убывания. По умолчанию сортировка производится по возрастанию. Указание порядка, в котором значения ключей хранятся в индексе, полезно тогда, когда запрос ссылается на таблицу с предложением ORDER BY, в котором указано другое направление для ключевого столбца индекса или индексированного столбца. В этом случае запрос будет выполняться значительно эффективнее.

В пояснительной записке должны быть перечислены все созданные индексы и кратко указан их тип и назначение.

2.6.4.4 Проектирование и реализация хранимых процедур и функций

Процедуры и функции – это подпрограммы, которые используются для инкапсуляции часто выполняемой логики. Вместо того чтобы повторять логику подпрограммы во многих местах, код может вызывать процедуру или функцию. Это делает код более удобным для обслуживания и легким для отладки. Процедуры также являются способом организации удобного доступа к данным из приложения, предоставляя приложению возможность вызывать процедуру для использования требуемых данных, при этом не позволяя манипулировать исходными объектами, содержащими весь набор данных.

Необходимо выполнить проектирование и реализацию хранимых процедур с параметрами:

- для реализации бизнес-логики задачи;
- вставки данных;
- обновления данных;
- удаления данных;
- выборки данных.

Выборка данных подразумевает, что может быть получен как один объект, так и их набор. В результате выполнения выборки при помощи процедуры должен быть получен набор данных, не требующий дополнительной обработки, например каких-либо вычислений или сортировки.

В процедурах должна быть реализована обработка ошибок. При необходимости должна быть использована обработка бизнес-логики в явной транзакции и установлен необходимый уровень изоляции транзакций.

Также необходимо выполнить проектирование и реализацию функций, определенных пользователем, для реализации бизнес-логики задачи. По необходимости могут быть использованы как скалярные, так и табличные функции. Встроенные функции не дают выигрыша в производительности ни в сравнении с явными SELECT-запросами, ни в сравнении с представлениями, однако позволяют существенно сократить размер SELECT-запроса и могут быть повторно применены.

В пояснительной записке должны быть перечислены все процедуры и функции с указанием входных и выходных параметров (при их наличии) и возвращаемых значений с указанием типов данных и описано, что именно выполняет каждая хранимая процедура или функция, определенная пользователем.

2.6.4.5 Проектирование и реализация триггеров

Триггеры языка манипуляции данными (DML) – это мощные инструменты, которые можно использовать для обеспечения доменной целостности, контроля сущности реляционных данных и бизнес-логики. Обеспечение целостности помогает создавать надежные приложения.

Требуется выполнить проектирование и реализацию необходимых триггеров для поддержания доменной целостности, контроля сущности реляционных данных и реализации бизнес-логики базы данных. Рекомендуется создать триггеры аудита. При необходимости можно использовать DDL-триггеры.

В пояснительной записке должны быть перечислены все триггеры и указано, что именно выполняет каждый триггер и по какому условию.

2.6.5 Изучение и применение технологии в созданной базе данных

Необходимо изучить и подробно описать избранную технологию для СУБД. В записке должно быть подробно отражено следующее:

– формальное описание назначения технологии;

– примеры приложений с использованием данной технологии;

– каким образом используется эта технология в базе данных;

– листинги процедур, использующих эту технологию (приводятся в приложении).

2.6.6 Импорт и генерация тестовых данных

Для демонстрации функционирования базы данных и оптимизации запросов к ней необходимо произвести импорт и экспорт данных. Данные могут импортироваться и/или дополнительно генерироваться, причем импорт и экспорт может производиться для данных следующих форматов:

– табличных данных;

– данных в формате XML или JSON;

– данных в текстовом формате;

– с использованием команд BCP и BULK INSERT.

Рекомендуется создать процедуры импорта и экспорта данных и привести их в пояснительной записке. Листинги процедур приводятся в приложении.

Также данные могут быть сгенерированы любым удобным способом. Для проведения оптимизации запросов в базе данных необходимо, чтобы количество строк в таблицах, к которым выполняются запросы, было не менее 100 000. Для оптимизации запросов в базе данных необходимо провести исследование SELECT-запросов и создать необходимые индексы и/или переформулировать запросы. В пояснительной записке необходимо описать планы выполнения запросов до оптимизации, предложенные меры и состояние после оптимизации.

2.6.7 Резервное копирование и восстановление данных

Необходимо описать схему резервного копирования и восстановления данных с учетом решаемой бизнес-задачи.

В самом простом случае необходимо выполнить полное резервное копирование данных и восстановление из резервной копии. В пояснительной записке необходимо описать скрипты создания резервной копии и восстановления из нее. При использовании специализированных утилит – указать порядок использования утилиты.

Для получения более высокой оценки необходимо создать план резервного копирования и восстановления данных. Следует учитывать различные аспекты потери данных: сбой хранения, сбой при передаче, умышленные или неумышленные действия третьих лиц. Также необходимо учитывать режим работы с данными: требуется ли режим работы 24/7, сколько времени может составлять допустимый простой и на какой момент времени должны быть восстановлены данные. Все эти аспекты должны быть описаны в пояснительной записке.

Во время защиты может потребоваться демонстрация потери данных и их восстановления.

2.6.8 Разработка демонстрационного приложения

Для демонстрации работы необходимо представить результаты курсовой работы в приложении.

Основная цель разработки демонстрационного приложения – показать все стороны решения задачи и выбранную технологию. Среду разработки, язык, структуру и внешний вид приложения студент определяет самостоятельно.

Доступ к данным базы данных осуществляется только при помощи вызова соответствующих процедур и/или функций.

В пояснительной записке приводится краткое описание приложения.

Структура, UML-диаграмма, основные классы могут быть вынесены в приложение.

2.7 Заключение

В заключении формулируются краткие выводы по результатам выполненной работы. Приводятся количественные и качественные характеристики реализации базы данных: решенные бизнес-задачи, результаты применения выбранной технологии в базе данных и т. п.

2.8 Графический материал

Графический материал выполняется на листе формата А4 по ГОСТ 2.301 и представляет собой логическую схему базы данных.

2.9 Список использованных источников

Необходимо перечислить книги, статьи, электронные ресурсы, которые были использованы при выполнении работы. Информация о правилах оформления этого списка приведена в пункте 3.12. Список использованных источников должен содержать не менее пяти наименований.

2.10 Приложения

Приложения содержат объемные материалы или материалы справочного характера, такие как схема базы данных, блок-схемы алгоритмов решения бизнес-задач, тексты SQL-скриптов, UML-диаграмма приложения, краткое описания классов приложения и т. д.

3 Оформление пояснительной записки

3.1 Общие требования

При оформлении пояснительной записки следует руководствоваться требованиями СТП 001-2010 БГТУ «Проекты (работы) дипломные».

Пояснительная записка к курсовой работе оформляется в соответствии с правилами оформления текстовых документов, изложенными в ГОСТ 2.105–95 «Общие требования к текстовым документам», и с правилами оформления курсовых работ, изложенными в данном учебно-методическом пособии. Текст записки должен быть напечатан на одной стороне листа формата А4. Используемый шрифт – Times New Roman, размер шрифта – 14, межстрочный интервал – одинарный. Цвет шрифта – черный. Абзацный отступ – 1,25 см. В тексте после знаков препинания обязательно ставится пробел. Нельзя сокращать слова (кроме сокращений, установленных правилами орфографии). Графический материал оформляется на листах формата А3 с рамкой.

Текст следует печатать, соблюдая поля: правое – 10±1 мм; верхнее – 20±1 мм; левое – 23±1 мм; нижнее – 15±1 мм. При наличии на листе рамки и основной надписи по форме 2 расстояние между верхней границей основной надписи и последней строкой текста, если лист полностью заполняется текстом, должно составлять 10–15 мм.

3.2 Структурные элементы записки

Структурные элементы записки: «Содержание», «Введение», «Заключение», «Список использованных источников», «Графический материал», а также каждый из основных разделов и каждое из приложений следует начинать с нового листа.

Заголовки элементов текста «Содержание», «Введение», «Заключение», «Список использованных источников», «Графический материал» следует записывать в начале соответствующих страниц строчными буквами, первая буква – прописная, шрифт – полужирный, расположение – симметрично тексту, и отделять от него интервалом в 18 пт.

Текст основной части делят на разделы, разделы – на подразделы, подразделы – на пункты и т. д. Разделы нумеруются арабскими цифрами, точка в конце не ставится; подразделы нумеруются в пределах

раздела. Номер раздела отделяется от номера подраздела точкой. В конце номера подраздела точка не ставится. За номером раздела или подраздела следует его название, записанное с прописной буквы без точки в конце. Переносы слов в заголовках не допускаются. Если заголовок состоит из двух предложений, их разделяют точкой. Номер и название раздела или подраздела записывается с абзацного отступа, шрифт – полужирный, отделяется от текста интервалом в 18 пт.

3.3 Нумерация страниц

Нумерация страниц пояснительной записки сквозная. На титульном листе (первой странице пояснительной записки) номер не указывается. Номер проставляется арабской цифрой без точки в правом верхнем углу страницы.

Графический материал, размещенный на листе формата А3, учитывается как одна страница.

3.4 Перечисления

В тексте пояснительной записки могут быть использованы перечисления. Пункты перечисления записывают после двоеточия в виде списка, каждый с абзацного отступа. Перед каждым пунктом ненумерованного списка перечисления следует ставить тире. При необходимости ссылки в тексте на один или несколько пунктов перечисление оформляют в виде списка, каждый пункт которого начинают со строчной буквы русского алфавита (за исключением ё, з, о, ч, ь, й, ы, ъ) с проставленной после нее круглой скобкой. Для дальнейшей детализации перечислений (сложные перечисления) необходимо использовать арабские цифры с проставленными после них круглыми скобками. Запись подчиненных пунктов сложного перечисления выполняют с абзацными отступами по отношению к основному пункту.

Ниже дан пример выполнения простого перечисления.

Перечислим объекты базы данных, используемые в настоящей курсовой работе:

– таблицы;
– представления;
– индексы;
– хранимые процедуры и функции, определенные пользователем;
– триггеры.

3.5 Изложение текста

Текст пояснительной записки должен быть кратким, четким и не допускать различных толкований. В тексте пояснительной записки не допускается применять:

– обороты разговорной речи;

– для одного и того же понятия различные термины;

– сокращения слов, кроме установленных правилами орфографии русского языка.

Перечень допускаемых сокращений русских слов установлен в ГОСТ 2.316 и ГОСТ 7.12, белорусских – в СТБ 7.12.

Если в пояснительной записке принята особая система сокращения слов или наименований, то в ней должен быть приведен перечень принятых сокращений.

3.6 Формулы

Формулы располагаются в тексте в отдельных строках, по центру строки. Формулы нумеруются внутри раздела (формат: номер раздела, порядковый номер формулы через точку). Номера формул записываются на уровне формулы в круглых скобках справа в конце строки. Пояснения символов и числовых коэффициентов, входящих в формулу, должны быть приведены непосредственно под формулой. Пояснения должны начинаться со слова «где» и далее следует описание каждого символа в той последовательности, в которой они приведены в формуле, по одному в строке.

Размер шрифта символов в формулах и уравнениях должен соответствовать размеру основного шрифта текста. Размер индексов при основных символах в формулах и уравнениях – 9 пт.

Ссылки на формулы, ранее приведенные в тексте записки, а также на формулы в приложениях необходимо выполнять с использованием их номера, например: «…по формуле (2.8)…».

3.7 Примечания

Примечания следует применять в пояснительной записке, если необходимы пояснения по содержанию текста, таблиц или иллюстраций.

Примечания необходимо помещать непосредственно после текстового материала (рекомендуется в конце пункта, подпункта), таблицы

или графического материала, к которым они относятся. Если примечание одно, то после слова «Примечание» следует ставить тире, а за ним с прописной буквы – его текст. Одно примечание не нумеруется. Номер примечания от его текста точкой не отделяют. Примечание к таблице необходимо помещать в конце таблицы над обозначающей ее окончание чертой.

Текст примечаний рекомендуется печатать шрифтом размером 12 пт.

3.8 Рисунки

Все рисунки (чертежи, схемы, графики, структурные схемы и др., кроме таблиц) должны располагаться непосредственно после ссылки на них в тексте. Рисунок располагается так, чтобы его удобно было смотреть без поворота листа или с поворотом по часовой стрелке. Рисунки нумеруются арабскими цифрами сквозной нумерацией внутри раздела и располагаются с абзацного отступа. Рисунки каждого приложения состоят из обозначения приложения, точки и сквозной нумерации, выполненной арабскими цифрами. Вначале располагается сам рисунок, затем подрисуночный текст в виде: Рисунок, номер, тире, наименование рисунка. Например: Рисунок 6.3 – Результат резервного копирования.

Рисунок отделяют от текста интервалом 14 пт.

Не допускается отрыв (перенос со страницы на страницу) рисунка и подрисуночного текста.

3.9 Таблицы

Таблицы нумеруются внутри раздела. Заголовок таблицы оформляют в виде: Таблица, номер, тире, наименование таблицы. Заголовок таблицы выравнивают по левому краю таблицы. Если таблица выходит за размер листа, то ее делят на части. При переносе части таблицы на другой лист заголовок помещают только над первой частью, над остальными частями пишут слова «Продолжение таблицы» с указанием номера таблицы. Шапку таблицы при переносе части таблицы повторяют. Заголовки строк и столбцов пишут с прописной буквы.

Таблицу располагают в записке непосредственно после текста, в котором она упоминается.

Таблицу следует отделять от текста интервалом 14 пт. Допускается выполнять таблицы, размещая их вдоль длинной стороны листа таким

образом, чтобы таблица читалась при повороте листа на 90° по часовой стрелке.

Таблицы слева, справа и снизу ограничивают линиями.

3.10 Ссылки

Ссылки на разделы, подразделы, перечисления, таблицы, иллюстрации, формулы и приложения пояснительной записки следует выполнять по следующим примерам:

– «…структура базы данных, описанная в разделе 2…»;

– «…по пункту б) перечисления…»;

– «…символы приведены в таблице 1.1…»;

– «…изображено на рисунке 3.8…»;

– «…текст процедуры представлен в приложении Д…».

Ссылку на литературный источник выполняют с указанием порядкового номера источника, под которым он внесен в «Список использованных источников» пояснительной записки. Номер ссылки проставляется арабскими цифрами в квадратных скобках, например: [1], [1, 2, 5].

Пример ссылки на источники:

– «…согласно п. 3.4 стандарта [7] …».

3.11 Приложения

Каждое приложение начинается с нового листа с указанием наверху посередине страницы слова «Приложение» с первой прописной буквы и его обозначения.

Приложения по ГОСТ 2.105 обозначаются заглавными буквами русского алфавита, начиная с А, за исключением букв Ё, З, Й, О, Ч, Ь, Ы, Ъ. После слова «Приложение» пишется буква, идентифицирующая его последовательность. Если в документе одно приложение, оно обозначается «Приложение А».

Приложения должны иметь заголовки, которые записывают симметрично тексту с прописной буквы в следующей строке.

Затем через одну пустую строчку следует текст приложения.

3.12 Список использованных источников

Источники – это книги, учебники, статьи из научных журналов и Интернета и т. д., использованные при выполнении курсовой работы. Источники в списке располагаются в порядке ссылок в тексте записки

или по алфавиту, нумеруются арабскими цифрами без точки и печатаются с абзацного отступа, при этом дается библиографическое описание каждого источника в соответствии с ГОСТ 7.1, ГОСТ 7.12.

Общий шаблон описания книги, у которой не более трех авторов: фамилия, запятая, инициалы первого автора, название книги, косая черта, инициалы и фамилии всех авторов, точка, тире, город, двоеточие, издательство, запятая, год издания, точка, тире, количество страниц, буква «с», точка. Название города дается целиком, допустимы только сокращения «М.» (Москва) и «СПб.» (Санкт-Петербург); название издательства – без кавычек. Если у книги более трех авторов, то вначале указывается название книги, а после косой черты – инициалы и фамилия первого автора, затем [и др.].

Пример оформления списка используемых источников приведен ниже.

Список использованных источников

1 Дейт, К. Дж. Введение в системы баз данных / К. Дж. Дейт. – М.: Издательский дом Вильямс, 2005. – 1328 с.

2 Microsoft SQL Server 2012. Реализация и обслуживание. Учебный курс Microsoft: пер. с англ. – М.: Русская редакция; СПб.: Питер, 2014. – 798 с.

3 Проектирование и реализация баз данных Microsoft SQL Server. Учебный курс Microsoft: пер. с англ. – 4-е изд. – М.: Русская редакция; СПб.: Питер, 2012. – 512 с.

4 Коннолли, Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли, К. Бегг. – 3-е изд.; пер. с англ. – М.: Вильямс, 2003. – 1440 с.

4 Примерная тематика курсовых работ

4.1 Общие замечания

Для написания курсовой работы студенту потребуется выбрать СУБД, тематику задачи и технологию для самостоятельного изучения.

Предполагается, что для реализации данной курсовой работы студент выберет СУБД Microsoft SQL Server версии 2012 и выше или Oracle 12c. Возможен выбор других СУБД по согласованию с преподавателем.

Обязательно нужно проконсультироваться с преподавателем по выбору СУБД, задачи и технологии. Для решения некоторых задач, возможно, потребуются специфические технологии.

4.2 Список областей применения баз данных

Разработать базу данных:

– для авиакомпании;
– железнодорожного вокзала;
– агентства недвижимости;
– финансово-консалтингового агентства;
– службы занятости;
– промышленного предприятия;
– страховой компании;
– структуры законодательной власти;
– сети санаторно-курортных учреждений;
– сети кинотеатров;
– сети супермаркетов;
– службы доставки;
– гостиничного комплекса;
– сети предприятий общественного питания;
– сети образовательных учреждений разного уровня (школы, лицеи, колледжи, учреждения высшего образования и т. д.);
– культурно-развлекательного учреждения;
– газетно-журнального издания;
– театра;
– системы тестирования;
– туристического агентства;
– сети сотовой связи;
– электронной площадки по предоставлению услуг;

– аукционной площадки;
– расписания занятий в университете;
– планирования совместных действий (поездок и т. д.);
– медицинского центра;
– физкультурно-оздоровительного центра;
– магазина автомобилей;
– магазина средств связи;
– магазина бытовой техники и т. д.

4.3 Примерный список рекомендуемых технологий для самостоятельного изучения

Если не указано иное, то предполагается, что технология может быть реализована для выбранной СУБД (MS SQL Server, Oracle, PostgreSQL). Предлагаются следующие технологии для самостоятельного изучения и применения в курсовой работе:
– секционирование данных в базе данных;
– установка и применение кластера в СУБД;
– использование SQL Server Profiler;
– применение Reporting Services для визуализации данных в СУБД MS SQL Server;
– применение Analysis Services в СУБД MS SQL Server;
– применение Integration Services в СУБД MS SQL Server;
– применение Master Data Services в СУБД MS SQL Server;
– применение Data Quality Services в СУБД MS SQL Server;
– миграция данных и объектов из одной СУБД в другую;
– разработка системы резервного копирования и восстановления базы данных;
– применение мультимедийных типов данных в базе данных;
– применение пространственных типов данных в базе данных;
– обработка графов в базе данных;
– шифрование и маскирование в базе данных;
– применение полнотекстового поиска в базе данных;
– репликация данных между серверами СУБД;
– настройка системы безопасности сервера СУБД;
– применение In-Memory технологий;
– применение средств мониторинга состояния СУБД;
– применение облачной версии СУБД;
– сравнение SQL и NoSQL решений.

5 Демонстрационный пример

В демонстрационном примере приводятся выдержки из пояснительной записки к курсовой работе студентки 3-го курса. Пример намеренно сокращен и предназначен только для справки. Количество объектов базы данных, ролей и пользователей указано не полностью. Некоторые разделы намеренно отсутствуют. Описание технологии отсутствует. Приложения с SQL-скриптами приводятся сокращенно. При выполнении курсового проекта студент должен руководствоваться своей постановкой задачи.

5.1 Постановка задачи

Для выполнения курсовой работы было спроектировано и реализовано приложение ControlDiet, позволяющее управлять режимом питания. Составной частью приложения является база данных, разработанная в СУБД Microsoft SQL Server 2012.

Одной из главных задач данной курсовой работы является освоение технологии SQL Server Reporting Services, применяемой для построения различных отчетов. Данная технология была выбрана для работы по причине того, что разрабатываемая база данных хранит статистические данные о состоянии пользователей, их приемах питания, по которым удобнее всего создавать отчеты. Данная технология дает возможность просматривать процессы изменения, информацию о конкретных объектах, а также преобразовывать и обрабатывать полученные данные для улучшения восприятия.

На сегодняшний день существует большое количество программных средств, помогающих людям организовывать свой ежедневный рацион, следить за нормами потребления калорий, белков, жиров и углеводов, распределять физическую нагрузку и т. д.

На популярных мобильных платформах есть два приложения: «Здоровье» у Apple и «Google Fit» у Google, которые могут собирать информацию из сторонних приложений, а также позволяют им обмениваться этой информацией между собой.

MyFitnessPal – одно из самых многофункциональных приложений для поддержания здорового образа жизни. В нем есть простой и удобный счетчик калорий с большой базой данных о продуктах, содержащей более 4 млн. наименований. Приложение в состоянии отслеживать физические нагрузки пользователя, например пройденное расстояние, а также планировать программы тренировки.

5.2 Логины и пользователи

Для базы данных ControlDiet было разработано два типа пользователей. Один из них – обычный пользователь приложения, а второй – администратор приложения. Пользователь может просматривать и редактировать только свои данные, администратор может просматривать и редактировать данные всех пользователей, имеет доступ к представлениям, содержащим объединенную информацию о пользователях.

Для авторизации пользователей на сервере СУБД и для подключения к базе данных были разработаны логины.

Далее в рамках базы данных ControlDiet были созданы пользователи, использующие данные логины. Для каждого из пользователей создается роль, включающая необходимые и достаточные разрешения.

SQL-скрипты создания логинов, ролей, пользователей и назначения разрешений приведены в приложении Б.

5.3 Проектирование базы данных

Для хранения и предоставления данных пользователю была разработана база данных, которая содержит в себе 8 таблиц, 7 из которых связаны друг с другом посредством ключей, а одна таблица Audit предназначена для протоколирования действий пользователей в базе данных. Схема базы данных представлена в приложении А.

Таблица Products содержит информацию о продуктах питания: ID, название, категорию, калорийность, БЖУ. Таблица Dishes хранит информацию о готовых блюдах, а именно его название и соответствующий ID.

Таблица Users содержит информацию о зарегистрированных пользователях приложения по управлению режимом питания: их возраст, вес, рост, цель, физическую нагрузку, а также норму калорий, высчитываемую из приведенных выше данных. History – это таблица для хранения информации об изменении веса и параметров отдельно взятых пользователей. Там хранится имя пользователя, дата замеров, параметры вес, объем грудной клетки, талии и бедер.

Таблица Meals и таблица Food хранят информацию о приемах пищи пользователями: в таблицу Meals записывается номер приема питания конкретного пользователя и время данного приема пищи, в таблицу Food записывается информация о том, из каких продуктов состоял данный прием пищи. Так как пользователь может указать продукт или блюдо в качестве пищи, данная таблица осуществляет хранение

продуктов питания. Если было указано блюдо, то оно раскладывается на соответствующие продукты питания и в ID_Dish указывается ID разложенного блюда. Если это был обычный продукт питания, то в ID_Dish записывается «нулевое блюдо», ID которого равно нулю. В данном случае нормализация базы данных намеренно нарушается для удобства обращения к данным.

Таблица Audit хранит данные пользователей для возможности отслеживания или восстановления каких-либо изменений, связанных с пользователями.

SQL-скрипты создания таблиц приведены в приложении Б.

5.4 Процедуры работы с данными

Для реализации доступа к данным были разработаны хранимые процедуры и функции, представленные в таблице 5.1.

Таблица 5.1 – Процедуры и функции

№ п/п	Наименование процедуры или функции	Назначение
7	UpdateUser	Выполняется транзакция вставки данных в таблицу Audit для хранения информации об изменении данных и обновлении данных пользователя

SQL-скрипты создания процедур и функций приведены в приложении В.

5.5 Процедуры извлечения и записи данных в XML

Для реализации импорта и экспорта данных из формата XML были разработаны хранимые процедуры, представленные в таблице 5.2.

Таблица 5.2 – Процедуры извлечения и записи данных в XML

№ п/п	Наименование процедуры или функции	Назначение
1	insertProductFromXML	Выполняется добавление новых данных в таблицу Product из XML-файла. Файл создан заранее и путь к нему передается в качестве параметра данной процедуры

SQL-скрипты создания процедур и функций приведены в приложении В.

5.6 Технология SQL Server Reporting Services (SSRS)

SQL Server Reporting Services (SSRS) – это набор служб для разработки, построения, доставки и просмотра отчетов. С помощью этих служб можно создавать табличные, интерактивные, графические и другие более сложные отчеты с использованием диаграмм и других отчетных элементов.

Для обзора технологии приведено описание создания внедренного отчета SSRS, содержащего информацию о выбранном блюде, его составе и соотношении белков, жиров и углеводов.

Для использования Reporting Services непосредственно из приложения необходимо подключить соответствующие сборки и библиотеки в ссылки проекта. После чего в XAML создать объект WindowsFormsHost, в котором будет отображаться отчет. Далее необходимо получить набор данных, содержащий сведения о калорийности продукта и содержании БЖУ. После этого в проект добавляем новый отчет, а в него источник данных, как изображено на рисунке 5.1.

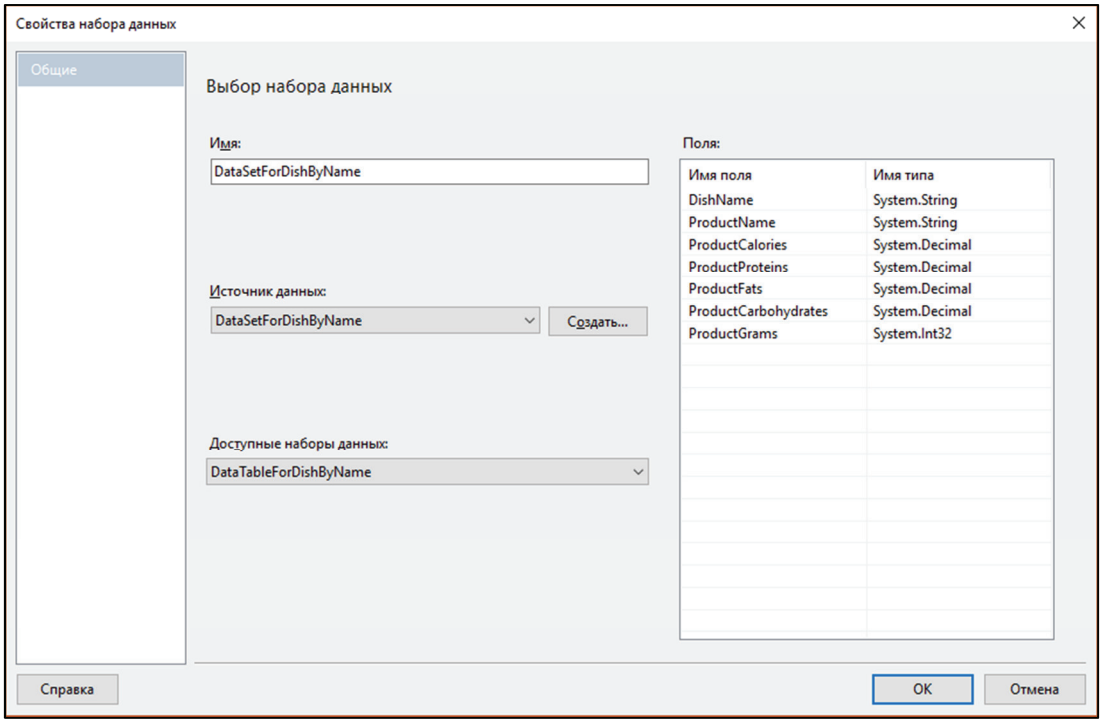

Рисунок 5.1 – Выбор набора данных

Далее отчет форматируется, добавляются стили, итоги, диаграммы. Последним этапом является добавление данного отчета в WPF приложение. Для этого используется функция ReportGetDishByName, код которой приведен на рисунке 5.2.

```
private void ReportGetDishByName_Click(object sender, RoutedEventArgs e)
{
    _reportViewerDishByName.Reset(); //спрасываем область для отображения отчета
    DataTable dt = GetDishByName(); //получаем набор данных
    ReportDataSource ds = new ReportDataSource("DataSetForDishByName", dt); // указываем название набора данных
    _reportViewerDishByName.LocalReport.DataSources.Add(ds); //добавляем данные в отчет
    _reportViewerDishByName.LocalReport.ReportEmbeddedResource = "ControlDiet.ReportForDishByName.rdlc"; //подключаем отчет
    _reportViewerDishByName.RefreshReport(); //загружаем отчет

}
```

Рисунок 5.2 – Функция загрузки отчета

Итогом работы является отчет, представленный на рисунке 5.3.

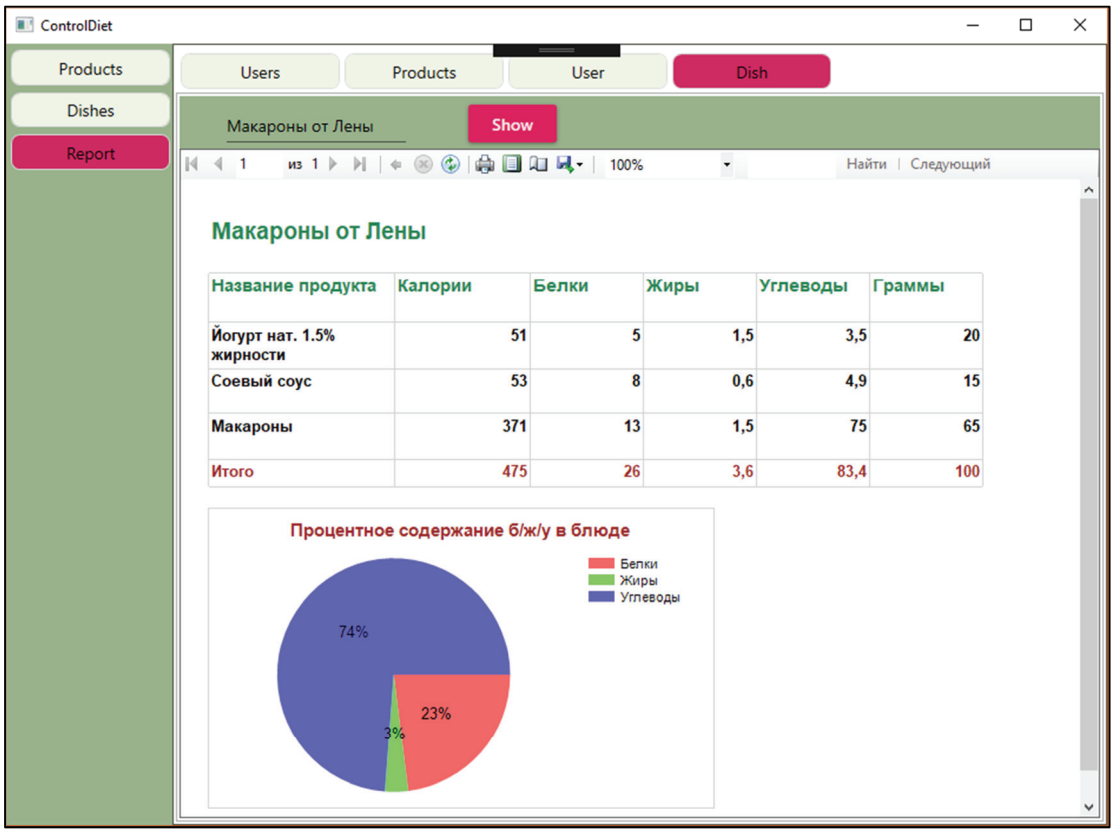

Рисунок 5.3 – Полученный отчет

5.7 Демонстрационное приложение

На вкладке «User» пользователь может просмотреть информацию о себе, а также изменить ее (рисунок 5.4).

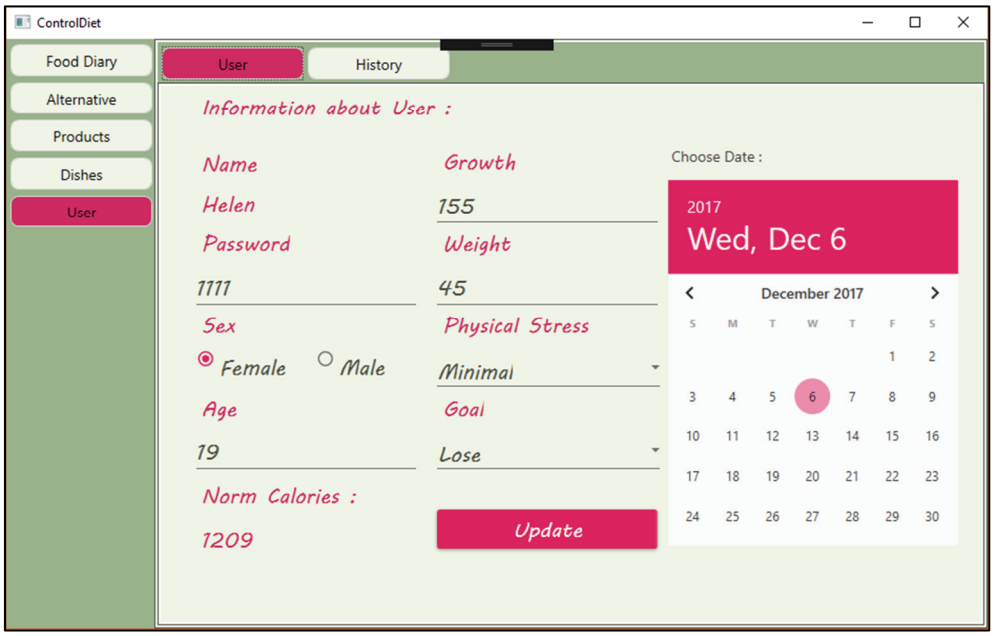

Рисунок 5.4 – Параметры пользователя

По норме калорий высчитываются нормальные количества грамм белков, жиров и углеводов на тот период, а также подводится итог на заданный день, а именно суммируется вся калорийность и все БЖУ. Далее производится построение диаграмм для наглядного отображения разницы требуемых калорий и БЖУ к употребленным. Пример отчета представлен на рисунке 5.5.

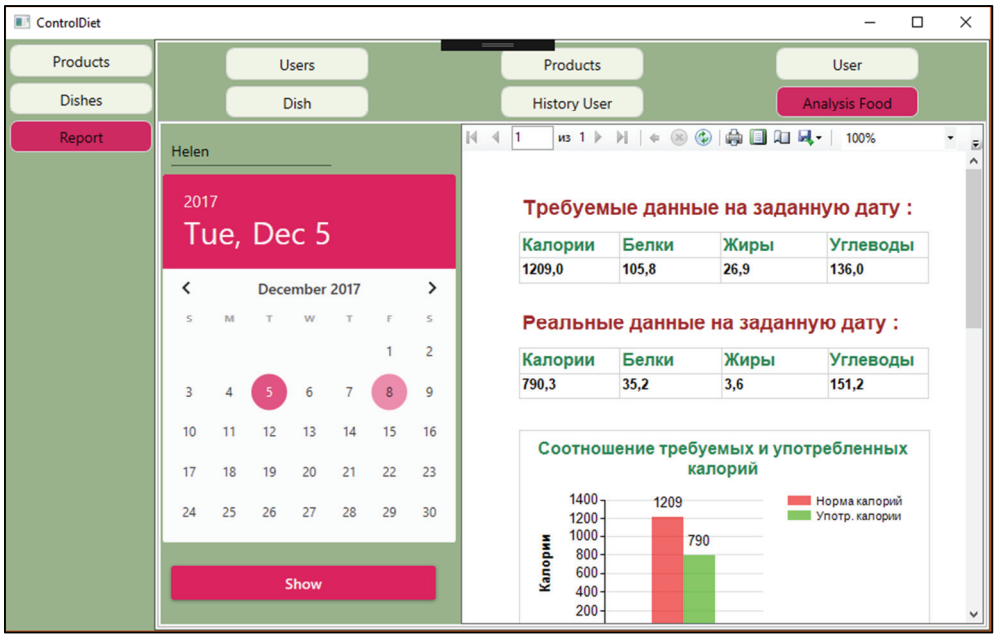

Рисунок 5.5 – Анализ питания

5.8 Приложения

Приложение А
Схема базы данных

Приложение Б
Логины, роли и пользователи

```
--connection login to user
--задаем имя пользователя и пароль
CREATE LOGIN [usercd] WITH PASSWORD=N'1111',
--задаем базу данных и язык
 DEFAULT_DATABASE=[ControlDiet], DEFAULT_LANGUAGE=[Русский],
--указываем политики для истечения время срока пароля и действия пароля
 CHECK_EXPIRATION=OFF, CHECK_POLICY=ON
GO
ALTER LOGIN [usercd] ENABLE
GO
--connection login to admin
CREATE LOGIN [admincd] WITH PASSWORD=N'admincd',
 DEFAULT_DATABASE=[ControlDiet], DEFAULT_LANGUAGE=[Русский],
 CHECK_EXPIRATION=OFF, CHECK_POLICY=ON
GO
ALTER LOGIN [admincd] ENABLE
GO
--создаем пользователей в рамках ControlDiet
CREATE USER [user] FOR LOGIN [usercd];
GO
CREATE USER [admin] FOR LOGIN [admincd];
GO
use ControlDiet;
--создаем роль для пользователей
CREATE ROLE [ControlDiet_user];
--грант на выполнение [SelectUser]
GRANT EXECUTE ON [dbo].[SelectUser]  TO [ControlDiet_user];
--грант на выполнение [AddUser]
GRANT EXECUTE ON [dbo].[AddUser]  TO [ControlDiet_user];
--грант на выполнение [AddMeals]
GRANT EXECUTE ON [dbo].[AddMeals]  TO [ControlDiet_user];
--грант на выполнение [AddFood]
GRANT EXECUTE ON [dbo].[AddFood]  TO [ControlDiet_user];
--грант на выполнение [UpdateUser]
GRANT EXECUTE ON [dbo].[UpdateUser]  TO [ControlDiet_user];
--грант на выполнение [AddHistory]
GRANT EXECUTE ON [dbo].[AddHistory]  TO [ControlDiet_user];
--присваем данную роль пользователю
sp_addrolemember 'ControlDiet_user', 'user';
```

Приложение В
Хранимые процедуры и функции, определенные пользователем

```
--Update User
create procedure UpdateUser -- Создание процедуры
		@UserName varchar(20),		-- Имя пользователя (параметр)
		@Password int,			-- Пароль (параметр)
		@Sex char(1),			-- Пол (параметр)
		@Age int,			-- Возраст (параметр)
		@Growth int,			-- Рост (параметр)
		@Weight int,			-- Вес (параметр)
		@PhysicalStress real,		-- Физическая нагрузка (параметр)
		@Goal real,			-- Цель (параметр)
		@DailyCaloriesNorm int,		-- Норма калорий (параметр)
		@Date Date			-- Дата изменения (параметр)
as  begin -- Начало процедуры
		begin tran -- Начало транзакции
		-- Вставка данных в таблицу Audit для хранения истории данных поль-
зователя
		insert into Audit( UserName, [Date], Age, Growth, [Weight], Physi-
calStress, Goal, DailyCaloriesNorm)
		values ( @UserName, @Date, @Age, @Growth, @Weight,
@PhysicalStress, @Goal, @DailyCaloriesNorm);
		-- Обновления данных пользователя по переданным параметрам
		update Users set UserName=@UserName, [Password]=@Password,
Sex=@Sex, Age=@Age, Growth=@Growth,
		 [Weight]=@Weight, PhysicalStress=@PhysicalStress,
		  Goal=@Goal, DailyCaloriesNorm=@DailyCaloriesNorm
		  where UserName = @UserName;
		commit -- Сохранения транзакции
end; -- Конец процедуры

-- PROC insertProductFromXML
CREATE PROC insertProductFromXML
	@path nvarchar(256) -- передаем путь
AS
begin
	SET NOCOUNT ON  -- отключаем вывод кол-ва обработанных строк
	SET XACT_ABORT ON  -- откат транзакции в случае ошибки
	BEGIN TRAN
		declare @results table (x xml)  -- объявляем таблицу с полем xml
	-- считывание данные с удаленного источника (xml-файла)
	-- BULK - загрузка большого набора данных,
	-- SINGLE_BLOB - возвращает содержимое в виде 1строки 1столбца бинар-
ных данных
		declare @sql nvarchar(300)=
```

```
                    'SELECT
                CAST(REPLACE(CAST(x AS VARCHAR(MAX)), "encoding="utf-16"", "encoding="utf-8"") AS XML)
                FROM OPENROWSET(BULK '"+@path+"', SINGLE_BLOB) AS T(x)';
        INSERT INTO @results EXEC (@sql)--выполняем и вставляем во временную таблицу
        declare @xml XML = (SELECT  TOP 1 x from  @results); -- записываем xml в xml-переменную
        INSERT INTO Products( Name, Category, Calories, Proteins, Fats, Carbohydrates)
                SELECT
                C3.value('Name[1]', 'varchar(30)') AS Name, -- создаем именование
                C3.value('Category[1]', 'varchar(30)') AS Category,
                C3.value('Calories[1]', 'float') AS Calories,
                C3.value('Proteins[1]', 'float') AS Proteins,
                C3.value('Fats[1]', 'float') AS Fats,
                C3.value('Carbohydrates[1]', 'float') AS Carbohydrates
                FROM @xml.nodes('products/product') AS T3(C3) -- формирование результирующих строк из метода nodes
    COMMIT;
end;
```

Приложение А (обязательное)

Образец оформления титульного листа

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий
Кафедра информационных систем и технологий
Специальность 1-40 01 01 Программное обеспечение информационных технологий
Специализация Программирование интернет-приложений

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К КУРСОВОЙ РАБОТЕ НА ТЕМУ:

_____
_____

Выполнил студент Фамилия Имя Отчество

Руководитель проекта Фамилия Имя Отчество

Заведующий кафедрой канд. техн. наук, доц. Смелов В.В.

Консультанты Фамилия Имя Отчество

Курсовой проект защищен с оценкой _____

Минск 201_

Приложение Б (обязательное)

Образец оформления листа задания

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий
Кафедра информационных систем и технологий

Утверждаю
Заведующий кафедрой
_____ В.В. Смелов
«___» _____201_г.

ЗАДАНИЕ
на курсовую работу по дисциплине «Базы данных»

Специальность 1-40 01 01 Программное обеспечение информационных технологий
Группа_____
Студент_____
Тема_____

1. Срок сдачи студентом законченной работы: «____» декабря 201__г.

2. Требования к работе:

2.1. Функционально должны быть выполнены следующие бизнес-задачи:

_____
_____
_____
_____

2.2. Прочие требования.

2.2.1. База данных должна быть спроектирована в СУБД _____.
2.2.2. Доступ к данным должен осуществляться только через соответствующие процедуры.
2.2.3. Должен быть проведен импорт данных из файлов формата _____, экспорт данных в формат _____.
2.2.4. Необходимо протестировать производительность базы данных (на таблицах, содержащих не менее 100 000 строк) и внести изменения в структуру в случае необходимости.
2.2.5. Необходимо при решении бизнес-задачи применить технологию базы данных согласно выбранной теме.
2.2.6. Листинги скриптов должны содержать комментарии.

3. Содержание расчетно-пояснительной записки

3.1. Введение

3.2. Постановка задачи (выбор СУБД, область решения бизнес-задачи, используемая технология, аналоги)

3.3. Установка и настройка сервера СУБД

3.4. Проектирование инфраструктуры базы данных (пользователи, файлы и пр.)

3.5. Проектирование и реализация объектов базы данных (таблицы, представления, индексы, хранимые процедуры и функции, триггеры и пр.)

3.6. Описание применения технологии в созданной базе данных

3.7. Импорт и генерация тестовых данных

3.8. Резервное копирование и восстановление данных

3.9. Демонстрационное приложение

3.10. Заключение

3.11. Список используемых источников

3.12. Приложения

4. Форма представления выполненного курсового проекта

4.1. Теоретическая часть курсового проекта должна быть представлена в формате MS Word согласно _____.

4.2. Необходимые схемы, диаграммы и рисунки допускается делать в MS Office Visio или PrintScreen.

4.3. Листинги скриптов представляются в приложении.

4.4. К записке необходимо приложить CD (DVD), который должен содержать: пояснительную записку, SQL-скрипты, файлы базы данных и приложение.

Календарный план

№ п/п	Наименование этапов курсовой работы	Срок выполнения этапов
1	Аналитический обзор литературы по теме работы	
2	Проектирование инфраструктуры и модели базы данных	
3	Проектирование и реализация объектов базы данных	
4	Применение технологии в базе данных	
5	Импорт и экспорт данных, тестирование производительности	
6	Резервное копирование и восстановление данных базы данных	
7	Разработка демонстрационного приложения	
8	Оформление пояснительной записки	

5. Дата выдачи задания «_____» сентября 201__ г.

Руководитель _____ _____
подпись инициалы и фамилия преподавателя

Задание принял
к исполнению _____ _____
подпись инициалы и фамилия студента

Учебное издание

Блинова Евгения Александровна
Мороз Леонарда Станиславовна

БАЗЫ ДАННЫХ

Учебно-методическое пособие

Редактор *О. П. Приходько*
Компьютерная верстка *О. П. Приходько*
Корректор *О. П. Приходько*

Издатель:
УО «Белорусский государственный технологический университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/227 от 20.03.2014.
Ул. Свердлова, 13а, 220006, г. Минск.