**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КОМИ**

**Государственное профессиональное образовательное учреждение**

**«Воркутинский арктический горно-политехнический колледж»**

**КУРСОВОЙ ПРОЕКТ**

По дисциплине МДК.07.01 Управление и автоматизация баз данных

**Разработка базы данных**

**«Учет расчётов за проживание в общежитии»**

Выполнил студент гр. ИСП-20 /\_\_\_\_\_\_\_\_\_\_\_/Ефремова Анастасия Сергеевна/

(подпись) (Ф.И.О.)

**ОЦЕНКА:\_\_\_\_\_\_\_\_\_\_\_\_\_**

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОВЕРИЛ**

Научный руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_/ Егоров Данил Павлович/

(подпись) (Ф.И.О.)

Воркута

2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc153484562)

[ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ 4](#_Toc153484563)

[1.1 СУБД SQL server 4](#_Toc153484564)

[1.2 Microsoft SQL Server Management Studio 5](#_Toc153484565)

[1.3 Entity Framework 6](#_Toc153484566)

[ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 8](#_Toc153484567)

[2.1 Разработка диаграммы ERD 8](#_Toc153484568)

[2.2 Разработка базы данных 10](#_Toc153484569)

[2.3 Словарь данных 16](#_Toc153484570)

[ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ 20](#_Toc153484571)

[3.1 Введение в безопасность SQL Server 20](#_Toc153484572)

[4.2 Управление безопасностью уровня сервера 21](#_Toc153484573)

[3.3 Управление участниками уровня базы данных 25](#_Toc153484574)

[3.4 Управление разрешениями уровня базы данных 26](#_Toc153484575)

[3.5 Шифрование данных баз данных 28](#_Toc153484576)

[ЗАКЛЮЧЕНИЕ 31](#_Toc153484577)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 32](#_Toc153484578)

**ВВЕДЕНИЕ**

В среде программных продуктов очень много программ для учёта информации и финансов в офисе, в учреждении или на предприятии.

Основными особенностями специализированных продуктов, предназначенных для отслеживания информации и расчётов финансов – удобство, простота использования и многофункциональность.

Часто имеющиеся функции автоматизированной информационной системы с соответствующей базой данных на рынке программных продуктов избыточны. Их наличие ведёт к повышению системных требований для обеспечения функционирования данных систем и к значительному увеличению расходов у потребителя на эксплуатации данных программных продуктов.

Поэтому нужно создать программный продукт, который имеет необходимых минимальный функционал для учёта информации и финансов, который можно приобрести по доступной цене.

Объект курсовой проекта – создание программы, взаимодействующая с базой данных через различные инструменты программирования.

Предмет курсовой проект – создание с помощью программирования автоматизированной информационной системы «Учёт расчетов за проживание в общежитии» и создание.

Целью курсового проекта является создание базы данных для автоматизированного учета оплат проживания жильцов общежития.

Для достижения цели проекта нужно выполнить следующие задачи:

1. на основании технического задания выбираются определённый инструментарий;
2. в соответствии с техническим заданием спроектировать базу данных;
3. сделать шифрование данных;
4. разработать информационную систему, работающая с базой данных.

**ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ**

**1.1 СУБД SQL server**

**SQL** (от англ. Structured Query Language — «язык структурированных запросов») — декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных. Является информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных. Изначально SQL был основным способом работы пользователя с базой данных и позволял выполнять следующий набор операций:

1. Создание в базе данных новой таблицы;
2. добавление в таблицу новых записей;
3. Изменение записей;
4. Удаление записей;
5. Выборка записей из одной или нескольких таблиц (в соответствии с заданным условием);
6. Изменение структур таблиц.

Появление баз данных позволило объединить разное понимание БД со стороны пользователей и системных администраторов. Неискушенные в технических деталях люди «видят» таблицы как некий перечень данных с колонками и строками. Системный подход включает файлы с табличными данными, связанными друг с другом согласно определенному алгоритму.

Клиентами БД являются прикладные программы, их интерфейс, различные интерактивные модули сайтов вроде калькуляторов и онлайн-редакторов. Но есть еще один компонент системы – СУБД. Он предназначен для ручного доступа к информации и позволяет извлекать данные, работать с ними в памяти сервера, в том числе с применением языка SQL.

Со временем SQL усложнился — обогатился новыми конструкциями, обеспечил возможность описания и управления новыми хранимыми объектами (индексы, представления, триггеры и хранимые процедуры) — и стал приобретать черты, свойственные языкам программирования. При всех своих изменениях SQL остаётся самым распространённым лингвистическим средством для взаимодействия прикладного программного обеспечения с базами данных. В то же время современные СУБД, а также информационные системы, использующие СУБД, предоставляют пользователю развитые средства визуального построения запросов.

**1.2 Microsoft SQL Server Management Studio**

**SQL Server Management Studio** (**SSMS**) — утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера.

Среда SSMS предназначена для доступа к службам Службы Analysis Services, Integration Services и Reporting Services, а также для их настройки, администрирования и управления ими.

Главным инструментом SSMS является Object Explorer, который позволяет пользователю просматривать, извлекать объекты сервера, а также полностью ими управлять.

Среда SSMS предоставляет единую комплексную служебную программу, которая сочетает в себе обширную группу графических инструментов с рядом многофункциональных редакторов скриптов для доступа к SQL Server для разработчиков и администраторов баз данных всех профессиональных уровней.

Основной состав функционала SSMS:

1. Подключение к службам SQL Server;
2. Редактор SQL кода;
3. Обозреватель решений;
4. Конструктор таблиц;
5. Конструктор баз данных (Диаграммы баз данных);
6. Конструктор запросов и представлений;
7. Просмотр свойств объектов;
8. Мастер создания скриптов;
9. Управление безопасностью SQL Server;
10. Присоединение и отсоединение баз данных;
11. Создание резервных копий баз данных;
12. Восстановление баз данных из архива.

**1.3 Entity Framework**

**Entity Framework** - специальная объектно-ориентированная технология на базе фреймворка .NET для работы с данными. Первоначально он поставлялся как неотъемлемая часть .NET Framework, однако, начиная с Entity Framework версии 6.0, он поставлялся отдельно от .NET Framework.

Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework — это более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

Entity Framework представляет собой набор технологий в ADO.NET, который поддерживает разработку программных приложений, ориентированных на данные. Архитекторы и разработчики приложений, ориентированных на данные, обычно борются с необходимостью достижения двух совершенно разных целей. Они должны моделировать сущности, связи и логику бизнес-задач, которые они решают, а также должны работать с обработчиками данных, используемыми для хранения и извлечения данных. Данные могут охватывать несколько систем хранения, каждая из которых имеет свои собственные протоколы; даже приложения, работающие с одной системой хранения, должны уравновешивать требования системы хранения данных с требованиями написания эффективного и поддерживаемого кода приложения. Эту проблему обычно называют «несоответствием объекта и реляционного импеданса».

Многие инструменты объектно-реляционного сопоставления (ORM) (также известные как «объектно-реляционные менеджеры») были разработаны, чтобы позволить разработчикам работать с данными в виде объектов и свойств, специфичных для предметной области, таких как клиенты и адреса клиентов, без необходимости беспокоиться о базовых таблицах и столбцах базы данных, где хранятся эти данные. С ORM разработчики могут работать на более высоком уровне абстракции, когда они имеют дело с данными, и могут создавать и поддерживать ориентированные на данные приложения с меньшим количеством кода, чем в традиционных приложениях. Entity Framework — это решение ORM, которое в настоящее время продвигается для использования в стеке разработки Майкрософт.

**ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ**

**2.1 Разработка диаграммы ERD**

Диаграмма ERD или ER-модель (от англ. Entity-Relationship model, модель «сущность — связь») — модель данных, позволяющая описывать концептуальные схемы предметной области. Это разновидность блок-схемы, где показано, как разные «сущности» (люди, объекты, концепции и так далее) связаны между собой внутри системы. ER-диаграммы чаще всего применяются для проектирования и отладки реляционных баз данных в сфере образования, исследования, разработки программного обеспечения и информационных систем для бизнеса.

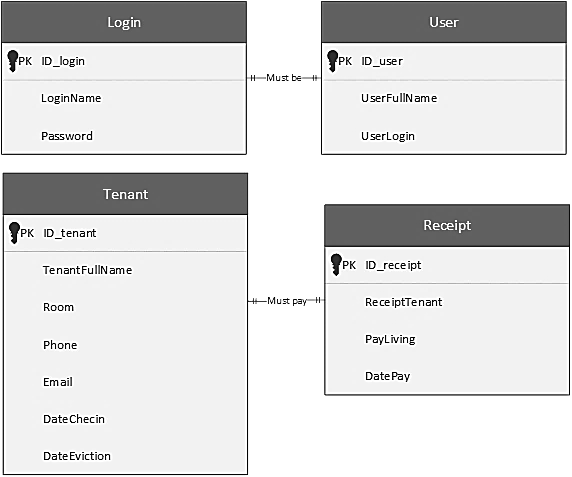
Будет использована графическая нотация (стиль оформления диаграммы) Crow’s Foot. Данная нотация была предложена Гордоном Эверестом под названием “inverted arrow” («перевёрнутая стрелка»), однако сейчас чаще называемая “Crow’s Foot”, или “crow’s foot” («воронья лапка») или “fork” («вилка»).

Согласно данной нотации, сущность изображается в виде прямоугольника, содержащего её имя, выражаемое существительным. Имя сущности должно быть уникальным в рамках одной модели. При этом, имя сущности — это имя типа, а не конкретного экземпляра данного типа. Экземпляром сущности называется конкретный представитель данной сущности.

Связь изображается линией, которая связывает две сущности. Степень конца связи указывается графически, множественность связи изображается в виде «вилки» на конце связи. Модальность связи также изображается графически — необязательность связи помечается кружком на конце связи. Наименование обычно выражается одним глаголом в изъявительном наклонении настоящего времени: «имеет», «принадлежит» и т. д.; или глаголом с поясняющими словами: «включает в себя», и т. п. Наименование может быть одно для всей связи или два для каждого из концов связи. Во втором случае, название левого конца связи указывается над линией связи, а правого — под линией. Каждое из названий располагаются рядом с сущностью, к которой оно относится.

Атрибуты сущности записываются внутри прямоугольника, изображающего сущность, и выражаются существительным в единственном числе (возможно, с уточняющими словами). Среди атрибутов выделяется ключ сущности — неизбыточный набор атрибутов, значения которых в совокупности являются уникальными для каждого экземпляра сущности.

Создадим ERD диаграмму для создания представления об базе данных для автоматизированной информационной системе «Учёт расчетов за проживание в общежитии».



*Рисунок 2.1 - ERD-диаграмма «Учёт расчетов за проживание в общежитии»*

В данной ERD-диаграмме есть следующие сущности: «Логин» («Login»), «Пользователь» («User»), «Житель» («Tenant») и «Квитанция» («Receipt»).

Сущность «Login» имеет следующие атрибуты: ID\_login – индификационный номер сущности, LoginName – логин пользователя для входа в систему, Password – пароль пользователя для входа в систему. «Login» должен быть связан с одним «Account».

Сущность «User» имеет следующие атрибуты: ID\_Account – индификайионный номер сущности, UserFullName – ФИО пользователя системы, UserLogin – внешний ключ для связки с сущностью Login для связывания пользователя с его логином, паролем и ролью пользователя. «Account» должен иметь один «Login».

Сущность «Tenant» имеет следующие атрибуты: ID\_tenant – индикационный номер сущности, FullNameTenant – ФИО жильца, Room– комната жильца, Phone - телефон жильца, Email – почта жильца, DateChecin – дата заселения жильца, DateEviction – дата выселения жильца. «Tenant» должен платить один «Receipt».

Сущность «Receipt» имеет следующие атрибуты: ID\_Receipt – индификационный номер сущности, TenantReceipt – номер списка жильца в общежитии в обще списке жильцов, PayLiving – оплата жильца за проживание, DataPay – дата оплаты квитанции. «Receipt» должен быть оплачен одним «Tenant».

**2.2 Разработка базы данных**

База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. Данные вместе с СУБД, а также приложения, которые с ними связаны, называются системой баз данных, или просто базой данных.

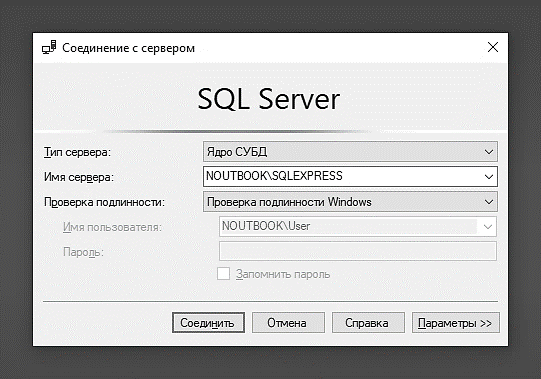
Для базы данных обычно требуется комплексное программное обеспечение, которое называется системой управления базами данных (СУБД). СУБД служит интерфейсом между базой данных и пользователями или программами. СУБД обеспечивает контроль и управление данными, позволяя выполнять различные административные операции, такие как мониторинг производительности, настройка, а также резервное копирование и восстановление.

В качестве примеров популярного программного обеспечения для управления базами данных, или СУБД, можно назвать MySQL, Microsoft Access, 1.5, Microsoft SQL Server Management Studio (MSSMS), СУБД Oracle Database и dBASE.

Данные в наиболее распространенных типах современных баз данных обычно хранятся в виде строк и столбцов, формирующих таблицу. Этими данными можно легко управлять, изменять, обновлять, контролировать и упорядочивать. В большинстве баз данных для записи и запросов данных используется язык структурированных запросов (SQL).

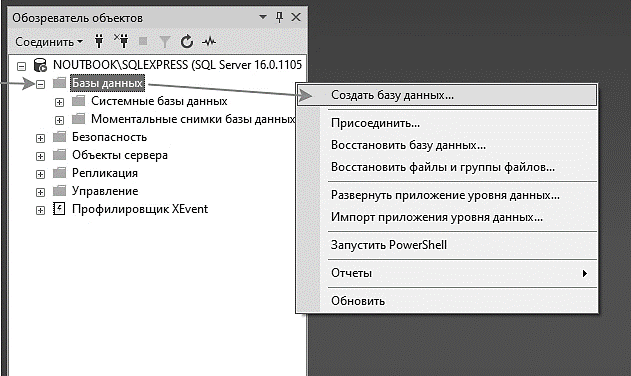
Создадим базу данных для курсового проекта «Учёт расчетов за проживание в общежитии». Воспользуемся программой Microsoft SQL Server Management Studio для этой задачи.

Подключимся к серверу базы данных. Именно здесь будет создаваться база данных «Учёт расчетов за проживание в общежитии». Вводим имя сервера и нажимаем на кнопку «Соединить».



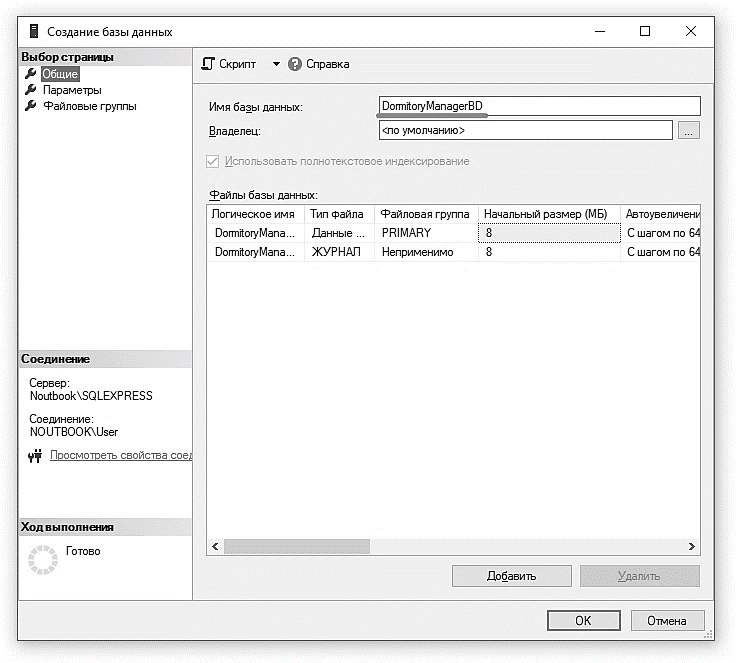
*Рисунок 2.2 - окно подключения к серверу в Microsoft SQL Server Management Studio*

Создадим базу данных под именем «DormitoryManagerBD». Для этого ищем в списке сервера папку «Базы данных», нажимаем правой кнопки мышки на него и из контекстного списка выбираем пункт «Создать базу данных…».



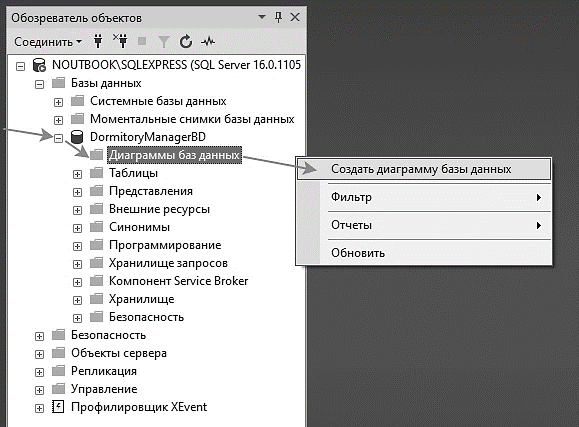
*Рисунок 2.*3–шаги к созданию базы данных

После чего появляется окно создания базы данных на текущем сервере. Тут достаточно ввести название базы данных и нажать на кнопку «ОК».

**

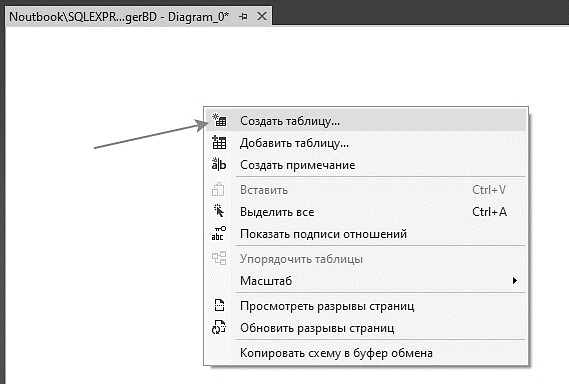
*Рисунок 2.*4–окно создания базы данных

Теперь можно добавить таблицы в созданную базу данных и в них же задавать параметры столбцов (выбирать тип данных у полей таблицы). Это проще всего сделать через создание и редактирование диаграмм баз данных. В списке нашей базы данных ищем папку «Диаграммы баз данных», нажимаем на него правой кнопкой мышки и в контекстном списке выбираем «Создать диаграмму базы данных».

**

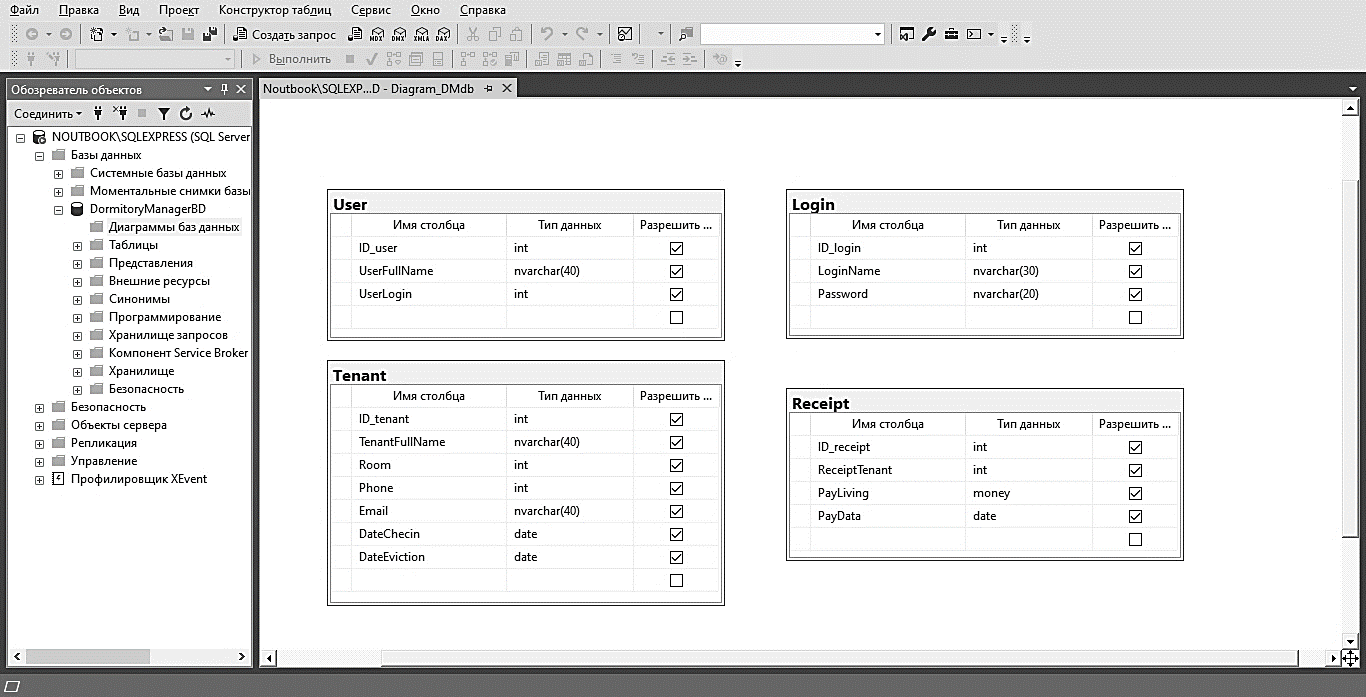
*Рисунок 2.4 –* ш*аги к созданию диаграммы базы данных*

По пространству файла диаграммы нажимаем правой кнопкой и из контекстного меню выбираем «Создать таблицу».



*Рисунок 2.5 – шаги к созданию базы данных*

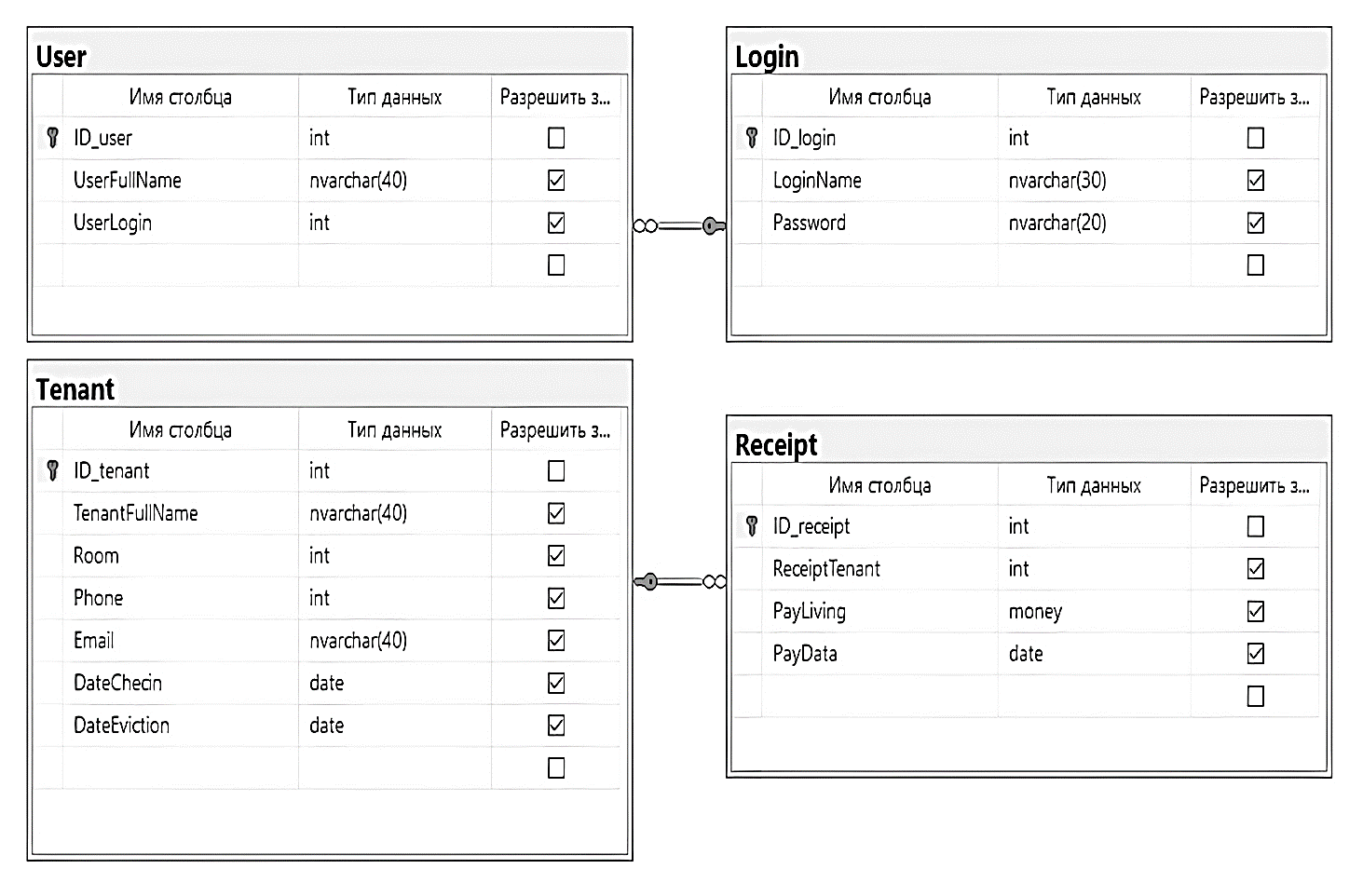
Затем появляется окно для ввода имя таблицы. Делаем так ещё 3 раза и получим 4 таблицы. Заполняем пустые поля нужными нам столбцами и выбираем из списка соответствующими типами данных.

**

*Рисунок 2.6 –* создание необходимых таблиц базы данных

Теперь необходимо устанавливать связи между таблицами. Сначала нужно указать первичный ключ для каждой таблицы. Для этого нажимаем правой кнопкой мышки на нужную строку, обычно это первая строка, которая является уникальным идентификатором таблицы, и из контекстного меню выбираем пункт «Задать первичный ключ».

Теперь можно сделать необходимые связи между таблицами, соединяя строку с первичным ключом одной таблицы с строку без первичного ключа другой таблицы. Строку «ID\_login» таблицы «Login» привязываем к строке «UserLogin» таблицы «User» - отсутствие логина и пароля не позволит пользователю войти, а строку «ID\_tenant» таблицы «Tenant» ассоциируем с строкой «ReceiptTenant» таблицы «Receipt» - данные жильца потребуются при формировании квитанции и других отчётов.

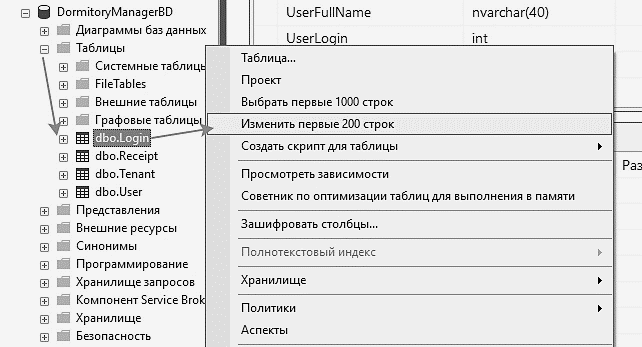


*Рисунок 2.7 - диаграмма таблиц в базе данных «DormitoryManagerBD»*

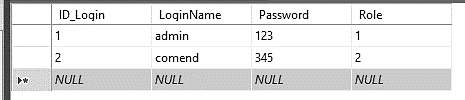
В итоге получилась диаграмма двух связный пар таблиц. Одна пар связана с пользователями системы и базы данных, а другая с данными о жильцах общежития и их квитанций.

Основа базы данных создана. Теперь осталось дополнить её нужной нам информацией в её таблицах. Для этого в папке базы данных раскрываем папку таблиц, нажимаем одну из имеющихся таблиц правой кнопкой мышки и из контекстного меню выбираем пункт «Изменить первые 200 строк» - откроется файл для редактирования данных в выбранной таблице.

.

**

*Рисунок 2.8 – Открытие строк таблицы для редактирования*

**

*Рисунок 2.9 - данные таблицы «Login»*

Таким же образом мы заполняем другие 3 таблицы необходимыми данными. Теперь база данных «DormitoryManagerBD» готова к использованию для курсового проекта «Учёт расчетов за проживание в общежитии».

**2.3 Словарь данных**

**Словарь данных** — это центральное хранилище информации о данных, такой как значение, взаимосвязи с другими данными, их источник, применение и формат. Этот словарь хранит метаданные, такие как структура таблиц, типы данных, ограничения целостности, индексы и другие характеристики базы данных.

**Метаданные** — это данные о данных, такие как имена таблиц, столбцов, типов данных, ограничений, правил и другие составляющие основы и функциональности баз данных.

Каждая база данных управляющей системы (например, MySQL, PostgreSQL, Microsoft SQL Server, Oracle) имеет свой собственный словарь данных. Этот словарь позволяет системе управления базами данных (СУБД) отслеживать и управлять метаданными, необходимыми для правильной работы базы данных.

В SQL и базах данных словарь данных обычно состоит из следующих объектов:

* **Таблицы**: Таблицы словаря данных обычно имеют имена, начинающиеся с префикса "sys". Эти таблицы содержат информацию о структуре базы данных, такую как имена таблиц, столбцов, типов данных, ограничений и правил.
* **Просмотры**: Просмотры словаря данных обычно предоставляют пользователям более удобный способ просмотра информации из таблиц словаря данных. Похожи на диаграммы или схемы, но в отличии от картинки способен влиять на метаданные базы данных.
* **Функции**: Функции словаря данных обычно используются для выполнения операций над метаданными.
* **Хранимые процедуры**: Хранимые процедуры словаря данных обычно используются для выполнения операций над метаданными.
* **Триггеры**: Триггеры словаря данных обычно используются для выполнения действий при изменении метаданных.

Составим словарь данных об базе данных «DormitoryManagerBD» для этого курсового проекта. Для этого напишем имеющейся таблицы и их основные типы данных. Всего таблиц 4 и поэтому объектов типа таблицы тоже будет 4: таблица «Login», таблица «User», таблица «Tenant», таблица «Receipt»

Также кратко распишем такие поля этих таблиц как:

* Типы ключей (нету, первичный или внешний);
* Наименование поля;
* Тип данных / Размер поля;
* Статус обязательного заполнения ячейки таблицы;
* Краткое описание поля.

Таблица 1. Объект Login

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data Dictionary | | | | |
| **Login** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| Primary | **ID\_Login** | INT | Y | Уникальный идентификатор |
|  | **LoginName** | NVARCHAR (30) | N | Наименование типа пользователя |
|  | **Password** | NVARCHAR(20) | N | Пароль для типа пользователя |

Таблица 2. Объект User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| Primary | **ID\_User** | INT | Y | Уникальный идентификатор |
|  | **FullNameUser** | NVARCHAR (30) | N | Полное ФИО пользователя программы |
| Foreign | **LoginUser** | INT | Y | Вторичный ключ для связывания с таблицей Login |

Таблица 3. Объект Tenant

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tenant** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| Primary | **ID\_Tenant** | INT | Y | Уникальный идентификатор |
|  | **FullNameTenant** | NVARCHAR (40) | N | Полное ФИО жителей |
|  | **RoomTenant** | INT | N | Номер комнаты жителя |
|  | **Phone** | INT | N | Номер телефона жителя |
|  | **Email** | NVARCHAR (40) | N | Электронная почта жителя |
|  | **DataChecin** | DATE | N | Дата заселения жителя |
|  | **DataEviction** | DATE | N | Дата выселения жителя |

Таблица 4. Объект Receipt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Receipt | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| Primary | **ID\_Receipt** | INT | Y | Уникальный идентификатор |
| Foreign | **TenantReceipt** | INT | Y | Порядковый номер жильца, к которому привязано квитанция |
|  | **PayLiving** | MONEY | N | Цена стоимости проживания жителя в общежитии |
|  | **PayAddService** | MONEY | N | Цена стоимости дополнительных услуг |
|  | **DataPay** | DATE | N | Дата выплаты итоговой суммы оплаты за проживание |

**ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ**

**3.1 Введение в безопасность SQL Server**

Безопасность SQL Server представляет собой важный аспект, обеспечивающий защиту конфиденциальности данных, целостность информации и обеспечение доступа только авторизованным пользователям. В данном контексте мы рассмотрим четыре ключевых аспекта безопасности SQL Server, связанных с управлением уровнем сервера и базы данных:

***Управление безопасностью уровня сервера****.* На уровне сервера SQL, важно правильно настроить аутентификацию и авторизацию пользователей. Обеспечение безопасности включает в себя создание безопасных паролей, управление учетными записями, и регулярное обновление политик безопасности сервера. Грамотное использование механизмов аутентификации помогает предотвратить несанкционированный доступ к серверу баз данных.

***Управление участниками уровня базы данных****.* На уровне базы данных, определение ролей и управление правами доступа к данным - критически важный аспект. Создание ролей, присвоение им конкретных функций, и управление правами обеспечивают гибкий контроль над тем, кто и как может взаимодействовать с данными в контексте учёта расчетов.

***Управление разрешениями уровня базы данных.*** Эффективное управление разрешениями в базе данных предоставляет точный контроль над доступом к объектам данных. Это включает в себя присвоение прав на уровне таблиц, представлений и хранимых процедур. Важно определить, какие пользователи или группы имеют доступ к данным расчетов, а какие - нет.

***Шифрование данных баз данных****.* Одним из важных методов обеспечения конфиденциальности данных является шифрование. Шифрование данных в базе данных SQL гарантирует, что даже в случае несанкционированного доступа к хранимой информации, данные остаются защищенными. Это может включать как в себя шифрование всей базы данных, так и отдельных чувствительных деталей базы данных.

Учитывая вышеуказанные факторы, надо настроить безопасность в SSMS для повышения безопасности базы данных этого курсового проекта.

* 1. **Управление безопасностью уровня сервера**

Безопасность сервера SQL Server является важной составляющей общей безопасности системы. Уязвимости на уровне сервера могут привести к потере данных, утечке конфиденциальной информации или даже к отказу в обслуживании.

Есть два основных варианта управления безопасностью уровня сервера в SQL Server:

* Резервное копирование и восстановление данных
* Создание пользователей

***Резервное копирование данных***

Резервное копирование данных является одним из важнейших аспектов обеспечения безопасности баз данных. Оно позволяет восстановить данные в случае их утраты или повреждения. В MSSMS резервное копирование данных можно выполнять с помощью следующих методов:

* **Резервное копирование базы данных**. Этот метод позволяет создать полную резервную копию базы данных, включая все ее таблицы, индексы, хранимые процедуры, триггеры и т.д.

**```**sql

-- Использование базы данных DormitoryManagerBD

USE DormitoryManagerBD

Go

-- Резервное копирование базы данных DormitoryManagerBD на диск D

BACKUP DATABASE DormitoryManagerBD

TO DISK = 'DormitoryManagerBD1.bak'

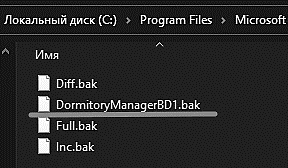
-- Восстановление базы данных DormitoryManagerBD из резервной копии на диске D

RESTORE DATABASE DormitoryManagerBD

FROM DISK = 'DormitoryManagerBD1.bak'

WITH FILE = 1

**```**

****

*Рисунок 3.1 – Файл BackUp базы данных DormitoryManagerBD*

* **Резервное копирование журнала транзакций.** Этот метод позволяет создать резервную копию журнала транзакций базы данных. Журнал транзакций используется для восстановления базы данных в случае ее повреждения или утраты.

**```**sql

-- Использование базы данных DormitoryManagerBD

USE DormitoryManagerBD

Go

-- Создание резервной копии базы данных DormitoryManagerBD

BACKUP DATABASE DormitoryManagerBD

TO DISK = 'D:\Backups\DormitoryManagerBDlog1.bak'

GO

-- Создание резервной копии журнала транзакций DormitoryManagerBD

BACKUP LOG DormitoryManagerBD

TO DISK = 'D:\Backups\DormitoryManagerBDlog1.trn';

GO

-- Восстановление базы данных DormitoryManagerBD

ALTER DATABASE DormitoryManagerBD SET SINGLE\_USER WITH ROLLBACK IMMEDIATE;

RESTORE DATABASE DormitoryManagerBD

FROM DISK = 'D:\Backups\DormitoryManagerBD.Bak'

WITH NORECOVERY;

-- Восстановление журнала транзакций DormitoryManagerBD

RESTORE LOG DormitoryManagerBD

FROM DISK = 'D:\Backups\DormitoryManagerBD.TRN'

WITH RECOVERY;

**```**

***Создание пользователей***

Пользователи являются основными объектами безопасности в SQL Server. Они предоставляют доступ к данным и функциям сервера.

В SQL Server пользователи могут быть созданы в двух уровнях:

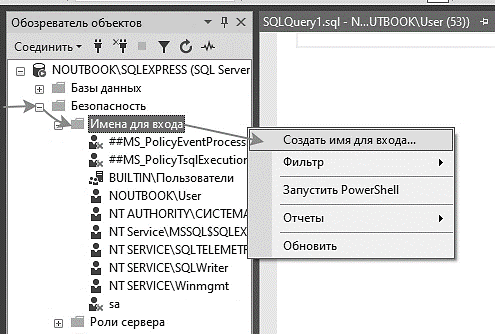
* **Уровень сервера**

Пользователи уровня сервера имеют доступ ко всем базам данных на сервере. Они могут создавать, изменять и удалять базы данных, а также создавать и удалять пользователей базы данных.

* **Уровень базы данных**

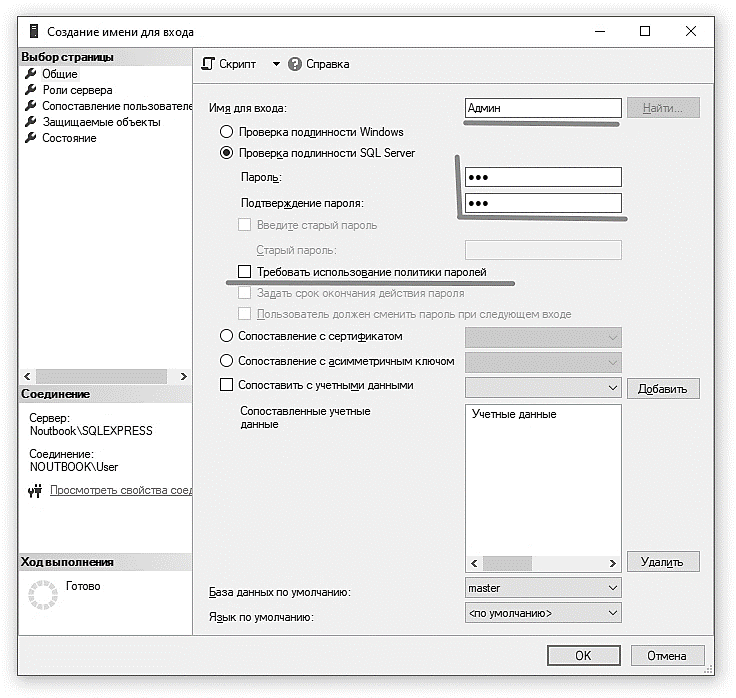
Пользователи уровня базы данных имеют доступ только к одной базе данных. Они могут выполнять операции с данными и функциями в этой базе данных.

Для курсового проекта подойдёт создать пользователя баз данных, так как мы пользуемся и работаем с конкретной базой данных. Чтобы его создать в папке «Безопасность» ищем папку и наживем правой кнопкой мышки на «Имена для входа» и в контекстном меню нажимаем на пункт «Создать для входа…».

**

*Рисунок 3.2 – Шаги создания пользователя базы данных*

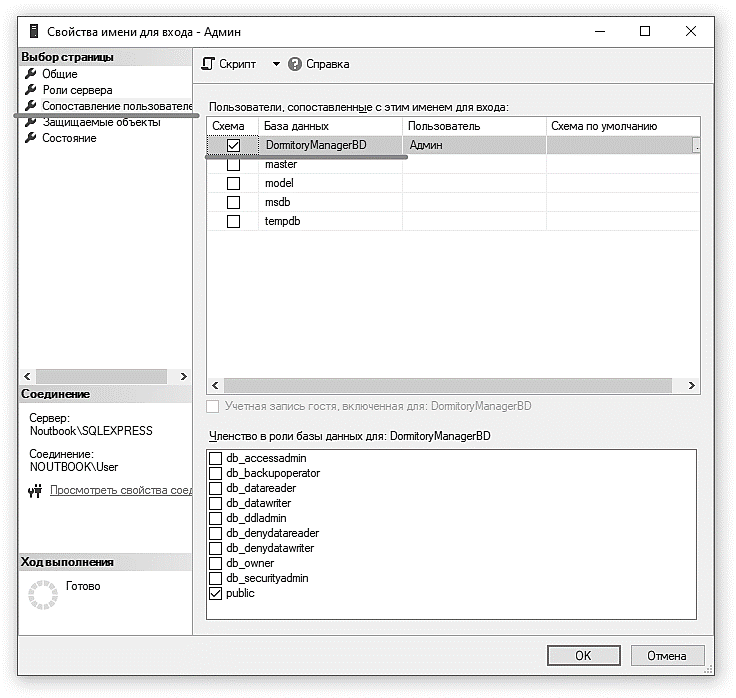
В появившемся окне вводим имя пользователя, выбираем способ входа в «Проверка подлинности SQL Server», дважды вводим пароль и снимаем галочку с пункта «Требовать использование политики паролей». После чего можем нажать на кнопку «ОК» и у нас появится пользователь, который будет отвечать и работать с базой данных DormitoryManagerBD.

**

*Рисунок 3.3 – Окно создания пользователя*

**3.3 Управление участниками уровня базы данных**

Пользователь Админ должен работать с базой данных DormitoryManagerBD. Поэтому надо его подключить к базе данных. Для этого в окне пользователя перехожим в пункт «Сопоставление пользователем» и там добавляем галочку к базе данных DormitoryManagerBD. Теперь Пользователь Админ может взаимодействовать с базой данных курсового проекта.



*Рисунок 3.4 – Связывание пользователя с базой данных*

**3.4 Управление разрешениями уровня базы данных**

Разрешения уровня базы данных определяют, что пользователи и роли базы данных могут делать с объектами базы данных, такими как таблицы, представления, хранимые процедуры и триггеры.

SQL Server предоставляет широкий спектр разрешений уровня базы данных, которые можно назначить пользователям и ролям базы данных. Эти разрешения можно разделить на несколько категорий:

* **Разрешения на объекты** определяют, что пользователи и роли базы данных могут делать с конкретными объектами базы данных, такими как таблицы, представления, хранимые процедуры и триггеры.
* **Разрешения на уровень базы данных** определяют, что пользователи и роли базы данных могут делать на уровне базы данных в целом.
* **Разрешения на уровень экземпляра** определяют, что пользователи и роли базы данных могут делать на уровне экземпляра SQL Server в целом.

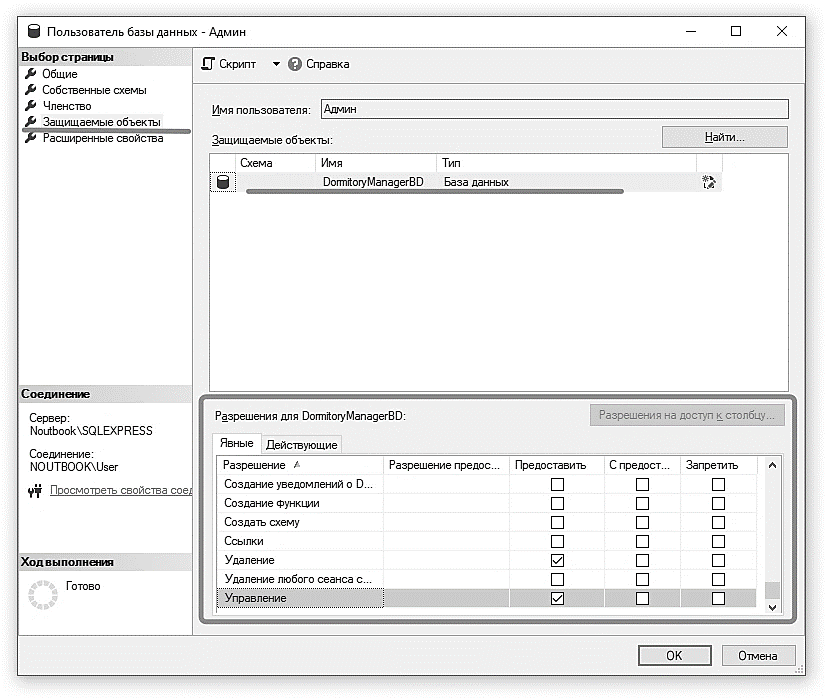
***Примеры разрешений уровня базы данных***

Вот несколько примеров разрешений уровня базы данных:

* SELECT - позволяет пользователю просматривать данные из таблицы.
* INSERT - позволяет пользователю вставлять данные в таблицу.
* UPDATE - позволяет пользователю обновлять данные в таблице.
* DELETE - позволяет пользователю удалять данные из таблицы.
* CREATE TABLE - позволяет пользователю создавать таблицы.
* ALTER TABLE - позволяет пользователю изменять таблицы.
* DROP TABLE - позволяет пользователю удалять таблицы.

Чтобы назначить разрешения пользователю, надо:

1. Щелкните правой кнопкой мыши пользователя, которому вы хотите назначить разрешения, и выберите «Защищаемые объекты».
2. Найдите объект, с которым пользователь после получения разрешения сможет взаимодействовать. В нашем случае это база данных DormitoryManagerBD.
3. На вкладке Разрешения выберите пользователя, которому вы хотите назначить разрешения.
4. В области Разрешения выберите разрешения, которые вы хотите предоставить пользователю. В данном случае дали разрешения управления, удаления и обновления базы данных.
5. Нажмите кнопку ОК.



*Рисунок 3.5 – Окно предоставления разрешений пользователю*

**3.5 Шифрование данных баз данных**

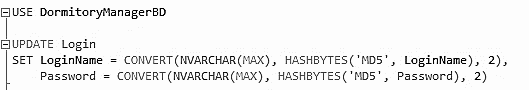
В современных информационных системах все чаще возникает необходимость в защите данных от несанкционированного доступа. Одним из способов защиты данных является шифрование. Шифрование - это процесс преобразования данных в нечитаемый вид с помощью ключа. Для расшифровки данных необходим тот же ключ, который использовался для шифрования.

В базах данных шифрование может использоваться для защиты различных данных, таких как:

* Данные пользователей, такие как логины, пароли и другие персональные данные.
* Данные приложений, такие как код, конфигурационные файлы и другие конфиденциальные данные.
* Данные инфраструктуры, такие как файлы журналов, файлы резервных копий и другие важные данные.

Чтобы защитить нашу базу данных курсового проекта, проведём один из популярных способов шифрования метод – хеширование.

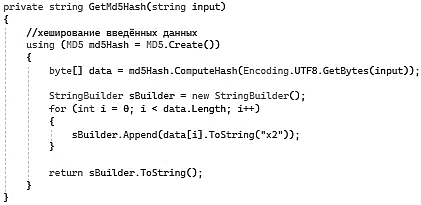
Сначала сделаем хеширование столбцов LoginName и Password таблицы Login. Так увеличим безопасность авторизированных в базе данных пользователей.



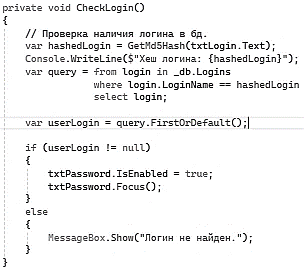
*Рисунок 3.6 – Запрос хеширования столбцов таблицы Login*

А чтобы программа, работающая и связанная с базой данных курсового проекта, понимала, что это те же данные, тоже будем использовать хеширование после ввода данных пользователей.

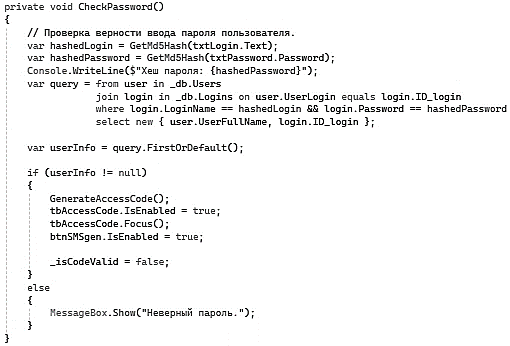
Для этого добавим метод, который делает хеширование введённых данных. Вставляем функцию в методы ввода, которые принимает и обрабатывает вводимые пользователем данные. Это будут методы проверки ввода логина и пароля пользователя в авторизированную информационную систему.



*Рисунок 3.7 – Код метода хеширования введённых данных*



*Рисунок 3.8 – Код метода проверки ввода логина пользователя*



*Рисунок 3.9 – Код метода проверки ввода пароля пользователя*

**ЗАКЛЮЧЕНИЕ**

Таким образом, курсовой проект по разработке информационной системы «Учёт расчетов за проживание в общежитии» является важным инструментом для упрощения и автоматизации обслуживания общежитий. Автоматизация процесса учёта и контроля задолженностей позволит ускорить работу бухгалтеров и администраторов общежитий.

В процессе выполнения проекта мы разработали базу данных и пользовательский интерфейс, позволяющий быстро удобно вводить просматривать данные о жителях общежития и их расчетах. Мы также реализовали функциональность для авторизации пользователей и управления доступом.

Кроме того, мы заботились о том, чтобы интерфейс был интуитивно понятным и простым в использовании, а база данных была надежной и безопасной для хранения информации.

Работа над проектом позволила улучшить навыки разработки информационных систем и расширить понимание принципов хранения и обработки данных. Кроме того, мы понимаем важность использования информационных технологий в повседневной жизни и в работе с пользовательскими данными.

Посмотреть на информационную систему можно по это ссылке:

https://github.com/KamillaYesa/exams\_2023\_12

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

Книги

1. Акулов, А. Язык программирования SQL. Основы синтаксиса и разработки баз данных. Издательство: Питер, 2022 год. - 368 с.

2. Ли, Джонсон. SQL: Полное руководство. Издательство: Вильямс, 2021 год. - 672 с.

3. Петкович, Д. Microsoft SQL Server 2022: Руководство для начинающих, 8-е издание / Д. Петкович. - М.: Издательский дом "Питер", 2021. - 512 с.

4. Петкович, Д. Microsoft SQL Server 2021: Руководство для начинающих / Д. Петкович. - М.: Издательский дом "Питер", 2020. - 464 с.

5. Петкович, Д. Microsoft SQL Server 2022: Руководство для начинающих, 8-е издание. Издательский дом "Питер", 2021 год. - 944 с.

6. Уорд, Б. SQL Server 2021 Revealed: Включая Big Data Clusters и Машинное Обучение / Б. Уорд. - М.: Издательский дом "Питер", 2020. - 320 с.

**Интернет-ресурсы**

1. “SQL Онлайн: 10 Лучших Платформ Для Тренировки Навыков” – Режим доступа: https://www.sqlonline.com/
2. “5 сайтов для оттачивания навыков написания SQL-запросов” – Режим доступа: https://www.sqlsites.com/
3. “6 бесплатных ресурсов для практики в SQL” – Режим доступа: https://www.sqlresources.com/
4. “sql-tutorial - это интерактивный онлайн-учебник по изучению SQL” – Режим доступа: https://www.sql-tutorial.com/
5. “SQL Tutorial - Full Database Course for Beginners” – режим доступа: https://www.youtube.com/watch?v=HXV3zeQKqGY
6. “sql-ex полностью ориентирован на задачи по базам данных” – Режим доступа: https://www.sql-ex.com/
7. “Администрирование SQL Server серия уроков” – Режим доступа: https://www.youtube.com/watch?v=X508KcPidyk&list=PLhhjwMYxzolgZyb5R3ZnTbVPG4\_gHHQJh&index=1
8. "Введение в .NET: что такое .NET и обзор основ" – Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/core/introduction
9. "Начало работы с .NET: что такое среда .NET и как она работает?" – Режим доступа: https://habr.com/ru/sandbox/13849/
10. Info-Comp (Особенности SQL Server Management Studio). – Режим доступа: <https://info-comp.ru/ssms-feature-overview>
11. Metanit (Entity Framework). – Режим доступа: <https://metanit.com/sharp/entityframework/1.1.php>
12. Metanit (Основы работы с Microsoft SQL Server). – Режим доступа: <https://metanit.com/sql/sqlserver/1.1.php>
13. Metanit (Основы работы с SQL Server Management Studio). – Режим доступа: <https://metanit.com/sql/sqlserver/1.3.php>
14. Microsoft Docs (Entity Framework). – Режим доступа: https://learn.microsoft.com/en-us/ef/
15. Microsoft Docs (SQL Server Management Studio (SSMS)). – Режим доступа: <https://learn.microsoft.com/ru-ru/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15>
16. Timeweb (Основы работы с MS SQL Server). – Режим доступа: <https://timeweb.com/ru/community/articles/osnovy-raboty-s-mssqlserver>
17. Wikipedia (Entity Framework). – Режим доступа: https://en.wikipedia.org/wiki/Entity\_Framework
18. Wikipedia (Microsoft SQL Server). – Режим доступа: <https://ru.wikipedia.org/wiki/Microsoft_SQL_Server>
19. Wikipedia (SQL Server Management Studio). – Режим доступа: <https://ru.wikipedia.org/wiki/SQL_Server_Management_Studio>
20. Wikipedia (SQL). – Режим доступа: <https://ru.wikipedia.org/wiki/SQL>