**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**

**«Учёт сдельной оплаты труда»**

Выполнил студент гр. ИСП-20 /\_\_\_\_\_\_\_\_\_\_/Шамсулин Артём Фанильевич/

(подпись) (Ф.И.О.)

**ОЦЕНКА:\_\_\_\_\_\_\_\_\_\_\_\_\_**

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

**ПРОВЕРИЛ**

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

(подпись) (Ф.И.О.)

Воркута

2023

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc153184217)

[ВВЕДЕНИЕ 3](#_Toc153184218)

[ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ 4](#_Toc153184219)

[1.1 СУБД SQL Server 4](#_Toc153184220)

[1.2 Microsoft SQL Server Management Studio 5](#_Toc153184221)

[1.3 Entity Framework 6](#_Toc153184222)

[ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 9](#_Toc153184223)

[2.1 Разработка диаграммы ERD 9](#_Toc153184224)

[2.2 Разработка базы данных 11](#_Toc153184225)

[2.3 Словарь данных 12](#_Toc153184226)

[ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ 15](#_Toc153184227)

[3.1 Введение в безопасность SQL Server 15](#_Toc153184228)

[3.2 Управление безопасностью уровня сервера 15](#_Toc153184229)

[3.3 Управление участниками уровня базы данных 17](#_Toc153184230)

[3.4 Управление разрешениями уровня базы данных 19](#_Toc153184231)

[3.5 Шифрование данных баз данных 21](#_Toc153184232)

[ЗАКЛЮЧЕНИЕ 23](#_Toc153184233)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 24](#_Toc153184234)

# ВВЕДЕНИЕ

Учёт сдельной оплаты труда на любом предприятии по праву занимает одно из центральных мест в системе учёта. Заработная плата – основной источник дохода рабочих и служащих, с её помощью осуществляется контроль за мерой труда и потребления, она используется как важнейший предмет в области экономики. Заработная плата – это вознаграждение за труд в зависимости от квалификации работника, сложности, количества, качества и условий выполнения работы.

Актуальность данной работы заключается в необходимости проектирования, разработки и внедрения информационной системы учёта и контроля рабочего времени сотрудников торговой организации, способной оказывать качественную поддержку для успешного ведения управленческой деятельности в частности и коммерческой деятельности в целом. И в отслеживании и учёте всех сотрудников на предприятии.

**Объект** - база данных «Учёт сдельной оплаты труда».

**Предмет** – автоматизация бизнес-процесса учёта оплаты труда.

**Цель** – создание базы данных «Расчёт сдельной оплаты труда».

Для достижения данной цели необходимо решить следующие задачи:

* выбрать инструментарий к информационной системе «Учёт сдельной оплаты труда»;
* спроектировать базу данных к информационной системе «Учёт сдельной оплаты труда»;
* сделать разграничение ролей для базы данных;
* сделать шифрование данных для базы данных;
* разработать информационную систему «Учёт сдельной оплаты труда».

# ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ

## 1.1 СУБД SQL Server

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL посредством специального API. СУБД должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

В зависимости от задачи, которую выполняет команда T-SQL, он может принадлежать к одному из следующих типов:

DDL (Data Definition Language / Язык определения данных). К этому типу относятся различные команды, которые создают базу данных, таблицы, индексы, хранимые процедуры и т.д. В общем определяют данные.

В частности, к этому типу мы можем отнести следующие команды:

* CREATE: создает объекты базы данных (саму базу данных, таблицы, индексы и т.д.)
* ALTER: изменяет объекты базы данных
* DROP: удаляет объекты базы данных
* TRUNCATE: удаляет все данные из таблиц

DCL (Data Control Language / Язык управления доступа к данным). К этому типу относят команды, которые управляют правами по доступу к данным. В частности, это следующие команды:

* GRANT: предоставляет права для доступа к данным
* REVOKE: отзывает права на доступ к данным

## 1.2 Microsoft SQL Server Management Studio

SQL Server Management Studio (SSMS) — это интегрированная среда для управления любой инфраструктурой SQL. Используйте SSMS для доступа, настройки, администрирования и разработки всех компонентов SQL Server, Базы данных SQL Azure, Управляемого экземпляра SQL Azure, SQL Server на виртуальной машине Azure и Azure Synapse Analytics. 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.

Среда SQL Server Management Studio предназначена для доступа к службам Службы Analysis Services, Integration Services и Reporting Services, а также для их настройки, администрирования и управления ими. Хотя все три технологии бизнес-аналитики полагаются на среду SQL Server Management Studio, административные задачи, связанные с каждой из этих технологий, несколько отличаются.

Среда SQL Server Management Studio позволяет управлять объектами Службы Analysis Services, например выполнять их резервное копирование и обработку.

Среда Management Studio позволяет создавать проекты скриптов служб Службы Analysis Services, в которых выполняются разработка и сохранение скриптов с использованием многомерных выражений (MDX), расширений интеллектуального анализа данных (DMX) и XML для аналитики (XMLA). Проекты скриптов служб Службы Analysis Services используются для выполнения задач управления или повторного создания баз данных, кубов и других объектов в экземплярах служб Службы Analysis Services. Например, можно разработать скрипт XMLA в проекте скрипта служб Службы Analysis Services, который создает объекты непосредственно в существующем экземпляре служб Службы Analysis Services. Проекты скриптов служб Службы Analysis Services могут быть сохранены в составе решения и интегрироваться с контролем исходного кода.

**Управление решениями Integration Services с помощью SQL Server Management Studio**

Среда SQL Server Management Studio позволяет использовать службу Integration Services для управления пакетами и наблюдения за выполняющимися пакетами. В среде Среда Management Studio можно организовать пакеты в папки, выполнять, импортировать и экспортировать пакеты, переносить пакеты служб DTS и обновлять пакеты служб Integration Services.

**Управление проектами Reporting Services с помощью SQL Server Management Studio**

Среда SQL Server Management Studio позволяет включать компоненты служб Reporting Services, администрировать серверы и базы данных, управлять ролями и заданиями.

Она реализует функции управления общими расписаниями (в папке «Общие расписания») и базами данных сервера отчетов (ReportServer, ReportServerTempdb). Можно также создать роль RSExecRole в системной базе данных Master, когда база данных сервера отчетов перемещается в новое или другое ядро СУБД SQL Server.

## 1.3 Entity Framework

Entity Framework — это решение для работы с базами данных, которое используется в программировании на языках семейства .NET. Оно позволяет взаимодействовать с СУБД с помощью сущностей (entity), а не таблиц. Также код с использованием EF пишется гораздо быстрее.

Например, работая с базами данных напрямую, разработчик должен беспокоиться о подключении, подготовке SQL и параметров, отправке запросов и транзакций. На Entity Framework всё это делается автоматически — программист же работает непосредственно с сущностями и только говорит EF, что нужно сохранить изменения.

В этой статье будут разобраны основы применения Entity Framework, для понимания которых нужно владеть ADO.NET — пользоваться базами данных, писать SQL-запросы и работать с подключениями.

Entity Framework позволяет значительно сократить код работы с базами данных. При этом он предоставляет большие возможности. Например, можно использовать:

* foreign keys;
* связи one-to-one, one-to-many и many-to-many;
* параметризацию запросов;
* хранимые процедуры.

Однако стоит учитывать, что EF выступает прослойкой между приложением и базой данных, поэтому может ухудшаться производительность. Для небольших проектов это допустимо, но если программа должна работать под большой нагрузкой, то лучше использовать чистый ADO.NET.

**Предоставление жизненного цикла моделям**

Многолетним и общим подходом к разработке является подход, при котором построение приложения или службы представляет собой его разделение на три части: модель домена, логическую модель и физическую модель. Модель домена определяет сущности и связи в моделируемой системе. Логическая модель для реляционной базы данных обеспечивает нормализацию сущностей и связей в целях создания таблиц с ограничениями внешнего ключа. В физической модели учитываются возможности конкретной системы обработки данных путем определения, зависящих от ядра базы данных подробных сведений о хранении данных, которые касаются секционирования и индексирование.

Физическая модель совершенствуется администраторами базы данных в целях повышения производительности, но программисты, которые разрабатывают код приложения, в основном вынуждены ограничиваться работой с логической моделью, подготавливая SQL-запросы и вызывая хранимые процедуры. Модели домена в основном используются как инструмент для представления и обмена мнениями о требованиях к приложению, поэтому чаще всего служат в качестве практически не изменяющихся схем, которые рассматриваются и обсуждаются на ранних стадиях проекта, после чего выходят из сферы внимания. Во многих коллективах разработчиков принято пропускать этап создания концептуальной модели и начинать с определения таблиц, столбцов и ключей в реляционной базе данных.

Entity Framework дает жизнь модели, позволяя разработчикам запрашивать сущности и связи в модели предметной области (называемой концептуальной моделью в Entity Framework), в то же время полагаться на Entity Framework преобразования этих операций в команды, относящиеся к источнику данных. Это позволяет отказаться от применения в приложениях жестко заданных зависимостей от конкретного источника данных.

При работе в режиме Code First концептуальная модель сопоставлена с режимом хранения в коде. Entity Framework может вычислять концептуальную модель на основе типов объектов и дополнительных конфигураций, которые вы определяете. Метаданные сопоставления формируются во время выполнения на основе сочетания определений типов домена и дополнительной информации о конфигурации, которая указана в коде. Entity Framework создает базу данных по мере необходимости на основе метаданных. Дополнительные сведения см. в разделе Создание модели.

# ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 2.1 Разработка диаграммы ERD

Данная диаграмма — (ER-модель данных)— 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 и разработки программного обеспечения и информационных систем для бизнеса. ER-диаграммы (или ER-модели) полагаются на стандартный набор символов, включая прямоугольники, ромбы, овалы и соединительные линии, для отображения сущностей, их атрибутов и связей. Эти диаграммы устроены по тому же принципу, что и грамматические структуры: сущности выполняют роль существительных, а связи — глаголов.

Базовыми понятиями ER-модели данных (ER — Entity— Relationship) являются сущность, атрибут и связь. Сущность — это класс однотипных реальных или абстрактных объектов (людей, событий, состояний, предметов и т.п.), информация о которых имеет существенное значение для рассматриваемой предметной области.

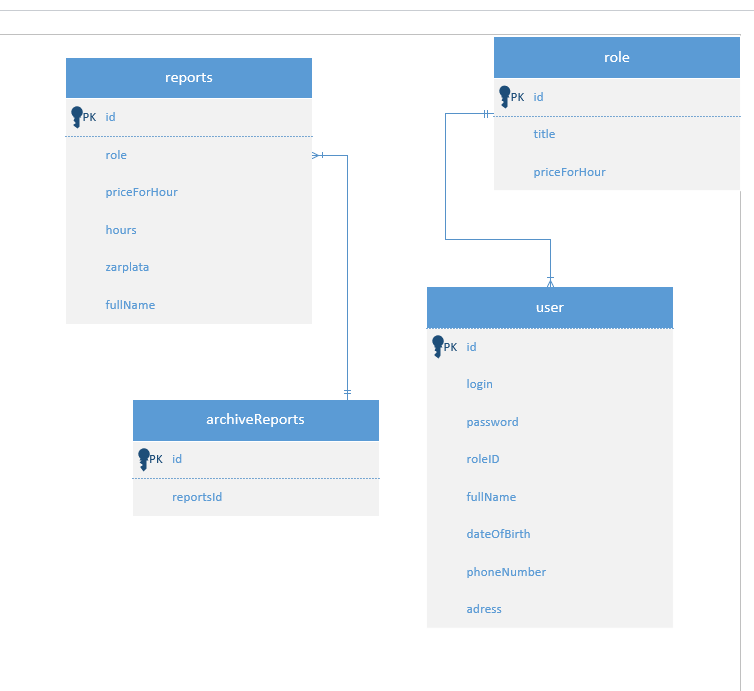
Атрибут — любая характеристика сущности, значимая для рассматриваемой предметной области и предназначенная для квалификации, идентификации, классификации, количественной характеристики или выражения состояния сущности. Атрибут, таким образом, представляет собой некоторый тип характеристик или свойств, ассоциированных с множеством реальных или абстрактных объектов.

Атрибуты делятся на ключевые, т. е. входящие в состав уникального идентификатора ключа, и описательные — прочие.

Первичный ключ — это атрибут или совокупность атрибутов и связей, предназначенная для уникальной идентификации каждого экземпляра сущности.

Описательные атрибуты могут быть обязательными или необязательными. Обязательные атрибуты для каждой сущности всегда имеют конкретное значение, необязательные могут быть не определены.

Связь — это отношение одной сущности к другой или к самой себе. Если любой экземпляр одной сущности связан хотя бы с одним экземпляром другой сущности, то связь является обязательной. Необязательная связь представляет собой условное отношение между сущностями. Каждая сущность может обладать любым количеством связей с другими сущностями модели. Различают три типа отношений «один-к-одному»; «один-ко-многим»; «многие-ко-многим».



*Рисунок 2.1. ER Диаграмма*

Основными сущностями для решения указанной задачи являются: user, reports. Сразу возникает очевидная связь между сущностями user и role для авторизации и reports и archiveReports.

Проанализируем атрибуты сущностей. reports может иметь ФИО, должность, ставку в час, часы, зарплату; user может иметь логин, пароль, айди должности, ФИО, дата рождения, номер телефона, адрес. У каждой сущности есть ID.

## 2.2 Разработка базы данных

Для подключения базы данных в обозревателе объектов нужно подключиться к экземпляру компонента SQL Server Database Engine. Правой кнопкой мыши нажать на узел базы данных и выбрать команду создать базу данных. В поле новая базы данных ввести имя базы данных.

Для создания базы данных, приняв все значения по умолчанию, необходимо нажать на кнопку ОК, иначе необходимо сделать следующее:

Поменять имя владельца, нажав ( … ) и выбрать другого владельца.

Изменить параметры сортировки базы данных, выбрав страницу параметров и указав из списка желаемые параметры сортировки.

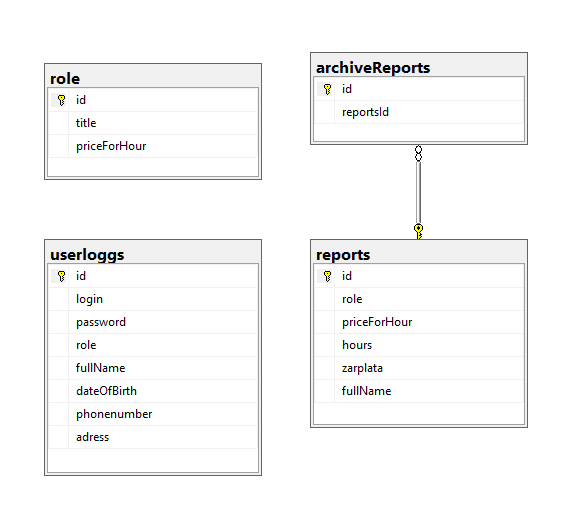
Изменить модель восстановления, выбрав страницу параметров и модель восстановления из списка.

Изменить параметры базы данных, выбрав страницу параметров и изменив параметры базы данных.

Добавить новую файловую группу, перейдя на страницу файловых групп. Нажать кнопку добавить и ввести значения для файловой группы.

Добавить расширенное свойство в базу данных, выбрав страницу расширенных свойств. В столбце имени ввести имя расширенного свойства. В столбце значений ввести текст расширенного свойства.

Чтобы создать базу данных, нажмите кнопку ОК.



*Рисунок 2.2. База данных приложения*

В базе данных имеется 4 таблицы, есть userloggs и role – они отвечают за авторизацию в приложении, reports – для сохранения информации по отчётам зарплат. И archiveReports для сохранение архивированных отчетов в отдельной таблице.

## 2.3 Словарь данных

Словарь данных - это структурированное описание данных, которое содержит информацию о структуре, организации и свойствах данных в базе данных. Он обычно используется для документирования метаданных, то есть информации о данных в базе данных. В словаре данных обычно содержатся следующие элементы:

1. Описание таблиц и полей: словарь данных содержит информацию о различных таблицах в базе данных, их названия, структуру, связи с другими таблицами, и описание каждого поля в таблице.

2. Описание ограничений и связей: словарь данных может также содержать информацию о различных ограничениях целостности данных, связях между таблицами и других ограничениях, которые определяют правила и условия для хранения данных.

3. Описания процедур и функций: метаданные о реализованных программных объектах, таких как хранимые процедуры, функции, триггеры, представления и т.д., могут быть также включены в словарь данных.

Словарь данных является важным инструментом для анализа и понимания структуры базы данных. Он помогает администраторам и разработчикам понять, как организованы данные, и обеспечить их правильное использование и обслуживание.

Таблица 1 Роль

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data Dictionary | | | | |
| **role** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| P/KEY | **id** | int | N | Уникальный идентификатор |
|  | **title** | nvarchar(50) | N | Название роли |
|  | **priceForHour** | nvarchar(50) | N | Ставка час |

Таблица 2 Пользователь

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **userloggs** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| P/KEY | **id** | int | N | Уникальный идентификатор |
|  | **login** | nvarchar(50) | N | Название роли |
|  | **password** | nvarchar(50) | N | Пароль |
|  | **role** | nvarchar(50) | N | Роль |
|  | **fullName** | nvarchar(50) | N | Полное имя |
|  | **dateOfBirth** | datetime | N | Дата рождения |
|  | **phonenumber** | nvarchar(50) | N | Номер телефона |
|  | **adress** | nvarchar(50) | N | Адрес проживания |

Таблица 3 Архивные отчёты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **archiveReports** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| P/KEY | **id** | int | N | Уникальный идентификатор |
|  | **reportsId** | int | N | Идентификатор роли |

Таблица 4 Отчёты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **reports** | | | | |
| **KEY** | **FIELD NAME** | **DATA TYPE / FIELD SIZE** | **REQUIRED?** | **NOTES** |
| P/KEY | **id** | int | N | Уникальный идентификатор |
|  | **role** | nvarchar(50) | N | Роль |
|  | **priceForHour** | nvarchar(50) | N | Ставка в час |
|  | **hours** | nvarchar(50) | N | Отработанные часы |
|  | **zarplata** | nvarchar(50) | N | Итоговая зарплата |
|  | **fullName** | nvarchar(50) | N | Номер телефона |

# ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ

## 3.1 Введение в безопасность SQL Server

Безопасность данных является одним из самых важных аспектов любой базы данных, включая SQL Server. Поддержание безопасности в SQL Server включает в себя ряд мероприятий, предназначенных для защиты данных от несанкционированного доступа, вмешательства и утечки информации.

При правильном создании резервных копий баз данных можно будет восстановить данные после многих видов сбоев, включая следующие:

• Сбой носителя.

• ошибки пользователей (например, удаление таблицы по ошибке);

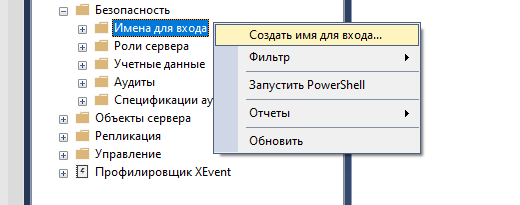
• сбои оборудования (например, поврежденный дисковый накопитель или безвозвратная потеря данных на сервере);

Для создания резервной копии базы данных с помощью SQL Server Management Studio (SSMS) следуйте этим шагам:

1. Нажмите правой кнопкой мыши на базе данных, выберите "Tasks" (Задания) и затем "Backup" (Резервное копирование).
2. В появившемся окне выберите тип резервного копирования (полное, дифференциальное, журнал транзакций).
3. Укажите путь для сохранения резервной копии базы данных.

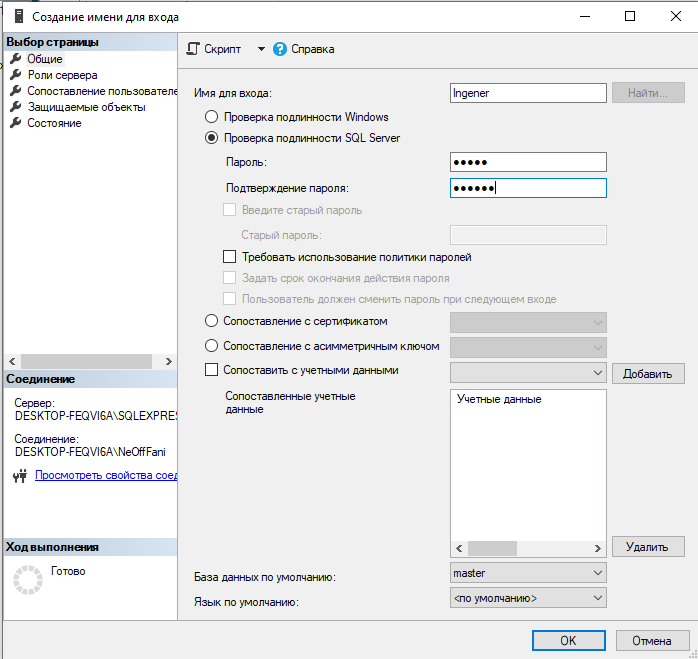
## Управление безопасностью уровня сервера

Существует два уровня доступа к экземпляру SQL Server: учетные записи пользователя сервера (или экземпляра) и пользователи базы данных. С помощью учетных записей серверы позволяют внешнему пользователю (далее термин «пользователь» применяется для любого приложения, службы, API и т. д., пытающихся подключиться к SQL Server) выполнить начальное соединение с экземпляром SQL Server. В случае безопасности на основе SQL для этого требуются имя пользователя и пароль. В случае доверенной проверки подлинности это учетная запись домена.

Способа создать эти учетные записи пользователя через графический интерфейс. Чтобы запустить диалоговое окно для создания учетных записей пользователей, подключитесь к экземпляру SQL Server в среде SQL Server Management Studio (SSMS) в обозревателе объектов, а затем разверните узел Security\Logins («Безопасность\Имена пользователя»). Щелкните правой кнопкой мыши на пункте Logins и выберите в контекстном меню пункт New Login («Создать учетную запись»).

***Рисунок 3.1. Создание учетной записи пользователя SQL Server***

Вы увидите диалоговое окно для настройки параметров учетной записи, показанное на рисунке 3.1. Изменить имя пользователя можно в том же окне.

***Рисунок 3.3 Настройка учетной записи пользователя SQL Server***

Это вкладка General («Общие») для создания (и изменения)

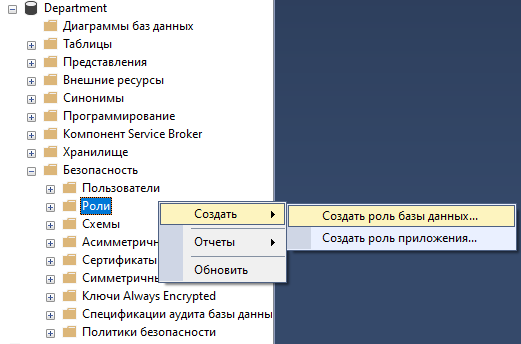
Далее следуем инструкциям на экране.

## 3.3 Управление участниками уровня базы данных

Чтобы легко управлять разрешениями в базах данных, SQL Server предоставляет несколько ролей, которые являются субъектами безопасности, которые группируют другие субъекты. Они похожи на группы в операционной системе Microsoft Windows. Разрешения ролей уровня базы данных распространяются на всю базу данных.

Существует два типа ролей уровня базы данных: предопределенные роли базы данных, являющиеся стандартными для базы данных, и пользовательские роли базы данных, которые можно создавать.

Предопределенные роли базы данных задаются на уровне базы данных и предусмотрены в каждой базе данных. Члены ролей базы данных db\_owner могут управлять членством в предопределенных ролях базы данных. Кроме того, в базе данных msdb имеются специальные роли базы данных.

В SQL Server Management Studio разверните базу данных табличной модели, для которой требуется создать новую роль, щелкните правой кнопкой мыши роли и выберите команду Создать роль.

*Рисунок 3.4. Настройка роли.*

В окне общих параметров введите имя роли в поле Имя.

Используйте имена, которые четко идентифицируют тип элемента, например, финансовые менеджеры или специалисты по персоналу, и убедитесь, что имя не содержит запятую.

В поле Разрешения базы данных для этой роли выберите один из следующих параметров разрешений.

Полный доступ (администратор). Члены могут вносить изменения в схему модели, а также просматривать все данные.

Process Database. Члены могут выполнять операции «Обработать» и «Обработать все». Не могут изменять схему модели и просматривать данные.

Чтение. Члены могут просматривать данные (с учетом фильтров строк), но не могут вносить изменения в схему модели.

В диалоговом окне Создание роли в окне «Выбор страницы» щелкните Членство.

В окне параметров членства нажмите кнопку добавить, а затем в диалоговом окне выбор пользователей или групп добавьте пользователей или группы, которые вы хотите добавить в качестве участников.

## 3.4 Управление разрешениями уровня базы данных

Соглашения об именовании разрешений

Ниже описаны общие соглашения, которые соблюдаются при задании имен разрешениям.

Производительности

Предоставляет возможности, схожие с владением. Имеющий это разрешение получает все установленные разрешения на защищаемую сущность. Участник, получивший разрешение CONTROL, может также предоставлять разрешения на защищаемую сущность другим участникам. Так как модель безопасности SQL Server является иерархической, CONTROL на определенном область неявно включает CONTROL для всех защищаемых объектов в этом область.

ИЗМЕНИТЬ

Предоставляет возможность изменения свойств определенной защищаемой сущности, кроме ее владельца. При предоставлении разрешения ALTER на ту или иную область также предоставляется возможность изменения, создания или удаления любой защищаемой сущности, содержащейся в пределах данной области.

ALTER ANY <Server Securable, где защищаемый> сервер может быть любым защищаемым сервером.

Предоставляет возможность создавать, изменять и удалять отдельные экземпляры Защищаемой сущности сервера.

ALTER ANY <Database Securable, где защищаемая> база данных может быть любой защищаемой на уровне базы данных.

Предоставляет возможность СОЗДАВАТЬ, ИЗМЕНЯТЬ и УДАЛЯТЬ отдельные экземпляры Защищаемой сущности базы данных.

TAKE OWNERSHIP

Позволяет получать во владение защищаемую сущность, на которую предоставлено разрешение.

IMPERSONATE <Имя\_для\_входа>

Позволяет олицетворять имя входа.

IMPERSONATE <Пользователь>

Позволяет олицетворять пользователя.

CREATE <Защищаемый объект сервера>

Предоставляет возможность создавать Защищаемую сущность сервера.

CREATE <Защищаемый объект базы данных>

Предоставляет возможность создавать Защищаемую сущность базы данных.

CREATE <Защищаемый объект, содержащийся в схеме>

Предоставляет возможность создавать защищаемую сущность, содержащуюся в схеме. Однако для создания защищаемой сущности в той или иной схеме на эту схему требуется разрешение ALTER.

VIEW DEFINITION

Разрешает доступ к метаданным.

ССЫЛКИ

Разрешение REFERENCES для таблицы необходимо для создания ограничения FOREIGN KEY, которое ссылается на эту таблицу.

Разрешение REFERENCES для объекта необходимо для создания FUNCTION или VIEW с предложением 2 WITH SCHEMABINDING , которое ссылается на этот объект.

## 3.5 Шифрование данных баз данных

Шифрование данных в базах данных SQL Server - это важный аспект безопасности, который позволяет защитить конфиденциальную информацию от несанкционированного доступа. В SQL Server Management Studio (SSMS) можно использовать различные методы шифрования для защиты данных.

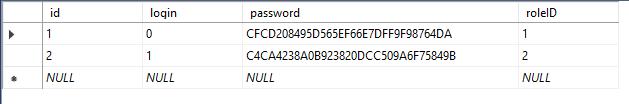
В нашей базе данных используется тип шифрования MD5. MD5 - это алгоритм хэширования, который используется для создания уникального хэш-значения из входных данных фиксированной длины. В SQL Server MD5 обычно используется как метод для создания хэш-значений, которые могут быть использованы для проверки целостности данных, а также в качестве метода необратимого шифрования паролей и других конфиденциальных данных.

Что бы зашифровать данные в SSMS понадобится небольшой запрос

UPDATE userlog

SET password = CONVERT(varchar(64),

HASHBYTES('MD5',password),2)



*Рисунок 3.10. Зашифрованные данные*

Благодаря этому запросу мы зашифровали строку с паролями пользователей

Но что бы в программе не надо было вводить зашифрованный пароль нужно реализовать расшифровку данных из базы данных так что бы вводился пароль установленный изначально.

Для этого нужно создать класс в котором будет данный код:

internal class Class1

{

public static string hashPassword(string password)

{

MD5 md5 = MD5.Create();

byte[] b = Encoding.UTF8.GetBytes(password);

byte[] hash = md5.ComputeHash(b);

StringBuilder sb = new StringBuilder();

foreach (var a in hash)

sb.Append(a.ToString("X2"));

return Convert.ToString(sb);

}

}

И теперь при входе в приложение нужно будет ввести обычный пароль в то время как в базе данных он будет зашифрован.

# ЗАКЛЮЧЕНИЕ

Таким образом, в ходе выполнения курсового проекта была успешно разработана база данных для учета сдельной оплаты труда. Было проведено разграничение ролей пользователей, обеспечено шифрование данных для защиты информации и настроено резервное копирование для обеспечения сохранности данных. Результаты проекта позволяют эффективно и безопасно управлять данными по сдельной оплате труда, что может быть полезно для многих предприятий и организаций.

Ссылка на GitHub: <https://github.com/AsulinShatem/uchet>

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Albahari, J., Albahari, B., & Albahari, O. (2018). C# 5.0 вкратце: Определенная справочная информация. O'Reilly Media.

2. Рихтер, Дж. Р. (2020). CLR через C# (4-е издание). Издательство Microsoft.

3. Сарка, Д., Пилат, С., & Кёниг, Р. (2019). Учебный комплект (экзамен 70-462): Администрирование баз данных Microsoft SQL Server 2012. Издательство Microsoft.

4. Стирк, Р., & Лич, С. (2018). Кулинарная книга SQL Server 2014 с PowerShell v5. Издательство Packt Publishing.Интернет-ресурсы

5. CodeProject [Электронный ресурс] – Режим доступа: <https://www.codeproject.com/>

6. Stack Overflow [Электронный ресурс] – Режим доступа: <https://stackoverflow.com/>

7. SQLShack [Электронный ресурс] – Режим доступа: <https://www.sqlshack.com/>

8. SQLServerCentral [Электронный ресурс] – Режим доступа: <https://www.sqlservercentral.com/>

9. C# Corner [Электронный ресурс] – Режим доступа: <https://www.c-sharpcorner.com/>

10. Pluralsight [Электронный ресурс] – Режим доступа: <https://www.pluralsight.com/>