НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

(повна назва інституту/факультету)

КАФЕДРА інформатики та програмної інженерії

(повна назва кафедри)

**КУРСОВА РОБОТА**

з дисципліни «Бази даних»

(назва дисципліни)

на тему: база даних для підтримки діловодства організації

Студента 2 курсу групи ІП-01

спеціальності 121 «Інженерія програмного забезпечення»

\_\_\_\_\_\_\_\_\_\_\_\_Танасієнка О. В.\_\_\_\_\_

(прізвище та ініціали)

Керівник \_\_\_\_Лебідь С. О. \_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_ Оцінка ECTS \_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Лебідь С. О.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

Київ – 2021 рік

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет Інформатики та обчислювальної техніки

(повна назва)

Кафедра Інформатики та програмної інженерії

(повна назва)

Дисципліна Бази даних

Курс \_\_2\_\_\_ Група \_\_ІП-01\_\_\_\_ Семестр \_\_1\_\_\_

**З А В Д А Н Н Я**

**НА КУРСОВУРОБОТУ СТУДЕНТУ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Танасієнку Олександру Вячеславовичу\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище, ім’я, по батькові)

1. Тема роботи база даних для підтримки діловодства організації\_\_\_

керівник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_Лебідь С. О.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 22» грудня 2021р.

3. Вихідні дані до роботи створена база даних відповідно до умови, SQL скрипти

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Вступ, опис предметного середовища, постановка задачі, побудова ER-моделі бази даних, побудова логічної(реляційної) схеми бази даних, створення бази даних з допомогою обраної СУБД, імпортування даних у базу даних, створення користувачів та реалізація функціоналу для них, створення SQL запитів, висновок, перелік посилань.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

ER-діаграма, реляційна схема бази даних, приклади виконання SQL скриптів

6. Дата видачі завдання «31» жовтня 2021р.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва етапів виконання курсового проекту | Строк виконання етапів проекту | Примітка |
| 1. | Вступ | 31.10.2021-01.11.2021 |  |
| 2. | Опис предметного середовища | 01.11.2021-03.11.2021 |  |
| 3. | Постановка задачі | 01.11.2021-03.11.2021 |  |
| 4. | Побудова ER-моделі | 03.11.2021-13.11.2021 |  |
| 5. | Побудова реляційної схеми | 13.11.2021-17.11.2021 |  |
| 6. | Створення бази даних з допомогою обраної СУБД | 17.11.2021-19.11.2021 |  |
| 7. | Імпортування даних | 17.11.2021-19.11.2021 |  |
| 8. | Створення користувачів та реалізація їх функціоналу | 19.11.2021-24.11.2021 |  |
| 9. | Створення SQL запитів | 19.11.2021-13.12.2021 |  |
| 10. | Висновок | 13.12.2021-14.12.2021 |  |
| 11. | Перелік посилань | 31.10.2021-14.12.2021 |  |

**Студент \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Танасієнко О. В.**

(підпис ) (прізвище та ініціали)

**Керівник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_Лебідь С. О.\_\_\_**

(підпис ) (прізвище та ініціали)

ЗМІСТ

[Вступ 5](#_Toc91112345)

[1 Опис предметного середовища 6](#_Toc91112346)

[2 Постановка задачі 7](#_Toc91112347)

[3 Побудова ER-моделі 8](#_Toc91112348)

[3.1 Аналіз предметного середовища 8](#_Toc91112349)

[3.2 Бізнес-правила 9](#_Toc91112350)

[3.3 Вибір сутностей 9](#_Toc91112351)

[3.4 Набори атрибутів сутностей 10](#_Toc91112352)

[3.5 Зв’язки між об’єктами 11](#_Toc91112353)

[3.6 ER-модель 13](#_Toc91112354)

[4 Побудова реляційної схеми 14](#_Toc91112355)

[4.1 Побудова необхідних відношень та визначення первинних і зовнішніх ключів 14](#_Toc91112356)

[4.2 Визначення обмежень цілісності для спроектованих відношень 14](#_Toc91112357)

[5 Створення бази даних у форматі системи управління базою даних MS SQL Server 16](#_Toc91112358)

[6 Імпортування даних з використанням засобів обраної субд 21](#_Toc91112359)

[7 Створення Користувачів Бази даних 23](#_Toc91112360)

[7.1 Створення користувача працівник, надання прав для виконання операцій, що наводилися у розділі Аналіз предметного середовища 23](#_Toc91112361)

[7.2 Створення користувача секретар, надання прав для виконання операцій, що наводилися у розділі Аналіз предметного середовища 23](#_Toc91112362)

[7.3 Створення користувача керівник відділу, надання прав для виконання операцій, що наводилися у розділі Аналіз предметного середовища 24](#_Toc91112363)

[7.4 Створення користувача власник організації, надання прав власника бази даних для виконання операцій, що наводилися у розділі Аналіз предметного середовища 25](#_Toc91112364)

[8 SQL запити 26](#_Toc91112365)

[8.1 Створення тригерів 26](#_Toc91112366)

[8.2 Функції/процедури, що реалізують можливості користувачів у базі даних 28](#_Toc91112367)

[8.2.1 Функції/процедури, що реалізують можливості користувача “WorkerUser” 28](#_Toc91112368)

[8.2.2 Функції/процедури, що реалізують можливості користувача “SecretaryUser” 30](#_Toc91112369)

[8.2.3 Функції/процедури, що реалізують можливості користувача “DepartmentChiefUser” 31](#_Toc91112370)

[8.3 Інші види запитів для управління базою даних 32](#_Toc91112371)

[8.4 Приклад використання індексів для пришвидшення запитів 47](#_Toc91112372)

[Висновок 50](#_Toc91112373)

[Cписок літератури 51](#_Toc91112374)

Вступ

Важко уявити роботу з сучасними складними інформаційними системами без баз даних, адже вони надають можливість зручного, легкого та надійного способу керування даними та їх збереженням.

Темою даної роботи є реалізація бази даних для підтримки діловодства організації. Дана модель буде дещо спрощена та не враховуватиме усіх особливостей конкретної організації, але загалом покаже, як можна ефективно та впорядковано організовувати діловодство з її допомогою.

Варто зазначити, що ця тема сьогодні є актуальною сьогодні, адже кожна компанія має безліч документів, взаємодіє щодня з багатьма партнерами і відповідно, має багато документів, які треба зберігати та впорядковувати. І зі зростанням розміру компанії ця проблема буде ще більш відчутною.

Метою даної роботи є створення бази даних діловодства організації, яка дозволить впорядковано та зручно зберігати та переглядати інформацію про документи і все, що з ними пов’язано.

# Опис предметного середовища

Діловодство – це діяльність, що забезпечує документування, документообіг, зберігання і використання документів.

Організація взаємодіє з різними іншими організаціями, відповідно вона має вхідні документи, отримані від них, вихідні(надіслані) та внутрішні документи. Документи можуть бути різних типів: бізнес-плани, рахунки, договори та інші подібні документи. Разом з документами можуть бути додані певні файли як додатки до них. Робітники компанії отримують та надсилають документи іншим компаніям. В організації відповідальною особою за діловодство є секретар, який слідкує за документообігом та вносить всі дані про вхідні, вихідні чи внутрішні документи, додатки до них, в систему. Кожен працівник організації має доступ до перегляду усіх документів для того, щоб виконувати ті чи інші завдання по роботі. Кожним документом займається певний відділ, який складається з керівника та робітників. Власник компанії має повні права доступу на зміну системи, а керівники департаментів компанії можуть керувати відділом та його складом.

# Постановка задачі

Створити базу даних для підтримки діловодства організації. У даній базі повинна міститись інформація про документи, файли, що входять до документів, про вхідні та вихідні документи, працівників, департаменти, керівників департаментів, організації, з якими взаємодіє дана організація та їх контактні дані й розміщення, а також інформація, необхідна для взаємодій “робітник - система діловодства”, “секретар - система діловодства”, “власник організації – система діловодства”, “керівник департаменту – система діловодства”. Передбачити користувачів бази даних: секретар, працівник організації, керівник департаменту, власник організації.

# Побудова ER-моделі

## Аналіз предметного середовища

Відповідно до опису предметного середовища встановлюємо, що система підтримки діловодства має інформацію про документи, файли, що додаються до них, вхідні та вихідні документи, працівників та відділи, в яких вони працюють, типи документів, які можуть бути в організації, дані про інші організації та їх контакти, а також контакти департаментів даної організації. Працівник може переглядати різні документи та дані системи; секретар може додавати нові документи, інформацію про вхідні та вихідні документи, файли до документів, додавати нові типи документів, додавати та редагувати інформацію про інші компанії; керівник департаменту може керувати складом працівників та відділів, а також інформацією про самі відділи; власник компанії має повий доступ до редагування чи зміни системи діловодства.

Виходячи з цього, можемо виокремити таких користувачів:

* Працівник
* Секретар
* Керівник відділу
* Власник організації

Працівник матиме такі можливості:

1. Перегляд документів, файлів, списків працівиків, інформації про вхідні та вихідні документи.

Секретар матиме такі можливості:

1. Перегляд документів, файлів, списків працівиків, інформації про вхідні та вихідні документи.
2. Додавання, видалення, редагування документів
3. Додавання, видалення, редагування файлів до документів
4. Внесення та редагування інормації про вхідні та вихідні документи
5. Додавання та редагування типів документів, з якими працює компанія
6. Додавання та редагування списку компаній та інформації про них
7. Додавання та редагування номерів телефонів компаній

Керівник департаменту матиме такі можливості:

1. Перегляд документів, файлів, списків працівиків, інформації про вхідні та вихідні документи.
2. Редагування списку телефонів департаментів
3. Редагування даних департаментів
4. Редагування списку робітників та їх входження до департаментів

Власник організації матиме такі можливості:

1. Перегляд документів, файлів, списків працівиків, інформації про вхідні та вихідні документи.
2. Отримання та редагування різної інформації системи

## Бізнес-правила

1. Дата відправлення документу не може бути меншою за дату отримання документу
2. Дата відправлення та дата отримання документів менші за поточну дату
3. Документ містить мінімальну кількість символів, що бути збереженим в системі

## Вибір сутностей

Після аналізу предметного середовища виділяємо такі сутності:

* Документ
* Файл документа
* Тип документа
* Вхідний документ
* Вихідний документ
* Журнал документів
* Працівник
* Працівник відділу
* Організація
* Телефон організації
* Відділ
* Телефон відділу

## Набори атрибутів сутностей

– набори атрибутів виділених сутностей

|  |  |
| --- | --- |
| Сутність | Набір атрибутів |
| Документ | * id документа * id типу * назва відділу * текст документу |
| Файл документа | * id документа * назва файлу * розширення файлу |
| Тип документа | * id типу * назва типу |
| Вхідний документ | * id вхідного документа * id документа * id організації * id отримувача * дата отримання |
| Вихідний документ | * id вихідного документа * id документа * id організації * id відправника * дата відправлення |

Продовження таблиці 3.1

|  |  |
| --- | --- |
| Журнал документів | * id вхідного документа * id вихідного документа |
| Працівник | * id працівника * прізвище * ім’я * по-батькові * посада * телефон |
| Працівник відділу | * назва відділу * id працівника |
| Організація | * id організації * назва організації * місто * вулиця * номер вулиці * електронна пошта |
| Телефон організації | * id організації * номер телефона |
| Відділ | * назва відділу * id начальника |
| Телефон відділу | * назва відділу * телефон |

## Зв’язки між об’єктами

Сутність Документ буде пов’язана зв’язком один до багатьох або до нуля з сутністю Файл документа, бо документ може мати декілька файлів, або не мати взагалі, один до одного або нуля з сутностями: Вхідний документ, Вихідний документ, адже документ може бути як внутрішнім, так і вхідним або вихідним.

Сутність Тип документа буде пов’язана зв’язком один до багатьох або до нуля з сутністю Документ, бо може бути багато документів одного типу або жодного.

Сутність Вхідний документ буде пов’язана зв’язком один до багатьох або до нуля з сутністю Журнал документів, бо одному вхідному може відповідати декілька вихідних документів, або не буде жодного.

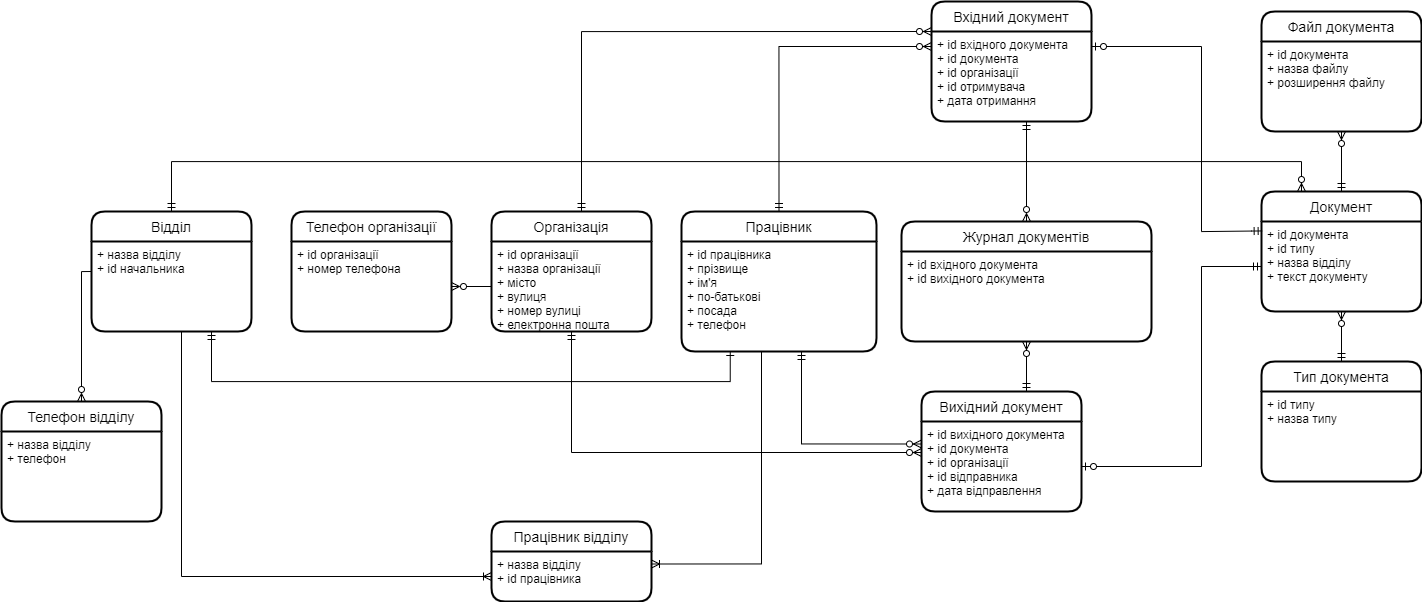
Сутність Вихідний документ буде пов’язана зв’язком один до багатьох або до нуля з сутністю Журнал документів, бо одному вихідному може відповідати декілька вхідних документів, або не буде жодного.

Сутність Працівник буде пов’язана зв’язком один до багатьох з сутністю Прцівник відділу, бо працівник може працювати у декількох відділах, один до багатьох або до нуля з сутностями: Вхідний документ, Вихідний документ, адже працівник міг отримувати чи надсилати декілька документів або жодного.

Сутність Організація буде пов’язана зв’язком один до багатьох або до нуля з сутностями: Вхідний документ, Вихідний документ, адже Організація могла надсилати або отримувати документи, або ще не отримала чи не надіслала жодного, один до багатьох або до нуля з сутністю Телефон організації, бо оганізація може мати декілька телефонів, або не збережено в системі жодного.

Сутність Відділ буде пов’язана зв’язком один до багатьох або до нуля з сутністю Документ, бо відділ міг знайматися декількома документами, або ще не займався жодним, один до одного з сутністю Працівник, бо кожен відділ має керівника, один до багатьох з сутністю Працівник відділу, бо відділ складається з декількох працівників, один до багатьох або нуля з сутністю телефон Відділу, бо відділ може мати декілька телефонів, або не збережено в системі жодного.

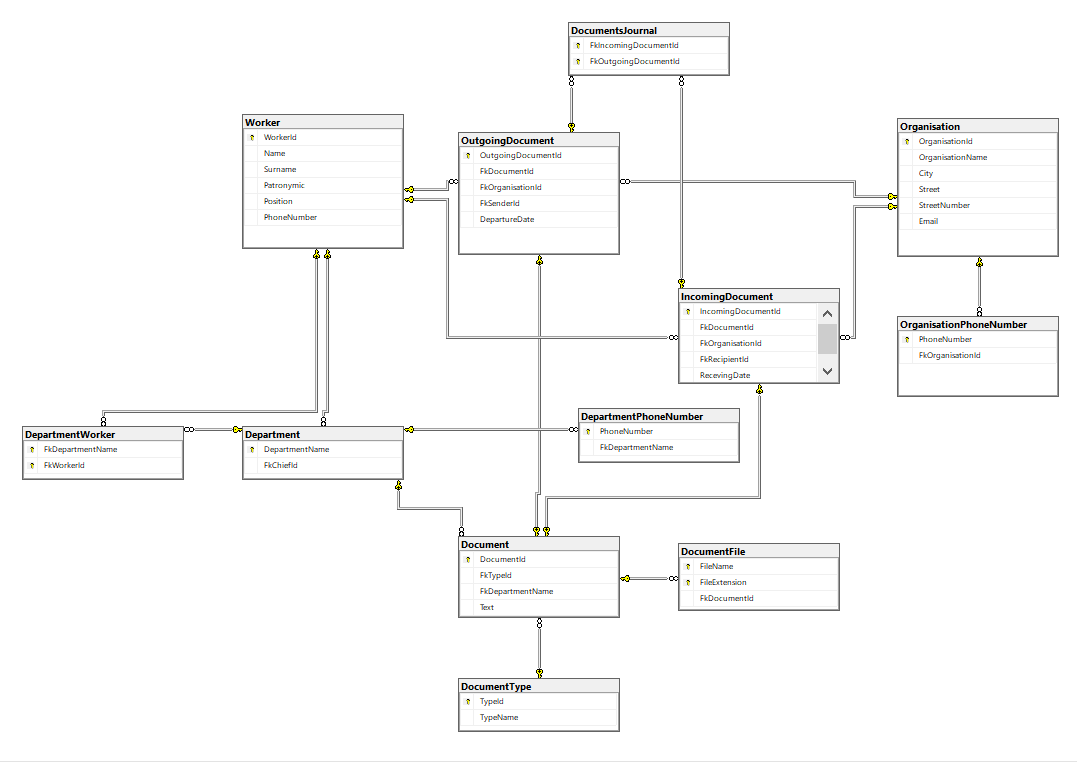
## ER-модель



– ER-модель

# Побудова реляційної схеми

## Побудова необхідних відношень та визначення первинних і зовнішніх ключів



– Реляційна схема бази даних

З даної схеми можна побачити, що база даних знаходиться у 3 нормальній формі, бо всі поля таблиць декомпозовані, а також всі атрибути таблиць залежать від первинного ключа, кожен неключовий атрибут не є транзитивно залежним від первинного ключа.

## Визначення обмежень цілісності для спроектованих відношень

Обмеження цілісності:

1. Атрибути таблиць, які є обов’язковими, мають обмеження NOT NULL, це запобігає додаванню рядку без даного значення, а також уникненню помилок при роботі з даними.
2. Завдяки відсутності ON DELETE CASCADE рядок батьківської таблиці може бути видаленим лише тоді, коли на нього немає зовнішніх ключів, що посилаються.
3. Телефони, типи документів, зовнішні ключі документів мають обмеження унікальності, бо телефон не може бути однаковий у різних компаній, а один і той же документ не можу бути вхідним двічі.

# Створення бази даних у форматі системи управління базою даних MS SQL Server

Запити для створення бази даних та її таблиць:

-- Створення таблиці Worker

CREATE TABLE Worker(

WorkerId int IDENTITY(1,1) PRIMARY KEY,

Name nvarchar(50) NOT NULL,

Surname nvarchar(50) NOT NULL,

Patronymic nvarchar(50),

Position nvarchar(50) NOT NULL,

PhoneNumber varchar(50) NOT NULL UNIQUE

)

GO

ALTER TABLE Worker

ADD CONSTRAINT CHK\_WorkerName

CHECK (Name LIKE '%[A-Za-z]%' AND Name NOT LIKE '%[^A-Za-z-]%')

GO

ALTER TABLE Worker

ADD CONSTRAINT CHK\_WorkerSurname

CHECK (Surname LIKE '%[A-Za-z]%' AND Surname NOT LIKE '%[^A-Za-z'' -]%')

GO

ALTER TABLE Worker

ADD CONSTRAINT CHK\_WorkerPatronymic

CHECK (Patronymic LIKE '%[A-Za-z]%' AND Patronymic NOT LIKE '%[^A-Za-z'' -]%')

GO

ALTER TABLE Worker

ADD CONSTRAINT CHK\_WorkerPhoneNumber CHECK (PhoneNumber LIKE '+%-%-%-%' AND PhoneNumber NOT LIKE '%[a-Z]%')

GO

-- Створення таблиці організація

CREATE TABLE Organisation(

OrganisationId int IDENTITY(1,1) PRIMARY KEY,

OrganisationName nvarchar(50),

City nvarchar(50) NOT NULL,

Street nvarchar(50) NOT NULL,

StreetNumber int NOT NULL,

Email nvarchar(50) NOT NULL

)

GO

ALTER TABLE Organisation

ADD CONSTRAINT CHK\_EmailFormat CHECK (Email LIKE '%\_@\_%\_.\_\_%')

GO

ALTER TABLE Organisation

ADD CONSTRAINT CHK\_OrganisationName CHECK (OrganisationName NOT LIKE '%[0-9]%' AND LEN(OrganisationName)>=5)

GO

ALTER TABLE Organisation

ADD CONSTRAINT CHK\_OrganisationCity CHECK (City NOT LIKE '%[0-9]%' AND LEN(City)>=3)

GO

ALTER TABLE Organisation

ADD CONSTRAINT CHK\_OrganisationStreet CHECK (LEN(Street)>=1)

GO

ALTER TABLE Organisation

ADD CONSTRAINT CHK\_OrganisationStreetNumber CHECK (StreetNumber>=0)

GO

--Створення таблиці Телефон організації

CREATE TABLE OrganisationPhoneNumber(

PhoneNumber varchar(50) NOT NULL PRIMARY KEY,

FkOrganisationId int NOT NULL FOREIGN KEY REFERENCES Organisation(OrganisationId)

)

GO

ALTER TABLE OrganisationPhoneNumber

ADD CONSTRAINT CHK\_OrganisationPhoneNumber CHECK (PhoneNumber LIKE '+%-%-%-%' AND PhoneNumber NOT LIKE '%[a-Z]%')

GO

--Створення таблиці відділ

CREATE TABLE Department(

DepartmentName nvarchar(50) NOT NULL PRIMARY KEY,

FkChiefId int NOT NULL FOREIGN KEY REFERENCES Worker(WorkerId)

)

GO

--Створення таблиці телефон відділу

CREATE TABLE DepartmentPhoneNumber(

PhoneNumber varchar(50) NOT NULL PRIMARY KEY,

FkDepartmentName nvarchar(50) NOT NULL

)

GO

ALTER TABLE DepartmentPhoneNumber

WITH CHECK ADD CONSTRAINT FK\_DepartmentName\_DepartmentPhoneNumber FOREIGN KEY(FkDepartmentName)

REFERENCES Department(DepartmentName)

ON UPDATE CASCADE

GO

ALTER TABLE DepartmentPhoneNumber

ADD CONSTRAINT CHK\_DepartmentPhoneNumber CHECK (PhoneNumber LIKE '+%-%-%-%' AND PhoneNumber NOT LIKE '%[a-Z]%')

GO

--Створення таблиці робітник віддділу

CREATE TABLE DepartmentWorker(

FkDepartmentName nvarchar(50) NOT NULL,

FkWorkerId int NOT NULL FOREIGN KEY REFERENCES Worker(WorkerId)

CONSTRAINT PkDepartmentWorker PRIMARY KEY(FkDepartmentName, FkWorkerId)

)

GO

ALTER TABLE DepartmentWorker

WITH CHECK ADD CONSTRAINT FK\_DepartmentName FOREIGN KEY(FkDepartmentName)

REFERENCES Department(DepartmentName)

ON UPDATE CASCADE

Go

--Створення таблиці тип документу

CREATE TABLE DocumentType(

TypeId int IDENTITY(1,1) PRIMARY KEY,

TypeName nvarchar(50) NOT NULL UNIQUE

)

GO

ALTER TABLE DocumentType

ADD CONSTRAINT CHK\_DocumentTypeMinLength

CHECK (LEN(DocumentType.TypeName)>=1)

GO

--Створення таблиці документ

CREATE TABLE Document(

DocumentId int IDENTITY(1,1) PRIMARY KEY,

FkTypeId int NOT NULL FOREIGN KEY REFERENCES DocumentType(TypeId),

FkDepartmentName nvarchar(50) NOT NULL,

Text nvarchar(2000) NOT NULL

)

GO

ALTER TABLE Document

WITH CHECK ADD CONSTRAINT FK\_DepartmentName\_DocumentTable FOREIGN KEY(FkDepartmentName)

REFERENCES Department(DepartmentName)

ON UPDATE CASCADE

ALTER TABLE Document

ADD CONSTRAINT CHK\_DocumentTextMinLength

CHECK (LEN(Document.Text)>=50)

GO

--Створення таблиці Файл документу

CREATE TABLE DocumentFile(

FileName nvarchar(50) NOT NULL,

FileExtension nvarchar(50) NOT NULL,

FkDocumentId int NOT NULL FOREIGN KEY REFERENCES Document(DocumentId),

CONSTRAINT PkDocumentFile PRIMARY KEY(FileName, FileExtension),

)

GO

ALTER TABLE DocumentFile

ADD CONSTRAINT CHK\_DocumentFileMinLength

CHECK ((FileName LIKE ('%\_%')) AND (FileExtension LIKE ('%\_%')))

GO

--Створення таблиці Вхідний документ

CREATE TABLE IncomingDocument(

IncomingDocumentId int IDENTITY(1,1) PRIMARY KEY,

FkDocumentId int NOT NULL UNIQUE FOREIGN KEY REFERENCES Document(DocumentId),

FkOrganisationId int NOT NULL FOREIGN KEY REFERENCES Organisation(OrganisationId),

FkRecipientId int NOT NULL FOREIGN KEY REFERENCES Worker(WorkerId),

RecevingDate date NOT NULL DEFAULT(GETDATE())

)

GO

ALTER TABLE IncomingDocument ADD CONSTRAINT CHK\_ReceivingDate CHECK (RecevingDate <= CONVERT(date, GETDATE()))

GO

--Створення таблиці вихідний документ

CREATE TABLE OutgoingDocument(

OutgoingDocumentId int IDENTITY(1,1) PRIMARY KEY,

FkDocumentId int NOT NULL UNIQUE FOREIGN KEY REFERENCES Document(DocumentId),

FkOrganisationId int NOT NULL FOREIGN KEY REFERENCES Organisation(OrganisationId),

FkSenderId int NOT NULL FOREIGN KEY REFERENCES Worker(WorkerId),

DepartureDate date NOT NULL DEFAULT(GETDATE())

)

GO

ALTER triggTABLE OutgoingDocument ADD CONSTRAINT CHK\_DepartureDate CHECK (DepartureDate <= CONVERT(date, GETDATE()))

GO

--Створення таблиці журнал документів

CREATE TABLE DocumentsJournal(

FkIncomingDocumentId int NOT NULL FOREIGN KEY REFERENCES IncomingDocument(IncomingDocumentId),

FkOutgoingDocumentId int NOT NULL FOREIGN KEY REFERENCES OutgoingDocument(OutgoingDocumentId),

CONSTRAINT PkDocumentsJournal PRIMARY KEY(FkIncomingDocumentId, FkOutgoingDocumentId)

)

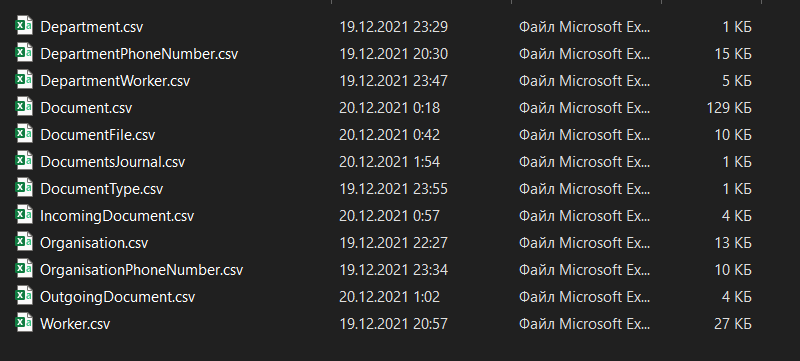
GO

# Імпортування даних з використанням засобів обраної субд

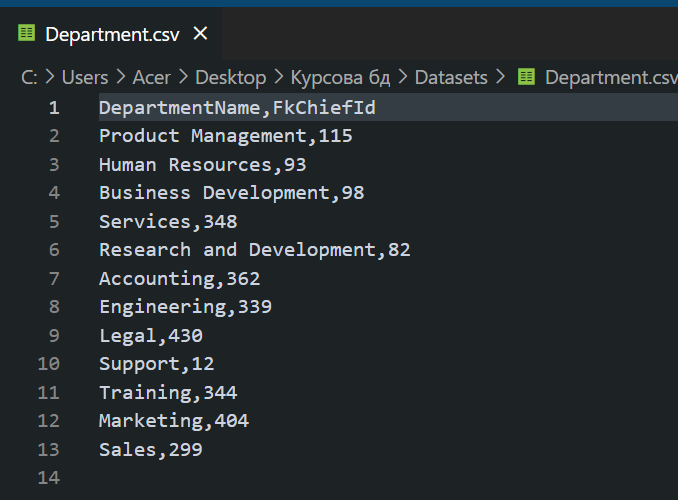
Для імпортування даних у СУБД Microsoft SQL Server існує декілька методів:

* Імпорт даних кожної таблиці з .csv файлів
* Імпорт даних з файлу скрипта .sql

Було обрано перший спосіб, імпорт таблиць з .csv файлів, які було згенеровано веб-сервісом Mockaroo. Кожну таблицю було імпортовано з даних файлів з допомогою майстра імпорту та експорт SQL Server:



- .csv файли з даними таблиць



– вміст одного з .csv файлів

Джерелом імпорту було обрано Flat file source та вказано шлях до файлу, а призначенням було вказано SQL server client та таблицю, куди відбувається імпорт.

# Створення Користувачів Бази даних

## Створення користувача працівник, надання прав для виконання операцій, що наводилися у розділі Аналіз предметного середовища

CREATE LOGIN Worker WITH PASSWORD='Worker'

CREATE USER WorkerUser FOR LOGIN Worker

GRANT SELECT ON DocumentsJournal TO WorkerUser

GRANT SELECT ON IncomingDocument TO WorkerUser

GRANT SELECT ON OutgoingDocument TO WorkerUser

GRANT SELECT ON DepartmentWorker TO WorkerUser

GRANT SELECT ON DepartmentPhoneNumber TO WorkerUser

GRANT SELECT ON DocumentFile TO WorkerUser

GRANT SELECT ON Document TO WorkerUser

GRANT SELECT ON DocumentType TO WorkerUser

GRANT SELECT ON Department TO WorkerUser

GRANT SELECT ON Worker TO WorkerUser

GRANT SELECT ON OrganisationPhoneNumber TO WorkerUser

GRANT SELECT ON Organisation TO WorkerUser

GRANT SELECT ON getDocumentFiles TO WorkerUser

GRANT SELECT ON getIncomingDocumentInfo TO WorkerUser

GRANT SELECT ON getOutgoingDocumentInfo TO WorkerUser

GRANT EXECUTE ON getAverageFilesPerDocumentStatistic TO WorkerUser

GRANT SELECT ON getOutgoingInfoLastMonth TO WorkerUser

GRANT SELECT ON getIncomingInfoLastMonth TO WorkerUser

## Створення користувача секретар, надання прав для виконання операцій, що наводилися у розділі Аналіз предметного середовища

CREATE LOGIN Secretary WITH PASSWORD='Secretary'

CREATE USER SecretaryUser FOR LOGIN Secretary

GRANT SELECT, INSERT, UPDATE, DELETE ON DocumentsJournal TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON IncomingDocument TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON OutgoingDocument TO SecretaryUser

GRANT SELECT ON DepartmentWorker TO SecretaryUser

GRANT SELECT ON DepartmentPhoneNumber TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON DocumentFile TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON Document TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON DocumentType TO SecretaryUser

GRANT SELECT ON Department TO SecretaryUser

GRANT SELECT ON Worker TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON OrganisationPhoneNumber TO SecretaryUser

GRANT SELECT, INSERT, UPDATE, DELETE ON Organisation TO SecretaryUser

GRANT SELECT ON getDocumentFiles TO SecretaryUser

GRANT SELECT ON getIncomingDocumentInfo TO SecretaryUser

GRANT SELECT ON getOutgoingDocumentInfo TO SecretaryUser

GRANT EXECUTE ON getAverageFilesPerDocumentStatistic TO SecretaryUser

GRANT SELECT ON getOutgoingInfoLastMonth TO SecretaryUser

GRANT SELECT ON getIncomingInfoLastMonth TO SecretaryUser

GRANT EXECUTE ON InsertOrganisationPhoneNumber TO SecretaryUser

GRANT EXECUTE ON setOutgoingDate TO SecretaryUser

GRANT EXECUTE ON setIncomingDate TO SecretaryUser

## Створення користувача керівник відділу, надання прав для виконання операцій, що наводилися у розділі Аналіз предметного середовища

CREATE LOGIN DepartmentChief WITH PASSWORD='DepartmentChief'

CREATE USER DepartmentChiefUser FOR LOGIN DepartmentChief

GRANT SELECT ON DocumentsJournal TO DepartmentChiefUser

GRANT SELECT ON IncomingDocument TO DepartmentChiefUser

GRANT SELECT ON OutgoingDocument TO DepartmentChiefUser

GRANT SELECT, INSERT, UPDATE, DELETE ON DepartmentWorker TO DepartmentChiefUser

GRANT SELECT, INSERT, UPDATE, DELETE ON DepartmentPhoneNumber TO DepartmentChiefUser

GRANT SELECT ON DocumentFile TO DepartmentChiefUser

GRANT SELECT ON Document TO DepartmentChiefUser

GRANT SELECT ON DocumentType TO DepartmentChiefUser

GRANT SELECT, INSERT, UPDATE, DELETE ON Department TO DepartmentChiefUser

GRANT SELECT, INSERT, UPDATE, DELETE ON Worker TO DepartmentChiefUser

GRANT SELECT ON OrganisationPhoneNumber TO DepartmentChiefUser

GRANT SELECT ON Organisation TO DepartmentChiefUser

GRANT SELECT ON getDocumentFiles TO DepartmentChiefUser

GRANT SELECT ON getIncomingDocumentInfo TO DepartmentChiefUser

GRANT SELECT ON getOutgoingDocumentInfo TO DepartmentChiefUser

GRANT EXECUTE ON getAverageFilesPerDocumentStatistic TO DepartmentChiefUser

GRANT SELECT ON getOutgoingInfoLastMonth TO DepartmentChiefUser

GRANT SELECT ON getIncomingInfoLastMonth TO DepartmentChiefUser

GRANT EXECUTE ON InsertOrganisationPhoneNumber TO DepartmentChiefUser

GRANT EXECUTE ON InsertDepartmentPhoneNumber TO DepartmentChiefUser

GRANT EXECUTE ON renameDepartment TO DepartmentChiefUser

## Створення користувача власник організації, надання прав власника бази даних для виконання операцій, що наводилися у розділі Аналіз предметного середовища

CREATE LOGIN OrganisationOwner WITH PASSWORD='OrganisationOwner'

CREATE USER OrganisationOwnerUser FOR LOGIN OrganisationOwner

EXEC sp\_addrolemember N'db\_owner', N'OrganisationOwnerUser'

# SQL запити

## Створення тригерів

Тригер для перевірки чи дата отримання вхідного документа менша за дату відправки відповідного вихідного документа.

CREATE TRIGGER OnInsertDocumentsJournal

ON DocumentsJournal

INSTEAD OF INSERT

AS

BEGIN

DECLARE @IncomingId int

DECLARE @OutgoingId int

DECLARE @IncomingDate date

DECLARE @OutgoingDate date

DECLARE del\_cursor CURSOR LOCAL FOR

SELECT FkIncomingDocumentId, FkOutgoingDocumentId from inserted

OPEN del\_cursor

FETCH NEXT FROM del\_cursor INTO @IncomingId, @OutgoingId

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @IncomingDate=(SELECT RecevingDate FROM IncomingDocument WHERE IncomingDocumentId=@IncomingId)

SET @OutgoingDate=(SELECT DepartureDate FROM OutgoingDocument WHERE OutgoingDocumentId=@OutgoingId)

if @IncomingDate < @OutgoingDate

insert DocumentsJournal(FkIncomingDocumentId, FkOutgoingDocumentId) values (@IncomingId, @OutgoingId)

ELSE THROW 50005, N'Error! Incoming date is less than departure', 1;

FETCH NEXT FROM del\_cursor INTO @IncomingId, @OutgoingId

END

CLOSE del\_cursor

DEALLOCATE del\_cursor

END

GO

Тригер для перевірки чи номер, що вставляється в таблицю номерів телефонів організацій, не використовується у таблиці відділів нашої організації.

CREATE TRIGGER OnInsertOrganisationPhoneNumber

ON OrganisationPhoneNumber

INSTEAD OF INSERT

AS

BEGIN

DECLARE @PhoneNum varchar(50)

DECLARE @OrganisationId int

DECLARE del\_cursor CURSOR LOCAL FOR

SELECT PhoneNumber, FkOrganisationId from inserted

OPEN del\_cursor

FETCH NEXT FROM del\_cursor INTO @PhoneNum, @OrganisationId

WHILE @@FETCH\_STATUS = 0

BEGIN

if @PhoneNum NOT IN (SELECT PhoneNumber FROM DepartmentPhoneNumber)

BEGIN

insert OrganisationPhoneNumber(PhoneNumber, FkOrganisationId) values (@PhoneNum, @OrganisationId)

END

ELSE THROW 50005, N'Error! This phone number is used in another table', 1;

FETCH NEXT FROM del\_cursor INTO @PhoneNum, @OrganisationId

END

CLOSE del\_cursor

DEALLOCATE del\_cursor

END

GO

Тригер для перевірки чи номер, що вставляється в таблицю номерів телефонів відділів, не використовується у таблиці номерів телефонів інших організацій.

CREATE TRIGGER OnInsertDepartmentPhoneNumber

ON DepartmentPhoneNumber

INSTEAD OF INSERT

AS

BEGIN

DECLARE @PhoneNum varchar(50)

DECLARE @DepartmentName nvarchar(50)

DECLARE del\_cursor CURSOR LOCAL FOR

SELECT PhoneNumber, FkDepartmentName from inserted

OPEN del\_cursor

FETCH NEXT FROM del\_cursor INTO @PhoneNum, @DepartmentName

WHILE @@FETCH\_STATUS = 0

BEGIN

if @PhoneNum NOT IN (SELECT PhoneNumber FROM OrganisationPhoneNumber)

BEGIN

insert DepartmentPhoneNumber(PhoneNumber, FkDepartmentName) values (@PhoneNum, @DepartmentName)

END

ELSE THROW 50005, N'Error! This phone number is used in another table', 1;

FETCH NEXT FROM del\_cursor INTO @PhoneNum, @DepartmentName

END

CLOSE del\_cursor

DEALLOCATE del\_cursor

END

GO

## Функції/процедури, що реалізують можливості користувачів у базі даних

### Функції/процедури, що реалізують можливості користувача “WorkerUser”

Функція, що реалізує можливість перегляду всіх файлів даного документу

CREATE FUNCTION getDocumentFiles(@DocumentId int)

RETURNS TABLE

AS

RETURN (

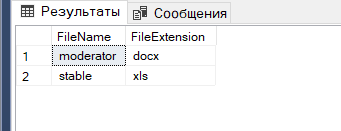
select FileName, FileExtension

from DocumentInfo

where @DocumentId=DocumentId

)

Результат виклику даної функції:



– результат виклику функції getDocumentFiles() для файлу з id 7

Функція, що реалізує можливість перегляду інформації про вхідний

документ

CREATE FUNCTION getIncomingDocumentInfo(@IncomingDocumentId int)

RETURNS TABLE

AS

RETURN (

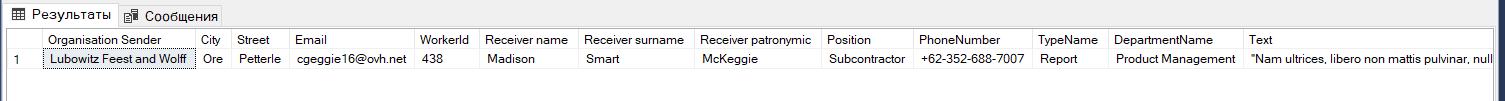
select DISTINCT OrganisationName[Organisation Sender], City, Street, Email, WorkerId, Name [Receiver name], Surname [Receiver surname], Patronymic [Receiver patronymic], Position, PhoneNumber, TypeName, DepartmentName, Text, RecevingDate

from IncomingDocumentInfo

where @IncomingDocumentId=IncomingDocumentId

)

Результат виклику даної функції:



– результат виклику функції getIncomingDocumentInfo() для вхідного документа з id 5

Функція, що реалізує можливість перегляду інформації про вихідний документ

CREATE FUNCTION getOutgoingDocumentInfo(@OutgoingDocumentId int)

RETURNS TABLE

AS

RETURN (

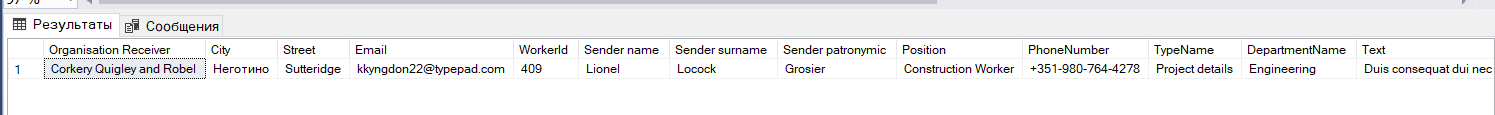
select DISTINCT OrganisationName[Organisation Receiver], City, Street, Email, WorkerId, Name [Sender name], Surname [Sender surname], Patronymic [Sender patronymic], Position, PhoneNumber, TypeName, DepartmentName, Text, DepartureDate

from OutgoingDocumentInfo

where @OutgoingDocumentId=OutgoingDocumentId

)

Результат виклику даної функції:



– результат виклику функції getOutgoingDocumentInfo() для вихідного документа з id 5

Функція, що реалізує можливість перегляду середньої кількості файлів,

що має кожен документ

CREATE FUNCTION getAverageFilesPerDocumentStatistic()RETURNS intASBEGIN DECLARE @tempTable table(DocumentsNumber int) DECLARE @Result int INSERT INTO @tempTable SELECT COUNT(\*) FROM DocumentInfo DocumentsNumber GROUP BY DocumentId SET @Result = (SELECT AVG(DocumentsNumber) FROM @tempTable) RETURN @ResultEND

### Функції/процедури, що реалізують можливості користувача “SecretaryUser”

Процедура, що реалізовує можливість встановлювати дату отримання документу

CREATE PROCEDURE setIncomingDate @IncomingDocumentId int, @date date

AS

BEGIN

UPDATE IncomingDocument SET RecevingDate = @date

WHERE IncomingDocumentId = @IncomingDocumentId

END

Процедура, що реалізовує можливість встановлювати дату відправлення документу

CREATE PROCEDURE setOutgoingDate @OutgoingDocumentId int, @date date

AS

BEGIN

UPDATE OutgoingDocument SET DepartureDate = @date

WHERE OutgoingDocumentId = @OutgoingDocumentId

END

Функція, що дає можливість додавати телефон організації

CREATE PROCEDURE InsertOrganisationPhoneNumber @PhoneNumber varchar(50), @OrganisationId int

AS

BEGIN

INSERT INTO OrganisationPhoneNumber(PhoneNumber, FkOrganisationId) VALUES(@PhoneNumber, @OrganisationId)

END

Функція, що дає можливість перегляду списку вихідних документів протягом заданої кількості останніх місяців

CREATE FUNCTION getOutgoingInfoLastMonth(@MonthNumber int)

RETURNS TABLE

AS

RETURN (

SELECT \*

FROM OutgoingDocumentInfo

WHERE ABS(DATEDIFF(MONTH, GETDATE(), DepartureDate))<=ABS(@MonthNumber))

Функція, що дає можливість перегляду списку вхідних документів протягом заданої кількості останніх місяців

CREATE FUNCTION getIncomingInfoLastMonth(@MonthNumber int)

RETURNS TABLE

AS

RETURN (

SELECT \*

FROM IncomingDocumentInfo

WHERE ABS(DATEDIFF(MONTH, GETDATE(), RecevingDate))<=ABS(@MonthNumber)

)

### Функції/процедури, що реалізують можливості користувача “DepartmentChiefUser”

Функція, що дає можливість переглядати всіх робітників конкретного відділу

CREATE FUNCTION getWorkersOfDepartment(@DepartmentName nvarchar(50))

RETURNS TABLE

AS

RETURN (

SELECT WorkerId, Name, Surname, Patronymic, Position, PhoneNumber, FkChiefId

FROM DepartmentWorkersInfo

WHERE @DepartmentName=DepartmentName

)

Процедура, що дає можливість перейменовувати відділ

CREATE PROCEDURE renameDepartment @OldDepartmentName nvarchar(50), @NewDepartmentName nvarchar(50)

AS

BEGIN

UPDATE Department SET DepartmentName = @NewDepartmentName

WHERE DepartmentName = @OldDepartmentName

END

Процедура, що дає можливість додавати телефон відділу

CREATE PROCEDURE InsertDepartmentPhoneNumber @PhoneNumber varchar(50), @DepartmentName nvarchar(50)

AS

BEGIN

INSERT INTO DepartmentPhoneNumber(PhoneNumber, FkDepartmentName) VALUES(@PhoneNumber, @DepartmentName)

END

Процедура, що дає можливість змінювати телефон відділу

CREATE PROCEDURE changeDepartmentPhoneNumber @NewPhoneNumber varchar(50), @OldPhoneNumber varchar(50), @DepartmentName nvarchar(50)

AS

BEGIN

UPDATE DepartmentPhoneNumber SET PhoneNumber = @NewPhoneNumber

WHERE FkDepartmentName = @DepartmentName AND PhoneNumber=@OldPhoneNumber

END

## Інші види запитів для управління базою даних

Запит, що виводить назву вхідного та назву відповідного йому вихідного файлу

SELECT IncomingFile, OutgoingFile

FROM

DocumentsJournal

JOIN (SELECT TRIM(FileName)+'.'+FileExtension IncomingFile, IncomingDocumentId

FROM IncomingDocument

JOIN Document ON FkDocumentId=DocumentId

JOIN DocumentFile ON DocumentId=DocumentFile.FkDocumentId) tab1

ON tab1.IncomingDocumentId=FkIncomingDocumentId

JOIN (SELECT TRIM(FileName)+'.'+FileExtension OutgoingFile, OutgoingDocumentId

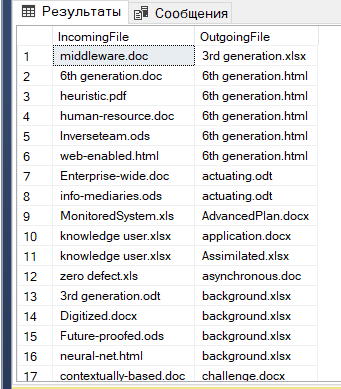
FROM OutgoingDocument

JOIN Document ON FkDocumentId=DocumentId

JOIN DocumentFile ON DocumentId=DocumentFile.FkDocumentId) tab2

ON tab2.OutgoingDocumentId=FkOutgoingDocumentId

GO



– результат запиту

Запит, що виводить інформацію про наявні документи та файли, прикріплені до них(id та тип документа та файли, що прикріплені до кожного документу)

SELECT DocumentId, TypeName, FileName as Name, FileExtension as Extension

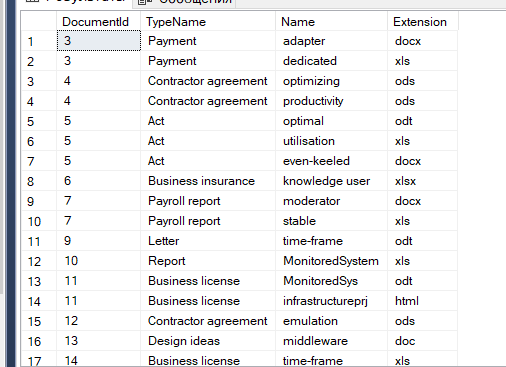
FROM Document

JOIN DocumentFile ON DocumentId=FkDocumentId

JOIN DocumentType ON FkTypeId=TypeId

ORDER BY DocumentId

GO



– результат запиту

Запит, що виводить вміст документу, його тип та відділ, що займається цим документом

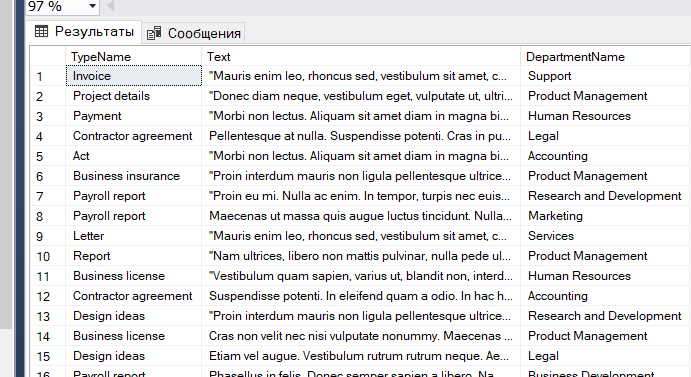
SELECT TypeName, Text, FkDepartmentName as DepartmentName

FROM Document

JOIN DocumentType ON FkTypeId=TypeId

ORDER BY DocumentId

GO



– результат запиту

Запит, який виводить керівників та назви відділів, якими вони керують

SELECT Name +' '+Surname+' '+Patronymic as [Chief FIO], DepartmentName

FROM Document

JOIN Department ON FkDepartmentName=DepartmentName

JOIN Worker ON WorkerId=FkChiefId

GROUP BY DepartmentName, Name, Surname, Patronymic

GO



– результат запиту

Запит виводить список організацій та номери телефонів цих компаній, які надсилали даній організації будь-які документи

SELECT OrganisationName, OrganisationPhoneNumber.PhoneNumber

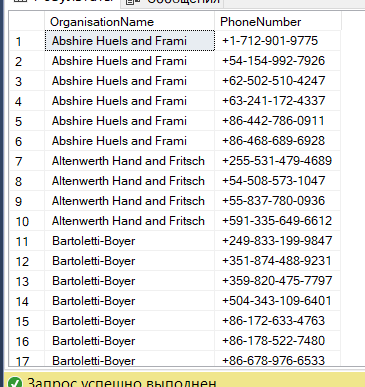
FROM IncomingDocument

JOIN Organisation ON FkOrganisationId=OrganisationId

LEFT JOIN OrganisationPhoneNumber ON OrganisationPhoneNumber.FkOrganisationId=OrganisationId

GROUP BY OrganisationName, PhoneNumber

GO



- результат запиту

Запит виводить список компаній та номери телефонів цих компаній, яким були надіслані коли-небудь документи

SELECT OrganisationName, OrganisationPhoneNumber.PhoneNumber

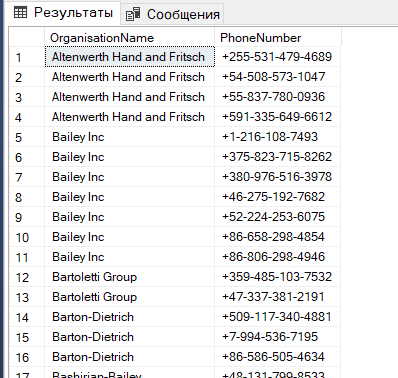
FROM OutgoingDocument

JOIN Organisation ON FkOrganisationId=OrganisationId

LEFT JOIN OrganisationPhoneNumber ON OrganisationPhoneNumber.FkOrganisationId=OrganisationId

GROUP BY OrganisationName, PhoneNumber

GO



– результат запиту

Запит, що виводить інформацію про внутрішні документи компанії(документ та його файли)

SELECT DocumentId, TypeName, Text, FileName as Name, FileExtension as Extension

FROM Document

JOIN DocumentFile ON DocumentId=FkDocumentId

JOIN DocumentType ON FkTypeId=TypeId

WHERE DocumentId NOT IN

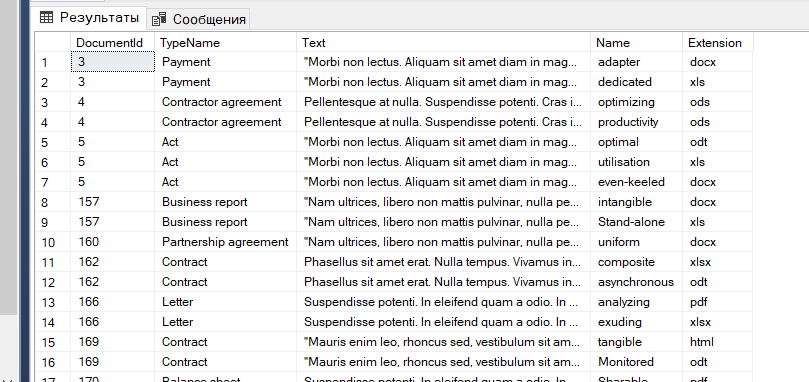
((SELECT DocumentId FROM Document JOIN IncomingDocument ON DocumentId=FkDocumentId)

UNION

(SELECT DocumentId FROM Document JOIN OutgoingDocument ON DocumentId=FkDocumentId))

ORDER BY DocumentId

GO



– результат запиту

Запит, що виводить всі відділи та їх номери телефонів

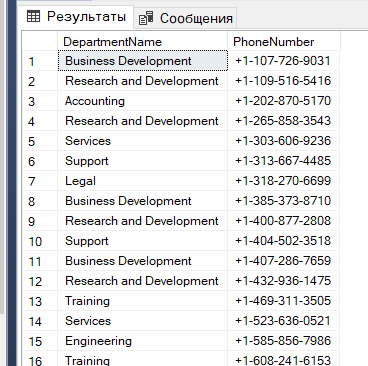
SELECT DepartmentName, PhoneNumber

FROM Department

JOIN DepartmentPhoneNumber ON DepartmentName=FkDepartmentName

GROUP BY DepartmentName, PhoneNumber

GO



– результат запиту

Запит, що виводить усіх працівників, їх відділи та їх начальників

SELECT workerDepartment.Name, workerDepartment.Surname, workerDepartment.Position, DepartmentName, chief.Name [Chief name], chief.Surname [Chief surname], chief.PhoneNumber [Chief phone number]

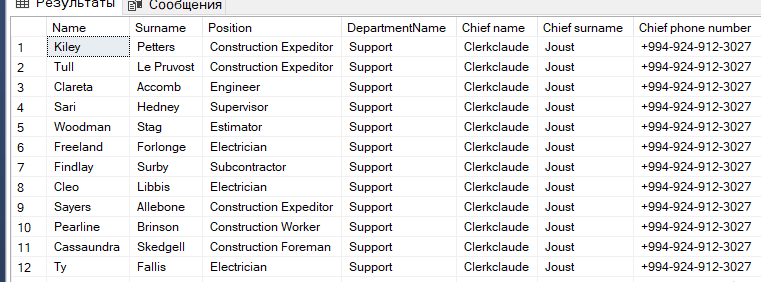
FROM DepartmentWorker

JOIN Worker workerDepartment ON WorkerId=FkWorkerId

JOIN Department ON FkDepartmentName=DepartmentName

JOIN Worker chief ON FkChiefId=chief.WorkerId

GO



– результат запиту

Запит, що виводить працівників, кількість прийнятих та відправлених ними документів

SELECT Name, Surname, Patronymic, PhoneNumber, COUNT(inDoc.FkDocumentId) [Number of received docs], COUNT(outDoc.FkDocumentId) [Number of sent docs]

FROM Worker

LEFT JOIN IncomingDocument inDoc ON FkRecipientId=WorkerId

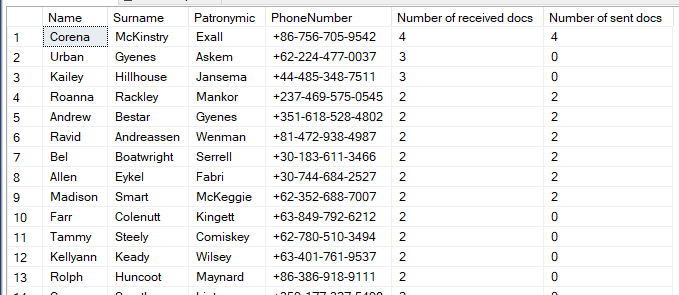
LEFT JOIN OutgoingDocument outDoc ON FkSenderId=WorkerId

GROUP BY WorkerId, Name, Surname, Patronymic, PhoneNumber

HAVING COUNT(inDoc.FkDocumentId)>0 OR COUNT(outDoc.FkDocumentId) >0

ORDER BY [Number of received docs] DESC, [Number of sent docs] DESC

GO



– результат запиту

Запит, що виводить тільки вхідні документи

SELECT TypeName [Incoming document type], Text, FileName as Name, FileExtension as Extension

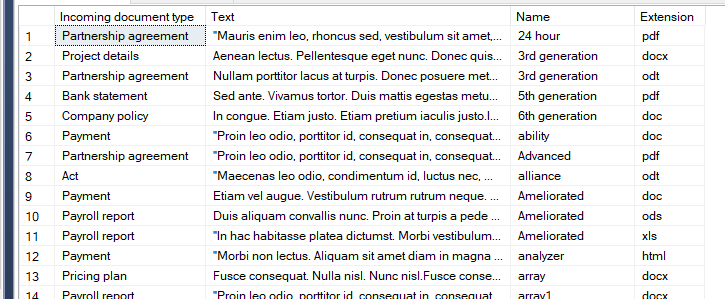
FROM Document

JOIN DocumentFile ON DocumentId=FkDocumentId

JOIN IncomingDocument ON IncomingDocument.FkDocumentId=DocumentId

JOIN DocumentType ON FkTypeId=TypeId

GO



– результат запиту

Запит, що виводить тільки вихідні документи

SELECT TypeName [Outgoing document type], Text, FileName as Name, FileExtension as Extension

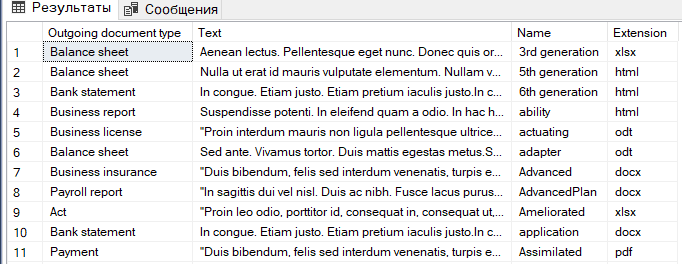
FROM Document

JOIN DocumentFile ON DocumentId=FkDocumentId

JOIN OutgoingDocument ON OutgoingDocument.FkDocumentId=DocumentId

JOIN DocumentType ON FkTypeId=TypeId

GO



– результат запиту

Запит, що виводить тип документа та кількість наявних документів даного типу

SELECT TypeName, COUNT(DocumentId) [Number of documents]

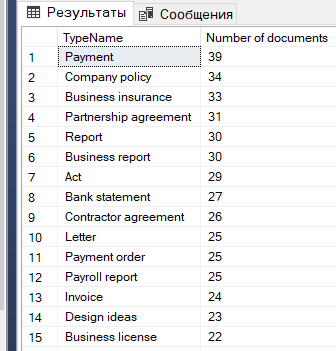
FROM Document

JOIN DocumentType ON TypeId=FkTypeId

GROUP BY TypeName

ORDER BY [Number of documents] DESC

GO



– результат запиту

Запит, що виводить усі документи, надіслані чи отримані протягом останнього місяця

SELECT DocumentId, TypeName, FileName as Name, FileExtension as Extension

FROM Document

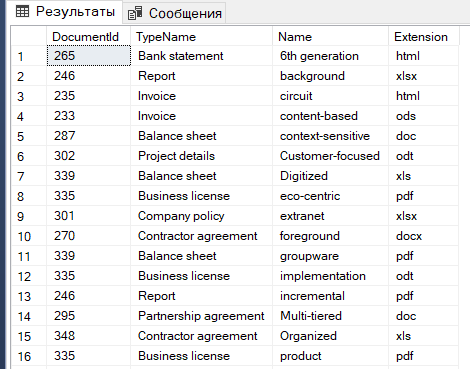
JOIN DocumentFile ON DocumentId=FkDocumentId

JOIN DocumentType ON FkTypeId=TypeId

WHERE DocumentId IN ((SELECT FkDocumentId FROM IncomingDocument WHERE ABS(DATEDIFF(month, GETDATE(), RecevingDate))<3 AND YEAR(RecevingDate)=YEAR(GETDATE()))

UNION (SELECT FkDocumentId FROM OutgoingDocument WHERE ABS(DATEDIFF(month, GETDATE(), DepartureDate))<3 AND YEAR(DepartureDate)=YEAR(GETDATE())))

GO



– результат запиту

Запит, що виводить усі документи, які не мають додаткових файлів

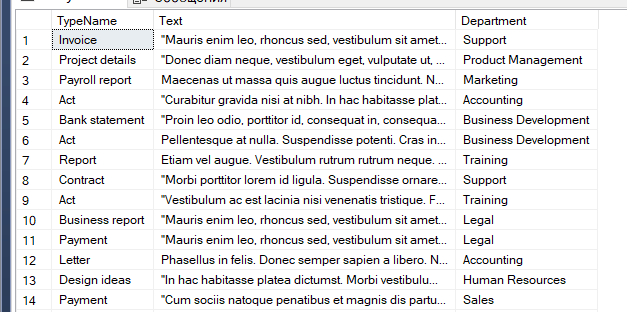
SELECT TypeName, Text, FkDepartmentName Department

FROM Document

JOIN DocumentType ON TypeId=FkTypeId

WHERE DocumentId NOT IN (SELECT FkDocumentId FROM DocumentFile)

GO



– результат запиту

Запит, що виводить вхідний документ та відповідний йому вихідний документ

SELECT [Incoming document text], [Outgoing document text]

FROM DocumentsJournal

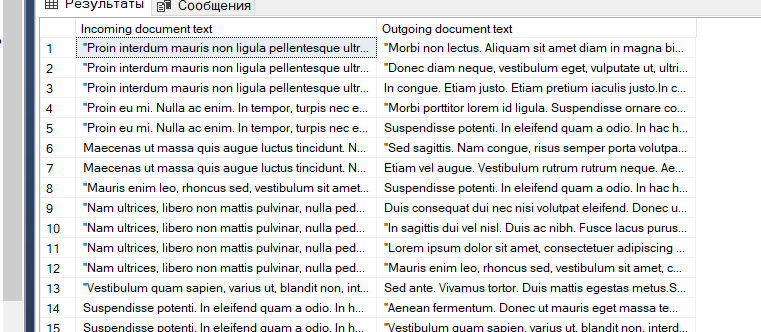
JOIN (SELECT IncomingDocumentId, Text [Incoming document text] FROM IncomingDocument JOIN Document ON FkDocumentId=DocumentId) tab1

ON tab1.IncomingDocumentId=FkIncomingDocumentId

JOIN (SELECT OutgoingDocumentId, Text [Outgoing document text] FROM OutgoingDocument JOIN Document ON FkDocumentId=DocumentId) tab2

ON tab2.OutgoingDocumentId=FkOutgoingDocumentId

GO



– результат запиту

Запит, який виводить назви відділів та типи документів, якими вони займаються

SELECT DepartmentName, TypeName as [Document type]

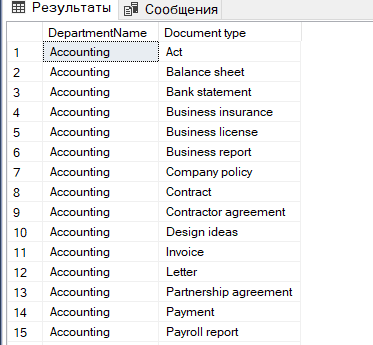
FROM Document

JOIN Department ON FkDepartmentName=DepartmentName

JOIN DocumentType On TypeId=FkTypeId

GROUP BY DepartmentName, TypeName

GO



– результат запиту

Запит, що виводить працівників, які не отримували жодних документів

SELECT Name, Surname, Patronymic, PhoneNumber

FROM Worker

LEFT JOIN IncomingDocument inDoc ON FkRecipientId=WorkerId

GROUP BY Name, Surname, Patronymic, PhoneNumber

HAVING COUNT(inDoc.FkDocumentId)=0

GO



– результат запиту

Запит, що виводить працівників, які не надсилали жодних документів

SELECT Name, Surname, Patronymic, PhoneNumber

FROM Worker

LEFT JOIN OutgoingDocument outDoc ON FkSenderId=WorkerId

GROUP BY Name, Surname, Patronymic, PhoneNumber

HAVING COUNT(outDoc.FkDocumentId)=0

GO



– результат запиту

Запит, що виводить топ-5 відділів, які опрацювали найбільшу кількість документів

SELECT TOP 5 DepartmentName, COUNT(\*) ProcessedDocuments

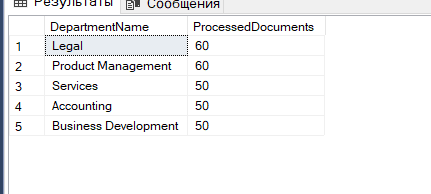
FROM Department

JOIN Document ON FkDepartmentName=DepartmentName

GROUP BY DepartmentName

ORDER BY ProcessedDocuments DESC

GO



– результат запиту

Представлення, що використовувалися у функціях та процедурах:

Створення VIEW для представлення повної інформації про документи

CREATE VIEW DocumentInfo AS

SELECT DocumentId, TypeName, FkDepartmentName as DepartmentName, Text, DocumentFile.FileName, FileExtension

FROM Document JOIN DocumentFile ON DocumentId=FkDocumentId

JOIN DocumentType ON TypeId = FkTypeId

GO

Створення VIEW для представлення інформації про вхідні документи

CREATE VIEW IncomingDocumentInfo AS

SELECT IncomingDocumentId, OrganisationId, OrganisationName, City, Street, Email, WorkerId, Name, Surname, Patronymic, Position, PhoneNumber, TypeName, DepartmentName, Text, RecevingDate, FileName, FileExtension

FROM IncomingDocument

JOIN Organisation On FkOrganisationId=OrganisationId

JOIN Worker ON WorkerId=FkRecipientId

JOIN DocumentInfo ON FkDocumentId=DocumentId

GO

Створення VIEW для представлення інформації про вихідні документи

CREATE VIEW OutgoingDocumentInfo AS

SELECT OutgoingDocumentId, OrganisationId, OrganisationName, City, Street, Email, WorkerId, Name, Surname, Patronymic, Position, PhoneNumber, TypeName, DepartmentName, Text, DepartureDate, FileName, FileExtension

FROM OutgoingDocument

JOIN Organisation On FkOrganisationId=OrganisationId

JOIN Worker ON WorkerId=FkSenderId

JOIN DocumentInfo ON FkDocumentId=DocumentId

GO

Створення VIEW для представлення інформації про робітників департаментів документи

CREATE VIEW DepartmentWorkersInfo AS

SELECT DepartmentName, FkChiefId, WorkerId, Name, Surname, Patronymic, Position, PhoneNumber

FROM DepartmentWorker

JOIN Department ON DepartmentName=FkDepartmentName

JOIN Worker ON WorkerId=FkWorkerId

GO

Деякі з попередніх DML запитів, але вже з використанням створених представлень:

--1((with view usage)

SELECT IncomingFile, OutgoingFile

FROM

DocumentsJournal

JOIN (SELECT TRIM(FileName)+'.'+FileExtension IncomingFile, IncomingDocumentId

FROM IncomingDocument

JOIN DocumentInfo ON FkDocumentId=DocumentId) tab1

ON tab1.IncomingDocumentId=FkIncomingDocumentId

JOIN (SELECT TRIM(FileName)+'.'+FileExtension OutgoingFile, OutgoingDocumentId

FROM OutgoingDocument

JOIN DocumentInfo ON FkDocumentId=DocumentId) tab2

ON tab2.OutgoingDocumentId=FkOutgoingDocumentId

GO

--2(with view usage)

SELECT DocumentId, TypeName, FileName as Name, FileExtension as Extension

FROM DocumentInfo

ORDER BY DocumentId

GO

--7(with view usage)

SELECT DocumentId, TypeName, Text, FileName as Name, FileExtension as Extension

FROM DocumentInfo

WHERE DocumentId NOT IN

((SELECT DocumentId FROM Document JOIN IncomingDocument ON DocumentId=FkDocumentId)

UNION

(SELECT DocumentId FROM Document JOIN OutgoingDocument ON DocumentId=FkDocumentId))

ORDER BY DocumentId

GO

--9(with view usage)

SELECT DepartmentWorkersInfo.Name, DepartmentWorkersInfo.Surname, DepartmentWorkersInfo.Position, DepartmentName, chief.Name [Chief name], chief.Surname [Chief surname], chief.PhoneNumber [Chief phone number]

FROM DepartmentWorkerSInfo

JOIN Worker chief ON FkChiefId=chief.WorkerId

GO

--11(with view usage)

SELECT TypeName [Incoming document type], Text, FileName as Name, FileExtension as Extension

FROM DocumentInfo

JOIN IncomingDocument ON IncomingDocument.FkDocumentId=DocumentId

GO

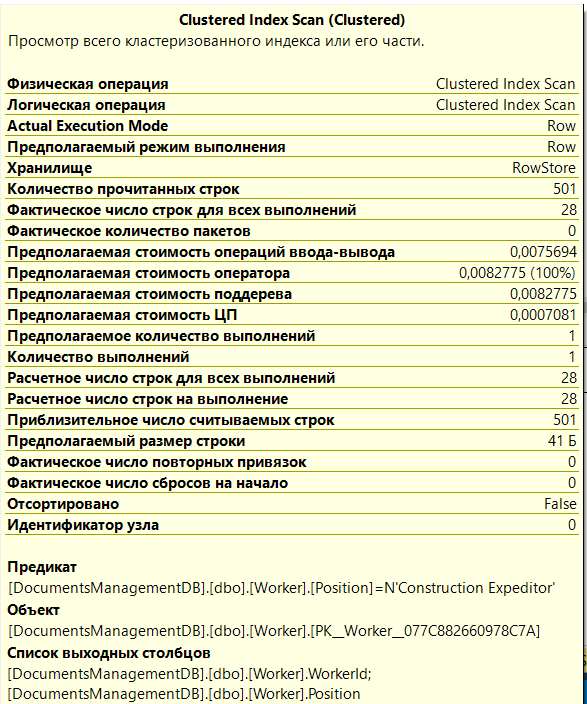
## Приклад використання індексів для пришвидшення запитів

Розглянемо простий sql запит:

SELECT WorkerId, Position FROM Worker

WHERE Position='Construction Expeditor'

Маємо наступні результати без додавання індексу:

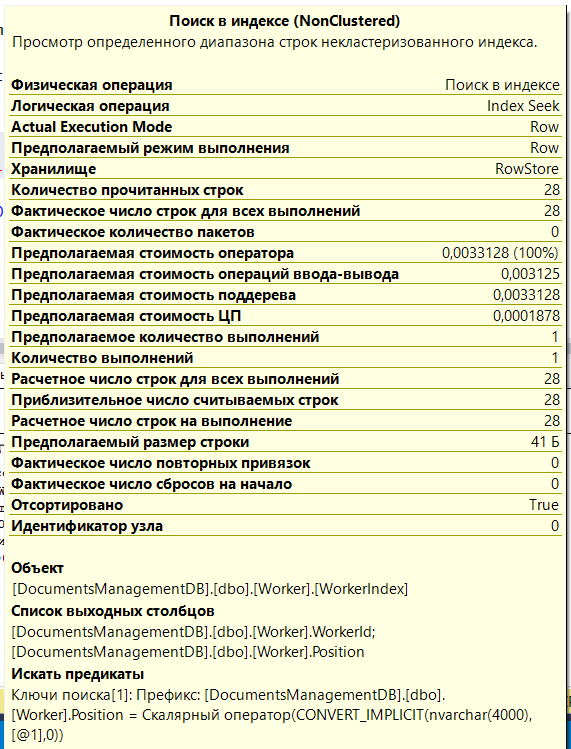


– Результати виконання запиту без індексу

Тепер створимо індекс:

CREATE INDEX WorkerIndex ON Worker(Position)

Повторно виконаємо запит:



– Повторне виконання запиту після додавання індексу

Отже, можна побачити, що замість 501 рядка було проглянуто лише 28, що є значним покращенням та показує, як можна пришвидшити виконання запитів, використовуючи індекси.

Висновок

Отже, бази даних є одним з основних засобів обробки та збереження даних. Вони пропонують зручний та гнучкий спосіб збереження та маніпулювання інформацією.

Отже, у результаті виконання даної роботи було створено базу даних для підтримки діловодства організації. Спочатку було проведено дослідження предметної області, далі було виділено основні сутності, функціонал та користувачів, створено ER – діаграму. Далі було реалізовано безпосередньо саму базу даних в СУБД MS SQL Server шляхом створення таблиць, визначення первинних і вторинних ключів та встановленням обмежень на значення стовпця. Створену базу даних я наповнив наборами даних, згенерованими з використанням веб-застосунку Mockaroo. Було створено користувачів для взаємодії з цією базою даних, створено функції та процедури, представлення та тригери. Провів порівняння роботи запитів без і з використанням індексів і впевнився у тому, що це один з способів, який дозволяє значно пришвидшити виконання запиту.

Cписок літератури

1. Лекції викладача Ліщук К. І. з дисципліни «Бази даних»
2. <https://metanit.com/sql/>
3. <https://docs.microsoft.com/en-us/sql/?view=sql-server-ver15>
4. <https://www.w3schools.com/sql/>
5. https://www.w3schools.com/sql/