

Міністерство освіти і науки України  
Національний університет “Львівська політехніка”  
Інститут прикладної математики та фундаментальних наук  
Кафедра прикладної математики

**Курсова робота**  
з курсу “Робота з великими базами даних”  
на тему: “Розробка та аналіз надвеликої бази даних з використанням технологій  
Microsoft SQL Server ”

Варіант 6

**Виконала:**

студентка групи ПМ-41

Гришечко Марта

**Прийняв:**

Любінський Богдан Богданович

---

(дата)                    (підпис викладача)

Львів 2026

Курсова робота присвячена розробці та аналізу надвеликої бази даних для предметної області поліклініки з використанням технологій Microsoft SQL Server. У роботі розглянуто повний цикл створення інформаційно-аналітичної системи, починаючи з аналізу предметної області та проєктування бази даних, і завершуючи побудовою OLAP-куба та формуванням аналітичних звітів.

У процесі виконання курсової роботи було проведено детальний аналіз діяльності поліклініки, визначено основні бізнес-процеси, сутності та їх взаємозв'язки. На основі цього спроектовано реляційну базу даних з дотриманням принципів нормалізації та цілісності даних. Для забезпечення ефективного аналізу великих обсягів інформації було розроблено сховище даних (Data Warehouse) за схемою зірки та реалізовано ETL-процеси з використанням SQL Server Integration Services (SSIS).

У роботі здійснено побудову багатовимірного OLAP-куба за допомогою SQL Server Analysis Services (SSAS), створено виміри, міри, ієархії та обчислювані показники для аналізу медичних даних. На основі OLAP-куба розроблено аналітичні звіти в SQL Server Reporting Services (SSRS), зокрема: список лікарів та їх пацієнтів, історію хвороби пацієнта, аналіз завантаженості лікарів та статистику захворювань. Реалізовано параметризацію звітів, drill-down, умовне форматування та інтерактивні елементи.

Результати курсової роботи демонструють можливості використання сучасних технологій Microsoft SQL Server для обробки та аналізу надвеликої кількості даних у медичній сфері, а також підтверджують доцільність впровадження OLAP-технологій для підтримки прийняття управлінських рішень.

## Зміст

<b>ВСТУП.....</b>	<b>5</b>
<b>РОЗДІЛ 1 .....</b>	<b>7</b>
1.1 Загальна характеристика предметної області «Поліклініка» .....	7
1.2 Аналіз основних бізнес-процесів поліклініки .....	8
1.3 Функціональні вимоги до інформаційної системи поліклініки .....	9
1.4 Бізнес-правила та обмеження предметної області .....	9
1.5 Обґрунтування використання надвеликих баз даних .....	10
Висновки до розділу .....	10
<b>РОЗДІЛ 2 .....</b>	<b>11</b>
2.1 Аналіз вимог до бази даних .....	11
2.2 Концептуальне проєктування бази даних .....	11
2.3 Логічне проєктування бази даних.....	12
2.4 Фізичне проєктування бази даних .....	13
2.5 Проєктування сховища даних (Data Warehouse) .....	17
2.6 Таблиці фактів .....	17
2.7 Таблиці вимірів .....	18
2.8 Забезпечення цілісності та узгодженості даних .....	19
Висновки до розділу .....	19
<b>РОЗДІЛ 3 .....</b>	<b>20</b>
3.1 Загальна характеристика ETL-процесів .....	20
3.2 Створення проекту SSIS .....	20
3.3 Налаштування з'єднань з джерелами даних.....	21
3.4 Проєктування сховища даних .....	21
3.5 Реалізація процесу Extract (витягування даних).....	21
3.6 Реалізація процесу Transform (трансформація даних) .....	22
3.7 Реалізація процесу Load (завантаження даних).....	24
Висновки до розділу 3 .....	25
<b>РОЗДІЛ 4 .....</b>	<b>26</b>
4.1 Створення проекту SQL Server Analysis Services.....	26
4.2 Проектування Data Source View (DSV) .....	26
4.3 Створення вимірів (Dimensions).....	27
4.5 Реалізація MDX-обчислень та KPI .....	32

4.6 Оптимізація та розгортання куба .....	32
4.7 Використання OLAP-куба в аналітичних звітах .....	33
<b>РОЗДІЛ 5 .....</b>	<b>34</b>
5.1 Загальна характеристика аналітичних звітів .....	34
5.2 Налаштування проекту SQL Server Reporting Services.....	34
5.3 Реалізація обов'язкових аналітичних звітів .....	34
5.4 Параметризація та інтерактивність звітів .....	39
5.5 Експорт та розгортання звітів .....	39
<b>ВИСНОВКИ .....</b>	<b>41</b>
<b>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....</b>	<b>43</b>

## **ВСТУП**

В умовах стрімкого розвитку інформаційних технологій та цифровізації медичної галузі особливого значення набуває ефективне управління великими обсягами медичних даних. Сучасні медичні заклади, зокрема поліклініки, щоденно обробляють значні масиви інформації, пов'язані з обліком пацієнтів, історіями хвороб, візитами до лікарів, призначенням лікування, використанням лікарських засобів та аналізом результатів лікування. Обсяг таких даних постійно зростає, що зумовлює необхідність використання спеціалізованих технологій для їх зберігання, обробки та аналізу.

Традиційні реляційні бази даних, які використовуються для операційної діяльності медичних установ, не завжди здатні забезпечити високопродуктивний аналітичний аналіз на великих обсягах даних. Саме тому актуальним є застосування концепцій надвеликих баз даних, сховищ даних (Data Warehouse) та багатовимірного аналізу (OLAP). Використання таких технологій дозволяє не лише зберігати історичні дані за тривалий період часу, але й здійснювати глибокий аналіз діяльності медичного закладу, виявляти тенденції захворюваності, оцінювати ефективність лікування та оптимізувати роботу лікарів.

У даній курсовій роботі розглядається предметна область “Поліклініка”, яка є типовим прикладом системи з надвеликим обсягом даних. Для цієї області характерна наявність десятків тисяч пацієнтів, сотень тисяч записів історій хвороб та багаторічної медичної статистики. Обробка таких даних потребує комплексного підходу, що включає проектування реляційної бази даних, реалізацію ETL-процесів, побудову сховища даних, створення OLAP-куба та розробку аналітичних звітів.

**Метою курсової роботи** є розробка повнофункціональної системи керування надвеликою базою даних для предметної області «Поліклініка» з використанням технологій Microsoft SQL Server, зокрема SQL Server Integration Services (SSIS), SQL Server Analysis Services (SSAS) та SQL Server Reporting Services (SSRS).

Для досягнення поставленої мети у курсовій роботі необхідно вирішити такі завдання:

- провести детальний аналіз предметної області поліклініки;
- визначити основні бізнес-процеси та інформаційні потоки;

- спроектувати концептуальну, логічну та фізичну модель бази даних;
- реалізувати генерацію та наповнення бази великими обсягами реалістичних даних;
- розробити ETL-процеси для завантаження даних у сховище;
- побудувати багатовимірний OLAP-куб;
- створити аналітичні звіти для підтримки прийняття управлінських рішень.

**Об'єктом дослідження** є інформаційна система медичного закладу (поліклініки).

**Предметом дослідження** є методи та засоби проектування, зберігання та аналітичної обробки надвеликих обсягів медичних даних.

Практична цінність роботи полягає у можливості використання розробленої системи для аналізу діяльності поліклініки, оцінки ефективності лікування та формування управлінської звітності.

# РОЗДІЛ 1

## АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

### 1.1 Загальна характеристика предметної області «Поліклініка»

Поліклініка є одним із ключових елементів системи охорони здоров'я, що забезпечує надання амбулаторної медичної допомоги населенню. Основною функцією поліклініки є проведення профілактичних оглядів, діагностики захворювань, лікування пацієнтів, а також ведення медичної документації. У сучасних умовах діяльність поліклінік супроводжується обробкою значних обсягів інформації, що зумовлює необхідність використання ефективних інформаційних систем і надвеликих баз даних.

Інформаційна система поліклініки акумулює дані про пацієнтів, лікарів, діагнози, хвороби, призначене лікування, лікарські засоби, результати візитів та історії хвороб. З огляду на багаторічне зберігання медичних даних, кількість записів у таких системах може досягати сотень тисяч або навіть мільйонів, що робить актуальним застосування технологій надвеликих баз даних (Very Large Databases, VLDB).

Особливістю предметної області «Поліклініка» є складна структура взаємозв'язків між даними. Один пацієнт може мати багато візитів до різних лікарів, кожен візит супроводжується встановленням одного або кількох діагнозів, призначенням лікування та фіксацією результатів. Крім того, лікарі спеціалізуються на різних напрямках медицини, що впливає на типи діагнозів і методи лікування.

У контексті аналітики керівництво поліклініки зацікавлене в отриманні узагальненої інформації, зокрема:

- статистики захворювань за періодами часу;
- аналізу ефективності лікування;
- оцінки завантаженості лікарів;
- динаміки відвідувань пацієнтів;
- виявлення сезонних тенденцій захворювань.

Реалізація таких аналітичних запитів у транзакційній базі даних є малоєфективною, тому виникає потреба у створенні сховища даних (Data Warehouse) та багатовимірного OLAP-куба.

Таким чином, предметна область «Поліклініка» є типовим прикладом системи, що потребує застосування сучасних технологій управління надвеликими базами даних для забезпечення високої продуктивності, масштабованості та аналітичних можливостей.

## 1.2 Аналіз основних бізнес-процесів поліклініки

Функціонування поліклініки базується на низці взаємопов'язаних бізнес-процесів, кожен з яких генерує та використовує значні обсяги даних. Основні бізнес-процеси поліклініки наведено нижче.

### 1.2.1 Реєстрація та обслуговування пацієнтів

Процес починається з реєстрації пацієнта в інформаційній системі. Під час реєстрації фіксуються персональні дані пацієнта: ім'я, прізвище, дата народження, стать, контактна інформація. Пацієнту присвоюється унікальний ідентифікатор, який використовується в усіх подальших операціях.

### 1.2.2 Запис на прийом до лікаря

Пацієнт може бути записаний на прийом до конкретного лікаря або спеціаліста. При цьому враховується спеціалізація лікаря, графік роботи та доступні часові слоти. Даний процес формує інформацію про заплановані візити.

### 1.2.3 Проведення лікарського прийому

Під час візиту лікар проводить огляд пацієнта, встановлює діагноз або кілька діагнозів, призначає лікування та лікарські засоби. Всі ці дані фіксуються в системі та формують запис історії хвороби.

### 1.2.4 Ведення історії хвороби

Історія хвороби є ключовим бізнес-об'єктом предметної області. Вона містить інформацію про всі візити пацієнта, діагнози, призначене лікування та результати лікування. Історія хвороби зберігається протягом тривалого часу (не менше 10 років).

### 1.2.5 Аналітична обробка даних

На основі накопичених даних здійснюється аналітична обробка, що включає формування звітів, побудову показників ефективності, аналіз тенденцій та прийняття управлінських рішень.

### 1.3 Функціональні вимоги до інформаційної системи поліклініки

Інформаційна система поліклініки повинна відповідати таким функціональним вимогам:

- зберігання великих обсягів медичних даних з можливістю масштабування;
- підтримка історичності даних без втрати попередніх значень;
- забезпечення цілісності та узгодженості інформації;
- швидкий пошук пацієнтів, лікарів та історій хвороб;
- підтримка аналітичних запитів та багатовимірного аналізу;
- формування параметризованих звітів;
- можливість аналізу даних у часовому розрізі;
- підтримка OLAP-операцій (slice, dice, drill-down, roll-up).

З метою виконання зазначених вимог у роботі доцільно використовувати архітектуру «Data Warehouse + OLAP-куб», що дозволяє відокремити аналітичні навантаження від транзакційної обробки даних.

### 1.4 Бізнес-правила та обмеження предметної області

Для коректного функціонування системи необхідно дотримуватись низки бізнес-правил:

1. Один пацієнт може мати необмежену кількість візитів.
2. Кожен візит пов'язаний лише з одним лікарем.
3. Один візит може містити кілька діагнозів.
4. Діагноз не може існувати без відповідного візиту.
5. Історія хвороби пацієнта не підлягає видаленню.
6. Дані повинні зберігатися щонайменше 10 років.
7. Лікар може обслуговувати багато пацієнтів.
8. Кожен лікар має унікальний ідентифікатор.

Дотримання цих правил забезпечує логічну цілісність даних та коректність аналітичних результатів.

## 1.5 Обґрунтування використання надвеликих баз даних

З огляду на вимоги курсової роботи, система повинна містити щонайменше 50 000 пацієнтів та понад 500 000 записів історій хвороб за період 10 років. Такий обсяг даних неможливо ефективно обробляти без застосування спеціалізованих технологій надвеликих баз даних.

Використання Microsoft SQL Server у поєднанні з SSIS, SSAS та SSRS дозволяє:

- реалізувати ефективні ETL-процеси;
- побудувати масштабоване сховище даних;
- створити багатовимірний OLAP-куб;
- забезпечити високопродуктивну аналітику та візуалізацію даних.

## Висновки до розділу

У даному розділі було проаналізовано предметну область «Поліклініка», визначено основні бізнес-процеси, функціональні вимоги та бізнес-правила. Обґрунтовано доцільність використання технологій надвеликих баз даних для зберігання та аналізу медичної інформації. Отримані результати є основою для подальшого проектування бази даних, реалізації ETL-процесів та побудови OLAP-куба.

## РОЗДІЛ 2

### ПРОЄКТУВАННЯ БАЗИ ДАНИХ ПОЛІКЛІНІКИ

#### 2.1 Аналіз вимог до бази даних

Проєктування бази даних для інформаційної системи поліклініки ґрунтуються на результатах аналізу предметної області, описаної в першому розділі. Основною метою бази даних є забезпечення надійного зберігання, обробки та аналізу великих обсягів медичної інформації, пов'язаної з діяльністю поліклініки за тривалий період часу.

База даних повинна забезпечувати:

- зберігання інформації про пацієнтів (персональні дані, демографічні характеристики);
- облік лікарів та їх спеціалізацій;
- збереження даних про візити пацієнтів;
- фіксацію діагнозів, результатів лікування та призначених медикаментів;
- підтримку історичності даних;
- можливість подальшого аналітичного опрацювання в OLAP-кубі.

Оскільки система орієнтована на роботу з **надвеликими обсягами даних** (понад 50 000 пацієнтів і 500 000 записів історій хвороб), при проєктуванні бази даних особливу увагу приділено питанням масштабованості, продуктивності та цілісності даних.

#### 2.2 Концептуальне проєктування бази даних

На етапі концептуального проєктування було визначено основні сутності предметної області та зв'язки між ними.

##### **Основні сутності:**

- Пацієнт (Patient)
- Лікар (Doctor)
- Візит (Medical Visit)
- Діагноз (Diagnosis)
- Хвороба (Disease)

- Ліки (Medicine)
- Результат лікування (Treatment Result)

Між сущностями визначено такі зв'язки:

- пацієнт може мати багато візитів;
- кожен візит пов'язаний з одним лікарем;
- у межах візиту встановлюється діагноз;
- один діагноз може зустрічатися у багатьох візитах;
- для кожного візиту фіксується результат лікування.

## 2.3 Логічне проєктування бази даних

На етапі логічного проєктування виконано нормалізацію бази даних до **третньої нормальної форми (ЗНФ)**. Це дозволило:

- усунути дублювання даних;
- забезпечити цілісність інформації;
- спростити підтримку та оновлення даних.

Дляожної сущності визначено:

- первинні ключі (Primary Key);
- зовнішні ключі (Foreign Key);
- типи даних атрибутів;
- обмеження цілісності.

Приклад:

- у таблиці **Patients** первинним ключем є PatientID;
- у таблиці **MedicalVisits** містяться зовнішні ключі PatientID, DoctorID, DateID.

Для оптимізації доступу до даних були передбачені індекси на найбільш використовуваних полях, зокрема на ключах та датах візитів.

## 2.4 Фізичне проєктування бази даних

Фізичне проєктування бази даних виконано в середовищі **Microsoft SQL Server**. На цьому етапі були створені таблиці, визначені типи даних, реалізовані обмеження цілісності та індекси.



*Рис. 2.1. Структура бази даних поліклініки в SQL Server*

### 2.4.1 Таблиця Patients (Пацієнти)

Таблиця Patients призначена для зберігання персональної інформації про пацієнтів поліклініки.

#### Атрибути таблиці:

- PatientID – унікальний ідентифікатор пацієнта
- FirstName – ім’я пацієнта
- LastName – прізвище пацієнта
- BirthDate – дата народження
- Gender – стать пацієнта
- RegistrationDate – дата реєстрації в поліклініці

#### 2.4.2 Таблиця Doctors (Лікарі)

Таблиця Doctors містить інформацію про лікарів, які працюють у поліклініці.

##### **Атрибути таблиці:**

- DoctorID – унікальний ідентифікатор лікаря
- FirstName – ім'я лікаря
- LastName – прізвище лікаря
- SpecializationID – ідентифікатор спеціалізації
- HireDate – дата прийому на роботу

#### 2.4.3 Таблиця DoctorSpecializations (Спеціалізації лікарів)

Таблиця використовується для зберігання інформації про медичні спеціалізації лікарів.

##### **Атрибути таблиці:**

- SpecializationID – унікальний ідентифікатор спеціалізації
- SpecializationName – назва спеціалізації

#### 2.4.4 Таблиця Diseases (Захворювання)

Таблиця Diseases зберігає довідникову інформацію про захворювання.

##### **Атрибути таблиці:**

- DiseaseID – унікальний ідентифікатор захворювання
- DiseaseName – назва захворювання
- ICD10Code – код захворювання за міжнародною класифікацією ICD-10

#### 2.4.5 Таблиця Diagnoses (Діагнози)

Таблиця Diagnoses містить інформацію про встановлені діагнози пацієнтів.

##### **Атрибути таблиці:**

- DiagnosisID – унікальний ідентифікатор діагнозу

- DiseaseID – ідентифікатор захворювання
- DiagnosisDescription – опис діагнозу

#### 2.4.6 Таблиця MedicalHistory (Історія хвороб)

Таблиця MedicalHistory є центральною таблицею бази даних і фіксує всі медичні візити пацієнтів.

#### Атрибути таблиці:

- HistoryID – ідентифікатор запису історії хвороби
- PatientID – ідентифікатор пацієнта
- DoctorID – ідентифікатор лікаря
- DiagnosisID – ідентифікатор діагнозу
- ResultID – ідентифікатор результату лікування
- VisitDate – дата візиту

#### 2.4.7 Таблиця TreatmentResults (Результати лікування)

Таблиця використовується для зберігання можливих результатів лікування.

#### Атрибути таблиці:

- ResultID – унікальний ідентифікатор результату
- ResultName – назва результату (вилікувався, без змін, погіршення тощо)

#### 2.4.8 Таблиця Medicines (Лікарські засоби)

Таблиця Medicines містить інформацію про медикаменти, що використовуються в лікуванні.

#### Атрибути таблиці:

- MedicineID – ідентифікатор лікарського засобу
- MedicineName – назва препарату
- Manufacturer – виробник

#### 2.4.9 Таблиця Contraindications (Протипоказання)

Таблиця Contraindications зберігає інформацію про протипоказання до застосування лікарських засобів.

##### **Атрибути таблиці:**

- ContraindicationID – ідентифікатор протипоказання
- MedicineID – ідентифікатор лікарського засобу
- Description – опис протипоказання

#### 2.4.10 Таблиця DiagnosisMedicines (Зв'язок діагнозів і ліків)

Таблиця реалізує зв'язок «багато-до-багатьох» між діагнозами та лікарськими засобами.

##### **Атрибути таблиці:**

- DiagnosisID – ідентифікатор діагнозу
- MedicineID – ідентифікатор лікарського засобу

#### 2.4.11 Реалізація обмежень цілісності

Для забезпечення коректності даних реалізовано:

- обмеження PRIMARY KEY;
- обмеження FOREIGN KEY;
- обмеження NOT NULL;
- перевірки CHECK для допустимих значень.

#### 2.4.12 Індекси та оптимізація

З метою підвищення продуктивності запитів були створені індекси для:

- ключових полів;
- дат візитів;
- ідентифікаторів пацієнтів та лікарів.

## 2.5 Проектування сховища даних (Data Warehouse)

Для забезпечення ефективного аналітичного опрацювання великих обсягів медичних даних було прийнято рішення про створення **сховища даних (Data Warehouse)**. Сховище даних призначене для зберігання історичних, узгоджених та агрегованих даних, що використовуються для багатовимірного аналізу та побудови аналітичних звітів.

Архітектура сховища даних базується на **схемі зірки (Star Schema)**, що є оптимальною для OLAP-аналізу завдяки простоті структури та високій швидкодії запитів.

### Основні компоненти сховища даних:

- Таблиці фактів (Fact Tables)
- Таблиці вимірів (Dimension Tables)

## 2.6 Таблиці фактів

У межах даного проекту було реалізовано дві основні таблиці фактів:

### 2.6.1 Fact Medical Visit

Таблиця **DW\_FactMedicalVisit** зберігає інформацію про всі візити пацієнтів до лікарів і є центральною таблицею сховища даних.

Основні показники таблиці фактів:

- кількість візитів;
- результат лікування;
- тривалість лікування (за наявності);
- інші числові показники для аналізу.

Таблиця містить зовнішні ключі до таблиць вимірів:

- PatientKey;
- DoctorKey;
- DiagnosisKey;
- DateKey;
- ResultKey.

## 2.6.2 Fact Visit Medicine

Додаткова таблиця фактів **DW\_FactVisitMedicine** використовується для аналізу призначених лікарських засобів у межах кожного візиту.

Вона дозволяє виконувати:

- аналіз частоти призначення лікарств;
- аналіз взаємозв'язку між діагнозами та медикаментами;
- оцінку ефективності лікування.

## 2.7 Таблиці вимірів

Для багатовимірного аналізу було створено набір таблиць вимірів, які описують різні аспекти предметної області.

### 2.7.1 Вимір «Пацієнт»

Таблиця **DW\_DimPatient** містить інформацію про пацієнтів:

- унікальний сурогатний ключ (PatientKey);
- ім'я та прізвище;
- стать;
- дата народження.

Для забезпечення коректного аналізу історичних даних у вимірі реалізовано підхід **Slowly Changing Dimension (SCD)**.

### 2.7.2 Вимір «Лікар»

Таблиця **DW\_DimDoctor** містить дані про лікарів:

- ім'я та прізвище;
- спеціалізацію;
- дату найму на роботу.

Даний вимір використовується для аналізу завантаженості лікарів та ефективності лікування.

### 2.7.3 Вимір «Діагноз»

Вимір **DW\_DimDiagnosis** дозволяє аналізувати статистику захворювань, їх поширеність та динаміку в часі.

Основні атрибути:

- назва захворювання;
- опис діагнозу;
- категорія хвороби.

### 2.7.4 Часовий вимір

Часовий вимір **DW\_DimDate** є ключовим елементом аналітичної системи. Він забезпечує можливість аналізу даних за роками, кварталами, місяцями та днями.

## 2.8 Забезпечення цілісності та узгодженості даних

Для забезпечення цілісності даних між таблицями фактів і вимірів використовуються зовнішні ключі та механізми перевірки відповідності ключових значень. Це дозволяє уникнути ситуацій, коли у таблиці фактів з'являються посилання на неіснуючі записи у вимірах.

Крім того, реалізовано перевірки коректності завантаження даних, що особливо важливо при роботі з великими обсягами інформації.

## Висновки до розділу

У другому розділі курсової роботи було виконано детальне проєктування бази даних та сховища даних для предметної області поліклініки. Розроблено концептуальну, логічну та фізичну моделі бази даних, визначено структуру таблиць фактів і вимірів, а також обґрунтовано вибір архітектури сховища даних.

Створена структура є основою для подальшої реалізації ETL-процесів, побудови OLAP-куба та розробки аналітичних звітів, що розглядаються у наступних розділах курсової роботи.

## РОЗДІЛ 3

### РЕАЛІЗАЦІЯ ETL-ПРОЦЕСІВ У SQL SERVER INTEGRATION SERVICES

#### 3.1 Загальна характеристика ETL-процесів

ETL-процеси (Extract, Transform, Load) є ключовим етапом у побудові надвеликої бази даних та сховища даних (Data Warehouse), оскільки саме на цьому етапі здійснюється підготовка, очищення, трансформація та завантаження даних з операційних джерел у аналітичну систему. Для реалізації ETL-процесів у даній курсовій роботі було використано інструмент **SQL Server Integration Services (SSIS)**, який входить до складу Microsoft SQL Server та надає потужні можливості для обробки великих обсягів даних.

У предметній області «Поліклініка» джерелом даних виступає транзакційна база даних, що містить інформацію про пацієнтів, лікарів, діагнози, історії хвороб, медикаменти та результати лікування. Основною метою ETL-процесів є перенесення цих даних у сховище даних з урахуванням вимог багатовимірного аналізу, збереження історичності та забезпечення високої продуктивності запитів.

ETL-процеси в межах даної роботи були спроектовані таким чином, щоб:

- забезпечити цілісність та коректність даних;
- реалізувати очищення та стандартизацію інформації;
- підтримувати інкрементальне завантаження;
- підготувати дані для подальшого використання в OLAP-кубі та звітах SSRS.

#### 3.2 Створення проекту SSIS

Для реалізації ETL-процесів було створено окремий проект типу **Integration Services Project** у середовищі **SQL Server Data Tools (SSDT)**. На початковому етапі було виконано налаштування середовища розробки, а також перевірено наявність необхідних компонентів SQL Server Integration Services.

У межах проекту було створено кілька пакетів (.dtsx), кожен з яких відповідає за завантаження окремих сутностей або груп пов'язаних таблиць. Такий підхід дозволяє підвищити керованість ETL-процесів, спростити налагодження та забезпечити повторне використання компонентів.

### 3.3 Налаштування з'єднань з джерелами даних

На наступному етапі було створено та налаштовано з'єднання з базами даних. У проекті використовуються два основні типи з'єднань:

- з'єднання з операційною (OLTP) базою даних поліклініки;
- з'єднання зі сховищем даних (Data Warehouse).

Для цього були використані компоненти **OLE DB Connection Manager**, де вказувалися параметри сервера, назва бази даних та облікові дані доступу. З метою підвищення безпеки та зручності адміністрування всі з'єднання були винесені в загальні (Shared) компоненти.

### 3.4 Проектування сховища даних

Сховище даних було побудовано з використанням **зіркоподібної схеми (Star Schema)**, що є найбільш ефективною для OLAP-аналізу. У центрі схеми розміщені таблиці фактів, які містять кількісні показники, а навколо них — таблиці вимірів, що описують контекст даних.

У межах даної курсової роботи було реалізовано:

- таблицю фактів історії хвороб (FactMedicalHistory);
- виміри пацієнтів, лікарів, діагнозів, захворювань, часу та результатів лікування.

Такий підхід дозволяє виконувати аналітичні запити за різними зрізами, наприклад, аналізувати кількість відвідувань за період, ефективність лікування конкретного лікаря або статистику захворювань.

### 3.5 Реалізація процесу Extract (витягування даних)

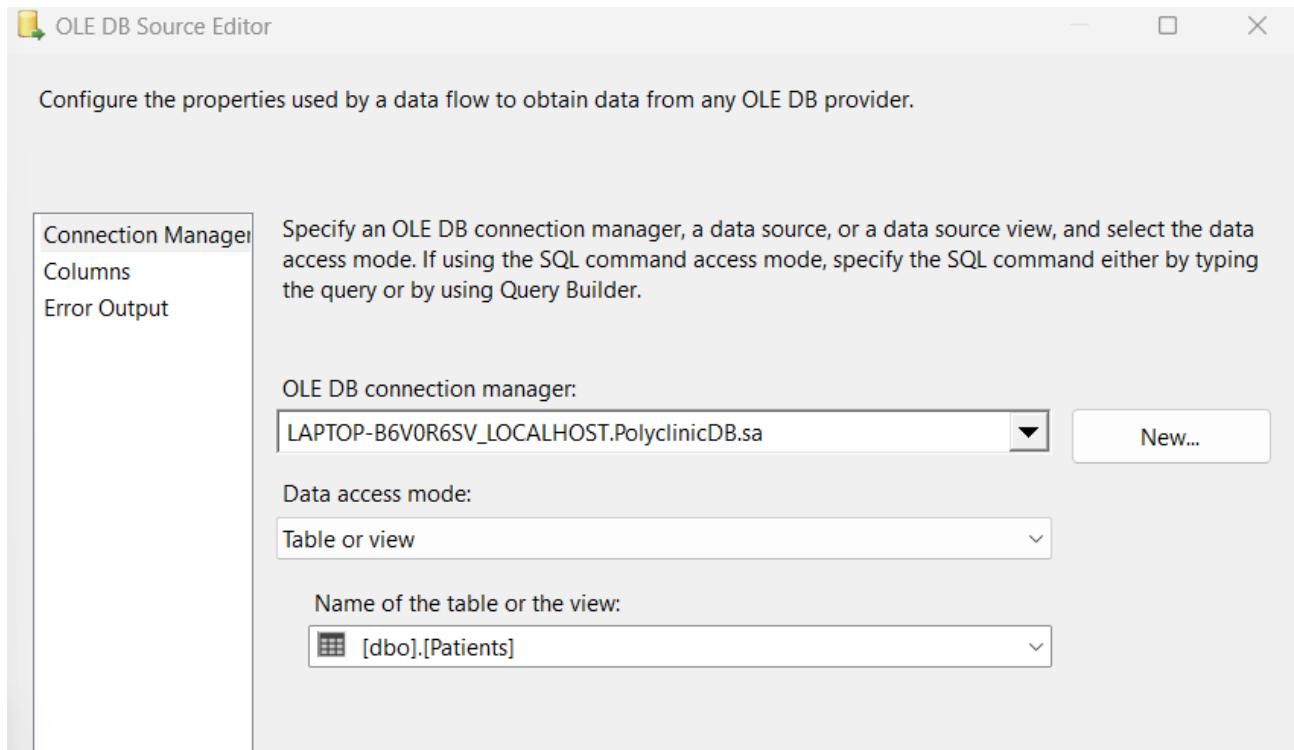
Етап Extract полягає у витягуванні даних з операційної бази даних. Для цього в кожному пакеті SSIS було створено компонент **Data Flow Task**, у межах якого використовується **OLE DB Source**. Цей компонент дозволяє виконувати SQL-запити до джерела даних та отримувати необхідні набори записів.

На даному етапі було реалізовано витягування даних з таких таблиць:

- Patients;
- Doctors;
- MedicalHistory;

- Diagnoses;
- Diseases;
- Medicines;
- TreatmentResults.

Для оптимізації продуктивності використовувалися лише необхідні поля, а також фільтрація даних за датами у випадку інкрементального завантаження.



*Рис. 3.1. Налаштування OLE DB Source у Data Flow Task*

### 3.6 Реалізація процесу Transform (трансформація даних)

Етап Transform є найбільш складним та важливим у ETL-процесах, оскільки саме на цьому етапі дані приводяться до вигляду, придатного для аналітичної обробки. У межах даної роботи було реалізовано кілька типів трансформацій.

#### 3.6.1 Очищення даних (Data Cleansing)

Для очищення даних використовувалися стандартні компоненти SSIS, зокрема:

- видалення або заміна NULL-значень;
- перевірка коректності дат;
- усунення дублікатів записів.

Наприклад, у таблиці пацієнтів перевірялися дати народження, а у таблиці історії хвороб — коректність дат візитів.

### 3.6.2 Конвертація типів даних (Data Conversion)

Оскільки типи даних в OLTP-системі можуть не відповідати вимогам сховища даних, було застосовано компонент **Data Conversion**, який забезпечує приведення типів (наприклад, з varchar у date або int).

### 3.6.3 Похідні колонки (Derived Column)

Компонент **Derived Column** використовувався для створення додаткових атрибутів, зокрема:

- обчислення віку пацієнта;
- формування повного імені лікаря;
- створення бізнес-ключів.

### 3.6.4 Lookup-трансформації

Lookup-трансформації використовувалися для зв'язування фактів з вимірами. Наприклад, при завантаженні таблиці фактів здійснювався пошук відповідних ключів пацієнтів, лікарів та діагнозів у таблицях вимірів.

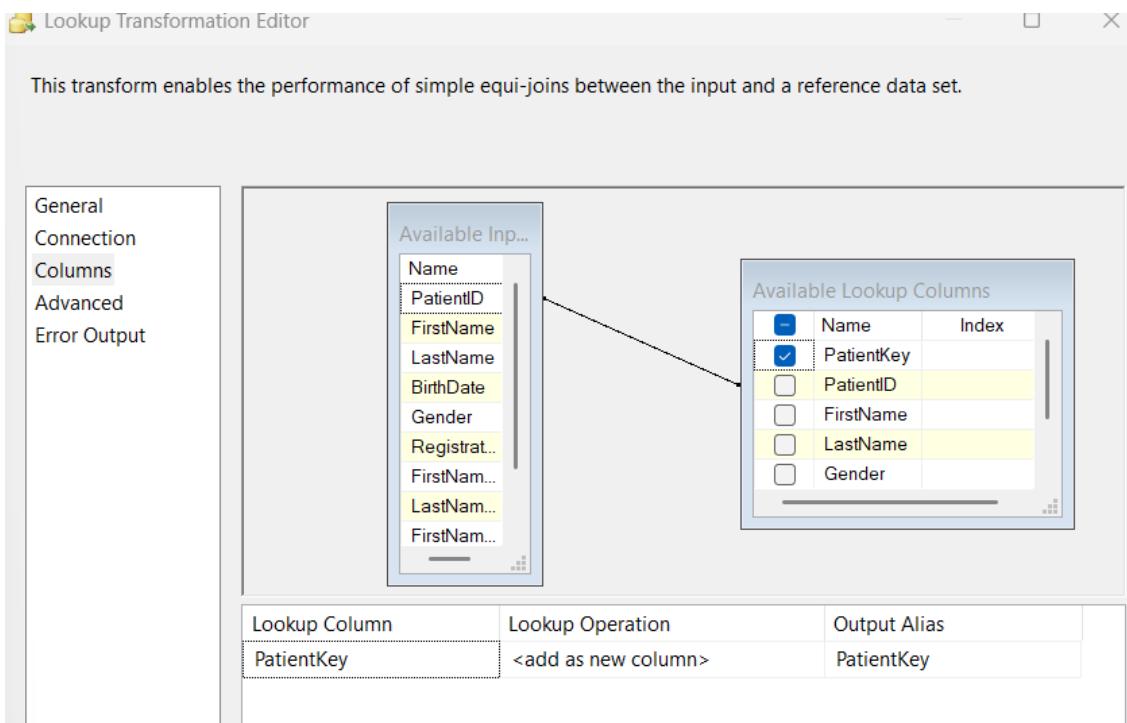


Рис. 3.2. Використання Lookup-трансформації

### 3.7 Реалізація процесу Load (завантаження даних)

На етапі Load трансформовані дані завантажуються у таблиці сховища даних за допомогою компонента **OLE DB Destination**. Для підвищення продуктивності використовувався режим швидкого завантаження (Fast Load).

Завантаження даних було реалізовано як у повному режимі (initial load), так і в інкрементальному режимі, що дозволяє додавати лише нові або змінені записи без повторного завантаження всього обсягу даних.

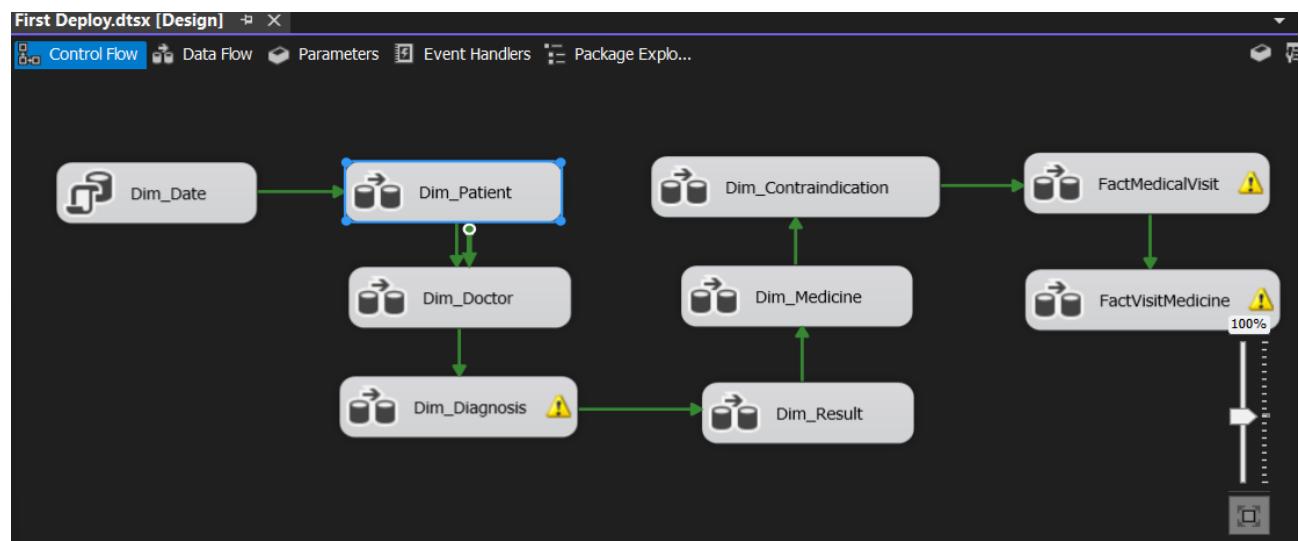


Рис. 3.3. Control flow

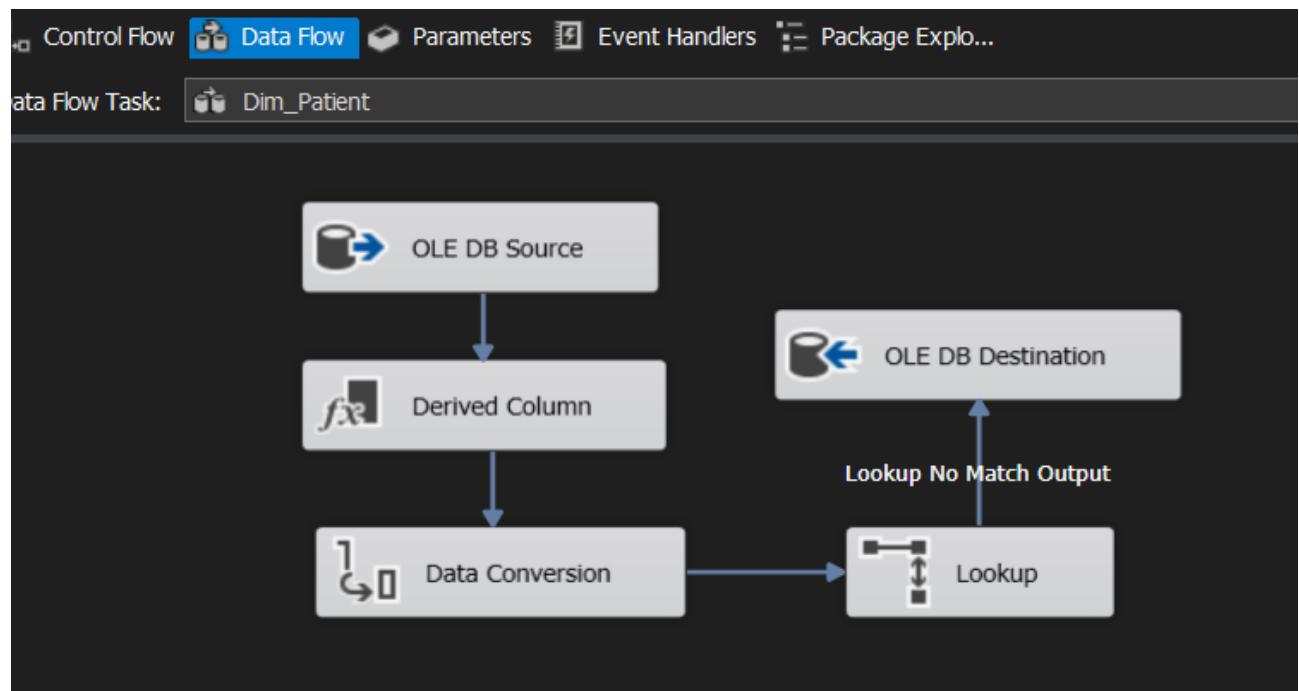


Рис. 3.4. Data flow – Dim\_Patient

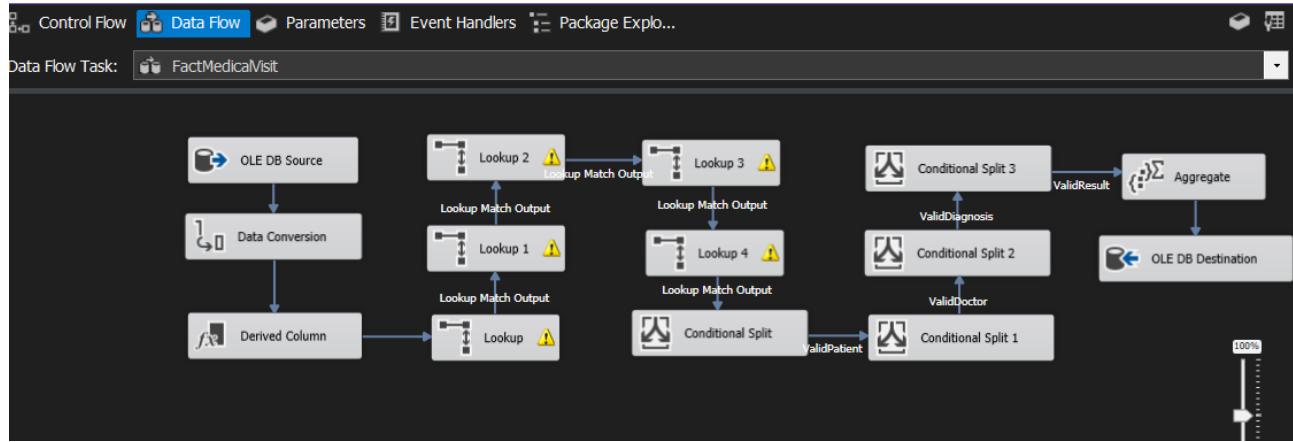


Рис. 3.5. Data flow – Fact\_MedicalVisit

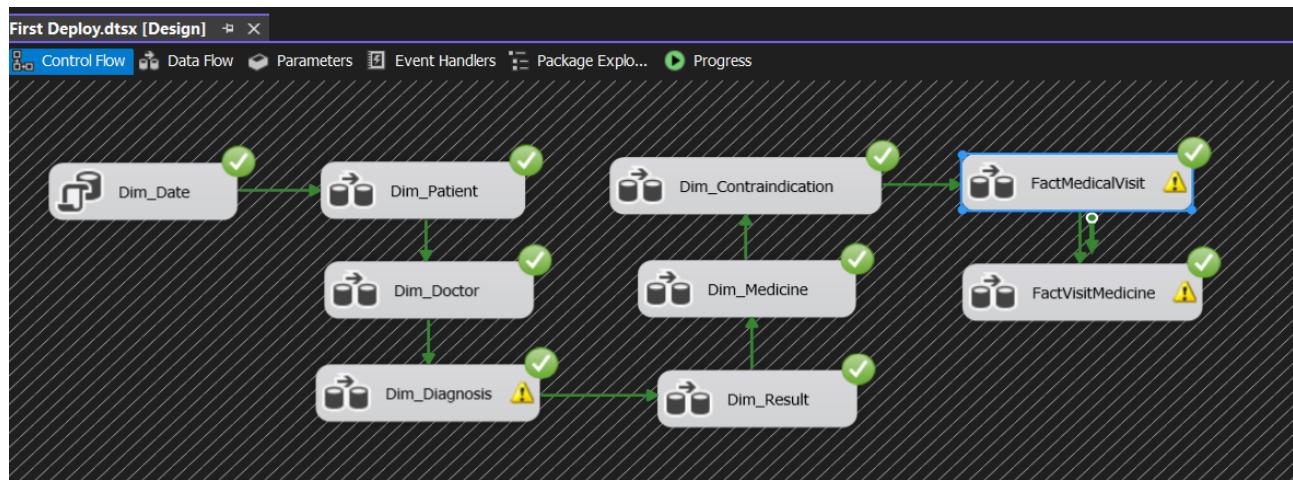


Рис. 3.6. Deployment

	DimContraindication_Count	DimDate_Count	DimDiagnosis_Count	DimDoctor_Count	DimMedicine_Count	DimPatient_Count	DimResult_Count	FactMedicalVisit_Count	FactVisitMedicine_Count
1	240	22280	375	60	135	50000	3	999998	10665746

Рис. 3.7. Дані успішно записані

## Висновки до розділу 3

У даному розділі було детально розглянуто процес реалізації ETL-процесів для предметної області «Поліклініка» з використанням SQL Server Integration Services. Реалізовані ETL-пакети забезпечують надійне, ефективне та масштабоване завантаження великих обсягів даних у сховище даних, що є основою для подальшого побудування OLAP-куба та аналітичних звітів.

## РОЗДІЛ 4

### ПОБУДОВА OLAP-КУБА ТА АНАЛІТИЧНИХ ЗВІТІВ

#### 4.1 Створення проєкту SQL Server Analysis Services

На даному етапі курсової роботи було реалізовано побудову багатовимірного OLAP-куба для аналізу даних поліклініки з використанням технології **SQL Server Analysis Services (SSAS)** у режимі **Multidimensional**.

Для створення проєкту було використано середовище **SQL Server Data Tools (SSDT)**, яке входить до складу Microsoft Visual Studio та забезпечує повний набір інструментів для розробки, налаштування та розгортання OLAP-рішень.

У середовищі Visual Studio було створено новий проект типу **Analysis Services Multidimensional and Data Mining Project**, у якому виконано налаштування підключення до сховища даних (Data Warehouse), сформованого на попередніх етапах курсової роботи.

**На цьому етапі виконано такі дії:**

- створено SSAS-проєкт;
- налаштовано підключення до бази даних Data Warehouse;
- підготовлено структуру для створення вимірів та таблиць фактів.

#### 4.2 Проектування Data Source View (DSV)

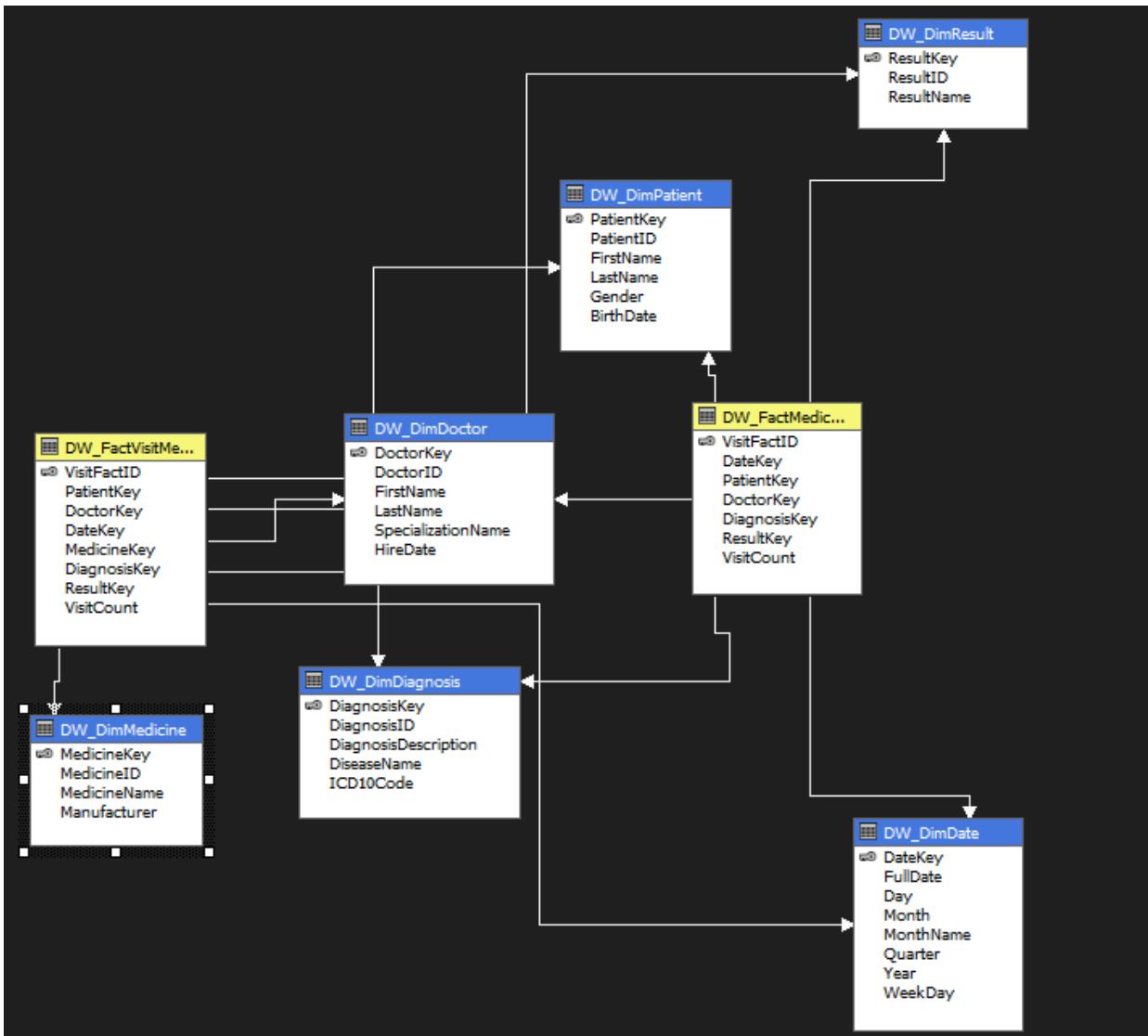
Після створення проєкту було розроблено **Data Source View (DSV)**, який є логічним поданням таблиць сховища даних, що використовуються в OLAP-кубі.

У DSV було додано всі необхідні таблиці вимірів та фактів, зокрема:

- таблиці вимірів:  
DW\_DimPatient, DW\_DimDoctor, DW\_DimDiagnosis, DW\_DimMedicine,  
DW\_DimDate, DW\_DimResult, DW\_DimContraindication;
- таблиці фактів: DW\_FactMedicalVisit, DW\_FactVisitMedicine.

Між таблицями було коректно налаштовано зв'язки за зовнішніми ключами, що дозволяє забезпечити цілісність даних та коректну агрегацію показників у кубі.

Також у DSV були використані **іменовані обчислення (Named Calculations)** для спрощення подальшої аналітики, зокрема для відображення повних імен пацієнтів та лікарів.



## 4.3 Створення вимірів (Dimensions)

На основі таблиць сховища даних у проекті SSAS було створено виміри, які дозволяють аналізувати дані за різними аспектами діяльності поліклініки.

### 4.3.1 Часовий вимір (Time Dimension)

Часовий вимір створено на основі таблиці DW\_DimDate. Він містить ієархію:

**Рік → Квартал → Місяць → День**

А також додаткові атрибути:

- день тижня;
- номер місяця;
- назва місяця.

Часовий вимір використовується для аналізу динаміки візитів, ефективності лікування та завантаженості лікарів у часі.

The screenshot shows the 'Dimension Structure' interface. On the left, under 'Attributes', there is a tree view for 'DW Dim Date' with nodes: Date Key, Day, Full Date, Month, and Year. In the center, under 'Hierarchies', a tooltip for 'Hierarchy' says: 'To create a new hierarchy, drag an attribute here.' Below it are four levels: Full Date, Day, Month, and Year, each with a dropdown arrow. On the right, the 'Data Source View' pane displays the schema for 'DW\_DimDate' with columns: DateKey, FullDate, Day, Month, MonthName, Quarter, Year, and WeekDay.

Рис. 4.1. Ієрархія Time Dimension

#### 4.3.2 Вимір “Пацієнти”

Вимір **Пацієнти** створено на основі таблиці DW\_DimPatient. Основними атрибутами виміру є:

- PatientKey (ключ виміру);
- ім’я та прізвище пацієнта;
- стать;
- дата народження.

Для зручності аналізу було налаштовано відображення повного імені пацієнта як **Name Column**, що дозволяє використовувати його у звітах та параметрах.

Attributes	Hierarchies	Data Source View
DW_Dim_Patient <ul style="list-style-type: none"> <li> Birth Date</li> <li> First Name</li> <li> Gender</li> <li> Last Name</li> <li> Patient Key</li> </ul>	To create a new hierarchy, drag an attribute here.	DW_DimPatient <ul style="list-style-type: none"> <li> PatientKey</li> <li> PatientID</li> <li> FirstName</li> <li> LastName</li> <li> Gender</li> <li> BirthDate</li> </ul>

Рис. 4.2. Налаштування виміру Patient

#### 4.3.3 Вимір “Лікарі”

Вимір **Лікарі** побудовано на основі таблиці DW\_DimDoctor. Він включає такі атрибути:

- ім’я та прізвище лікаря;
- спеціалізація;
- дата прийому на роботу.

Даний вимір використовується для аналізу навантаження лікарів, кількості прийомів та результатів лікування.

Attributes	Hierarchies	Data Source View
DW_Dim_Doctor <ul style="list-style-type: none"> <li> Doctor Key</li> <li> First Name</li> <li> Hire Date</li> <li> Last Name</li> <li> Specialization Name</li> </ul>	To create a new hierarchy, drag an attribute here.	DW_DimDoctor <ul style="list-style-type: none"> <li> DoctorKey</li> <li> DoctorID</li> <li> FirstName</li> <li> LastName</li> <li> SpecializationName</li> <li> HireDate</li> </ul>

Рис. 4.3. Вимір Doctor з атрибутами

#### 4.3.4 Вимір “Діагнози”

Вимір **Діагнози** створено на основі таблиці DW\_DimDiagnosis та включає:

- опис діагнозу;
- назву хвороби;
- код за класифікацією ICD-10.

Цей вимір дозволяє проводити аналіз поширеності захворювань та результатів лікування за конкретними діагнозами.

Attributes	Hierarchies	Data Source View
<input checked="" type="checkbox"/> DW_Dim_Diagnosis <ul style="list-style-type: none"><li><input type="checkbox"/> Diagnosis Description</li><li><input type="checkbox"/> Diagnosis Key</li><li><input type="checkbox"/> Disease Name</li><li><input type="checkbox"/> ICD10 Code</li></ul>	To create a new hierarchy, drag an attribute here.	<input checked="" type="checkbox"/> DW_DimDiagnosis <ul style="list-style-type: none"><li><input type="checkbox"/> DiagnosisKey</li><li><input type="checkbox"/> DiagnosisID</li><li><input type="checkbox"/> DiagnosisDescription</li><li><input type="checkbox"/> DiseaseName</li><li><input type="checkbox"/> ICD10Code</li></ul>

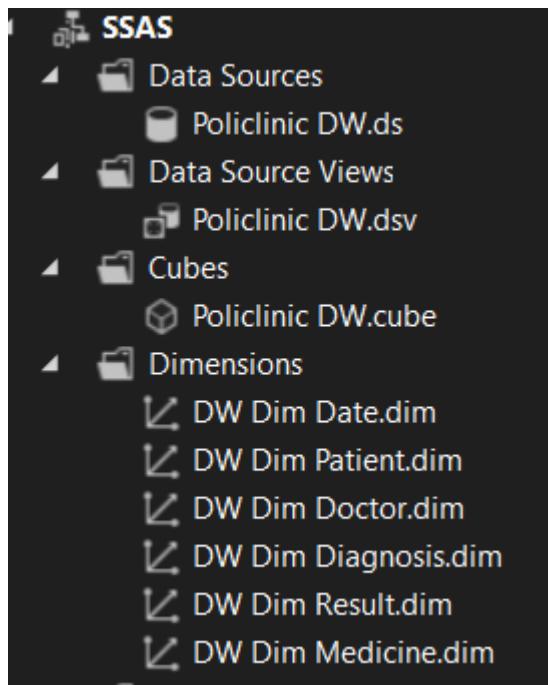
4.4. Вимір Diagnosis

#### 4.3.5 Інші виміри

Крім основних, у кубі також створено додаткові виміри:

- Результат лікування** (DW\_DimResult);
- Ліки** (DW\_DimMedicine);
- Протипоказання** (DW\_DimContraindication).

Вони забезпечують глибоку аналітику лікувального процесу та використовуються у звітах і показниках ефективності.



#### 4.5. Список усіх створених вимірів

### 4.4 Створення мір та груп мір (Measures)

На основі таблиць фактів було створено групи мір:

- **Medical Visit Measures**
- **Visit Medicine Measures**

Основною мірою є **Visit Count**, яка використовується для:

- підрахунку кількості відвідувань;
- аналізу завантаженості лікарів;
- статистики захворювань.

Також у кубі налаштовано:

- Count-міри;
- обчислювані міри для відсоткових показників;
- середні значення.

Measures		Command
1	Policlinic DW	CALCULATE
2	[DW Fact Medical Visit]	[AvgVisitsperPatient]
3	Visit Count	[VisitsYTD]
4	DW Fact Medical Visit Count	[VisitsPY]
5	[DW Fact Visit Medicine]	[VisitsYoY%]
	Visit Count - DW Fact Visit Medicine	
	DW Fact Visit Medicine Count	

4.6. Measures у кубі

#### 4.5 Реалізація MDX-обчислень та KPI

Для розширення аналітичних можливостей куба були реалізовані MDX-обчислення, зокрема:

- аналіз динаміки візитів по роках;
- ранжування лікарів за кількістю прийомів;
- показники ефективності лікування.

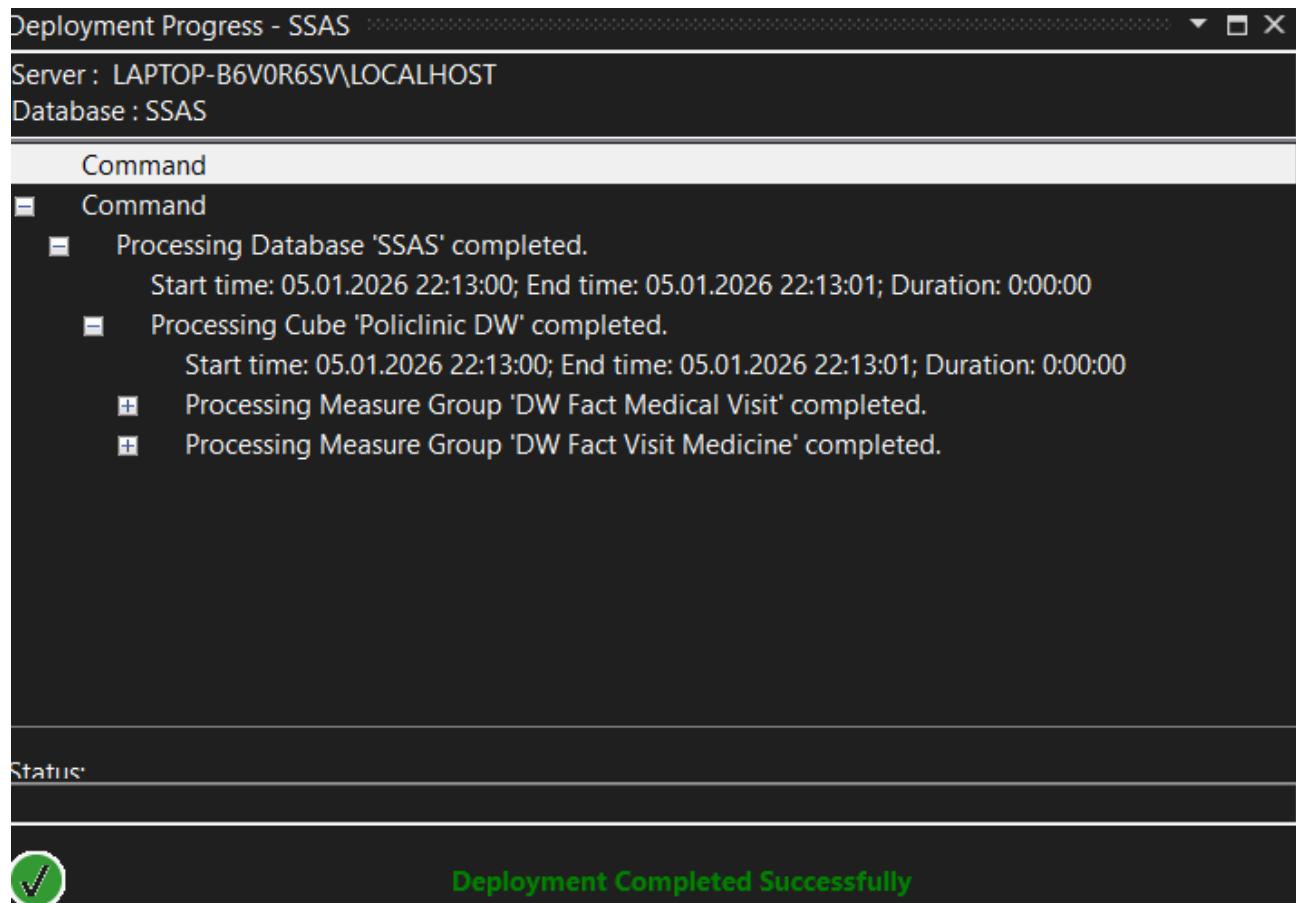
Також створено **KPI**, які дозволяють оцінювати:

- ефективність лікування;
- динаміку звернень пацієнтів.

#### 4.6 Оптимізація та розгортання куба

На фінальному етапі виконано:

- налаштування агрегацій;
- обробку куба (Process Full);
- тестування коректності даних;
- розгортання куба на сервері SSAS.



*Рис. 4.7. Успішне Process Cube*

#### 4.7 Використання OLAP-куба в аналітичних звітах

Створений OLAP-куб було використано як джерело даних для побудови аналітичних звітів у SQL Server Reporting Services, що дозволило реалізувати всі обов'язкові типи звітів, визначені технічним завданням.

## РОЗДІЛ 5

### СТВОРЕННЯ АНАЛІТИЧНИХ ЗВІТІВ З ВИКОРИСТАННЯМ SQL SERVER REPORTING SERVICES

#### 5.1 Загальна характеристика аналітичних звітів

На завершальному етапі курсової роботи було реалізовано систему аналітичних звітів з використанням технології **SQL Server Reporting Services (SSRS)**. Аналітичні звіти побудовані на основі **OLAP-куба поліклініки**, створеного за допомогою **SQL Server Analysis Services (SSAS)**, що забезпечує високу продуктивність обробки даних та можливість багатовимірного аналізу.

Метою створення звітів є:

- надання зручного доступу до аналітичної інформації;
- підтримка прийняття управлінських рішень у сфері охорони здоров'я;
- аналіз роботи лікарів, стану захворюваності та ефективності лікування;
- візуалізація ключових показників діяльності поліклініки.

У межах курсової роботи було розроблено **п'ять обов'язкових аналітичних звітів**, а також реалізовано параметризацію, інтерактивність, умовне форматування та можливість експорту результатів.

#### 5.2 Налаштування проєкту SQL Server Reporting Services

Для створення звітів у середовищі **Visual Studio (SQL Server Data Tools)** було виконано наступні кроки:

1. Створено новий проєкт типу **Report Server Project**.
2. Налаштовано **Shared Data Source**, що підключається до OLAP-куба *Policlinic DW*.
3. Для кожного звіту створено окремий файл формату *.rdl*.
4. Як джерело даних використано **MDX-запити** до багатовимірного куба.

#### 5.3 Реалізація обов'язкових аналітичних звітів

##### 5.3.1 Звіт “Список лікарів та їх пацієнтів”

Даний звіт призначений для відображення інформації про лікарів поліклініки та пацієнтів, яких вони обслуговують. Звіт реалізований у вигляді **табличного звіту** (**Table Report**) з можливістю групування та сортування.

У звіті відображаються такі дані:

- ім'я та прізвище лікаря;
- діагноз та назва хвороби;
- ім'я та прізвище пацієнта;
- кількість візитів пацієнта до лікаря;
- дата візиту.

Звіт підтримує:

- сортування за ім'ям пацієнта;
- відображення підсумкової кількості візитів.

The screenshot shows a table report interface with a blue header bar. The header includes tabs for 'Design' and 'Preview', and a toolbar with various icons. Below the header is a navigation bar with buttons for page number (1), search, and navigation. The main area displays a table with the following data:

First Name	Last Name	First Name2	Last Name2	Diagnosis Description	Disease Name	Date Key	Visit Count
Aaron	Brennan	Cheri	Gould	Chronic Nervous System Syndrome	Acute Inflammation of Digestive System	20130718	1
Aaron	Lin	Alexander	Mc Clure	Moderate Respiratory System Disorder	Recurrent Infection of Lung	20170928	1
Aaron	Sexton	Katie	Cantu	Severe Liver Syndrome	Severe Infection of Nervous System	20060326	1
Aaron	Haynes	Larry	Glover	Chronic Heart Disorder	Chronic Disorder of Nervous System	20221004	1
Aaron	Green	Tonia	Baldwin	Severe Respiratory System Disease	Infection of Respiratory System	20030731	1

*Рис. 5.1. Скріншот табличного звіту “Список лікарів та їх пацієнтів” у режимі перегляду*

### 5.3.2 Матричний звіт: «Кількість захворювань за роками»

Матричний звіт (Matrix Report) призначений для аналізу **динаміки захворювань у часовому розрізі**.

Структура звіту:

- рядки — назви захворювань;

- колонки — роки (з часового виміру);
- значення — кількість випадків захворювання.

### **Функціональні можливості:**

- автоматична агрегація даних;
- порівняння кількості випадків захворювань за різні роки;
- наочне відображення тенденцій росту або зменшення поширеності хвороб.

	2000	2001	2002	2003	2004	2005	2006	
Acute Disease of Bone	21	24	32	28	24	37	34	
Acute Disease of Digestive System	74	103	107	111	123	135	125	
Acute Disease of Respiratory System	53	55	71	63	68	68	43	
Acute Disease of Stomach	44	48	50	52	61	58	60	
Acute Disorder of Bl	24	27	26	27	28	45	13	
Acute Disorder of Joi	27	33	23	36	37	21	24	
Acute Disorder of Ki	51	55	60	55	61	49	65	

*Рис. 5.2. Матричний звіт «Кількість захворювань за роками»*

### **5.3.3 Стовпчикова діаграма (Column Chart): «Завантаженість лікарів за рік»**

Даний звіт реалізовано у вигляді **стовпчикової діаграми (Column Chart)** та використовується для аналізу навантаженості лікарів.

#### **Опис діаграми:**

- вісь X — прізвища лікарів;
- вісь Y — кількість відвідувань за обраний рік;
- значення — агрегована міра *Visit Count*.

Звіт дозволяє швидко визначити лікарів із найбільшим та найменшим навантаженням, що є корисним для управлінських рішень.

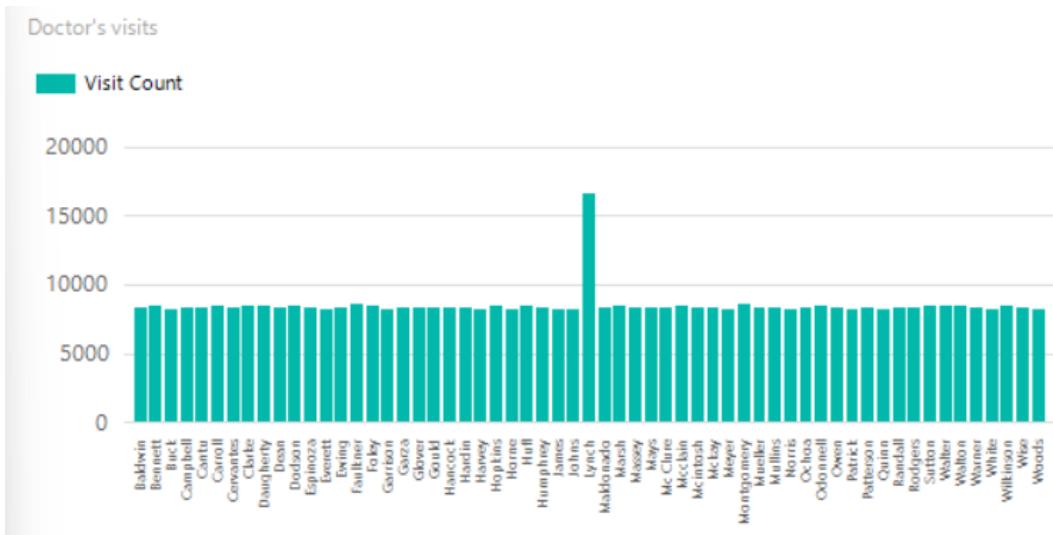


Рис. 5.3. Column Chart «Завантаженість лікарів за рік»

### 5.3.4 Лінійна діаграма (Line Chart): «Результати лікування за роками»

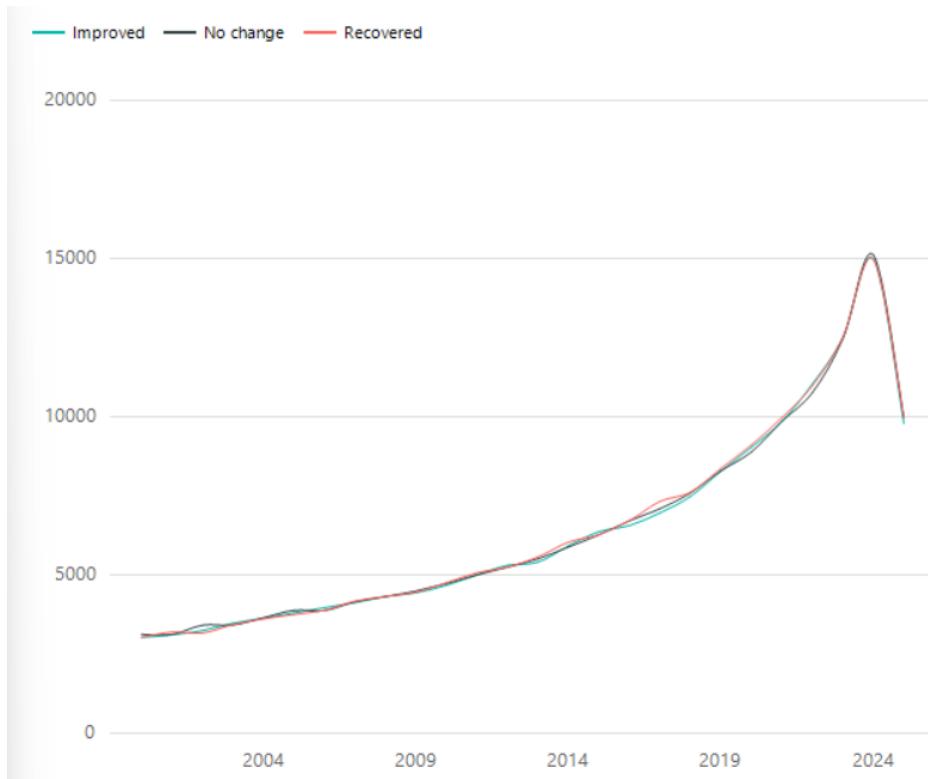
Лінійний звіт використовується для аналізу **результатів лікування пацієнтів у динаміці**.

**Структура звіту:**

- вісь X — роки;
- вісь Y — кількість візитів;
- окремі лінії — результати лікування (наприклад: *вилікувано, без змін*).

**Переваги звіту:**

- відображення тенденцій ефективності лікування;
- можливість порівняння результатів між роками;
- наочне представлення медичної статистики.



*Рис. 5.4. Line Chart «Результати лікування за роками»*

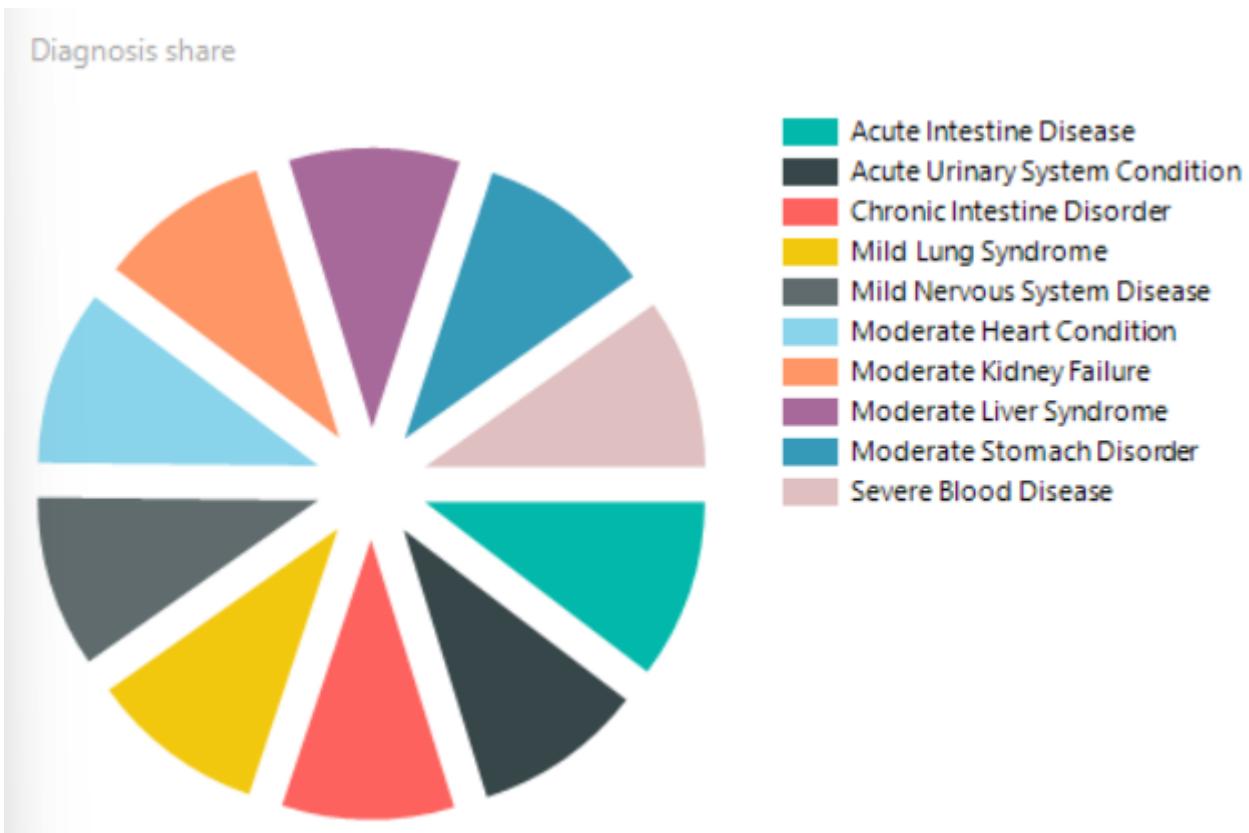
### **5.3.5 Кругова діаграма (Pie Chart): «Топ-10 найпоширеніших захворювань»**

Кругова діаграма призначена для відображення **структурі захворюваності**.

**Особливості звіту:**

- відображення лише **10 найпоширеніших захворювань**;
- значення сегментів — кількість випадків;
- автоматичне ранжування захворювань за популярністю.

Звіт дозволяє швидко оцінити, які захворювання є найбільш актуальними для поліклініки.



*Рис. 5.5. Pie Chart «Top-10 найпоширеніших захворювань»*

## 5.4 Параметризація та інтерактивність звітів

У розроблених звітах реалізовано такі типи параметрів:

- **Dropdown-параметри** для вибору лікаря та пацієнта;
- **Multi-select параметри** для вибору декількох діагнозів;
- **Date Range параметри** для аналізу даних за заданий період;
- **Cascading параметри**, де вибір лікаря впливає на список доступних пацієнтів.

Також реалізовано:

- сортування при натисканні на заголовки колонок;
- розгортання та згортання груп (toggle items);
- умовне форматування для візуального виділення критичних показників.

## 5.5 Експорт та розгортання звітів

Розроблені звіти можуть бути експортовані у такі формати:

- PDF;
- Microsoft Excel;
- Microsoft Word.

# ВИСНОВКИ

У ході виконання курсової роботи було досягнуто поставленої мети — розроблено та проаналізовано повнофункціональну систему керування надвеликою базою даних для предметної області «Поліклініка» з використанням технологій Microsoft SQL Server.

У першому розділі проведено детальний аналіз предметної області поліклініки. Було визначено основні сутності системи, такі як пацієнти, лікарі, діагнози, захворювання, лікарські засоби, результати лікування та протипоказання. Описано ключові бізнес-процеси, зокрема реєстрацію пацієнтів, облік візитів, постановку діагнозів, призначення лікування та аналіз результатів. Сформульовано функціональні вимоги та бізнес-правила, що стали основою для подальшого проєктування бази даних.

У другому розділі виконано проєктування бази даних поліклініки. Побудовано концептуальну ER-модель із визначенням сутностей, атрибутів та зв'язків між ними. Проведено логічне проєктування з нормалізацією до третьої нормальної форми, визначено первинні та зовнішні ключі, а також спроектовано фізичну структуру бази даних. Реалізовано SQL-скрипти створення таблиць, обмеження цілісності та зв'язки між об'єктами бази даних.

Третій розділ присвячено реалізації ETL-процесів з використанням SQL Server Integration Services. Було спроектовано сховище даних (Data Warehouse) за схемою «зірка», створено таблиці фактів і вимірів, реалізовано процеси витягування, трансформації та завантаження даних. У рамках ETL застосовано різні типи трансформацій, зокрема очищення даних, конвертацію типів, похідні колонки, пошук у довідниках та агрегацію. Забезпечено контроль якості даних і перевірено коректність завантаження великих обсягів інформації.

У четвертому розділі виконано побудову багатовимірного OLAP-куба з використанням SQL Server Analysis Services. Створено виміри, міри та групи мір, реалізовано ієархії та обчислювані показники. Налаштовано бізнес-метрики, рейтинги, часову аналітику та показники ефективності лікування. Проведено оптимізацію куба та його тестування на коректність і продуктивність.

П'ятий розділ присвячено створенню аналітичних звітів у SQL Server Reporting Services. Розроблено різні типи звітів, зокрема табличні, матричні та графічні. Реалізовано дашборд із використанням KPI, діаграм і інтерактивних елементів. Створено параметризовані звіти з випадаючими списками, діапазонами дат та каскадними параметрами. Забезпечено можливість drill-down, сортування, умовного форматування та експорту звітів у різні формати.

У результаті виконання курсової роботи було продемонстровано практичні навички роботи з надвеликими базами даних, ETL-процесами, OLAP-аналітикою та системами звітності. Отримана система може бути використана для аналізу діяльності поліклініки, оцінки ефективності лікування, контролю завантаженості лікарів та підтримки управлінських рішень.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Microsoft Corporation. **SQL Server Documentation.**  
URL: <https://learn.microsoft.com/sql>
2. Microsoft Corporation. **SQL Server Integration Services (SSIS).**  
URL: <https://learn.microsoft.com/sql/integration-services>
3. Microsoft Corporation. **SQL Server Analysis Services (SSAS).**  
URL: <https://learn.microsoft.com/analysis-services>
4. Microsoft Corporation. **SQL Server Reporting Services (SSRS).**  
URL: <https://learn.microsoft.com/sql/reporting-services>
5. Kimball R., Ross M. **The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling.** — 3rd Edition. — Wiley, 2013.
6. Inmon W. H. **Building the Data Warehouse.** — 4th Edition. — Wiley, 2005.
7. Silberschatz A., Korth H., Sudarshan S. **Database System Concepts.** — 6th Edition. — McGraw-Hill, 2011.
8. Date C. J. **An Introduction to Database Systems.** — 8th Edition. — Pearson Education, 2003.
9. Elmasri R., Navathe S. **Fundamentals of Database Systems.** — 7th Edition. — Pearson, 2016.
10. ISO/IEC 9075. **Information technology — Database languages — SQL.**