

## КУРСОВА РОБОТА

з дисципліни «Бази даних»

на тему: «Розроблення бази даних для розв’язання завдання «Облік  
отриманих коштів за продажі в книжковому магазині»

Виконав: здобувач вищої освіти 3 курсу,  
групи ПЗ-22-1ду  
спеціальності 121 «Інженерія програмного  
забезпечення»

(код і найменування спеціальності)

Освітньо-професійної програми «Інженерія  
програмного забезпечення»

Тамбовцев А.Є.

(прізвище та ініціали)

Керівник: к.ф.-м.н., доцент Фірсов О.Д.

Національна шкала: \_\_\_\_\_

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

Члени комісії:

\_\_\_\_\_  
(підпис)

\_\_\_\_\_  
(підпис)

\_\_\_\_\_  
(підпис)

Фірсов О.Д.  
(прізвище та ініціали)

Жульковський О.О.  
(прізвище та ініціали)

Журавський О.Д.  
(прізвище та ініціали)

Дніпровський державний технічний університет

Факультет Комп'ютерних технологій та енергетики

Кафедра Програмне забезпечення систем

Спеціальність 121 Інженерія програмного забезпечення

## **ЗАВДАННЯ** **на курсову роботу**

з дисципліни Бази даних

здобувачу вищої освіти групи ПЗ-22-1ду

ПІБ Тамбовцеву Артему Євгеновичу

тема курсової роботи «Розроблення бази даних для розв'язання завдання  
«Облік отриманих коштів за продажі в книжковому магазині»»

дата видачі завдання 23.01.2024 р.

дата здачі роботи 10.05.2024 р.

Завдання видав  
керівник курсової роботи  
к.ф.-м.н., доцент

Олександр ФІРСОВ

Завдання прийняв до виконання  
здобувач вищої освіти

Артем ТАМБОВЦЕВ

Міністерство освіти і науки України

Дніпровський державний технічний університет

Здобувач Тамбовцев А.Є групи ПЗ-22-1ду

Тема курсової роботи: «Розроблення бази даних для розв'язання завдання  
«Облік отриманих коштів з продажі книжкового магазину»

Оцінювання

Критерій оцінювання	Максимальна оцінка, балів
Опис предметної області	10
Проектування бази даних:	30
- Інфологічна модель БД	15
- Логічна структура БД	15
Фізична реалізація БД за допомогою мови Transact-SQL	10
Реалізація багатокористувацького доступу до БД	5
Реалізація тригерів	10
Реалізація збережених процедур	10
Реалізація представлень	10
Клієнтська частина	10
Оформлення пояснювальної записки	5
<b>Всього</b>	<b>100б</b>

Захист відбувся 10.05.2024

Загальна оцінка \_\_\_\_\_

Керівник курсової роботи \_\_\_\_\_ Олександр ФІРСОВ

## РЕФЕРАТ

Курсова робота: 57 с., 21 рис., 8 таблиць, 6 додатків, 7 джерел.

Об'єктом дослідження є розроблення бази даних для розв'язання завдання "Облік отриманих коштів з продажу книжкового магазину"

Мета роботи: аналіз та опис предметної області, розробка структури бази даних, створення SQL-запитів для ефективного управління даними та реалізація клієнтського додатка для взаємодії з базою даних.

У розділі «Аналіз та опис предметної області» – описується предметна область, вхідні та вихідні документи.

У розділі «Проектування реляційної бази даних» – будується інфологічна модель бази даних з описом сутностей та їх атрибутів. Далі будується ER-діаграма де показані зв'язки між сутностями та будується даталогічна модель.

У розділі «Створення бази даних за допомогою MS SQL Server» – в MS SQL Server за допомогою мови Transact-SQL створюється база даних, таблиці, діаграми, заповнюються таблиці даними.

У розділі «Організація багатокористувацького режиму доступу до БД» – наводяться результати створення користувачів БД.

У розділі «Тригери бази даних» – наводяться запити створення тригерів для операцій додавання, видалення, оновлення даних таблиць БД.

У розділі «Збережені процедури» наводяться запити створення збережених процедур.

У розділі «Представлення» наводяться запити розробки представлень, що вирішують поставлену задачу автоматизації.

У розділі «Створення клієнтської частини програми для роботи з базою даних» – створюється клієнтський додаток з використанням Visual C# 2022, який працює з БД «Облік отриманих коштів за продажі в книжковому магазині» і дозволяє вести облік продажі в книжковому магазині.

У розділі «Перелік посилань» – наведено перелік літератури, що використовувалася для написання курсової роботи.

У розділі «Додатки» – наведено приклади створення таблиць БД, заповнення таблиць БД, лістингу програми..

КЛЮЧОВІ СЛОВА: БАЗА ДАНИХ, SQL SERVER, АНАЛІЗ, ПРОЕКТУВАННЯ, РЕАЛІЗАЦІЯ, КНИЖКОВИЙ МАГАЗИН , ОБЛІК.

## ЗМІСТ

Вступ.....	7
1. Аналіз та опис предметної області.....	9
1.1 Загальний опис предметної області.....	9
1.2 Опис вхідних даних.....	11
1.3 Опис вихідних даних.....	11
2. Проектування реляційної бази даних.....	12
2.1 Інфологічна модель бази даних.....	12
2.1.1 Опис сутностей.....	12
2.1.2 Опис зв'язків.....	14
2.1.3 ER-діаграма.....	16
2.2 Логічна структура бази даних.....	16
3. Створення бази даних за допомогою MS SQL Server.....	19
3.1 Створення бази даних.....	19
3.2 Створення таблиць.....	19
3.3 Створення діаграми бази даних.....	20
3.4 Заповнення таблиць бази даних даними.....	22
4. Організація багатокористувацького режиму доступу до бази даних.....	23
5. Тригери бази даних.....	26
6. Збережені процедури.....	28
7. Представлення бази даних.....	32
8. Створення клієнтської частини програми для роботи з БД.....	34
Висновки.....	43
Перелік посилань.....	45
Додатки.....	46

## ВСТУП

Протягом останніх тридцяти років в галузі теорії систем баз даних пройшли значні досягнення, що стали ключовими в інформатиці за цей період. Базы даних стали необхідним елементом інформаційних систем і суттєво змінили способи роботи багатьох організацій. Недавній розвиток технологій баз даних призвів до створення потужних та простих у використанні систем, які стали доступними для широкого кола користувачів.

Проте, на жаль, з цим зростанням доступності користувачі почали створювати бази даних і додатки самостійно, часто не маючи достатніх знань про ефективні методи проектування систем. Це часто призводило до марних витрат ресурсів та неякісних результатів. Це, в свою чергу, стало причиною "кризи програмного забезпечення", наслідки якої залишаються актуальними і нині.

База даних - це модель, яка дозволяє структурувати та зберігати дані про об'єкти з однаковими властивостями. Сучасні бази даних, незалежно від того, чи вони реалізовані на комп'ютері, використовують таблиці для зберігання даних. Останнім часом реляційні бази даних стали найпоширенішими. У реляційних базах даних інформація зберігається у відношеннях, які можна сприймати як таблиці, пов'язані між собою.

Основні функції системи SQL Server включають організацію одночасного доступу великої кількості користувачів до даних та маніпуляції інформацією, збереженою в базі даних. SQL Server підтримує реляційну модель даних та здійснює створення об'єктів бази даних, перевірку цілісності та забезпечення безпеки даних у системі.

Доступ користувача до даних зазвичай відбувається з комп'ютера робочої станції, через спеціалізовані додатки. Адміністрування баз даних в SQL Server зручно виконується безпосередньо з комп'ютера-сервера за допомогою мови Transact SQL. Файли баз даних зберігаються на дисках сервера з розширенням MDF, а системні файли - з розширенням LDF.

Управління роботою SQL Server здійснюється за допомогою різних утиліт, які включають SQL Server BooksOnline для надання довідкової підтримки, SQL Server QueryAnalyzer для виконання SQL-операцій та інші. Реляційна СУБД є основою для структурування та організації даних в сучасних підприємствах і організаціях, що забезпечує інформаційну діяльність у різних сферах від виробництва до фінансів.

Зазначу, що на фоні швидкого розвитку технологій і зростання обсягів даних, важливість правильного проектування баз даних і використання ефективних методів управління ними стає ще більшою. Наприклад, збереження даних відповідно до нормативних вимог, забезпечення їх цілісності та конфіденційності, а також оптимізація запитів для швидкого доступу до інформації - це лише деякі з аспектів, які потребують уваги при розробці баз даних.

Крім того, інтеграція баз даних з іншими системами, такими як веб-сервери, хмарні сервіси та мобільні додатки, стає все більш важливою для забезпечення швидкого та ефективного обміну інформацією між різними платформами.

Тому використання сучасних інструментів та розумних підходів до розробки і адміністрування баз даних є ключовим для успішної роботи будь-якої організації у сучасному цифровому світі.

У цій курсовій роботі була розроблена база даних у Microsoft SQL Server 2022.



# 1 АНАЛІЗ ТА ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Загальний опис предметної області

У сучасному світі книжкові магазини становлять важливу складову культурного та освітнього середовища. Вони виступають не лише як торгові точки для купівлі та продажу книг, але й як місця, де люди знаходять знання, розвагу та відпочинок. Підприємства цього типу є ключовими учасниками книжкового ринку, і їхній успіх значною мірою залежить від ефективного управління фінансами та ведення обліку прибутку.

Облік отриманих коштів за продажі є критично важливою складовою функціонування книжкового магазину. Цей процес включає в себе збір, систематизацію та аналіз фінансових даних, пов'язаних з продажем книг. Ці дані охоплюють такі аспекти, як кількість проданих примірників, ціна кожної книги, інформація про книги, кількість покупців та їх інформація.

Ця інформація не лише дозволяє магазину відстежувати свої фінансові операції та визначати прибутковість, але й надає базу для прийняття стратегічних рішень щодо асортименту товарів, ціноутворення, маркетингових кампаній та інших аспектів діяльності. Таким чином, ефективний облік отриманих коштів є ключовим елементом успішного ведення бізнесу у сфері книготоргівлі.

Функціонал бази даних дозволяє нам виконувати різноманітні операції для керування та отримання інформації про ваш книжковий магазин. Ось деякі з них:

- 1) Додавання нових записів:
  - Додавання нових книг, авторів, жанрів, клієнтів, замовлень та інших даних до відповідних таблиць у базі даних.
- 2) Редагування існуючих записів:

- Зміна інформації про книги, авторів, клієнтів, замовлення тощо.

### 3) Видалення записів:

- Видалення зайвих записів, якщо вони більше не потрібні або їх інформація більше не актуальна.

### 4) Пошук та фільтрація даних:

- Пошук книг за назвою, автором, жанром або іншими параметрами.
- Фільтрація клієнтів за місцезнаходженням, електронною адресою або іншими атрибутами.

### 5) Генерація звітів:

- Створення звітів про продажі, доходи, популярність книг та іншу статистику.
- Аналіз даних для прийняття управлінських рішень та планування діяльності магазину.

### 6) Аналіз продажів:

- Визначення найпопулярніших книг, авторів чи жанрів за певний період.
- Встановлення тенденцій у споживанні літератури для покращення асортименту.

### 7) Керування замовленнями:

- Відстеження стану замовлень, обробка нових замовлень та відправка товару клієнтам.

### 8) Управління клієнтами:

- Ведення бази даних клієнтів, зберігання їх контактної інформації та історії покупок.

Цей функціонал дозволяє ефективно керувати нашим книжковим магазином, відстежувати продажі та забезпечувати задоволення потреб клієнтів. Дані таблиці взаємопов'язані за допомогою зовнішніх ключів, що забезпечує цілісність даних та дозволяє виконувати різноманітні запити та аналізи.

## **1.2 Опис вхідних даних**

У базі даних "Облік отриманих коштів за продажі в книжковому магазині" використовуються наступні вхідні дані:

- Інформація про продажі книг, включаючи назву книги, кількість проданих примірників, ціну кожної книги та дату продажу.
- Інформація про клієнтів, яка включає їхні імена, контактні дані (електронна пошта, телефон) та ідентифікатори.
- Інформація про книги, включає їхні назви, авторів, жанри та ціни.

## **1.3 Опис вихідних даних**

Вихідними даними є результати обліку та аналізу фінансової діяльності книжкового магазину, які включають:

- Звіти про продажі за певний період, включаючи загальну кількість проданих книг, суму отриманих коштів та статистику продажів по книгах, авторах чи жанрах.
- Звіти про клієнтів, які містять інформацію про покупців, їхні покупки та зворотний зв'язок.

Аналіз рентабельності, що включає прибутковість від продажів, витрати та інші фінансові показники, які допомагають у прийнятті управлінських рішень.

## 2 ПРОЕКТУВАННЯ РЕЛЯЦІЙНОЇ БАЗИ ДАНИХ

### 2.1 Інфологічна модель бази даних

Мета інфологічного проектування – забезпечення найбільш природних для людини способів збору і представлення тієї інформації, яку передбачається зберігати в створеній БД. Тому інфологічну модель намагаються будувати за аналогією з природною мовою. Основним конструктивними елементами інфологічних моделей є сутності, зв'язки між ними та їх властивості, що описуються у вигляді відповідних специфікацій предметної області.

#### 2.1.1 Опис сутностей

У відповідності з описом предметної області було отримано такі сутності, опис яких наведено в таблиці 2.1.

Таблиця 2.1 – Специфікація сутностей предметної області

№	Назва сутності	Опис сутності
1	Автори	Інформація про авторів книг
2	Жанри	Інформація про жанри книг
3	Книги	Інформація про книги та їх атрибути
4	Клієнти	Інформація про клієнтів книгарні
5	Продажі	Інформація про продажі книг та їх атрибути
6	Замовлення	Інформація про замовлення книг та їх атрибути

Опис атрибутів сутностей предметної області наведено в таблиці 2.2.

Таблиця 2.2 – Специфікація атрибутів сутностей предметної області

Назва сутності	Назва атрибуту	Опис атрибуту
Автори	ID автора	Унікальний ID автора
	ПІБ	ПІБ автора
	Дата народження	Дата народження автора
	Дата смерті	Дата смерті автора (якщо є)
Жанри	ID жанру	Унікальний ID жанру
	Назва	Назва жанру
Книги	ID книги	Унікальний ID книги
	Назва книги	Назва книги
	ID автора	Посилання на ID автора книги
	ID жанру	Посилання на ID жанру книги
	Ціна	Ціна книги
	Дата написання	Дата написання книги
Клієнти	ID клієнта	Унікальний ID клієнта
	ПІБ	ПІБ клієнта
	Електронна пошта	Електронна пошта клієнта
	Телефон	Номер телефону клієнта
	ID адреси	Посилання на ID адреси клієнта
Продажі	ID продажу	Унікальний ID продажу
	Кількість	Кількість проданих книг
	Ціна	Ціна проданої книги
	Дата продажу	Дата продажу
Замовлення	ID замовлення	Унікальний ID замовлення
	ID клієнта	Посилання на ID клієнта
	ID продажу	Посилання на ID продажу
	ID книги	Посилання на ID книги

### 2.1.2 Опис зв'язків

Зв'язок – асоціювання двох і більше сутностей. Якби призначенням БД було тільки збереження окремих, не пов'язаних між собою даних, то її структура могла бути дуже проста. Проте одна з основних вимог до організації бази даних – це забезпечення можливості відшукування одних сутностей за призначенням інших, для чого необхідно встановити між ними певні зв'язки.

Модель «сутність – зв'язок» заснована на використанні 3-х основних конструктивних елементів: сутність, атрибут, зв'язок. Взаємозв'язки між таблицями БД можуть бути типізовані за такими основними видами:

Відношення «один до одного» (1:1) означає, що кожному запису однієї таблиці відповідає тільки один запис в іншій таблиці;

Відношення «один до багатьох» (1:Б) виникає, коли один запис взаємопов'язаний з багатьма іншими;

Відношення «багато до одного» означає, що багато записів пов'язані з однією (Б:1);

Відношення «багато до багатьох» (Б:N) виникає між двома таблицями в тих випадках, коли:

Один запис із першої таблиці може бути пов'язаний більш ніж з одним записом із другої таблиці;

Один запис з другої таблиці може бути пов'язаний більш ніж з одним записом з першої таблиці.

Недоліком даної моделі є те, що одні й ті ж елементи можуть виступати одночасно і як сутності, і як атрибути, і як зв'язки. В даному випадку, будемо вважати, що кожен об'єкт може виступати тільки в якості одного конструктивного елемента. В курсовій роботі були використані наступні типи зв'язків (таблиця 2.3).

Таблиця 2.3 – Специфікація зв'язків

Номер зв'язку	Головна таблиця	Дочірня таблиця	Тип зв'язку
1	Автори	Книги	1:Б
2	Жанри	Книги	1:Б
3	Книги	Продажі	1:Б
4	Клієнти	Замовлення	1:Б

Зв'язок між таблицями "Автори" та "Книги" вказує на те, що один автор може мати декілька книг.

Зв'язок між таблицями "Жанри" та "Книги" вказує на те, що один жанр може мати декілька книг.

Зв'язок між таблицями "Книги" та "Продажі" вказує на те, що одна книга може бути продана у декількох екземплярах.

Зв'язок між таблицями "Клієнти" та "Замовлення" вказує на те, що один клієнт може робити декілька замовлень.

### 2.1.3 ER-діаграма

На рисунку 2.1 представлено ER-діаграма на якій відображені всі сутності, їх атрибути та зв'язки між сутностями.

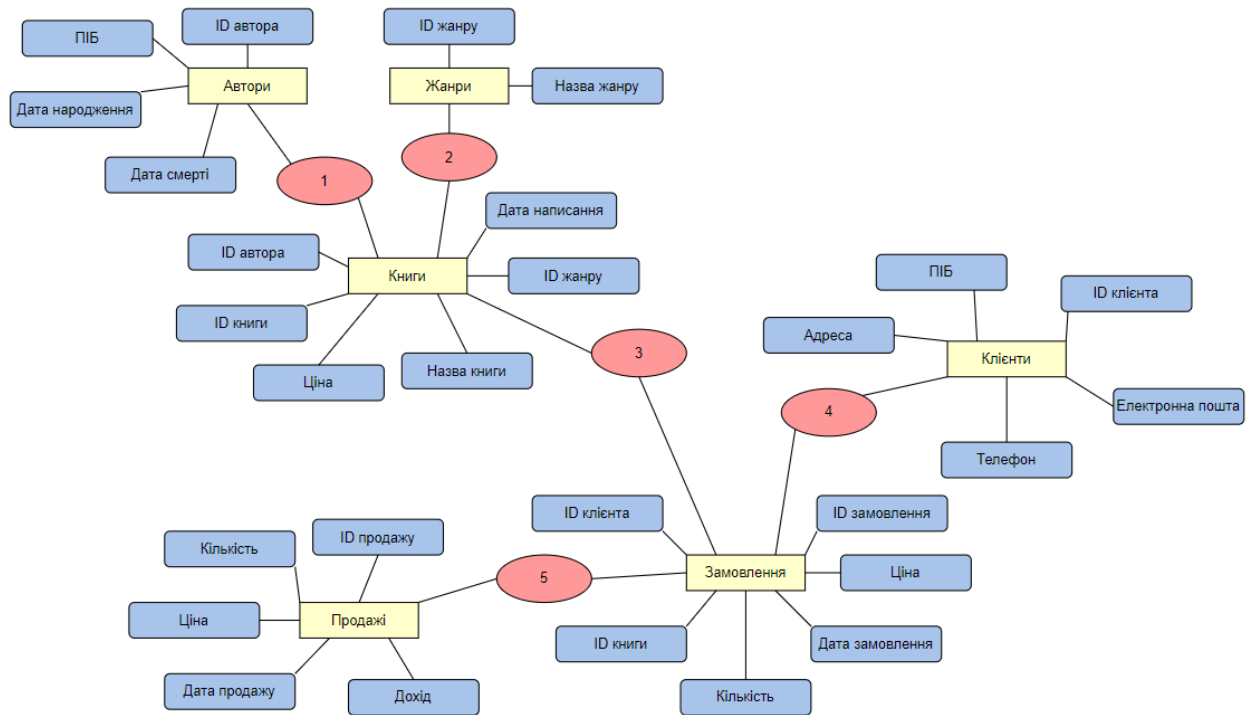


Рисунок 2.1 – ER-діаграма

### 2.2 Логічна структура бази даних

Для кожного поля таблиці бази даних (таблиці 2.4-2.9) вказується розмір поля (кількість символів), тип. Для первинних ключів необхідно ввести заборону невизначених значень. Для інших полів – можливість заборони невизначених значень визначається семантикою предметної області.



Таблиця 2.4 – Таблиця Автори (Authors)

Найменування атрибута	Тип поля	Розмір поля	Обмеження
ID автора (AuthorID)	INT	4	NOT NULL,AUTO_INCREMENT
ПІБ (Name)	NVARCHAR	100	NOT NULL
Дата народження (Birthdate)	DATE		NOT NULL
Дата смерті (DateOfDeath)	DATE		

Таблиця 2.5 – Таблиця Жанри (Genres)

Найменування атрибута	Тип поля	Розмір поля	Обмеження
ID жанру (GenreID)	INT	4	NOT NULL,AUTO_INCREMENT
Назва жанру (Name)	NVARCHAR	50	NOT NULL

Таблиця 2.6 – Таблиця Книги (Books)

Найменування атрибута	Тип поля	Розмір поля	Обмеження
ID книги (BookID)	INT	4	NOT NULL,AUTO_INCREMENT
Назва книги (Title)	NVARCHAR	100	NOT NULL
ID автора (AuthorID)	INT		FOREIGN KEY (Authors)
ID жанру (GenreID)	INT		FOREIGN KEY (Genres)
Ціна (Price)	DECIMAL	10,2	NOT NULL
Дата написання (WritingDate)	DATE		NOT NULL

Таблиця 2.7 – Таблиця Клієнти (Customers)

Найменування атрибута	Тип поля	Розмір поля	Обмеження
ID клієнта (CustomerID)	INT	4	NOT NULL,AUTO_INCREMENT
ПІБ (Name)	NVARCHAR	100	NOT NULL
Електронна пошта (Email)	NVARCHAR	100	NOT NULL
Телефон (Phone)	NVARCHAR	13	NOT NULL
Адреса (Address)	NVARCHAR	100	NOT NULL

Таблиця 2.8 – Таблиця Продажі (Sales)

Найменування атрибута	Тип поля	Розмір поля	Обмеження
ID продажу (SaleID)	INT	4	NOT NULL,AUTO_INCREMENT
Кількість (Quantity)	INT	4	NOT NULL
Ціна (Price)	DECIMAL	10,2	NOT NULL
Дата продажу (SaleDate)	DATE		NOT NULL

Таблиця 2.9 – Таблиця Замовлення (Orders)

Найменування атрибута	Тип поля	Розмір поля	Обмеження
ID замовлення (OrderID)	INT	4	NOT NULL,AUTO_INCREMENT
ID клієнта (CustomerID)	INT		FOREIGN KEY (Customers)
ID книги (BookID)	INT		FOREIGN KEY (Books)
ID продажу (SaleID)	INT		FOREIGN KEY (Sales)

## 3 СТВОРЕННЯ БАЗИ ДАНИХ ЗА ДОПОМОГОЮ MS SQL SERVER

### 3.1 Створення бази даних

Для запуску SQL Server виберіть інструмент SQL Server Management Studio та відкрийте його.

Для написання програмного коду в SQL Server Management Studio натисніть кнопку "Новий запит" на панелі інструментів "Стандартна".

Створюємо нову базу даних з назвою "BookstoreManagement" за допомогою наступної команди: `CREATE DATABASE BookstoreManagement;` Для виконання команди натискаємо F5.

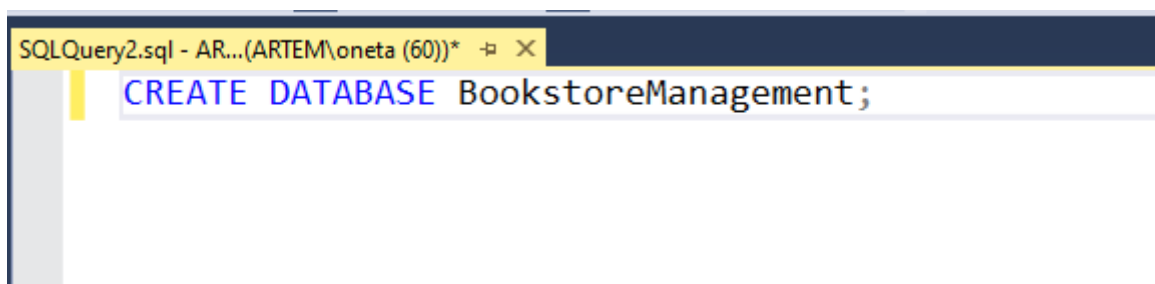
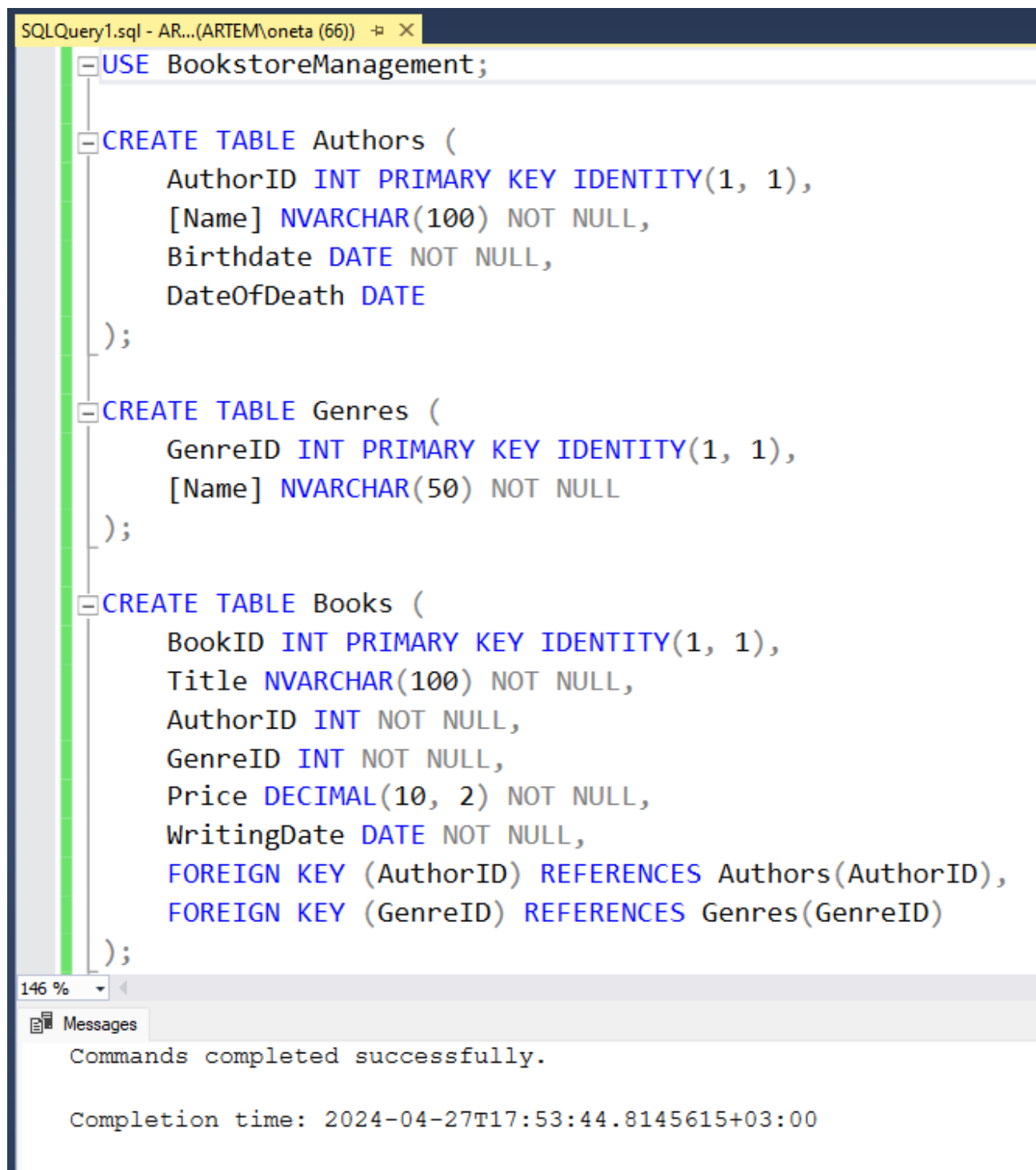


Рисунок 3.1 – Створення бази даних «BookstoreManagement»

### 3.2 Створення таблиць

В базі даних BookstoreManagement створюємо таблиці, що відображають основні сутності предметної області: книги (books), продажі (sales), клієнти (customers), замовлення (orders), автори (authors), жанри (genres). Використовуємо команди SQL Server Management Studio для створення нових таблиць. (Додаток А – Створення таблиць БД). Для виконання команди натискаємо F5. Результат зображено на рисунку 3.1.



```
SQLQuery1.sql - AR...(ARTEM\oneta (66))  X
USE BookstoreManagement;

CREATE TABLE Authors (
    AuthorID INT PRIMARY KEY IDENTITY(1, 1),
    [Name] NVARCHAR(100) NOT NULL,
    Birthdate DATE NOT NULL,
    DateOfDeath DATE
);

CREATE TABLE Genres (
    GenreID INT PRIMARY KEY IDENTITY(1, 1),
    [Name] NVARCHAR(50) NOT NULL
);

CREATE TABLE Books (
    BookID INT PRIMARY KEY IDENTITY(1, 1),
    Title NVARCHAR(100) NOT NULL,
    AuthorID INT NOT NULL,
    GenreID INT NOT NULL,
    Price DECIMAL(10, 2) NOT NULL,
    WritingDate DATE NOT NULL,
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),
    FOREIGN KEY (GenreID) REFERENCES Genres(GenreID)
);

146 %
Messages
Commands completed successfully.

Completion time: 2024-04-27T17:53:44.8145615+03:00
```

Рисунок 3.2 – Створення таблиць бази даних «BookstoreManagement»

### 3.3 Створення діаграми бази даних

В утиліті SQL Server ManagementStudio необхідно перевірити наявність бази даних BookstoreManagement і таблиць в ній.

У розділі діаграм створюємо нову діаграму, в яку додаємо зі списку наших таблиць, перевіряємо зв'язки між таблицями. Результат створення діаграми зображено на рисунку 3.2.

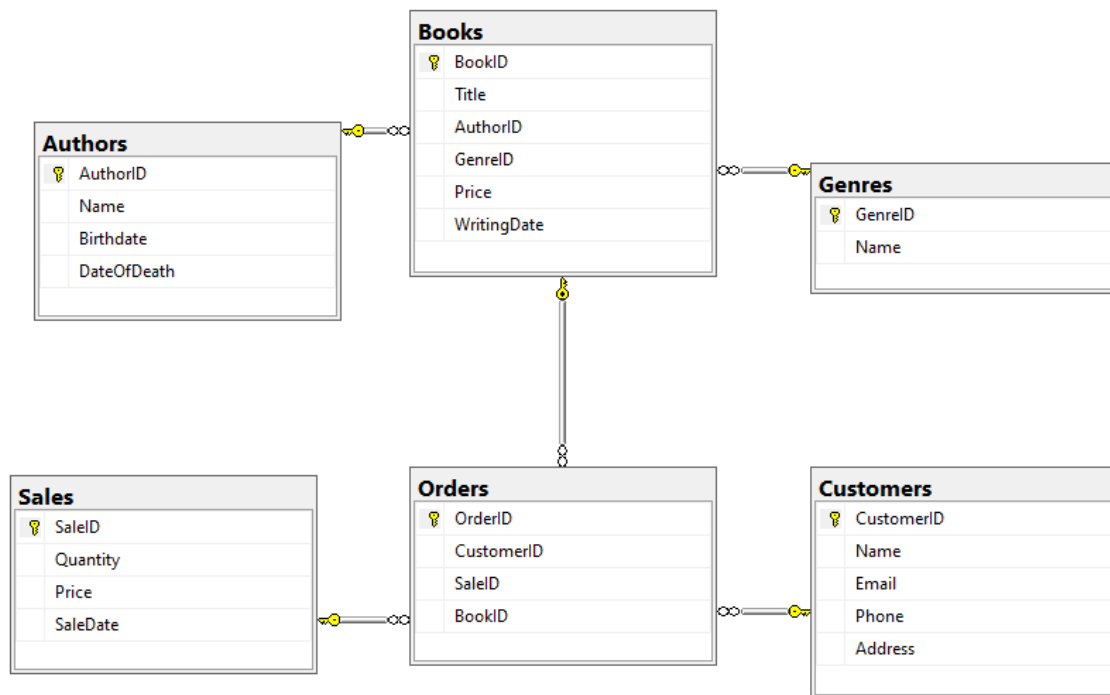
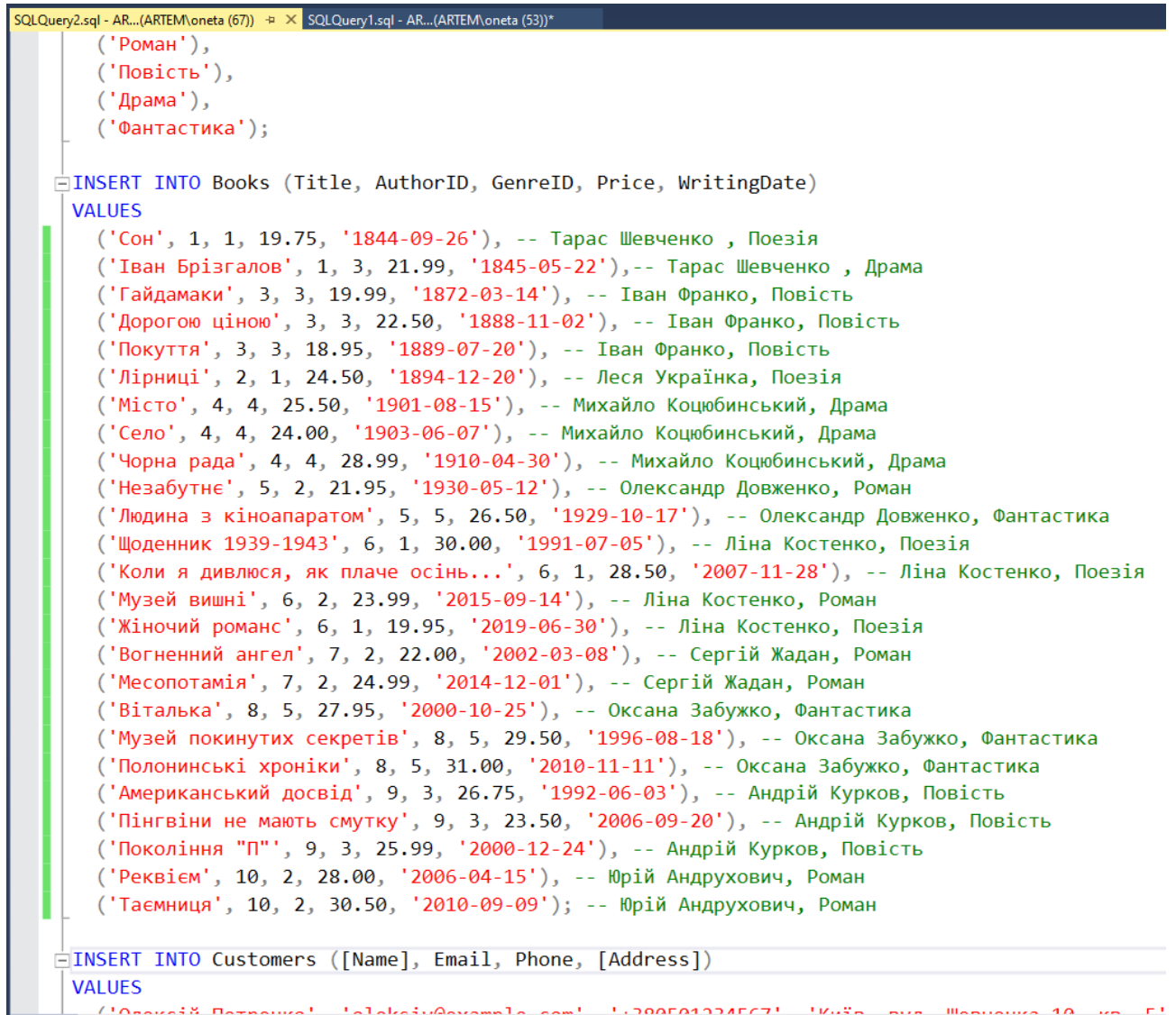


Рисунок 3.3 – Створення діаграми бази даних «BookstoreManagement»

### 3.4 Заповнення таблиць бази даних даними

Для написання програмного коду в SQL Server ManagementStudio потрібно натиснути кнопку «Створити запит» («Newquery») на панелі інструментів «Стандартна» («Standart»).



```
SQLQuery2.sql - AR... (ARTEM\oneta (67))  SQLQuery1.sql - AR... (ARTEM\oneta (53))

('Роман'),
('Повість'),
('Драма'),
('Фантастика');

INSERT INTO Books (Title, AuthorID, GenreID, Price, WritingDate)
VALUES
('Сон', 1, 1, 19.75, '1844-09-26'), -- Тарас Шевченко , Поезія
('Іван Брізгалов', 1, 3, 21.99, '1845-05-22'), -- Тарас Шевченко , Драма
('Гайдамаки', 3, 3, 19.99, '1872-03-14'), -- Іван Франко, Повість
('Дорогою ціною', 3, 3, 22.50, '1888-11-02'), -- Іван Франко, Повість
('Покуття', 3, 3, 18.95, '1889-07-20'), -- Іван Франко, Повість
('Лірниці', 2, 1, 24.50, '1894-12-20'), -- Леся Українка, Поезія
('Місто', 4, 4, 25.50, '1901-08-15'), -- Михайло Коцюбинський, Драма
('Село', 4, 4, 24.00, '1903-06-07'), -- Михайло Коцюбинський, Драма
('Чорна рада', 4, 4, 28.99, '1910-04-30'), -- Михайло Коцюбинський, Драма
('Незабутнє', 5, 2, 21.95, '1930-05-12'), -- Олександр Довженко, Роман
('Людина з кіноапаратом', 5, 5, 26.50, '1929-10-17'), -- Олександр Довженко, Фантастика
('Щоденник 1939-1943', 6, 1, 30.00, '1991-07-05'), -- Ліна Костенко, Поезія
('Коли я дивлюся, як плаче осінь...', 6, 1, 28.50, '2007-11-28'), -- Ліна Костенко, Поезія
('Музей вишні', 6, 2, 23.99, '2015-09-14'), -- Ліна Костенко, Роман
('Жіночий романс', 6, 1, 19.95, '2019-06-30'), -- Ліна Костенко, Поезія
('Вогненний ангел', 7, 2, 22.00, '2002-03-08'), -- Сергій Жадан, Роман
('Месопотамія', 7, 2, 24.99, '2014-12-01'), -- Сергій Жадан, Роман
('Віталька', 8, 5, 27.95, '2000-10-25'), -- Оксана Забужко, Фантастика
('Музей покинутих секретів', 8, 5, 29.50, '1996-08-18'), -- Оксана Забужко, Фантастика
('Полонинські хроніки', 8, 5, 31.00, '2010-11-11'), -- Оксана Забужко, Фантастика
('Американський досвід', 9, 3, 26.75, '1992-06-03'), -- Андрій Курков, Повість
('Пінгвіни не мають смутку', 9, 3, 23.50, '2006-09-20'), -- Андрій Курков, Повість
('Покоління "П"', 9, 3, 25.99, '2000-12-24'), -- Андрій Курков, Повість
('Реквієм', 10, 2, 28.00, '2006-04-15'), -- Юрій Андрухович, Роман
('Таємниця', 10, 2, 30.50, '2010-09-09'); -- Юрій Андрухович, Роман

INSERT INTO Customers ([Name], Email, Phone, [Address])
VALUES
('Олександр Петрович', 'olalekai@gmail.com', '+380501334567', 'Київ, вул. Шевченка 10, кв. 5')
```

Рисунок 3.4 – Вставка даних до таблиці Книги (Books)

Далі заповнюємо інші таблиці бази даних (Додаток Б - Заповнення таблиць БД).

## **4 ОРГАНІЗАЦІЯ БАГАТОКОРИСТУВАЦЬКОГО РЕЖИМУ ДОСТУПУ ДО БАЗИ ДАНИХ**

Багатокористувацький доступ у базі даних Microsoft SQL Server є фундаментальним аспектом її функціонування. Він дозволяє ефективно управляти доступом до даних та забезпечує безпеку і цілісність інформації. Цей доступ організований на трьох рівнях, кожен з яких має свою важливу роль у забезпеченні правильного функціонування системи.

Перший рівень - це облікові записи SQL Server. Кожен користувач має свій унікальний обліковий запис, який використовується для аутентифікації при підключенні до сервера баз даних. Ці облікові записи мають різні ролі і привілеї, які визначають рівень доступу до даних та можливості виконання операцій з ними.

Другий рівень - це ролі бази даних. Ролі дозволяють групувати користувачів з однаковими потребами щодо доступу до даних і надавати їм однакові права. Наприклад, роль "адміністратор" може мати повний доступ до всіх об'єктів бази даних, тоді як роль "користувач" може мати обмежений доступ лише для читання даних.

Третій рівень - це управління дозволами на об'єкти бази даних. Це означає, що адміністратор бази даних може явно вказати, які користувачі або ролі мають доступ до певних таблиць, представлень або інших об'єктів бази даних, і які операції вони можуть виконувати.

У цілому, багатокористувацький доступ у базі даних Microsoft SQL Server дозволяє ефективно керувати доступом до даних та забезпечує їх безпеку і цілісність. Це важливий елемент будь-якої бази даних, який дозволяє організаціям ефективно управляти своїми ресурсами та забезпечувати конфіденційність інформації.

Створення користувачів AdminUser та RegularUser:

```
CREATE LOGIN AdminUser WITH PASSWORD = 'AdminPassword';  
CREATE LOGIN RegularUser WITH PASSWORD = 'UserPassword';
```

Створення Ролей:

```
CREATE ROLE AdminRole;  
CREATE ROLE UserRole;
```

Призначення ролей відповідним користувачам:

```
ALTER ROLE AdminRole ADD MEMBER AdminUser;  
ALTER ROLE UserRole ADD MEMBER RegularUser;
```

Надання адміністратору повних прав доступу :

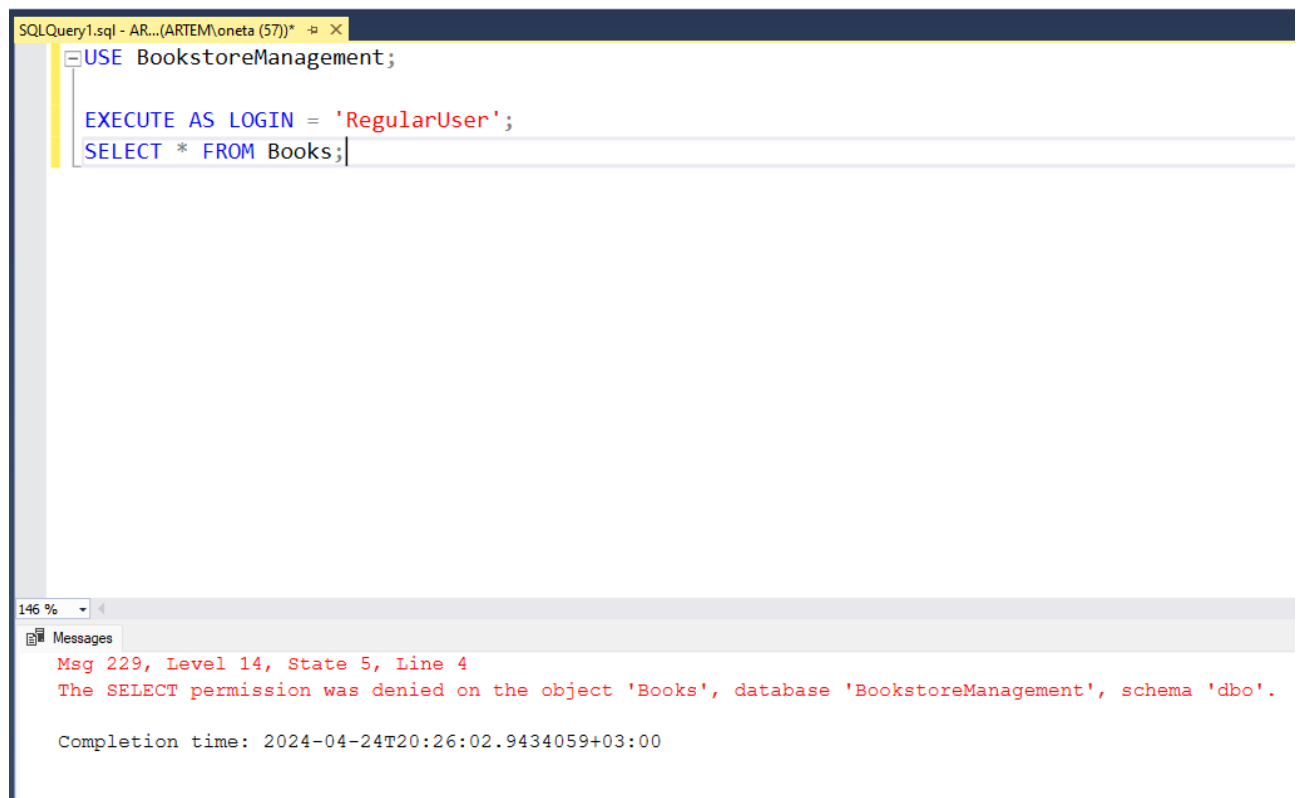
```
GRANT CONTROL ON Authors TO AdminUser;  
GRANT CONTROL ON Genres TO AdminUser;  
GRANT CONTROL ON Books TO AdminUser;  
GRANT CONTROL ON Customers TO AdminUser;  
GRANT CONTROL ON Sales TO AdminUser;  
GRANT CONTROL ON Orders TO AdminUser;
```

Надання вибірових прав доступу для користувача:

```
GRANT SELECT ON Authors TO RegularUser;  
GRANT SELECT ON Genres TO RegularUser;  
GRANT SELECT ON Customers TO RegularUser;  
GRANT SELECT, INSERT, UPDATE ON Sales TO RegularUser;  
GRANT SELECT, INSERT ON Orders TO RegularUser;
```



## Перевірка працездатності ролі користувача:



The screenshot displays the SQL Query window in SQL Server Enterprise Manager. The query being executed is:

```
USE BookstoreManagement;  
  
EXECUTE AS LOGIN = 'RegularUser';  
SELECT * FROM Books;
```

Below the query window, the Messages pane shows the following error:

```
Msg 229, Level 14, State 5, Line 4  
The SELECT permission was denied on the object 'Books', database 'BookstoreManagement', schema 'dbo'.  
  
Completion time: 2024-04-24T20:26:02.9434059+03:00
```

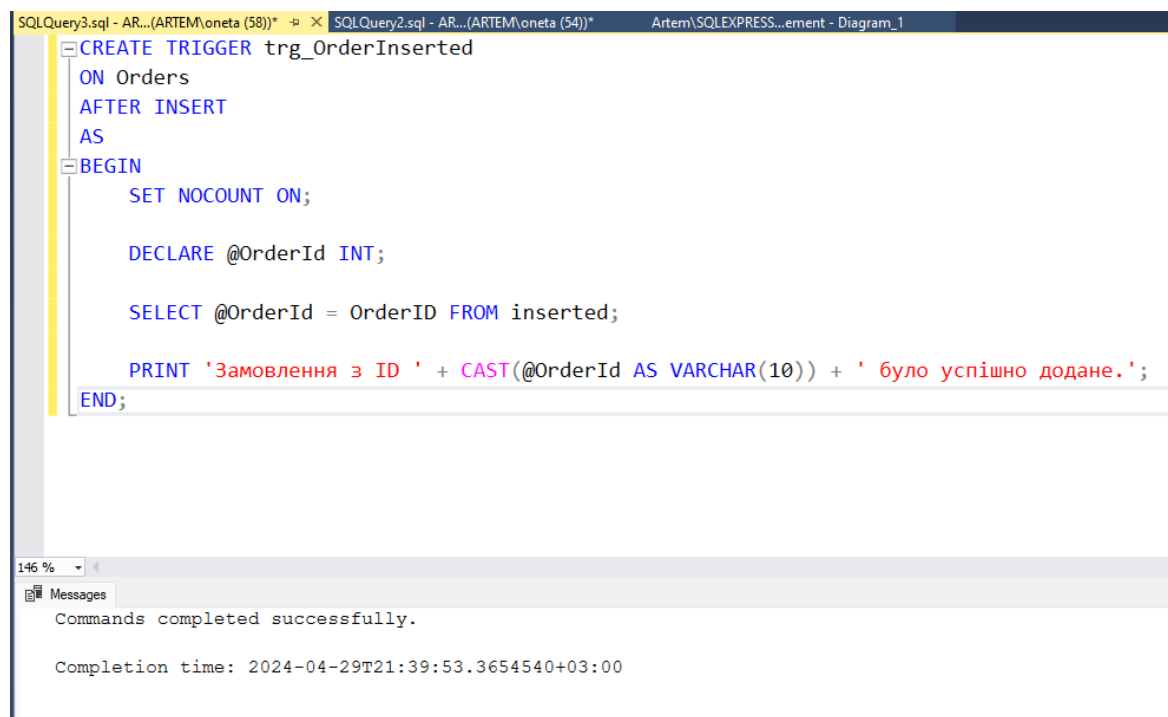
Рисунок 4.1 – спроба (RegularUser) зробити вибірку в таблиці (Books)

## 5 ТРИГЕРИ БАЗИ ДАНИХ

### Створення тригерів

Тригер (trigger) – це спеціальний тип збережених процедур, що запускаються сервером автоматично при виконанні тих чи інших дій з даними таблиці. Кожен тригер прив'язується до конкретної таблиці. Коли користувач намагається, наприклад, змінити дані в таблиці, сервер автоматично запускає тригер і, тільки якщо він завершується успішно, дозволяється виконання змін. Всі здійснені тригером модифікації даних розглядаються як одна транзакція. У разі виявлення помилки або порушення цілісності даних відбувається відкат цієї транзакції. Тим самим внесення змін буде заборонено. Будуть також відмінені всі зміни, вже зроблені тригером.

Створимо для таблиці Замовлення (Orders) тригер, , для виводу повідомлення у консоль про успішну вставку в таблицю .



```
CREATE TRIGGER trg_OrderInserted
ON Orders
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @OrderId INT;

    SELECT @OrderId = OrderID FROM inserted;

    PRINT 'Замовлення з ID ' + CAST(@OrderId AS VARCHAR(10)) + ' було успішно додане.';
END;
```

Messages

Commands completed successfully.

Completion time: 2024-04-29T21:39:53.3654540+03:00

Рисунок 5.1 – Тригер, для виводу повідомлення у консоль про успішну вставку в таблицю .

Цей тригер спрацьовуватиме кожного разу після того, як ви вставите новий запис у таблицю Orders і виведе повідомлення у консоль про успішну вставку замовлення разом із його ідентифікатором.

Перевіримо працездатність тригера:

Додамо до таблиці, замовлення. Для виконання команди натискаємо F5.

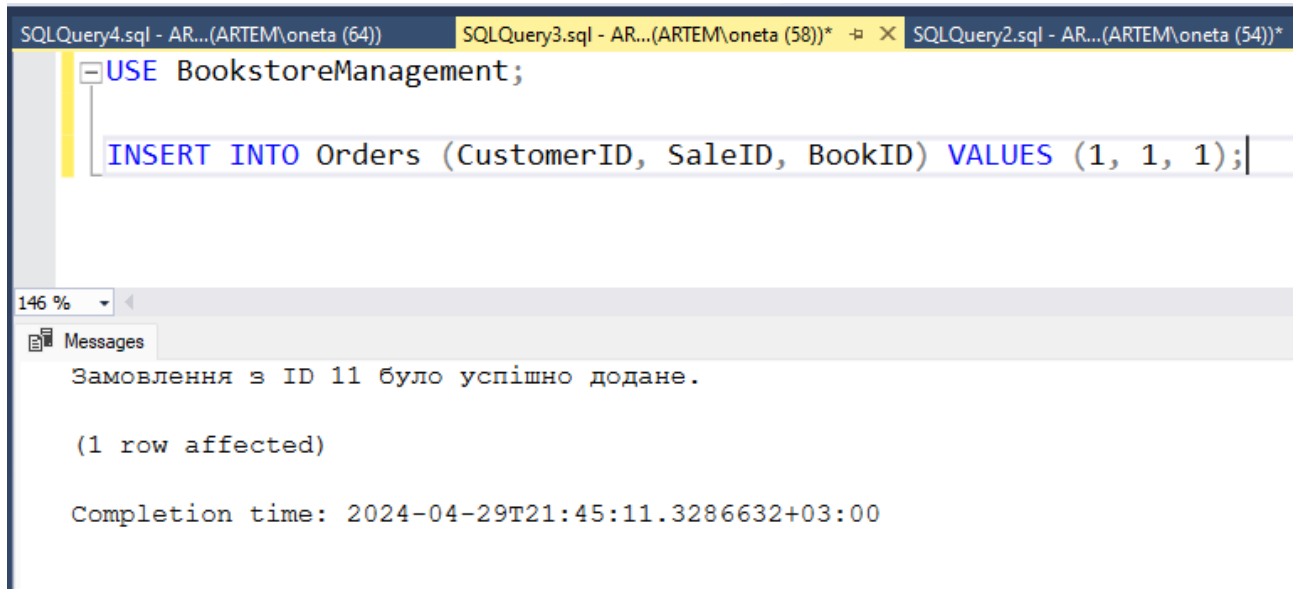


Рисунок 5.2 – Результат роботи тригера

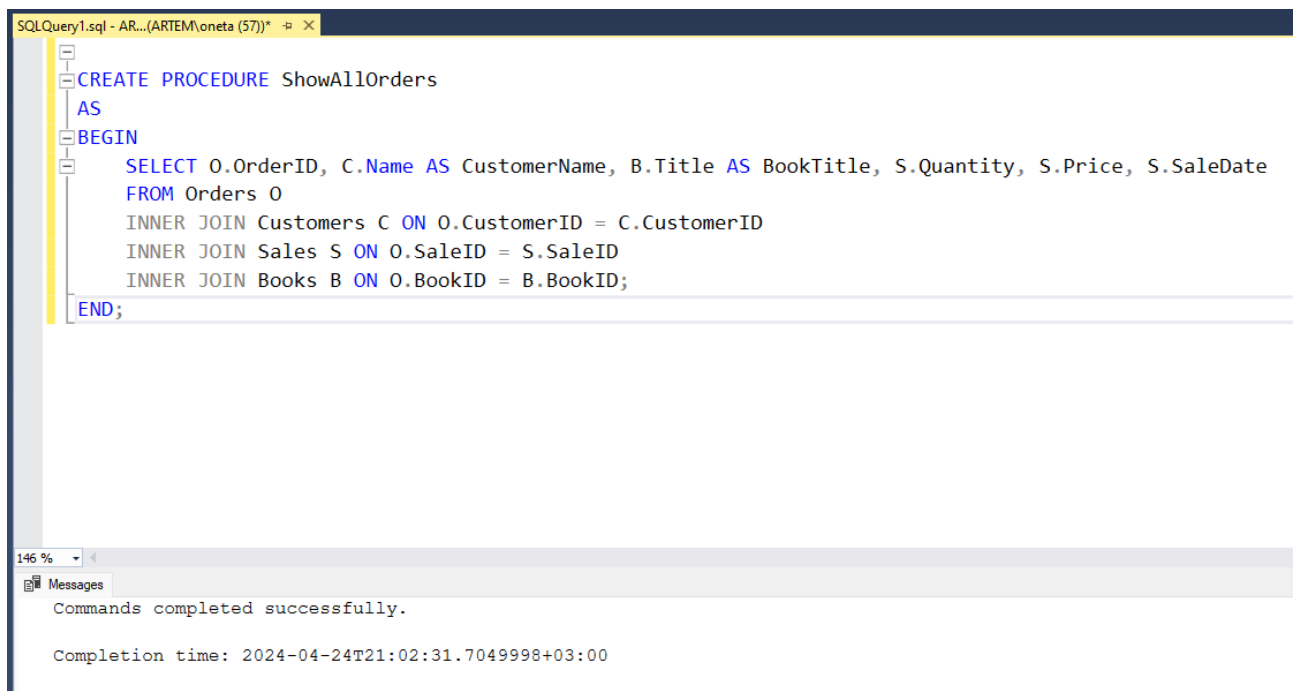
Далі створюємо тригери для інших таблиць (Додаток В – Створення тригерів).

## 6 ЗБЕРЕЖЕНІ ПРОЦЕДУРИ

### 6.1 Розроблення процедур

Збережена процедура (stored procedure) – це іменований набір команд Transact-SQL, що зберігається безпосередньо на сервері і представляє собою самостійний об'єкт – базу даних. Вона існує незалежно від таблиць або яких-небудь інших об'єктів баз даних. Збережена процедура може бути викликана клієнтською програмою, іншою збереженою процедурою або тригером.

Розробимо процедуру для відображення всіх замовлень та даних про них :



```
SQLQuery1.sql - AR... (ARTEM\oneta (57)) *  
--  
CREATE PROCEDURE ShowAllOrders  
AS  
BEGIN  
    SELECT O.OrderID, C.Name AS CustomerName, B.Title AS BookTitle, S.Quantity, S.Price, S.SaleDate  
    FROM Orders O  
    INNER JOIN Customers C ON O.CustomerID = C.CustomerID  
    INNER JOIN Sales S ON O.SaleID = S.SaleID  
    INNER JOIN Books B ON O.BookID = B.BookID;  
END;
```

146 %  
Messages  
Commands completed successfully.  
Completion time: 2024-04-24T21:02:31.7049998+03:00

Рисунок 6.1 – Процедура відображення всіх замовлень та даних про них

SQLQuery1.sql - AR...(ARTEM\oneta (60))\* -p X

```
USE BookstoreManagement;

EXEC ShowAllOrders;
```

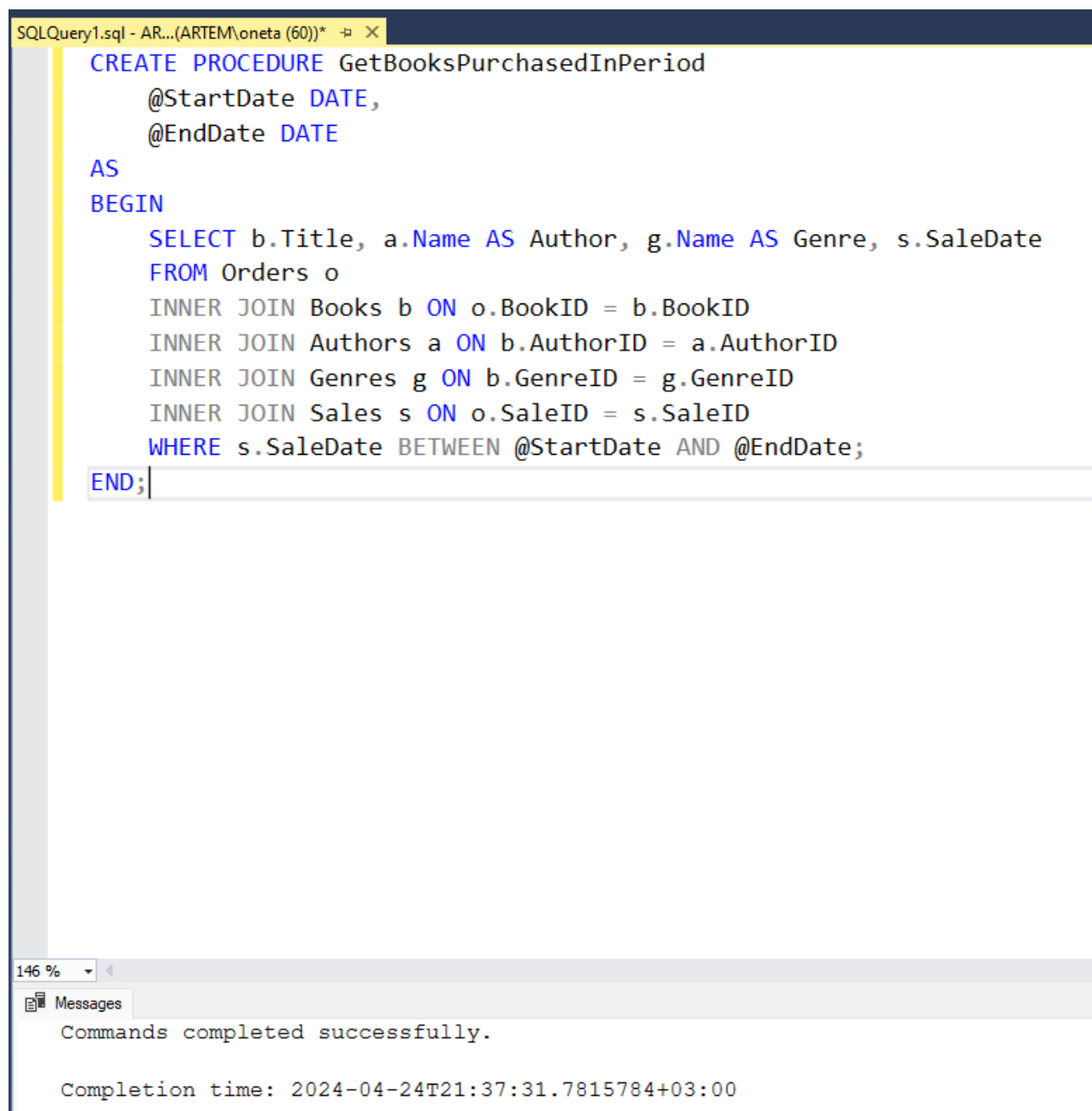
146 %

Results Messages

	OrderID	CustomerName	Book Title	Quantity	Price	SaleDate
1	1	Олексій Петренко	Сон	2	25.00	2024-04-15
2	2	Наталія Іванова	Іван Брізгалов	3	35.50	2024-04-16
3	3	Віталій Сидоренко	Гайдамаки	1	15.75	2024-04-17
4	4	Ірина Ковальчук	Дорогою ціною	4	45.25	2024-04-18
5	5	Михайло Попов	Покуття	2	22.00	2024-04-19
6	6	Ольга Семенова	Лірниці	2	25.00	2024-04-15
7	7	Ігор Кравченко	Місто	3	35.50	2024-04-16
8	8	Марія Григорович	Село	1	15.75	2024-04-17
9	9	Андрій Бондаренко	Чорна рада	4	45.25	2024-04-18
10	10	Тетяна Мельник	Незабутнє	2	22.00	2024-04-19

Рисунок 6.2 – Виклик процедури (ShowAllOrders)

Розробимо процедуру для відображення інформації про книги , куплені в певному періоді :



The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows the definition of a stored procedure named `GetBooksPurchasedInPeriod`. The procedure takes two parameters: `@StartDate DATE` and `@EndDate DATE`. The body of the procedure uses a `SELECT` statement to retrieve book titles, author names, genre names, and sale dates from the `Orders`, `Books`, `Authors`, `Genres`, and `Sales` tables. The results are filtered by the `WHERE` clause, which specifies that the `SaleDate` must fall between the `@StartDate` and `@EndDate`. The bottom pane shows the execution results, indicating that the commands were completed successfully and providing the completion time.

```
SQLQuery1.sql - AR...(ARTEM\oneta (60))* -> X
CREATE PROCEDURE GetBooksPurchasedInPeriod
    @StartDate DATE,
    @EndDate DATE
AS
BEGIN
    SELECT b.Title, a.Name AS Author, g.Name AS Genre, s.SaleDate
    FROM Orders o
    INNER JOIN Books b ON o.BookID = b.BookID
    INNER JOIN Authors a ON b.AuthorID = a.AuthorID
    INNER JOIN Genres g ON b.GenreID = g.GenreID
    INNER JOIN Sales s ON o.SaleID = s.SaleID
    WHERE s.SaleDate BETWEEN @StartDate AND @EndDate;
END;
```

146 %

Messages

Commands completed successfully.

Completion time: 2024-04-24T21:37:31.7815784+03:00

Рисунок 6.3 – процедура відображення інформації про книги , куплені в певному періоді

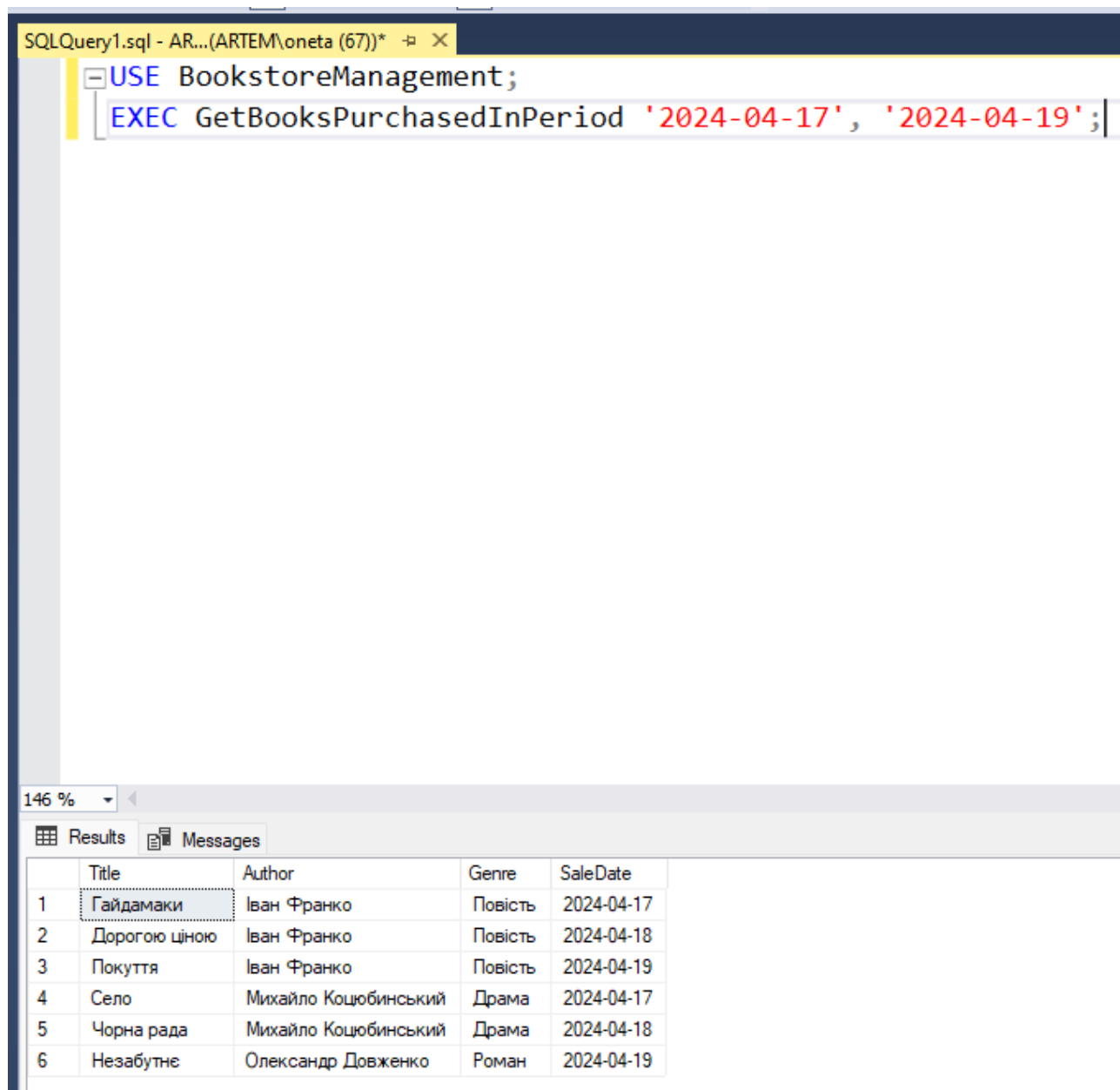


Рисунок 6.4 – Виклик процедури (GetBooksPurchasedInPeriod)

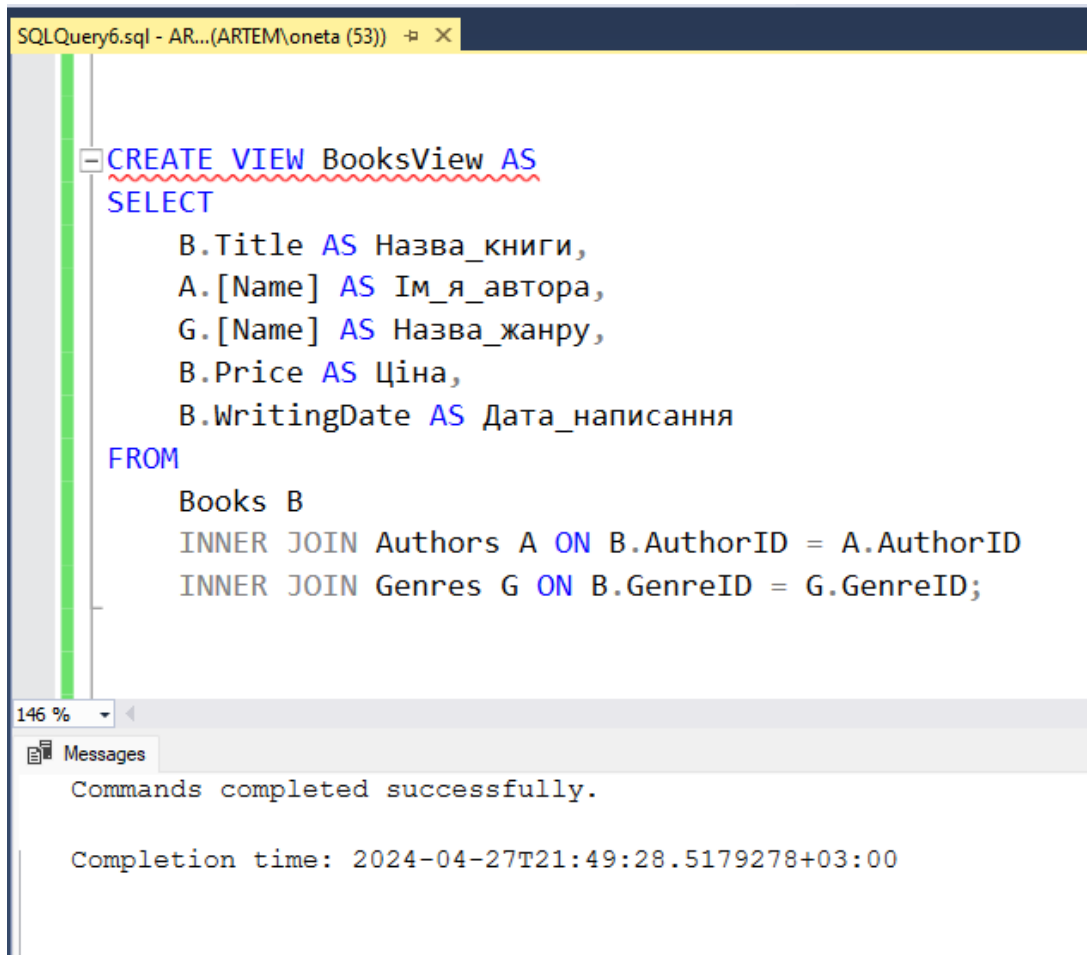
Далі створюємо процедури для інших таблиць (Додаток Г – Створення процедур).

## 7 ПРЕДСТАВЛЕННЯ БАЗИ ДАНИХ

### 7.Створення представлень

Представлення дозволяють повертати набори даних, що задовольняють потреби конкретних користувачів або груп. Після створення уявлення до них можна звертатися точно так само, як і до звичайної таблиці. Уявлення може будуватися на базі однієї або декількох таблиць, або навіть на основі інших уявлень.

Створюємо представлення для таблиці Книги (Books):



```
SQLQuery6.sql - AR...(ARTEM\oneta (53)) - + X
CREATE VIEW BooksView AS
SELECT
    B.Title AS Назва_книги,
    A.[Name] AS Ім_я_автора,
    G.[Name] AS Назва_жанру,
    B.Price AS Ціна,
    B.WritingDate AS Дата_написання
FROM
    Books B
    INNER JOIN Authors A ON B.AuthorID = A.AuthorID
    INNER JOIN Genres G ON B.GenreID = G.GenreID;

146 %
Messages
Commands completed successfully.

Completion time: 2024-04-27T21:49:28.5179278+03:00
```

Рисунок 7.1 – Створення представлення для таблиці Книги (Books)

Переглянемо уявлення:



SQLQuery2.sql - AR...(ARTEM\oneta (54))\* Artem\SQLEXPRESS...ement - Diagram\_1

```
USE BookstoreManagement;
```

```
SELECT * FROM BooksView;
```

146 %

Results Messages

	Назва_книги	Ім_я_автора	Назва_жанру	Ціна	Дата_написання
1	Сон	Тарас Шевченко	Поезія	19.75	1840-09-26
2	Іван Брізгалов	Тарас Шевченко	Повість	21.99	1845-05-22
3	Гайдамаки	Іван Франко	Повість	19.99	1872-03-14
4	Дорогою ціною	Іван Франко	Повість	22.50	1888-11-02
5	Покуття	Іван Франко	Повість	18.95	1889-07-20
6	Лірниці	Леся Українка	Поезія	24.50	1894-12-20
7	Місто	Михайло Коцюбинський	Драма	25.50	1901-08-15
8	Село	Михайло Коцюбинський	Драма	24.00	1903-06-07
9	Чорна рада	Михайло Коцюбинський	Драма	28.99	1910-04-30
10	Незабутнє	Олександр Довженко	Роман	21.95	1930-05-12
11	Людина з кіноапаратом	Олександр Довженко	Фантастика	26.50	1929-10-17
12	Щоденник 1939-1943	Ліна Костенко	Поезія	30.00	1991-07-05
13	Коли я дивлюся, як плаче осінь...	Ліна Костенко	Поезія	28.50	2007-11-28
14	Музей вишні	Ліна Костенко	Роман	23.99	2015-09-14
15	Жіночий романс	Ліна Костенко	Поезія	19.95	2019-06-30
16	Вогненний ангел	Сергій Жадан	Роман	22.00	2002-03-08
17	Месопотамія	Сергій Жадан	Роман	24.99	2014-12-01
18	Віталька	Оксана Забужко	Фантастика	27.95	2000-10-25
19	Музей покинутих секретів	Оксана Забужко	Фантастика	29.50	1996-08-18
20	Полонинські хроніки	Оксана Забужко	Фантастика	31.00	2010-11-11
21	Американський досвід	Андрій Курков	Повість	26.75	1992-06-03
22	Пінгвіни не мають смутку	Андрій Курков	Повість	23.50	2006-09-20
23	Покоління "П"	Андрій Курков	Повість	25.99	2000-12-24
24	Реквієм	Юрій Андрухович	Роман	28.00	2006-04-15
25	Таємниця	Юрій Андрухович	Роман	30.50	2010-09-09

Рисунок 7.2 – Представлення для таблиці Книги(Books)

Далі створюємо уявлення для інших таблиць (Додаток Д – Створення представлень).

## **8 СТВОРЕННЯ КЛІЄНТСЬКОЇ ЧАСТИНИ ПРОГРАМИ ДЛЯ РОБОТИ З БД**

### **Створення клієнтської частини програми для роботи з базою даних**

Під час розробки клієнтського додатку для системи управління книгарнею була використана технологія Windows Forms. Windows Forms - це фреймворк для створення клієнтських додатків для операційної системи Microsoft Windows.

У додатку ми обрали техніку програмування за патерном MVC (Model-View-Controller), що дозволяє нам ефективно розділити логіку додатку на три основні компоненти: Модель (Model), Вид (View) та Контролер (Controller). Цей підхід сприяє збереженню чистоти коду, полегшує розробку та робить додаток більш масштабованим.

Вид (View) - це інтерфейс користувача, який відображає дані користувачу та дозволяє йому взаємодіяти з додатком. У додатку ми використовуємо Windows Forms для створення різноманітних екранів, таких як екран списку авторів, книг, клієнтів та інших.

Модель (Model) - це компонент, який представляє дані та логіку їх обробки. У додатку модель відповідає за роботу з базою даних, виконання запитів та збереження даних.

Контролер (Controller) - це посередник між видом та моделлю, який обробляє вхідні дані від користувача та виконує відповідні дії. У додатку контролер відповідає за обробку подій користувача, таких як натискання кнопок або введення тексту.

Нижче наведені зразки екранів нашого додатку:

- Екран списку авторів: Цей екран відображає список всіх авторів книг у системі разом з їхніми даними, такими як ім'я та дата народження.
- Екран списку книг: На цьому екрані можна переглядати список усіх книг у системі разом з інформацією про назву, автора, жанр та ціну.
- Екран списку жанрів: Цей екран показує список усіх жанрів книг у системі разом з описом кожного жанру.
- Екран списку клієнтів: Цей екран показує список усіх клієнтів книгарні, включаючи їхнє ім'я, контактні дані та адресу.
- Екран замовлення книг: На цьому екрані користувач може робити замовлення книг, вибираючи книги зі списку та вказуючи кількість кожної книги.
- Екран списку продажів: На цьому екрані можна переглянути список усіх продажів у системі разом з інформацією про клієнта, куплені книги, кількість та ціну. Кожен запис представляє окрему угоду про продаж книг та включає такі дані, як ім'я клієнта, назва книги, кількість проданих екземплярів та сума грошей, отриманих в результаті продажу. Цей екран надає можливість аналізувати продажі та відстежувати динаміку руху товарів у вашій книгарні.

Екранні форми представлені на рисунках 8.1-8.7.

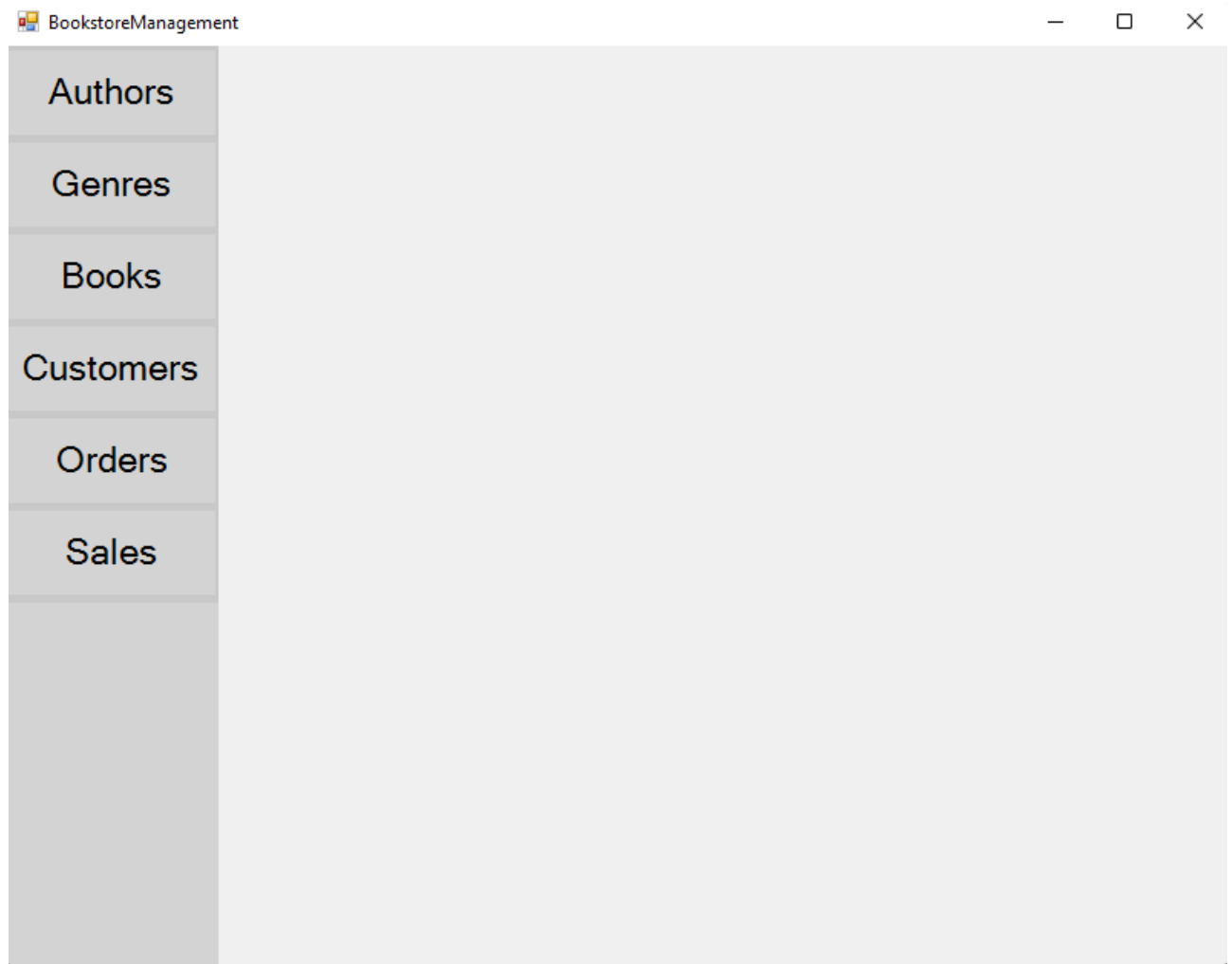


Рисунок 8.1 – Головна форма програми

BookstoreManagement			
Authors	Ім_я_автора	Дата_народження	Дата_смерті
	Тарас Шевченко	09.03.1814	10.03.1861
Genres	Леся Українка	25.02.1871	01.08.1913
	Іван Франко	27.08.1856	28.05.1916
	Михайло Коцюбинський	17.09.1864	25.08.1913
Books	Олександр Довженко	10.09.1894	25.11.1956
	Ліна Костенко	19.03.1930	
	Сергій Жадан	23.08.1974	
Customers	Оксана Забужко	19.09.1960	
	Андрій Курков	23.04.1961	
	Юрій Андрухович	13.03.1960	
Orders			
Sales			

Рисунок 8.2 – Екрана форма «Автори»

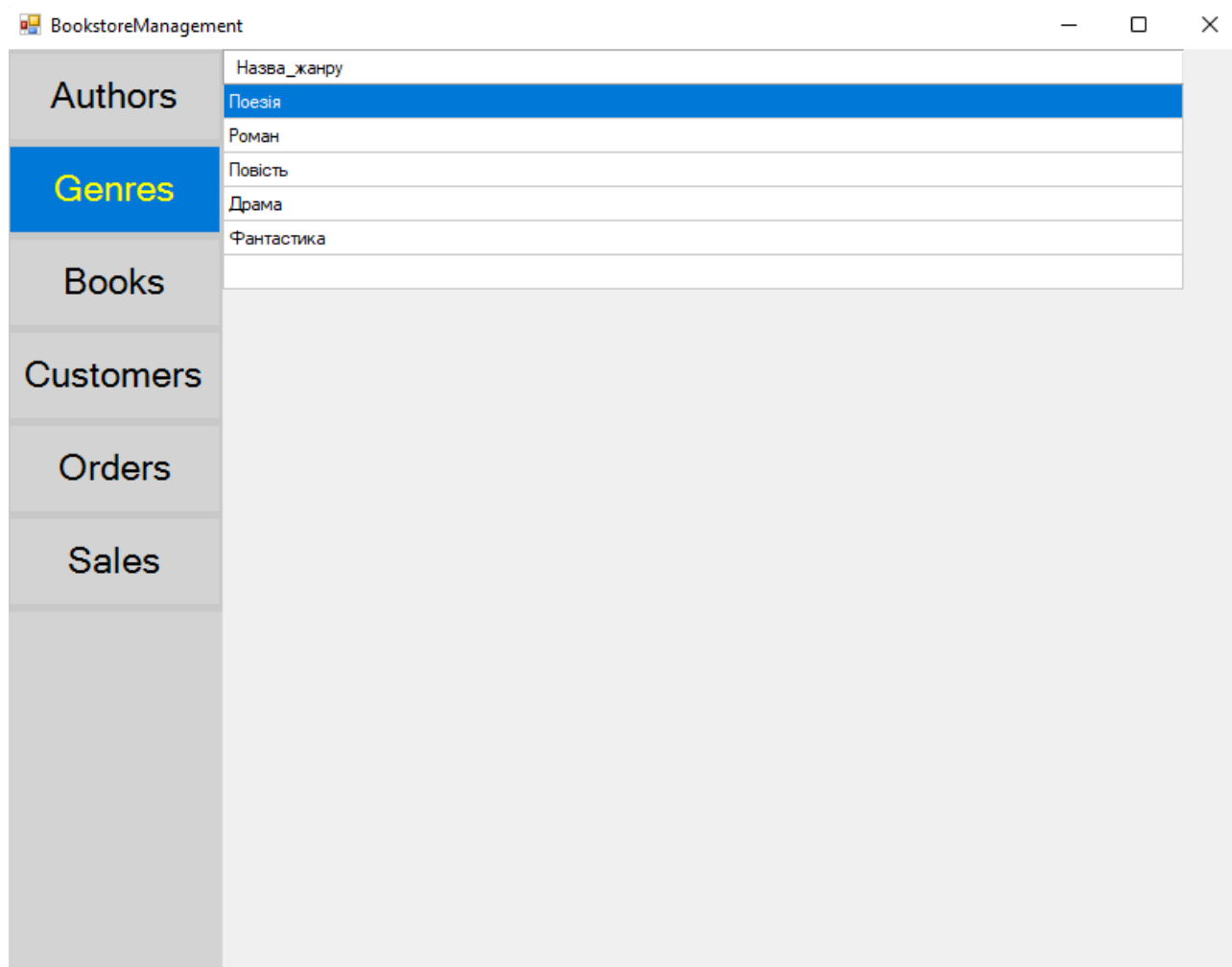


Рисунок 8.3 – Екрана форма «Жанри»

BookstoreManagement					
Authors	Назва_книги	Ім_я_автора	Назва_жанру	Ціна	Дата_написання
	Сон	Тарас Шевченко	Поезія	19,75	26.09.1840
Genres	Іван Брізгалов	Тарас Шевченко	Повість	25,50	22.05.1845
	Гайдамаки	Іван Франко	Повість	19,99	14.03.1872
	Дорогою ціною	Іван Франко	Повість	28,99	02.11.1888
Books	Покуття	Іван Франко	Повість	18,95	20.07.1889
	Лірниці	Леся Українка	Поезія	19,75	20.12.1894
	Місто	Михайло Коцюбинсь...	Драма	25,50	15.08.1901
Customers	Село	Михайло Коцюбинсь...	Драма	19,99	07.06.1903
	Чорна рада	Михайло Коцюбинсь...	Драма	28,99	30.04.1910
	Незабутнє	Олександр Довженко	Роман	18,95	12.05.1930
Orders	Людина з кіноапара...	Олександр Довженко	Фантастика	26,50	17.10.1929
	Щоденник 1939-1943	Ліна Костенко	Поезія	30,00	05.07.1991
	Коли я дивлюся, як ...	Ліна Костенко	Поезія	28,50	28.11.2007
Sales	Музей вишні	Ліна Костенко	Роман	23,99	14.09.2015
	Жіночий романс	Ліна Костенко	Поезія	19,95	30.06.2019
	Вогненний ангел	Сергій Жадан	Роман	22,00	08.03.2002
	Месопотамія	Сергій Жадан	Роман	24,99	01.12.2014
	Віталька	Оксана Забужко	Фантастика	27,95	25.10.2000
	Музей покинутих се...	Оксана Забужко	Фантастика	29,50	18.08.1996
	Полонинські хроніки	Оксана Забужко	Фантастика	31,00	11.11.2010
	Американський дос...	Андрій Курков	Повість	26,75	03.06.1992
	Пінгвіни не мають с...	Андрій Курков	Повість	23,50	20.09.2006
	Покоління "П"	Андрій Курков	Повість	25,99	24.12.2000
	Реквієм	Юрій Андрухович	Роман	28,00	15.04.2006
	Таємниця	Юрій Андрухович	Роман	30,50	09.09.2010

Рисунок 8.4 – Екрана форма «Книги»

BookstoreManagement				
Authors	Ім'я_клієнта	Електронна_пошта	Телефон	Адреса
	Олексій Петренко	oleksiy@example.com	+380501234567	Київ, вул. Шевченка 10, кв. 5
Genres	Наталія Іванова	nataliya@example.com	+380501234568	Львів, пр. Героїв 25, кв. 12
	Віталій Сидоренко	vitaliy@example.com	+380501234569	Харків, вул. Лесі Українки ...
Books	Ірина Ковальчук	iryna@example.com	+380501234570	Одеса, вул. Івана Франка ...
	Михайло Попов	mikhailo@example.com	+380501234571	Дніпро, пр. Перемоги 42, к...
Customers	Ольга Семенова	olga@example.com	+380501234572	Харків, вул. Коцюбинськог...
	Ігор Кравченко	igor@example.com	+380501234573	Київ, вул. Гагаріна 11, кв. 9
Orders	Марія Григорович	maria@example.com	+380501234574	Львів, пр. Свободи 29, кв. 6
	Андрій Бондаренко	andriy@example.com	+380501234575	Дніпро, вул. Володимира В...
Sales	Тетяна Мельник	tetiana@example.com	+380501234576	Одеса, вул. Жуковського ...
	Павло Лисенко	pavlo@example.com	+380501234577	Київ, пр. Космонавтів 37, к...
	Катерина Тимошенко	kateryna@example.com	+380501234578	Львів, вул. Соборна 8, кв. 11
	Іван Савченко	ivan@example.com	+380501234579	Харків, вул. Лісопильна 4, ...
	Надія Козлова	nadiya@example.com	+380501234580	Дніпро, вул. Шевченка 31, ...
	Володимир Гончаренко	volodymyr@example.com	+380501234581	Київ, пр. Шевченка 54, кв. 7
	Юлія Шевченко	yuliya@example.com	+380501234582	Одеса, вул. Героїв Майдан...
	Роман Коваль	roman@example.com	+380501234583	Львів, пр. Пiмназійний 67, ...
	Анастасія Біленька	anastasiya@example.com	+380501234584	Харків, вул. Степана Банд...
	Денис Мороз	denys@example.com	+380501234585	Київ, вул. Грушевського 2...
	Марина Волошина	marina@example.com	+380501234586	Дніпро, пр. Свободи 73, кв...

Рисунок 8.5 – Екрана форма «Клієнти»



BookstoreManagement			
Authors	Ім_я_клієнта	Дата_замовлення	Назва_книги
	Олексій Петренко	15.04.2024	Сон
	Наталія Іванова	16.04.2024	Іван Брізгалов
Genres	Віталій Сидоренко	17.04.2024	Гайдамаки
	Ірина Ковальчук	18.04.2024	Дорогою ціною
	Михайло Попов	19.04.2024	Покуття
Books	Ольга Семенова	15.04.2024	Лірниці
	Ігор Кравченко	16.04.2024	Місто
	Марія Григорович	17.04.2024	Село
Customers	Андрій Бондаренко	18.04.2024	Чорна рада
	Тетяна Мельник	19.04.2024	Незабутнє
Orders			
Sales			

Рисунок 8.6 – Екрана форма «Замовлення»

BookstoreManagement						
Authors	Ім_я_клієнта	Назва_книги	Кількість	Ціна	Дата_продажу	Сума_отримана
	Олексій Петренко	Сон	2	19,75	15.04.2024	39,50
	Наталія Іванова	Іван Брізгалов	3	25,50	16.04.2024	76,50
Genres	Віталій Сидоренко	Гайдамаки	1	19,99	17.04.2024	19,99
	Ірина Ковальчук	Дорогою ціною	4	28,99	18.04.2024	115,96
Books	Михайло Попов	Покуття	2	18,95	19.04.2024	37,90
	Ольга Семенова	Лірниці	2	19,75	15.04.2024	39,50
	Ігор Кравченко	Місто	3	25,50	16.04.2024	76,50
Customers	Марія Григорович	Село	1	19,99	17.04.2024	19,99
	Андрій Бондарен...	Чорна рада	4	28,99	18.04.2024	115,96
	Тетяна Мельник	Незабутнє	2	18,95	19.04.2024	37,90
Orders						
Sales						

Рисунок 8.7 – Екрана форма «Продажі»

Програмний код клієнтської частини наведено у додатку Е.

## ВИСНОВКИ

Метою курсової роботи було проектування бази даних «Облік отриманих коштів за продажі в книжковому магазині».

Для виконання курсової роботи були проведені всі необхідні дослідження, що стосуються розробки стратегії автоматизації, в результаті яких була надана відповідь на принципові запитання, що стосуються автоматизації доходів та обліку магазину.

Після цього була побудована концептуальна модель. Для цього була використана мова ER-опису предметної області, яка базується на концепції, що інформаційна модель будь-якої предметної області може бути описана із застосування таких понять, як сутність, атрибут, зв'язок. Крім того, ця мова є суттєво графічною, що дає можливість наочно представляти концептуальну модель предметної області. При побудові концептуальної моделі неявно використовувалися результати теорії нормалізації, у зв'язку з цим побудована модель представлена у третій нормальній формі. Необхідності використання більш високих нормальних форм не було, так як у предметній області не були виявлені складні види функціональних залежностей, а також багатозначні залежності.

Логічне та фізичне проектування бази даних складалося з конвертації концептуальної моделі предметної області у реляційну модель даних. При цьому був використаний алгоритм конвертування схеми предметної області у мові ER в схему реляційної бази даних.

Після цього реляційна база даних була представлена у вигляді команд створення таблиць бази даних у мові SQL. Крім того, у мові SQL описані деякі пошукові запити,

В даній курсовій роботі була розроблена база даних «Облік отриманих коштів за продажі в книжковому магазині» в системі управління базами даних Microsoft SQL Server.

Потужність бази даних обумовлена можливістю її постійного поповнення новими даними, причому в необмеженій кількості інформації. Це є дуже зручним для користувача. Також був створений клієнтський додаток а використанням вбудованих інструментів на Visual C# 2022, який працює з БД «Облік отриманих коштів за продажі в книжковому магазині» і дозволяє вести облік авторів, жанрів, книг, клієнтів, замовлень та продажів. Таким чином, створення бази даних «Облік отриманих коштів за продажі в книжковому магазині» є досить актуальним і корисним.

## ПЕРЕЛІК ПОСИЛАНЬ

1. SQL Tutorials: сайт URL: <https://www.w3schools.com/sql>
2. C# Tutorials: сайт URL: <https://www.w3schools.com/cs>
3. Documentation of MS SQL Server and Trasnact-SQL: сайт URL: <https://learn.microsoft.com/en-us/sql/sql-server>
4. Михальов О.І., Крамаренко В.В., Бистров Є.Є., Ялова К.М., Завгородній В.В. Довідник термінів та понять з методів проектування автоматизованих систем, баз даних і структур даних: ДДТУ, 2011.
5. Михальов О.І., Крамаренко В.В., Завгородній В.В., Михайловська Т.В. Організація баз даних: ДДТУ, 2010.
6. Ялова К.М., Завгородній В.В. Проектування автоматизованих інформаційних систем: ДДТУ, 2010.
7. Бабенко М.В., Яшина К.В. Бази даних: ДДТУ, 2020.

## ДОДАТОК А. Створення таблиць

```
USE BookstoreManagement;
```

```
CREATE TABLE Authors (  
    AuthorID INT PRIMARY KEY IDENTITY(1, 1),  
    [Name] NVARCHAR(100) NOT NULL,  
    Birthdate DATE NOT NULL,  
    DateOfDeath DATE  
);
```

```
CREATE TABLE Genres (  
    GenreID INT PRIMARY KEY IDENTITY(1, 1),  
    [Name] NVARCHAR(50) NOT NULL  
);
```

```
CREATE TABLE Books (  
    BookID INT PRIMARY KEY IDENTITY(1, 1),  
    Title NVARCHAR(100) NOT NULL,  
    AuthorID INT NOT NULL,  
    GenreID INT NOT NULL,  
    Price DECIMAL(10, 2) NOT NULL,  
    WritingDate DATE NOT NULL,  
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID),  
    FOREIGN KEY (GenreID) REFERENCES Genres(GenreID)  
);
```

```
CREATE TABLE Customers (  
    CustomerID INT PRIMARY KEY IDENTITY(1, 1),  
    [Name] NVARCHAR(100) NOT NULL,  
    Email NVARCHAR(100) NOT NULL,  
    Phone NVARCHAR(20) NOT NULL,  
    Address NVARCHAR(255) NOT NULL  
);
```

```
CREATE TABLE Sales (  
    SaleID INT PRIMARY KEY IDENTITY(1, 1),  
    Quantity INT NOT NULL,  
    Price DECIMAL(10, 2) NOT NULL,  
    SaleDate DATE NOT NULL  
);
```

```
CREATE TABLE Orders (  
    OrderID INT PRIMARY KEY IDENTITY(1, 1),  
    CustomerID INT NOT NULL,  
    SaleID INT NOT NULL,  
    BookID INT NOT NULL,  
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),  
    FOREIGN KEY (SaleID) REFERENCES Sales(SaleID),  
    FOREIGN KEY (BookID) REFERENCES Books(BookID)  
);
```

## ДОДАТОК Б. Заповнення таблиць

USE BookstoreManagement;

INSERT INTO Authors ([Name], Birthdate, DateOfDeath)  
VALUES

```
('Тарас Шевченко', '1814-03-09', '1861-03-10'),
('Леся Українка', '1871-02-25', '1913-08-01'),
('Іван Франко', '1856-08-27', '1916-05-28'),
('Михайло Коцюбинський', '1864-09-17', '1913-08-25'),
('Олександр Довженко', '1894-09-10', '1956-11-25'),
('Ліна Костенко', '1930-03-19', NULL),
('Сергій Жадан', '1974-08-23', NULL),
('Оксана Забужко', '1960-09-19', NULL),
('Андрій Курков', '1961-04-23', NULL),
('Юрій Андрухович', '1960-03-13', NULL);
```

INSERT INTO Genres ([Name])  
VALUES

```
('Поезія'),
('Роман'),
('Повість'),
('Драма'),
('Фантастика');
```

INSERT INTO Books (Title, AuthorID, GenreID, Price, WritingDate)  
VALUES

```
('Сон', 1, 1, 19.75, '1843-01-01'), -- Тарас Шевченко , Поезія
('Іван Брізгалов', 1, 3, 21.99, '1845-01-01'), -- Тарас Шевченко , Драма
('Гайдамаки', 3, 3, 19.99, '1872-01-01'), -- Іван Франко, Повість
('Дорогою ціною', 3, 3, 22.50, '1888-01-01'), -- Іван Франко, Повість
('Покуття', 3, 3, 18.95, '1889-01-01'), -- Іван Франко, Повість
('Лірниці', 2, 1, 24.50, '1894-01-01'), -- Леся Українка, Поезія
('Місто', 4, 4, 25.50, '1901-01-01'), -- Михайло Коцюбинський, Драма
('Село', 4, 4, 24.00, '1903-01-01'), -- Михайло Коцюбинський, Драма
('Чорна рада', 4, 4, 28.99, '1910-01-01'), -- Михайло Коцюбинський, Драма
('Незабутнє', 5, 2, 21.95, '1930-01-01'), -- Олександр Довженко, Роман
('Людина з кіноапаратом', 5, 5, 26.50, '1929-01-01'), -- Олександр Довженко, Фантастика
('Щоденник 1939-1943', 6, 1, 30.00, '1991-01-01'), -- Ліна Костенко, Поезія
('Коли я дивлюся, як плаче осінь...', 6, 1, 28.50, '2007-01-01'), -- Ліна Костенко, Поезія
('Музей вишні', 6, 2, 23.99, '2015-01-01'), -- Ліна Костенко, Роман
('Жіночий романс', 6, 1, 19.95, '2019-01-01'), -- Ліна Костенко, Поезія
('Вогненний ангел', 7, 2, 22.00, '2002-01-01'), -- Сергій Жадан, Роман
('Месопотамія', 7, 2, 24.99, '2014-01-01'), -- Сергій Жадан, Роман
('Віталька', 8, 5, 27.95, '2000-01-01'), -- Оксана Забужко, Фантастика
('Музей покинутих секретів', 8, 5, 29.50, '1996-01-01'), -- Оксана Забужко, Фантастика
('Полонинські хроніки', 8, 5, 31.00, '2010-01-01'), -- Оксана Забужко, Фантастика
('Американський досвід', 9, 3, 26.75, '1992-01-01'), -- Андрій Курков, Повість
('Пінгвіни не мають смутку', 9, 3, 23.50, '2006-01-01'), -- Андрій Курков, Повість
('Покоління "П"', 9, 3, 25.99, '2000-01-01'), -- Андрій Курков, Повість
('Реквієм', 10, 2, 28.00, '2006-01-01'), -- Юрій Андрухович, Роман
('Таємниця', 10, 2, 30.50, '2010-01-01'); -- Юрій Андрухович, Роман
```

INSERT INTO Customers ([Name], Email, Phone, [Address])  
VALUES

```
('Олексій Петренко', 'oleksiy@example.com', '+380501234567', 'Київ, вул. Шевченка 10, кв. 5'),
('Наталія Іванова', 'nataliya@example.com', '+380501234568', 'Львів, пр. Героїв 25, кв. 12'),
('Віталій Сидоренко', 'vitaliy@example.com', '+380501234569', 'Харків, вул. Лесі Українки 7, кв. 3'),
('Ірина Ковальчук', 'iryna@example.com', '+380501234570', 'Одеса, вул. Івана Франка 15, кв. 8'),
('Михайло Попов', 'mikhailo@example.com', '+380501234571', 'Дніпро, пр. Перемоги 42, кв. 21'),
('Ольга Семенова', 'olga@example.com', '+380501234572', 'Харків, вул. Коцюбинського 3, кв. 17'),
('Ігор Кравченко', 'igor@example.com', '+380501234573', 'Київ, вул. Гагаріна 11, кв. 9'),
```

```

('Марія Григорович', 'maria@example.com', '+380501234574', 'Львів, пр. Свободи 29, кв. 6'),
('Андрій Бондаренко', 'andriy@example.com', '+380501234575', 'Дніпро, вул. Володимира Великого
5, кв. 22'),
('Тетяна Мельник', 'tetiana@example.com', '+380501234576', 'Одеса, вул. Жуковського 12, кв.
14'),
('Павло Лисенко', 'pavlo@example.com', '+380501234577', 'Київ, пр. Космонавтів 37, кв. 19'),
('Катерина Тимошенко', 'kateryna@example.com', '+380501234578', 'Львів, вул. Соборна 8, кв.
11'),
('Іван Савченко', 'ivan@example.com', '+380501234579', 'Харків, вул. Лісопильна 4, кв. 23'),
('Надія Козлова', 'nadiya@example.com', '+380501234580', 'Дніпро, вул. Шевченка 31, кв. 2'),
('Володимир Гончаренко', 'volodymyr@example.com', '+380501234581', 'Київ, пр. Шевченка 54, кв.
7'),
('Юлія Шевченко', 'yuliya@example.com', '+380501234582', 'Одеса, вул. Героїв Майдану 16, кв.
28'),
('Роман Коваль', 'roman@example.com', '+380501234583', 'Львів, пр. Гімназійний 67, кв. 10'),
('Анастасія Біленька', 'anastasiya@example.com', '+380501234584', 'Харків, вул. Степана Бандери
9, кв. 15'),
('Денис Мороз', 'denys@example.com', '+380501234585', 'Київ, вул. Грушевського 22, кв. 18'),
('Марина Волошина', 'marina@example.com', '+380501234586', 'Дніпро, пр. Свободи 73, кв. 13');

```

```

INSERT INTO Sales (Quantity, Price, SaleDate)
VALUES

```

```

(2, 25.00, '2024-04-15'),
(3, 35.50, '2024-04-16'),
(1, 15.75, '2024-04-17'),
(4, 45.25, '2024-04-18'),
(2, 22.00, '2024-04-19');

```

```

INSERT INTO Orders (CustomerID, SaleID, BookID)
VALUES

```

```

(1, 1, 1),
(2, 2, 2),
(3, 3, 3),
(4, 4, 4),
(5, 5, 5),
(6, 1, 6),
(7, 2, 7),
(8, 3, 8),
(9, 4, 9),
(10, 5, 10);

```



## ДОДАТОК В. Створення тригерів

```
USE BookstoreManagement;

CREATE TRIGGER trg_DeleteAuthor
ON Authors
INSTEAD OF DELETE
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @AuthorID INT;

    SELECT @AuthorID = deleted.AuthorID FROM deleted;

    -- Перевірка, чи автор має книги
    IF EXISTS (SELECT 1 FROM Books WHERE AuthorID = @AuthorID)
    BEGIN
        PRINT 'Не можна видалити автора, який має книги.';
    END
    ELSE
    BEGIN
        DELETE FROM Authors WHERE AuthorID = @AuthorID;
    END
END;

CREATE TRIGGER trg_PreventInsertOrder
ON Orders
INSTEAD OF INSERT
AS
BEGIN
    SET NOCOUNT ON;

    IF EXISTS (
        SELECT 1
        FROM inserted i
        INNER JOIN Sales s ON i.SaleID = s.SaleID
        WHERE s.Quantity <= 0
    )
    BEGIN
        PRINT 'Неможливо здійснити замовлення, книга вже продана.';
    END
    ELSE
    BEGIN
        INSERT INTO Orders (CustomerID, SaleID, BookID)
        SELECT CustomerID, SaleID, BookID FROM inserted;
    END
END;

CREATE TRIGGER trg_OrderInserted
ON Orders
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @OrderId INT;

    SELECT @OrderId = OrderID FROM inserted;

    PRINT 'Замовлення з ID ' + CAST(@OrderId AS VARCHAR(10)) + ' було успішно додане.';
END;
```

## ДОДАТОК Г. Створення процедур

```
USE BookstoreManagement;
```

```
CREATE PROCEDURE ShowAllOrders
```

```
AS
```

```
BEGIN
```

```
    SELECT O.OrderID, C.Name AS CustomerName, B.Title AS BookTitle, S.Quantity, S.Price,  
    S.SaleDate
```

```
    FROM Orders O
```

```
    INNER JOIN Customers C ON O.CustomerID = C.CustomerID
```

```
    INNER JOIN Sales S ON O.SaleID = S.SaleID
```

```
    INNER JOIN Books B ON O.BookID = B.BookID;
```

```
END;
```

```
CREATE PROCEDURE GetBooksPurchasedInPeriod
```

```
    @StartDate DATE,
```

```
    @EndDate DATE
```

```
AS
```

```
BEGIN
```

```
    SELECT b.Title, a.Name AS Author, g.Name AS Genre, s.SaleDate
```

```
    FROM Orders o
```

```
    INNER JOIN Books b ON o.BookID = b.BookID
```

```
    INNER JOIN Authors a ON b.AuthorID = a.AuthorID
```

```
    INNER JOIN Genres g ON b.GenreID = g.GenreID
```

```
    INNER JOIN Sales s ON o.SaleID = s.SaleID
```

```
    WHERE s.SaleDate BETWEEN @StartDate AND @EndDate;
```

```
END;
```

```
CREATE PROCEDURE GetCustomerOrders
```

```
    @CustomerName NVARCHAR(100)
```

```
AS
```

```
BEGIN
```

```
    SELECT o.OrderID, c.Name AS CustomerName, b.Title AS BookTitle, s.SaleDate
```

```
    FROM Orders o
```

```
    INNER JOIN Customers c ON o.CustomerID = c.CustomerID
```

```
    INNER JOIN Sales s ON o.SaleID = s.SaleID
```

```
    INNER JOIN Books b ON o.BookID = b.BookID
```

```
    WHERE c.Name = @CustomerName;
```

```
END;
```

## ДОДАТОК Д. Створення представлень

```
--USE BookstoreManagement;
```

```
CREATE VIEW AuthorsView AS
SELECT
    [Name] AS Ім_я_автора,
    Birthdate AS Дата_народження,
    DateOfDeath AS Дата_смерті
FROM
    Authors;
```

```
CREATE VIEW GenresView AS
SELECT
    [Name] AS Назва_жанру
FROM
    Genres;
```

```
CREATE VIEW BooksView AS
SELECT
    B.Title AS Назва_книги,
    A.[Name] AS Ім_я_автора,
    G.[Name] AS Назва_жанру,
    B.Price AS Ціна,
    B.WritingDate AS Дата_написання
FROM
    Books B
    INNER JOIN Authors A ON B.AuthorID = A.AuthorID
    INNER JOIN Genres G ON B.GenreID = G.GenreID;
```

```
CREATE VIEW CustomersView AS
SELECT
    [Name] AS Ім_я_клієнта,
    Email AS Електронна_пошта,
    Phone AS Телефон,
    Address AS Адреса
FROM
    Customers;
```

```
CREATE VIEW SalesView AS
SELECT
    C.[Name] AS Ім_я_клієнта,
    B.Title AS Назва_книги,
    S.Quantity AS Кількість,
    S.Price AS Ціна,
    S.SaleDate AS Дата_продажу,
    (S.Quantity * S.Price) AS Сума_отримана
FROM
    Sales S
    INNER JOIN Orders O ON S.SaleID = O.SaleID
    INNER JOIN Customers C ON O.CustomerID = C.CustomerID
    INNER JOIN Books B ON O.BookID = B.BookID;
```

```
CREATE VIEW OrdersView AS
SELECT
    C.[Name] AS Ім_я_клієнта,
    S.SaleDate AS Дата_замовлення,
    B.Title AS Назва_книги
FROM
    Orders O
    INNER JOIN Customers C ON O.CustomerID = C.CustomerID
    INNER JOIN Sales S ON O.SaleID = S.SaleID
    INNER JOIN Books B ON O.BookID = B.BookID;
```

## ДОДАТОК Е. Програмний код

### MainForm.cs

```
using System;
using System.Data;
using System.Drawing;
using System.Windows.Forms;

namespace BookstoreManagementApp
{
    public partial class MainForm : Form
    {
        private DataGridView dataGridView;
        private ListBox viewListBox;

        public MainForm()
        {
            InitializeComponent();
            this.ClientSize = new Size(800, 600);
        }

        private void MainForm_Load(object sender, EventArgs e)
        {
            InitializeDataGridView();
            InitializeViewListBox();
            LoadData();
        }

        private void InitializeDataGridView()
        {
            dataGridView = new DataGridView();
            dataGridView.Dock = DockStyle.Fill;
            dataGridView.BorderStyle = BorderStyle.None;
            dataGridView.BackgroundColor = Color.FromArgb(240, 240, 240);
            dataGridView.GridColor = Color.FromArgb(200, 200, 200);
            dataGridView.DefaultCellStyle.BackColor = Color.White;
            dataGridView.DefaultCellStyle.ForeColor = Color.Black;
            dataGridView.DefaultCellStyle.SelectionBackColor = Color.FromArgb(0, 120,
215);
            dataGridView.DefaultCellStyle.SelectionForeColor = Color.White;
            dataGridView.ColumnHeadersDefaultCellStyle.BackColor = Color.FromArgb(0,
120, 215);
            dataGridView.ColumnHeadersDefaultCellStyle.ForeColor = Color.White;
            dataGridView.RowHeadersVisible = false;
            dataGridView.DataBindingComplete += DataGridView_DataBindingComplete;
            Controls.Add(dataGridView);
        }

        private void InitializeViewListBox()
        {
            viewListBox = new ListBox();
            viewListBox.Dock = DockStyle.Left;
            viewListBox.Width = 140;
            viewListBox.BorderStyle = BorderStyle.None;
            viewListBox.BackColor = Color.LightGray;
            viewListBox.ForeColor = Color.Black;
            viewListBox.Font = new Font("Below", 18);
            viewListBox.SelectedIndexChanged += ViewListBox_SelectedIndexChanged;
            viewListBox.DrawMode = DrawMode.OwnerDrawFixed;
            viewListBox.DrawItem += ViewListBox_DrawItem;
        }
    }
}
```

```

        viewListBox.ItemHeight = 60;
        Controls.Add(viewListBox);
    }

    private void ViewListBox_DrawItem(object sender, DrawItemEventArgs e)
    {
        if (e.Index < 0) return;
        e.DrawBackground();

        string itemText = viewListBox.Items[e.Index].ToString();
        Color textColor = e.ForeColor;

        if ((e.State & DrawItemState.Selected) == DrawItemState.Selected)
        {
            textColor = Color.Yellow;
        }

        using (SolidBrush brush = new SolidBrush(textColor))
        {
            StringFormat sf = new StringFormat();
            sf.Alignment = StringAlignment.Center;
            sf.LineAlignment = StringAlignment.Center;

            e.Graphics.DrawString(itemText, viewListBox.Font, brush, e.Bounds,
sf);
        }

        using (Pen pen = new Pen(Color.FromArgb(200, 200, 200), 5))
        {
            e.Graphics.DrawRectangle(pen, e.Bounds);
        }

        e.DrawFocusRectangle();
    }

    private void LoadData()
    {
        viewListBox.Items.Add("Authors");
        viewListBox.Items.Add("Genres");
        viewListBox.Items.Add("Books");
        viewListBox.Items.Add("Customers");
        viewListBox.Items.Add("Orders");
        viewListBox.Items.Add("Sales");
    }

    private void ViewListBox_SelectedIndexChanged(object sender, EventArgs e)
    {
        string selectedView = viewListBox.SelectedItem.ToString();
        switch (selectedView)
        {
            case "Authors":
                dataGridView.DataSource = DataLoader.LoadAuthors();
                break;
            case "Genres":
                dataGridView.DataSource = DataLoader.LoadGenres();
                break;
            case "Books":
                dataGridView.DataSource = DataLoader.LoadBooks();
                break;
            case "Customers":
                dataGridView.DataSource = DataLoader.LoadCustomers();
                break;
            case "Orders":
                dataGridView.DataSource = DataLoader.LoadOrders();
                break;
            case "Sales":

```

```

        dataGridView.DataSource = DataLoader.LoadSales();
        break;
    default:
        break;
    }
}

private void DataGridView_DataBindingComplete(object sender,
DataGridViewBindingCompleteEventArgs e)
{
    if (dataGridView.DataSource != null)
    {
        int columnCount = dataGridView.Columns.Count;
        int totalWidth = dataGridView.ClientSize.Width -
dataGridView.RowHeadersWidth;
        int columnWidth = totalWidth / columnCount;

        for (int i = 0; i < columnCount; i++)
        {
            dataGridView.Columns[i].Width = columnWidth;
        }
    }
}
}

```

### DataGridViewLoader.cs

```

using System.Data;
using System.Windows.Forms;

namespace BookstoreManagementApp
{
    public static class DataGridViewLoader
    {
        public static void LoadDataIntoDataGridView(DataGridView dataGridView,
        DataTable dataTable)
        {
            dataGridView.DataSource = dataTable;
        }
    }
}

```

### DataLoader.cs

```

using System;
using System.Data;
using System.Data.SqlClient;

namespace BookstoreManagementApp
{
    public static class DataLoader
    {
        private const string connectionString =
"Server=.\SQLEXPRESS;Database=BookstoreManagement;Integrated Security=True;";

        public static DataTable LoadAuthors()
        {
            using (SqlConnection connection = new SqlConnection(connectionString))
            {
                string query = "SELECT * FROM AuthorsView";
                SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
                DataTable dataTable = new DataTable();
                adapter.Fill(dataTable);
            }
        }
    }
}

```

```

        return dataTable;
    }
}

public static DataTable LoadGenres()
{
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM GenresView";
        SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
        DataTable dataTable = new DataTable();
        adapter.Fill(dataTable);
        return dataTable;
    }
}

public static DataTable LoadBooks()
{
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM BooksView";
        SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
        DataTable dataTable = new DataTable();
        adapter.Fill(dataTable);
        return dataTable;
    }
}

public static DataTable LoadCustomers()
{
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM CustomersView";
        SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
        DataTable dataTable = new DataTable();
        adapter.Fill(dataTable);
        return dataTable;
    }
}

public static DataTable LoadSales()
{
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM SalesView";
        SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
        DataTable dataTable = new DataTable();
        adapter.Fill(dataTable);
        return dataTable;
    }
}

public static DataTable LoadOrders()
{
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM OrdersView";
        SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
        DataTable dataTable = new DataTable();
        adapter.Fill(dataTable);
        return dataTable;
    }
}
}
}

```



## Program.cs

```
using System;
using System.Windows.Forms;

namespace BookstoreManagementApp
{
    static class Program
    {
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new MainForm());
        }
    }
}
```