# ВСТУП

Сьогодні комп’ютерні технології присутні майже всюди, у будь-якій сфері людського життя. Вони активно використовуються для автоматизації різних дій, в тому числі і праця з тваринами зоопарку, процес годівлі.

Предметною областю ІС, тобто сферою застосування нашої бази даних [1, с. 6], є музей картин. Використання БД при правильній його організації суттєво змінить діяльність організації, де вона впроваджується: привести до забезпечення більшої доступності даних для всіх категорій співробітників, скорочення документообігу та інше [2]. Саме для полегшення роботи працівників, а точніше – спрощення таких процесів, як годівля та розселення тварин зоопарку, була створена інформаційна система «Музей картин».

За допомогою бази даних, яка зберігає усі необхідні дані про експонати, зали, митців, їх картини, напрями мистецтва, адміністратору музею буде набагато легше контролювати свій музей, використовуючи різноманітні функції, як фільтрація, сортування та пошук, які надаються даною ІС.

У якості СУБД - спеціальне програмне забезпечення для роботи з файлами баз даних [3], для розробки та рішення поставленої задачі була вибрана MS SQL Server, тому що вона є швидкою та легкою у використанні та у вивчені. Для реалізації ІС використовується .NET Framework, WPF, що містить багато можливостей, які можна використовувати для поліпшення програм з метою задоволення конкретних потреб користувачів [4], ця технологія дозволяє складати програму з гарним і легким для користувача інтерфейсом. Середою розробки є Microsoft Visual Studio 2017 року, яка дозволяє швидко створити інтелектуальну програму [5] для поставленої задачі.

# 1 АНАЛІЗ ТА КОНЦЕПТАЛЬНЕ ПРОЕКТУВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Аналіз предметної області

Живопис - один з найскладніших, і, безумовно, самих чудових напрямків в мистецтві. Саме в картинах художник може в повному обсязі висловити свої почуття, емоції або передати ставлення до якоїсь ситуації, зберегти на папері ідею за допомогою всього лише пензля і десятка фарб різних кольорів. Ідеї, безумовно, вічні, а ось полотно - ні. І для збереження величніших творів не менш величних авторів і були створені музеї мистецтва - тут картини знаходяться в безпеці, під наглядом і в умовах, сприятливих для їх тривалого збереження в первісному вигляді. А відвідувачі тим часом можуть насолоджуватися творами митців, які жили півтисячі років тому!

Але, щоб музей функціонував без нарікань, необхідна система обліку творів живопису, щоб своєчасно оновлювати колекцію картин. Дана ІС являє собою таку систему.

В ході виконання даної роботи планується створити інформаційну систему «Облік творів живопису». Дана система дозволить спростити управління музеєм картин. Вона прийме на себе функції зберігання, систематизації і пошуку всієї необхідної інформації, що полегшує адміністрування музею, облік витворів мистецтва, додавання, видалення та модифікацію поточної колекціїї експонатів, дозволяє відмовитись від використовування паперових документів, прискорити внесення змін до облікових записів, заощадити на витратних матеріалах.

Для реалізації даної ІС використання бази даних представляється розумним і обґрунтованим рішенням - в зв'язку з чим основу даного проекту складе база даних музею картин. Це пов'язано з тим, що проект буде повинен зберігати великі обсяги пов'язаних даних, які потрібно буде оновлювати і синхронізувати.

Проаналізувавши предметну галузь, можна виділити основну інформацію, що має зберігатися в базі даних – список залів музею, їх розміри, колекція експонатів відповідно до зали, база знань про існуючи картини, митців і напрямів живопису.

Головні нефункціональні вимоги до системи:

- Її надійність;

- Коректна робота системи з урахуванням некоректних дій користувача;

- Адекватність та відповідність ПГ;

- Забезпечення цілісності інформації;

- Забезпечення коректного завершення роботи системи;

1.2 Концептуальне моделювання предметної області

На рисунку 1.1 можна побачити функціональну структуру інформаційної системи, яка демонструє можливого користувача, того, хто буде взаємодіяти з ІС: Адміністратор – адміністратор музея, що володіє повними правами доступу на додавання, модифікацію та видалення даних.



Рисунок 1.1 - Функціональная структура ІС.

Надалі опишемо інформаційні потреби користувача системи.

Адміністратор має потреби в:

а) перегляді колекції експонатів картин, присутніх в музеї;

б) перегляд інформаціїї про відомих митців, картини, напрями мистецтва;

б) сортуванні колекціїї експонатів за параметрами експонату, як-то: розмір, зал, назва картину, що він представляє, тощо;

в) фільтрації колекції експонатів за роками написання, розмірами, оригінальністю;

г) фільтрації колекцій авторів і пов’язаних із ними картин за роками життя, датами написання картин та їх напрямом;

г) додаванні, видаленні та редагуванні інформації, яка присутня в базі даних, тобто інформаціїї про експонати, митців, картини та ін;

д) формуванні статистичних звітів:

1) графік, що показує розподілення кількості картин для обраного митця по роках протягом періоду його життя (до його смерті або то теперішнього року.);

2) перехресний запит, що виводить інформацію про кількість картин кожного митця для кожного напряму мистецтва;

3) статистика по залах – кількість експонатів у кожному, їх середній розмір, домінуючий напрям мистецтва (якщо є), та ін.;

4) статистика по експонатах кожного художника, якого обирає користувач;

Задача автоматизації проекта полягає в:

а) програмній реалізаціії перехресних запитів (див. попередній пункт);

б) створення візуального графіка статистики картин для митця впродовж часу його життя, по кожних 5 років;

На рисунку 1.1 представлені головні об’єкти інформаційної системи та зв’язки між ними.

Розглянемо основні сутності, що присутні в даній інформаційній системі. Головними сутностями цієї ІС є: експонат, зала музею, картина, митець та напрямок мистецтва. Зв’язки між ними виглядатимуть наступним чином:

а) митець може написати багато картин, але одна картина пишеться лише одним митцєм;

б) одна картина може бути представлена лише одним експонатом, та один експонат може представляти лише одну картину;

в) в одному залі може виставлятися багато експонатів, але один експонат может знаходитися лише в одному залі;

г) картина може бути написана лише в одному напрямку мистецтва, але багато картин можуть бути написані в одному тому ж напрямі мистецтва;

Головною сутністю інформаційної системи є експонат. Для нього в базі даних зберігаються:

а) Id картини – первісний ключ, також є зовнішним ключом для зв’язку з таблицею «Картина».

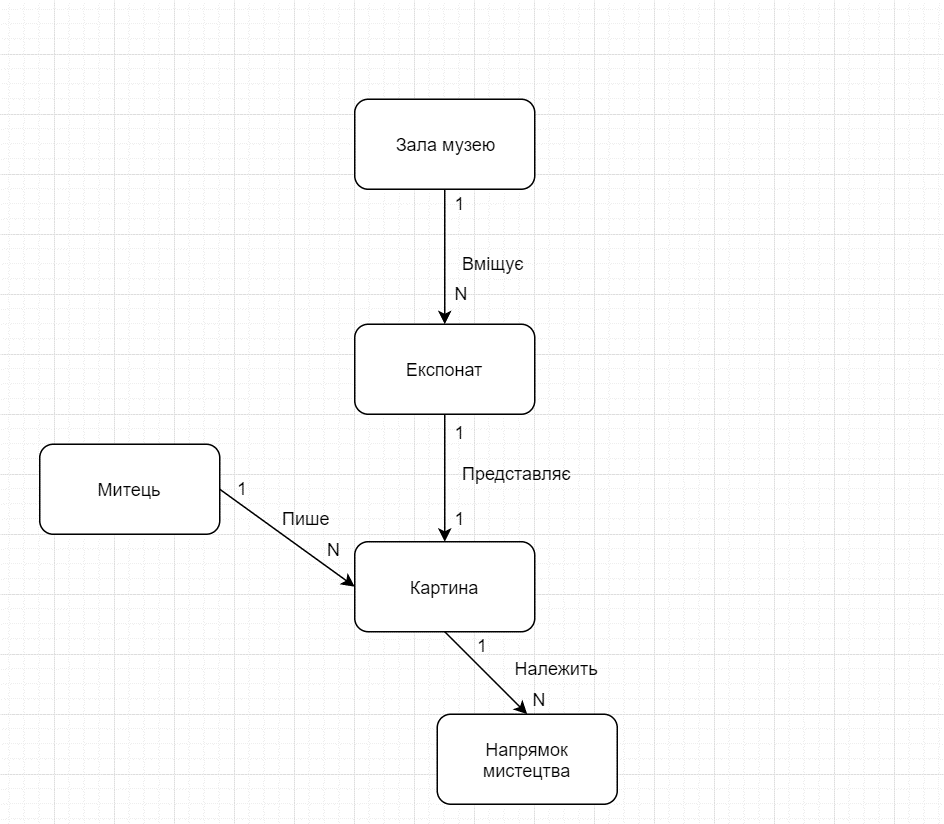


Рисунок 1.2 – Головні об’єкти ІС

б) дата придбання – дата, коли експонат було придбано/набуто і виставлено в музеї;

в) чи є орігіналом – атрибут логічного типу, вказує, чи є експонат оригіналом або репродукцією;

г) ширина – ширина експонату у см.;

д) висота – висота експонату у см.;

е) тип полотна – тип полотна, на якому написан картина (напр, холст, папір та ін);

ж) тип фарби – тип фарби, якою було написано картину (напр, масло, акварель та ін);

з) Id зали – зовнішний ключ, слугує для зв’язку з таблицею «Зала», щоб зв’язати експонат із залою, де він знаходиться;

Іншими сутностями є (надалі без пояснень):

а) картина;

б) зала;

в) митець;

г) напрям мистецтва;

До складу існуючого документообігу входять інформаційний аркуш експонату, що заповнюється під час надходження нового експонату до музея (картина, дата придбання, чи є оригіналом, довжина, висота, тип полотна, тип фарби, зала), довідник з картин/митці/напрямів мистецтва, що містить їх назви, описи/біографіїї та ін.

До алгоритмічних залежностей належать:

- розрахування наступного кроку для абсциси статистичного графіку по картинах митця як (попередній\_крок + 5) % рік\_смерті if not null else поточний\_рік.  
 - розрахування домінуючого напряму мистецтва для кожної зали як максимум(кількість\_експонатів\_для\_кожного\_напряму) порівняти з припустимим відхиленням: «-», якщо більше, домінуючий\_напрям, якщо менше або дорівнює.

Лінгвістичні залежності, що існують у межах даної предметної області, мають бути упорядковані термінологією, що є прийнятною для кадрів даної організації (музей картин). Більшість термінів є широковживаними у буденному житті і зрозумілою будь-якому користувачеві, але слід уточнити значення одного з термінів, який рідко вживається в повсякденній мові.

Напрям мистецтва – термін, що позначає принципову спільність художніх явищ упродовж певного часу. Тобто, картини одного напряму будуть співпадати по стилю написання, тематиці, темам.

В якості мови програмування був обраний C# на базі технології Windows Presentation Foundation (WPF). Програму буде розроблено в середовищі Microsoft Visual Studio 2017 з використанням СУБД Microsoft SQL Server, ORM Entity Framework і можливістю формування звітів у форматі .docx (текстовий процесор Microsoft Office Word).

# 2 ПОСТАНОВКА ЗАДАЧИ

Необхідно спроектувати і програмно реалізувати систему, яка дозволить працювати з базами музею картин. Система повинна бути зрозумілою, надійної, відмовостійкою в разі помилок користувача.

Існує інформація про напрями мистецтва, митців, картини, що вони написали і які належать до конкретного напряму, експонати музею, кожний з яких представляє конкретну картину і знаходиться в конкретній залі музею, в кожної з яких є своя тематика і розмір. Є інформація щодо років життя митців, сторіч існування напрямів, фізичних характеристик експонатів. Системою можуть користуватися незареєстровані користувачі, зареєстровані користувачі, а також адміністратор.

Необхідно реалізувати:

а) додавання, редагування та видалення інформації щодо картин, митців, експонатів, напрямів мистецтва тощо;

б) пошук, сортування і фільтрацію даних:

1) пошук по ключовому слову із виділенням кольором знайдених записів;

2) можливість переходити взад/уперед по знайдених записах за допомогою клавіш управління;

3) сортування по обраному полю в обраному порядку (зростаючий, спадаючий);

4) фільтрація по декількох критеріях;

5) можливість додавати/знімати декілька фільтрів;

6) можливість задавати спосіб одночасного використання фільтрів – логічне ТА (усі одразу) або логічне АБО (хоча б один);

в) 3 найбільш часто виникаючих до бази даних запита (статистика);

г) Візуальний графік розподілення картин обраного митця впродовж періоду його життя;

д) Формування звітів у вигляді текстових документів про переважний напрямок кожного з митців, про кількість картин митця у різних напрямах, звіт експонатів по залах, з повними назвами картин та ПІБ митців – авторів;

**3 ПРОЕКТУВАННЯ БАЗИ ДАНИХ**

3.1 Побудова UML-діаграми

Розглянемо цілі створення інформаційної системи «Музей картин» та функції, які вона буде виконувати, більш детальніше, сбудувавши діаграму прецедентів (USE-CASE діаграму). Для початку треба виділити границі системи та основних акторів – користувачів системи, основні функції, якими будуть користуватися актори.

Актор (адміністратор) може виконувати функції з ІС, а саме:

- мати повний доступ до даних – експонатів, митців та їх картин;

- перегляди, фільтрувати, виконувати пошук по колекціях експонатів, картин, митців тощо;

- додавати, видаляти, модифікувати дані;

- переглядати статистику та формувати звіти у вигляді документів;

Отримана діаграма знаходиться на рисунку 3.1.

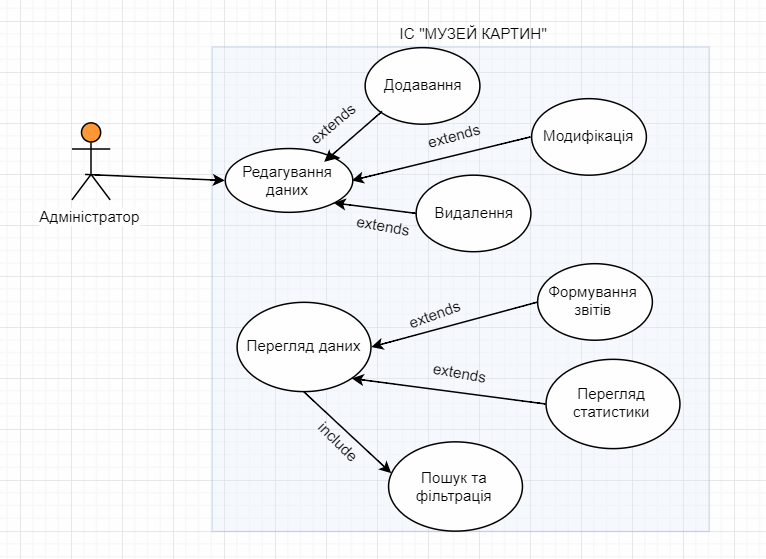


Рисунок 3.1 – UML-діаграма ІС.

3.2 ER – моделювання

Для більш швидкої і ефективної побудови бази даних для створення даної інформаційної системи була створена ER-діаграма. Ця діаграма дає можливість уявити собі майбутню структуру бази даних без необхідності цю бази створювати на етапі проектування. На створеній ER-діаграмі (рисунок 3.2.1) представлені наступні сутності й атрибути:

а) картина – характеризується її номером (Id), назвою, номером (Id) митця (її автора), роком свого написання, безпосередньо зображеням і номером (Id) напряму мистецтва, до якого належить;

б) митець – характеризується своім номером (Id), ПІБ, Датою народження, датою смерті (якщо є) та біографією;

в) експонат – головна сутність ІС, характеризується номером (Id) картини, яку представляє, датою отримання (коли він з’явився в музеї), атрибутом, що показує, оригінал це чи репродукція, фізичними характеристиками (шириною, висотою, типом фарби та холста) та номером (Id) зали, до якої належить;

г) зала - номером (Id) зали, назвою зали, тематикою та номером (Id) розміра, якого вона є;

д) розмір – характеризується номером (Id), назвою (маленький, середній та ін) і максимальною кількістю картин, що може вмістити в себе;

е) напрям мистецтва – характеризується своїм номером (Id), назвою, сторіччами початку та кінця та описом;

ж) користувач – запис у базі даних, яка зберігає пари { ключ : значення } для усіх зареєстрованих користувачів системи;

На даній ER – моделі необхідно виділити наступні зв’язки:

а) картина – митець: 1 – N , бо митець може протягом життя написати багато картин, але ми припускаємося, що автором картини може бути лише один митець.

б) напрям мистецтва – картина: 1 – N, бо у картини может бути лише один напрям – той, в якому її написали, але можуть існувати багато картин спільного напряму.

в) картина – експонат: 1 – 1, бо експонат, звісно, не може являти собою декілька картин одночасно, також і картина не може бути представлена багаться експонатами, бо в музеї немає жодного сенсу виставляти декілька експонаті однієї тієї ж картини поруч.

г) зала – експонат: 1 – N, бо один і той самий експонат фізично не може знаходитися в декількох залах одночасно, але у залі можуть знаходитися декілька експонатів.

д) розмір – зала: 1 – N, бо одна зала не може бути одночасно великою і маленькою, але в музеї можуть знаходитися декілька залів, і, відповідно, в кожного буде свій розмір.

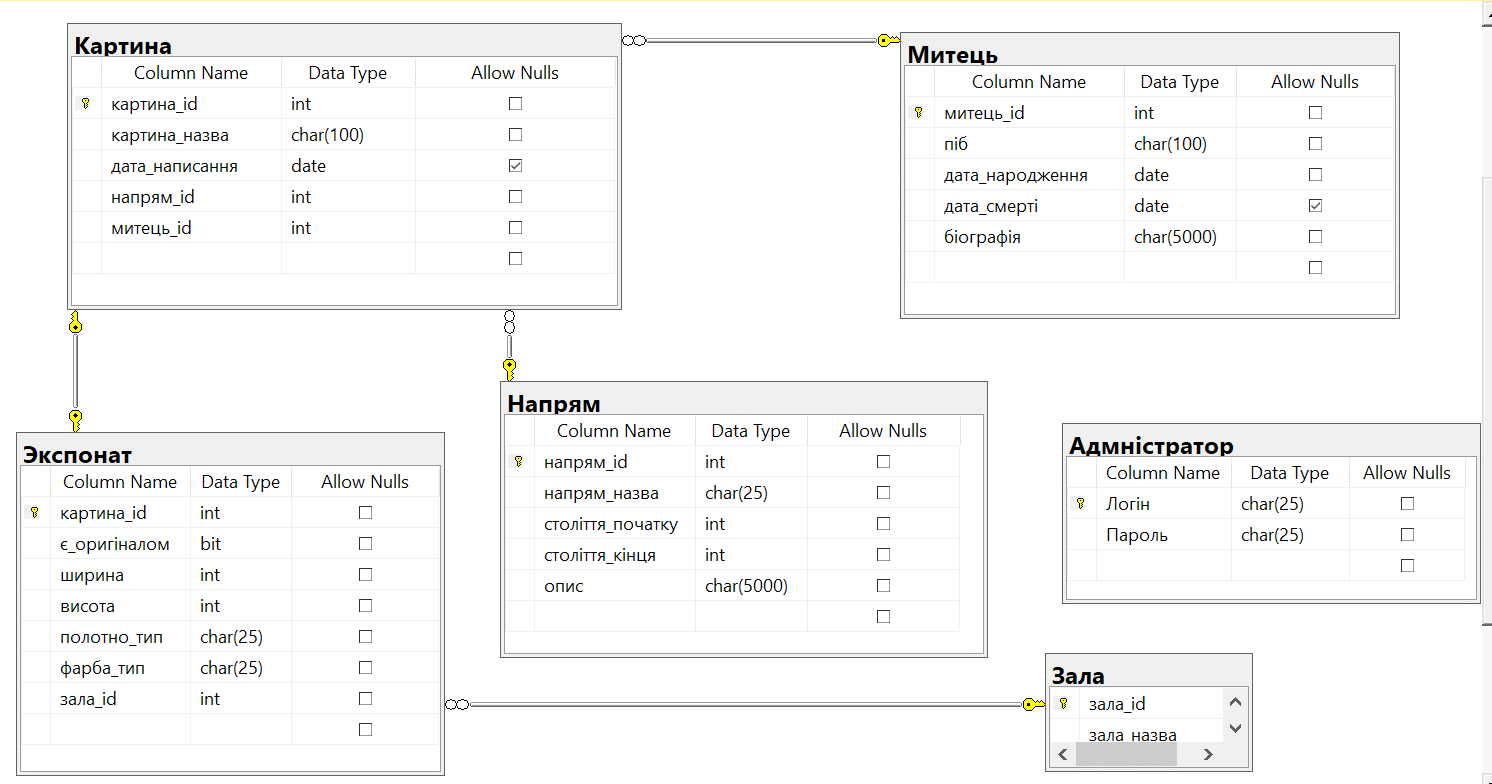


Рисунок 3.2 - ER-діаграма

3.3 Побудова схеми реляційної бази даних в третій нормальній формі

Розглянемо базу даних, що складається лише з одного відношення –Картина. Первинним ключем у цьому відношенні є картина\_Id.

Розглянемо усі функціональні залежності, що присутні у цьому предметній області (див. рисунок 3.2), а саме:

а) залежності від первинного ключа:

{картина\_id} → картина\_назва, дата\_написання, дата\_отримання, є\_оригіналом, ширина, висота, полотно\_тип, фарба\_тип, митець\_id, піб, дата\_народження, дата\_смерті, біографія, напрям\_id, напрям\_назва, століття\_початку, століття\_кінця, опис, зала\_id, зада\_назва.

б) залежності від неключових атрибутів:

1) митець\_id → ПІБ, дата\_народження, дата\_смерті, біографія.

2) напрям\_id → напрям\_назва, століття\_початку, століття\_кінця, опис.

3) зала\_id → зала\_назва.

Вважаємо, що відношення знаходиться у 1НФ, бо у кожному записі буде знаходитися атомарне значення і кожний неключовий атрибут є функціонально-залежним від первісного ключа.

Через те, що у відношення первісний ключ є простим(тобто, немає таких ятрибутів, які б залежали від підмножини множини значення первісного ключа) і відношення вже знаходиться у 1НФ, одразу можна сказати, що воно знаходиться у 2НФ.

Рисунок 3.1 - Приклад транзитивної залежності у відношенні «Картина»

Відношення (Картина) НЕ знаходиться у 3НФ, бо присутні транзитивні залежності, наприклад на рисунку 3.1

Щоб привести відношення Картина до 3НФ, треба розбити його на декілька відношень: Картина (залишається), Митець, Напрям, Зала. (див. рисунок 3.3).

|  |
| --- |
| картина\_id |
| картина\_назва |
| дата\_написання |
| дата\_отримання |
| є\_оригіналом |
| ширина |
| висота |
| полотно\_тип |
| фарба\_тип |
| митець\_id |
| піб |
| дата\_народження |
| дата\_смерті |
| біографія |
| напрям\_id |
| напрям\_назва |
| століття\_початку |
| століття\_кінця |
| опис |
| зала\_id |
| зала\_назва |

Рисунок 3.2 - Основні функціональні залежності у відношенні «Картина»

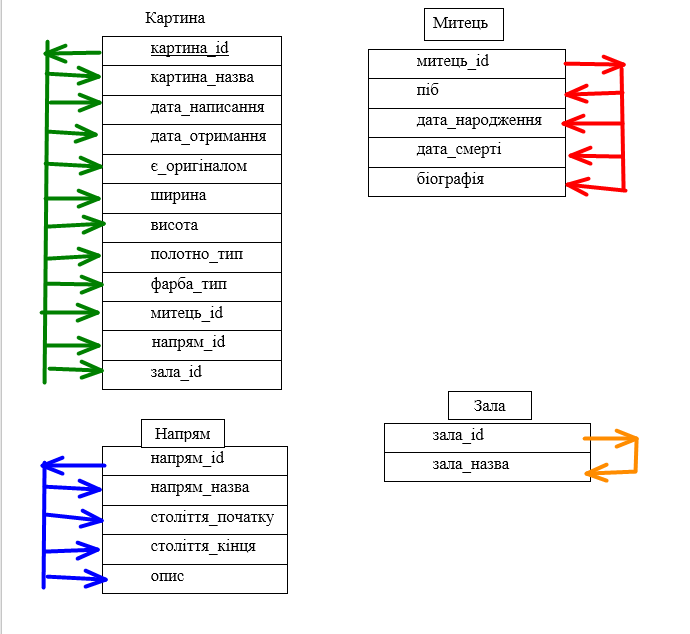


Рисунок 3.3 - Декомпоноване відношення Картина

Ці відношення знаходяться у 3НФ кожне, (бо усі знаходяться у 2НФ (бо усі знаходяться у 1НФ, бо ми вважаємо, що значення в усіх записах атомарні. Також кожен неключовий атрибут функціонально залежить від первісного ключа.) та в кожного відношення простий первісний ключ – одже, не існує атрибутів, що залежать від його підмножини.), бо в жодного відношення не має транзитивних залежностей (немає таких атрибутів, які б залежали не від первісного ключа).

На рисунку 3.4 приведено схему БД «Музей», що містить 5 відношень у 3НФ.

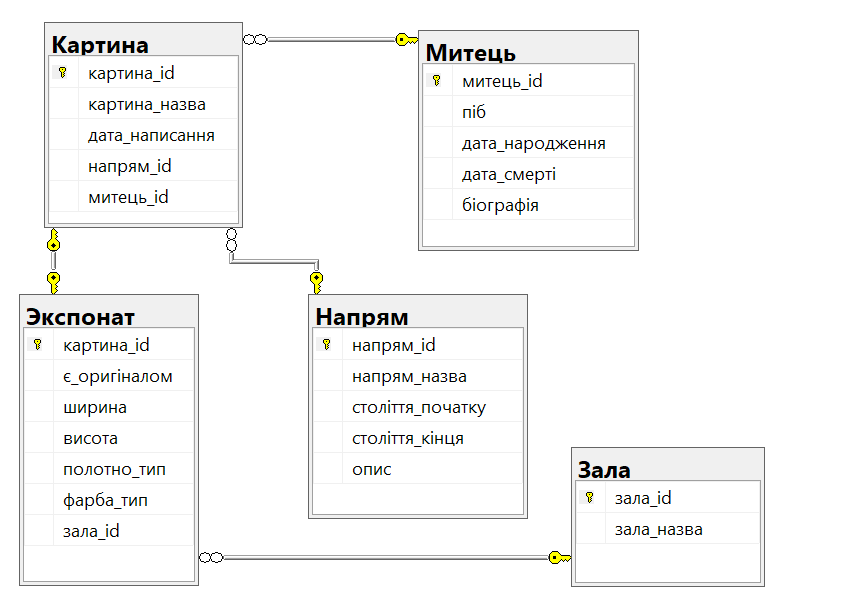


Рисунок 3.4 - Нормалізована схема БД «Музей»

Варто додати, що, незважаючи на те, що відношення Картина вже знаходилося у 3НФ, його було розділено на два – власне картину як витвір мистецтва та експонам, як ії фізичне представлення. Це було зроблено для наближення схеми БД до більш реалістичного відображення предметної області системи.

# 4 ОПИС ПРОГРАМИ

4.1 Загальні відомості

Даний програмний продукт було розроблено у вигляді десктопного додатку для платформи Microsoft Windows, за допомогою мови програмування С# та мови розмітки XAML. Для розроблення продукту використовувалася технологія Windows Presentation Foundation ті бібліотека OxyPlot. В якості технології доступу к даним використовувався Entity Framework. В якості СУБД було використано MS Sql Server, бо це сучасна, надійна, швидка та відмовостійка СУБД, яка поставляється зі зручним додатком для простого адміністрування бази даних – Sql Server Management Studio абсолютно безкоштовно. Також Entity Framework надає найбільш сумісний функіонал тілььки для Sql Server, бо обидва продукта входять до складу .NET Franework.

Підсумкова програма без додаткових модулів займає 149 кб. Для запуску програми комп’ютер повинен мати операційну систему Windows та встановлений Sql Server.

4.2 Виклик та завантаження

Додаток знаходиться у директорії «CourseDB» диску С. Для запуску програми необхідно:

- переконатися, що на комп’ютері встановлено Sql Server та що його запущено (це можна зробити у вікні служб Windows. Щоб його відкрити, натисніть комбінацію клавіш «Win» та «R», далі впищіть у вікні «services.msc» та натисніть клавішу Enter);

- переконатися, що у конфігураційному файлі додатка прописано правильну пусть підключення до бази даних – ConnectionString;

- відкрити додаток, натиснувши два рази ЛКМ або Enter;

4.3 Призначення і логічна структура

При реалізації інформаційної системи «Система обліку творів живопису музея» було використано так звану тришарову модель програмного додатку.

Тобто архітектурно додаток складається з трьох прошарок:

а) шар подання – відповідає за візуальне подання інформації користувачу, за його взаємодію з елементами інтерфейсу, розмітку, позіціонування тощо;

б) шар бізнес-логіки – відповідає за обробку, аналіз та валідацію даних, введених користувачем, визначає, як буде себе поводити програма, як реагуватиме на дії користувача, які функції мати і т.д.;

в) шар доступу к даним – визначає, як буде проходити зв’язок між базою даних та додатком, отримує від шара бізнес-логіки запити та виконує їх безпосередньо над базою даних, повертаючи результат знову до шара бізнес-логіки;

В цьому додатку в якості елементів шару подання слугують файли з розширенням .xaml – файли мови розмітки XAML. Саме в них задекларована стуктура користувацького інтерфейсу, розміщення та візуальні характеристики його елементів.

Шар бізнес-логіки являє собою файли з розширенням .cs і поділяється на 2 групи:

а) обробники подій користувацького інтерфейсу – мають ті ж назви, що й відповідні XAML файли, але інше розширення. Відповідають за виклики спеціалізованих методів обробки даних, введених користувачем;

б) власне бізнес-логіка – конвертери даних, парсери, інтерфейси та ін. Відповідають за аналіз та обробку переданих даних, не є специфічними для цього програмного додатку, тобто, можуть бути винесені як частина загальної реалізації для групи додатків;

Шар доступу к даним поданий фреймворком роботи з базами даних на платформі .NET – Entity Framework. Цей фреймворк бере на себе усі операції роботи з даними (як-то INSERT, UPDATE, SELECT, DELETE та ін.), надаючи розробнику зручну можливість абстрагуватися від специфіки реалізації конкретної СУБД, можливість роботи із сутностями бази даних як зі звичайними колекціями об’єктів, що, беручи до уваги об’єктну орієнтовність .NET, дозволяє швидко і легко створювати.зручні методи роботи з даними з БД.

4.4 Опис фізичної моделі бази даних

Схему фізичної моделі бази даних наведено на рисунку 4.1. Її опис міститься в таблицях 4.1 - 4.6.

База даних складається з п’яти головних таблиць:

- «Картина»;

- «Митець»;

- «Експонат»;

- «Напрям»;

- «Зала»;

та однієї системної таблиці, що не має прямого відношення до ПО ІС – «Адміністратор», що зберігає логін (-и) та пароль (-і) адміністратора (-рів) музею.

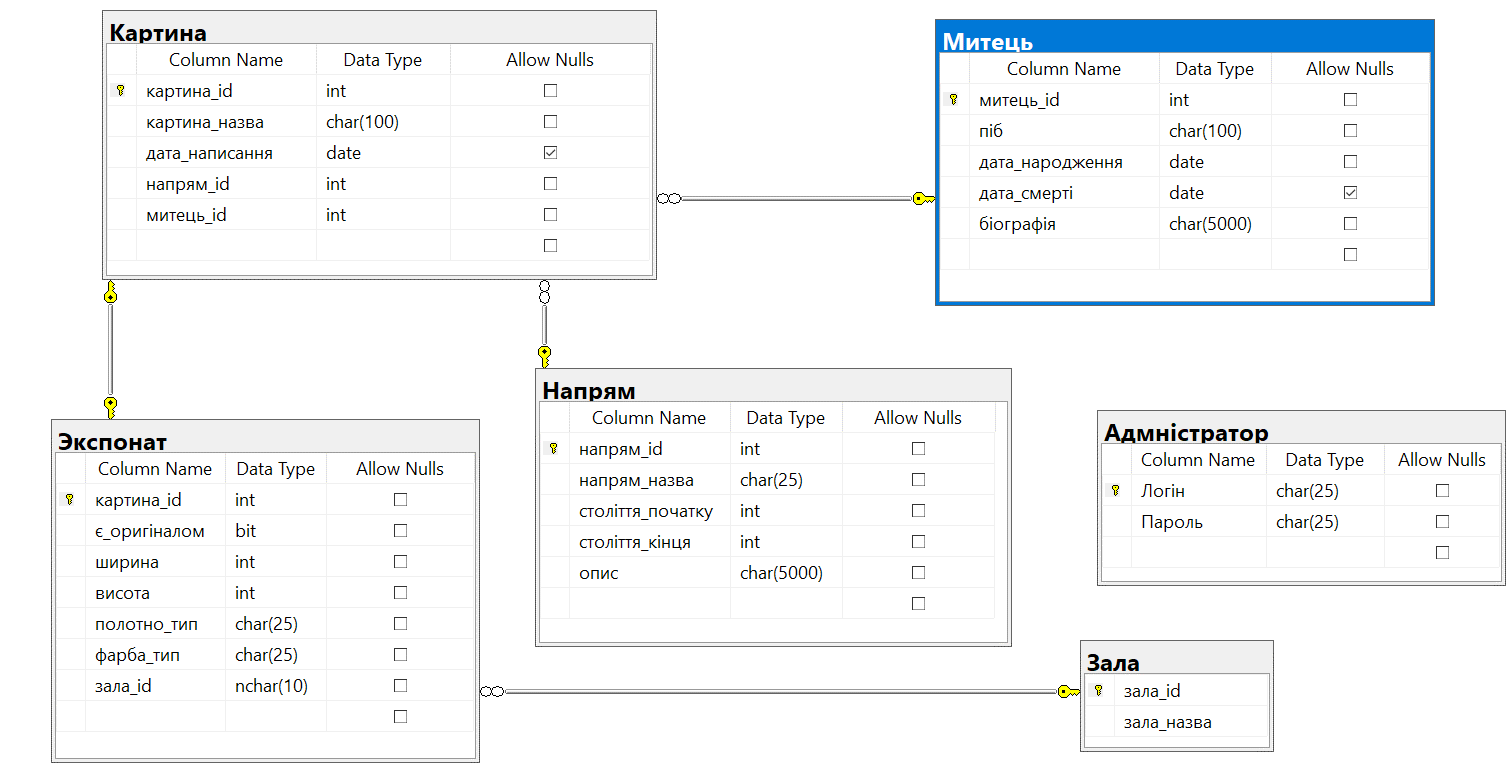


Рисунок 4.1 Схема фізичної моделі бази даних

Надалі представлено поля кожної з таблиць, що утворюють БД.

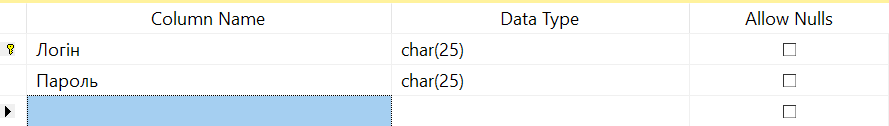


Рисунок 4.1 - Поля таблиці «Адміністратор»

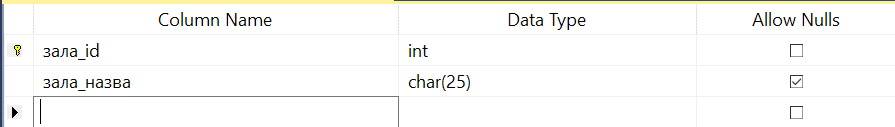


Рисунок 4.2 Поля таблиці «Зала»

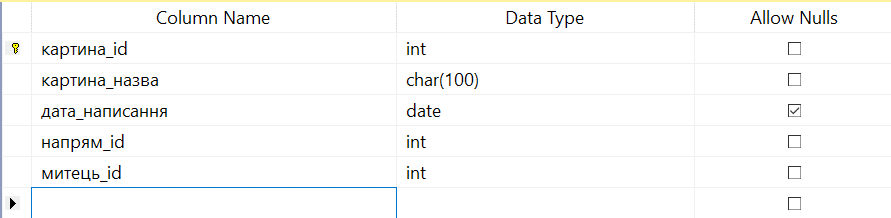


Рисунок 4.3 Поля таблиці «Картина»

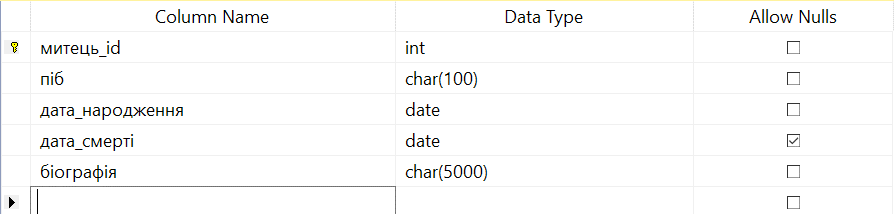


Рисунок 4.4 Поля таблиці «Митець»

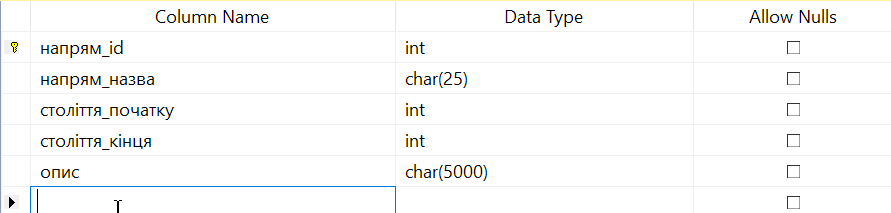


Рисунок 4.5 Поля таблиці «Напрям»

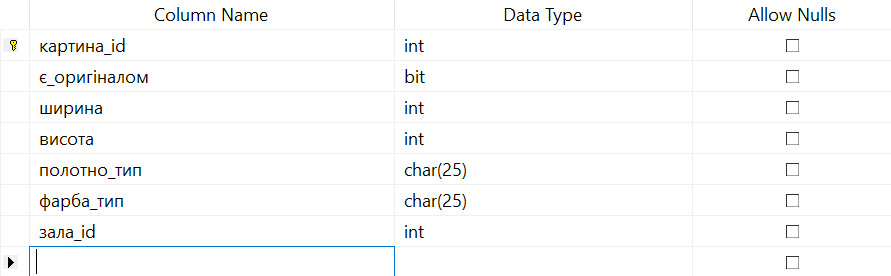


Рисунок 4.6 Поля таблиці «Експонат»

4.5 Опис програмної реалізації

Розглянемо програмний додаток з точки зору користувача. Коли він запустить його, відкриється вікно входу до програми (див. рисунок 4.2). Треба буде ввести логін та пароль, щоб підтвердити, що користувач є адміністратором та має необхідні права.

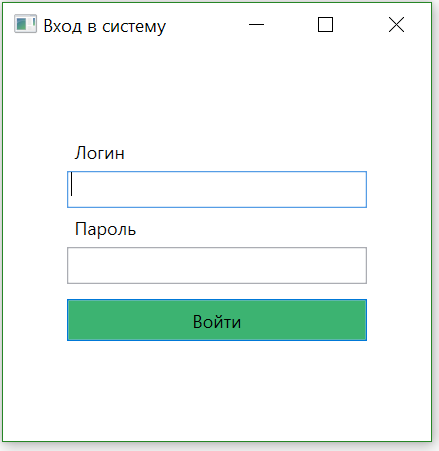


Рисунок 4.2 Вікно входа до програми

Якщо користувач є адміністратором і успішно здійснив вхід до системи, перед ним відкриється головне меню додатку (див. рисунок 4.3). Як можна побачити, вікно додатку візуально поділене на 2 частини:

а) верхня частина з «консоллю» - туди записуються дії користувача під час роботи з програмою (куди щось добавив, які вікна відвідав та ін.). Кнопки «Главное меню» та «Выход» дозволяють, перейти до цього вікна з будь-якого положення в системі, та, відповідно, вийти з додатку;

б) нижня частина являє собою вікно навігації з можливістю дивитися її історію та повертатися назад. Коли користувач зробить будь-яку навігацію, верхня частина вікна програми не зміниться, але вміст вікна навігації буде замінено (як у браузері).

Натиснувши кнопку «Экспонаты музея», користувач побачить таблицю з переліком усіх експонатів та їх атрибутів, які присутні в музеї (див. рисунок 4.4). Користувач може додавати, видаляти, модифікувати (див. рисунок 4.5), записи в таблиці, проводити пошук (див. рисунок 4.7), фільтрацію (див. рисунок 4.6).

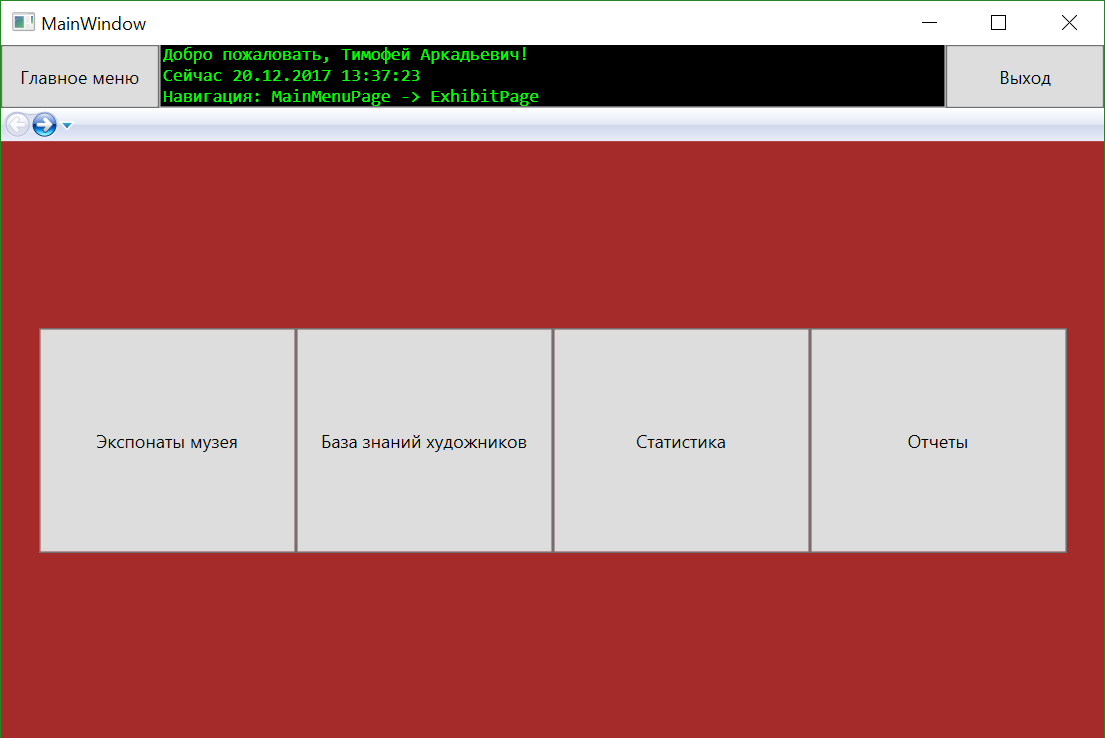


Рисунок 4.3 Головне вікно системи.

Функціонал пошуку та фільтрації є дуже схожим з іншими таблицями, тому його буде наведено лише один раз. На рисунку 4.4 користувач може побачити вікно з експонатами та поле для пошуку, кнопки «Добавить фильтры», «Снять все фильтры», «Добавить», «Изменить», «Удалить» та кнопки переходу між знайденими елементами. Натиснувши кнопку «Добавить», користувач відкриє діалогове вікно з формою додавання нового експонату (див. рисунок 4.5). Він може обрати картну, яка буде представлена новим експонатом, серед тих картин, у яких експонатів ще немає, може налаштувати інші параметри, як-то:

- дата набуння

- чи є оригіналом

- розміри

та інші.

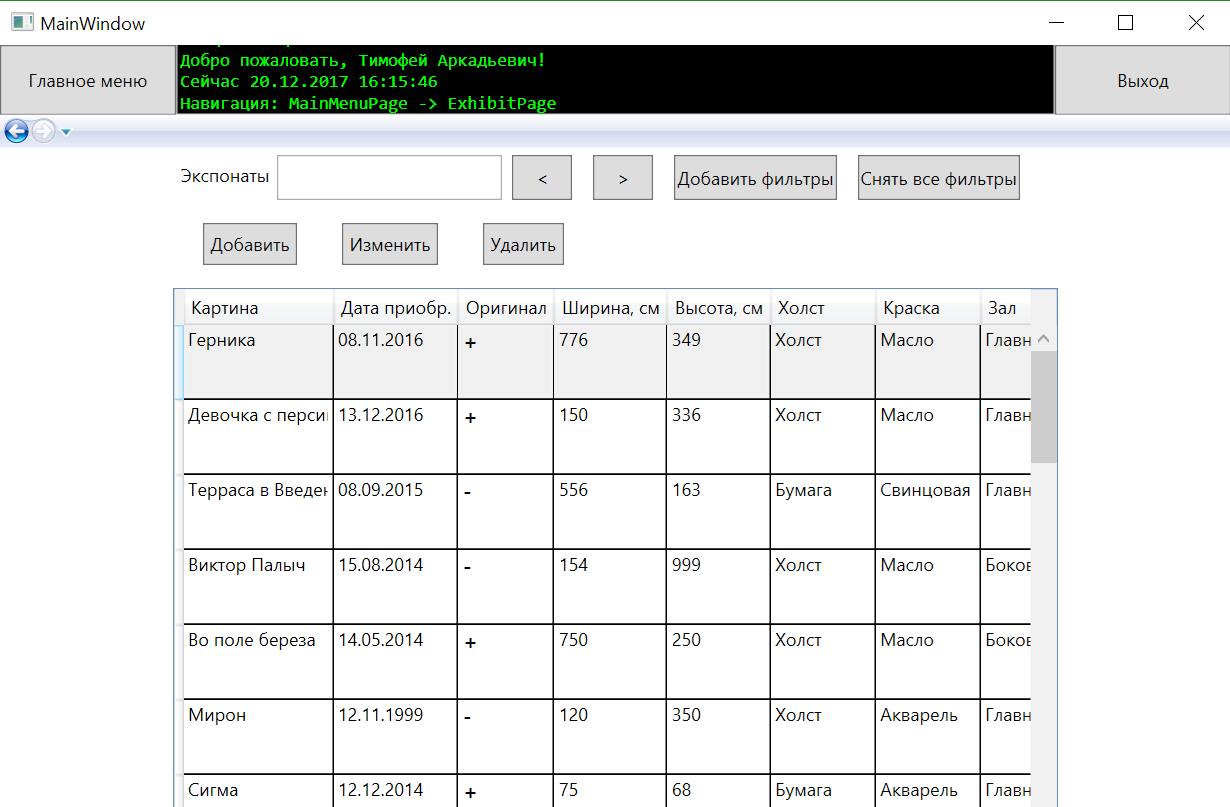


Рисунок 4.4 Колекція експонатів музея

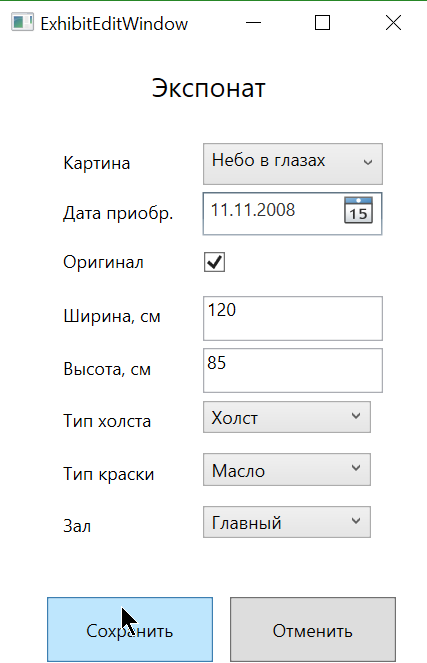


Рисунок 4.5 Вікно редагування / додавання експонатів

Далі користувач має можливість або зберегти зміни, або відмінити їх, натиснувши, відноповідно, на кнопку «Сохранить» чи «Отменить».

Модифікація виконується аналогчно, лише треба спочатку виділити потрібний елемент та натиснути кнопку «Изменить». Також під час модифікації неможливо змінити картину, яку представляє експонат.

Натиснувши кнопку «Добавить фильтры», користувач откриє вікно роботи з фільтрами таблиці (див. рисунок 4.7). У списку зліва відображаються усі поточні фільтри. Знизу можна обрати режими їх роботи – одночасний (елемент повинен проходити усі фільтри) чи паралельний (елемент повинен проходити хоча б один фільтр). Щоб додати фільтр, введіть справа необхідні дані та натисніть «Добавить фильтр», щоб видалити вже існуючий – виділить його у списку зліва та натисніть «Удалить фильтр».

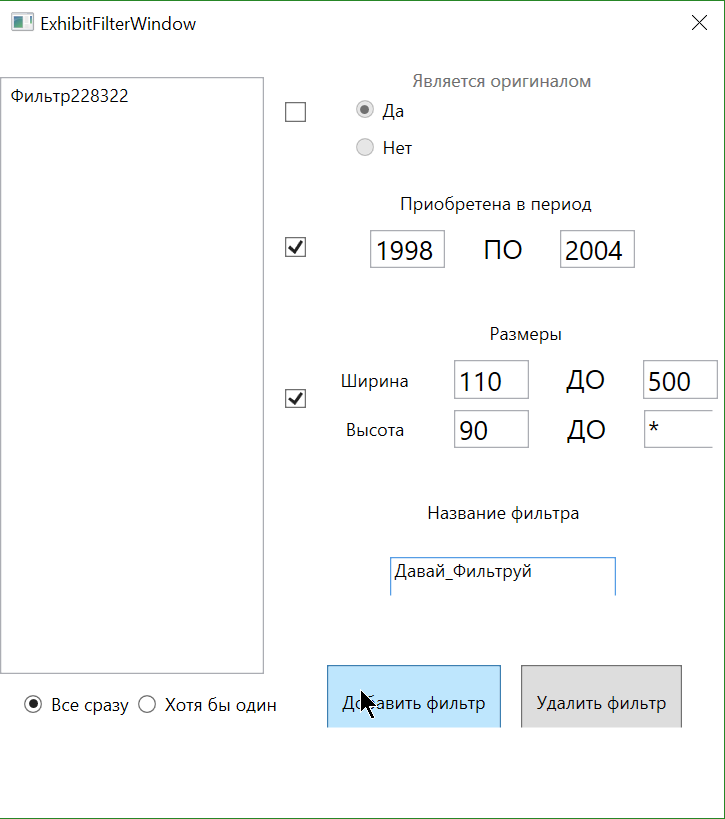


Рисунок 4.6 Вікно фільтрації колекції експонатів

Якщо користувач натисне кнопку «Снять все фильтры», то усі поточні фільтри буде знято і при наступному відкриванні вікна зліва буде порожній список.

Пошук здійснюється наступним чином – користувач вводить дані для пошуку у поле, а програма виділяє помаранчевим кольором елементи, що відповідають пошуку. Навігація між цими елементами здійснюється за допомогою кнопок зі стрілками – уперед і назад (див. рисунок 4.6)

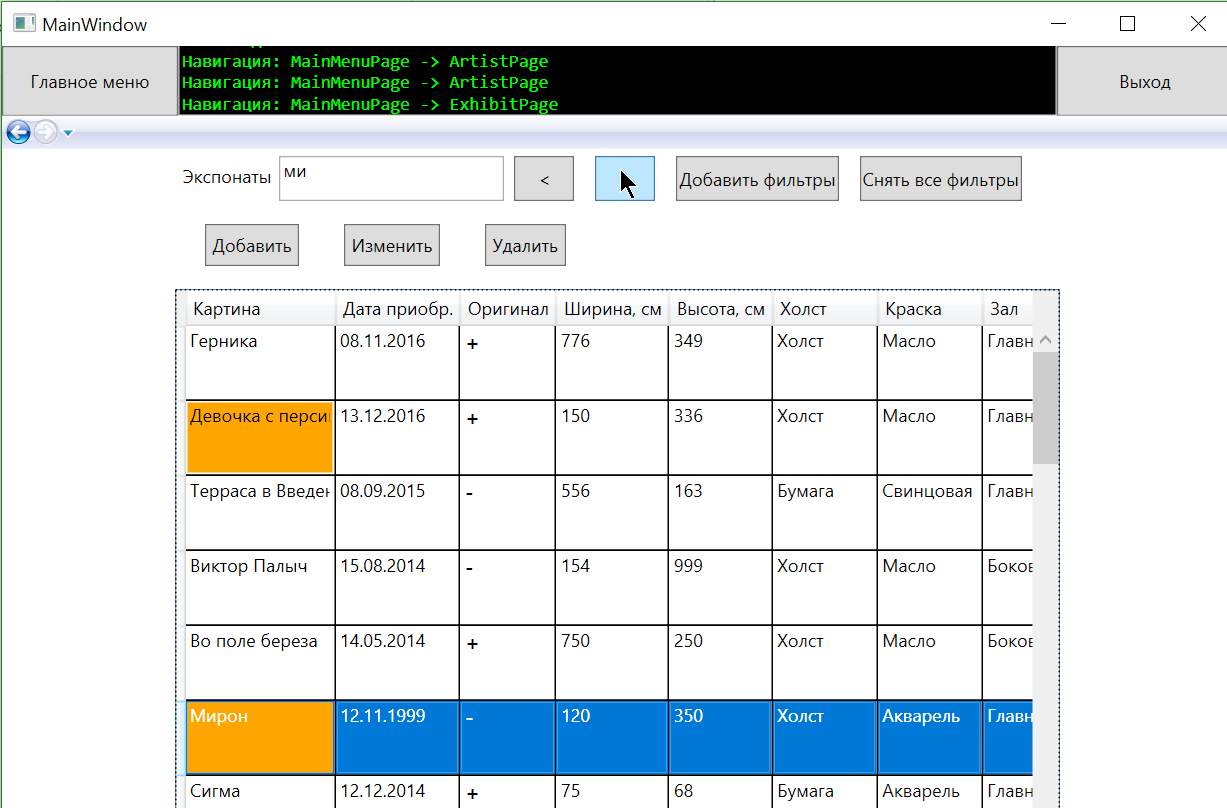


Рисунок 4.7 Здійснення пошуку по таблиці експонатів

Для видалення елементу треба його виділити та натиснути кнопку «Удалить».

Друга кнопка головного меню веде до сумісної колекції митців та відповідних їм картин (див. рисунок 4.8)

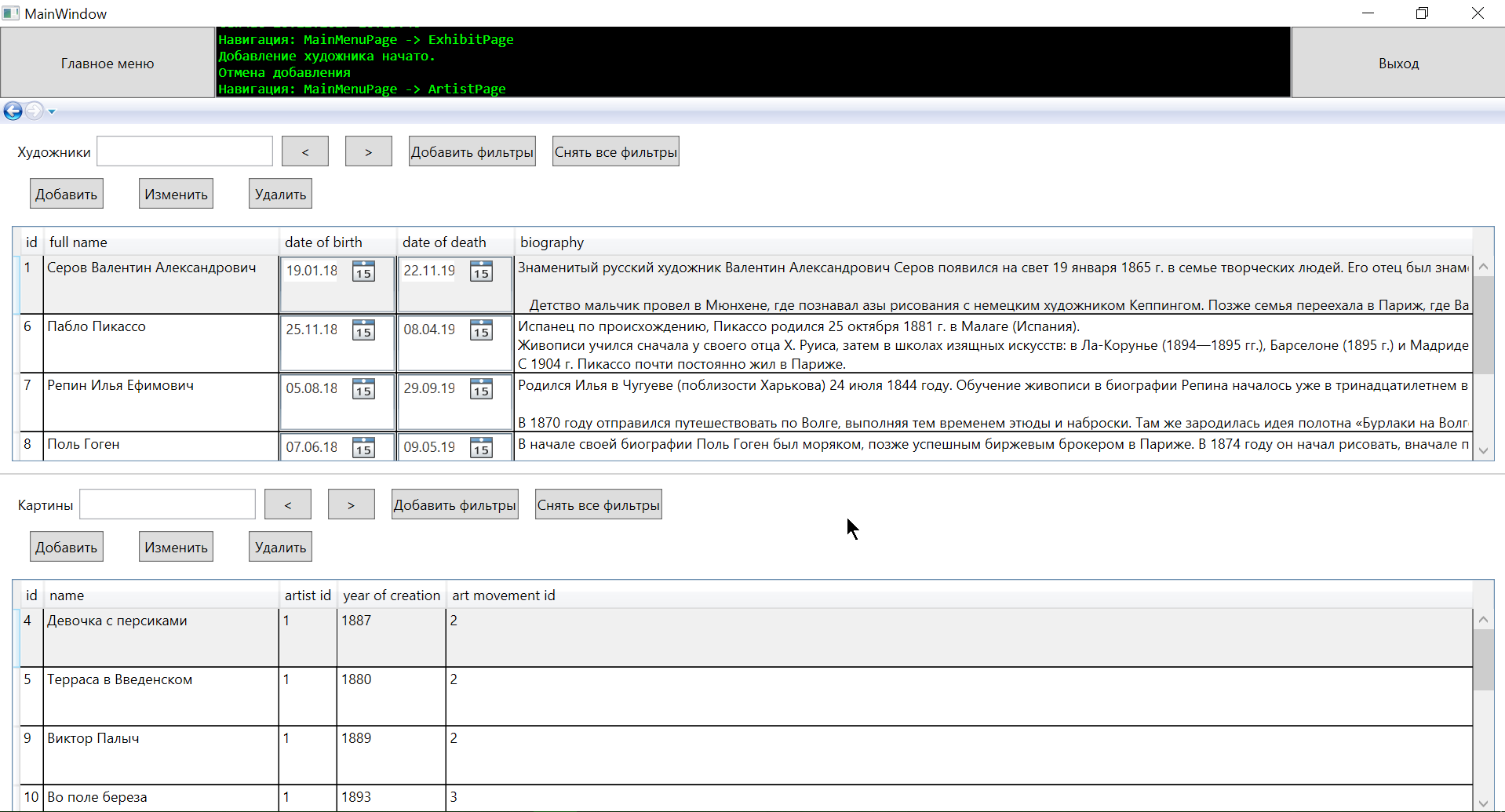


Рисунок 4.8 Сумісні таблиці митців та їх картин

Кожному митцю відповідають усі картини із таблиці знизу. Щоб отримати інформацію про картини будь-якого митця, треба лише виділити йоо зверху і знизу одразу з’являться його картини. Додавання/модифікування, Фільтрація та пошук реалізовані так само, як для експонатів (див. рисунки 4.5, 4.6, 4.7).

Третя кнопка головного меню відкриває вікно з графіком – статистикою по картинах обраного художника (див. рисунок 4.9). Користувач обирає художника, якого йому потрібно, у списку справа, і графік поновлюється. Сам графік розбито на періоди по 5 років, і для кожного періоду вказано число картин, написаних в цей час. Користувач має також можливість зберегти цей графік у вигляді зображення або PDF-документа, для чого слугують 2 кнопки справа.

Четверта кнопка головного меню веде до трьох статистичних таблиць:

а) Перехресна статистика по напрямах мистецтва – кількість картин кожного митця в кожному напрямі (див. рисунок 4.10). Реалізація – через перехресні запити (див. задачу автоматизації)

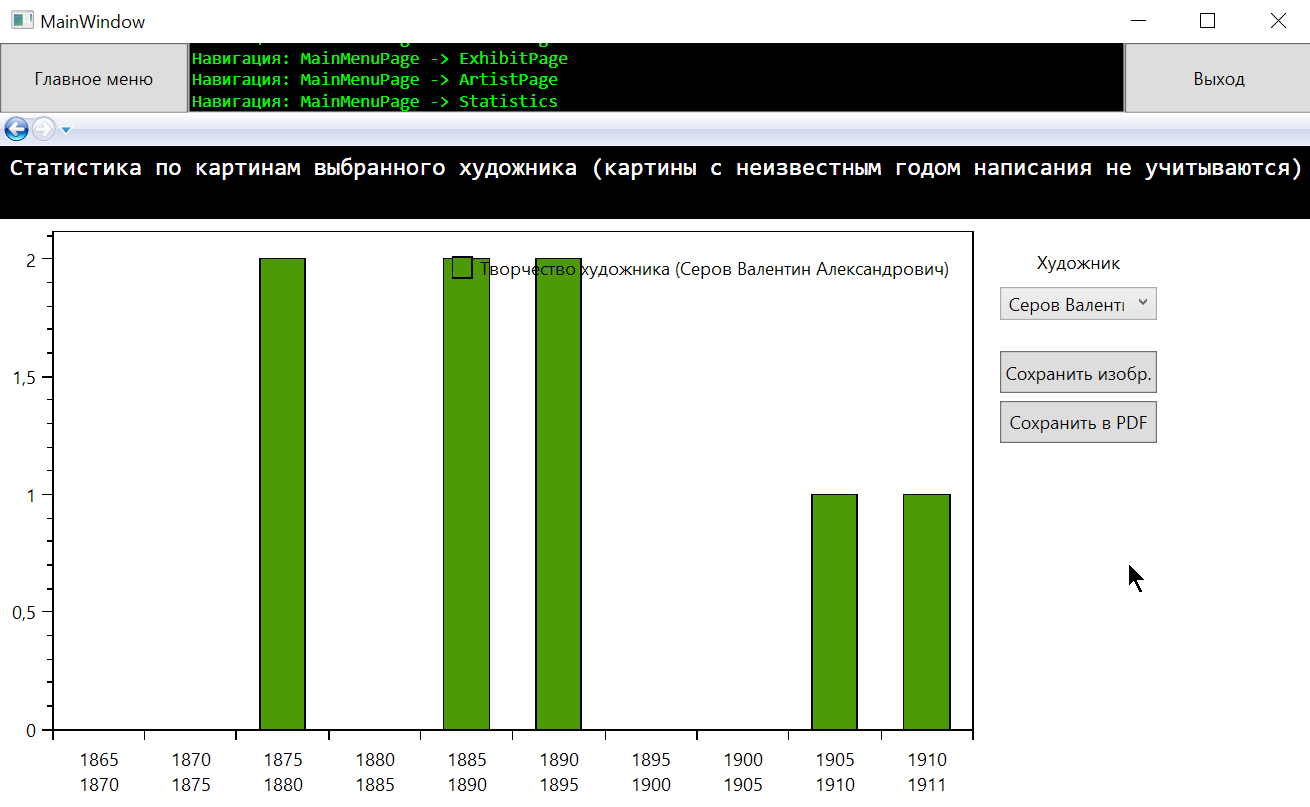


Рисунок 4.9 Графік статистики картин митців

б) Статистика по залах – інформація про кількість експонатів у залі, кількістб репродукцій серед них, головний напрям зали та ін.

в) Статистика по експонатах картин митця – є чи немає, де виставляється та ін.

Також перші дві статистичні таблиці можна сформувати у вігляді звіту, натиснувши відповідну кнопку «Отчет».

Запити, що використовувалися під час розробки додатку:

- Запит для формування статистики по залах (вибір головного напряму спрощено):

SELECT

Hall.name AS [Название зала],

COUNT(a.painting\_id) as [Всего экспонатов],

(SELECT COUNT(Exhibit.painting\_id) FROM Exhibit

WHERE Exhibit.is\_original = 1

AND Exhibit.hall\_id = a.hall\_id)

AS [Из них оригиналов],

COUNT(a.painting\_id)

- (SELECT COUNT(Exhibit.painting\_id) FROM Exhibit

WHERE Exhibit.is\_original = 1

AND Exhibit.hall\_id = a.hall\_id)

AS [И репродукций],

(SELECT MAX(paintingCount) FROM

(SELECT DISTINCT Painting.art\_movement\_id,

COUNT(Exhibit.painting\_id) AS paintingCount FROM Exhibit

JOIN Painting ON Exhibit.painting\_id = Painting.id

WHERE Exhibit.hall\_id = a.hall\_id

GROUP BY Painting.art\_movement\_id) AS qqq)

AS [Основное направление],

CONCAT(AVG(a.width), 'x', AVG(a.height)) AS [Средний размер экспоната, см]

FROM Hall JOIN Exhibit a ON Hall.id = a.hall\_id

GROUP BY Hall.name, a.hall\_id;

- Запит для формування статистики по експонатах картин митця:

SELECT

Painting.name,

CASE

WHEN Painting.id IN (SELECT Exhibit.painting\_id FROM Exhibit)

THEN '+'

ELSE

'-'

END,

CASE

WHEN Exhibit.is\_original = 1

THEN '+'

ELSE

'-'

END,

CASE

WHEN Exhibit.hall\_id IS NULL

THEN '-'

ELSE

CAST(Exhibit.hall\_id AS char)

END

FROM Painting LEFT JOIN Exhibit ON Exhibit.painting\_id = Painting.id;

* Запит на фільтрацію колекції експонатів (запит модульний, тобто користувач у додатку обирає, що включати до фільтру, а що – ні)

-Декларації змінних лише для прикладу.

DECLARE @use\_original\_filter BIT; SET @use\_original\_filter = 1;

DECLARE @use\_period\_filter BIT; SET @use\_period\_filter = 0;

DECLARE @use\_size\_filter BIT; SET @use\_size\_filter = 0;

DECLARE @from\_year INT; SET @from\_year = 2008;

DECLARE @to\_year INT; SET @to\_year = 2016;

DECLARE @from\_width INT; SET @from\_width = 0;

DECLARE @to\_width INT; SET @to\_width = 1000;

DECLARE @from\_height INT; SET @from\_height = 0;

DECLARE @to\_height INT; SET @to\_height = 1000;

SELECT

Exhibit.painting\_id,

Exhibit.date\_of\_acquiring,

Exhibit.width,

Exhibit.height,

Exhibit.is\_original,

Exhibit.canvas\_type,

Exhibit.paint\_type,

Exhibit.hall\_id

FROM Exhibit

WHERE

Exhibit.is\_original =

CASE

WHEN @use\_original\_filter = 1

THEN 1

ELSE

Exhibit.is\_original

END

AND Exhibit.date\_of\_acquiring BETWEEN

CASE

WHEN @use\_period\_filter = 1

THEN CAST(CAST(@from\_year AS CHAR) AS DATE)

ELSE

Exhibit.date\_of\_acquiring

END

AND

CASE

WHEN @use\_original\_filter = 1

THEN CAST(CAST(@to\_year AS CHAR) AS DATE)

ELSE

Exhibit.date\_of\_acquiring

END

AND Exhibit.width BETWEEN

CASE

WHEN @use\_period\_filter = 1

THEN @from\_width

ELSE

Exhibit.width

END

AND

CASE

WHEN @use\_original\_filter = 1

THEN @to\_width

ELSE

Exhibit.height

END

AND Exhibit.height BETWEEN

CASE

WHEN @use\_period\_filter = 1

THEN @from\_height

ELSE

Exhibit.width

END

AND

CASE

WHEN @use\_original\_filter = 1

THEN @to\_height

ELSE

Exhibit.height

END

;

4.6 Опис задачі автоматизації

Задача автоматизаціїї у цьому проекту складається з трьох частин:

а) побудова графіка розподілення написаних картин митця упродовж його життя, розділення на періоди по п’ять років, автоматичне вирівнювання границь інтервалів відповідно до дати смерті митця, або, якщо він ще живий, до поточного року. Спочатку період життя митця (до його смерті або до поточного року, якщо митець ще живий) ділиться на періоди по п’ять років, потім береться колекція картин цього митця, фільтрується від картин, для яких не вказано рік написання, потім групується по 5 років (для крайніх періодів окремо), потім для кожного періоду будується стовпець відповідної довжини, і з цих стовпців і будується графік;

private void DrawGraph(Artist artist)

{

StatModel.Series.Clear();

StatModel.Axes.Clear();

var categoryAxis = new OxyPlot.Axes.CategoryAxis

{

Position = AxisPosition.Bottom

};

var s1 = new OxyPlot.Series.ColumnSeries

{

Title = $"Творчество художника ({artist.full\_name})",

StrokeColor = OxyColors.Black, StrokeThickness = 1

};

var paintings = context.Paintings.Where(x => x.artist\_id == artist.id

&& x.year\_of\_creation != null)

.OrderBy(x => x.year\_of\_creation)

.ToArray();

if (paintings.Any())

{

var min = artist.date\_of\_birth.Value.Year;

var max = artist.date\_of\_death?.Year != null ?

artist.date\_of\_death.Value.Year :

artist.date\_of\_birth.Value.AddYears(120).Year;

int currIndex = 0;

for (int i = min; ; i += 5)

{

var nextBound = i + 5;

if (nextBound > max)

{

nextBound = max;

}

int count = 0;

while (currIndex < paintings.Length

&& paintings[currIndex].year\_of\_creation <= nextBound)

{

count++;

currIndex++;

}

s1.Items.Add(new ColumnItem(count));

categoryAxis.Labels.Add(i + Environment.NewLine + nextBound);

if (nextBound == max)

{

break;

}

}

var valueAxis = new OxyPlot.Axes.LinearAxis

{

Position = AxisPosition.Left,

MinimumPadding = 0,

MaximumPadding = 0.06,

AbsoluteMinimum = 0

};

StatModel.Series.Add(s1);

StatModel.Axes.Add(categoryAxis);

StatModel.Axes.Add(valueAxis);

StatModel.InvalidatePlot(true);

}

}

б) реалізація перехресного запиту. В додатку реалізовано перехресний запит за допомогою програмних засобів, що надані ADO.NET та мовою С#. Спочатку створюється порожня таблиця DataTable, додається стовпець, який буде зберігати ім’я митя, потім за допомогою запиту отримуються усі напрями мистецтва, що присутні у БД. Для кожного напряму у DataTable додається один стовпець із назвою, що співпадає з назвою напряму. Потім для кожного митця створюється рядок, що заповнюється даними про кількість картин для обраного напряму (номера стовпця), потім цей рядок додається у DataTable;

private DataTable GetCrossTable()

{

var pivot = new DataTable("Статистика по направлениям");

pivot.Columns.Add("Полное имя художника", typeof(string));

var artMovements = context.Art\_movement.ToArray();

foreach (var movement in context.Art\_movement)

{

pivot.Columns.Add(movement.name, typeof(int));

}

foreach (var artist in context.Artists)

{

DataRow entry = pivot.NewRow();

int columnNumber = 0;

entry[columnNumber] = artist.full\_name;

columnNumber++;

while (columnNumber < pivot.Columns.Count)

{

var paintingsCount = artist.Paintings.Where(x =>

x.Art\_movement == artMovements[columnNumber - 1]).Count();

entry[columnNumber] = paintingsCount;

columnNumber++;

}

pivot.Rows.Add(entry);

}

return pivot;

}

в) визначення головного (домінуючого) напряму мистецтва для зали, як частина статистичного запиту. Спочатку визначається параметр delta – це припустиме значення відхилення. Воно залежить від кількості експонатів у залі – якщо експонатів до 10, delta береться = 1, якщо від 10 до 100 – 2, якщо більше 100 – 5. Це – кількість картин, на яку один напрям повинен випереджувати інший, щоб стати головним. Тобто, якщо у залі 51 експонат, і 26 належать одному напряму, а 25 – іншому, то перший не є головним, бо кількість експонатів майже однакова. Потім для кожного експонату визначається його напрям, і відбувається группування – створення словника, зі напрямами в якості ключа, і кількістю картин цього напряму в якості відповідного значення. Потім він упорядковується за спаданням кількості, перші два елементи порівнюються – якщо різниця між ними менша за delta – повертається «-», як знак відсутності головного (домінуючого) напряму, якщо дорівнює або більша за delta – повертається значення першого елементу;

private string GetMainMovement(IList<Exhibit> list)

{

if (list.Count < 2)

{

return "-";

}

int delta = list.Count < 10 ? 1

: list.Count > 100 ? 5

: 2;

var counts = list.GroupBy(x => x.Painting.Art\_movement)

.Select(x => new

{

MovementName = x.Key.name,

PaintingCount = x.Count()

})

.OrderByDescending(x => x.PaintingCount)

.ToArray();

if (counts[0].PaintingCount - counts[1].PaintingCount < delta)

{

return "-";

}

else

{

return counts[0].MovementName;

}

}

**ВИСНОВКИ**

В даному курсовому проекті була спроектована та створена база даних і інформаційна система, призначена для використання адміністраторами музею картин. Було закріплено знання та робота з СУБД MS SQL Server та ORM Entity Framework.

Для проекта був проведений концептуальний аналіз предметної області, який використовувався далі для створення бази даних та інформаційної системи.

Сама база даних була створена для зберігання різноманітної інформації про митців, їх картини, експонати, тощо.

Під час проектування бази даних були також вивчені перші три нормальні форми і застосовані на практиці, проведена нормалізація бази даних до третьої нормальної форми, що є компромісом між швидкодією і якістю структури бази даних.

Було розглянуто основні переваги СУБД MS SQL Server, яка стала найкращим кандидатом для виконання розроблюваної системи, а також вона була інтегрована в середу розробки Microsoft Visual Studio 2017.

У процесі створення програми було вивчено взаємодію технології .NET і СУБД MS SQL Server та Entity Framework.

Були виконані всі вимоги, які були перераховані в розділі постановки задачі.

**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ**

1. Завадський І.О. Основи баз даних [Текст] навч. посіб./ І.О. Завадський– К.: Видавець І.О. Завадський, 2011. – 192с.
2. Диго С.М. Бази даних: проектування та використання [Текст] навч. посіб./ С.М. Диго – М.: Финансы и статистика, 2005 – 518 с.

Зарицька О.Л. Бази даних та інформаційні системи [Текст] навч. посіб./ О.Л. Зарицька – Ж.: Вид-во ЖДУ ім. І. Франка, 2009. – 132 с.

4. WPF: Windows Presentation Foundation в .Net 4.5 с примерами на C# 5.0 для профессионалов [Електронний ресурс]/ Издательский дом «Уильямс» Режим доступа: URL: <http://www.williamspublishing.com/Books/978-5-8459-1854-3.html#ogl>

5. Новые возможности Visual Studio 2017 [Електронний ресурс]/ Microsoft Visual Studio Режим доступа: URL: https://www.visualstudio.com/ru/vs/whatsnew/