Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Дослідження консолідації і роботи з даними у LINQ/PLINQ

Виконав: студент VI курсу, групи СНнм-61
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

_____ Лабчук В.А.
(підпис) (прізвище та ініціали)

Керівник _____ Матійчук Л.П.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Никитюк. В.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ Цуприк Г.Б.
(підпис) (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«_27_» _травня_ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Лабчуку Віталію Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження консолідації і роботи з даними у LINQ/PLINQ

Керівник роботи Матійчук Любомир Павлович, д.е.н., професор кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 29 » листопада 2024 року № 4/7-1139

2. Термін подання студентом завершеної роботи 26 травня 2025 р.

3. Вихідні дані до роботи Наукові публікації про LINQ, покращення швидкодії у PLINQ та порівняння з SQL для консолідації даних на наукових конференціях XIII та XII 11-12 та 18-19 грудня

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1 Аналіз предметної області, консолідація інформації. 1.1 Консолідована інформація та її внесок в розвиток інформаційного суспільства. 1.2 Питання безпеки. 1.3 Оцінювання якості отримуваних даних. 1.4 Конкурентна розвідка. 2 Дослідження застосування LINQ та PLINQ для завдань консолідації інформації. 3 Обчислювальний експеримент. 3.1 Обробка даних паралельно, визначення швидкості. 3.2 Обробка медичних даних. 4 Охорона праці та безпека в надзвичайних ситуаціях. Висновки. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Титульна сторінка. 2 Актуальність дослідження. 3 Мета дослідження. 4 Об'єкт, предмет та наукова новизна. 5 Консолідація даних. 6 Класифікація даних. 7 Обробка даних різної структурованості. 8 Схема консолідації даних. 9 Оператори LINQ. 10 PLINQ 11 Порівняння продуктивності LINQ та PLINQ. 12 Стратегія обчислень і продуктивність. 13 Відкладене виконання. 14 Паралельна обробка даних. 15 Різнорідні джерела даних для консолідації. 16 Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Сенчишин В.С., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., ст. викладач		

7. Дата видачі завдання 29 листопада 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	29.11.2024	*Виконано*
2.	Підбір та опрацювання наукових публікацій, збір	02.12.2024-10.01.2025	*Виконано*
	даних по темі роботи		
3.	Виконання дослідження згідно теми кваліфікаційної	13.01.2025-14.03.2025	*Виконано*
	роботи		
4.	Оформлення розділу «Аналіз предметної області.	17.03.2025-28.03.2025	*Виконано*
	Консолідація інформації»		
5.	Оформлення розділу «Дослідження застосування	31.03.2025-11.04.2025	*Виконано*
	технологій LINQ/PLINQ для завдань консолідації		
	інформації»		
6.	Оформлення розділу «Обчислювальний експеримент»	14.04.2025-25.04.2025	*Виконано*
7.	Виконання завдання до підрозділу «Охорона праці»	28.04.2025-02.05.2025	*Виконано*
8.	Виконання завдання до підрозділу «Безпека в	28.04.2025-02.05.2025	*Виконано*
	надзвичайних ситуаціях»		
9.	Оформлення кваліфікаційної роботи	05.05.2025-09.05.2025	*Виконано*
10.	Нормоконтроль	12.05.2025-15.05.2025	*Виконано*
11.	Перевірка на плагіат	16.05.2025	*Виконано*
12.	Попередній захист кваліфікаційної роботи	19.05.2025	*Виконано*
13.	Захист кваліфікаційної роботи	28.05.2025	

Студент _____ Лабчук В.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Матійчук Л.П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дослідження консолідації і роботи з даними у LINQ/PLINQ // Кваліфікаційна робота освітнього рівня «Магістр» // Лабчук Віталій Андрійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СНнм-61 // Тернопіль, 2025 // С. 73, рис. – 35, табл. – 2, додат. – 3, бібліогр. – 59.

Ключові слова: консолідація даних, гетерогенні дані, LINQ, паралельна обробка, великі набори даних, уніфікація, стандартизація, дублювання.

Кваліфікаційна робота присв'ячена дослідженню консолідації через LINQ та паралельне виконання. В першому розділі кваліфікаційної роботи описані принципи та причини консолідації. Висвітлено структурованість різних джерел інформації. Розглянуто використання об'єднаних даних для конкурентної розвідки. Проаналізовано питання якості отримуваних консолідованих даних. В другому розділі кваліфікаційної роботи досліджено можливість застосування LINQ для завдань консолідації даних. Подано порівняльну характеристику в швидкості паралельної та послідовної обробки.

В третьому розділі кваліфікаційної роботи описано використання LINQ на практиці. Проаналізовано використання LINQ для консолідації значних за обсягом медичних даних. Проведено дослідження швидкості паралельної обробки для різної кількості ядер. Об'єкт дослідження: консолідація даних. Предмет дослідження: технологія LINQ в контексті консолідованих даних.

ANNOTATION

Study of data consolidation and data work in LINQ/PLINQ // The educational level "Master" qualification work // Vitalii Labchuk // Ternopil Ivan Pulyuy National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, SNnm-61 group // Ternopil, 2025 // P. 73, fig. – 35, tables – 2, annexes – 3, ref. – 59.

Key words: data consolidation, heterogenous data sources, LINQ, parallel processing, big datasets, unification, standartisation, duplicating.

Thesis is devoted to the development and studying of an consolidation resource of medical data with the help of LINQ and parrallelisation. консолідації через LINQ та паралельне виконання. In the first chapted main principles and reasons for condolidation are described. The structure of different heterogeneous sources are showed. Using of consolidation data for market and business showed as well. The question of quality of received data was analised. In the second chapter the possibility of using LINQ for the tasks of data consolidation was studied. Differences in speed for sequential and parallel processing is shown.

In the third chapter using of LINQ with real cases is shown. Using of LINQ for consolidation of big amounts of data is analysed. The studying of the speed of parallel processing for different amounts of cores is shown. Object of studying: data consolidation. Subject of studying: using of LINQ in context of data consolidation.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ОДД — Оперативне джерело даних.

СППР — Системи підтримки прийняття рішень.

API — (англ. Applied programming interface) — прикладний програмний інтерфейс.

CSV (англ. Comma-Separated Values) — текстовий формат значень, розділених комами.

JSON (англ. JavaScript Object Notation) — запис об’єктів JavaScript.

LINQ (англ. Language Integrated Query) — інтегрована мова запитів.

ORM (англ. Object Relational Mapper) — об’єктно-реляційний мапер.

PLINQ (англ. Parallel Language Integrated Query) — паралельна інтегрована мова запитів.

URI (англ. Uniform Resource Identifier) — уніфікований ідентифікатор русурсу.

URL (англ. Uniform Resource Locator) — єдиний вказівник ресурсу.

XML (англ. EXtensible Markup Language) — розширювана мова розмітки.

ЗМІСТ

ВСТУП

Актуальність теми. У міру збільшення обсягу сучасних даних і відсутності доступу до всіх даних у одному джерелі інформації, а також різноманітності форм подання однієї інформації, з’являється потреба консолідації даних та їх паралельної високопродуктивної обробки. Актуальним це є для різних сфер людської діяльності, зокрема медичної, комерційної тощо.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Магістр» є підвищення якості та зручності консолідації даних. Для досягнення мети потрібно виконати ряд завдань, зокрема:

– Проаналізувати стан досліджень в області консолідації.

– Знайти спосіб взаємодії з даними з джерел різної структурованості.

– Розробити продуктивне на великих наборах даних рішення.

Об’єкт дослідження консолідаційні процеси з гетерогенних джерел інформації.

Предмет дослідження використання LINQ для консолідації та паралелізація запитів для прискорення обробки даних.

Наукова новизна одержаних результатів кваліфікаційної роботи полягає у знайденні високопродуктивного способу якісної консолідації.

Практичне значення одержаних результатів. Знайдено зручний і продуктивний спосіб виконання консолідації даних з різнорідних джерел.

Апробація результатів магістерської роботи. Результати досліджень обговорювались на XIII міжнародній науково-практичній конференції (м. Тернопіль, 11-12 грудня 2024 р.) та XII науково-технічній конференції «Інформаційні моделі, системи та технології» (18-19 грудня 2024р.)

Публікації. Основні результати кваліфікаційної роботи опубліковано у трьох працях конференції (Див. додатки В).

Структура й обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку літератури з 59 найменувань та 3 додатків. Загальний обсяг кваліфікаційної роботи складає 66 сторінок, з них 56 основного тексту, що містить 35 рисунків та 2 таблиць.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ. КОНСОЛІДАЦІЯ ІНФОРМАЦІЇ

1.1 Консолідована інформація та її внесок в розвиток інформаційного суспільства

Консолідація інформації – це процес пошуку, акумуляції, оброблення та зберігання даних з метою подальшого використання в корисних цілях. Уперше вона стала предметом дослідження 70-х роках XX ст. у США (Міллер, 2003). Становлення цього напряму пов’язується з реакцією суспільства на активну міжнародну економічну співпрацю, розвиток ділового партнерства за здорової конкуренції. Тому з поняттям консолідація інформації пов’язана конкурентна розвідка (competitive intelligence) – вона була покликана одержати з декількох джерел дані, обробити їх та систематизувати, для того щоб на основі цих набутих знань приймати правильні управлінські рішення та вирішувати проблемні питання. Але на сьогодні консолідовану інформацію розглядають як у виробничій сфері, так і у невиробничій, а також у структурах спеціального призначення – особливістю такого аспекту є те, що частина даних тут подаються як таємні. В XXI столітті консолідована інформація – це вже невід’ємна складова корпоративної культури бізнес-партнерства, яка забезпечує інформацію і знання про бізнес. Внесок у встановлення консолідованої інформації як галузі знань зробили зокрема і вітчизняні науковці, однак сутність консолідованої інформації як домінантної складової в розвитку інформаційного суспільства ще не була предметом комплексного аналізу вчених [1].

Для використання інформації можуть бути різні бар’єри, які утрюднюють роботу з даними і їх розпізнавання і через які потрібна консолідація даних. Серед них можна виділити наступні [2]:

1 Велика кількість стандартів і форматів, зокрема тих, з якими користувач може бути не знайомий
2 Інформація різноманітними мовами
3 Експоненційний ріст інформації та її старіння
4 Інформація має обмежений режим доступу

На сьогодні світ сповнений величезним обсягом даних. Однак ці дані можуть надходити з різних джерел, а найголовніше в різних форматах, і це все навіть в межах одного бізнесу. Консолідація може включати в себе різні аспекти, опис яких приведено на рисунку нижче [3]:

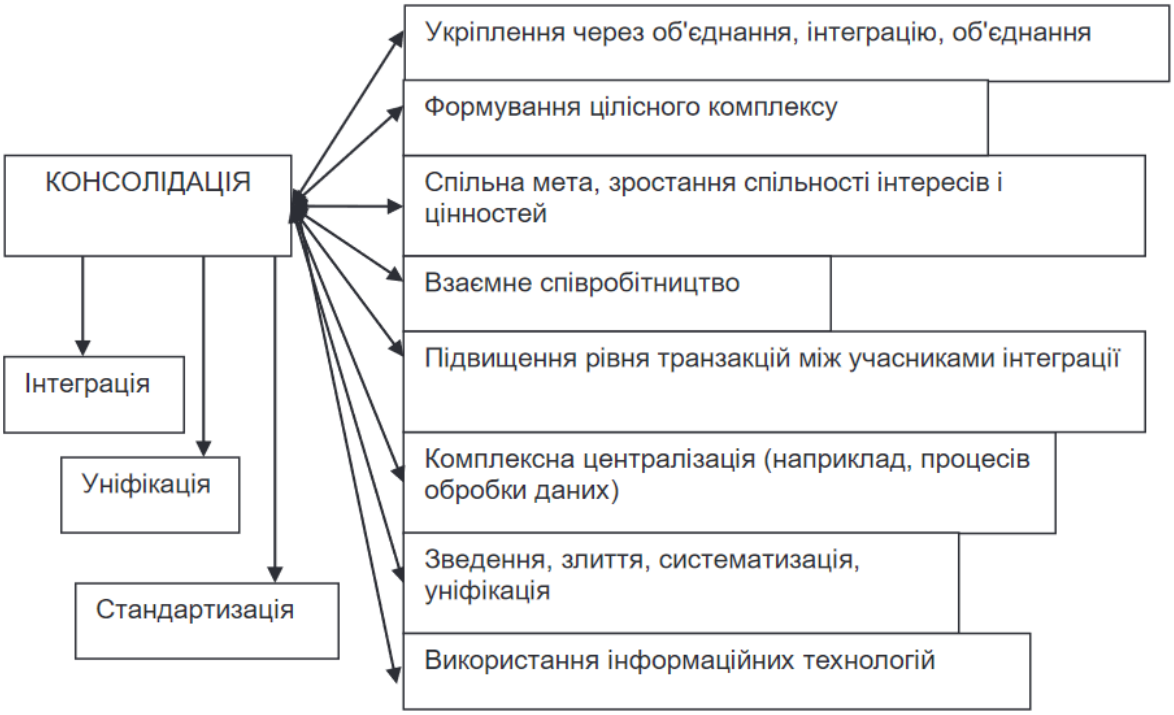


Рисунок 1.1 – Схема консолідації даних

Консолідація даних може відбуватися з використанням стохатистичних методів, зокрема пропорційного та рівномірного, а також скінченого чи повного автоматів. Скінчений автомат пам’яті передбачає обмежену кількість можливих внутрішніх станів, а повний автомат – необмежену кількість можливих внутрішніх станів. Рівномірний метод передбачає поділ кількості документів, що переглядалися, порівну між всіма джерелами. Пропорційний метод передбачає поділ кількості знайдених документів на загальну кількість [4].

Під час комбінування даних з різних ресурсів застосовуються методи [5]:

- Класифікації
- Регресії
- Кореляції
- Візуалізації

- Описової статистики.

Консолідація даних може призвести до розширення числа користувачів продуктом або сервісом. Однак слід пам’ятати, що консолідація не є панацеєю і є лише одним із методів для підвищення ефективності діяльності інформаційними зусиллями. Однак разом з тим метод є дуже ефективним. Але цей процес є дуже комплексним і досить складним, оскільки передбачає великі обсяги даних, різноманіття форматів і форм подачі, недоступність деякої інформації [6].

Рисунок 1.2 – Вимоги для консолідації хмарних обчислень

Консолідація може застосовуватися і для технологій хмарних обчислень і для цього є такі вимоги, щоб її застосувати [7]:

- Продуктивність – для уникнення деградації на великих вибірках;
- Масштабованість – для здатності системи підтримувати зростаючі навантаження через використовувані нові ресурси;
- Ефективність – асоціюється з гнучкістю і показує як хмарні ресурси можуть бути ефективно врегульовані для розширення і зменшення;
- Надійність – збільшення надійності заліза і сервісу;
- Доступність – для максимізації доступності сервісу, що може бути порушена внаслідок поломок або надмірних перевантажень обладнання.

Для консолідації потрібно виконати декілька завдань. Серед них [8]:

- Системний аналіз предметної сфери – ідентифікація головного об’єкту завдання, переваг, недоліків, а також розгляд даних, що характеризують дану предметну сферу.
- Вибір інструментів і засобів розв’язання проблеми — такі інструменти, які максимально б задовольняли вирішення проблем користувача, в контексті даної роботи це LINQ.
- Практична імплементація — реалізація високопродуктивних запитів до змішаних наборів даних за допомогою PLINQ з використанням багатоядерного апаратного забезпечення.

Мета і процес створення консолідованого ресурсу показані на рисунку нижче [9]:

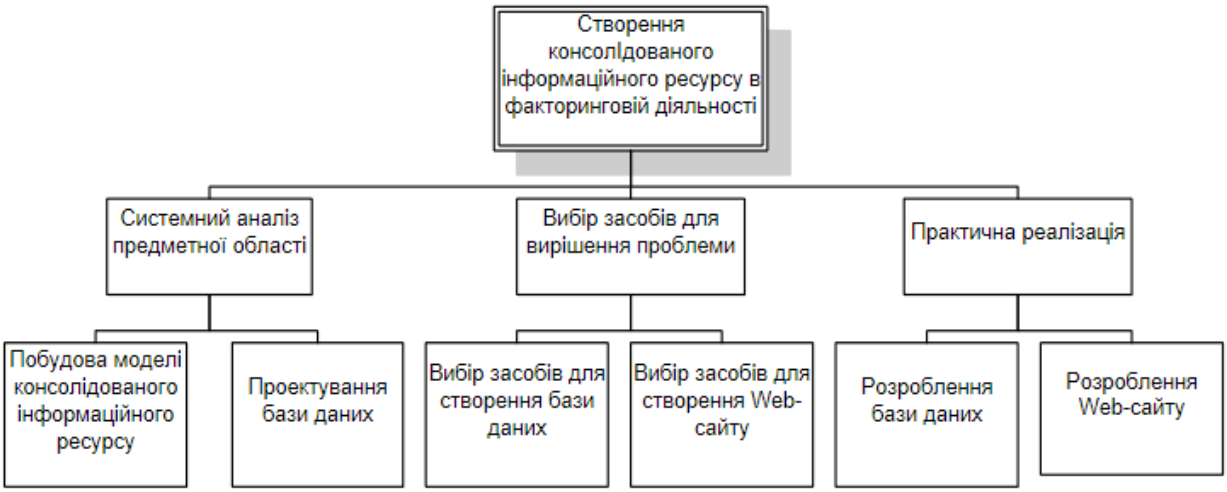


Рисунок 1.3 – Мета і процес створення консолідованого ресурсу

Крім безпосереднього збору і стандартизації інформації під час консолідації її потрібно ще також упакувати чи переупакувати в певний вигляд для донесення до користувача, якому вона призначена, в зрозумілому для нього вигляді і зрозумілою для нього мовою. Це може включати використання якогось з варіантів піднесення інформації, як-от: візуальний, аудіо тощо. Тобто загалом вона має бути дружня до користувача і такою, щоб користувачеві зручно з нею було працювати і нею оперувати [10]. Загалом існують певні вимоги до

користувацьких інтерфейсів (графічних або консольних) консолідованих інформаційних ресурсів [11]:

- *Концептуальність*. Має надаватися єдність інтерфейсу, незважаючи на те що джерел інформації декілька і кожне з них могло мати різні формати представлення однакової інформації.
- *Гнучкість*. Передусім для забезпечення доступу до необхідної для користувача інформації різними способами.
- *Пізнавальність*. Інтуїтивність та простота відображення наданого функціоналу.
- *Підтримуваність*. Підтримка можливостей до взаємодії з іншими аналогічними проєктами.
- *Здатність до розширень в майбутньому*. Адже можуть зрости вимоги до продукту, а отже і перелік доступних фукнцій та можливостей.

Консолідована інформація збирається для різних цілей. Однією зі сфер її застосування є системи підтримки прийняття рішень (СППР), що працюють з великими наборами даних і допомагають на основі наявної інформації дуже значних обсягів здійснювати аналітику та приймати правильні рішення. Але вони передбачають оперування різними видами даних, класифікація яких наведена на рисунку нижче [12].

Рисунок 1.4 – Класифікація даних для консолідації інформації

Але залежно від структури даних ці дані можна поділити на 3 класи [12, 13]:

- Структуровані – часто зберігаються в базах даних, тому є обробленими. Мають чітку структуру, формати, правила заповнення. Наприклад, в медичній сфері це можуть бути результати лабораторних аналізів.
- Неструктуровані – зазвичай поступають з гетерогенних джерел, що містять комбінацію простих текстових файлів, картинок, відео, матеріали з камер відстеження на дорогах для розпізнавання транспорту тощо. Характеризуються складнощами для обробки та вилучення корисної інформації звідти. В медичній сфері це можуть бути висновки лікарів та будь-яка інформація, що записана у вигляді довільного тексту.
- Напівструктуровані – зазвичай визначаються у файлах формату XML, JSON. Деколи вони можуть містити чітку структуру, однак передбачають більше гнучкості на відміну від баз даних. Тобто

визначені деякі правила та формати, але у дуже узагальненому вигляді.

Чим вища структурованість даних, тим легше їх обробляти автоматизовано, зокрема й засобами LINQ. При потребі та за можливості може проводитися покращення структурованості даних або їх синтаксичний аналіз [13].

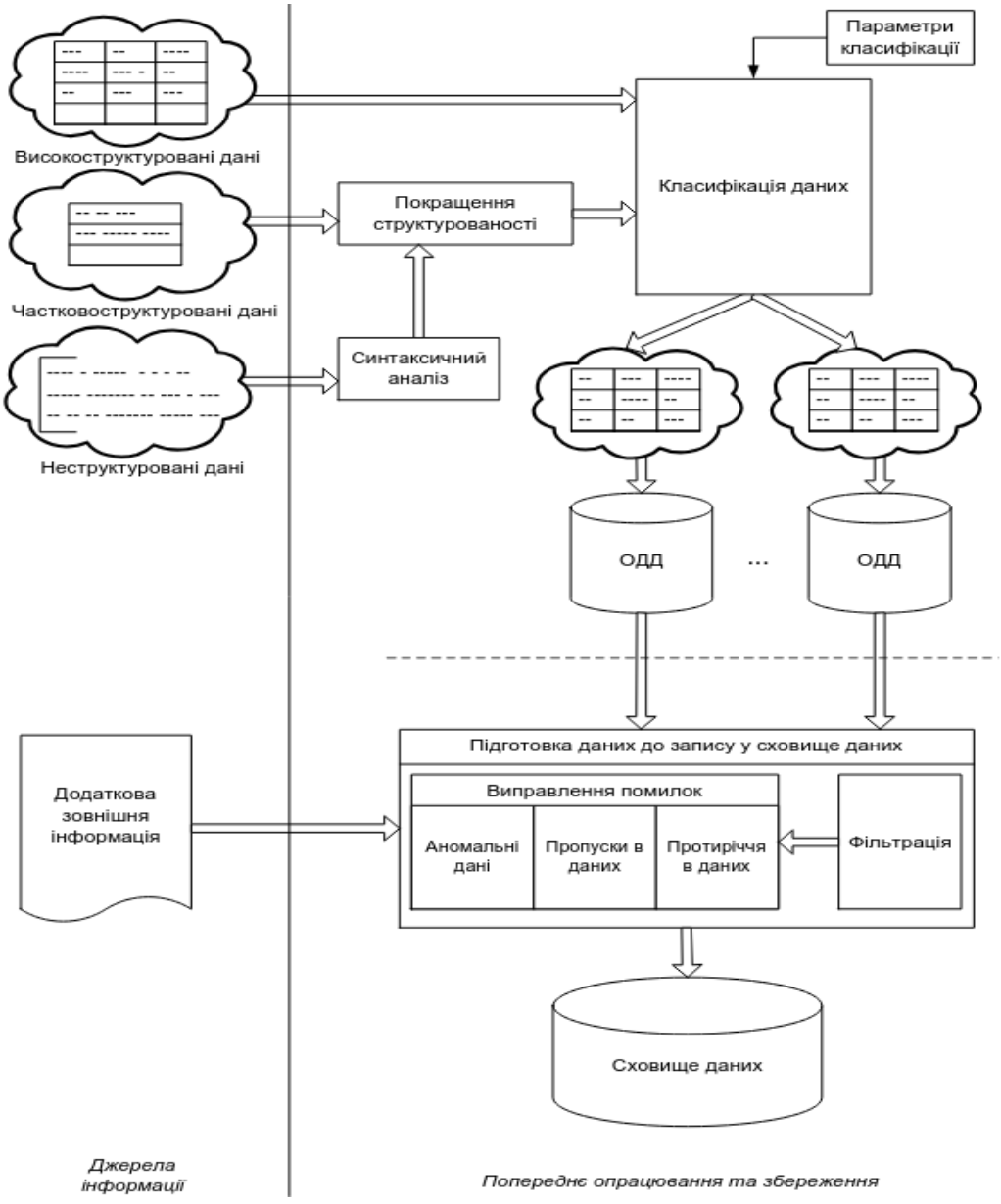


Рисунок 1.5 – Обробка даних різної структурованості

Серед джерел, звідки можна збирати інформацію є різні варіанти. Один з таких – державні дані. Державні органи збирають значні масиви даних, наприклад, про реєстрацію проживання, майна, цивільних станів, податків тощо.

Але згідно із законом “Про доступ до публічної інформації” ці дані мають бути публічними (окрім конфіденційних, службових та таємних), отже їх можна отримати. Крім того деякі дані з недержавних можуть бути відкритими, тобто open-source. З допомогою даних Укрзалізниці була створена мапа пасажирських перевезень поїздами, за допомогою якої виникло нове розслідування – “Янукович-тревел”. Це вдалося зробити через збір й аналіз масивів інформації про станції прибуття, відправлення, а також наявності нерегулярного сполучення однієї зі столичних станцій. Масиви таких даних можуть бути великими, що передбачає використання Big data, адже згідно з законом Мура обчислювальні потужності зростають експоненційно [14].

При роботі з джерелами даних постає питання аномальності даних. Аномальні стани можна віднайти у тестових наборах даних за допомогою наприклад методів машинного навчання. Однак не можна допускати ситуацій негативно-позитивних результатів, коли якісь дані аномальними не є але визначаються такими. Пошук аномалій особливо важливий в умовах консолідації даних, якщо застосовуються дані з максимально великої кількості джерел, деколи зокрема і ненадійних, тобто таких, які є фактично новими невідомими даними — це створює певні виклики, але й збільшує ступінь охоплення даних. Для виявлення аномалій може застосовуватися метод ізоляційного лісу, де припускається, що аномалій небагато та вони різні [15].

1.2 Питання безпеки

Разом із консолідацією інформації особливо актуальним стає і питання безпеки. Часто люди не приділяють цьому належної уваги і їх дані опиняються в руках у тих, у кого вони не підозрювали і до кого не хотіли, щоб вони потрапляли. Наприклад, співробітники спецслужб під час своєї професійної діяльності колекціонують та обробляють великий набір даних. Однак варто відзначити, що практично 85% цієї інформації отримується з відкритих джерел. Зокрема це може бути відкрита сторінка у facebook або іншій соціальній мережі. До того ж facebook пропонує API, що дозволяє не просто побачити ці дані, але й

отримати їх стосовно різних користувачів в структурованому вигляді. Але тій стороні, яка в той чи інший спосіб консолідує інформацію, теж треба забезпечувати безпеку консолідованої інформації. Зокрема, по можливості надавати доступ до таких даних лише тим, кому вони дійсно призначені. Також потрібно враховувати і можливість видобування даних з веб-сторінок бізнесом конкурента, наприклад, хоча початково ці дані могли призначатися для того щоб показати цінність фірми користувачу продукції (наприклад, компанія має вже офіси в 15 містах України, виробничі потужності здатні виготовляти 2000 одиниць товару на рік тощо) [16].

На основі даних, які явно не обмежуються в доступі, насправді часто можна зібрати і проаналізувати ту інформацію, яку б власники цих даних не були б раді бачити.

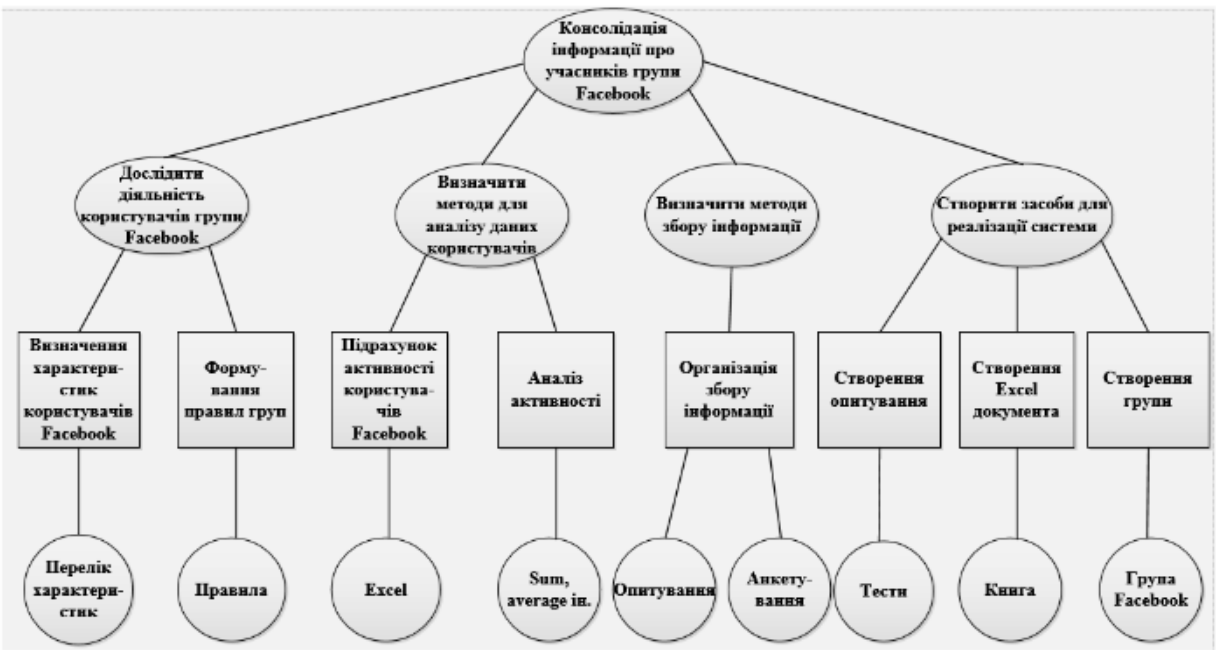
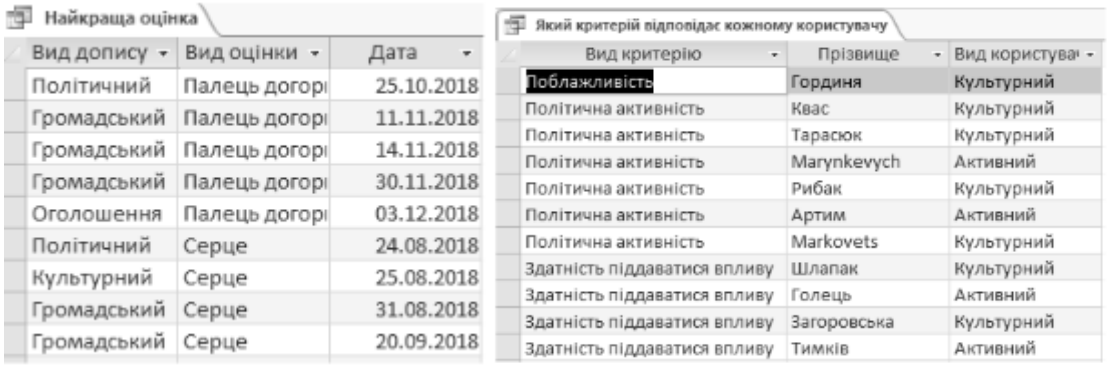


Рисунок 1.6 – Консолідація даних про учасників груп з facebook

Наприклад, створивши групу про проблеми якогось міста і дочекавшися там певної кількості користувачів (для цього групу можна рекламувати), можна отримувати інформацію про профілі користувачів (це може бути ім’я, прізвище, фото, місце проживання, в залежності від того до чого вони самі надали доступ) через їх коментарі, а також аналізувати їх прихильність до дописів групи, в

залежності від того яку реакцію вони ставлять. Тим паче що facebook надає власне API.



Найкраща оцінка

Вид допису	Вид оцінки	Дата
Політичний	Палець догор	25.10.2018
Громадський	Палець догор	11.11.2018
Громадський	Палець догор	14.11.2018
Громадський	Палець догор	30.11.2018
Оголошення	Палець догор	03.12.2018
Політичний	Серце	24.08.2018
Культурний	Серце	25.08.2018
Громадський	Серце	31.08.2018
Громадський	Серце	20.09.2018

Який критерій відповідає кожному користувачу

Вид критерію	Прізвище	Вид користува
Поблажливість	Гординя	Культурний
Політична активність	Квас	Культурний
Політична активність	Тарасюк	Культурний
Політична активність	Marynkevych	Активний
Політична активність	Рибак	Культурний
Політична активність	Артим	Активний
Політична активність	Markovets	Культурний
Здатність піддаватися впливу	Шлапак	Культурний
Здатність піддаватися впливу	Голець	Активний
Здатність піддаватися впливу	Загоровська	Культурний
Здатність піддаватися впливу	Тимків	Активний

Рисунок 1.7 – Отримані та структуровані дані при аналізі користувачів facebook та їх поведінки

Результати такої консолідації можуть використовуватися і в негативному сенсі міськими головами, політиками тощо. Також на основі цієї інформації можна транслювати потрібну та вигідну для себе інформацію цій аудиторії [17].

1.3 Оцінювання якості отриманих даних

Консолідовані дані отримуються з різних джерел, що вважається цінним. Однак ця інформація має бути цілісною і несуперечливою між собою. Отже, постає питання забезпечення якості отриманої інформації. Якісна консолідована інформація має відповідати вимогам користувачів та бути придатною до підтримки прийняття рішень. Навіть якщо якість даних з кожного окремого джерела є задовільною, то якість сукупності цієї інформації з різних джерел може різко падати. Це спричинено різним поданням в різних джерелах, неузгодженістю форматів тощо. Для опрацювання різнотипних інформаційних джерел можуть застосовуватися простори даних – вони містять поділ даних на структуровані (сховища даних, базиси), напівструктуровані (електронні таблиці, xml) та неструктуровані (текст) [18].

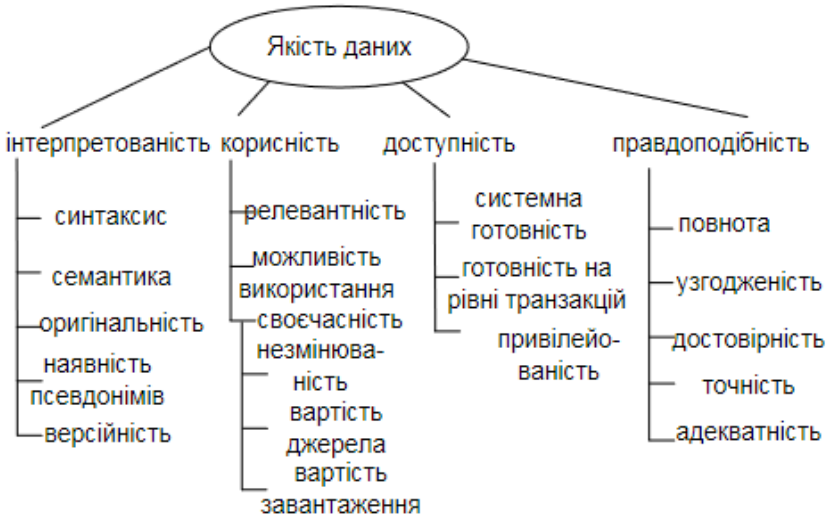


Рисунок 1.8 – Складові якості даних

Однією із проблем у процесі консолідації є невизначеність даних, яка виникає в результаті дублювання, неточності, відсутності, протиріччя даних [19].

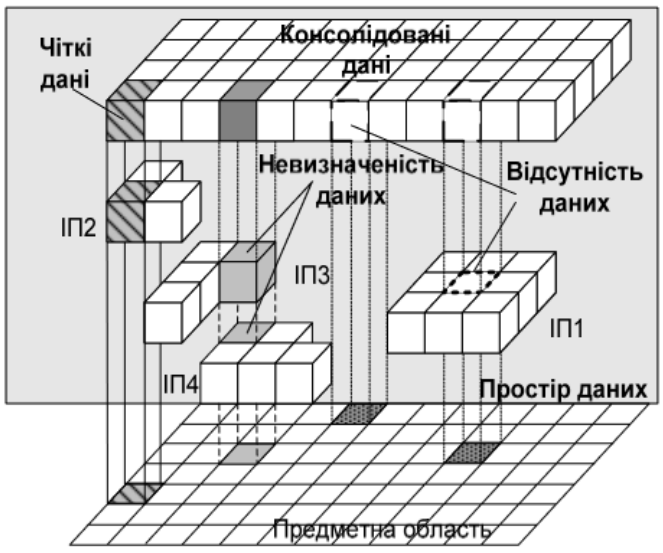


Рисунок 1.9 – Схема консолідації даних

Невизначеності не дозволяють мати однозначного уявлення про частину даних і тому погіршують якість отримуваних даних, що в подальшому впливає на користь прийнятих рішень на основі таких даних. Тому невизначеності по можливості слід зменшувати. Для цього слід розуміти які саме існують типи невизначеностей, адже для кожного з них існують свої методи боротьби. Отже, існують наступні типи невизначеностей [20]:

- Відсутнє значення, тобто невідоме (використовуються NULL або так звані nullable типи даних). У LINQ існують відповідники багатьох звичайних методів з суфіксом «OrDefault» для роботи з такою інформацією, наприклад, FirstOrDefault, LastOrDefault тощо.
- Неповнота інформації.
- Ненадійність даних — наприклад, взяті з ненадійного джерела даних.
- Неточність (числових даних) — наприклад, заокруглення.
- Нечіткість (стохастичність) — використовують розподіли для встановлення істиності.
- Недетермінованість (випадковість).
- Багатозначність інтерпретації.
- Лінгвістична невизначеність — при роботі з текстовими даними, оскільки природня мова передбачає різноманіття конструкцій та певну складність для алгоритмічної обробки.

У просторах даних самі дані, а також їхнє зберігання та опрацювання набувають домінуючого значення. Можна виділити наступні показники якості даних:

- Функціональна придатність – повнота накопичених об’єктів.
- Коректність (достовірність) – ступінь відповідності даних про об’єкти в базі даних реальним об’єктам, з врахуванням реального часу.
- Використовуваність ресурсів – наскільки економічно використовується пам’ять, процесор тощо.
- Практичність – застосовуваність консолідованих даних для певного користувача, корисність від прийнятих рішень з використанням цих консолідованих даних.
- Супроводжуваність – зручність і ефективність внесення змін, адаптації структури тощо.
- Мобільність – тривалість і трудомісткість інсталяції продукту на іншу платформу, а також адаптації і заміни.

При консолідації інформації важливо не лише отримати масив даних, але й забезпечити їх якість, адже якщо ці дані будуть невірними і будуть призводити до прийняття неправильних рішень, то навіщо нам такі дані? Корисність даних залежить від ступеня довіри до джерела даних.

$$Trust_j = \frac{\sum_{i=1}^{m}(Trust_k(i,j))}{(n*m)}, \quad (1)$$

де n – кількість звернень користувача до ресурсу; m – кількість користувачів, що звертались до ресурсу; $Trust_k(i,j)$ – значення лінгвістичної змінної, що відображає довіру i-го користувача до j-го джерела даних при k-ому зверненні.

Рисунок 1.10 – Формула визначення користувача до джерела даних

Базовими показниками якості відповідно до стандарту ISO 9126 є функціональна придатність до використання, коректність (достовірність), практичність, ресурсна економічність, супроводжуваність, мобільність. Загалом якість консолідованих даних – інтегральна характеристика, що відображає повноту накопичення даних, коректність, мобільність та корисність прийнятих рішень [21]

$$s_1 quality + s_2 v(r) \rightarrow Max, \quad (2)$$

де *quality* – інтегральний безрозмірний показник характеристик якості даних, 0≤quality≥1, s_1 – коефіцієнт важливості повноти накопичення даних; $v(r)$ – значення багатовимірної функції корисності; s_2 – коефіцієнт важливості якості прийнятих рішень, $s_1+s_2=1$.

Рисунок 1.11 – Формула визначення якості консолідованих даних

Процес оцінювання якості консолідованих даних складається з трьох стадій: встановлення вимог до якості консолідованих даних (адже ми повинні розуміти що можна вважати якісним, а що ні), підготовка до оцінювання, процедура оцінювання [22].

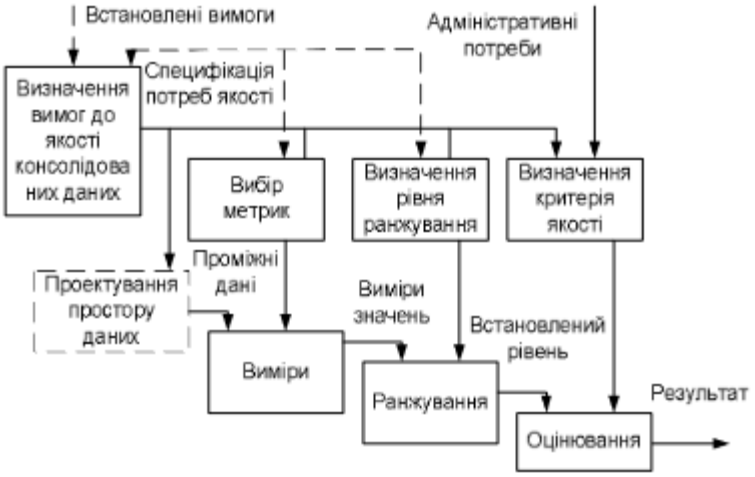


Рисунок 1.12 – Схема процесу оцінювання якості консолідованих даних

Підсистема забезпечення і підтримки якості даних в просторі даних слугує для реалізації алгоритмів і процедур, які забезпечують оцінку якості даних, збір та обробку інформації для підтримки якості даних. Тобто до складу підсистеми входить аналітичний центр, засоби збору і передачі даних, а джерелами є інформаційні продукти [23].

1.4 Конкурентна розвідка

За допомогою консолідованої інформації можливо будувати соціальні портрети, що дозволяє підвищити оперативність прийняття управлінських рішень, а також покращити якість надання послуг. Це одна зі складових успіху у висококонкурентному середовищі [24].

Під час консолідації інформації важливо також враховувати правовий аспект і консолідувати його, адже правова інформація теж потребує консолідації. Зокрема важливо дотримуватися в межах закону під час видобування даних засобами Data mining [25]. Конкурентна розвідка має і свої межі легітимності, про що буде описано далі [26].

На сьогодні світова економіка розвивається так, що набувають поширення тенденції взємопроникнення та взаємодоповнення окремих складових різного роду економічних об’єктів, які раніше не вступали в тісну взаємодію. Інтеграційні процеси стає вагомим фактором успіху. Облікова інформація

відіграє значну роль в управлінні діяльністю підприємств та об'єднань, це ядро їх інформаційно-аналітичного забезпечення. Для успішного ведення бізнесу керівництво потребує різного характеру даних у потрібний час, в потрібному місці. Це потрібно для розуміння повної картини ситуації та прийняття правильних рішень. Впливає на це і процес глобалізації, який тільки посилюється. Однак важливим є враховувати при консолідації вид та специфіку діяльності інформації, хоча й консолідація звісно може застосовуватися для підприємств різних сфер діяльності. Але врахування галузі діяльності значно покращить облікову складову інформаційного забезпечення менеджменту підприємства і дозволить отримати більш якісний аналіз [27].

Часто консолідація інформації використовується підприємствами для конкурентної розвідки, оскільки без цього дуже важко ефективно працювати, а можливо, й існувати взагалі, якщо тільки ви не є монополістом або держава штучно не забезпечує вам такий статус.

Рисунок 1.13 – Мета конкурентної розвідки

Джерелами розвідувальних даних можуть бути: документація і звітність, зокрема перед органами влади і місцевим самоврядуванням, відслідковування

поведінки, дій конкурентів, зразки продукції та макети, експерти та інші люди з певною інформацією [28].

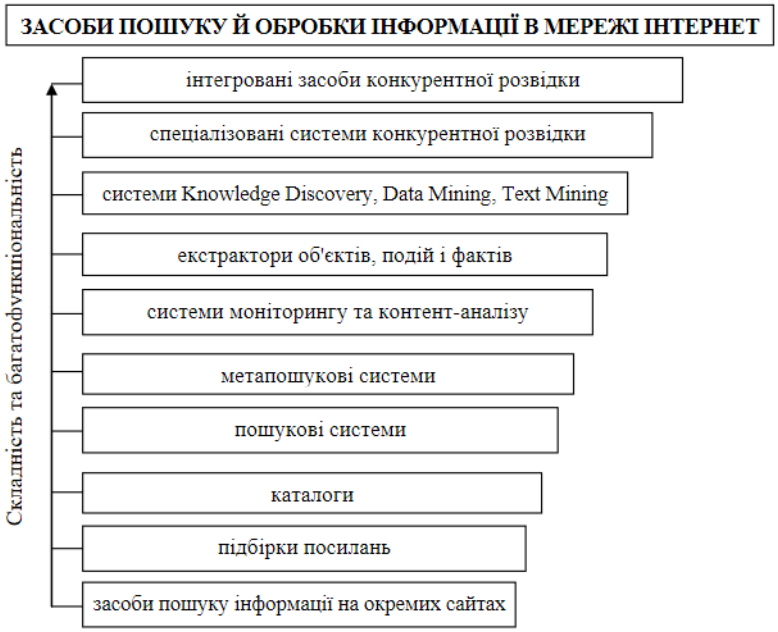


Рисунок 1.14 – Засоби пошуку й обробки інформації в інтернеті для здійснення конкурентної розвідки

Також конкуренція між підприємствами може бути не лише на національному рівні, але й на міжнародному. Існують транснаціональні компанії (ТНК), які діють глобально. Крім того з метою підвищення загальної конкурентоспроможності потенційні суперники можуть об’єднуватися, щоб спільними зусиллями перемагати компанію конкурента. Але тоді консолідація даних ще необхідніша, але й ще складніша, адже взаємодіють різні компанії між собою. Крім того в різних країнах можуть бути зовсім різні стандарти, закони, зрештою інша мова. Звісно що і обсяг самої інформації може сягати петабайти даних. Мікель Б. Расмусен вказує, що сучасні бази даних – це лише верхівка льодовика, оскільки для успішного менеджменту корпорацій доводиться оперувати значно більшими об’ємами даних. Однак консолідація інформації підприємством відрізняється від “промислового шпигунства”, оскільки під час консолідації процеси і процедури мають здійснюватися виключно згідно законодавства, тобто дозволяється застосування лише легальних форм та методів такої діяльності. Крім того, варто враховувати що у інших компаній може бути

комерційна таємниця, а отже намагання добути дані про неї може вже переростати у промислове шпигунство [29].

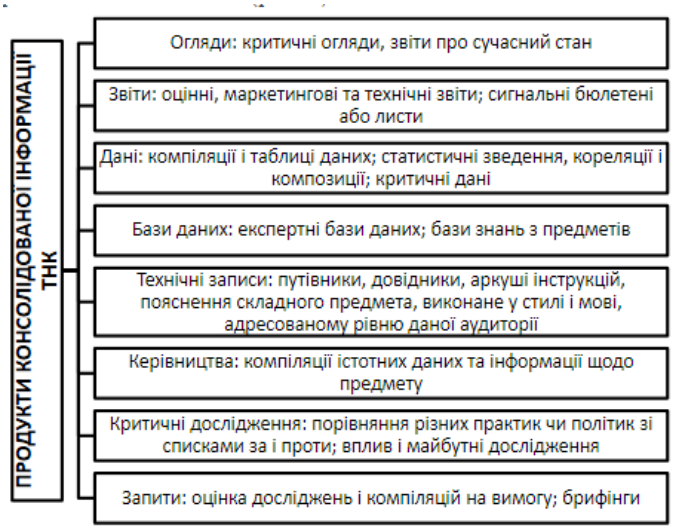


Рисунок 1.15 – Продукти консолідованої інформації транснаціональних корпорацій

За кордоном саме по собі промислове шпигунство не вважається злочином і не є підставою для кримінальної відповідальності. Але якщо в процесі розкрадання секретної інформації виникає збиток установі чи працівнику, то особа підлягає кримінальному покаранню саме за збиток, а не за факт розкрадання цінних відомостей. І це дозволяє зосередитися на охороні своєї комерційної таємниці, а не вдаватися до кримінально-правового кодексу [30].

Особливо актуальна конкуренція між великими підприємствами. Наприклад, IBM налічувала 2500 конкурентів у 1965 році та 50000 вже у 1995. Тому консолідація в такому випадку є складною багатоступеневою процедурою і найважливішою складовою аналітичного процесу для забезпечення високого рівня аналітичних рішень у діяльності підприємства, що надзвичайно важливо при такій кількості конкурентів [31].

Консолідація даних на великому підприємстві зображена на рисунку нижче

Рисунок 1.16 – Організаційно-структурна схема консолідації інформації підприємства

Існують наступні причини у консолідації інформації на підприємствах:

- Підвищення надійності роботи корпоративних програм
- Зниження втрат від простою інформаційних систем
- Скорочення бюджету на обслуговуванні IT-інфраструктури
- Необхідність підвищення якості пропонованих послуг
- Плановий розвиток, удосконалення діючих інформаційних систем

Рисунок 1.17 – Схема інтеграції складових управління в процесі консолідації інформації

Можна зазначити, що з точки зору споживача інформації, не існує ніяких сховищ інформації, форматів, структур зберігання, а лише сервіс, де ця інформація надається, тому інформація – це сервіс. Отже, з точки зору користувача вся інформація має бути уніфікована, приведена до єдиного формату, вигляду [32].

1.5 Висновок до першого розділу

В першому розділі кваліфікаційної роботи освітнього рівня «Магістр» описано консолідацію даних в цілому, без прив’язки до конкретних інструментів. Розглянуто проблеми та виклики, які можуть виникнути, перевірку якості отриманих результатів та роботу з невизначеностями, завдання та сфери для яких можна застосовувати. Питання консолідації інформації часто пов’язане з конкурентною розвідкою та комерційною вигодою. Однак часто консолідація вимагає значних обчислювальних потужностей в силу значних розмірів даних.

2 ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ ТЕХНОЛОГІЙ LINQ ТА PLINQ ДЛЯ ЗАВДАНЬ КОНСОЛІДАЦІЇ ДАНИХ

2.1 Дослідження можливостей застосування Microsoft LINQ для задач консолідації даних

Однією із задач у процесі консолідації є невизначеність даних, яка виникає в результаті дублювання, неточності, відсутності, протиріччя даних. Звісно для роботи з цим на практиці у LINQ можна застосовувати nullable-типи, здатні оперувати значенням null, а також використовувати методи, які не генеруватимуть виключень при відсутності значення, а повертатимуть значення за замовчуванням (як-от, FirstOrDefault, LastOrDefault, SingleOrDefault, де ключовими є «OrDefault»). Такого роду проблеми з даними виникають через обмежений або деколи й відсутній фізичний доступ до деяких даних, різний процес збору даних в різних джерелах тощо [33].

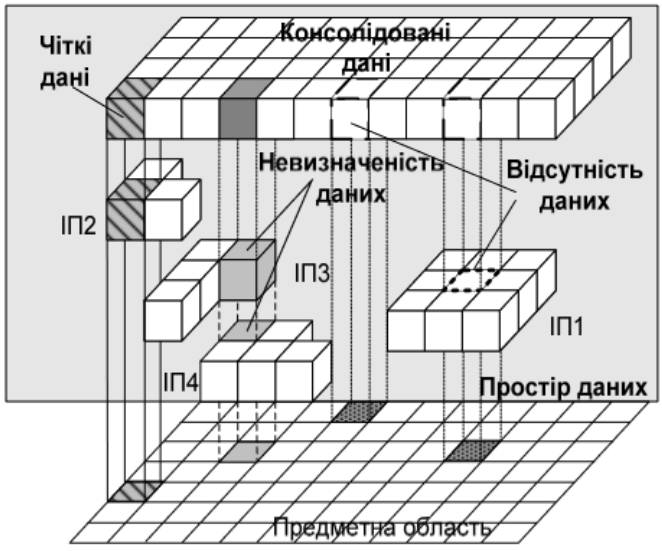


Рисунок 2.1 – Схема консолідації даних

При консолідації даних з різних ресурсів постає питання роботи з даними різних форматів і приведення їх до єдиного. LINQ вміє працювати з різними джерелами даних, зокрема отриманих у вигляді наборів даних (зокрема з CSV-файлів), сутностей (описуються класами), баз даних, xml-файлів тощо. Для цього

існують відповідні імплементації LINQ to SQL, LINQ to XML, LINQ to Dataset, LINQ to Entities. Хоча на сьогодні велика частина інформації зберігається у базах даних, однак звісно далеко не вся [34].

Тобто технологія підтримує парадигму ROX (relations, objects, xml), що передбачає надання стандартизованого інтерфейсу для запитів до реляційних структурованих даних з баз даних (LINQToSQL, EntityFramework), об'єктів (LINQToObjects) та даних XML (LINQToXML). Таким чином LINQ абстрагує розробника від того, звідки саме беруться дані і якої вони форми подання. Для нас існують просто дані [35].

Для роботи з LINQ може використовуватися два C# інтерфейси — IEnumerable та IQueryable. Перший підходить для ітерації колекцією об'єктів, другий — для зовнішніх ресурсів, наприклад роботи з базою даних. В даному випадку LINQ не обмежується лише Microsoft SQL Server, але й може працювати з іншими постачальниками даних. Тобто фактично LINQ створює абстракцію рівня доступу до даних. При цьому запити можна створювати в SQL-подібному стилі або ж за допомогою методів розширення. В першому випадку код нагадує SQL, але з деякою різницею в порядку операторів. В другому ж випадку методи розширення викликаються через крапку після виклику колекції. Microsoft рекомендує застосовувати саме методи розширення в ситуаціях коли це можливо [36].

```
var arrs List<List<...>> = ints.AsParallel().Select(i int => new
{
    number = i,
    divisors = ints.Skip(1).Take(ints.Count - 3).Where(x int => i % x == 0).ToList()  //методи розширення
}).Select(a (number,divisors) =>a.divisors).ToList();
var arrs2 ParallelQuery<int> = from i int in ints.AsParallel()  //sql-стиль
    orderby i ascending
    where i>13
    select i;
```

Рисунок 2.2 – Порівняння стилів запису операторів LINQ

Насправді на цьому можливості LINQ не закінчуються і є можливість інтегруватися і з іншими технологіями для отримання даних. Однак деякі зі

специфічних розширень можуть постачатися третіми сторонами. Зокрема можна згадати LINQ to Twitter для взаємодії з відповідною соцмережею, LINQ to Result (для підтримки залізнично-орієнтованого програмування в мові C#), LINQ to Anything. Останній варіант дозволяє перетворити будь-яке джерело даних у IQueryable для зручної взаємодії з ним [37].

У LINQ неважливо працюємо ми з масивами, списками, об'єктами бази даних чи файлами xml та json. До всіх них ми можемо застосовувати доступні методи розширення, зокрема фільтрацію, сортування, агрегатні функції, вибір полів тощо. Наприклад, LINQ дозволяє обробляти дані з xml чи з бази даних і порівнювати їх з масивом LINQtoObjects, який зберігається в пам'яті. Наприклад візьмемо 2 різних типи джерела, але обидвоє містимуть числові значення, які ми і обробляємо разом, шукаючи більше з них. Однак в першому випадку у нас це класові xml дані, об'єкти містять декілька атрибутів. В другому випадку ж просто масив чисел. І тим не менш, завдяки LINQ є можливість порівнювати їх між собою. Звісно, постає питання що саме порівнювати, якщо масив містить просто якісь скалярні значення, а об'єкти можуть містити безліч атрибутів. Порівнювати треба значення з масиву з атрибутом об'єктів який співпадає за типом даних (в даному випадку рік побудови будинку).

Лістинг 2.1 – Спільна робота з числовими даними LINQ to XML та LINQ to Objects

```
string xmlData = @"
<buildings>
<building name='Empire State Building' address='New York'
year='1931' />
<building name='Burj Khalifa' address='Dubai' year='2010' />
<building name='Eiffel Tower' address='Paris' year='1889' />
<building name='Chrysler Building' address='New York' year='1930' />
</buildings>";

// Load the XML Data
XDocument doc = XDocument.Parse(xmlData);

// Example 1: Filtering and Projection (Methods extension)
var modernBuildings = doc.Descendants("building")
    .Where(building => (int)building.Attribute("year") > 1900)
    .Select(building => new Building
    {
        Name = (string)building.Attribute("name"),
```

```
        Address = (string)building.Attribute("address"),
        YearBuilt = (int)building.Attribute("year")
    });

List<int> years = new List<int>(){2010,2015,1985,1913,1940};
var
newYears=years.Where(y=>y>=modernBuildings.Select(mb=>mb.YearBuilt
).Max());
```

Тобто фактично це означає що технологія дійсно підходить для поєднання даних з джерел різної структурованості, пропонуючи єдиний інтерфейс, а отже може використовуватися для консолідації. Тим паче що після лише збірки даних все одно отримуються як-не-як сирі дані. Щоб вони такими не були їх треба інтелектуально обробляти, і LINQ дозволяє це робити [8].

2.1.1 LINQ для консолідації розподілених даних

Однак також неправильним буде не зазначити обмеження фреймворку для виконання поставлених задач. До цього часу консолідація даних розглядалася як якесь локальне поняття, зі збереженням значень у внутрішній пам’яті. Якщо ж розглядати консолідацію як розподілений процес, який виконується не на одному комп’ютері, а на кластері, то потрібно шукати інші засоби для реалізації. В цілому якщо розглянути які технології доступні для розподіленої обробки даних, то можна виявити що існує розподілена версія PLINQ — DryadLINQ. Її завданням є перетворення запитів LINQ в Dryad-job, які виконуються на кластері і повертають результат програмі.

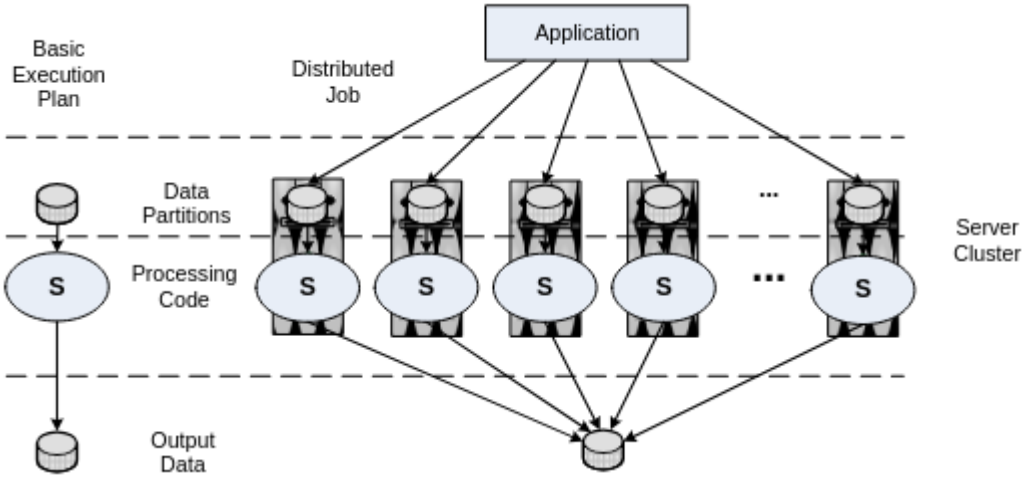


Рисунок 2.3 – Робота Dryad

Оператори DryadLINQ можна поділити на 3 категорії: немодифіковані оператори LINQ (Select, Where), модифіковані оператори LINQ (якщо в звичайному варіанті застосовуються такі оператори як First, FirstOrDefault, Last, LastOrDefault, то для Dryad це оператори FirstAsQuery, LastAsQuery тощо), специфічні DryadLINQ оператори (Apply, fork).

Тобто LINQ можна використовувати не лише в паралелізованому виконанні, а й навіть в розподілених системах. Але недоліком DryadLinq є те, що він не набув широкого поширення і поступається Hadoop або ж Spark і в комерції переважно не застосовується, а є більше для навчальних цілей [38]. Тобто на практиці в комерції LINQ не використовується для консолідації інформації в розподіленому контексті, лише в академічних цілях.

Актуальним в розподіленому використанні є застосування для цілей розумного міста. Це включає додатки для обробки великих даних, зокрема і в секторі охорони здоров’я — профілактика захворювань, діагностика лікувань, ведення медичної документації (що містить і неструктуровані дані) [39].

З іншого боку для інтеграції або консолідації розподілених даних нема універсального інструментарію не існує. Але для розподілених даних ефективнішим підходом буде «знизу вгору» - використовувати проміжне програмне забезпечення, а саме посередників (mediators) семантичної інтеграції через уніфікацію метаописів джерел даних та адаптерів які забезпечать зв’язок з включеними базами даних через процедуру віддаленого виклику [40].

2.2 Переваги та недоліки LINQ в порівнянні з SQL-кодом

Досить часто для зберігання і подальшого оперування даними використовуються бази даних та сховища даних. Для цього існує зокрема структурована мова запитів SQL, хоча дані можуть і зберігатися як noSQL. За допомогою неї можна працювати з різними таблицями бази даних та з різними базами даних в цілому, однак не все так легко. В таких питаннях постає питання «а чи взагалі є зміст користування інтегрованою мовою запитів?».

Використання LINQ надає багато переваг в порівнянні з аналогічними запитами SQL. Такі запити більш високорівневі та інтегровані в мову програмування. Звідси слідує, що такий код компільований, а отже виявлення помилок відбувається під час компіляції, а не під час виконання.

Але ж крім цього, існують не лише SQL-дані, тобто інформація, яку ми можемо добути, зовсім не обов'язково буде зберігатися саме у базі даних. Далеко не завжди дані подаються саме у структурованому форматі, тобто у вигляді баз даних, інформація має певне різноманіття форм зберігання. LINQ на відміну SQL придатна не лише для роботи зі структурованими даними з баз даних, але й наприклад зі слабкоструктурованими, що робить цю мову запитів універсальнішою. Наприклад, це може бути інформація з csv-файлів. Тобто інтегрована мова запитів абстргує нас від форми подання інформації. Це важливо коли різні джерела інформації можуть бути зовсім не однієї структури, а зазвичай це так і є, адже в інакшому разі це означало б що ми не можемо сконсолідувати всієї інформації [41].

LINQ містить оператори для агрегації, групування, вибору, поділу, конвертування, сортування, генерації даних тощо. Вони можуть застосовуватися універсально до даних різних форматів і форм подання, тобто є можливість абстрагуватися від цього. Більшість операторів виконуються за відкладеним принципом, тобто при оголошенні в програмі вони ще не виконуються, однак в разі потреби це можна виправити, скориставшися методами конвертування, наприклад звівши результат до списку [42].

Таблиця 2.1 – Оператори LINQ

Тип операторів	**Оператори**
Агрегації	Aggregate, Average, Count, Sum, Max, Min
Вибірки	Select, SelectMany
Групування	GroupBy, GroupJoin, Join
Конвертування	ToList, ToArray, ToDictionary, ToSequence, ToLookup
Сортування	OrderBy, ThenBy (коли сортування більш ніж за 1 параметром), Reverse (зворотнє сортування)
Обмеження	Where
Розділу	Skip (пропустити), SkipWhile, Take, TakeWhile (отримати елементи колекції до виконання певної умови, наприклад досягнення якогось індексу в масиві)
Порівняння	EqualAll
Порівняння множин	Except, Intersect, Union, Concat, Distinct, DistinctBy (різниця за параметром)
Генерації	Empty, Range, Repeat

Насправді LINQ не замінює SQL, більше того його запити автоматично конвертуються в SQL за певними правилами, хоч і не завжди ідеально. Для технології існують провайдери до різних постачальників баз даних, тобто немає різниці це Microsoft SQL Server, MySQL, PostgreSQL, Oracle чи щось інше. Але потрібно ще враховувати, що LINQ має вищий рівень абстракції при запитах гетерогенних даних. Вищий рівень абстаркції це зазвичай добре, адже дозволяє сконцентруватися на більш важливих завданнях. Взагалі LINQ є своєрідним містком між об’єктно-орієнтованими мовами програмування та базами даних, які в свою чергу притримуються декларативного стилю [35].

XML Result

```
<minSalaryEmployees minSalary="36369.00">
  <employee>
    <name>First369 Last369</name>
    <title>Software Engineer</title>
  </employee>
</minSalaryEmployees>
```

LINQ to SQL

```
var minEmpSalary = employee.Min(e => e.eSalary);

var minSalaryEmployees =
  new XElement("minSalaryEmployees",
     new XAttribute("minSalary", minEmpSalary),
     from e in employee
     where e.eSalary == minEmpSalary
     select new XElement("employee",
                  new XElement("name", e.eFirst + " " + e.eLast),
                  new XElement("title", e.eTitle)));
```

Oracle using SQL/XML

```
select xmlelement("minSalaryEmployees",
            xmlattributes(min(e.eSalary) as "minSalary"),
            xmlagg(xmlelement("employee",
                xmlforest(e.eFirst || ' ' || e.eLast as "name",
                          e.eTitle as "title"))))
   from employee e
   where e.eSalary = (select min(eSalary) from employee);
```

SQL Server FOR XML clause

```
select (select min(eSalary) from employee) as "@minSalary",
    (select e.eFirst + ' ' + e.eLast as "name",
            e.eTitle as "title"
     from employee e
     where e.eSalary = (select min(eSalary) from employee)
     for xml path('employee'),  type)
for xml path('minSalaryEmployees');
```

Рисунок 2.4 – Робота з XML даними у LINQ та базах даних Oracle та Microsoft SQL Server

Мова запитів LINQ зокрема застосовується і в інших інструментах Microsoft, наприклад, Entity Framework Core. Хоча дана технологія дозволяє використовувати і чистий SQL-код, однак необхідно враховувати доцільність цього в конкретній ситуації, а також пам’ятати про архітектуру програмного продукту. Враховуючи переваги мови запитів, в контексті архітектури в більшості випадків доцільніше використовувати саме її. Крім того її застосування підходить для здійснення архітектурних припущень, аналізу патернів використання та антипатернів [43].

Крім того при використанні LINQ ризик SQL-ін’єкцій, тобто вставок, значно послаблюється, оскільки не використовується звичайне об’єднання значень рядків, а запити є параметризованими. А для вибірки даних з різних

таблиць чи сутностей непотрібно з’єднувати таблиці, а достатньо викликати відповідну властивість цього об’єкту, що скорочує код. Якщо ж використовується звичайний SQL, то є ризик що досвідчений користувач при користуванні веб-сайтом, замість вказання якогось значення в полі введення задасть SQL-код, який може призвести до несанкціонованих дій, наприклад видалення якихось значень чи таблиць тощо, адже веб-сайти автоматично «довіряють» даним, що вводять користувачі. Хоча SQL може теж бути безпечнішим якщо використовувати його в параметризованій формі, однак це вже вимагає ручної обробки [44].

Звісно у мови запитів є і недоліки. В деяких випадках при розробці дійсно доцільніше використовувати код SQL. До ситуацій, де застосування LINQ може бути неефективним або неможливим, можна віднести:

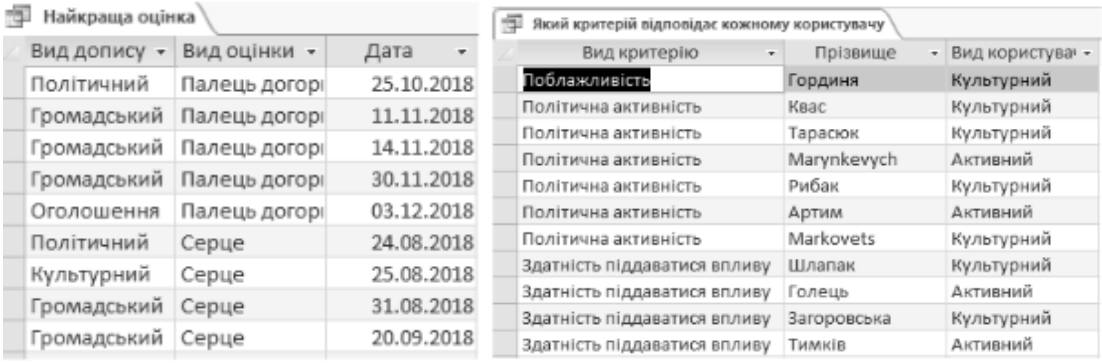
- Використання тригерів — коли потрібна обробка якихось подій та реакції на них.
- Надто складні та специфічні запити, для певних операцій яких немає відповідників в інтегрованій мові запитів.
- Критичність до швидкості виконання.

Однак відсоток випадків, коли дійсно потрібно застосовувати SQL без можливості обмежитися LINQ, є незначним, близько 5%, тобто лише у досить специфічних випадках [45].

2.3 Дослідження покращення швидкодії обробки даних у PLINQ в порівнянні з LINQ для різних розмірів вхідних даних

Швидкодія обробки зазвичай неактуальна при незначних обсягах даних, але її важливість зростає при значному обсягу вибірок, а також і при різкому збільшенні кількості цих вибірок. Прикладом завдання де швидкодія може бути критичною, може стати консолідація інформації з соціальних мереж. На сьогодні їх багато, наприклад, tiktok, youtube, facebook, x тощо. Кожна з них містить мільйони користувачів, відео, коментарів тощо. Звісно кожна соцмережа може передбачати якісь свої особливості отримання вибірок та формат їх

представлення, однак в цілому за допомогою LINQ ми можемо видобувати це з різних форм, зводити все до IQueryable для зручної маніпуляції над консолідованою вибіркою даних. Це можна застосовувати і зараз в умовах війни. Наприклад, співробітники спецслужб під час своєї професійної діяльності колекціонують та обробляють великий набір даних. Однак варто відзначити, що практично 85% цієї інформації отримується з відкритих джерел.



Найкраща оцінка

Вид допису	Вид оцінки	Дата
Політичний	Палець догор	25.10.2018
Громадський	Палець догор	11.11.2018
Громадський	Палець догор	14.11.2018
Громадський	Палець догор	30.11.2018
Оголошення	Палець догор	03.12.2018
Політичний	Серце	24.08.2018
Культурний	Серце	25.08.2018
Громадський	Серце	31.08.2018
Громадський	Серце	20.09.2018

Який критерій відповідає кожному користувачу

Вид критерію	Прізвище	Вид користува
Поблажливість	Гординя	Культурний
Політична активність	Квас	Культурний
Політична активність	Тарасюк	Культурний
Політична активність	Marynkevych	Активний
Політична активність	Рибак	Культурний
Політична активність	Артим	Активний
Політична активність	Markovets	Культурний
Здатність піддаватися впливу	Шлапак	Культурний
Здатність піддаватися впливу	Голець	Активний
Здатність піддаватися впливу	Загоровська	Культурний
Здатність піддаватися впливу	Тимків	Активний

Рисунок 2.5 – Отримані та структуровані дані при аналізі користувачів facebook та їх поведінки

Консолідація даних може мати проблеми при намаганні отримувати дані з різних джерел паралельно. Серед таких проблем можна виділити [19]:

1. Вибір репозиторію звідки братимуться набори даних для виконання завдання.
2. Вибір місця звідки будуть братися набори даних і виконуватиметься завдання консолідації.
3. Вибір шляхів для наборів даних при передачі мережею.

Крім того не можна забувати про синхронізацію у випадку вибору паралельної обробки. Разом з тим при паралельній обробці без використання AsSequential збереження порядку негарантоване.

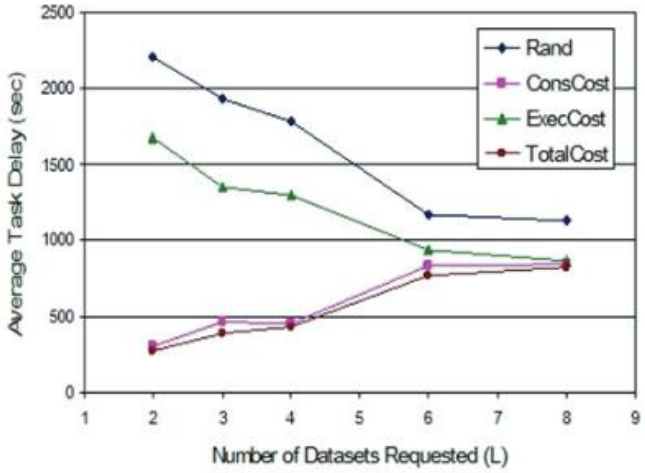


Рисунок 2.6 – Середній час затримки консолідації даних, із середнім обсягом для завдання 600ГБ

Під час консолідації даних з різних ресурсів ми зазвичай отримуємо дані великих обсягів. Щоб продуктивність їх обробки була високою, доцільно використовувати не лише LINQ, але й його паралельну версію – PLINQ. Однак чи в усіх випадках PLINQ буде продуктивніше ніж LINQ? Чи можливі ситуації коли PLINQ навіть програватиме у швидкості? Коротка відповідь – так. Основний механізм роботи паралелізованої технології — це розподіл завдань між ядрами ПК. Але при цьому необхідний додатковий час на цей самий розподіл. Звісно якщо цей додатковий час є зовсім незначним в порівнянні з прискоренням від розподілу, то все добре і паралелізація доцільна. Проблеми виникають коли помітного підвищення продуктивності обробки нема, але час на розподіл між процесами треба і він перевищує значення від потенційного покращення продуктивності — в такому випадку послідовна обробка краща. А факторами, які роблять PLINQ повільнішим на початкових етапах ніж LINQ є:

1. Необхідність розпаралелювання — вхідне завдання потрібно розподілити між процесами системи, і це займає додатковий час. Послідовний LINQ цього не потребує, тому на початкових етапах може бути швидшим.

2. Синхронізація — оскільки кожний потік працює тільки над якоюсь частиною завдання і одержує результат, то доводиться потім результати кожного потоку ще якось синхронізувати між собою для одержання результату загального.

Можуть бути різні причини, коли паралельна обробка LINQ не буде виконуватися швидше ніж послідовна, зокрема розмір вибірки. Розмір вибірки є ключовим параметром і безпосередньо впливає на швидкість обробки. Звісно, чим більший розмір наших даних, тим довше їх потрібно обробляти. Однак саме в таких випадках спостерігається користь від паралельної обробки. Якщо ж даних у нас мало, то PLINQ не здатний забезпечити перевагу у швидкості, а деколи навіть навпаки потребуватиме додаткового часу. Звісно у умовах великих обсягів даних цей додатковий час — некритично. А ось при малих вибірках, які швидко обробляються в будь-якому випадку в силу свого розміру, витрачення часу на розпаралелювання та синхронізацію недоцільне. Тобто якщо ми знаємо, що набір даних передбачається невеликий, використання PLINQ буде лише на шкоду.

Даний ефект наглядно видно на наступному рисунку, адже це надлишковість [46].

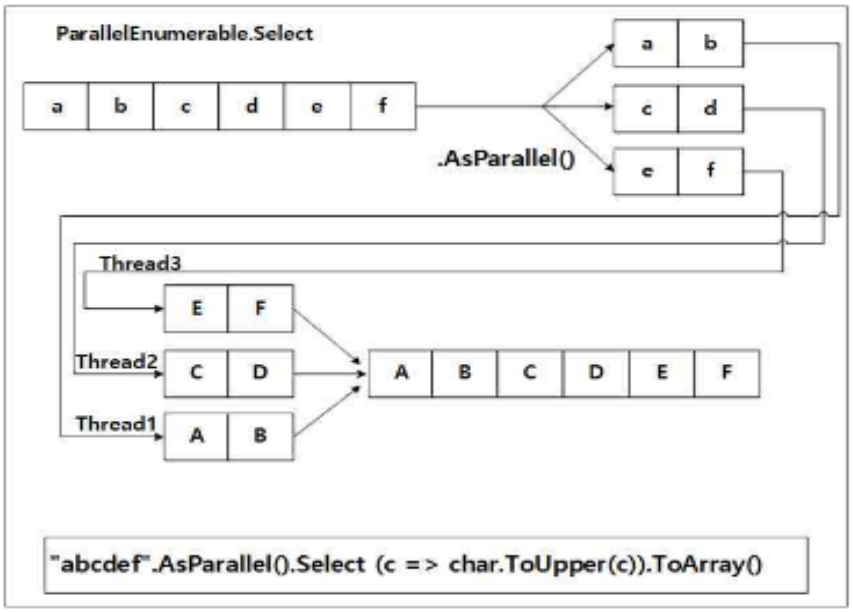
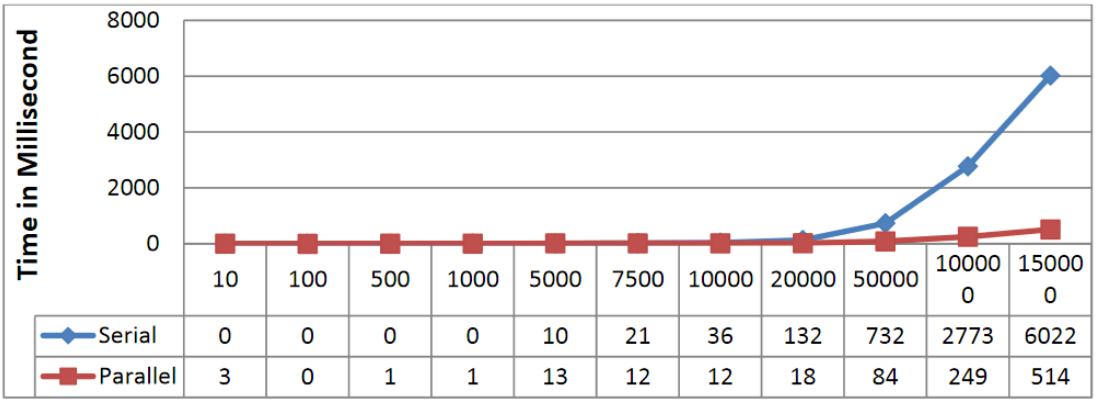


Рисунок 2.7 – Паралелізація виконання

Є й інший фактор, який слід враховувати, хоча він поступово і втрачає актуальність. Якщо код виконуватиметься на одноядерному ПК, то використання PLINQ також не має змісту, ця технологія призначена для багатоядерного апаратного забезпечення, адже робота розподіляється на доступну кількість ядер.

В такому випадку час виконання просто буде орієнтовно такий ж самий як і для LINQ, тобто конструкція AsParallel розподіляє завдання лише якщо можливо, а у випадку використання 1 ядра це неможливо. Переконатися в цьому можна або запустивши запити на 1-ядерному ПК, або ж вказавши ступінь розпаралелювання 1. Якщо ж ядер багато, то чим їх більше, тим більшої швидкості обробки даних можна досягти при використанні паралелізованої версії. За замовчуванням використовуються всі ядра апаратного забезпечення, але можна вказати лише певну кількість, за допомогою функції WithDegreeOfParallelism. Важливо рахувати саме кількість ядер апаратного забезпечення, а не потоків.

Але позитивним фактором від використання PLINQ саме при консолідації є те, як зростає час обробки при збільшенні розміру вибірки. А так як дані отримуються з різних джерел і отже переважно є значними і дуже значними за обсягом, то це питання стає дедалі актуальнішим. Під час використання LINQ час збільшується експоненційноподібно, і вже на вибірках середнього розміру починає перебільшувати відповідний час при використанні PLINQ. Використання ж PLINQ дозволяє часу збільшуватися більш лінійно. Для прикладу можемо дослідити залежність часу знаходження простих чисел (рис. 1) від розміру вибірки, тобто діапазону в якому ми шукаємо ці прості числа [47].



	10	100	500	1000	5000	7500	10000	20000	50000	100000	150000
Serial	0	0	0	0	10	21	36	132	732	2773	6022
Parallel	3	0	1	1	13	12	12	18	84	249	514

Рисунок 2.8 – Порівняння часу паралельного та послідовного обчислення простих чисел у PLINQ та LINQ на різних розмірах вибірок

Для оброблення даних паралельно, потрібно для бажаної колекції викликати метод AsParallel — це вхідна точка для PLINQ, всі інші методи

працюють лише після виклику даного. Загалом PLINQ має декілька додаткових до послідовного LINQ операторів [46].

Таблиця 2.2 – Оператори PLINQ

AsParallel	**AsSequential**	**AsOrdered**	**ForAll**	**WithDegreeOfParallelism**
Вхідна точка для переходу від LINQ до PLINQ. Паралелізація відбувається лише якщо можливо	Виконання залишку запиту послідовно, збереження порядку	Збереження порядку до кінця запиту або до оператора OrderBy	Паралельна обробка кожного елементу без збереження порядку	Ступінь розпаралелювання, на який поділяти вхідне завдання

Окремим питанням продуктивності є стратегія виконання запитів. В LINQ для більшості операторів за замовчуванням використовується ліниве виконання (lazy loading). Сутністю такого підходу є те, що запит не виконується зразу після його оголошення в програмі, а лише зберігається. Для виконання його треба явно викликати або конвертувати наприклад отриманий результат в список за допомогою методу ToList. Ліниве виконання зазвичай використовується коли обсяг даних може перевершувати розмір оперативної пам’яті. Для завдань консолідації інформації це зазвичай є актуальним, тому що інформації може бути багато, більше того багато також і джерел цієї інформації. Використовуючи ж ліниве виконання, особливо при значних розмірах колекцій, ми можемо отримувати в пам’ять не автоматично всю колекцію при звертанні до неї, а лише її необхідну частину. Це особливо актуально, коли обсягів оперативної пам’яті недостатньо або ж коли колекції занадто великі, або ж тим паче у випадку виконання обох умов одночасно. В інакшому ж випадку створюється зайве непотрібне навантаження як на розмір використовуваної пам’яті апаратного забезпечення, так і на швидкість виконання запитів. Хоча часто використання

додаткової пам’яті якраз таки призводить до покращення продуктивності, і навпаки, однак це не той випадок [48].

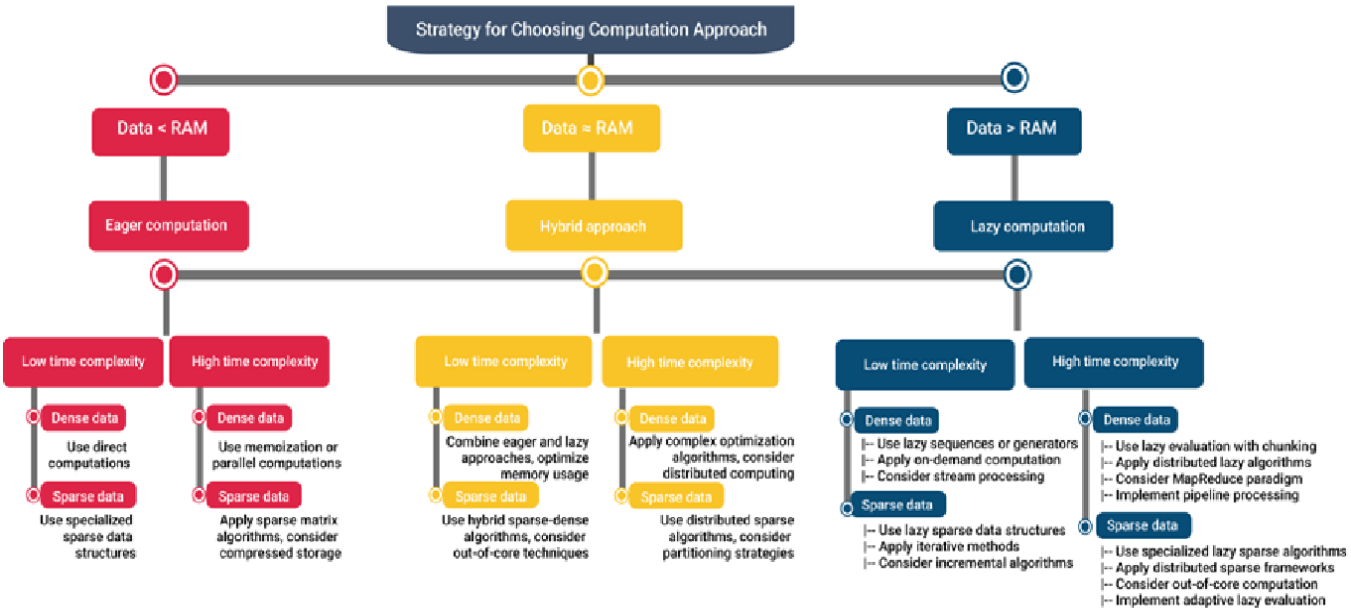


Рисунок 2.9 – Стратегія обчислень залежно від розміру колекцій та доступного обсягу пам’яті

В LINQ ліниве виконання фактично є відкладеним виконанням (deffered execution). Його підтримують більшість операторів, тобто навіть коли оголошений вираз, він не буде автоматично викликатися. Такі методи викликатимуться лише при ітераціях колекціями (foreach). Однак є винятки і певний список операторів, які забезпечують негайне виконання. Це методи конвертації, агрегатні операції, пошук елементу (Any, FirstOrDefault, ElementAt) тощо. Тобто до методів з відкладеним виконанням не відносяться лише ті, які безпосередньо передбачають негайне отримання якогось значення [49].

2.3.1 Характеристики значних обсягів даних при консолідації

Оскільки консолідовані дані характеризуються зазвичай значними обсягами інформації, то можна виділити характеристики таких даних [50]:

1. Обсяг (volume) — сумарний кортеж типів даних з всієї сукупності різнотипових джерел інформації, перелік яких продовжує розширюватися.

2. Швидкість (velocity) — експоненційний зріст швидкості накопичення даних, що спричиняє динамічні зміни в колекціях даних.

3. Різноманітність (variety) — широкий перелік форм подачі інформації різних типів, що може включати текст, числові ряди, аудіо та відеофайли, геопросторові дані, подані у структурованих та неструктурованих форматах.

4. Валідність (validity) — актуальність даних в контексті використовуваного методу аналітичного опрацювання чи процесу.

5. Значення (value) — виявлення прихованих значень і знань у швидкоплинних різнотипових наборах даних консолідованої інформації.

6. Точність (veracity) — якісний показник колекцій даних, показує придатність колекції даних для процесів аналітичного опрацювання.

7. Видимість (visibility)/візуалізація (visualization) — можливість ефективного подання важливої інформації навіть попри великі обсяги, та здійснення ефективної візуальної аналітики.

8. Віртуальність (virtuality) — стосується процесів управління даними, які є віртуальними.

9. Мінливість (variability)/волатильність (volatility) — зміни швидкості отримання якісних та контекстних характеристик даних.

10. Валентність (valence) — можливість подання зв’язків значних за обсягом даних у вигляді графів для генерації потреби використання складних алгоритмів аналітичного опрацювання даних.

Окремі параметри, наведені у списку, можуть мати інші англійські відповідники, однак застосовуються саме значення, які починаються на v. Це називається принципом «10v», який початково починався з концепції «3v» (volume, variety, velocity), однак характеризував значні обсяги даних не повною мірою. PLINQ фактично враховує ці принципи, і тому підходить для роботи з інформацією подібного класу, адже забезпечує добру швидкість, різноманіття форм подачі інформації, обсяги тощо.

2.4 Висновок до другого розділу

В другому розділі кваліфікаційної роботи досліджено вибір технології LINQ для застосування в завданнях консолідації даних. Її вибір обгрунтований створенням абстракції незалежно від типу джерела надходження інформації і здатності здійснювати запити до них. Розглянуто також можливості застосування і швидкодію у випадках значних за обсягом вибірок даних або/і їх кількості, а ще для випадків розподіленого виконання. Швидкодія паралельного виконання перед усім залежить від обчислювальних потужностей, однак на малих наборах даних може навіть поступатися швидкості послідовного виконання через необхідність розпаралелювання та синхронізації процесів. Однак вона сильно перевершує швидкодію послідовного виконання у випадку збільшення кількості джерел консолідованої інформації або ж їх розміру.

3 ОБЧИСЛЮВАЛЬНИЙ ЕКСПЕРИМЕНТ

3.1 Застосування паралельної обробки до даних і визначення швидкості

Для прикладу обчислимо числа які є та не є простими на практиці, тобто фактично знайдемо дільники кожного числа (якщо це буде тільки 1 та саме число — то значення буде простим). Діапазон для обчислень будемо брати однаковим — від 1 до 27900. Це доволі середній діапазон, обчислення на якому вже не будуть займати лічені мілісекунди щоб можна було результат вважати похибкою, але й ще не будуть займати надто багато часу щоб слід було чекати результату півгодини. Звісно чим більшим ставатиме обсяг даних, тим більше переваги буде від поділу завдання між ядрами, а чим меншими — тим незначнішим це буде.

Для початку результат обчислюватимемо без паралелізації за допомогою LINQ. Для фіксації часу можна використати Stopwatch. Однак необхідно пам’ятати що більшість LINQ-операторів не призводять до миттєвого виконання запиту адже працюють за схемою відкладеного виконання. Для того щоб був виконаний запит (а отже щоб рахувати саме час його виконання, а не лише час створення самого виразу який фактично не відрізнятиметься для LINQ та PLINQ) може знадобитися його явний виклик, наприклад це можна зробити, конвертувавши результат в список за допомогою функції toList.

Для діапазону чисел до 27900 результат непаралельного виконання LINQ склав орієнтовно 12 секунд часу. Це прийнятний час, щоб все не було в межах похибки, але й не надто тривалий щоб довго чекати. Тим паче, що як розглядалося в попередньому розділі, час зростає експонеційно, тобто підвищуючи розмір вибірки навіть незначно будемо отримувати значне збільшення часу обчислень.

Рисунок 3.1 – Пошук дільників чисел в межах до 27900 та визначення часу виконання операції за допомогою звичайного LINQ

Для паралелізації виконання необхідно вказати AsParallel при виборі набору даних, з яким працюватимемо — це вже фактично буде PLINQ. Розпаралелення буде відбуватися відповідно до наявної кількості ядер на апаратному забезпеченні, на якому виконуються запити, тобто якщо це 4 ядра, то це розпаралелення на 4. Результат як і очікувалося, покращився приблизно в 4 рази (трохи менше, що логічно, враховуючи що на розпаралелювання і синхронізація потоків теж потрібен час, хай і невеликий), адже на даному ПК розпаралелювання відбувається за замовчуванням на 4. Звісно апаратне забезпечення яке має більше ядер, дозволяє організувати розпаралелювання на більшу кількість процесів. Однак якщо даних недостатньо багато, то ефект від цього може бути незначним (або навіть відсутній), а ось зайвий час на розпаралелювання, а потім ще й на синхронізацію цих потоків між собою - все одно потрібний.

Рисунок 3.2 – Паралелізоване виконання PLINQ та визначення часу

Однак можна вказувати явно на скільки процесів слід здійснювати розпаралелювання за допомогою інструкції WithDegreeOfParallelism. Звісно це впливатиме на час виконання. Якщо вказати 1 ядро, то фактично результат співпадатиме з LINQ, тобто 12-13секунд для обраного розміру вибірки. Повної точності співпадіння часу може і не бути, так як навіть при різних запусках з однаковими параметрами він може відрізнятися. Але запуск інструкції AsParallel все одно може займати певну кількість часу.

Рисунок 3.3 – Виконання PLINQ на 1-ядерному апаратному забезпеченні

При збільшенні кількості використовуваних ядер для завдання результат буде покращуватися приблизно в 2 рази. Для повного розуміння картини варто поцікавитися кількістю ядер/потоків на апаратному забезпеченні, на якому планується виконання запитів.

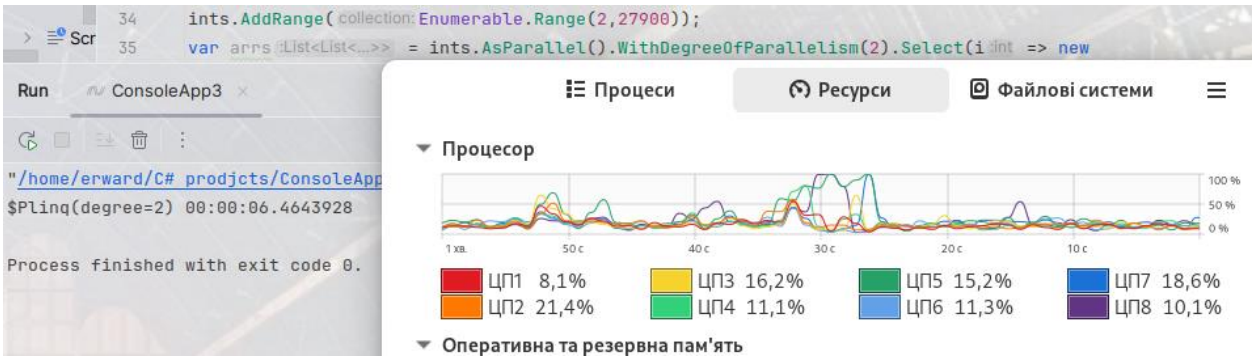


Рисунок 3.4 – Часткове розпаралелювання

При вказанні максимальної кількості ядер результат вже співпадатиме з виконанням без вказання їх кількості і буде мінімальним — близько 3 секунд для даного розміру вибірки.

Рисунок 3.5 – Повне розпаралелювання

Слід розрізняти кількість доступних ядер та потоків. Хоча й ПК, на якому здійснюється тестування, містить 8 ядер, згідно диспетчеру, однак фактично це 8 потоків, а ядер 4. В такому випадку розпаралелювання на 8 і далі процесів замість 4 вже не дає ніякого приросту продуктивності, результат все одно становить близько 3 секунд. Результат буде згідно кількості ядер, а не потоків.

Рисунок 3.6 – Різниця при розпаралелюванні при неоднаковій кількість ядер та потоків

Однак PLINQ не зберігає порядку, тому якщо це необхідно, варто застосовувати AsSequential. Однак він забирає різницю в швидкодії.

3.2 Обробка медичних даних

Сферою, де може знадобитися консолідація даних, є медична сфера. В даній сфері зібрані великі за обсягом набори та колекції медичних даних можуть застосовуватися для досить різних задач, як-от діагностика, пошук лікування та лікарів, підтримка процесів, пов’язаних з адміністративно-територіальними центрами надання медичних послуг тощо. Залежно від задачі, в якій використовується медична інформація, вона може передбачати різну форму подання, наприклад текстову або ж у формі біомедичних зображень. Тому в цій сфері так само важлива класифікація та визначення структурованості даних. Колекції даних зазвичай надаються від постачальників медичних послуг, а для їх розміру, як і для аналогічних колекцій в інших сферах, характерне експоненційне зростання [51]. Міністерство охорони здоров’я є головним постачальником медичних послуг та надає певні набори даних (значних за обсягом) у вигляді CSV файлів. Звісно не вся інформація присутня у відкритому доступі. В цілому це актуальні дані, які регулярно оновлюються, однак вони не завжди є повними, навіть незважаючи на їх розмір. Для прикладу набори містять велику кількість лікарів, однак там немає багатьох лікарських спеціальностей, що робить список потенційно неповним. До прикладу там міститься інформація про 27030 лікарів та декілька тисяч відділень. Тут слід врахувати деяке дублювання даних при обробці, наприклад, якщо лікар працює терапевтом та сімейним лікарем — інформація в таблицю заноситься двічі (не обов’язково це сусідні записи). У LINQ слід використати оператор Distinct або навіть краще DistinctBy для вказання критерію неоднаковості. Проблемами даної консолідації можуть бути як нестача гетерогеності, формалізації, територіальна дисперсія знань, так і різна форма збереження знань (особливо рецептів, які є неструктурованими даними) [52].

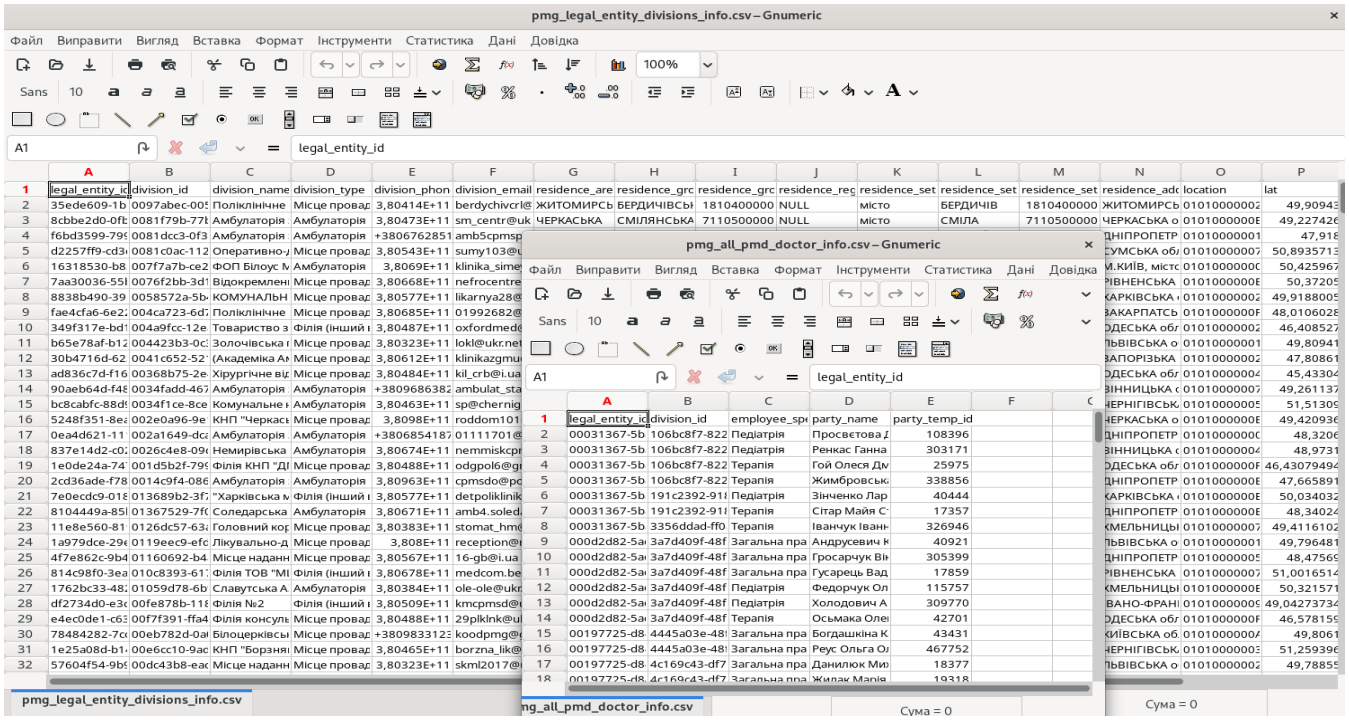


Рисунок 3.7 – Відкриті медичні дані МОЗ

Однак в таблиці вказані лише лікарі трьох лікарських спеціальностей — загальної практики сімейної медицини, терапії та педіатрії. Потенційно ми можемо отримати більше даних, тобто також про лікарів інших спеціальностей, якщо скористаємося ресурсом medics.ua. Він надає дані в іншому форматі, а саме у вигляді API, до того ж тут значно більше лікарських спеціальностей. Для LINQ це не проблема, він може поєднувати отримання даних і з CSV, і з API. CSV файли фактично є слабкоструктурованими джерелами інформації, оскільки певна структура даних є, однак жорстких вимог нема, стовпці розділені за допомогою роздільників, наприклад ком. API ж дані — структуровані і зберігаються в базах даних, можуть містити вкладені об’єкти.

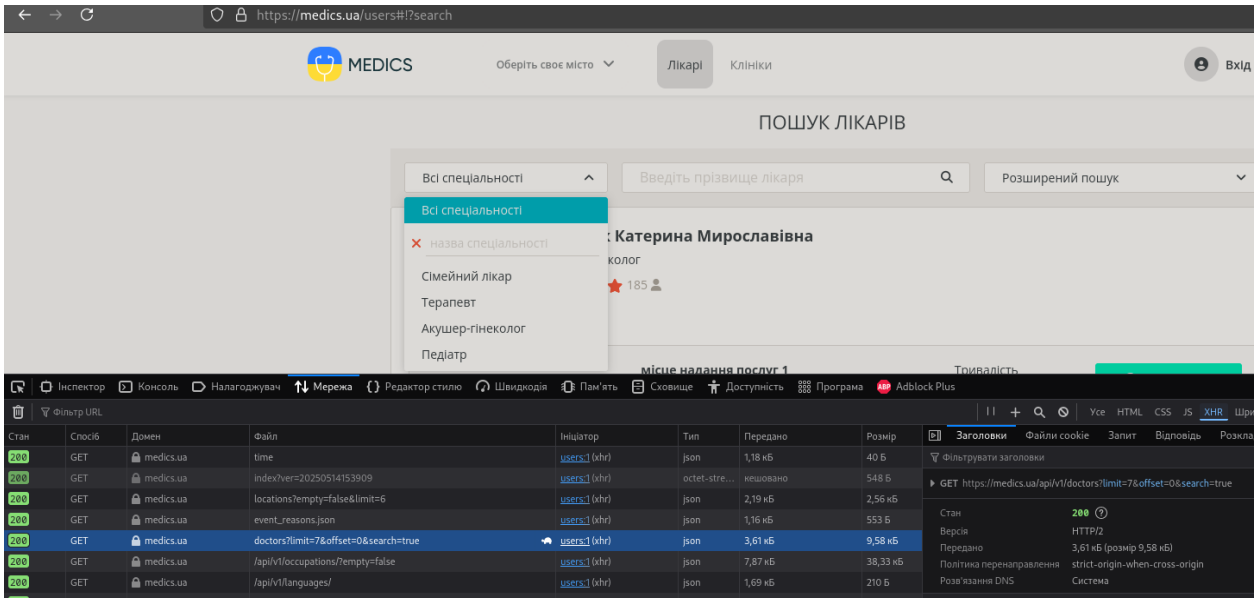


Рисунок 3.8 – Платформа medics

Тобто фактично при консолідації ми отримуємо ситуацію, коли у нас є 2 набори даних і вони частково перетинаються, а частково ні. Іншою проблемою постає неоднакове подання даних в наборах. До прикладу CSV-файли не містять інформації про рейтинг лікаря і кількість відгуків, а також URL-адресу фотографії, тобто виникає завдання стандартизації. Фактично потрібне об’єднання множин (Union), а також комбінування інформації про працівників, знайдених в обох множинах. Якщо ж якісь записи зустрічаються у множині з меншим переліком атрибутів, а в множині з більшим переліком атрибутів — ні, тоді при об’єднанні відповідні поля цих записів будуть містити невизначеності, тобто значення null. В даному випадку це стосується рейтингів і фотографій лікарів, які буле знайдені в файлах, однак інформації про яких нема на medics.

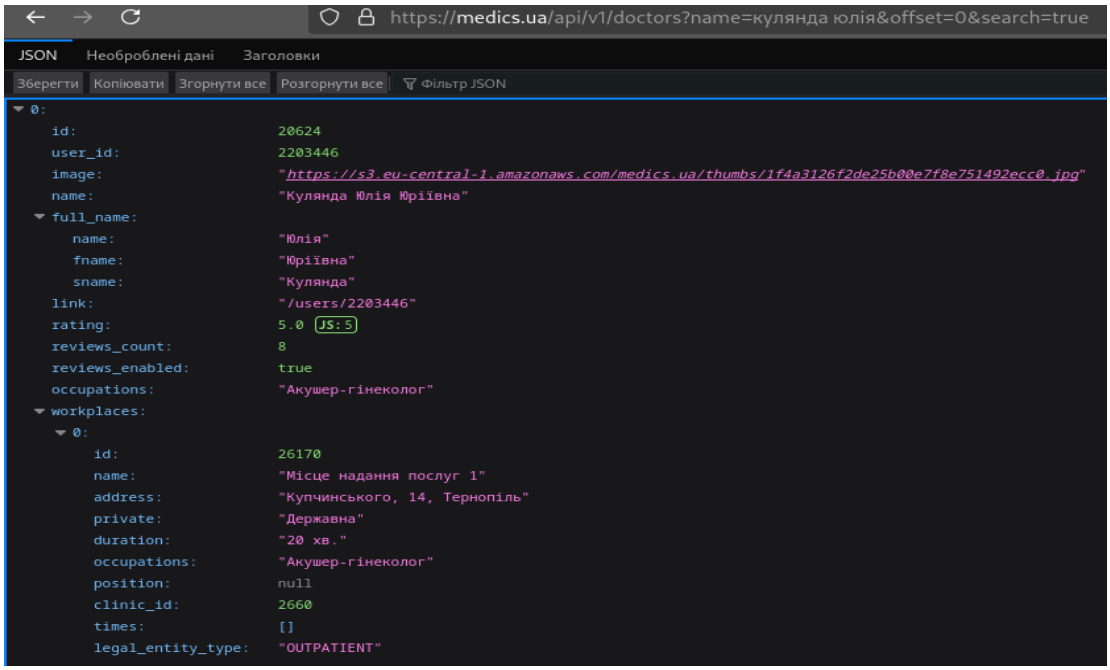


Рисунок 3.9 – Дані лікаря у API Medics

Для збереження інформації повинні бути класи (моделі) сутностей з якими ми працюємо, тобто лікарі, відділення і т.д — вони мають містити ті атрибути, які надаються джерелами даних, тобто ім’я, лікарська спеціальність, рейтинг, фото, місце роботи. Для добування слабкоструктурованих даних необхідна бібліотека LINQToCSV.

Лістинг 3.1 – Отримання даних з CSV-файлу

```
using LINQtoCSV;
CsvContext cc=new CsvContext();
IEnumerable<Doctor>
doctors=cc.Read<Doctor>("pmg_all_pmd_doctor_info.csv");
IEnumerable<Division>
divisions=cc.Read<Division>("pmg_legal_entity_divisions_info.csv")
```

Для зчитування ж інформації з API потрібно скористатися запитом, де вставити URI адресу ресурсу. Для формування адреси слід вказувати ім’я лікаря в якості параметру, однак можливі й інші способи формування – за id територіальної одиниці чи медзакладу де знаходиться лікар або ж за лікарською спеціальністю. API medics в такому випадку видає результат лише невеликими порціями, наприклад по декілька лікарів. Для переходу ж далі слід задавати offset. При використанні першого способу, тобто пошуку за повним ім’ям, можна

отримати додаткові інформаційні атрибути до тих, що вказана в наборах даних міністерства охорони, однак лише при умові що цей лікар буде знайдений в API. Хоча список лікарів за спеціальностями на medics ширший, однак за спеціальностями з наборів даних деякі лікарі на medics відсутні.

Лістинг 3.2 – Отримання даних з API

```
static async Task<JsonElement.ArrayEnumerator> DataFromApi()
{
    HttpClient client = new HttpClient();
    HttpRequestMessage request = new HttpRequestMessage
    {
        Method = HttpMethod.Get,
        RequestUri                          =                          new
Uri($"https://medics.ua/api/v1/doctors?name={name}&offset=0&search
=true")
    };
   HttpResponseMessage response = await client.SendAsync(request);
    string apiData = await response.Content.ReadAsStringAsync();
    JsonDocument doc = JsonDocument.Parse(apiData);
    JsonElement root = doc.RootElement;
    JsonElement.ArrayEnumerator jsonArray = root.EnumerateArray();
    return jsonArray;
}
```

Звісно наявність додаткових атрибутів в одному із наборів не означає що їх значення будуть вказані для всіх записів. Так є лікарі з відсутнім фото та/або рейтингом. В такому випадку записи між наборами не розрізняються. При консолідації інформації потрібне не лише звичайне об’єднання записів, але й уникнення дублювань. Так якщо інформація про якогось лікаря міститься в двох наборах даних, то не потрібно двічі дублювати однакову інформацію, оскільки вона є надлишковою, а також буде створювати зайве навантаження на займану пам’ять, швидкість роботи запитів та зручність відображення. Для уникнення дублювань LINQ передбачає Distinct та DistinctBy. Враховуючи кількість записів в таблицях та на веб-сайті є потреба використовувати PLINQ. При цьому послідовність запису, обробки великої ролі в даному випадку не грає, в інакшому ж випадку є інструкція AsSequential. Для роботи ж з невідомими значеннями використовуються nullable-типи та методи, що містять суфікс OrDefault. Прикладом таких даних можуть бути лікарі, що знайдені лише в наборах даних,

і не знайдені на medics – тоді в них відсутнє фото, рейтинг. Якщо ж можлива відсутність лікарів за певною спеціальністю, наприклад в контексті якогось міста – дані будуть пустими і в такому випадку завжди слід перестрахуватися. В контексті питань забезпечення якості даних важлива також достовірність. Якщо між джерелами даних є якісь протиріччя, довіряти слід тому джерелу, яке вважається більш достовірним. Набори даних CSV можна вважати більш достовірною і оновлюваною інформацією в разі розбіжностей.

3.3 Висновок до третього розділу

В третьому розділі кваліфікаційної роботи описано завдання консолідації з 2 різних гетерогених джерел різної структурованості з великим обсягом записів з допомогою PLINQ для медичної сфери. Досліджено методи які б забезпечували коректний процес консолідування. Подано опис обчислювального експерименту порівняння швидкостей послідовної та паралельної обробки. Розглянуто приклад взаємодії з неповними, невідомими та недостовірними даними через консолідацію інформації з джерел, де частина даних у відношенні до інших джерел неповна.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Використання пристроїв з flicker-free моніторами без мерехтіння для роботи зі значними обсягами даних

Тема кваліфікаційної роботи освітнього рівня «Магістр» присвячена дослідженню консолідації і роботи з даними у LINQ/PLINQ. Тому доцільно розглянути питання використання моніторів з низькою шкідливістю при обробці таких даних, а також правильне їх застосування. Це безпосередньо впливає як на зір, так і на втому людини, як очей так і загальну. Людина не може здійснювати ефективний аналіз даних в умовах значних обсягів цих даних, коли з’являється дискомфорт в очах і неможливо нормально дивитися або ж коли є втомлюваність. Тим паче що напруга очей залежить і від виду діяльності за екраном.

Звісно при роботі з технікою мають бути регулярні перерви, корисно також переводити погляд у вікно, тим паче при роботі з даними виникають затримки при отриманні в зв’язку з великими обсягами цих даних, чим можна скористатися для розвантаження зорового нерву. Але в будь-якому випадку важливо також забезпечити правильний рівень освітлення. В першу чергу необхідно уникати різкого контрасту між освітленістю пристрою відображення та навколишнім середовищем, зокрема не варто дивитися на яскравий екран у темному приміщені. Для молодих людей, а також людей, в яких нема проблем з зором, ця необхідність може здатися неочевидною (але це не підстава ігнорувати дані норми), однак в іншому випадку можна буквально відчути тиск на очі при великій різниці контрастів між зовнішнім і внутрішнім середовищем. Нормою освітленості кімнати є 300-500 люкс [53]. При потребі забезпечення правильного відношення між яскравістю монітору та навколишнього приміщення, допускається використовувати штучне освітлення (переважно люмінісцентними лампами), однак згідно державного стандарту, відсутність можливості природного освітлення (відсутність вікон) не допускається. Разом з тим орієнтація вікон має бути на північ чи північний схід для уникнення надмірного

засліплення, в протилежному ж випадку на вікнах повинні бути жалюзі, ролети або штори з можливістю регулювання.

Рівень максимальної яскравості обмежується параметрами матриці, яка видає зображення. Яскравим вважається зображення при величині яскравості дисплею >200Кд*м^2[54]. На ноутбуках ця величина зазвичай нижче ніж в телефонах, тому працювати в умовах яскравого сонця може бути складніше, однак у випадку приміщення рекомендовано застосовувати ролети чи інші засоби світлозахисту. До того ж необхідно враховувати і покриття екранів. В телефонів воно зазвичай глянцеве, що ускладнює перегляд на сонці. Розпізнати його можна за характерною дзеркальністю коли дивитися на нього. В ноутбуків ж попри зазвичай меншу максимальну яскравість, екрани зазвичай матові або хоча б містять антивіддзеркалюючий ефект, що покращує ситуацію і допомагає уникнути блискавості [54]. Візуально пристрій відображення такого пристрою не працює як дзеркало і це ключова відмінність. Це важливо, адже намагання розгледіти щось в затемненому зображенні викликає зайву перенапругу зорових нервів, а в окремих випадках і головну біль, що не сприяє продуктивній роботі.

Однак не менш важливим також є і встановлення нижнього порогу яскравості, але не завжди це відбувається безпечно, з питанням його встановлення тісно пов’язане питання ШІМ, яке не слід плутати з частотою оновлення екрану. В ідеалі для цього на дисплей слід подавати знижену напругу в момент коли його яскравість не максимальна, що власне і буде регулювати її рівень. Чим нижча напруга, тим нижчий рівень. Але часто це робиться іншим способом — застосуванням PWM (широтно-імпульсної модуляції). Це шкідливий для людини спосіб встановлення яскравості, і на відмінну від зниження напруги для подачі меншої енергії, тут знижений рівень яскравості досягається за рахунок подачі максимальної напруги для будь-якого рівня яскравості, але не на постійній основі. Тобто дисплей отримує максимальне живлення, але щоб візуально досягти цієї нижчої яскравості, він блимає, чергуються моменти максимальної яскравості екрану з моментами його виключення, що для людини виглядає просто як нижча яскравість. Рівень її зниження залежить від того на скільки часу виключається дисплей. Звісно

частота цього блимання візуально непомітна для ока людини, однак це зовсім не означає що впливу на зоровий нерв, а також на загальну втомлюваність немає. Особливо важко буде використовувати пристрої з ШІМ людям, які вже відчувають дискомфорт, біль або втому при роботі за комп'ютером. Але звісно в зоні ризику всі люди, навіть з відмінним зором, тому спосіб зниження яскравості не можна ігнорувати.

Хоча частота модуляції завжди достатньо висока щоб людина не бачила її візуально, але в кожному пристрої, що використовує ШІМ вона відрізняється. Найнебезпечнішим випадком є коли ця частота мала — в такому випадку шкоди від неї найбільше. При збільшенні частоти рівень небезпеки дещо падає. Також часто існує практика, коли дисплей використовує ШІМ не на всьому діапазоні яскравостей, а тільки на якомусь (наприклад, коли яскравість встановлена на рівні менше 60%). Однак в будь-якому випадку необхідно прагнути повної відсутності модуляції. Але це питання не пов'язане з частотою оновлення екрану (тобто плавністю картинки), яка зазвичай становить 60Гц або все частіше 90, 120 та 144, а в деяких випадках і до 240Гц, а стосується саме мерехтіння для регуляції яскравості зображення [55]. ДСТУ встановлює межі яскравості моніторів та приміщень, в яких вони встановлені, однак не згадує за спосіб регуляції, тому питання ШІМ в контексті регуляції яскравості пристроїв відображення ніяк не врегульоване ДСТУ. Саме тому так легко знайти як ШІМ-пристрої, так і flicker-free, при чому часто навіть явно не вказується до якої категорії відноситься обладнання. Однак частота може бути від 100Гц до декількох кГц, і низька частота зазвичай використовується в найдешевших варіантах.

Зазвичай виробники будь-яких пристроїв (особливо якщо він містить не лише функцію відображення, як комп'ютерний монітор, але й якісь обчислювальні функції, як-от ноутбук чи телефон) прямо не вказують чи використовує їх пристрій модуляцію, однак цю інформацію можна знайти на спеціальних ресурсах (наприклад, notebookcheck. Цей ресурс містить багато інформації про апаратні засоби, в тому числі їх дисплеї. Інформація є і стосовно ноутбуків, і стосовно телефонів. Для цього потрібно знати модель використовуваного пристрою). Також у деяких типах дисплеїв завжди за

замовчуванням використовується ШІМ, наприклад у екранах amoled — зазвичай такі екрани використовуються саме в смартфонах, але можна їх зустріти також і на комп'ютерних моніторах та ноутбуках. Це пов'язано з технологією виготовлення самих amoled дисплеїв. Це звичайно дає і свої переваги, як-от можливість включення чи виключення лише окремих пікселів дисплею, що добре позначається як на енергоспоживанні, так і на якості відображення чорного кольору (часто з цим проблеми, адже чорний нерідко виглядає як підсвічений темносиній). Однак з точки зору впливу на зір та втомлюваність це недолік. В інших ж типах екранів (наприклад, IPS) модуляція може бути як присутня, так і відсутня, потрібно шукати або за пристроєм де це встановлено, або ж знати назву конкретного дисплейного модулю, який застосовується тут, і шукати вже за ним. Можна також перевірити за допомогою «тесту олівця» - помахати олівцем або увімкнути вентилятор перед дисплеєм — якщо зображення буде плавним, то модуляція відсутня.

Дисплейні модулі, які не використовують технологію ШІМ, називаються flicker-free, тобто вільні від мерехтіння. Саме пристрої з такими дисплеями мають бути в пріорітеті як загалом, так і тим паче для обробки даних з допомогою LINQ/PLINQ [56].

4.2 Вплив мікроклімату на продуктивність праці при аналізі даних

На продуктивність аналізу даних і взагалі будь-якої мозкової діяльності або діяльності, яка потребує підвищеної уваги, впливають різні внутрішні і зовнішні фактори. Наприклад, людина не може ефективно мислити і щось вирішувати коли у неї погане самопочуття. Самопочуття в свою чергу значно залежить в тому числі і від мікроклімату приміщення, в якому людина здійснює свою діяльність. Це пояснюється тим, що від зміни мікроклімату можуть коливатися показники рівня кисню та глюкози в крові, частота серцебиття, тиск, температури тіла. Важливим фактором є вентиляція, адже необхідне насичення киснем мозку для продуктивної роботи. Звісно, люди стійкі до цього, можливо не відчують сильної різниці (але якщо це буде на регулярній основі, то ситуація

буде поступово змінюватися), але деякі люди мають підвищену чутливість до нестачі кисню. Тому треба мати якісну вентиляцію, здійснювати провітрювання приміщення (в разі якщо необхідне швидке, то слід здійснювати перехресне, тобто вікна з протилежних сторін). Іншим фактором є вплив зовнішньої температури. По-перше температура, а також її різкі зміни може спричиняти розширення та звуження судин, зокрема голови, що може викликати дискомфорт і навіть біль. В умовах навіть не дуже підвищеного болю такого аналітичного центру як голова, продуктивність і деколи навіть працездатність втрачається. Тому приміщення повинні бути обладнані сонцезахисними системами (ролетами) та за можливістю, кондиціонерами та спеціальними склопакетами. Це актуально як для надзвичайно низьких температур, так і високих. Однак основна небезпека при впливі саме підвищених температур є виділення тепла та дегідратація. Крім впливу на продуктивність це може спричиняти навіть ризик порушення кровообігу та зміна тиску. Ризик смертності через хвороби кровообігу при роботі з порушенням теплового порушення та при роботі без нього складає різницю у 1.8-3.3 рази. Звісно, обробка даних — це не фізичні навантаження, тому вимоги щодо максимальних температур тут дещо нижчі ніж при фізичних навантаженнях. Однак навіть тут є свої нормативи. Згідно державних стандартнів, при роботі за персональними комп’ютерами повинна дотримуватися температура 19.5+-0.5 С, а відносна вологість на рівні 60+-5% [53]. В кожної людини сприйняття температури різне, однак золоте правило — це дотримуватися такої температури, щоб людина про неї не замислювалася. В такому випадку означатиме, що людині мікроклімат комфортний і вона готова зосередитись на виконанні своїх безпосередніх завдань. Відносна вологість в цьому випадку може впливати на відчуття температури, наприклад, висока вологість робить сприйняття високих та низьких температур важчими. Низька ж вологість може викликати пересихання слизових оболонок, очей. Крім того, швидкість руху повітря має бути до 0.1м/с, що дозволяє уникнути проблем з простудними захворюваннями та очима [53]. Однак норми мікроклімату дещо відрізняються відрізняються для холодної та теплої пори року, а також від інтенсивності робіт [57].

4.3 Планування заходів цивільного захисту на об’єкті у випадку надзвичайних ситуацій

Тема кваліфікаційної роботи освітнього рівня «Магістр» присвячена дослідженню консолідації і роботи з даними у LINQ/PLINQ. Тому доцільно розглянути питання планування заходів цивільного захисту на об’єкті у випадку надзвичайних ситуацій. Особливо це актуально в умовах воєнного стану та підвищених ризиків. Однак навіть за відсутності бойових дій і його ризику це актуально завжди, адже неможна уникнути природних катастроф, а деколи навіть і негативного впливу людського фактору. Тому планування заходів цивільного захисту має бути спрямоване на ліквідацію та пом’якшення наслідків, а в ідеалі і їх уникнення. Найважливішим при цьому залишається життя і здоров’я людей. Але звісно що по можливості проводяться і заходи для зменшення економічної шкоди.

Задача консолідації даних відбувається за ПК в приміщенні. Будь-яке приміщення є закритим об’єктом, відповідно існує ризик надзвичайної ситуації. Звісно надзвичайна ситуація може виникнути буде-де, зокрема і на відкритому просторі. Однак на об’єкті відмінність полягає в тому що є закритий простір і не завжди є можливість швидко покинути територію, де сталася надзвичайна ситуація. Цьому можуть сприяти різні фактори такі як: обмежена пропускна здатність приміщення, перекриті певні ділянки об’єктів в межах контролю доступу (тобто доступні не для всіх)

До надзвичайних ситуацій можуть відноситися різні небезпечні події та катастрофи, спричинені внаслідок природнього фактору або ж людської недбалості, від яких не застрахований ніхто. Прикладами таких подій можуть бути пожежі, повені, землетруси, вибухи.

Слід також пам’ятати що робота за комп’ютером та консолідація інформації часто може проводитися у висотних будівлях. В такому випадку існує підвищена небезпека в разі пожежі. Вибратися з верхніх поверхів може бути проблематично. До того ж існує максимальна висота до якої може дотягнутися

пожежна машина. Але в таких випадках обов’язкове встановлення протипожежних сигналізацій та інших засобів захисту. До того ж в умовах України навіть при безпечній висоті будівлі пожежна машина незавжди може під’їхати через припарковані автівки.

З іншого боку підвищена висотність будівлі може бути корисною в разі повені. В ситуаціях коли рівень води значний і затопив вже все навколо, верхні поверхи можуть допомогти сховатися від надмірної води і перебувати там тривалий час до стабілізації ситуації. Однак в такому разі слід запастися запасами продуктів та води, щоб була змога безпечно перечекати ситуацію. І звісно має бути справною система вентиляції, адже довга нестача кисню може призводити до задишки, недостатнього кровопостачання мозку (що в свою чергу не дозволяє працювати та здійснювати аналіз даних, особливо продуктивно) та тканин тощо.

Ще дещо іншою є ситуація в разі потенційних вибухів та землетрусів. В таких випадках потрібно терміново покинути приміщення, при чому просто вийти на вулицю до приміщення недостатньо. Адже якщо об’єкт зазнає землетрусу чи вибуху то він може бути зруйнований і відповідно почати падати на прилеглу територію. Якщо там будуть люди, то вони в зоні підвищеного ризику. При чому чим висотніша будівля, тим подалі від неї доведеться триматися, адже вона в такому разі падатиме на більшу відстань.

До планування заходів захисту на об’єкті в разі надзвичайних ситуацій відноситься зокрема встановлення систем оповіщення населення. Зокрема в містах країни встановлені гучні сирени, які оповіщають населення в разі тривог, ракетних небезпек та дронових атак. Це є безпосереднім завданням міських влад на місцях, адже вони відповідальні за життя та здоров’я місцевого населення. Звичайно, якщо хтось з громадян ігнорує оповіщення, то це вже особиста відповідальність. Однак система оповіщення має бути встановлена та постійно підтримуватися в справному стані [58].

Іншим заходом захисту в разі потенційних надзвичайних ситуацій є тренування населення, і зокрема працівників підприємств, а особливо об’єктів з підвищеною небезпекою. Це можуть бути наприклад атомні станції, хімічні

підприємства тощо. Але в контексті інформаційних технологій та обробки даних це можуть бути серверні приміщення. Такі приміщення містять потужне обладнання, яке потребує певних умов зберігання і є об'єктом підвищеного ризику. Працівники таких приміщень мають проходити щорічний інструктаж з техніки безпеки, а також проходити навчання, які симулюють виникнення надзвичайних ситуацій, щоб в разі виникнення реальної загрози, знати як діяти. Серверні приміщення належать до ПНО (потенційно небезпечних об'єктів), адже містять електроніку у великих кількостях, що робить об'єкт потенційно пожежонебезпечним. Тому там обов'язково мають бути зокрема і вогнегасники. Не допускається перегрів серверів, що може не тільки вивести їх з ладу, але й потенційно спричинити пожежу і навіть вибух. Також не допускається виникнення повеней у серверному приміщенні, адже якщо є приміщення з яких потім воду можна просто відкачати, то електроніка з водою фактично несумісна. Отже має бути відповідна герметизація, гідроізоляція такого об'єкту.

Загалом планування заходів цивільного захисту на об'єкті є досить комплексним процесом і його суть полягає в аналізі стану, оцінці обстановки що може статися під час виникнення аварії чи стихійного лиха, а також розробці заходів для захисту населення.

Важливою є також профілактика виникнення надзвичайних ситуацій. Це передбачає зокрема і запобігання виникненню надзвичайних ситуацій, наприклад відмову від продукції небезпечних виробництв (наприклад неперевірених китайських моделей серверів) та закриття аварійних об'єктів. Якщо причини виникнення НС усунути неможливо, то має бути принаймні запобігання ситуаціям виникнення. Якщо і це неможливо, то необхідно робити все для пом'якшення наслідків надзвичайних ситуацій, здійснювати стабілізаційні заходи або ж такі, які скомпенсують можливу шкоду та ризики.

Важливим етапом є також захист персоналу в разі надзвичайних ситуацій та ліквідації наслідків аварії. Звісно треба захищати всіх людей, які є потенційно в зоні ризику, але персонал звичайно ж що завжди в зоні найбільшого ризику, оскільки перебуває безпосередньо на об'єкті або поблизу нього. Це включає

наявність інформації про наявні захисні спорудження, укриття, їх клас, місткість і використовуються для екстренного укриття працівників [58].

За планування заходів захисту цивільного захисту відповідають місцеві чиновники, а також керівники конкретних об’єктів. Необхідно пам’ятати, що бувають НС, в яких задіяний не якийсь 1 об’єкт, а серія об’єктів або ж ціла територія. В такому випадку за планування відповідають міські, селищні голови. До етапів розроблення заходів захисту належить призначення розробників плану, аналіз законодавчої і нормативно-правової бази, збір і узагальнення даних та відомостей про об’єкт, територію, чисельність цивільного населення і працівників які можуть зазнати впливу НС. Сюди ж відноситься уточнення переліку об’єктів та територій небезпечних для працівників. Після цього відбувається практична розробка та оформлення плану, де зазначаються дії органів управління та сил захисту в режимі підвищеної готовності, сили і засоби доступні для залучення до наслідків НС, порядок залучення доступних сил а також порядок їх взаємодії тощо. Після цього план погоджується із заінтересованими підрозділами, особами та підписується і затверджується [59].

Отже, планування заходів цивільного захисту на об’єкті є надзвичайно важливим для уникнення та адекватної реакції на небезпечні обставини і мінімізацію їх впливу на людей (в умовах об’єкту в першу чергу на персонал). Надзвичайна ситуація зазвичай стається дуже швидко і до цього треба бути підготованими. Окрема увага має приділятися плануванням об’єктів, їх пропускній здатності (для можливості швидкого покинення великою кількістю людей). Не допускається утеплення об’єктів пінополістирольними та пінопластовими компонентами, особливо багатоповерхових, оскільки такі матеріали мають надзвичайно велику горючість, а також високу токсичність.

4.4 Висновок до четвертого розділу

В четвертому розділі кваліфікаційної роботи описано безпечні умови праці при виконанні завдань консолідації даних за комп’ютерним обладнанням, а також розглянуто алгоритм дій в разі виникнень надзвичайних ситуаціях в

робочих приміщеннях, та запобігання цьому впливу. Розглянуто деякі державні стандарти, які регулюють безпечні та нешкідливі умови праці. В цілому вони містять широкий перелік, однак з приводу деяких питань могли би бути доопрацьовані, наприклад, стосовно регуляції відсутності мерехтіння дисплеїв пристроїв при встановленні необхідного рівня яскравості.

ВИСНОВКИ

Отже, в дипломній роботі значну увагу було приділено проблемам та викликам консолідації даних, її необхідності, можливим інструментам для застосування та питанням швидкодії. Окремо були розглянуті питання забезпечення доступу до джерел інформації різних типів, а також уніфікації їх результатів між собою.

В першому розділі кваліфікаційної роботи освітнього рівня «Магістр»:

– Подано проблеми та виклики консолідації

– Розглянуто консолідацію інформації в цілому

– Висвітлено питання конкурентної розвідки

– Проаналізовано питання якості отримуваних даних з різних ресурсів

– Досліджено структурованість джерел інформації

– Обґрунтовано доцільність проведення консолідації даних

В другому розділі кваліфікаційної роботи:

– Описано конкретні інструменти, які можуть використовуватися для консолідації інформації

– Досліджено доцільність використання LINQ для завдань консолідації даних

– Подано порівняльний опис послідовної технології LINQ та паралельної PLINQ, а також їх швидкодії

В третьому розділі кваліфікаційної роботи:

– Запропоновано спосіб консолідації медичних даних значного обсягу з різногетерогених джерел інформації

– Протестовано зміни в швидкодії при паралелізації для різної кількості ядер.

У розділі «Охорона праці та безпека в надзвичайних ситуаціях» проаналізовано забезпечення безпечних умов праці та алгоритм дій в разі надзвичайних ситуацій.

ПЕРЕЛІК ДЖЕРЕЛ

1 Консолідація інформації як основа розвитку інформаційного суспільства XXI століття. [Електронний ресурс]. URL: https://sdc-journal.com.php/journal/article/view/362/296

2 Перспективні напрямки досліджень консолідації інформації та документних даних. [Електронний ресурс] / Лугова Тетяна Анатоліївна, Раєва Вікторія Русланівна, Безугла Світлана Миколаївна, Гребенюк Владислав Олександрович. URL: https://www.researchgate.net/publication/337044819_PERSPEKTIVNI_NAPRAMKI_DOSLIDZEN_KONSOLIDACII_INFORMACII_TA_DOKUMENTNIH_DANIH/fulltext/5dc2294e4585151435ec63a3/PERSPEKTIVNI-NAPRAMKI-DOSLIDZEN-KONSOLIDACII-INFORMACII-TA-DOKUMENTNIH-DANIH.pdf

3 Information consolidation and repackaging. [Електронний ресурс]. URL: https://www.egyankosh.ac.in/bitstream/123456789/35903/5/Unit-6.pdf

4 Information analysis and consolidation centres. [Електронний ресурс]. URL: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjr8_P_9OGEAxXD9rsIHa2jBcY4ChAWegQIDxAB&url=https%3A%2F%2Fzenodo.org%2Frecord%2F3782369%2Ffiles%2FINFORMATIONANALYSISAND.pdf&usg=AOvVaw2jdMv7qqe62KTx7hXfLaRM&opi=89978449

5 Консолідація даних з використанням стохастичних методів оптимізації. [Електронний ресурс]. URL: http://apeps.kpi.ua/downloads/Хомицький_2018.pdf

6 A survey of data center consolidation in cloud computing systems. [Електронний ресурс]. URL: http://ijeais.org/wp-content/uploads/2021/3/IJEAIS210311.pdf

7 Data consolidation and information aggregation in grid networks. [Електронний ресурс], P.Kokkinos, Manos Varvarigos. URL: https://www.researchgate.net/profile/Manos-Varvarigos/publication/221910989_Data_Consolidation_and_Information_Aggregati

on_in_Grid_Networks/links/02e7e5395cd3cb3142000000/Data-Consolidation-and-Information-Aggregation-in-Grid-Networks.pdf

8 Consolidated information resource in factoring operations. [Електронний ресурс], N.Vovk, M.Komova. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2017/may/2365/vovk.pdf

9 Consolidation of information, a handbook on evaluation, restructuring and repackaging of scientific and technical information. [Електронний ресурс]. URL: https://unesdoc.unesco.org/ark:/48223/pf0000047738/PDF/047738engo.pdf.multi

10 Класифікація та особливості збору даних у системах підтримки прийняття управлінських рішень. [Електронний ресурс], Антоненко А.В, Бондаренко Є.С, Каньшин К.В, Степанчук В.І, Ланевський Л.А. URL: http://journals.ksauniv.ks.ua/index.php/tech/article/download/412/380/772

11 Modelling of consolidated information resource for social data institutions. [Електронний ресурс], N. Kunanets, V. Pasichnyk, H. Lypak, O. Duda, O. Matsiuk. URL: https://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-7c4963e1-2a6f-47f1-ad09-5aef96c88613/c/Kunanets_25_30-1.pdf

12 Збирання і джерела даних. [Електронний ресурс]. URL: https://socialdata.org.ua/manual/manual1/

13 Особливості побудови підсистеми збирання, попередньої обробки та збереження медичних даних. [Електронний ресурс] / Батюк.А, Пилипчук. С, Цмоць І. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2024/feb/33373/vis663komp-nauky-138-143.pdf

14 Integrating multiple data sources for learning analytics – review of literature. [Електронний ресурс], Jeanette Samuelsen, Weiqin Chen, Barbara Wasson. URL: https://d-nb.info/1202638457/34

15 Порівняння ефективності методів некерованого машинного навчання для виявлення аномалій в OBD2 даних. [Електронний ресурс], Матійчук, Готович, Бонар. URL: https://doi.org/10.31891/2219-9365-2025-81-52

16 Консолідація інформації та інформаційна безпека. [Електронний ресурс] / Н.Е. Кунанець, В.В. Пасічник. URL: http://irbis-nbuv.gov.ua/cgi-

bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&P21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/soi_2010_3_56.pdf

17 Консолідація інформації про діяльність учасників групи в соціальній мережі Facebook. [Електронний ресурс] / Олександр Марковець, Руслана Паздерська. URL: http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&P21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/vkp_2019_6_7.pdf

18 Оцінювання якості консолідованих даних. [Електронний ресурс] / Н.Б. Шаховська. URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/113605/05-Shahovska.pdf?sequence=1

19 Методи опрацювання консолідованих даних за допомогою просторів даних. [Електронний ресурс] / Н.Б. Шаховська. URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/51001/8_s_72_84.pdf?sequence=1

20 Модель консолідованих даних та їх опрацювання за умов невизначеності. [Електронний ресурс], Шаховська Н. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2024/feb/33838/vis751komp-nauky-203-212.pdf

21 Дослідження якості консолідованих даних у просторах даних. [Електронний ресурс], Н.Б. Шаховська: URL: http://www.immsp.kiev.ua/publications/articles/2012/2012_1/01_2012_Shakhovska.pdf

22 Проблеми якості консолідованих даних у просторах даних. [Електронний ресурс], Н.Б. Шаховська, О.Ю. Пшеничний, І.М. Чорней. URL: http://irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&P21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/soi_2011_3_20.pdf

23 Проблеми та визначення якості консолідованих даних. [Електронний ресурс], Думанський Нестор, Видойник Марія. URL: https://d1wqtxts1xzle7.cloudfront.net/53273386/ICS-2017-Proceedings-103-104-

libre.pdf?1495715121=&response-content-disposition=attachment%3B+filename%3D53273386.pdf

24 Моделі та методи диференціації даних консолідованої інформації для систем підтримки рішень. [Електронний ресурс, дисертація] / Буслов Павло Володимирович, ХНУРЕ. URL: https: https://er.chdtu.edu.ua/bitstream/ChSTU/2067/12/dis_Буслов П.В..pdf

25 Напрями консолідації в інформаційному забезпеченні правової науки України. [Електронний ресурс, монографія] / Ю.В. Пасмор. URL: https://library.nlu.edu.ua/POLN_TEXT/MONOGRAFII_2015/Pasmor_2013.pdf

26 Консолідація облікової інформації в управлінні діяльністю підприємств та їх об'єднань. [Електронний ресурс, монографія] / А.А. Пилипенко. URL: http://repository.hneu.edu.ua/bitstream/123456789/19634/1/2016-Пилипенко А А.pdf

27 Конкурентна розвідка як інструмент інформаційно-аналітичного супроводу забезпечення інформаційної безпеки підприємства. [Електронний ресурс] / Мужанова Т. М. URL: https://economyandsociety.in.ua/journals/16_ukr/65.pdf

28 Правові питання конкурентної розвідки. [Електронний ресурс] / Ланде Д.В. URL: https://ippi.org.ua/sites/default/files/7_16.pdf

29 Система консолідованої інформації ТНК. [Електронний ресурс], Федоронько Н.І, Вайда О.С. URL: http://molodyvcheny.in.ua/files/journal/2016/7/36.pdf

30 Консолідація даних у бізнес-аналізі діяльності підприємств. [Електронний ресурс], Зайцева Н.В, Яковенко В.С. URL: global-national.in.ua/archive/8-2015/257.pdf

31 Економічна та інформаційна безпека підприємств у системі консолідованої інформації. [Навчальний посібник, електронний ресурс], Кавун С.В, Пилипенко А.А, Ріпка Д.О. URL: http://www.repository.hneu.edu.ua/bitstream/123456789/6818/1/Економічна та інформаційна безпека підприємств у системі консолідованої інформації%20 навчальний посібник.pdf

32 Розробка схеми консолідації інформації на великому підприємстві. [Електронний ресурс], Н.О. Нещадим. URL: https://dspace.nuft.edu.ua/server/api/core/bitstreams/42b1f6f5-c981-4972-a523-cd9097215212/content

33 Методи опрацювання консолідованих даних за допомогою просторів даних. [Електронний ресурс] / Н.Б. Шаховська. URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/51001/8_s_72_84.pdf?sequence=1

34 John Paul Mueller. LINQ for dummies. P.105-210. https://srikanthkamuju.wordpress.com/wp-content/uploads/2011/05/free-ebooks-download-org-for-dummies-linq-for-dummies.pdf

35 LINQ ROX!: integrating LINQ into the database curriculum. [Електронний ресурс] / Mahesh Chaudhari, Suzanne Dietrich. URL: https://www.researchgate.net/publication/228727592_LINQ_ROX_integrating_LINQ_into_the_database_curriculum

36 Marian Rusek, Jakub Maguza, Waldemar Karwowski. Integration of queries to heterogeneous data sources using LINQ technology. https://www.infona.pl/resource/bwmeta1.element.baztech-4e4868a0-984f-4a92-8427-e08bcf08c1b4/content/partContents/c62cd082-38a8-387a-a04c-c481345a5313

37 Harry McIntyre, Luke McGregor. LINQ to anything library. https://github.com/mcintyre321/LinqToAnything

38 Dryad and DryadLINQ, unlocking the power of parallelism and data centers. [Електронний ресурс]. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2be87009638e4e36b3c080faa1a5b306356f6d35

39 Використання великих даних в розумному місті. [Електронний ресурс] / Г. Козбур, Є. Гоцко. URL: https://elartu.tntu.edu.ua/handle/lib/37594

40 Підходи до інтеграції розподілених даних на базі Polystore. [Електронний ресурс], І.С. Зінов'єва, Н.В. Ситник

41 Harun Cerim. Extendind C# with a library of functional programming concepts.Md. Rashedul Islam, Aloke Kumar Shah, Md. Rofiqul Islam. Experimental evaluation of parallelism in real time execution. P.15-22. Dspace. https://dspace.cuni.cz/bitstream/handle/20.500.11956/121344/120370216.pdf

42 Processing and analysis of trolleybus traction data using LINQ. [Електронний ресурс] / A. Wilk, M. Bartłomiejczyk, J. Skibicki, L. Jarzębowicz, D. Karkosiński, Ł. Hupka, J. Hupka, P. Kaczmarek, N. Karkosińska-Brzozowska. URL: https://journals.pan.pl/Content/134556/PDF/BPASTS-04722-EA.pdf

43 B. Frackowiak, R. Dabrowski. Using LINQ as an universal tool for defining architectural assertions. Federated conference of computer science and information systems. https://annals-csis.org/Volume_8/pliks/pliks/587.pdf

44 Eliminate SQL injection using LINQ. [Електронний ресурс] / URL: https://scispace.com/pdf/eliminate-sql-injection-using-linq-2lx9s50r1t.pdf

45 Joseph Albahari. Why LINQ beats SQL. https://www.linqpad.net/WhyLINQBeatsSQL.aspx

46 Inspection of guided missiles applied with parallel processing algorithm. [Електронний ресурс], Eul-Jae Jung, Sang-Hoon Koh, You-Sang Lee, Young-Sung Kim. URL: https://koreascience.kr/article/JAKO202125658906561.pdf

47 13 Міжнародна науково-практична конференція молодих учених та студентів “Актуальні задачі сучасних технологій”. В.А. Лабчук. Дослідження покращення швидкодії обробки даних у PLINQ в порівнянні з LINQ для різних розмірів вхідних даних.. P.430-431.

48 Optimizing big data processing through lazy computaions. A systematic review of techniques and applications. [Електронний ресурс], M.V. Talakh, Yu.O. Usenko, E.V. Vatamanitsa, Yu. Halin. URL: https://jnas.nbuv.gov.ua/j-pdf/oeiet_2024_2_5.pdf

49 Efficient query processing in managed runtimes, p19. [Електронний ресурс], Fabian Nagel. URL: https://era.ed.ac.uk/bitstream/handle/1842/15869/Nagel2015.pdf?sequence=1&isAllowed=y

50 Big data: концепції, терміни та параметризація. [Електронний ресурс], Дуда, Кунанець, Мацюк, Пасічник. URL: https://scholar.google.com.ua/citations?view_op=view_citation&hl=uk&user=1nIRL6sAAAAJ&cstart=20&pagesize=80&citation_for_view=1nIRL6sAAAAJ:MhiOAD_qIWkC

51 Великі за обсягом набори біомедичних даних та машинне навчання. [Електронний ресурс], В.А. Готович, Л.В. Волинець, Н.А. Гарматюк. URL: https://elartu.tntu.edu.ua/bitstream/lib/43843/2/MNPK_2023_Volynets_L_V-Big_biomedical_data_370-371.pdf

52 Secure information system for chinese image medicine knowledge consolidation. [Електронний ресурс], H. Lypak, S. Lupenko, O. Orobchuk, I. Kateryniuk, R. Kozak. URL: https://ceur-ws.org/vol-3896/paper30.pdf

53 Державні санітарні правила та норми “Влаштування і обладнання кабінетів комп’ютерної техніки в навчальних закладах та режим праці учнів на персональних комп’ютерах”. ДсанПін 5.5.6.009-98. [Електронний ресурс]. URL: https://zakon.rada.gov.ua/rada/show/v0009588-98/ed20170626#Text

54 Національний стандарт України. Світло та освітлення. Освітлення робочих місць. Внутрішні робочі місця. ДСТУ EN 12464-1:2016. https://pdfcoffee.com/-en-12464-1-2016--pdf-free.html

55 Зниження втоми очей при роботі в режимі дистанційного та комбінованого навчання. Братусь І.В, Свердлик З.М, Гунька А.М. URL: https://elibrary.kubg.edu.ua/id/eprint/36533/1/I_Bratus_Z_Sverdlyk_A_Hunka_MV_4.pdf

56 Шкідливий вплив сучасних моніторів та робочого положення на якість зору. [Електронний ресурс], Зима О.Є, Нікітченко Є. URL: hhttps://reposit.nupp.edu.ua/bitstream/PoltNTU/12590/1/75%20%D0%A2.2-160-161.pdf

57 Санітарні норми мікроклімату виробничих приміщень. ДСН 3.3.6.042-99. https://zakon.rada.gov.ua/rada/show/va042282-99#Text

58 Цивільна безпека. Методика планування заходів цивільного захисту на потенційно небезпечних об’єктах. Дніпровський державний технічний

університет, м. Кам’янське. Левчук К.О, Романюк Р.Я. DOI 10.31319/2519-2884.34.2019.28

59 Державна служба України з надзвичайних ситуацій. Інститут державного управління та наукових досліджень з цивільного захисту. Організація планування заходів у сфері цивільного захисту. Серія 5. 2021

ДОДАТКИ

Додаток А

Вихідний код отримання даних з CSV-файлу

```
using LINQtoCSV;
CsvContext cc=new CsvContext();
IEnumerable<Doctor>
doctors=cc.Read<Doctor>("pmg_all_pmd_doctor_info.csv");
IEnumerable<Division>
divisions=cc.Read<Division>("pmg_legal_entity_divisions_info.csv")
```

Додаток Б

Вихідний код отримання даних з API

```
static async Task<JsonElement.ArrayEnumerator> DataFromApi()
{
    HttpClient client = new HttpClient();
    HttpRequestMessage request = new HttpRequestMessage
    {
        Method = HttpMethod.Get,
        RequestUri                    =                    new
Uri($"https://medics.ua/api/v1/doctors?name={name}&offset=0&search
=true")
    };
   HttpResponseMessage response = await client.SendAsync(request);
    string apiData = await response.Content.ReadAsStringAsync();
    JsonDocument doc = JsonDocument.Parse(apiData);
    JsonElement root = doc.RootElement;
    JsonElement.ArrayEnumerator jsonArray = root.EnumerateArray();
    return jsonArray;
}
```

Додаток В

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний технічний університет імені Івана Пулюя (Україна)
Університет імені П'єра і Марії Кюрі (Франція)
Маріборський університет (Словенія)
Технічний університет у Кошице (Словаччина)
Вільнюський технічний університет ім. Гедимінаса (Литва)
Наукове товариство ім. Т.Шевченка

АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ

Збірник
тез доповідей

XIII Міжнародної науково-практичної конференції молодих учених та студентів
11-12 грудня 2024 року

УКРАЇНА
ТЕРНОПІЛЬ – 2024

А43
Актуальні задачі сучасних технологій : зб. тез доповідей XIII міжнар. наук.-практ. конф. Молодих учених та студентів, (Тернопіль, 11-12 грудня 2024) / М-во освіти і науки України, Терн. націон. техн. ун-т ім. І. Пулюя [та ін.]. – Тернопіль: ФОП Паляниця В. А., 2024. – 543.
ISBN 978-617-7875-88-7

ПРОГРАМНИЙ КОМІТЕТ

Адреса оргкомітету:
ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001,
тел. 0971640355, факс (0352) 255798
E-mail: **sergiigladol@gmail.com**
Редагування, оформлення, верстка: Гладьо С.В.

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- фізико-технічні основи розвитку нових технологій;
- нові матеріали, міцність і довговічність елементів конструкцій;
- сучасні технології в будівництві, машино- та приладобудуванні;
- сучасні технології на транспорті;
- електротехніка та енергозбереження;
- фундаментальні проблеми харчових, біо- та нанотехнологій;
- економічні та соціальні аспекти нових технологій;
- комп'ютерно-інформаційні технології та системи зв'язку.

Матеріали XIII Міжнародної науково-практичної конференції молодих учених та студентів «АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ» – Тернопіль, 11-12 грудня 2024 року

УДК 621.855

В.А. Лабчук
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ДОСЛІДЖЕННЯ ПОКРАЩЕННЯ ШВИДКОДІЇ ОБРОБКИ ДАНИХ У PLINQ В ПОРІВНЯННІ З LINQ ДЛЯ РІЗНИХ РОЗМІРІВ ВХІДНИХ ДАНИХ

V.A. Labchuk

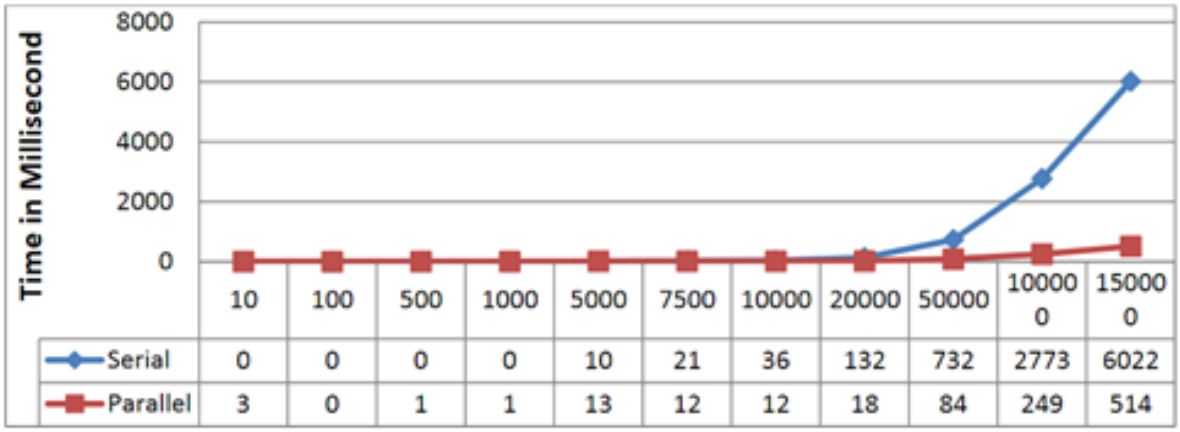
RESEARCH OF THE DATA PROCESSING PERFORMANCE IMPROVEMENT USING PLINQ IN COMPARISON WITH LINQ ON DIFFERENT SIZED DATASETS

Під час консолідації даних з різних ресурсів ми зазвичай отримуємо дані великих обсягів. Щоб продуктивність їх обробки була високою, доцільно використовувати не лише LINQ, але й його паралельну версію – PLINQ. Однак чи в усіх випадках PLINQ буде продуктивніше ніж LINQ? Чи можливі ситуації коли PLINQ взагалі програватиме у швидкості? Коротка відповідь – так.

Можуть бути різні причини, коли паралельна обробка LINQ не буде виконуватися швидше ніж послідовна, зокрема розмір вибірки. Розмір вибірки є ключовим параметром і безпосередньо впливає на швидкість обробки. Звісно, чим більший розмір наших даних, тим довше їх потрібно обробляти. Однак саме в таких випадках спостерігається користь від паралельної обробки. Якщо ж даних у нас мало, то PLINQ не здатний забезпечити перевагу у швидкості, а деколи навіть навпаки потребуватиме додаткового часу на розпаралелювання, а також синхронізацію потоків, яка непотрібна при послідовному виконанні. Тобто якщо ми знаємо, що набір даних передбачається невеликий, використання PLINQ буде лише на шкоду.

Однак позитивним фактором від використання PLINQ саме при консолідації є те, як зростає час обробки при збільшенні розміру вибірки. Під час використання LINQ час збільшується експоненційноподібно, і вже на вибірках середнього розміру починає перебільшувати відповідний час при використанні LINQ. Використання ж PLINQ дозволяє часу збільшуватися більш лінійно при збільшенні обсягу вибірки. Дослідимо залежність часу знаходження простих чисел (рис. 1) від розміру вибірки, тобто діапазону в якому ми шукаємо ці прості числа [1].

Висновок. Досліджено залежність швидкості обробки та пошуку даних залежно від розміру вхідних даних. Зокрема досліджено час пошуку простих чисел на різних діапазонах для PLINQ та LINQ. Отримані результати показують велику вигоду від використання паралельної версії LINQ на великих вхідних масивах, які зазвичай і виникають при консолідації даних, та беззмістовність її використання для вхідних вибірок невеликих розмірів.



	10	100	500	1000	5000	7500	10000	20000	50000	100000	150000
Serial	0	0	0	0	10	21	36	132	732	2773	6022
Parallel	3	0	1	1	13	12	12	18	84	249	514

Рисунок 1. Порівняння паралельної та послідовної обробки даних LINQ та PLINQ на різних розмірах вхідних даних

Література

1. Md. Rashedul Islam, Aloke Kumar Shah, Md. Rofiqul Islam. Experimental evaluation of parallelism in real time execution No 36, P.49-51. Academia edu https://d1wqtxts1xzle7.cloudfront.net/32324848/RNIS114PPL-libre.pdf?1391541786=&response-content-disposition=inline%3B+filename%3DExperimental_Evaluation_of_parallelism_i.pdf&Expires=1733085003&Signature=T2Xh9IZd7EQYufmU~i0UzlXfDVHFdS3xp2gSP~GrOzrDqNxtKB0Ex0J6qIfEHgcx7pRjiXSBtTCdC5ekSsRtHgS3LrR-Wl5W0MRTDUe4Jv~2eejLdEmKYqtWdZCom5c3y5kPfejwQX5ZlhPB3HRR7cghk4xdrEA6-EFZyCpU2lwMOV~~w32ZYd4J0c7Odun8wgLmaFSkf20B2q6aUPJpLN0mVdBMdH7brNPvbIF44PSqn5ldU9r~5emEwqTZ7Oe~DfUemlV~gdFaqJamOjaPC8E2eS4GHD5cIGHfZns4DxHqYBdr6RHws8dY~hgs6W-P4cZUomLxuzf1Y14u87P5g_&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18–19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 001
М34

ПРОГРАМНИЙ КОМІТЕТ

Голова: ПриймакМикола – професор кафедри комп'ютерних систем тамереж, д.т.н., професор.
Співголови: Марущак Павло – проректор з наукової роботи, докт. техн. наук, професор.
Баран Ігор – канд. техн. наук, доцент, декан факультету ФІС.
Науковий секретар: Надія Крива – старший викладач.

Члени: Василь Кривень – завідувач кафедри математичних методів в інженерії д.ф.-м.н., професор; Галина Осухівська – завідувач кафедри комп'ютерних систем та мереж, к.т.н., доцент; Микола Карпінський - професор кафедри кібербезпеки, д.т.н., професор; Жанна Баб'як - завідувач кафедри української та іноземних мов, к.пед. н., доцент; Ярослав Литвиненко – професор кафедри комп'ютерних наук, д.т.н., професор; Михайло Петрик - завідувач кафедри програмної іженерії, д.ф.-м.н., професор; Наталія Загородна – завідувач кафедри кібербезпеки, к.т.н., доцент.

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

Голова: Скоренький Юрій Любомирович – канд. техн. наук, доцент кафедри фізики.
Члени: доцент кафедри комп'ютерних наук, к.т.н. В. Никитюк; доцент кафедри програмної інженерії, к.т.н. Д. Михалик; доцент кафедри кібербезпеки, к.т.н. М. Стадник; доцент кафедри комп'ютерних систем та мереж, к.т.н. Є. Тиш; ст. викладач Л. Джиджора.

М34 Матеріали XII науково-технічної конфіції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 18–19 грудня 2024 р.). – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2024. – 238 с.

Адреса оргкомітету: ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001, тел. (0352) 52-41-33, факс (0352) 25-49-83.
E-mail: conffis2024@gmail.com
Редагування, оформлення та верстка: Надія Крива

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- Математичне моделювання
- Інформаційні системи та технології, кібербезпека
- Комп'ютерні системи та мережі
- Програмна інженерія та моделювання складних розподілених систем
- Новітні фізико-технічні та освітні технології

В збірнику надруковано тези доповідей XII науково-технічної конференції «Інформаційні моделі, системи та технології» (Тернопіль, 18–19 грудня 2024 р.) за такими науковими напрямками: математичне моделювання; інформаційні системи та технології, кібербезпека; комп'ютерні системи та мережі; програмна інженерія та моделювання складних розподілених систем; новітні фізико-технічні та освітні технології.

Розрахований на науковців, викладачів та студентів вузів.

За зміст тез та дотримання норм академічної доброчесності відповідальність несе автор.

УДК 004.6
В. А. Лабчук
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ПЕРЕВАГИ ТА НЕДОЛІКИ LINQ В ПОРІВНЯННІ З SQL-КОДОМ

UDC 004.6
V. A. Labchuk

PROS AND CONS OF USING LINQ INSTEAD OF RAW-SQL

Використання LINQ надає багато переваг в порівнянні з аналогічними запитами SQL. Такі запити більш високорівневі та інтегровані в мову програмування. Звідси слідує, що такий код компільований, а отже виявлення помилок відбувається під час компіляції, а не під час виконання. Крім того ризик SQL-ін'єкцій, тобто вставок, значно послаблюється, оскільки не використовується звичайне об'єднання значень рядків, значення дозволяється передавати в якості параметрів. А для вибірки даних з різних таблиць чи сутностей непотрібно з'єднувати таблиці, а достатньо викликати відповідну властивість цього об'єкту, що скорочує код. До того ж LINQ може працювати не лише зі структурованими даними з баз даних, але й наприклад зі слабкоструктурованими, що робить цю мову запитів універсальнішою [1].

Мова запитів LINQ зокрема застосовується і в інших інструментах Microsoft, наприклад, Entity Framework Core. Хоча дана технологія дозволяє використовувати і чистий SQL-код, однак необхідно враховувати доцільність цього в конкретній ситуації, а також пам'ятати про архітектуру програмного продукту. Враховуючи переваги мови запитів, в контексті архітектури в більшості випадків доцільніше використовувати саме її. Крім того її застосування підходить для здійснення архітектурних припущень, аналізу патернів використання та антипатернів [2].

Однак у мови запитів є і недоліки та ситуації, коли доцільніше використовувати код SQL. Наприклад, швидкість виконання запитів LINQ може бути нижчою ніж у запитів SQL. Крім того використання коду SQL потенційно може знадобитися для деяких складних запитів, а також його доведеться застосовувати для того, щоб використовувати тригери, тобто реакцію на якусь подію. Однак відсоток випадків, коли дійсно потрібно застосовувати SQL, є незначним, близько 5% [3].

Література

1. Harun Cerim. Extending C# with a library of functional programming concepts.Md. Rashedul Islam, Aloke Kumar Shah, Md. Rofiqul Islam. Experimental evaluation of parallelism in real time execution. P. 15-22. Dspace. URL: https://dspace.cuni.cz/bitstream/handle/20.500.11956/121344/120370216.pdf?sequence=1&isAllowed=y.
2. Bartosz Frackowiak, Robert Dabrowski. Using LINQ as an universal tool for defining architectural assertions. Federated conference of computer science and information systems. URL: https://annals-csis.org/Volume_8/pliks/pliks/587.pdf.
3. Joseph Albahari. Why LINQ beats SQL. URL: https://www.linqpad.net/WhyLINQBeatsSQL.aspx.

УДК 004.6
В. А. Лабчук
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ ЗАСТОСУВАННЯ MICROSOFT LINQ ДЛЯ ЗАДАЧ КОНСОЛІДАЦІЇ ДАНИХ

UDC 004.6
V. A. Labchuk

RESEARCHMENT OF THE DATA CONSOLIDATION TASKS' OPPORTUNITIES WITH THE HELP OF USING MICROSOFT LINQ TECHNOLOGY

При консолідації даних з різних ресурсів постає питання роботи з даними різних форматів і приведення їх до єдиного. LINQ вміє працювати з різними джерелами даних, зокрема отриманих у вигляді наборів даних (зокрема з CSV-файлів), сутностей (описуються класами), баз даних, xml-файлів тощо. Для цього існують відповідні імплементації LINQ to SQL, LINQ to XML, LINQ to Dataset, LINQ to Entities. Хоча на сьогодні велика частина інформації зберігається у базах даних, однак звісно далеко не вся. До того ж дані можуть бути як структурованими, так і напівструктурованими, і неструктурованими взагалі [1].

Для роботи з LINQ може використовуватися два C# інтерфейси – IEnumerable та IQueryable. Перший підходить для ітерації колекцією об'єктів, другий – для зовнішніх ресурсів, наприклад роботи з базою даних. В даному випадку LINQ не обмежується лише Microsoft SQL Server, але й може працювати з іншими постачальниками даних. Тобто фактично LINQ створює абстракцію рівня доступу до даних. При цьому запити можна створювати в SQL-подібному стилі або ж за допомогою методів розширення. Microsoft рекомендує застосовувати саме методи розширення в ситуаціях коли це можливо [2].

Насправді на цьому можливості LINQ не закінчуються і є можливість інтегруватися і з іншими технологіями для отримання даних. Однак деякі зі специфічних розширень можуть постачатися третіми сторонами. Зокрема можна згадати LINQ to Twitter для взаємодії з відповідною соцмережею, LINQ to Result (для підтримки залізнично-орієнтованого програмування в мові C#), LINQ to Anything. Останній варіант дозволяє перетворити будь-яке джерело даних у IQueryable для зручної взаємодії з ним [3].

Висновок. Досліджено можливості застосування технології Microsoft LINQ для задач консолідації даних. Ця мова дозволяє нам абстрагуватися від різноманіття форм даних, що надходять на вхід і сконцентруватися безпосередньо на роботі з даними, а отже може бути доцільним інструментом для використання в задачах консолідації даних. Для кращої продуктивності при об'єднанні великих масивів даних краще використовувати паралельну версію даного інструменту – PLINQ, однак це не варто робити при роботі на 1-ядерних пристроях та коли масиви даних незначні за обсягом.

Література

1. John Paul Mueller. LINQ for dummies. P.105-210. URL: https://srikanthkamuju.wordpress.com/wp-content/uploads/2011/05/free-ebooks-download-org-for-dummies-linq-for-dummies.pdf.
2. Marian Rusek, Jakub Maguza, Waldemar Karwowski. Integration of queries to heterogeneous data sources using LINQ technology. URL: https://www.infona.pl/resource/bwmetal.element.baztech-4e4868a0-984f-4a92-8427-e08bcf08c1b4/content/partContents/c62cd082-38a8-387a-a04c-c481345a5313.
3. Harry McIntyre, Luke McGregor. LINQ to anything library. URL: https://github.com/mcintyre321/LinqToAnything.