ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «**Система контролю цілісності та актуальності бази даних у корпоративних ERP-системах**»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
(код,найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Дарія ЄМЕЛЬЯНОВА
(підпис) *(ім'я, ПРІЗВИЩЕ здобувача)*

Виконала: здобувачка вищої освіти група ІСД-42
Дарія ЄМЕЛЬЯНОВА
(ім'я, ПРІЗВИЩЕ)
Керівник *PhD* Віктор САГАЙДАК
(ім'я, ПРІЗВИЩЕ)
Рецензент: _____
(ім'я, ПРІЗВИЩЕ)

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інформаційних систем та технологій
Ступінь вищої освіти бакалавр
Спеціальність 126 Інформаційні системи та технології
Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ
Каміла СТОРЧАК
«___» _____2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Ємельянова Дарія Юріївна

(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Система контролю цілісності та актуальності бази даних у корпоративних ERP-системах

керівник кваліфікаційної роботи Віктор САГАЙДАК, PhD,

(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» 02 2026р. № 51

2. Строк подання кваліфікаційної роботи «01» 06 2026р.

3. Вихідні дані до кваліфікаційної роботи: принципи управління якістю даних у корпоративних ERP-системах, архітектура та структура бази даних Microsoft Dynamics NAV, науково-технічна література з тематики Data Quality Management.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
   1. Дослідження теоретичних засад управління якістю даних у корпоративних інформаційних системах та огляд ефективних методологій валідації.
   2. Аналіз якості товарних даних у Dynamics NAV та порівняльний огляд існуючих інструментів контролю якості даних.

3. Розробка персоналізованого інструменту контролю цілісності та актуальності даних у корпоративній ERP-системі..

5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання «20» 02 2026р.

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір технічної літератури	23.02 – 03.03.2026	
2	Дослідження теоретичних засад управління якістю даних та методологій валідації	04.03 – 17.03.2026	
3	Аналіз якості товарних даних та огляд існуючих інструментів контролю	18.03 – 31.03.26	
4	Розробка архітектури та реалізація функціональних модулів інструменту	01.04 – 24.04.2026	
5	Тестування, оцінка ефективності та висновки	27.04 – 06.05.2026	
6	Розробка демонстраційних матеріалів, доповідь	07.05 – 08.05.2026	
7	Оформлення кваліфікаційної роботи	11.05 – 18.05.2026	

Здобувачка вищої освіти _____ Дарія ЄМЕЛЬЯНОВА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Віктор САГАЙДАК
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 67 стор., 2 табл., 45 рис., 29 джерел.

Мета роботи – розробка персоналізованого інструменту самоконтролю якості даних у корпоративній ERP-системі, що забезпечує виявлення логічних помилок у товарних картках без залучення IT-підрозділу та без модифікації ERP-системи.

Об'єкт дослідження – процеси контролю цілісності та актуальності товарних даних у корпоративних ERP-системах.

Предмет дослідження – система валідації товарних даних у корпоративній ERP-системі Microsoft Dynamics NAV.

Короткий зміст роботи: у роботі досліджено сучасні підходи до управління якістю даних у корпоративних інформаційних системах та проаналізовано типові помилки в товарних даних ERP-системи Dynamics NAV. Розглянуто інструменти контролю якості даних, зокрема вбудовану валідацію Dynamics NAV, Microsoft SQL Server Data Quality Services, Great Expectations і Talend Data Fabric. Розроблено архітектуру та алгоритм роботи інструменту валідації, а також реалізовано ключові функції: автоматичне зчитування коду менеджера з PERSONAL.XLSB, формування персоналізованої вибірки помилок, підключення до бази даних Dynamics NAV через ADO-з'єднання з параметризованими SQL-запитами, перевірку комерційно-критичних помилок, логічної узгодженості атрибутів і заповненості обов'язкових полів. Інструмент створено у форматі макросної Excel-книги .xlsb із трирівневою архітектурою, що не потребує модифікації ERP-системи та може адаптуватися до внутрішніх бізнес-правил підприємства.

КЛЮЧОВІ СЛОВА: ERP-СИСТЕМА, ЯКІСТЬ ДАНИХ, ВАЛІДАЦІЯ, DYNAMICS NAV, SQL, VBA, ПЕРСОНАЛІЗАЦІЯ, ЦІЛІСНІСТЬ ДАНИХ, ТОВАРНІ ДАНІ, DATA QUALITY MANAGEMENT.

ABSTRACT

The text part of the qualification work for a bachelor's degree: 67 pages, 2 tables, 45 figures, 29 sources.

The purpose of the work – development of a personalized data quality self-monitoring tool for a corporate ERP system, designed to detect logical errors in product records without involving the IT department and without modifying the ERP system.

The object of research – processes for controlling the integrity and relevance of product data in corporate ERP systems.

The subject of research is a product data validation system within the corporate ERP system Microsoft Dynamics NAV.

Summary of the work: the study examines modern approaches to data quality management in corporate information systems and analyzes typical errors in product data within the Dynamics NAV ERP system. Existing data quality control tools are reviewed, including the built-in Dynamics NAV validation mechanism, Microsoft SQL Server Data Quality Services, Great Expectations, and Talend Data Fabric. The architecture and operating algorithm of the validation tool were developed, and its key functions were implemented: automatic retrieval of the manager code from PERSONAL.XLSB, generation of a personalized error report, connection to the Dynamics NAV database via ADO with parameterized SQL queries, and validation of commercially critical errors, logical consistency of attributes, and completeness of mandatory fields. The tool is implemented as an Excel macro-enabled .xlsb workbook with a three-tier architecture that does not require ERP system modification and can be adapted to the company’s internal business rules.

KEYWORDS: ERP SYSTEM, DATA QUALITY, VALIDATION, DYNAMICS NAV, SQL, VBA, PERSONALIZATION, DATA INTEGRITY, PRODUCT DATA, DATA QUALITY MANAGEMENT.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

SQL – Structured Query Language

ERP – Enterprise Resource Planning

DQM – Data Quality Management

VBA – Visual Basic for Applications

ADO – ActiveX Data Objects

NAV – Microsoft Dynamics NAV

DQS – Data Quality Services

SSIS – SQL Server Integration Services

ETL – Extract, Transform, Load

GX – Great Expectations

XLSB – Excel Binary Workbook

KPI – Key Performance Indicators

B2B – Business-to-Business

OE – Original Equipment

УКТЗЕД – Українська класифікація товарів зовнішньоекономічної діяльності

BI – Business Intelligence

IT - Information Technology

ВСТУП

Актуальність. У сучасних корпоративних ERP-системах дані є основою всіх бізнес-процесів – від формування замовлень і цінових розрахунків до управлінської аналітики та митного оформлення. Коли в базі накопичуються логічні помилки, такі як дублі цін, незаповнені обов'язкові поля, порушення ієрархічних зв'язків між довідниками, система формально продовжує працювати, але результати її роботи стають ненадійними. Виявлення таких помилок є нетривіальним завданням, оскільки стандартні механізми ERP контролюють лише формат і тип даних, але не їхню логічну узгодженість.

Розробка інструменту контролю цілісності та актуальності даних є актуальною з кількох причин. По-перше, вбудовані засоби валідації Dynamics NAV не здатні виявляти семантичні суперечності між полями та не піддаються гнучкому налаштуванню без втручання розробника. По-друге, існуючі зовнішні рішення, такі як Great Expectations або Talend, є неприйнятними для рівня окремого аналітика, оскільки потребують окремої технічної інфраструктури, програмістських компетенцій та значних фінансових витрат. По-третє, жоден із розглянутих аналогів не забезпечує персоналізації результатів, через що кожен аналітик змушений вручну фільтрувати масивні загальні звіти, щоб знайти помилки саме у своїх даних. Окрім цього, наявна потреба в рішенні, яке є безкоштовним, не потребує модифікації ERP-системи та може бути легко адаптоване під внутрішні бізнес-правила конкретного підприємства. Нарешті, практична підтвердженість підходу засвідчується тим, що розроблений інструмент вже впроваджений і використовується в реальній компанії, що доводить його ефективність та відтворюваність в аналогічних корпоративних середовищах.

Об'єктом дослідження є процеси контролю цілісності та актуальності товарних даних у корпоративних ERP-системах.

Предметом дослідження є система валідації товарних даних у корпоративній ERP-системі Microsoft Dynamics NAV.

Метою роботи є розробка персоналізованого інструменту самоконтролю якості даних, що забезпечує виявлення логічних помилок у товарних картках без залучення IT-підрозділу та без модифікації ERP-системи.

Методи дослідження: першим кроком було проведено дослідження у сфері управління якістю даних у корпоративних інформаційних системах та визначено ключові характеристики якості даних в ERP-середовищі. Другим кроком було проаналізовано існуючі рішення для валідації даних – вбудовану валідацію Dynamics NAV, Microsoft SQL Server Data Quality Services, фреймворк Great Expectations та платформу Talend Data Fabric, з метою виявлення їхніх функціональних обмежень та формулювання вимог до розроблюваного інструменту. Після цього було проведено класифікацію типових помилок у товарних даних аналітичного відділу та визначено перелік необхідних перевірок бізнес-логіки. На основі отриманих результатів розроблено архітектуру системи та реалізовано інструмент у вигляді макросної Excel-книги (.xlsb) з прямим підключенням до бази даних Dynamics NAV через ADO-з'єднання та параметризованими SQL-запитами. Дослідження практичної ефективності проводилося під час безпосереднього впровадження та експлуатації інструменту в реальному робочому середовищі компанії.

Наукова новизна роботи полягає в таких складових: розроблено механізм персоналізованої фільтрації аудиторської вибірки на основі автоматичного зчитування коду менеджера з особистої книги макросів PERSONAL.XLSB; запропоновано підхід до зовнішньої валідації бізнес-логіки ERP-системи без модифікації її внутрішнього коду, що забезпечує незалежність від версії та конфігурації системи.

Практична значущість результатів полягає в можливості розгортання інструменту контролю якості товарних даних без додаткових фінансових витрат, залучення розробників або зміни конфігурації ERP-системи, а також у можливості адаптації переліку перевірок під специфічні внутрішні бізнес-правила будь-якого підприємства, що використовує Microsoft Dynamics NAV або сумісну реляційну СУБД.

Апробація результатів. Основні положення та результати дослідження пройшли апробацію на:

1. **VII Міжнародної науково-технічної конференції «Сучасний стан та перспективи розвитку IoT»** (Київ, ДУІКТ, 20 квітня 2026 р.). Тези доповіді на тему «МЕТОДОЛОГІЯ АЛГОРИТМІЧНОГО УСУНЕННЯ АНОМАЛІЙ ДАНИХ У КОРПОРАТИВНИХ ERP-СИСТЕМАХ (НА ПРИКЛАДІ MICROSOFT DYNAMICS NAV)» опубліковані у збірнику матеріалів конференції (стор. 274).

2. **VII Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях»** (Київ, ДУІКТ, 23 квітня 2026 р.). Тези доповіді на тему «ВПЛИВ ПЕРСОНАЛІЗАЦІЇ КОНТРОЛЬНИХ ФУНКЦІЙ НА МІНІМІЗАЦІЮ ІНФОРМАЦІЙНОЇ ЕНТРОПІЇ В КОРПОРАТИВНИХ БАЗАХ ДАНИХ» опубліковані у збірнику матеріалів конференції (стор. 441).

1 ОСНОВИ КОНТРОЛЮ ЯКОСТІ ДАНИХ У КОРПОРАТИВНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

1.1 Теоретичні засади управління якістю даних у корпоративних системах

У сучасних умовах цифрової трансформації дані є одним із ключових стратегічних ресурсів організації, що визначає ефективність управлінських рішень, конкурентоспроможність та стабільність бізнес-процесів. У корпоративних інформаційних системах, зокрема ERP, дані виступають основою для планування, обліку, аналізу та контролю діяльності підприємства, а їхня якість безпосередньо впливає на коректність результатів роботи системи [1].

Дані можна визначити як формалізовані представлення фактів, подій або об’єктів, придатні для зберігання, обробки та передачі в інформаційних системах. На відміну від інформації, дані самі по собі не несуть смислового навантаження до моменту їх інтерпретації в певному контексті. У корпоративних системах дані набувають цінності лише тоді, коли вони є структурованими, актуальними та узгодженими між різними бізнес-модулями.

ERP акумулюють значні обсяги різнорідних даних: довідникові, транзакційні, аналітичні, нормативні [2]. Зокрема, товарні дані (номенклатура, одиниці виміру, вагові характеристики, ціни, групи та підгрупи) є критично важливими для коректної роботи ланцюгів постачання, фінансового обліку та звітності. Помилки або неузгодженості в таких даних здатні призводити до системних збоїв, фінансових втрат та викривлення управлінської інформації. Дослідження Panorama Consulting 2025 року підтверджує, що неналежна підготовка даних стабільно входить до трійки найпоширеніших причин невдалого впровадження ERP-систем [3].

Управління даними охоплює сукупність процесів, політик, стандартів та інструментів, спрямованих на забезпечення повного життєвого циклу даних - від створення та зберігання до використання, архівації та видалення [4]. У

корпоративному середовищі управління даними розглядається як міжфункціональна діяльність, що поєднує IT-підрозділи, бізнес-користувачів та аналітиків.

Сучасні підходи до управління даних підкреслюють необхідність переходу від суто технічного адміністрування баз даних до бізнес-орієнтованого управління, у межах якого визначаються відповідальні особи за якість даних, встановлюються правила їх заповнення та контролю, а також забезпечується прозорість змін [5].

Якість даних визначається ступенем придатності даних для використання з конкретною метою. Високоякісні дані повинні відповідати як технічним, так і бізнес-вимогам. У науковій та практичній літературі виділяють низку ключових характеристик якості даних, серед яких найважливішими є повнота, точність, узгодженість, актуальність та цілісність [6].

Цілісність даних передбачає логічну та структурну узгодженість значень між собою, зокрема відповідність довідниковим зв’язкам та бізнес-правилам. Актуальність даних означає їх відповідність поточному стану об’єктів та процесів, що особливо важливо для оперативного аналізу та прийняття рішень у ERP [7].

Управління якістю даних (DQM) – це комплексний процес, що поєднує технології, стандарти та організаційні заходи для забезпечення точності, повноти та актуальності інформаційних активів підприємства. Як продемонстровано на рис. 1.1, модель DQM функціонує в синергії з такими ключовими напрямами, як стратегічне управління даними (Data Governance), профілювання (Data Profiling) та інтеграція клієнтських даних (Customer Data Integration).

Рис. 1.1 Модель DQM [8]

Для корпоративних ERP характерною є проблема обмеженості стандартних механізмів валідації, які зазвичай перевіряють лише форматні або обов’язкові поля. Водночас логічні помилки, пов’язані з порушенням внутрішніх бізнес-правил компанії, часто залишаються непоміченими, що зумовлює необхідність розробки додаткових інструментів контролю якості даних.

Якість даних безпосередньо впливає на ефективність функціонування ERP, оскільки всі аналітичні звіти, фінансові розрахунки та управлінські рішення базуються на даних, що зберігаються в єдиній корпоративній базі. Низька якість даних знижує довіру користувачів до системи, збільшує витрати на виправлення помилок та ускладнює масштабування IT-рішень. За оцінками McKinsey Global Institute, низька якість даних спричиняє зниження продуктивності організацій на 20% та зростання операційних витрат на 30% [9]. У зв'язку з цим управління якістю даних розглядається не як разова технічна задача, а як безперервний процес, що має

бути інтегрований у загальну систему корпоративного управління та підтриманий відповідними інструментами автоматизованого контролю.

1.2 ERP-системи як джерело та сховище корпоративних даних

ERP-системи відіграють ключову роль у забезпеченні цілісного управління діяльністю підприємства. ERP являє собою інтегроване програмне рішення, призначене для автоматизації та координації основних бізнес-процесів організації на основі єдиного інформаційного простору та спільної бази даних. Саме завдяки цьому ERP виступає центральним джерелом і сховищем корпоративних даних.

Основною метою впровадження ERP-систем є підвищення ефективності управління ресурсами підприємства шляхом усунення розрізненості інформації, зменшення дублювання даних та забезпечення прозорості бізнес-процесів. На відміну від локальних або вузькоспеціалізованих інформаційних систем, ERP забезпечує наскрізну інтеграцію процесів, за якої дані вводяться один раз і надалі використовуються всіма функціональними модулями системи.

У сучасному бізнес-середовищі ERP-система розглядається не лише як інструмент автоматизації операційної діяльності, а як стратегічна платформа управління підприємством. Вона забезпечує підтримку оперативного, тактичного та стратегічного рівнів управління за рахунок централізованого доступу до актуальних і узгоджених даних.

ERP-системи є основою для формування фінансової та управлінської звітності, планування ресурсів, контролю витрат, управління запасами та аналізу ефективності бізнес-процесів. Якість і повнота даних, що зберігаються в ERP, безпосередньо впливають на достовірність аналітичних звітів та обґрунтованість управлінських рішень, що особливо важливо в умовах високої конкуренції та динамічних змін ринку у сучасному бізнесі.

Ключовою особливістю ERP є використання централізованої реляційної бази даних, у якій зберігається інформація про всі аспекти діяльності підприємства.

Такий підхід реалізує концепцію single source of truth - єдиного джерела істини, що передбачає наявність одного узгодженого набору даних для всіх користувачів і підрозділів компанії [10].

У структурі даних ERP-систем зазвичай виділяють довідникові, транзакційні та аналітичні дані, як показано на рис. 1.2. Кожен тип даних виконує окрему функціональну роль у забезпеченні бізнес-процесів підприємства.

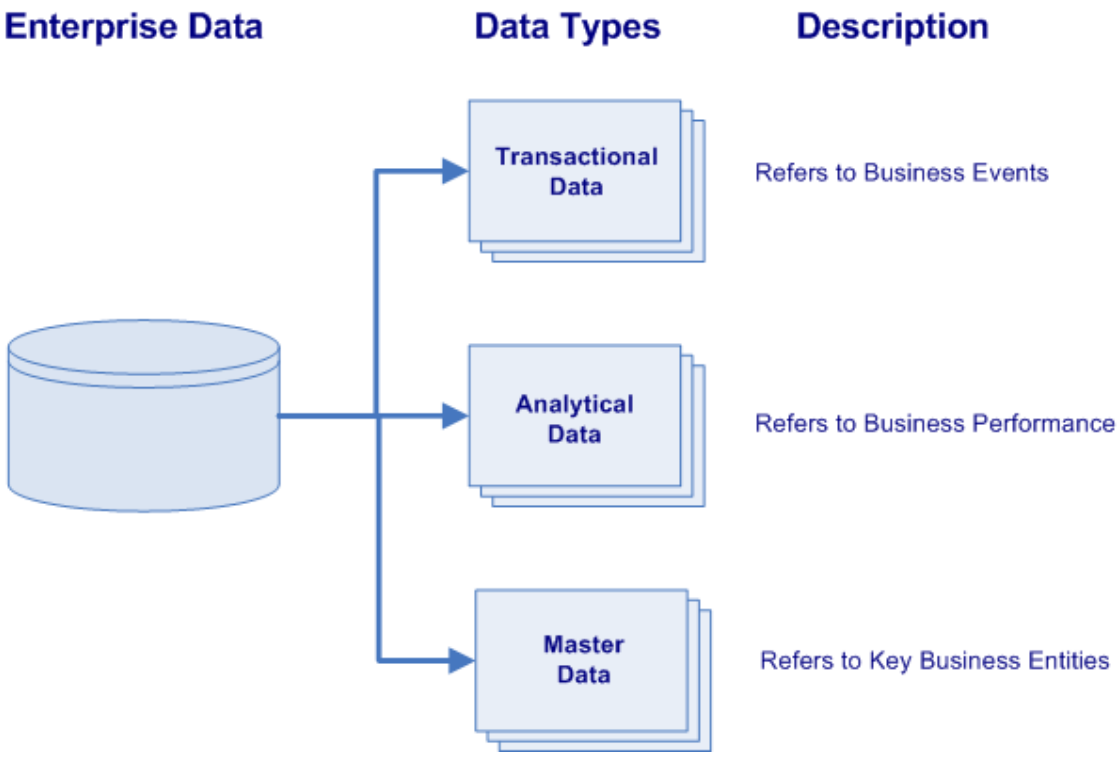


Рис. 1.2 Структура корпоративних даних в ERP-системі [11]

Довідникові дані (master data) являють собою відносно стабільні сутності, що описують ключові об’єкти бізнесу. До них належать дані про товари, клієнтів, постачальників, одиниці виміру, валюту, класифікатори, ієрархії товарних груп, а також інші базові атрибути, які використовуються у різних модулях ERP. Особливістю довідникових даних є те, що вони створюються або змінюються значно рідше, ніж транзакційні, але при цьому мають системоутворююче значення, оскільки на них спираються всі операційні процеси.

Транзакційні дані (transactional data) відображають поточну операційну діяльність підприємства та фіксують конкретні бізнес-події, такі як закупівлі,

продажі, переміщення товарів, виробничі операції, фінансові проводки та інші господарські транзакції. Кожна транзакція зазвичай містить посилання на відповідні довідникові дані - наприклад, артикул товару, контрагента, склад, одиницю виміру або податкову групу. Таким чином, транзакційні дані є динамічними та безпосередньо залежать від коректності та узгодженості довідників.

Помилки у довідникових даних мають кумулятивний ефект: вони автоматично відображаються у транзакціях і можуть призводити до некоректних розрахунків, неправильного обліку запасів або спотворення фінансових показників. Наприклад, невірно задана одиниця виміру або неправильний зв’язок товару з групою може вплинути на облік залишків, калькуляцію собівартості та формування звітності.

Аналітичні (analytical data) дані формуються на основі транзакційних даних шляхом їх узагальнення, групування та обчислення показників ефективності. До таких даних належать звіти, підсумкові таблиці, показники KPI, фінансові зведення та інші аналітичні представлення, що використовуються для прийняття управлінських рішень. Аналітичні дані зазвичай не вводяться вручну, а є результатом обробки транзакційних даних у межах ERP або інтегрованих BI-систем. Важливо зазначити, що аналітичні дані успадковують усі помилки, допущені на попередніх рівнях. Некоректні довідникові або транзакційні дані призводять до викривлення агрегованих показників, що може створювати хибне уявлення про стан бізнесу та негативно впливати на процес ухвалення рішень.

Таким чином, між довідниковими, транзакційними та аналітичними даними існує чіткий ієрархічний та логічний зв’язок: довідникові дані формують основу, на якій створюються транзакції, а транзакційні дані, у свою чергу, є джерелом для аналітики, що показано на рис. 1.3. Порушення якості даних на будь-якому з цих рівнів поширюється на всі наступні, що робить контроль цілісності та актуальності даних у ERP-системах критично важливим завданням.

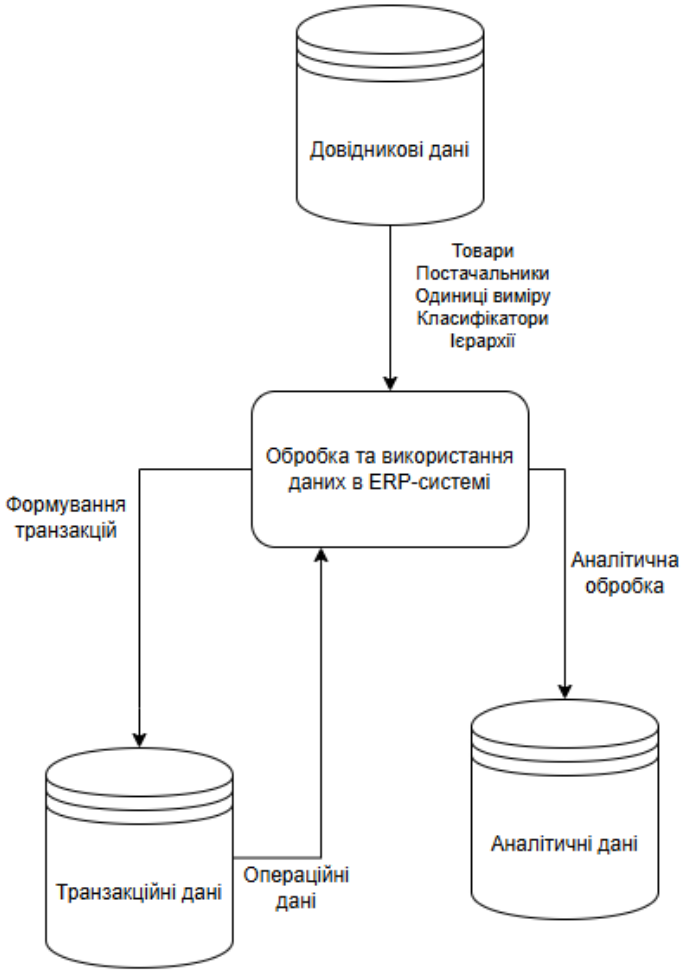


Рис. 1.3 Взаємозв’язок довідникових, транзакційних та аналітичних даних в ERP-системі

ERP-системи мають модульну архітектуру, приклад якої представлено на рис. 1.4, у межах якої кожен модуль відповідає за окрему функціональну область, зокрема фінанси, логістика, склад, продажі, виробництво тощо, але всі модулі працюють із спільною базою даних. Така архітектура забезпечує гнучкість та масштабованість системи, дозволяючи адаптувати її до специфіки бізнесу. Водночас модульна структура ERP створює складну мережу взаємозв’язків між даними. Наприклад, товарні характеристики використовуються одночасно в модулях закупівель, складу, продажів і фінансів. Порушення логіки цих зв’язків або некоректне заповнення атрибутів може мати наслідки для кількох бізнес-процесів одночасно, що ускладнює виявлення та локалізацію помилок.

Рис. 1.4 Модульна архітектура ERP-системи та єдина база даних

Практика експлуатації ERP-систем свідчить, що значна частина корпоративних даних вводиться вручну або імпортується з зовнішніх джерел, зокрема електронних таблиць. Такий підхід часто зумовлений необхідністю масового завантаження даних, адаптації під внутрішні бізнес-процеси або обмеженнями стандартного функціоналу ERP [12].

Незважаючи на наявність вбудованих механізмів перевірки, більшість ERP зосереджуються на контролі формальних обмежень, таких як типи даних, обов’язковість заповнення полів або базові довідникові зв’язки [13]. Логічні перевірки, що враховують специфічні бізнес-правила компанії, часто залишаються поза межами стандартної функціональності системи.

Таким чином, ERP виступає центральним елементом корпоративної інформаційної інфраструктури, поєднуючи функції автоматизації бізнес-процесів, зберігання та поширення корпоративних даних. Висока залежність усіх бізнес-процесів від даних ERP зумовлює особливу важливість контролю їх цілісності, актуальності та узгодженості.

У цьому контексті ERP не лише акумулюють дані, а й формують вимоги до впровадження додаткових механізмів управління якістю даних, що дозволяє мінімізувати ризики, пов’язані з помилками, та забезпечити надійність управлінської інформації.

1.3 Огляд ефективних методологій валідації даних

Розвиток корпоративних інформаційних систем зумовив необхідність переходу від локального контролю коректності даних до системного та багатоаспектного управління їх якістю. У цьому контексті валідація даних розглядається як сукупність методів і процедур, спрямованих на перевірку відповідності даних заданим вимогам, правилам і очікуванням користувачів на різних етапах їх життєвого циклу. На рис. 1.5 зображена схема життєвого циклу даних, як видно, валідація є одним із базових механізмів забезпечення якості даних, оскільки саме вона дозволяє виявляти помилки до того, як дані починають використовуватися в операційних, аналітичних або управлінських процесах.

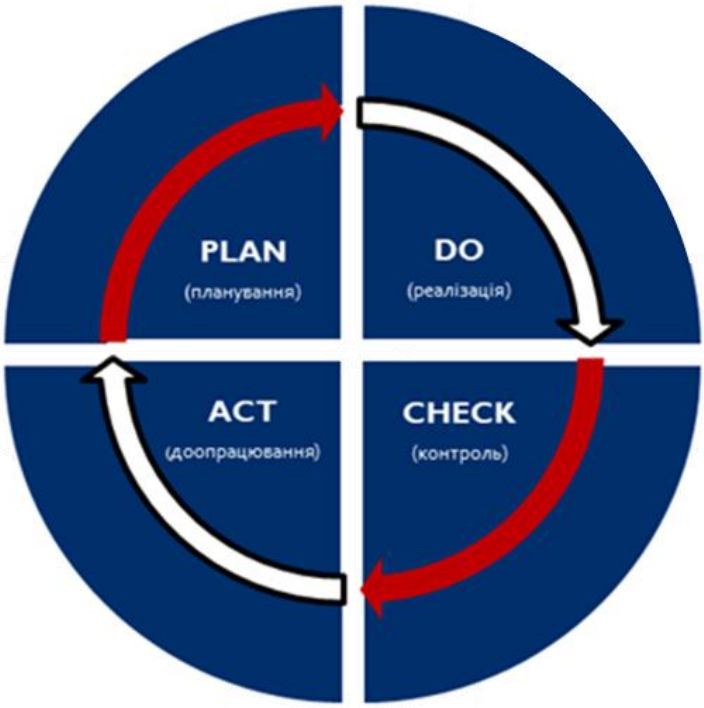


Рис. 1.5 Схема життєвого циклу даних

На практиці підкреслюється, що валідація не зводиться виключно до перевірки формальної коректності значень, а має враховувати семантичний зміст даних і контекст їх використання. Саме тому в сучасних концепціях управління якістю даних валідація розглядається як інтегрований елемент загальної системи DQM, тісно пов’язаний із процесами моніторингу, оцінювання та вдосконалення якості корпоративної інформації.

В ході дослідження, було виділенно кілька методологій:

Методологія профілювання даних (Data Profiling) розглядається в сучасності як фундаментальний процес аналізу інформаційних активів, що базується на застосуванні статистичних та аналітичних методів для оцінки фактичного стану, структури та взаємозв'язків усередині наборів даних. Основним принципом, на якому базується цей метод, є перехід від припущень про якість даних до об'єктивного вимірювання їхніх реальних характеристик шляхом безпосереднього дослідження вмісту бази [14]. Реалізація профілювання передбачає використання спеціалізованих алгоритмів для збору метаданих, що дозволяє визначити типи

даних, частотний розподіл значень, наявність дублікатів та ступінь заповненості полів.

Процес реалізації методології є ітеративним і включає шість ключових фаз, які забезпечують перетворення сирих даних на якісний інформаційний актив. Схему реалізації представлено на рис. 1.6. Початкова фаза виявлення (Discover) передбачає ідентифікацію джерел даних та взаємодію з експертами предметної галузі, після чого слідує етап збору (Collect), де відбувається валідація ролей даних та безпосередня агрегація інформації. Центральною ланкою є фаза аналізу (Analyze), під час якої визначаються стратегії очищення та цільові показники якості. Отримані результати підлягають документуванню (Document) у каталогах даних, що стає основою для активних дій (Act) – впровадження правил якості та безпосереднього очищення [15]. Завершальний етап відстеження (Track) передбачає використання панелей моніторингу для вимірювання прогресу в реальному часі.

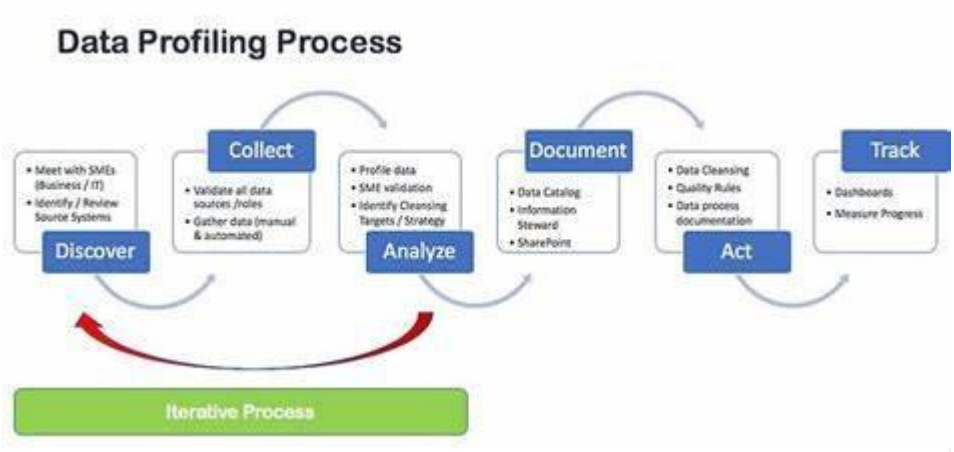


Рис. 1.6 Процес профілювання даних [16]

Функціонально ця методологія вирішує задачу виявлення прихованих аномалій ще до початку аналітичної обробки, забезпечуючи розрахунок метрик повноти, точності та відповідності встановленим форматам. Завдяки профілюванню аналітик отримує можливість локалізувати сегменти бази, де інформація вносилася некоректно або застаріла, що особливо актуально для ERP-систем з інтенсивним ручним введенням. Практичні кейси впровадження

профілювання демонструють значну ефективність цього підходу: зокрема, в межах проєкту NHSBSA було реалізовано понад 700 правил перевірки якості даних та виявлено більше 150 потенційних напрямів покращення інформаційних активів підприємства [17].

Проте методологія має певні обмеження, оскільки вона переважно фокусується на синтаксичному аспекті та структурі, не маючи змоги самостійно оцінювати семантичну правильність без інтеграції складних бізнес-правил. Наприклад, профілювання підтвердить наявність числового значення у полі ціни, але не виявить логічну невідповідність вартості товарів, якщо для цього не задано специфічних предикатів. Таким чином, профілювання є критично важливим діагностичним інструментом, що створює базу для подальшої валідації в межах корпоративних регламентів.

Методологія DAQUA-MASS, що базується на міжнародному стандарті ISO 8000-61, представляє собою процесно-орієнтований підхід, де якість даних розглядається як результат сталого виробничого циклу. На відміну від разових перевірок, цей метод базується на принципі безперервного вдосконалення (Continuous Improvement) та регламентує кожен етап взаємодії з інформаційними активами підприємства. Реалізується дана методологія через впровадження замкненого циклу управління, який інтегрується безпосередньо в операційну діяльність компанії та охоплює не лише технічні аспекти, а й організаційні процедури [18].

Ключова архітектура цього підходу базується на циклічному управлінні, що включає стратегічне планування якості, контроль виконання та постійний моніторинг, як це концептуально відображено рис. 1.7.

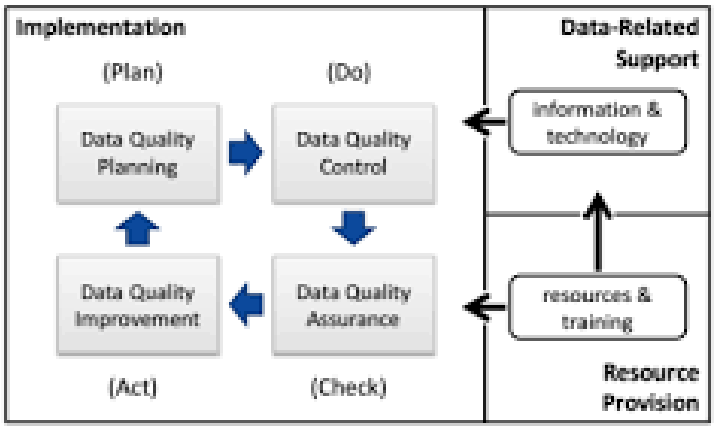


Рис. 1.7 Схема процесної моделі управління даними

Практична реалізація DAQUA-MASS у великих корпоративних системах дозволяє вирішити проблему фрагментарності контролю, оскільки методологія вимагає створення єдиного репозиторію бізнес-правил та метрик якості, доступних для всіх учасників процесу. За даними Gartner, середня кількість інцидентів якості даних на одну організацію зросла з 59 на місяць у 2022 році до 67 у 2023-му, а частка бізнесу, що не довіряє власним даним для прийняття рішень, збільшилась з 55% до 67% [19]. Це свідчить про те, що разові перевірки не дають сталого ефекту, а тому процесно-орієнтований підхід із безперервним моніторингом, який є основою DAQUA-MASS, є стратегічно виправданим для великих корпоративних систем.

Центральним елементом методу є розробка та впровадження контрольних панелей (dashboards), які дозволяють у реальному часі відстежувати рівень точності, повноти та надійності даних, що надходять до ERP-системи. Такий підхід забезпечує стабільність аналітичних звітів та дозволяє менеджменту приймати рішення на основі перевіреної інформації, мінімізуючи ризики, пов'язані з людським фактором.

Основним обмеженням методології DAQUA-MASS є її висока організаційна складність та потреба у значних ресурсах для початкового впровадження. Оскільки метод вимагає повної стандартизації процесів згідно з ISO 8000, компаніям часто важко адаптувати свої існуючі гнучкі бізнес-процеси під суворі вимоги стандарту без втрати оперативної швидкості [20]. Крім того, методологія залежить від

високого рівня зрілості IT-інфраструктури, що може бути бар'єром для підприємств з успадкованими архітектурами ERP, де автоматизація збору метрик є технічно обмеженою. Таким чином, DAQUA-MASS є ідеальним рішенням для довгострокового забезпечення якості в межах холдингів, проте вона потребує доповнення більш гнучкими інструментами для оперативного виправлення помилок на рівні окремих аналітиків.

Валідація структурної цілісності розглядається як критичний механізм забезпечення логічної консистентності реляційних моделей, що гарантує точність взаємозв’язків між сутностями в межах бази даних. В архітектурі великих ERP-систем, таких як Microsoft Dynamics NAV, ця методологія базується на суворому дотриманні правил зовнішніх ключів, де кожен запис у підпорядкованій таблиці повинен мати коректне посилання на існуючий об’єкт у батьківському довіднику [21]. Реалізація цього підходу через зовнішній аналітичний контур дозволяє виявляти специфічні структурні дефекти, які часто ігноруються стандартними інтерфейсами при масовій обробці даних через прикладні розширення.

Ключовим об'єктом аналізу в межах цієї методології є ідентифікація "сирітських записів" (orphaned records) – елементів, чиї посилання на батьківські сутності були розірвані або вказані некоректно під час міграції чи ручного редагування. Як продемонстровано на рис. 1.8, відсутність відповідного ключа в головній таблиці призводить до втрати контексту даних, що робить їх непридатними для подальшої аналітики та порушує ієрархічну побудову звітів.

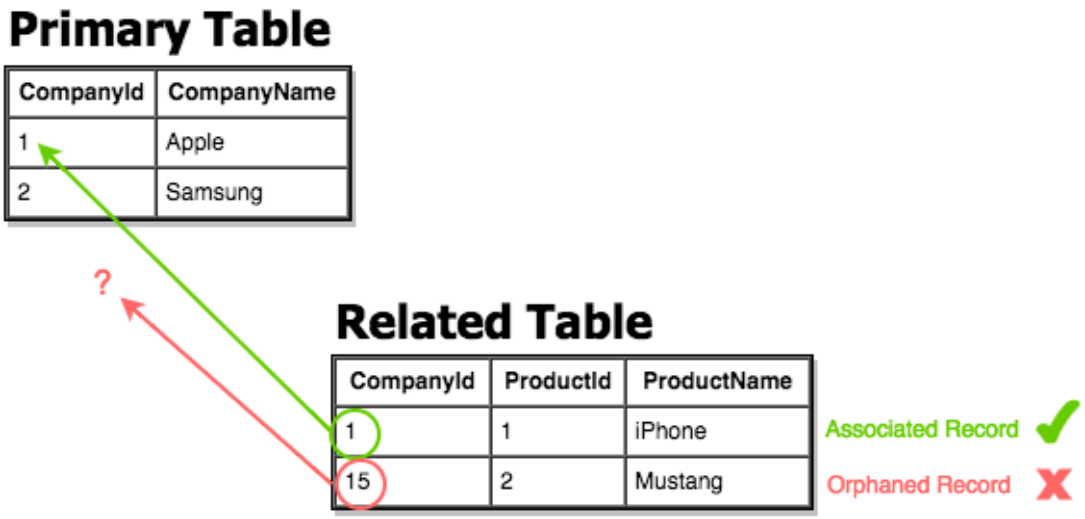


Рис. 1.8 Схематичне представлення порушення посилальної цілісності (Referential Integrity) між сутностями ERP-системи

Практичне застосування структурної валідації дозволяє системно розв'язувати проблему інформаційних розривів, коли товарна картка формально існує в системі, але її атрибути посилаються на неіснуючі або видалені групи, склади чи митні коди. Методологія усуває найбільш критичні структурні дефекти. У практичних кейсах встановлено, що 7% замовлень можуть бути не пов'язані з відповідними записами в інших таблицях, що спричиняє затримки обробки; впровадження перевірок посилальної цілісності дозволяло підвищити ефективність обробки замовлень на 34% [22].

Проте імплементація даної методології стикається з певними технічними викликами. По-перше, будь-яка кастомізація ERP-системи, що супроводжується зміною структури таблиць, потребує негайної перевірки та адаптації SQL-скриптів валідації. По-друге, перевірка цілісності на мільйонах записів у реальному часі може негативно впливати на продуктивність продуктивного сервера, що робить доцільним винесення цих операцій у фоновий режим або в окремий шар даних. Незважаючи на це, структурна валідація залишається незамінним фундаментом, адже без підтвердження цілісності зв'язків будь-який подальший аналіз бізнес-логіки втрачає сенс.

Проведений огляд методологій валідації даних демонструє, що жодна з розглянутих методологій не є самодостатньою – кожна з них вирішує окремий клас проблем якості даних і потребує комплементарного застосування.

Загальний контекст проблеми підкреслює масштаб її фінансових наслідків: за даними звіту IBM Institute for Business Value 2025 року, 43% операційних директорів називають якість даних своїм головним пріоритетом, а понад чверть організацій оцінюють свої річні втрати від неякісних даних у понад 5 млн доларів [23]. За оцінками Gartner, середньорічні втрати від низької якості даних складають 12,9 млн доларів на організацію [24].

Таким чином, розглянуті методології утворюють логічну ієрархію: профілювання діагностує фактичний стан даних, DAQUA-MASS забезпечує системний і безперервний контроль якості, а валідація посилальної цілісності гарантує коректність структурних зв'язків, без яких будь-який подальший аналіз втрачає достовірність. Саме ця тришарова архітектура контролю стала методологічною основою для розробки інструменту валідації товарних даних у корпоративній ERP-системі Microsoft Dynamics NAV. Розроблений інструмент поєднує принципи профілювання – через автоматичне виявлення незаповнених полів та аномалій у структурі даних, – перевірку посилальної цілісності між пов'язаними довідниками (групи, підгрупи, склади, митні коди) та системний моніторинг відповідно до внутрішніх бізнес-правил компанії, які виходять за межі стандартних обмежень холдингової конфігурації NAV.

2 АНАЛІЗ ЯКОСТІ ТОВАРНИХ ДАНИХ ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ДЛЯ ЇХ ВАЛІДАЦІЇ

2.1 Характеристика бази дослідження та джерел даних Dynamics Nav

Базою практичного дослідження став аналітичний відділ дистриб'юторської компанії, що спеціалізується на оптовій торгівлі автозапчастинами та є частиною міжнародного холдингу. Варто зазначити, що попри назву, функціональні обов'язки працівників відділу переважно відповідають ролі продукт-менеджерів з глибоким фокусом на управлінні життєвим циклом товару в інформаційній системі. Відділ налічує 7 спеціалістів, кожен з яких відповідає за окремий товарний проєкт – загалом ведеться 9 товарних напрямів, один менеджер може бути відповідальний за кілька проєктів. Робота відділу є критично важливою для бізнесу, оскільки від точності та повноти товарних даних залежить коректність замовлень, митне оформлення, логістика та кінцева реалізація продукту.

Функціональні обов'язки продукт-менеджерів охоплюють три основні напрями діяльності. Перший – управління асортиментом, куди входить створення та актуалізація карток товарів, заповнення технічних критеріїв, прив'язка товарів до автомобілів та ведення крос-посилань (OE-номери, коди постачальника). Другий – ціноутворення, що означає оновлення цін у системі, моніторинг статистики продажів, коригування маржинальності та встановлення цін на нові надходження. Третій – комунікація із постачальниками: створення замовлень, підтвердження наявності і характеристик товарів, узгодження термінів відправки, підтвердження замовлення та взаємодія з логістичними підрозділами, для відслідковування руху товару.

Основним інструментом роботи є корпоративна ERP-система Microsoft Dynamics NAV. Всі операції введення даних реалізуються через заповнення спеціалізованих Excel-шаблонів з подальшим імпортом до системи. Використовуються окремі шаблони для різних типів даних: основна картка товару

(артикул, опис, одиниці виміру, фізичні параметри, проект, відповідальний менеджер, опис митний, код УКТЗЕД), технічні критерії, крос-посилання. Оскільки система є уніфікованою для всіх підприємств холдингу, налаштування валідаційних правил обмежені загальнокорпоративними регламентами і не можуть бути розширені відповідно до специфічних внутрішніх правил окремого підприємства.

Важливою організаційною особливістю відділу є персоналізація відповідальності за товарні дані. Кожен продукт-менеджер має унікальний код, який зберігається в особистій книзі макросів Microsoft Excel – файлі PERSONAL.XLSB, що автоматично завантажується при запуску Excel на комп’ютері кожного співробітника. Цей код відповідає ідентифікатору менеджера в базі даних NAV, за яким у системі закріплені артикули відповідного товарного проєкту. Така архітектура дозволяє розробленому інструменту валідації автоматично зчитувати код поточного користувача і формувати персоналізовану вибірку даних – кожен аналітик бачить виключно помилки у власних артикулах, без необхідності ручного фільтрування чужих проєктів. Це рішення суттєво підвищує зручність роботи в умовах багатопроєктного відділу та мінімізує ризик випадкового редагування даних суміжних проєктів.

Джерелами даних для розробленої системи валідації є реляційні таблиці бази даних NAV, доступ до яких здійснюється через два SQL-сервери з різними зонами відповідальності. Сервер sql_1 є основним сховищем товарної інформації та містить таблиці карток товарів, цінових записів, довідників, технічних критеріїв, крос-посилань, залишків на складах та стандартизованих описів. Сервер sql_2 має вужчу спеціалізацію і зберігає дані, пов'язані з автомобільним каталогом: таблиці моделей та модифікацій транспортних засобів, а також таблиці прив'язки конкретних артикулів до відповідних автомобілів. Така архітектурна розділеність серверів відображає логічну межу між товарним доменом і каталожним доменом системи, що враховується при побудові SQL-запитів інструменту валідації. Перелік основних таблиць, що використовуються для реалізації перевірок даних та їх опис, наведено в таблиці 2.1.

Табл. 2.1

Основні таблиці бази даних Dynamics NAV, що є джерелами валідації

Таблиця	Опис
Items	Основна картка товару: артикул, опис, категорія, група, підгрупа, код УКТЗЕД, вага нетто/брутто, об'єм, одиниці виміру
Prices	Цінові записи: ціна, валюта, одиниця виміру, група знижки, дати початку та закінчення
Cross_Item	Крос-посилання: OE-номери, єврокоди, типи та бренди
Criteria_Value	Фактичні значення критеріїв для кожного товару
Art_Pic	Наявність фотографій для артикулів
Balance_InStock	Залишки товарів на складах із зонуванням
Item_Car	Прив'язка товарів до автомобілів
Criterias_Handbook	Довідник допустимих пар «критерій – значення»
Standard_Description	Стандартизовані описи у прив'язці до кодів УКТЗЕД
Group	Довідник категорій та груп товарів
SubGroup	Довідник підгруп із прив'язкою до груп

Наведені таблиці охоплюють усі ключові сутності товарного та каталожного доменів системи і є вичерпною основою для реалізації перевірок бізнес-логіки. Розподіл між двома серверами враховується при побудові SQL-запитів інструменту валідації та визначає логіку маршрутизації запитів у межах розробленої архітектури.

2.2 Аналіз та класифікація найпоширеніших помилок у товарних даних

Аналіз операційної діяльності аналітичного відділу дозволив виявити та систематизувати типові помилки, що виникають у товарних даних ERP-системи. За своєю природою більшість із них є прямим наслідком структурних особливостей процесу введення даних: ручного заповнення шаблонів, відсутності вбудованих перевірок специфічної бізнес-логіки підприємства та складності внутрішніх довідникових ієрархій. Усі виявлені типи помилок класифіковано за трьома

групами відповідно до їх природи та механізму виникнення, що відображено на рис. 2.1.

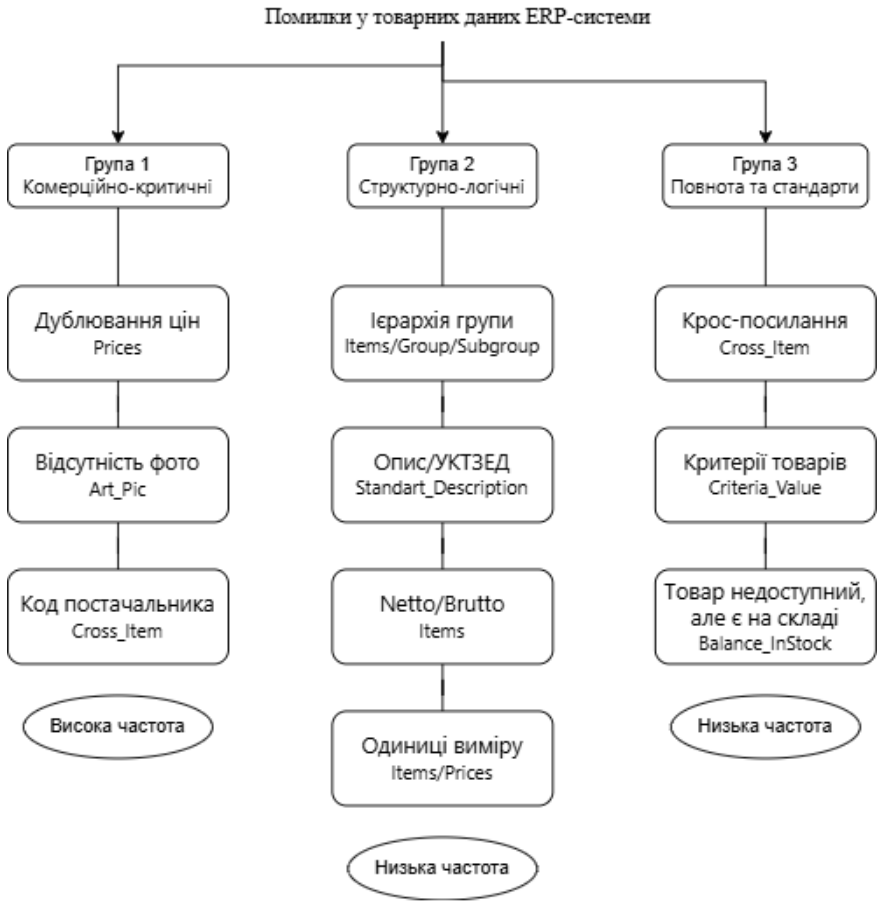


Рис. 2.1 Класифікація типових помилок у товарних даних ERP-системи

Група 1. Комерційно-критичні помилки є найбільш частотними та такими, що безпосередньо впливають на можливість продажу товару й операційну діяльність компанії.

Найпоширенішою є проблема дублювання активних цін. Її механізм виникнення закладений у логіці самої ERP-системи: Dynamics NAV закриває попередній ціновий запис і створює новий лише за умови абсолютної ідентичності всіх параметрів рядка, крім ціни та дати. Якщо продукт-менеджер при оновленні допускає будь-яке відхилення – наприклад, вказує іншу одиницю виміру – система сприймає це як новий запис, залишаючи попередній активним. У результаті в картці товару одночасно існують дві активні ціни, що унеможливлює коректне формування замовлень. Особливу небезпеку становить те, що ця помилка є

практично невидимою в стандартному інтерфейсі NAV – менеджер не отримує жодного попередження.

Другою за частотою є відсутність зображення товару в системі. Завантаження фотографії є окремим процесом поза основним шаблоном імпорту, тому артикул може бути повністю коректно прописаний у всіх полях, але залишитись без медіаконтенту. Це безпосередньо знижує конверсію на B2B-платформі, оскільки клієнт не може візуально верифікувати деталь перед покупкою.

Третьою проблемою цієї групи є відсутність коду постачальника – крос-посилання типу «Vendor», під яким постачальник ідентифікує товар у своїй системі. Без нього автоматизоване формування замовлень є неможливим, що призводить до ручного узгодження та затримок у ланцюжку постачання.

Група 2. Помилки структурної та логічної цілісності виникають значно рідше, як правило, кілька випадків на квартал, проте є потенційно небезпечними для аналітики та митного оформлення.

Порушення ієрархії «Категорія – Група – Підгрупа» безпосередньо впливає на автоматичне визначення коду УКТЗЕД. Допустимі комбінації цих рівнів регламентовані внутрішніми правилами підприємства, однак не закріплені в ядрі уніфікованої холдингової системи, тому контроль їх коректності повністю покладається на уважність аналітика. Пов'язаною проблемою є невідповідність між текстовим описом товару та його кодом УКТЗЕД – будь-яке відхилення від затверджених комбінацій є потенційним ризиком при митному контролі.

Сюди ж належать логічні суперечності між фізичними параметрами товару: ситуації, коли маса нетто перевищує масу брутто або об'єм є від'ємним. Система не перевіряє співвідношення між цими значеннями, тому такі помилки безперешкодно потрапляють до бази через некоректне копіювання значень у шаблоні. Окремою проблемою є розбіжність одиниць виміру між карткою товару та ціновим записом – прямий наслідок того, що ці два шаблони заповнюються роздільно, без автоматичної синхронізації.

Група 3. Помилки повноти та відповідності стандартам не блокують операційну діяльність безпосередньо, проте знижують загальну якість

інформаційних активів. Невалідні крос-посилання спотворюють дані про аналоги та замінники товарів. Некоректні значення критеріїв – наприклад, вільний текст замість стандартизованого значення з довідника – унеможливлюють точну фільтрацію на B2B-платформі та в аналітичних звітах. Наявність заблокованих товарів із ненульовими складськими залишками спотворює звітність і ускладнює інвентаризацію.

Загалом аналіз виявлених помилок демонструє чітку закономірність: найбільш частотні проблеми є наслідком роз'єднаності окремих етапів введення даних, тоді як помилки другої та третьої груп мають переважно структурний характер і виникають через неможливість закріпити специфічну бізнес-логіку підприємства в уніфікованій холдинговій ERP-системі. Саме ця двоїста природа проблеми зумовлює необхідність зовнішнього інструменту валідації, здатного охопити обидва виміри контролю якості даних.

2.3 Огляд популярних інструментів контролю якості даних

У ході дослідження проведено порівняльний аналіз технічних рішень, що застосовуються для забезпечення якості даних у корпоративних інформаційних системах. Критеріями оцінки обрано: наявність нативної інтеграції з екосистемою Microsoft SQL Server та Dynamics NAV; глибину та гнучкість перевірок бізнес-логіки, включно з перехресними перевірками між таблицями; вплив на продуктивність ERP-системи під час транзакцій; а також доступність інструменту для аналітика без залучення IT-підрозділу та можливість інтеграції з робочим середовищем Microsoft Excel. Кожен із розглянутих інструментів проілюстровано структурною схемою, що відображає типовий маршрут даних у межах відповідного рішення.

Вбудована валідація Dynamics NAV (тригер OnValidate):

Стандартний механізм валідації в Dynamics NAV реалізований через систему тригерів OnValidate. Як показано на рисунку 2.2, маршрут перевірки є лінійним і

атомарним: після введення користувачем значення тригер спрацьовує безпосередньо на рівні поля, перевіряє відповідність типу даних та властивостей поля, і або підтверджує запис, або повертає помилку – жодного проміжного шару між введенням і реакцією системи немає [25].

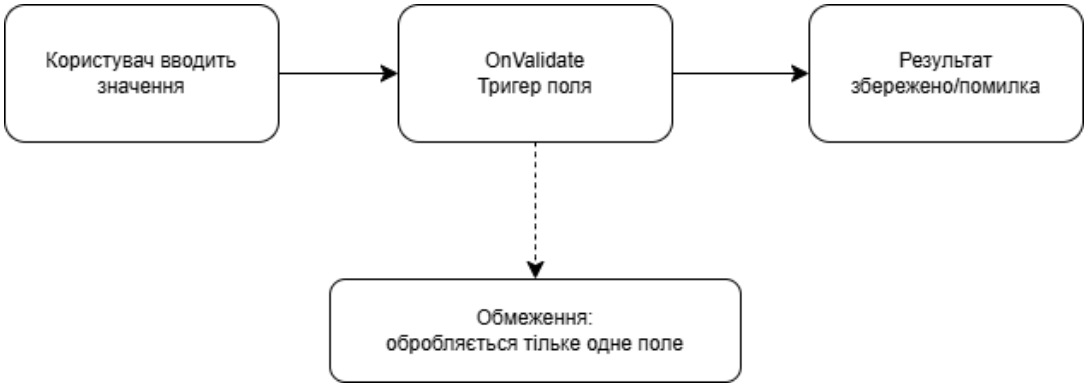


Рис. 2.2 Схема роботи тригера OnValidate в Dynamics Nav: лінійний маршрут атомарної перевірки

Архітектура тригерної моделі накладає суттєві обмеження на глибину валідації. Система не здатна виявляти семантичні суперечності між кількома полями одночасно: вона прийме значення Net Weight = 20 та Gross Weight = 10, оскільки обидва є коректними числами окремо. Крім того, реалізація складних перехресних перевірок між таблицями через тригери призводить до геометричного зростання часу запису, що є неприйнятним для системи з високою інтенсивністю транзакцій. Також, метод Validate не лише присвоює значення, а й запускає всю логіку валідації, визначену для поля в таблиці, а отже будь-яка зміна цієї логіки потребує втручання в C/AL-код, повноцінного тестування та оновлення системи – що унеможливлює оперативну адаптацію до нових бізнес-правил без залучення розробника.

Microsoft SQL Server Data Quality Services (DQS)

SQL Server Data Quality Services (DQS) – це рішення з управління якістю даних на основі бази знань (knowledge-driven), що дозволяє виконувати виправлення, збагачення, стандартизацію та дедублікацію даних. З точки зору інтеграції з екосистемою Microsoft, DQS є логічним вибором, він встановлюється

разом із SQL Server та взаємодіє з SQL Server Integration Services (SSIS) [26]. На відміну від OnValidate, DQS працює не на рівні окремого поля, а над цілими наборами даних. Архітектура DQS, яка представлена на рис. 2.3, реалізована через три каталоги SQL Server: DQS_MAIN, DQS_PROJECTS і DQS_STAGING_DATA, де остання є проміжною базою даних для копіювання вихідних даних перед виконанням операцій якості.

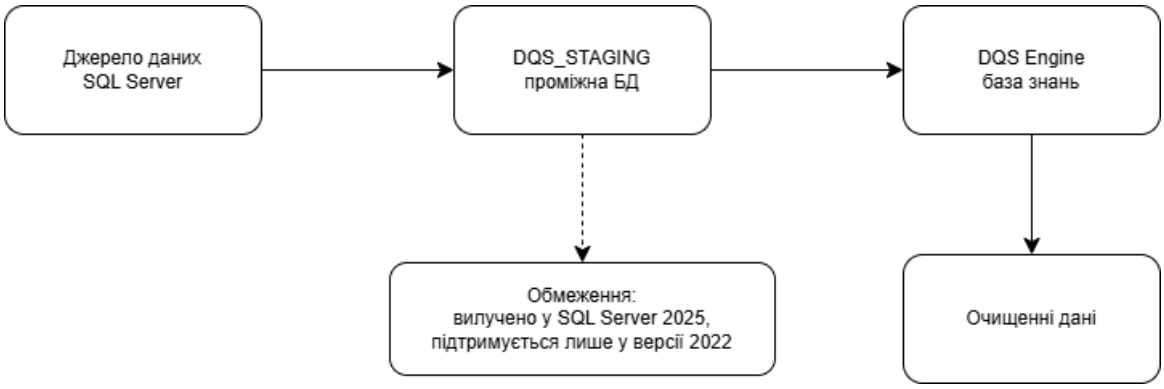


Рис. 2.3 Архітектура DQS: дані проходять через проміжну базу DQS_STAGING перед будь-якою перевіркою

Ключовим стратегічним обмеженням є питання довгострокової підтримки. DQS повністю виключено зі складу SQL Server 2025 (версія 17.x); підтримка зберігається лише у версіях SQL Server 2022 та більш ранніх. Вилучення відбулося без попереджень про застарілість і без режиму підтримки сумісності – функція просто зникла з продукту. Це свідчить про стратегічне згортання напряму з боку Microsoft та робить інвестиції в DQS безперспективними для будь-якої організації, що планує оновлення серверної платформи [27].

Фреймворк Great Expectations

Great Expectations (GX) – відкритий Python-фреймворк для тестування та документування якості даних, побудований на концепції тверджень (assertions): аналітик описує, якими мають бути дані, а фреймворк автоматично перевіряє відповідність і генерує HTML-звіт [28]. Фреймворк підтримує широкий спектр джерел даних і форматів, включно з популярними базами даних на кшталт PostgreSQL, MySQL та SQL Server, а також файловими форматами CSV, Parquet і

JSON. Як показує рисунок 2.4, конвеєр GX є потужним, але принципово відчуженим від інтерфейсу кінцевого користувача: між вхідними даними і результатом знаходяться Python-середовище, GX Context і Expectation Suite.

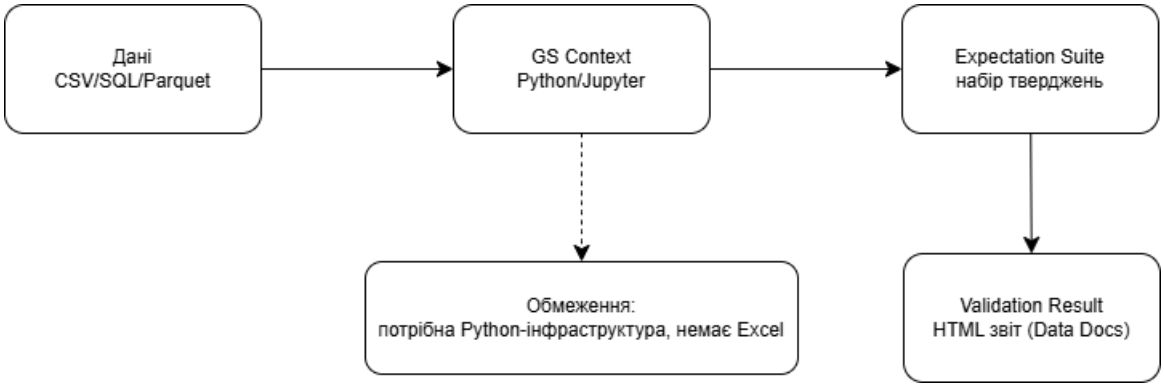


Рис. 2.4 Конвеєр валідації Great Expectations: потужність обмежена Python-інфраструктурою

Незважаючи на технічну потужність, застосування Great Expectations в середовищі аналітичного відділу стикається з кількома принциповими бар'єрами. Фреймворк орієнтований на використання з Python і Jupyter Notebook як основним робочим середовищем, що є надмірно інертним для продукт-менеджера, чиїм основним інструментом є Microsoft Excel: замість миттєвої перевірки безпосередньо в інтерфейсі таблиці, користувач змушений звертатися до командного рядка або спеціалізованих веб-інтерфейсів. Також варто зазначити, що повноцінне розгортання GX для команди вимагає підготовки окремої інфраструктури: налаштування оточень для розробки та виробничого середовища, вибору інструменту оркестрації та забезпечення доступу до обчислювальних ресурсів. Фреймворк не має нативної підтримки взаємодії з персональною книгою макросів Excel (PERSONAL.XLSB), що унеможливлює автоматичне розпізнавання зони відповідальності конкретного менеджера під час аудиту.

Платформа Talend Data Fabric (Qlik)

Talend Data Fabric – це low-code платформа, що об'єднує інтеграцію даних, управління їх якістю в єдиному рішенні, яке підтримує наскрізне управління даними у хмарних, локальних та гібридних середовищах. З травня 2023 року

платформа увійшла до складу Qlik після завершення поглинання, об'єднавши можливості аналітики Qlik із інструментами інтеграції та якості даних Talend; протягом семи років поспіль Gartner позиціонував Talend як лідера у Magic Quadrant для інструментів інтеграції даних та п'ять років – у Magic Quadrant для рішень з якості даних.

Попри широкі функціональні можливості, Talend має принципові обмеження для розглянутого кейсу. Рисунок 2.5 ілюструє типову архітектуру обробки: дані з різноманітних джерел проходять через ETL-конвеєр та Staging Area, потрапляють до ML-рушія якості, і лише після цього формується підсумковий Trust Score™.

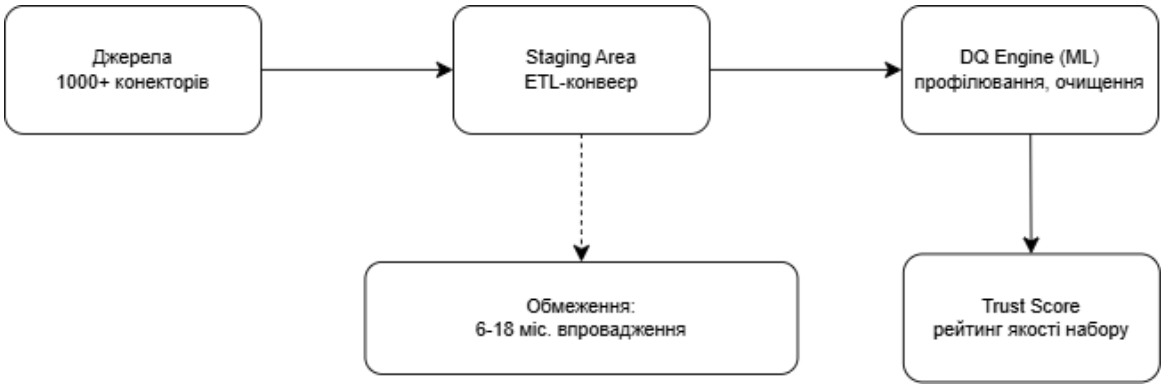


Рис. 2.5 ETL-конвеєр Talend Data Fabric: від джерел через Staging Area до Trust Score™

Процес валідації вимагає попереднього завантаження даних у проміжне сховище, що позбавляє аналітика оперативності: перевірити правильність заповнення конкретного поля неможливо без проходження повного ETL-циклу в сторонній системі [29]. Крім того, комплексна платформа вимагає залучення команди розробників, що робить його економічно недоцільним для вирішення операційних задач окремого аналітичного відділу.

Результати оцінки всіх розглянутих рішень за ключовими критеріями систематизовано в таблиці 2.2. Як демонструє порівняльний аналіз, жодне з розглянутих рішень не відповідає комплексу вимог аналітичного відділу дистриб'юторської компанії. Вбудована валідація NAV є технічно обмеженою та не підтримує перевірку семантичних суперечностей між полями. DQS стратегічно

вилучено з нового покоління SQL Server без шляху міграції. Great Expectations та Talend забезпечують глибокі перевірки, однак вимагають технічної інфраструктури та компетенцій, недоступних на рівні окремого аналітика, а також позбавлені нативної інтеграції з Excel-середовищем.

Табл. 2.2

Порівняльний аналіз інструментів контролю якості даних

Критерій	Вбудована валідація Nav	MS SQL DQS	Great Expectations	Talend Data Fabric
Інтеграція з NAV / SQL Server	Нативна	Нативна	Відсутня	Через коннектор
Глибина перевірок	Атомарна(поле)	Середня	Висока	Висока
Перехресні перевірки між таблицями	Обмежено	Частково	Так	Так
Перевірки бізнес-логіки	Ні	Частково	Так	Так
Вплив на продуктивність ERP	Високий	Середній	Відсутній	Відсутній
Гнучкість бізнес-правил (без розробника)	Низька	Середня	Висока	Середня
Доступність для аналітика без IT-підтримки	Так	Ні	Ні	Ні
Нативна інтеграція з робочим середовищем Excel	Відсутня	Відсутня	Відсутня	Відсутня
Персоналізація за зоною відповідальності	Відсутня	Відсутня	Відсутня	Відсутня
Перспективність (стратегічна підтримка)	Обмежена (legacy)	Знята (SQL 2025)	Активна	Активна
Вартість впровадження	Включена в NAV	Включена в SQL Server	Безкоштовно	Близько 45000$/рік
Складність впровадження	Низька	Висока	Висока	Висока

Результати порівняльного аналізу підтверджують, що жодне з розглянутих рішень не відповідає комплексу вимог аналітичного відділу дистриб'юторської компанії. Вбудована валідація NAV є технічно обмеженою і не підтримує перевірку семантичних суперечностей між полями. DQS стратегічно вилучено з нового покоління SQL Server без шляху міграції. Great Expectations та Talend забезпечують глибокі перевірки, однак вимагають технічної інфраструктури та компетенцій, недоступних на рівні окремого аналітика, і не мають нативної інтеграції з Excel-середовищем. Виявлена прогалина обґрунтовує необхідність розробки власного

інструменту, що поєднував би глибину перехресних перевірок з операційною доступністю для аналітика без залучення IT-підрозділу.

3 РОЗРОБКА ІНСТРУМЕНТУ КОНТРОЛЮ ЦІЛІСНОСТІ ТА АКТУАЛЬНОСТІ ДАНИХ У КОРПОРАТИВНИХ ERP-СИСТЕМАХ

3.1 Концепція, принципи роботи та архітектура розробленої системи

За результатами аналізу, проведеного в попередніх розділах, було встановлено, що жодне з наявних рішень не відповідає комплексу вимог аналітичного відділу: забезпечити глибокі перехресні перевірки бізнес-логіки, зберегти пряму інтеграцію з робочим середовищем Excel та надати кожному аналітику персоналізований результат без залучення IT-підрозділу. Відповідно до цих вимог було сформульовано концепцію розробленої системи: інструмент самоконтролю для валідації товарних даних, реалізований як макросна Excel-книга (.xlsb), що безпосередньо підключається до бази даних Dynamics NAV через ADO-з'єднання та виконує серію параметризованих SQL-запитів, формуючи персоналізований звіт про помилки для кожного менеджера.

В основу концепції покладено три ключові принципи. Перший – принцип мінімального втручання в ERP-систему: усі перевірки виконуються поза межами Dynamics NAV, у зовнішньому аналітичному контурі, що унеможливлює будь-який вплив на продуктивність продуктивного сервера та не потребує модифікації C/AL-коду системи. Другий – принцип персоналізації відповідальності: інструмент автоматично зчитує код поточного менеджера з особистої книги макросів PERSONAL.XLSB і використовує його як параметр фільтрації в кожному SQL-запиті, завдяки чому кожен аналітик бачить виключно дані своїх товарних проєктів. Третій – принцип операційної доступності: запуск будь-якої перевірки зводиться до одного кліку по кнопці безпосередньо в знайомому середовищі Excel, що не вимагає від користувача жодних технічних компетенцій у галузі баз даних чи програмування.

Багаторівнева архітектура системи. Розроблена система побудована за трирівневою архітектурою, що наочно представлена на рис. 3.1. Виділення трьох

чітко розмежованих шарів – презентаційного, бізнес-логіки та даних – забезпечує модульність, незалежність інтерфейсу від логіки обробки та можливість розширення системи без переписування наявного коду.

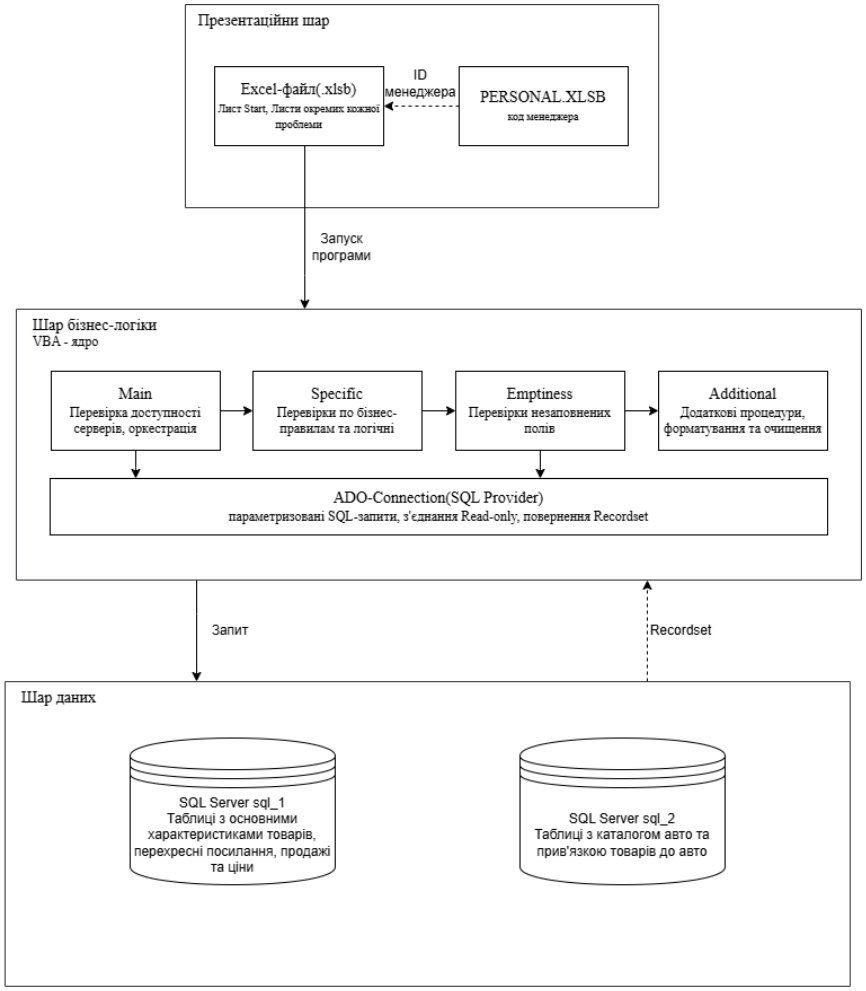


Рис. 3.1 Багаторівнева архітектура системи контролю якості товарних даних

Презентаційний шар реалізований у вигляді Excel-файлу формату .xlsb, що містить стартовий лист Start та набір листів, кожен з яких відповідає окремому типу перевірки. Стартовий лист виконує роль центральної панелі керування: на ньому розміщено кнопку «Update All», що ініціює повну перевірку всіх аркушів, а також відображається загальна статистика виявлених помилок. На кожному з тематичних листів розміщено кнопку «Update Sheet» для запуску окремої перевірки, а де це необхідно – додаткову кнопку для вивантаження відповідного довідника. До презентаційного шару належить і файл PERSONAL.XLSB – особиста книга

макросів, що автоматично завантажується у кожного співробітника при запуску Excel. Саме з неї зчитується код менеджера, який передається як параметр у всі SQL-запити.

Шар бізнес-логіки є VBA-ядром системи і складається з чотирьох функціональних модулів: Main, Specific, Emptiness та Additional. Модуль Main відповідає за оркестрацію процесу: він перевіряє доступність серверів, ініціює цикл по листах при повній перевірці та диспетчеризує виклики до відповідних процедур конкретних перевірок. Модуль Specific містить процедури для перевірок за бізнес-правилами та логічними обмеженнями – дублювання цін, порушення ієрархії груп-підгруп, невідповідність фізичних параметрів тощо. Модуль Emptiness реалізує окремий клас перевірок – контроль незаповненості обов'язкових полів, де для кожного поля реалізована індивідуальна процедура та зведений SQL-запит підрахунку. Модуль Additional містить допоміжні процедури, такі як очищення листів перед оновленням, форматування вивантажених даних, а також функцію Init_Manager(), що зчитує код поточного менеджера з PERSONAL.XLSB. Взаємодія з базою даних у межах шару бізнес-логіки здійснюється через ADO-з'єднання (SQL Provider). Усі запити є параметризованими, з'єднання відкривається у режимі Read-only, а результат повертається у вигляді об'єкта Recordset, який надалі вивантажується на відповідний лист.

Шар даних представлений двома SQL-серверами Dynamics NAV з розмежованими зонами відповідальності. Сервер sql_1 є основним сховищем товарної інформації та містить таблиці карток товарів, цінових записів, крос-посилань, залишків та довідників. Сервер sql_2 зберігає дані автомобільного каталогу – таблиці моделей транспортних засобів та прив'язки артикулів до автомобілів. Така архітектурна розділеність відображає логічну межу між товарним і каталожним доменами системи та враховується при побудові SQL-запитів.

Загальний принцип роботи системи. Логіка взаємодії з системою на рівні користувача визначається двома сценаріями запуску, що показані на рис. 3.2. Перший сценарій – натискання кнопки «Update All» на стартовому листі, він

ініціює повну перевірку всіх аркушів. Другий сценарій – натискання кнопки «Update Sheet» на будь-якому тематичному листі, він відповідає за запуск перевірки лише для цього конкретного аркуша. В обох випадках результат виводиться безпосередньо на відповідний лист або листи книги.

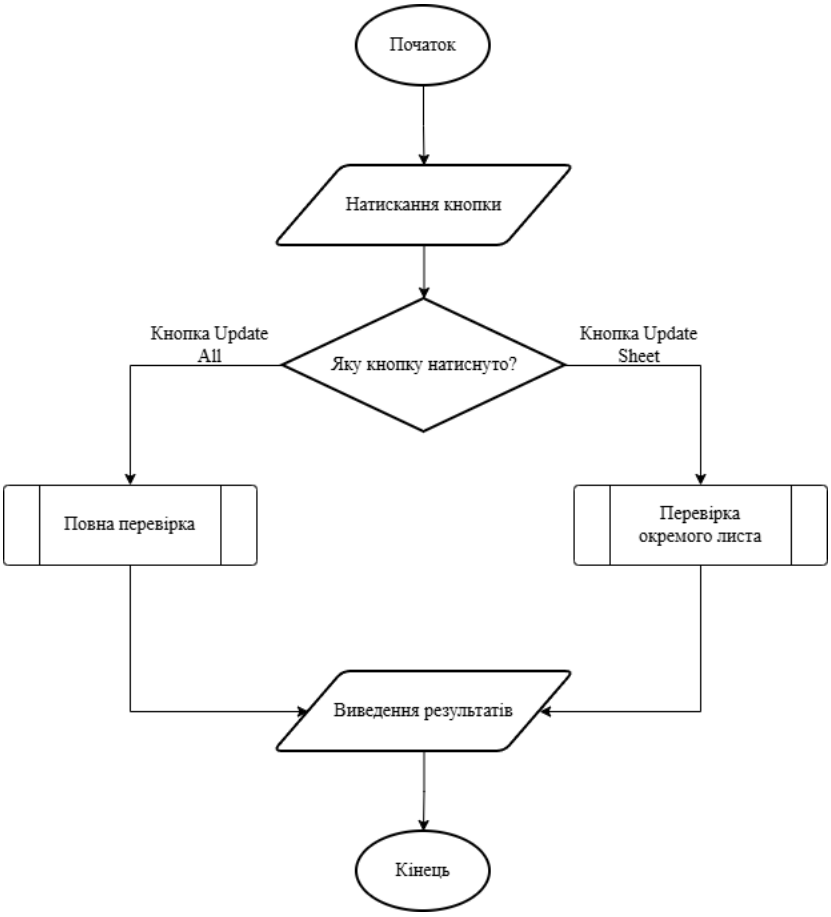


Рис. 3.2 Загальна блок-схема взаємодії користувача з системою

Алгоритм повної перевірки. Детальний алгоритм виконання повної перевірки представлений на рис. 3.3. Після натискання кнопки «Update All» модуль Main першочергово викликає функцію перевірки доступності серверів – це критичний захисний механізм, що запобігає запуску тривалого циклу перевірок за умов відсутності з'єднання з базою. У разі недоступності хоча б одного з серверів користувач отримує повідомлення про помилку і процедура завершується. Якщо обидва сервери доступні, система отримує загальну кількість аркушів та ініціалізує цикл перебору. На кожній ітерації вибирається поточний аркуш, запускається відповідна процедура валідації, після чого цикл переходить до наступного аркуша.

По завершенні всіх ітерацій формується підсумковий результат і виводиться повідомлення з інформацією про готовність та загальний час виконання.

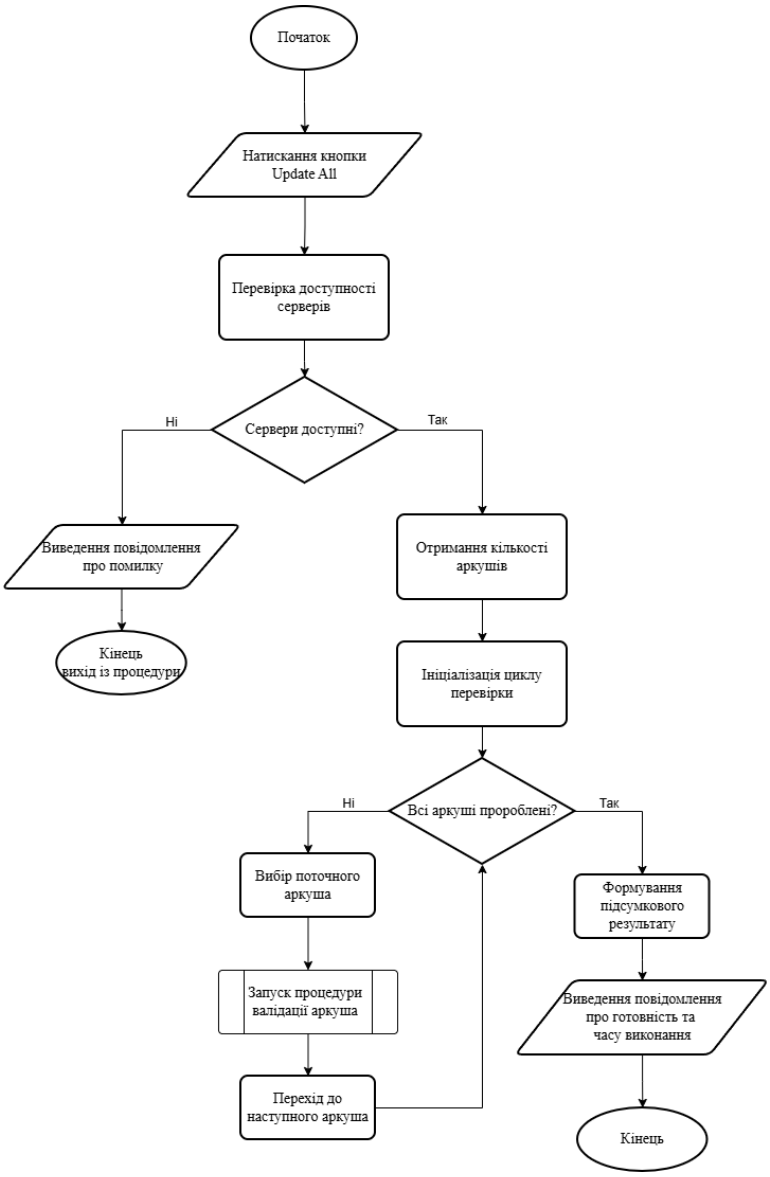


Рис. 3.3 Блок-схема алгоритму повної перевірки

Алгоритм перевірки окремого аркуша. Алгоритм виконання окремої перевірки представлений на рис. 3.4 і є спільним для всіх тематичних листів. Після натискання кнопки «Update Sheet» система також розпочинає з перевірки доступності серверів. За умови успішного з'єднання система отримує назву поточного аркуша та передає її до процедур, яка реалізує явну диспетчеризацію через конструкцію Select Case: кожному можливому імені аркуша відповідає виклик конкретної процедури перевірки з модуля Specific або Emptiness. Після

виконання SQL-запиту вивантажується результат: якщо помилок не виявлено, викликається процедура обробки чистого результату, а якщо помилки присутні – формується список з артикулами, після чого виконується форматування, і аркуш оновлюється фінальним результатом.

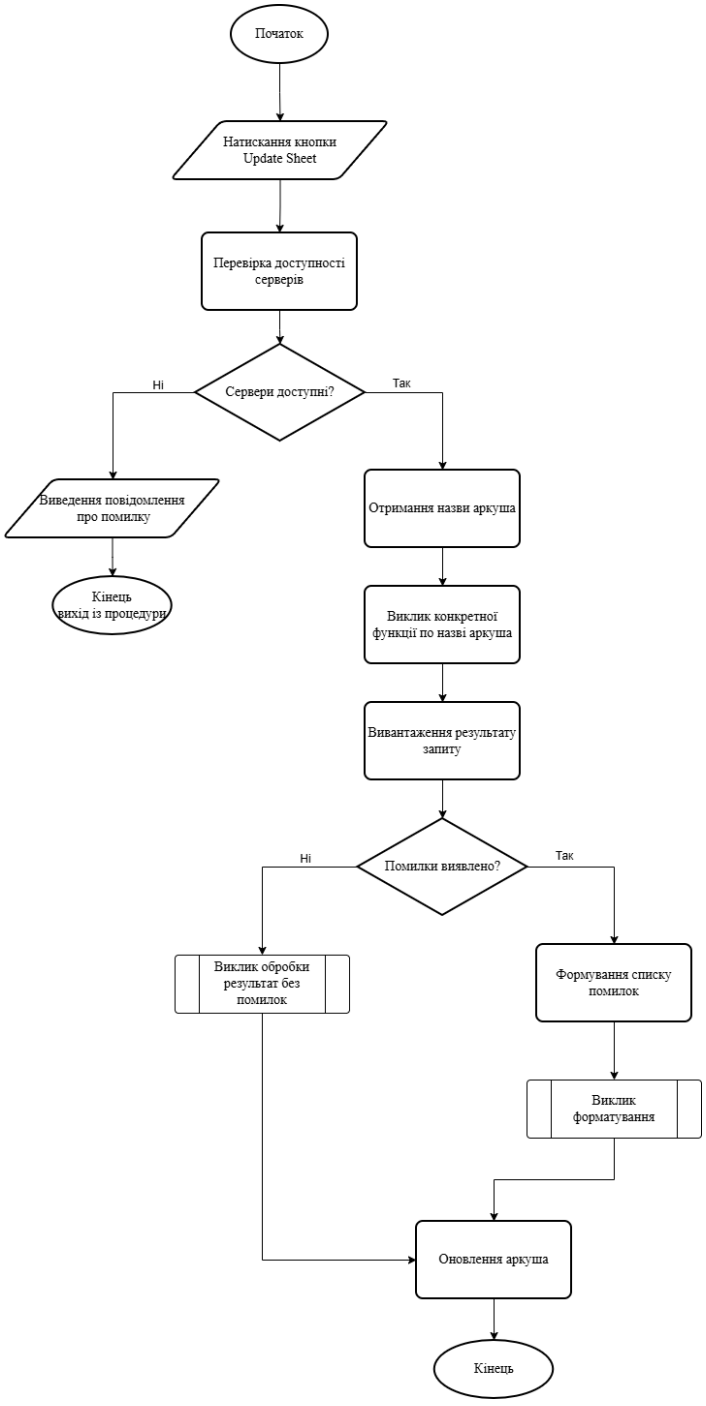


Рис. 3.4 Блок-схема алгоритму перевірки окремого аркуша

Чіткий поділ відповідальності між шарами дозволяє додавати нові типи перевірок без зміни архітектурного ядра, а захисний механізм перевірки доступності серверів гарантує коректне завершення роботи в будь-яких умовах мережевого середовища. Описана архітектура є фундаментом, на якому побудовані всі функціональні модулі системи, детальний розгляд яких наведено у підрозділі 3.2.

3.2 Реалізація функціональних модулів: SQL-запити, вивантаження довідників, обробка помилок

Процес технічної реалізації системи контролю цілісності даних базується на впровадженні ряду функціональних та нефункціональних вимог, що забезпечують стабільність роботи інструменту в умовах корпоративного середовища. До основних функціональних вимог віднесено персоналізацію аудиту, яка дозволяє ідентифікувати аналітика через параметри PERSONAL.XLSB, модульну архітектуру для забезпечення масштабованості коду, децентралізацію обчислень з винесенням логіки на сторону клієнтського VBA-інтерфейсу, а також автоматизацію підготовки робочого простору. Нефункціональні вимоги зосереджені на продуктивності (час обробки масивів не більше 5 хвилин), надійності ADO-підключення до SQL Server та повноцінній локальній інтеграції в середовище MS Excel.

Програмна реалізація системи контролю якості даних виконана мовою VBA (Visual Basic for Applications), вбудованою мовою програмування пакету Microsoft Office. Вибір VBA обумовлений кількома практичними міркуваннями: мова є нативною для Excel, не потребує встановлення додаткового програмного забезпечення, надає повний доступ до об'єктної моделі книги (аркуші, діапазони, клітинки) та підтримує роботу з COM-об'єктами – зокрема ADODB для підключення до зовнішніх баз даних. Розробка виконувалась у вбудованому середовищі Visual Basic Editor (VBE), що відкривається з Excel через комбінацію клавіш Alt+F11. Код організований у чотири окремих модулі: Main, Specific,

Emptiness та Additional, кожен з яких виконує строго визначену функцію відповідно до принципу єдиної відповідальності.

Модуль Main

Модуль Main є точкою входу в систему і виконує роль оркестратора, він не містить логіки перевірки даних, але керує послідовністю виклику всіх інших процедур, забезпечує перевірку доступності серверів і отримує код менеджера. На початку модуля оголошено шість глобальних змінних, доступних з будь-якого модуля книги, що продемонстровано на рис. 3.5.

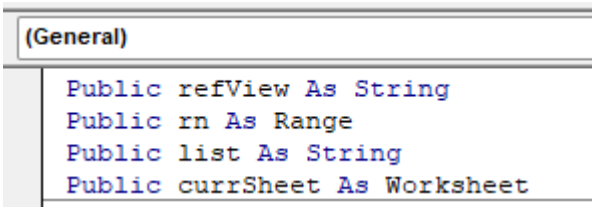


Рис. 3.5 Оголошення глобальних змінних в модулі Main

Змінна Manager As String зберігає ідентифікатор менеджера, отриманий з PERSONAL.XLSB. Змінна ssql As String призначена для формування рядка SQL-запиту перед передачею до універсальної процедури execute(). Змінна currSheet As Worksheet зберігає посилання на поточний аркуш, що обробляється в даний момент циклом або активований користувачем. Оголошення змінних на рівні модуля з ключовим словом Public забезпечує їх доступність з будь-якого модуля книги без необхідності передавати їх як аргументи між процедурами.

Процедура Start_all() реалізує режим повного аудиту і викликається натисканням кнопки «Update All» на стартовому аркуші. Її завданням є послідовний запуск перевірки на кожному аркуші книги і повідомити аналітика про завершення із зазначенням часу виконання.

Як показано на рис. 3.6, на початку процедура викликає check_servers() для перевірки доступності обох SQL-серверів і якщо хоча б один недоступний, виконання переривається ще до початку циклу, що унеможливлює отримання частково заповнених результатів. Далі фіксується початковий момент часу через Timer для подальшого розрахунку тривалості аудиту. Цикл For i = 2 To shCount

обходить усі аркуші книги починаючи з другого (перший аркуш є стартовим,тому не враховується). На кожній ітерації об'єкт поточного аркуша записується в глобальну змінну currSheet, після чого викликається select_validation(i), яка визначає і запускає відповідну процедуру перевірки. Після завершення циклу аналітик отримує повідомлення MsgBox з нагадуванням переглянути аркуші та часом виконання у форматі hh:mm:ss, розрахованим через різницю Timer - t, поділену на 86400 (кількість секунд на добу) для конвертації у формат часу.

```
'Запуск всіх макросів однією кнопочкою
Sub Start_all()
Dim i As Integer
Dim shCount As Integer
Dim t As Single

Call check_servers

t = Timer
shCount = ThisWorkbook.Worksheets.count

For i = 2 To shCount
    Set currSheet = ThisWorkbook.Worksheets(i)
    currSheetName = currSheet.Name
    Call select_validation(i)
Next
MsgBox "Всі проблеми були знайдені! Будь ласка, прогляньте всі аркуші та виправте помилки. Час виконання: " & _
Format((Timer - t) / 86400, "hh:mm:ss"), vbOKOnly, "Сповіщення про закінчення перевірки"
End Sub
```

Рис. 3.6 Процедура Start_all

Процедура Start_specific() реалізує режим точкового контролю і прив'язана до кнопки «Update Sheet» на кожному аркуші перевірки. Логіка є спрощеною версією Start_all(): зафіксувати початковий час – перевірити сервери – викликати select_validation() без аргументу – вивести повідомлення з назвою аркуша і часом виконання. Посилання на currSheet встановлюється до виклику check_servers(), це гарантує, що в повідомленні після завершення буде виведено назву саме того аркуша, який перевірявся. Процедура представлена на рис. 3.7

```
Sub Start_specific()
Dim t As Single

Set currSheet = ActiveSheet
t = Timer

Call check_servers
Call select_validation

MsgBox "Перевірка " & currSheet.Name & " закінчена! Перегляньте помилки. Час виконання: " & _
Format((Timer - t) / 86400, "hh:mm:ss"), vbOKOnly
End Sub
```

Рис. 3.7 Процедура Start_specific

Процедура check_servers() виконує перевірку мережевої доступності обох SQL-серверів перед будь-яким зверненням до бази даних. Це критично важливий захисний механізм, спроба відкрити ADO-з'єднання з недоступним сервером призводить до неконтрольованого зависання Excel на час очікування таймауту, тоді як ping-перевірка повертає результат протягом 1 секунди.

На рис. 3.8 показаний механізм перевірки, який реалізований через COM-об'єкт WScript.Shell, що виконує системну команду ping -n 1 -w 1000, тобто один пакет з тайм-аутом одна секунда. Метод objShell.Run() повертає код завершення процесу, 0 означає успішний ping, будь-яке інше значення – недоступність. Результат записується у булеву змінну boolPingSuccess. Якщо перевірка повертає False, об'єкт Shell звільняється, аналітику виводиться повідомлення з назвою недоступного сервера, і процедура завершується командою Exit Sub. Перевірки виконуються послідовно для sql_1 і sql_2, тож при недоступності першого сервера друга перевірка не виконується. Параметр windowStyle = 0 приховує вікно командного рядка від користувача, забезпечуючи непомітне виконання у фоні.

```
Sub check_servers()
Dim strServer As String
Dim objShell As Object
Dim boolPingSuccess As Boolean

'перевірка серверів
Set objShell = CreateObject("WScript.Shell")
strServer = "sql_1"
boolPingSuccess = objShell.Run("ping -n 1 -w 1000 " & strServer, 0, True) = 0
If boolPingSuccess = False Then
Set objShell = Nothing
MsgBox "Сервер " & strServer & " не працює"
Exit Sub
End If
strServer = "sql_2"
boolPingSuccess = objShell.Run("ping -n 1 -w 1000 " & strServer, 0, True) = 0
If boolPingSuccess = False Then
Set objShell = Nothing
MsgBox "Сервер " & strServer & " не працює"
Exit Sub
End If
Set objShell = Nothing
End Sub
```

Рис. 3.8 Процедура перевірки доступності серверів check_servers

Процедура select_validation(Optional n As Long) є центральним диспетчером системи, продемонстрована нижче на рис. 3.9.

```
Sub select_validation(Optional n As Long)
Init_manager
If Manager = "" Then
    MsgBox "Не прошитий код менеджера в Excel"
    Exit Sub
End If

If n Is Null Then
    Set currSheet = ActiveSheet
Else
    Set currSheet = ThisWorkbook.Worksheets(n)
End If

Select Case currSheet.Name
    Case "ГрупиПідгрупи"
    Call groups
    Case "ДвіЦіни"
    Call twoPrices
    Case "Од.продажу та зберіг."
    Call measurs
    Case "ЗнятоЗВиробнц.і є в Наявн."
    Call disc_and_avail
    Case "Перехресн.Посилання(ОЕ)"
    Call crossedLinksOE
    Case "Перехресн.Посилання"
    Call crossedLinks
    Case "Код1!=Код2"
    Call code1_and_code2
    Case "Критерії"
    Call criterias
    Case "Об`ємВагаНеттоБрутто"
    Call weightNettoBrutto
    Case "КодУКТЗЕД"
    Call desc
    Call pic
    Case "Незаповнено"
    Call Emptiness
End Select
End Sub
```

Рис. 3.9 Процедура select_validation

Першим кроком виконується виклик Init_manager(), процедури з модуля Additional, яка звертається до PERSONAL.XLSB і записує ідентифікатор менеджера в глобальну змінну Manager. Якщо змінна залишилась порожньою, процедура виводить повідомлення і завершується, що унеможливлює запуск перевірок без персоналізації.

Далі визначається цільовий аркуш: якщо аргумент n не передано (n Is Null), використовується ActiveSheet, в іншому випадку аркуш визначається через ThisWorkbook.Worksheets(n). Конструкція Select Case currSheet.Name зіставляє назву аркуша з переліком відомих перевірок і викликає відповідну процедуру. Всього Select Case охоплює одинадцять аркушів, що відповідає одинадцяти типам перевірок, реалізованих у системі.

Процедура execute(str, rng, shtname, num) є ключовим інфраструктурним компонентом системи, що збирає всю логіку безпосередньої взаємодії з базою даних і вивантаження результатів на робочий аркуш. Кожна процедура перевірки формує відповідний SQL-запит і передає в execute() чотири обов'язкові аргументи, якими є рядок SQL-запиту, діапазон для очищення, назва цільового аркуша та початковий стовпець для вивантаження даних. Реалізацію цієї універсальної процедури вивантаження показано на рис. 3.10.

```
Public Sub execute(str As String, rng As Range, shtname As String, num As Integer)
Dim conn As Object
Dim rs As Object
Dim strSQL As String
Dim i As Integer

Application.ScreenUpdating = False
rng.Cells.ClearContents
Set conn = CreateObject("ADODB.Connection")

conn.CommandTimeOut = 1000
conn.Open

strSQL = str
Set rs = CreateObject("ADODB.Recordset")
rs.Open strSQL, conn

For i = 0 To rs.Fields.count - 1
    Sheets(shtname).Cells(1, i + num).Value = rs.Fields(i).Name
Next i

Sheets(shtname).Cells(2, num).CopyFromRecordset rs
noProblems (shtname)
rs.Close
conn.Close
Set rs = Nothing
Set conn = Nothing
End Sub
```

Рис. 3.10 Процедура execute, яка відповідає за виконання SQL-запитів та вивантаження даних на аркуші

Алгоритм виконання процедури передбачає послідовну реалізацію кількох технологічних етапів. На початковому етапі вимикається оновлення екрана за допомогою команди Application.ScreenUpdating = False, що значно прискорює вивантаження великих наборів даних. Після цього здійснюється очищення цільового діапазону методом rng.Cells.ClearContents, що гарантує відсутність застарілих даних від попереднього сеансу аудиту. Наступний крок передбачає створення ADO-з'єднання з встановленням тайм-ауту тривалістю 1000 секунд через властивість conn.CommandTimeOut = 1000. Дане значення обрано з урахуванням того, що запити з перехресними посиланнями між таблицями бази даних можуть тривало виконуватися на великих обсягах інформації холдингу.

Безпосереднє виконання SQL-запиту реалізується через об'єкт ADODB.Recordset. Після отримання даних керуючий цикл For i = 0 To rs.Fields.Count - 1 забезпечує запис назв полів результуючої вибірки у перший рядок аркуша, автоматично формуючи заголовки стовпців звіту. Всі знайдені рядки з помилками вивантажуються на аркуш єдиним викликом методу CopyFromRecordset, який є значно ефективнішим за послідовний поочередний запис клітинка-за-клітинкою. На завершальному етапі викликається допоміжна процедура noProblems(shtname), яка перевіряє наявність даних у вивантаженому діапазоні, і якщо аркуш залишився порожнім, автоматично виводить повідомлення «Помилок не знайдено!». Робота процедури завершується закриттям з'єднання та очищенням об'єктів за допомогою команд Set rs = Nothing та Set conn = Nothing, що забезпечує обов'язкове звільнення системних ресурсів і запобігає витоку оперативної пам'яті.

Модуль Specific

Модуль Specific є найбільшим за обсягом кодової бази і містить окремі процедури для кожного типу бізнес-перевірки. Кожна процедура відповідає одному аркушу книги і виконує одне чітко визначене завдання, а саме сформувати SQL-запит з урахуванням коду менеджера, передати його до процедури execute() і вивантажити результати на відповідний аркуш. Модуль містить такі процедури перевірок: groups(), desc(), pic(), criterias(), crossedLinks(), twoPrices(), measurs(),

weightNettoBrutto(), code1_and_code2(), disc_and_avail(). Окрім основних перевірок, модуль містить допоміжні процедури вивантаження довідників, до них належать groups_handbook(), desc_handbook(), criterias_handbook().

Процедура groups() є найскладнішою з точки зору структури SQL-запиту, адже вона виявляє товари, у яких комбінація категорія + група + підгрупа не відповідає жодному допустимому значенню корпоративного довідника. Логіка побудована у три рівні вкладеності, що продемонстровано у рис. 3.11.

```
Sub groups()
refView = "SELECT DISTINCT Довідник FROM(" & _
          "SELECT CASE WHEN category = 'TOBAP' THEN category +  ' ' + group_ref + ' ' + subgroup ELSE category + ' ' + group_ref END AS Довідник " & _
          "FROM(SELECT Group.[Category Code] AS category, Group.[Group] AS group_ref FROM Group) AS catGroup " & _
          "LEFT OUTER JOIN(SELECT [Category Code] AS categ, Subgroup.[Group Code] AS group_sub, SubGroup.[Subgroup Code] AS subgroup " & _
          "FROM SubGroup) AS groupSUB " & _
         "ON catGroup.category + catGroup.[group_ref] = groupSUB.categ + groupSUB.[group_sub]) AS subquery"

ssql = "SELECT Items.[Item Art], Items.[Category], Items.[Group], Items.[SubGroup], Довідник FROM " & _
       "Items LEFT OUTER JOIN(" & refView & ") AS referenceTable ON Items.[Category] + ' ' + Items.[Group] + ' ' + Items.[SubGroup] = referenceTable.Довідник " & _
       "WHERE [Manager] = ('" & Manager & "') AND Довідник IS NULL "
Set currSheet = ThisWorkbook.Sheets("ГрупиПідгрупи")
Set rn = currSheet.Cells
Call execute(ssql, rn, currSheet.Name, 2)
End Sub
```

Рис. 3.11 Реалізація процедури groups(): тривнівневий SQL-запит

Перший рівень є внутрішнім підзапитом catGroup, що формує плоску таблицю з усіх пар категорія/група з таблиці Group. Другий рівень відповідає за приєднання groupSUB через LEFT OUTER JOIN, а саме додає підгрупи з таблиці SubGroup та конкатенує категорію і групу як ключ з'єднання. Третій рівень – це зовнішній підзапит, який збирає всі допустимі комбінації у єдиний рядок через CASE WHEN: для товарів типу «TOBAP» формується рядок «категорія група підгрупа», для інших категорій буде тільки «категорія група». Результат зберігається у змінній refView як рядок підзапиту для повторного використання в основному SELECT.

Основний запит виконує LEFT OUTER JOIN між таблицею Items та підзапитом refView з умовою з'єднання через конкатенацію полів категорія, група і підгрупа товару з рядком довідника. Фільтр WHERE Довідник IS NULL залишає лише ті товари, для яких не знайшлося відповідності, тобто ті, чия комбінація атрибутів є некоректною. Додатково застосовується фільтр за менеджером.

Процедура-двійник groups_handbook() реалізує окремий механізм вивантаження довідника допустимих комбінацій безпосередньо на той самий

аркуш «ГрупиПідгрупи», але в стовпці F:H, тоді як результати перевірки починаються зі стовпця B. Ця процедура прив'язана до окремої кнопки «Довідник» на аркуші та не викликається з циклу Start_all() чи Start_specific(). Аналітик може завантажити довідник у будь-який момент для ручної звірки, наприклад, щоб з'ясувати, яку саме комбінацію треба використати для конкретного товару. Аналогічний підхід реалізовано для процедур desc_handbook() та criterias_handbook(), що завантажують еталонні значення описів і критеріїв відповідно. Таким чином, кожен аркуш з перевіркою перетворюється на самодостатнє робоче місце: ліворуч знаходиться список помилок, праворуч – довідник допустимих значень для виправлення. Демонстрація процедур вивантаження довідників зображена на рис. 3.12.

```
Sub groups_handbook()
ssql = "SELECT [Довідник категорія], [Довідник група], [Довідник підгрупа] FROM(" & _
         "SELECT Group.[Category Code] AS [Довідник категорія], Group.[Group] AS [Довідник група]  FROM Group) AS catGroup " & _
         "LEFT OUTER JOIN(SELECT SubGroup.[Group Code] AS group_sub, [Category Code] As category,  SubGroup.[Subgroup Code] AS [Довідник підгрупа] FROM SubGroup) AS groupSUB " & _
         "ON catGroup.[Довідник категорія] + catGroup.[Довідник група] = groupSUB.[category] + groupSUB.[group_sub]"
Set currSheet = ThisWorkbook.Sheets("ГрупиПідгрупи")
Set rn = currSheet.Range("F:H")
Call execute(ssql, rn, currSheet.Name, 6)
End Sub

Sub desc_handbook()
ssql = "SELECT Standard_Description.[Description] AS [Довідник опис], Standard_Description.[Custom Category Code] AS [Довідник код УКТЗЕД] FROM Standard_Description"
Set currSheet = ThisWorkbook.Sheets("КодУКТЗЕД")
Set rn = currSheet.Range("E:F")
Call execute(ssql, rn, currSheet.Name, 5)
End Sub

Sub criterias_handbook()
ssql = "SELECT Criterias_Handbook.[Criteria] AS [Довідник критерія], Criterias_Handbook.[Value] AS [Довідник значення] FROM Criterias_Handbook  "
Set currSheet = ThisWorkbook.Sheets("Критерії")
Set rn = currSheet.Range("E:F")
Call execute(ssql, rn, currSheet.Name, 5)
End Sub
```

Рис. 3.12 Процедури вивантаження довідників.

Як видно на рис. 3.13, процедура crossedLinks() демонструє особливий патерн реалізації, вона виконує два незалежних SQL-запити і вивантажує їх результати на один аркуш у різні стовпці. Перший запит виявляє перехресні посилання з помилками форматування: порожній бренд, невідповідність між Replace No та Cross_Ref_Code, або наявність спецсимволів (#, +, *). Результат вивантажується починаючи зі стовпця B.

```
Sub crossedLinks()
ssql = "SELECT [Item Art], [Replace No], [Brand], [Cross_Ref_Code] FROM Cross_Item " & _
         "WHERE [Brand] <> 'auto' AND [Responsible Manager] IN ('" & Manager & "') AND [Type] = 'Cross' AND [Cross_Ref_Type] NOT IN('Vendor', 'Automatic', 'Customer') AND ([Brand] = '' " & _
         "OR [Replace No] NOT LIKE [Cross_Ref_Code] OR Cross_Item.[Cross_Ref_Code] LIKE '%#%' OR Cross_Item.[Cross_Ref_Code] LIKE '%+%' OR Cross_Item.[Cross_Ref_Code] LIKE '%*%')"
Set rn = currSheet.Cells
Set currSheet = ThisWorkbook.Sheets("Перехресн.Посилання")
Call execute(ssql, rn, currSheet.Name, 2)
ssql = "SELECT [Item Art] AS [Код Товару, в якого більше 1 перехресн.посилання] FROM Cross_Item " & _
       "WHERE [Cross_Ref_Type] = 'TecDoc' AND [Responsible Manager] IN ('" & Manager & "') " & _
       "GROUP BY [Item Art] HAVING COUNT(*) > 1 "
Set rn = currSheet.Columns(7)
Call execute(ssql, rn, currSheet.Name, 7)
End Sub
```

Рис. 3.13 Реалізація процедури crossedLinks(): два запити на один аркуш

Другий запит вирішує самостійне завдання, він виявляє товари, у яких більше одного перехресного посилання типу TecDoc, що є порушенням бізнес-правила однозначності. Використовується конструкція GROUP BY [Item Art] HAVING COUNT(*) > 1. Результат вивантажується в стовпець G того самого аркуша через окремий виклик execute() з параметром num = 7. Така компоновка дозволяє аналітику бачити два пов'язаних типи проблем одночасно на одному аркуші без необхідності перемикатись між листами.

Процедура weightNettoBrutto() реалізує класичний приклад перевірки числової бізнес-логіки, яку неможливо відтворити через стандартну валідацію ERP-системи на рівні окремого поля. SQL-запит відбирає товари, у яких виконується хоча б одна з чотирьох умов: брутто-вага менша за нетто-вагу, брутто-вага рівна або менша нуля, нетто-вага рівна або менша нуля, або об'єм є від'ємним. На рис. 3.14 показано, що умови об'єднані оператором OR – це дозволяє виявити будь-яке з порушень єдиним запитом.

```
Sub weightNettoBrutto()
Set currSheet = ThisWorkbook.Sheets("Об`ємВагаНеттоБрутто")
Set rn = currSheet.Cells
ssql = "SELECT Items.[Item Art], Items.[Netto weight] AS 'Нетто вага', Items.[Brutto weight] AS 'Брутто вага', Items.[Volume] As 'Об`єм' " & _
       "FROM Items " & _
       "Where [Manager] IN ('" & Manager & "') AND (Items.[Brutto weight] < Items.[Netto weight] OR Items.[Brutto weight] <= 0 OR Items.[Netto weight] <= 0 OR Items.[Volume] < 0) "
Call execute(ssql, rn, currSheet.Name, 2)
End Sub
```

Рис. 3.14 Перевірка логічної узгодженості числових атрибутів товару

Решта процедур модуля – measurs(), code1_and_code2(), twoPrices(), disc_and_avail(), desc(), pic(), criterias(), побудовані за єдиним патерном, а саме визначення цільового аркуша, формування рядка ssql з фільтром за Manager, встановлення діапазону очищення і виклик execute(). Така уніфікація дозволяє додавати нові перевірки за шаблоном, достатньо написати SQL-запит з бізнес-правилом, обернути його в процедуру за зразком існуючих і зареєструвати в select_validation(). Час розробки нової перевірки за таких умов не перевищує 15–30 хвилин навіть без глибоких знань VBA.

Спільною характеристикою всіх запитів модуля є застосування LEFT OUTER JOIN замість INNER JOIN для з'єднання з довідниками. Це принципове архітектурне рішення, оскільки INNER JOIN повертав би лише товари з

коректними даними, тоді як LEFT OUTER JOIN у поєднанні з фільтром WHERE ... IS NULL повертає виключно проблемні записи – ті, для яких у правій таблиці не знайшлося відповідності. Такий підхід є стандартним патерном SQL для виявлення відсутніх зв'язків і семантичних невідповідностей у реляційних даних.

Модуль Emptiness

Модуль Emptiness відповідає за окремий тип контролю якості – виявлення незаповнених обов'язкових полів у картках товарів. Функціонально він суттєво відрізняється від модуля Specific, якщо Specific шукає семантичні суперечності між полями, то EmptinessModule перевіряє наявність значення як такого. Всі результати виводяться на один спеціалізований аркуш «Незаповнено».

Модуль складається з двох логічних блоків. Перший відповідає за зведений підрахунок незаповнених полів по всьому асортименту менеджера. Другий про деталізацію по конкретному полю, вивантаження переліку артикулів, у яких саме це поле порожнє. Обидва блоки доступні як через повний цикл аудиту, так і через окремі кнопки безпосередньо на аркуші.

Процедура Emptiness() є точкою входу модуля і викликається з select_validation() при обробці аркуша «Незаповнено». Її SQL-запит (рисунок 3.15) реалізує повернення єдиного рядку з кількістю незаповнених значень по кожному з десяти атрибутів товару.

```
Public rng As Range
Public fltr As String
Public str As String
Const sheetName As String = "Незаповнено"
'Перевірка на заповненість(підрахунок)
Sub Emptiness()
Dim conn As Object
Dim rs As Object
Dim strSQL As String
Dim i As Integer

Application.ScreenUpdating = False
Sheets(sheetName).Cells.clear
Set conn = CreateObject("ADODB.Connection")
conn.ConnectionString = "Provider=SQLOLEDB.1;Integrated Security=SSPI;Initial Catalog=NAV_19_Reports;Data Source=sqldev"
conn.CommandTimeOut = 120
conn.Open
strSQL = "SELECT " & _
         "SUM(CASE WHEN [Vendor No] = '' THEN 1 ELSE 0 END) AS [Код постачальника], SUM(CASE WHEN [Description] = '' THEN 1 ELSE 0 END) AS [Опис], " & _
         "SUM(CASE WHEN [Search description] = '' THEN 1 ELSE 0 END) AS [Опис пошуку], SUM(CASE WHEN [Base unit of measurement] = '' THEN 1 ELSE 0 END) " & _
         "AS [Базова одиниця], SUM(CASE WHEN [Brand] = '' THEN 1 ELSE 0 END) AS [Бренд], " & _
         "SUM(CASE WHEN [Volume] = 0 THEN 1 ELSE 0 END) AS [Об'єм], " & _
         "SUM(CASE WHEN [Discount group] = '' THEN 1 ELSE 0 END) AS [Група знижки], " & _
         "SUM(CASE WHEN [TOP] = '' THEN 1 ELSE 0 END) AS [ТОП], " & _
         "SUM(CASE WHEN [Project] = '' THEN 1 ELSE 0 END) AS [Проект], " & _
         "SUM(CASE WHEN [Substitution Key] = '' THEN 1 ELSE 0 END) AS [Ключ заміни] " & _
         "FROM Items WHERE [Manager] = ('" & Manager & "') "
Set rs = CreateObject("ADODB.Recordset")
rs.Open strSQL, conn

For i = 0 To rs.Fields.count - 1
    Sheets(sheetName).Cells(1, i + 3).Value = rs.Fields(i).Name
Next i

Sheets(sheetName).Cells(2, 3).CopyFromRecordset rs

rs.Close
conn.Close
Set rs = Nothing
Set conn = Nothing
fp_auto
End Sub
```

Рис. 3.15 Зведений SQL-запит модуля Emptiness: підрахунок порожніх полів

Технічно це досягається конструкцією SUM(CASE WHEN [поле] = '' THEN 1 ELSE 0 END), повторена для кожного з перевірених атрибутів, а саме код постачальника, опис, опис пошуку, базова одиниця, бренд, об'єм, група знижки, ТОП, проєкт, ключ заміни. Результатом є один рядок з десятьма числами, який вивантажується в рядок 2 аркуша починаючи зі стовпця C, тоді як рядок 1 містить автоматично сформовані заголовки з назв полів. Аналітик одразу бачить зведену картину, скільки товарів у його зоні відповідальності мають проблеми по кожному атрибуту, без необхідності гортати довгі списки.

Для кожного з десяти перевірених атрибутів у модулі реалізована окрема процедура-делегат: VendorNo(), Description(), Brand(), Volume() тощо. Кожна з них лише визначає два параметри: діапазон стовпця для вивантаження і рядок фільтра fltr з умовою порожнього поля, після чого делегує виконання до спільної процедури showItemNo(). Такий підхід уникає дублювання коду, логіка підключення до БД, виконання запиту і вивантаження реалізована рівно один раз. Кілька з цих функцій продемонстровані на рис. 3.16

```
'Переглянути незаповнені поля для номеру постачальника
Sub VendorNo()
Set rng = Range("C4:C" & Sheets(sheetName).Rows.count)
fltr = " [Vendor No] = '' "
Call showItemNo(fltr, 3, rng)
End Sub

'Переглянути незаповнені поля для опису
Sub Description()
Set rng = Range("D4:D" & Sheets(sheetName).Rows.count)
fltr = " [Description] = '' "
Call showItemNo(fltr, 4, rng)
End Sub

'Переглянути незаповнені поля для опису пошуку
Sub SearcDescription()
Set rng = Range("E4:E" & Sheets(sheetName).Rows.count)
fltr = " [Search description] = '' "
Call showItemNo(fltr, 5, rng)
End Sub

'Переглянути незаповнені поля для базової одиниці
Sub Measurement()
Set rng = Range("F4:F" & Sheets(sheetName).Rows.count)
fltr = " [Base unit of measurement] = '' "
Call showItemNo(fltr, 6, rng)
End Sub

'Переглянути незаповнені поля для коду УКТВЕД
Sub Brand()
Set rng = Range("G4:G" & Sheets(sheetName).Rows.count)
fltr = " [Brand] = '' "
Call showItemNo(fltr, 7, rng)
End Sub
```

Рис. 3.16 Патерн процедур-делегатів

Процедура showItemNo(), продемонстрована на рис. 3.17, приймає чотири параметри: рядок фільтра, номер стовпця і діапазон для очищення. Далі формується стандартний запит SELECT [Item Art] FROM Items WHERE [фільтр] AND [Manager] = поточний менеджер. Результат вивантажується у вказаний стовпець, заголовок формується автоматично з назв полів Recordset.

```
Sub showItemNo(fltr, n As Integer, rng)
Dim conn As Object
Dim rs As Object
Dim i As Long
Dim strSQL As String

Application.ScreenUpdating = False
rng.Cells.clear

Set conn = CreateObject("ADODB.Connection")

conn.CommandTimeOut = 120
conn.Open

strSQL = "SELECT [Item Art] FROM Items WHERE " & fltr & " AND [Manager] IN ('" & Manager & "')"

Set rs = CreateObject("ADODB.Recordset")
rs.Open strSQL, conn

For i = 0 To rs.Fields.count - 1
    Sheets(sheetName).Cells(4, i + n).Value = rs.Fields(i).Name
Next i

Sheets(sheetName).Cells(5, n).CopyFromRecordset rs
rs.Close
conn.Close
Set rs = Nothing
Set conn = Nothing
End Sub
```

Рис. 3.17 Загальна процедура вивантаження showItemNo()

Модуль Additional

Модуль Additional виконує допоміжну роль в архітектурі системи, він містить процедури, що забезпечують коректний стан робочого середовища до і після виконання перевірок. Жодна з трьох процедур модуля не містить бізнес-логіки перевірки даних, натомість вони відповідають за персоналізацію (Init_manager()), зворотний зв'язок з аналітиком (noProblems()) та підготовку аркушів до нового циклу аудиту (clears()).

Процедура Init_manager() реалізує механізм персоналізації системи, що є ключовою функціональною вимогою, що відрізняє розроблений інструмент від універсальних рішень. Її логіка наведена на рисунку 3.18.

```
Public Sub Init_manager()
Dim w As Workbook

If Manager <> "" Then
    Exit Sub
End If

'витягуємо код менеджера
For Each w In Workbooks
    If w.Name Like "PERSONAL*" Then
        On Error Resume Next
            Manager = Workbooks(w.Name).CustomDocumentProperties("ManagerCode").Value
        On Error GoTo 0
        Exit For
    End If
Next
End Sub
```

Рис. 3.18 Реалізація персоналізації аудиту

Механізм зчитування реалізований через перебір усіх відкритих книг Excel у циклі For Each w In Workbooks. При знаходженні книги, чия назва відповідає шаблону PERSONAL*, процедура звертається до її CustomDocumentProperties (колекції користувацьких властивостей документа) і зчитує значення властивості ManagerCode. Конструкція On Error Resume Next ... On Error GoTo 0 захищає від ситуації, коли властивість ще не створена в PERSONAL.XLSB конкретного аналітика, в такому випадку змінна Manager залишається порожньою, і select_validation() зупинить виконання з відповідним повідомленням.

Процедура noProblems(), логіку якої показано на рис. 3.19, викликається наприкінці кожного виконання execute() і відповідає за зворотний зв'язок у випадку, коли перевірка не виявила жодної помилки.

```
Public Sub noProblems(sheetName As String)
Dim arr As Range
Dim cell As Range
Dim ws As Worksheet

Set ws = ThisWorkbook.Sheets(sheetName)
Set arr = ws.Range("B2:B2")

For Each cell In arr
    If cell.Value = "" Then
        cell.Value = "Помилок не знайдено!"
        Exit For
    End If
Next cell
End Sub
```

Рис. 3.19 Реалізація процедури noProblems()

Процедура перевіряє значення клітинки B2 цільового аркуша: якщо вона порожня після вивантаження Recordset – це означає, що SQL-запит не повернув жодного рядка, тобто порушень не виявлено. У такому випадку в клітинку записується рядок «Помилок не знайдено!». Це просте, але продумане рішення потрібне, щоб аналітик не залишався перед порожнім аркушем, що могло б бути сприйнято як збій, а отримував явне підтвердження коректності даних. Перевірка саме клітинки B2 обумовлена тим, що всі результати перевірок починаються зі стовпця B, а рядок 1 завжди зайнятий заголовками полів.

3.3 Інтеграція рішення в робочий процес аналітиків

Успішність практичного впровадження розробленого програмного забезпечення безпосередньо залежить від рівня його інтеграції в існуючі бізнес-процеси холдингу та зручності взаємодії користувачів із розробленим інтерфейсом. З метою мінімізації часових витрат аналітиків на освоєння нового інструментарію та забезпечення максимальної ергономічності, користувацький інтерфейс системи конфігурується безпосередньо в межах звичного середовища табличного процесора MS Excel. Головна сторінка інструменту, що розміщується на робочому аркуші «Старт», виконує роль інтерактивної панелі керування. На ній зосереджено спеціалізовані графічні елементи управління у вигляді кнопок, за допомогою яких користувач може в один клік ініціювати сеанси комплексного або локального аудиту бази даних ERP-системи. Зовнішній вигляд розробленого інтерфейсу головної сторінки з кнопками запуску перевірок та інформаційними блоками продемонстровано на рис. 3.20.

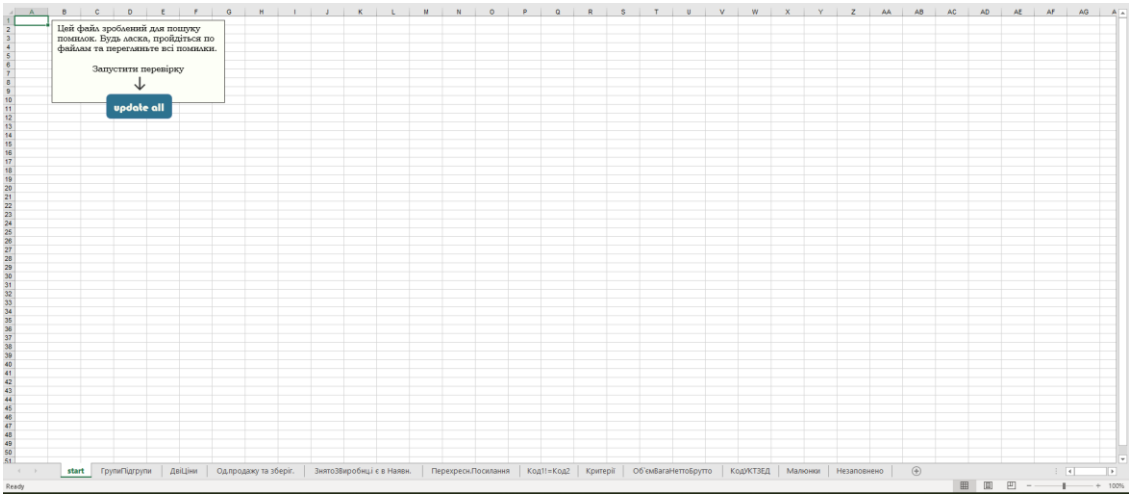


Рис. 3.20 Інтерфейс головної сторінки інструменту контролю цілісності даних

Особливу увагу при проектуванні користувацького інтерфейсу було приділено структуризації робочих аркушів, призначених для відображення результатів аудиту. Серед усього масиву вкладок програмного комплексу виділяються два унікальні типи інтерфейсних рішень, що відображають специфіку обробки різних категорій помилок. Перший тип архітектури робочого простору орієнтований на перевірки, які тісно пов'язані з нормативно-довідковою інформацією холдингу, прикладом чого є аркуш «Групи-підгрупи». Окрім стандартної табличної частини для виведення помилкових позицій, дана вкладка містить унікальний функціональний елемент у вигляді графічної кнопки «Вивантажити довідник». Наявність цього елемента дозволяє аналітику оперативно отримати еталонну структуру ієрархії категорій безпосередньо з бази даних ERP для проведення швидкої звірки. Описану унікальну структуру інтерфейсу з можливістю завантаження довідкових даних продемонстровано на рис. 3.21.

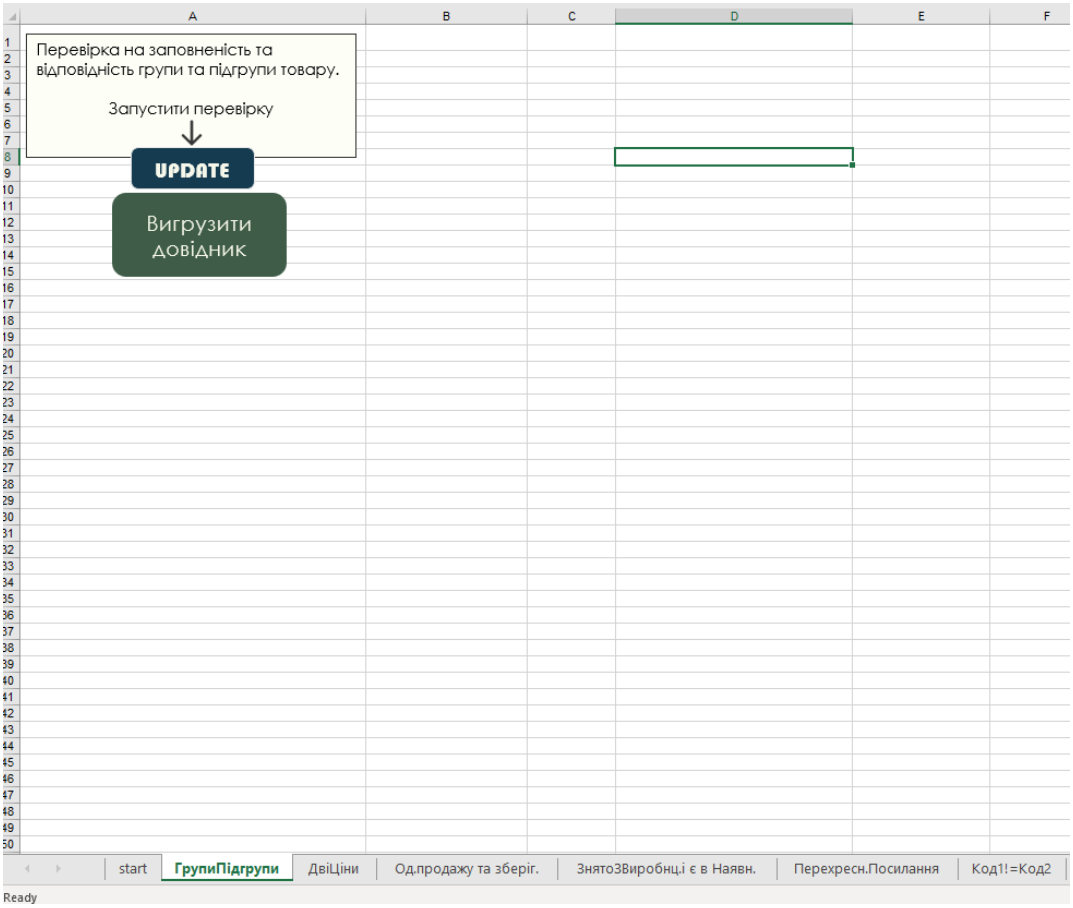


Рис. 3.21 Інтерфейс аркуша «Групи-підгрупи» з інтегрованою кнопкою вивантаження довідника

В межах розробленого інструменту є спеціалізований аркуш «Незаповнено», який відображає логіку роботи раніше аналізованого модуля Emptiness. На відміну від решти вкладок, які фіксують конкретні критичні помилки в атрибутах, цей аркуш виконує роль панелі превентивного моніторингу та загального аудиту повноти даних картки товару. Інтерфейс даного аркуша містить комплексну зведену таблицю, де у розрізі категорій відповідальності аналітика відображається точна кількісна статистика незаповнених обов’язкових полів, таких як базові одиниці виміру, коди постаяальника тощо. Візуальне представлення інтерфейсу цього унікального зведеного звіту повноти бази даних представлено на рис. 3.22.

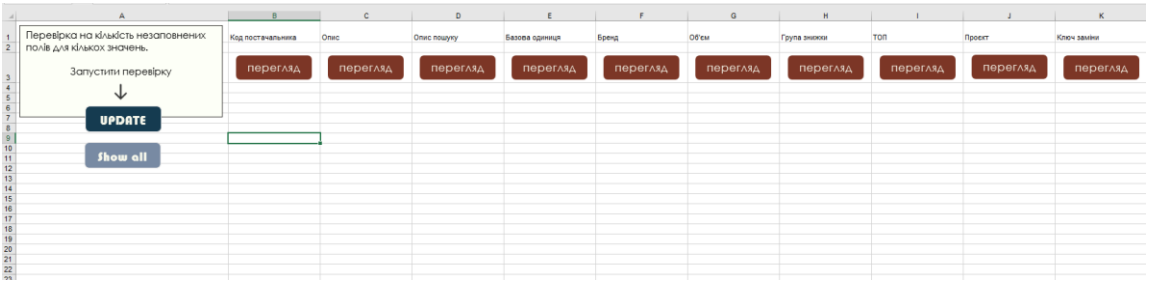


Рис. 3.22 Інтерфейс спеціалізованого аркуша «Незаповнено»

Технічна інтеграція інструменту на робочих станціях менеджерів компанії реалізується через використання централізованих механізмів автоматичного завантаження додатків. Весь програмний комплекс, включаючи логіку SQL-запитів, процедури обробки помилок та екранні форми, інкапсульовано у спеціалізований файл бінарної книги макросів. З метою оптимізації адміністрування та захисту вихідного коду від несанкціонованих модифікацій, дане рішення інтегрується безпосередньо в системний файл Інструменти.xlsb, який розміщується в єдиній спільній папці налаштувань на корпоративному сервері компанії. Для звичайних аналітиків цей файл є доступним виключно у режимі читання, що повністю виключає ризики випадкової зміни або пошкодження алгоритмів валідації користувачами.

Така архітектурна конфігурація унеможливлює необхідність локального дублювання програмного файлу на кожен окремий комп'ютер працівника холдингу. Інтеграція інструменту в робочий простір конкретного аналітика реалізується шляхом конфігурації параметрів табличного процесора MS Excel, де в налаштуваннях каталогу автозавантаження (XLSTART) прописується прямий повний мережевий шлях до згаданого спільного файлу. Завдяки цьому під час кожного запуску табличного процесора програмний комплекс автоматично ініціалізується у фоновому режимі, звертаючись до актуальної версії макросів на сервері та залишаючись повністю доступним для проведення аудиту без необхідності локального збереження чи ручного відкриття файлу з кодом розробника.

Розроблене програмне рішення підключається до користувацької панелі швидкого доступу у вигляді персоналізованої вкладки стрічки. Конфігурацію адаптованої стрічки швидкого доступу з інтегрованою кнопкою запуску системи зображено на рис. 3.23.

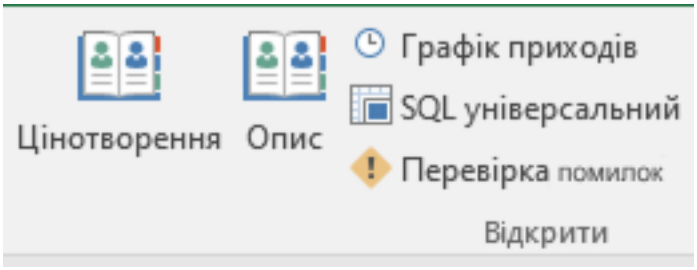


Рис. 3.23 Панель швидкого доступу MS Excel з інтегрованою кнопкою запуску інструменту

Шляхом взаємодії з цим кастомним елементом аналітик має можливість миттєво здійснити перехід до робочого файлу «Перевірка помилок» з будь-якого іншого відкритого документа, автоматично розгортаючи необхідний функціональний простір для аудиту товарної номенклатури.

3.4 Тестування і оцінка ефективності системи

Тестування системи проводилось в умовах реального виробничого середовища компанії на діючій базі даних Dynamics NAV. Переважна більшість наведених нижче результатів отримана в рамках єдиного сеансу повної перевірки, запуском кнопки «Update All» на стартовому аркуші, після чого система послідовно виконала всі одинадцять перевірок і сформувала результати на відповідних аркушах. Скриншоти аркушів із результатами зроблено безпосередньо після завершення повного аудиту. Слід зазначити, що на всіх наведених скриншотах реальні артикули товарів приховані з міркувань комерційної конфіденційності, замість них відображаються частково замасковані значення.

Аркуш «ГрупиПідгрупи» після повного аудиту містить повідомлення «Помилок не знайдено!» (рисунок 3.24), заголовки стовпців сформовано автоматично (Item Art, Category, Group, Subgroup), але рядки з даними відсутні. Це підтверджує, що на момент тестування всі товари у зоні відповідальності менеджера мали коректні комбінації категорія/група/підгрупа відповідно до корпоративного довідника. Процедура noProblems() спрацювала коректно: порожній Recordset інтерпретовано як відсутність помилок і в клітинку B2 записано відповідний текст.

Рис. 3.24 Аркуш «ГрупиПідгрупи» після повного аудиту

Перевірка для аркуша «Од.продажу та зберіг.» виявила реальні розбіжності в даних, що продемонстровано на рис. 3.25. Система повернула список товарних номерів, у яких одиниця розміщення відрізняється від одиниці продажу. У виявлених записах одиниця продажу вказана «ШТ», тоді як одиниця розміщення пуста. SQL-умова [Unit of measurement for stock] != [Unit of Meas Code] коректно відфільтрувала саме проблемні позиції, надавши аналітику готовий перелік для виправлення в ERP.

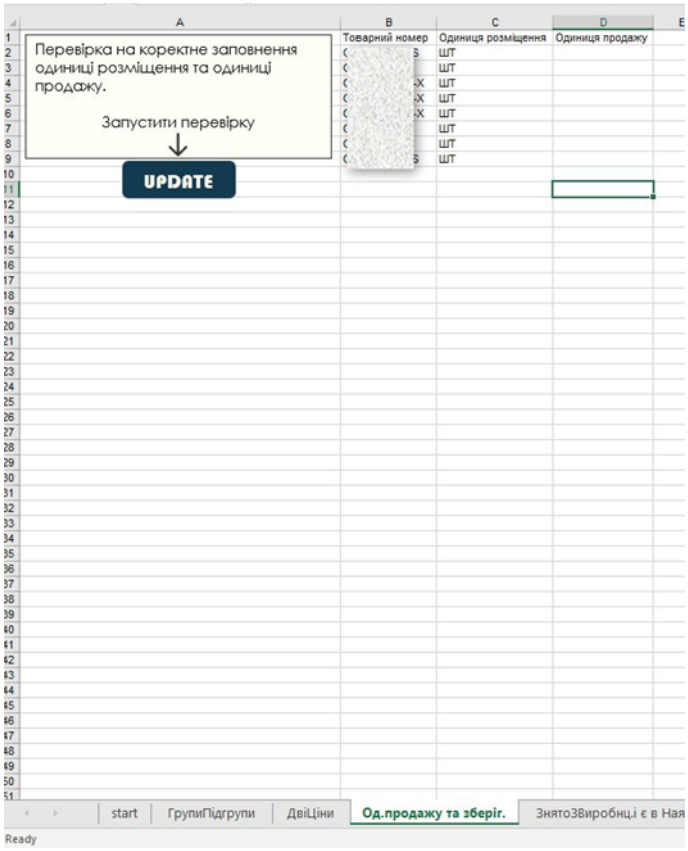


Рис. 3.25 Аркуш «Од.продажу та зберіг.» після повного аудиту

Аркуш «Незаповнено» після повного аудиту відображає зведену статистику по десяти атрибутах. З реальних даних, які показано на рис. 3.26, видно, що найбільша проблема зосереджена у полі «Код постачальника», система знайшла 1236 товарів без заповненого значення, поле «Бренд» має 13 незаповнених записів, а поле «Опис» всього 2. Решта атрибутів на момент тестування були заповнені повністю. Зведений рядок дозволяє аналітику за секунди оцінити загальний стан даних і визначити пріоритети виправлення.

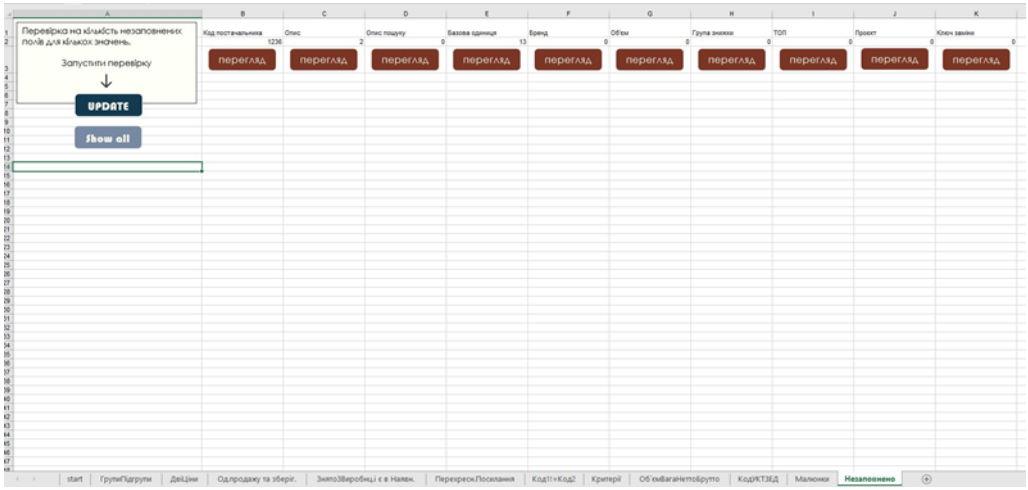


Рис. 3.26 Аркуш «Незаповнено» після повного аудиту

Після перегляду зведеної статистики аналітик натиснув кнопку «show all» для отримання деталізованих списків артикулів. На рис. 3.27 видно, що стовпець «Код постачальника» містить значно більше записів, ніж «Опис» та «Базова одиниця», що підтверджує різну природу проблем, відсутність постачальника є системною проблемою для великої частини асортименту, тоді як незаповнений опис чи базова одиниця вказують на точкові одиничні помилки.

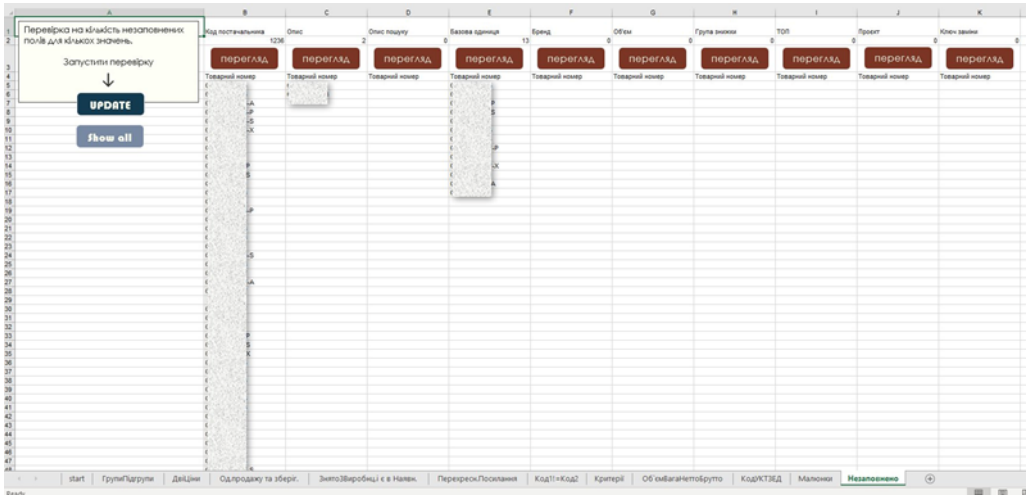


Рис. 3.27 Аркуш «Незаповнено»: деталізація

Після завершення повного циклу перевірок система вивела фінальне повідомлення «Всі проблеми були знайдені! Будь ласка, прогляньте всі аркуші та виправте помилки. Час виконання: 00:03:17». Зафіксований час є ключовим

показником продуктивності системи. Він суттєво вкладається у визначену нефункціональну вимогу (не більше 4–5 хвилин) і досягнутий завдяки поєднанню кількох технічних рішень, а саме фільтрації даних на стороні SQL Server, вимкненні оновлення екрану через Application.ScreenUpdating = False та масове вивантаження результатів через єдиний виклик CopyFromRecordset. Ознайомитись з фінальним повідомленням після перевірки можна на рис. 3.28

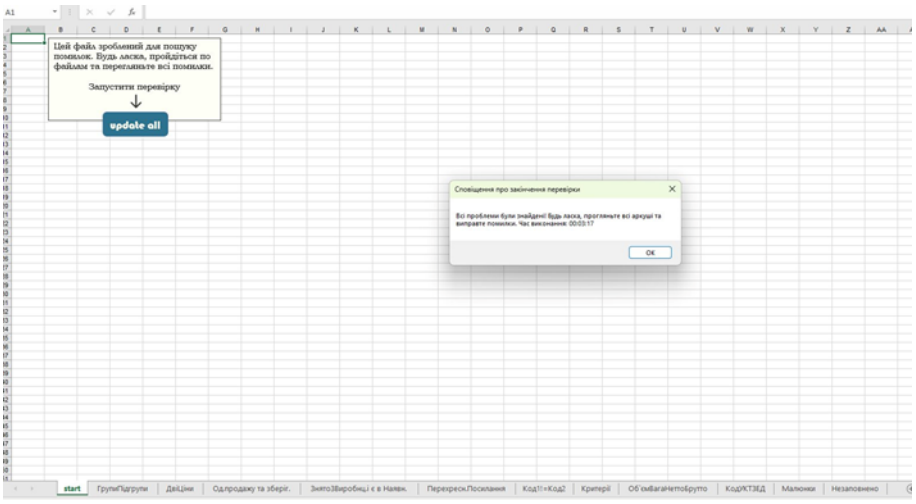


Рис. 3.28 Повідомлення про завершення повного аудиту

Аркуш «КодУКТЗЕД» отримав дані вже в рамках локальної перевірки на конкретному аркуші. Як продемонстровано на рис.3.29, чотири товари мають комбінації опис + код УКТЗЕД, що не відповідають жодному запису в таблиці Standard_Description. Виявлено позиції з описами «Молдінг скла гумовий» (код 4016995790) та «Скло автомобільне» (код 7007212090). На відміну від інших аркушів, для «КодУКТЗЕД» аналітик додатково натиснув кнопку «Вигрузити довідник», також не в рамках повного аудиту, а вручну після перегляду результатів, це показано на рис. 3.30. У стовпці F:G завантажено повний перелік допустимих комбінацій, що дозволяє одразу на місці підібрати коректне значення для виправлення.

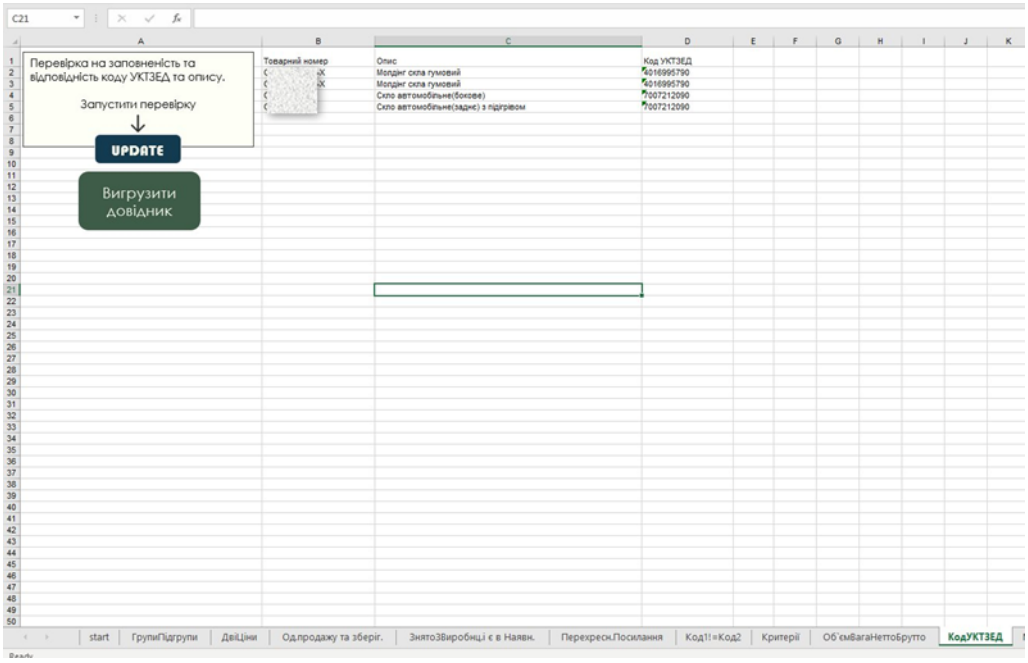


Рис. 3.29 Аркуш «КодУКТЗЕД» після локальної перевірки

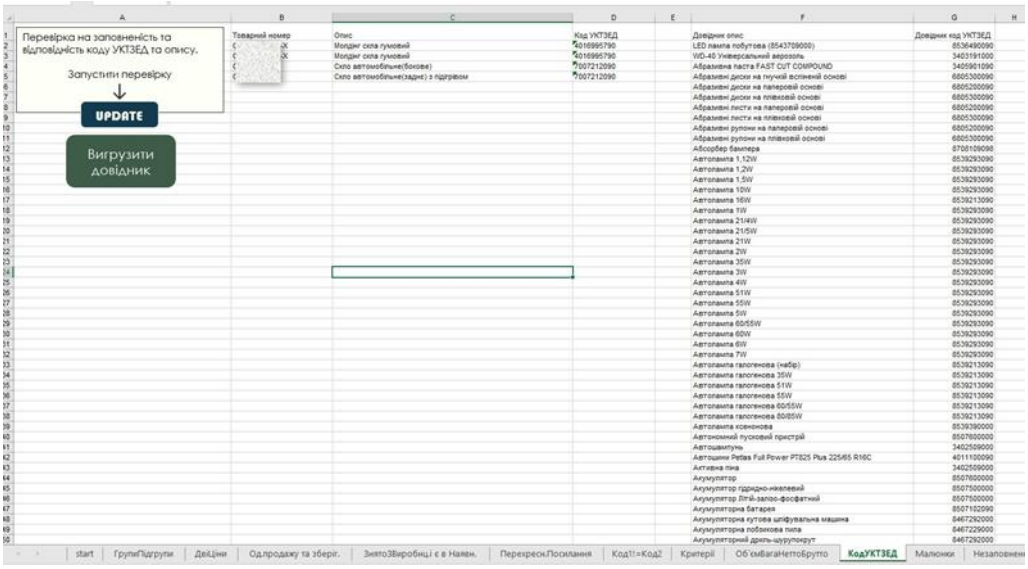


Рис. 3.30 Аркуш «КодУКТЗЕД» після вивантаження довідника

Окремим етапом тестування була перевірка поведінки системи в специфічних сценаріях. Ці тести проводились цілеспрямовано, умови помилки відтворювались вручну, щоб переконатись у коректності захисних механізмів.

Перший сценарій про відсутність коду менеджера. Для його відтворення керівник тимчасово видалив прошитий код менеджера з PERSONAL.XLSB на робочій станції аналітика, після чого було здійснено спробу запустити аудит.

Система коректно зупинила виконання ще на етапі Init_manager() і вивела повідомлення: «Не прошитий код менеджера в Excel! Будь ласка, зверніться до керівника». Жодного SQL-запиту без фільтра менеджера виконано не було, що стало підтвердженням того, що система не може бути запущена у «знеособленому» режимі, і будь-яка спроба це зробити завершується інформативною відмовою, як продемонстровано на рис. 3.31.

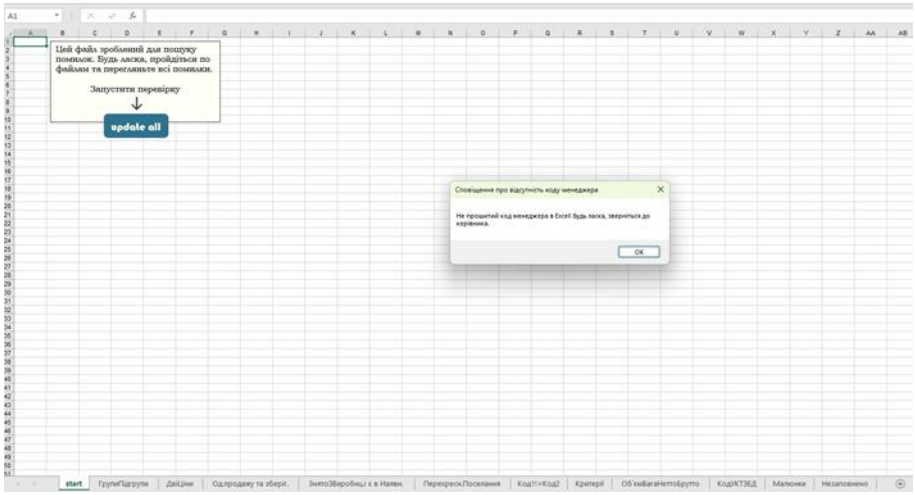


Рис. 3.31 Повідомлення при відсутності коду менеджера в PERSONAL.XLSB

Другий сценарій – недоступність сервера. Тест відтворювався шляхом відключення мережевого доступу до sql_2. При спробі запуску система виконала ping-перевірку, отримала негативний результат і одразу вивела повідомлення: «Сервер sql_2 не працює». Виконання перервалось до будь-якої спроби відкрити ADO-з'єднання, Excel не завис. Повідомлення можна побачити на рис. 3.32.

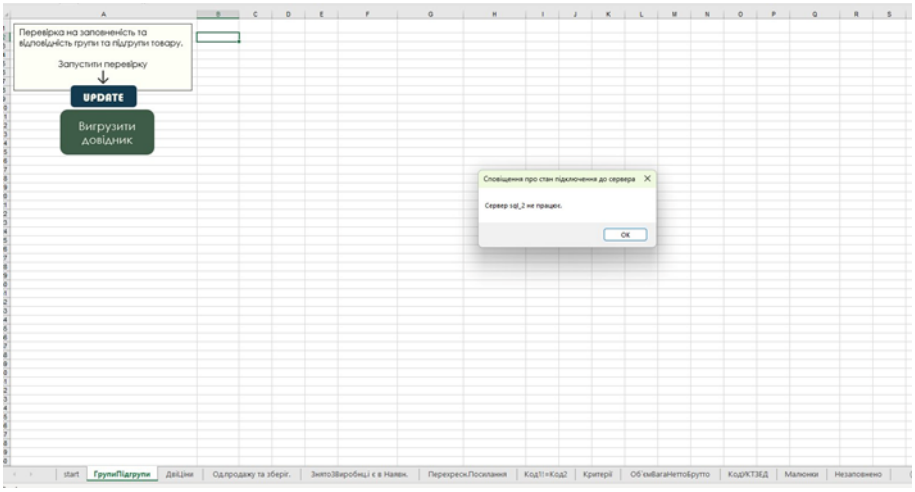


Рис. 3.32 Повідомлення при недоступності сервера sql_2

Результати тестування підтвердили відповідність системи всім визначеним вимогам. З функціонального боку: перевірки коректно повертають як позитивні результати (відсутність помилок з явним підтвердженням), так і негативні (конкретні переліки проблемних записів); механізм довідників дозволяє виправляти помилки безпосередньо в середовищі аудиту; персоналізація надійно блокує запуск без налаштованого коду менеджера.

З нефункціонального боку ключовим результатом є час повного аудиту, який за кілька тестувань не перевищував 3 хвилин 48 секунд для одинадцяти перевірок на реальному виробничому асортименті, що підтверджує виконання вимоги продуктивності. Обробка позаштатних ситуацій також відповідає вимогам надійності: в обох тестових сценаріях система завершила сеанс коректно, з інформативним повідомленням і без негативних наслідків для даних. Сукупність отриманих результатів свідчить про практичну готовність системи до регулярного використання в робочому процесі аналітичного відділу.

ВИСНОВКИ

Результатом виконання дипломної роботи став персоналізований інструмент самоконтролю якості товарних даних у корпоративній ERP-системі Microsoft Dynamics NAV, реалізований у вигляді макросної Excel-книги формату .xlsb з прямим підключенням до реляційної бази даних через ADO-з'єднання. Інструмент дозволяє кожному аналітику самостійно виявляти логічні помилки у власних товарних даних у режимі реального часу без залучення IT-підрозділу та без модифікації ERP-системи. У ході розробки були вирішені всі поставлені задачі.

У першому розділі досліджено теоретичні засади управління якістю даних у корпоративних інформаційних системах, визначено ключові характеристики якості даних та обґрунтовано критичність їх забезпечення в ERP-середовищі. Також проаналізовано методології валідації даних, а саме профілювання, DAQUA-MASS та валідацію структурної цілісності, і визначено їхню роль як методологічної основи для розробленого інструменту.

У другому розділі оцінено існуючі рішення контролю якості даних: вбудовану валідацію Dynamics NAV, Microsoft SQL Server Data Quality Services, фреймворк Great Expectations та платформу Talend Data Fabric. Визначено ключові обмеження кожного з них – відсутність перехресних перевірок бізнес-логіки, потреба в технічній інфраструктурі, висока вартість або відсутність персоналізації, та обґрунтовано потребу у розробці власного інструменту. Окрім цього, класифіковано типові помилки у товарних даних аналітичного відділу за трьома групами: комерційно-критичні, помилки логічної узгодженості атрибутів та помилки повноти й відповідності стандартам, а також визначено перелік перевірок, необхідних для їх виявлення.

У третьому розділі спроєктовано трирівневу архітектуру системи, що складається з презентаційного шару на базі Excel, шару бізнес-логіки у вигляді VBA-модулів та шару даних, представленого двома SQL-серверами Dynamics NAV з розмежованими зонами відповідальності. Реалізовано механізм персоналізованої фільтрації: інструмент автоматично зчитує код менеджера з особистої книги

макросів PERSONAL.XLSB та використовує його як параметр у всіх SQL-запитах, завдяки чому кожен аналітик отримує вибірку виключно своїх помилок без ручного сортування. Розроблено та впроваджено повний набір перевірок бізнес-логіки, що виходять за межі стандартного функціоналу NAV, а саме виявлення дублів активних цін, відсутності зображень та кодів постачальника, порушень ієрархії груп і підгруп, логічних невідповідностей фізичних параметрів та некоректних значень критеріїв. Забезпечено розгортання інструменту через мережевий шлях до спільного файлу на сервері з прописуванням у каталозі автозавантаження Excel, що виключає необхідність локального встановлення та гарантує автоматичне оновлення для всіх користувачів.

Інструмент успішно впроваджено в операційну діяльність реального аналітичного відділу дистриб'юторської компанії, що підтверджує його практичну ефективність та можливість відтворення в аналогічних корпоративних середовищах. Цілі, поставлені на початку кваліфікаційної роботи, досягнуто повністю. Розроблений інструмент забезпечує глибокий контроль якості товарних даних, є безкоштовним у розгортанні, легко адаптується під внутрішні бізнес-правила підприємства та готовий до подальшого розширення переліку перевірок.

ПЕРЕЛІК ПОСИЛАНЬ

1. Bradford D. M. Modern ERP: select, implement, and use today's advanced business systems. Marianne Bradford, 2020. 289 с.
2. International D. DAMA-DMBOK: data management body of knowledge. Technics Publications, 2020. 585 с.
3. Panorama Consulting Group. 2025 ERP Report. Panorama Consulting, 2025. URL: https://www.panorama-consulting.com/resource-center/erp-report/ (дата звернення: 25.02.2026)
4. Ladley J. The data governance business case. Data governance. 2020. С. 51–60. URL: https://doi.org/10.1016/B978-0-12-815831-9.00005-9 (дата звернення: 11.03.2026).
5. Khatri V., Brown C. V. Designing data governance. Communications of the ACM. 2010. Т. 53, № 1. С. 148–152. URL: https://doi.org/10.1145/1629175.1629210 (дата звернення: 11.03.2026).
6. Scannapieco M., Batini C. Data quality: concepts, methodologies and techniques. Springer London, Limited, 2006.
7. King T., Schwarzenbach J. Managing data quality: a practical guide. BCS Learning & Development Limited, 2020. 150 с.
8. Gaur C. Data Quality Management and its Best Practices. XenonStack - AI Reasoning Foundry for Agentic Enterprises. URL: https://www.xenonstack.com/insights/data-quality-management (дата звернення: 19.03.2026).
9. McKinsey Global Institute. The Age of Analytics: Competing in a Data-Driven World. McKinsey & Company, 2016. URL: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-age-of-analytics-competing-in-a-data-driven-world (дата звернення: 06.03.2026)
10. Hustad E., Olsen D. H. ERP Post-Implementation. Sociotechnical Enterprise Information Systems Design and Integration. 2013. С. 72–85. URL: https://doi.org/10.4018/978-1-4666-3664-4.ch005 (дата звернення: 07.03.2026).

11. Types of Enterprise Data (Transactional, Analytical, Master). *BI / DW Insider*. URL: https://bi-insider.com/posts/types-of-enterprise-data-transactional-analytical-master/ (дата звернення: 24.03.2026).

12. Ponniah P. Data Warehousing Fundamentals for IT Professionals. 2010. URL: https://doi.org/10.1002/9780470604137 (дата звернення: 26.03.2026).

13. Kimball R., Ross M. The Data Warehouse Toolkit : 3rd ed. 2021. 600 с.

14. Ehrlinger L., Wöß W. A Survey of Data Quality Measurement and Monitoring Tools. Frontiers in Big Data. 2022. Т. 5. URL: https://doi.org/10.3389/fdata.2022.850611 (дата звернення: 28.03.2026).

15. Bernardo B. M. V. та ін. Data governance & quality management–Innovation and breakthroughs across different fields. Journal of Innovation & Knowledge. 2024. Т. 9, № 4. С. 100598. URL: https://doi.org/10.1016/j.jik.2024.100598 (дата звернення: 03.04.2026).

16. Nivedita Singh. Data Profiling. LinkedIn. URL: https://www.linkedin.com/pulse/data-profiling-nivedita-singh (дата звернення: 07.04.2026).

17. Case study - Enhanced data quality | NHSBSA. URL: https://www.nhsbsa.nhs.uk/case-study-enhanced-data-quality (дата звернення: 07.04.2026).

18. ISO - International Organization for Standardization. URL: https://www.iso.org/obp/ui/es/#iso:std:iso:8000:-61:ed-1:v1:en (дата звернення: 07.04.2026).

19. Robinson S., Sheldon R., Stedman C. What is Data Quality and Why is it Important? | Definition from TechTarget. Search Data Management. URL: https://www.techtarget.com/searchdatamanagement/definition/data-quality (дата звернення: 14.04.2026).

20. Touil Y. Architectural Evolution of Microsoft ERP Systems: From Monolithic Dynamics NAV to Event-Driven Business Central Extensions. 2026. URL: https://www.researchgate.net/publication/401975724_Architectural_Evolution_of_Microsoft_ERP_Systems_From_Monolithic_Dynamics_NAV_to_Event-Driven_Business_Central_Extensions (дата звернення: 20.04.2026).

21. Laranjeiro N., Soydemir S. N., Bernardino J. A Survey on Data Quality: Classifying Poor Data. PRDC 2015 : The 21st IEEE Pacific Rim International Symposium on Dependable Computing. 2015. P. 179–188. DOI: 10.1109/PRDC.2015.41 (дата звернення: 28.04.2026).

22. 8 Core Dimensions of Data Quality: A Guide to Data Excellence - SixSigma.us. SixSigma.us. URL: https://www.6sigma.us/six-sigma-in-focus/dimensions-of-data-quality/ (дата звернення: 28.04.2026).

23. Krantz T., Jonker A. The True Cost of Poor Data Quality | IBM. IBM. URL: https://www.ibm.com/think/insights/cost-of-poor-data-quality (дата звернення: 28.04.2026).

24. Data Quality: The Impact of Poor Data Quality. The Agile Data (AD) Method - Strategies for effective data-oriented development. URL: https://agiledata.org/essays/impact-of-poor-data-quality.html (дата звернення: 02.05.2026).

25. OnValidate (Page Field) trigger - Business Central. Microsoft Learn: Build with answers in reach. URL: https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/triggers-auto/pagefield/devenv-onvalidate-pagefield-trigger (дата звернення: 02.05.2026).

26. Introduction to Data Quality Services - Data Quality Services (DQS). Microsoft Learn: Build with answers in reach. URL: https://learn.microsoft.com/en-us/sql/data-quality-services/introduction-to-data-quality-services?view=sql-server-ver16 (дата звернення: 08.05.2026).

27. Discontinued Database Engine Functionality - SQL Server. Microsoft Learn: Build with answers in reach. URL: https://learn.microsoft.com/en-us/sql/database-engine/discontinued-database-engine-functionality-in-sql-server?view=sql-server-ver17 (дата звернення: 10.05.2026).

28. Try GX Core | Great Expectations. Home | Great Expectations. URL: https://docs.greatexpectations.io/docs/core/introduction/try_gx/ (дата звернення: 14.05.2026).

29. Qlik Talend Cloud | Trusted, AI-Ready Data Integration & Quality. Qlik. URL: https://ua.talend.com/about-us/press-releases/qlik-completes-acquisition-of-talend (дата звернення: 14.05.2026).

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (ПРЕЗЕНТАЦІЯ)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра Інформаційних систем та технологій

Кваліфікаційна робота
на тему: **“Система контролю цілісності та актуальності бази даних у корпоративних ERP-системах”**

Виконала: здобувачка вищої освіти групи ІСД-42
Дарія Ємельянова
Керівник роботи: доктор філософії Віктор Сагайдак

Мета та актуальність роботи

Мета роботи
Розробка персоналізованого інструменту самоконтролю якості даних у корпоративній ERP-системі без залучення IT-підрозділу та без модифікації системи

Об'єкт дослідження
Процеси контролю цілісності та актуальності товарних даних у корпоративних ERP-системах

Предмет дослідження
Система валідації товарних даних у корпоративній ERP-системі Microsoft Dynamics NAV

Актуальність роботи

Актуальність роботи обумовлена відсутністю на ринку безкоштовного інструменту контролю якості товарних даних у ERP-системах, який не потребує IT-спеціалістів ні для впровадження, ні для щоденного використання - на відміну від існуючих рішень, що або технічно обмежені, або коштують десятки тисяч доларів на рік. Додаткову цінність забезпечує персоналізація: кожен аналітик автоматично бачить лише власні помилки без ручного фільтрування, що пришвидшує роботу та знімає психологічний тиск від публічності проблемних даних у колективі.

Завдання кваліфікаційної роботи

- Дослідити засади управління якістю даних у корпоративних інформаційних системах та методології валідації (Data Profiling, DAQUA-MASS, структурна цілісність)
- Проаналізувати якість товарних даних у Dynamics NAV, класифікувати помилки та порівняти існуючі інструменти контролю якості
- Спроектувати та реалізувати персоналізований інструмент контролю цілісності та актуальності даних у корпоративній ERP-системі

Класифікація помилок у товарних даних ERP

Комерційно-критичні

- Дублювання активних цін (невидимо в інтерфейсі NAV);
- Відсутність фото товару - знижує конверсію B2B;
- Відсутність коду постачальника (Vendor) - блокує автоматичні; замовлення

Структурно-логічні

- Порушення ієрархії Категорія→Група→Підгрупа (ризик митного оформлення);
- Невідповідність опису та коду УКТЗЕД;
- Маса нетто > маса брутто або від'ємний об'єм;
- Розбіжність одиниць виміру картки та цінового запису;

Повнота та стандарти

- Невалідні крос-посилання (OE-номери);
- Некоректні значення критеріїв - вільний текст замість довідника;
- Заблоковані товари з ненульовими складськими залишками;

Аналіз існуючих рішень

Критерій	Вбудована валідація Nav	MS SQL DQS	Great Expectations	Talend Data Fabric	Розроблений інструмент
Перехресні перевірки між таблицями	Обмежено	Частково	Так	Так	Так
Перевірки бізнес-логіки	Ні	Частково	Так	Так	Так
Вплив на продуктивність ERP	Високий	Середній	Відсутній	Відсутній	Відсутній
Доступність для менеджера без IT-підтримки	Так	Ні	Ні	Ні	Так
Нативна інтеграція з робочим середовищем Excel	Відсутня	Відсутня	Відсутня	Відсутня	Так
Персоналізація за зоною відповідальності	Відсутня	Відсутня	Відсутня	Відсутня	Так
Перспективність (стратегічна підтримка)	Обмежена (legacy)	Знята (SQL 2025)	Активна	Активна	Активна
Вартість впровадження	Включена в NAV	Включена в SQL Server	Безкоштовно	Близько 45000$/рік	Безкоштовно

Вимоги до системи

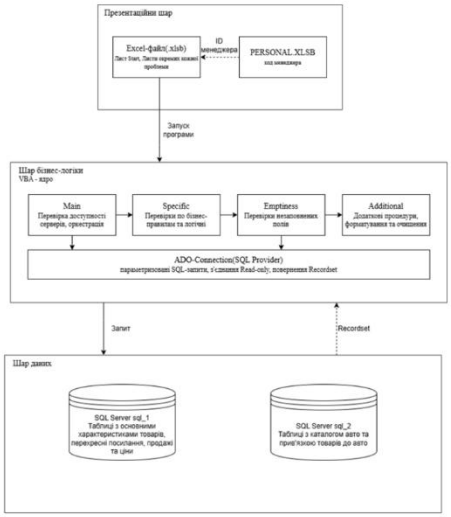
Функціональні:

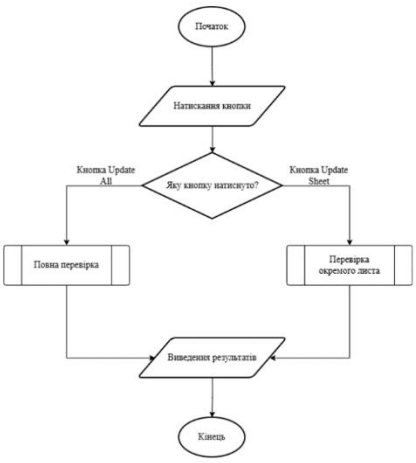
- Персоналізація аудиту
- Модульна архітектура
- Автоматизоване представлення даних

Нефункціональні:

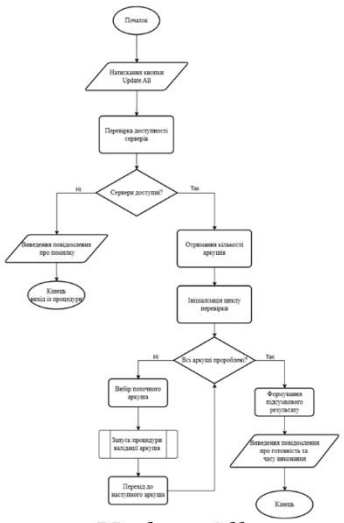
- Продуктивність.
- Надійність з'єднання та обробка виключень
- Локальна інтеграція

Архітектура та загальна логіка





Блок-схеми Update All та Update Sheet



Update All

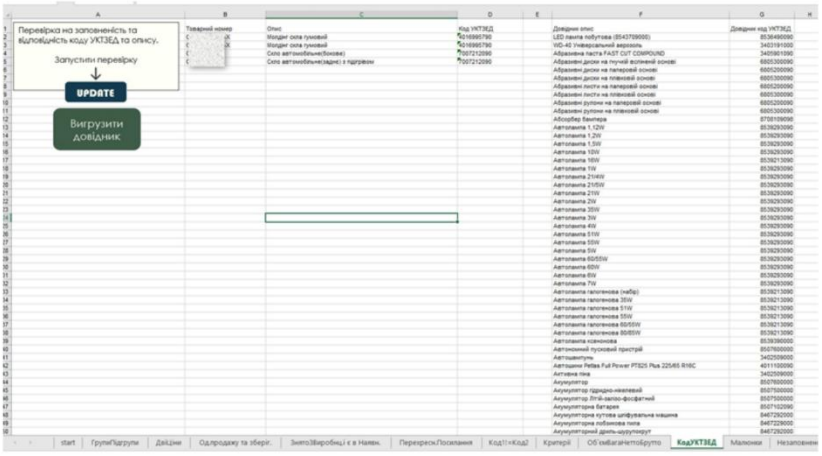
Update Sheet

Результат повного аудиту

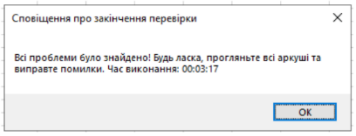
Помилок не було знайдено

Помилки були знайдені

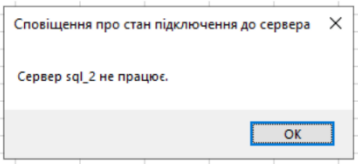
Результат локальної перевірки та вигрузки довідника



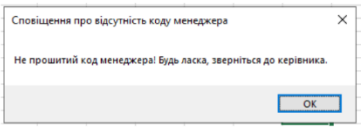
Сповіщення користувача



Закінчення повного аудиту



Сервер не працює



Відсутній код менеджера

Оцінка ефективності

00:03:17	1236	4	0
Час виконання повного аудиту	Знайдених артикулів без коду постачальника	Знайдених артикулів з некоректним кодом УКТЗЕД	Порушень ієрархії

Відповідність вимогам

- Продуктивність: протягом тестування час повного аудиту не перевищував 3 хв 48 с (вимога: до 5 хв)
- Надійність: ADO Read-only з'єднання, параметризовані запити, захист від некоректного запуску
- Персоналізація: кожен менеджер бачить виключно власні помилки без ручного фільтрування
- Модульна архітектура: 4 незалежні VBA-модулі: Main, Specific, Emptiness, Additional
- Автоматизоване представлення даних: CopyFromRecordset заповнює аркуші автоматично, форматування через Additional
- Локальна інтеграція: інтерфейс є файлом MS EXCEL, знаходиться на корпоративному сервері, доступ через панель швидкого доступу

Апробація результатів

VII Міжнародної науково-технічної конференції «Сучасний стан та перспективи розвитку IoT» (Київ, ДУІКТ, 20 квітня 2026 р.). Тези доповіді на тему «МЕТОДОЛОГІЯ АЛГОРИТМІЧНОГО УСУНЕННЯ АНОМАЛІЙ ДАНИХ У КОРПОРАТИВНИХ ERP-СИСТЕМАХ (НА ПРИКЛАДІ MICROSOFT DYNAMICS NAV)» опубліковані у збірнику матеріалів конференції (стор. 274).

VII Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях» (Київ, ДУІКТ, 23 квітня 2026 р.). Тези доповіді на тему «ВПЛИВ ПЕРСОНАЛІЗАЦІЇ КОНТРОЛЬНИХ ФУНКЦІЙ НА МІНІМІЗАЦІЮ ІНФОРМАЦІЙНОЇ ЕНТРОПІЇ В КОРПОРАТИВНИХ БАЗАХ ДАНИХ» опубліковані у збірнику матеріалів конференції (стор. 441).

Висновки

Класифіковано типові помилки в товарних даних ERP за 3 групами; проведено порівняльний аналіз 4 наявних інструментів - підтверджено прогалину у ринкових рішеннях

Спроєктовано трирівневу архітектуру (Excel + VBA + SQL Server) та реалізовано механізм персоналізованої фільтрації через PERSONAL.XLSB

Інструмент успішно впроваджено в реальному аналітичному відділі: час повного аудиту - до 3 хв 48 с, виявлено 1 236+ помилок незаповнених полів

Реалізовано 11 перевірок бізнес-логіки та модуль контролю незаповнених полів

Дякую за увагу!