Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____*Комп'ютерних наук*_____
(повна назва)
Кафедра_____*Системотехніки*_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____

Дослідження варіантів реалізації алгоритмів майнінгу
для рекомендаційної е-системи
(тема)

Виконав:
студент _2__ курсу, групи _*ІТПм-22-1*
_____*Гуляєв Д.С.*_____
(прізвище, ініціали)

Спеціальність *122 Комп'ютерні науки*____
(код і повна назва спеціальності)
Тип програми __освітньо-професійна_____
Освітня програма _*Інформаційні технології проектування*_____
(повна назва освітньої програми)

Керівник ___*доцент каф. СТ Коваленко А.І.*__
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ __*Гребеннік І.В.*__
(підпис) (прізвище, ініціали)

2024 р.

Я як студент ХНУРЕ розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

(дата, підпис, прізвище ін. студента)

19.01.2024 Гуляєв Д.С.

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведено 19 січня 2024 р.

Керівник кваліфікаційної роботи доц. каф. СТ Коваленко А.І.

Харківський національний університет радіоелектроніки

Факультет_____*Комп'ютерних наук*_____

Кафедра_____*Системотехніки*_____

Рівень вищої освіти_____другий (магістерський)_____

Спеціальність_____*122 Комп'ютерні науки*_____
(код і повна назва)

Тип програми _____освітньо-професійна_____

Освітня програма_____*Інформаційні технології проектування*_____
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри *СТ проф. Гребеннік І.В*

_____

(підпис)

«___»_____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові _____*Гуляєву Дмитру Сергійовичу*_____
(прізвище, ім'я, по батькові)

1. Тема роботи: *Дослідження варіантів реалізації алгоритмів майнінгу для рекомендаційної е-системи*
затверджена наказом по університету від *20.11* 2023 р. №*1373Ст*
2. Термін подання студентом роботи до екзаменаційної комісії: 19.01.2024 р
3.Вихідні дані до роботи: *Провести дослідження варіантів реалізації алгоритмів майнінгу для рекомендаційної с-системи*. Програмне забезпечення: *ОС macOS Monterey та вище. Технічне забезпечення: MacBook Pro 15 2017 та вище. СУБД MySQL-server версія 8.0 або вище*
4. Перелік питань, що потрібно опрацювати в роботі: *Вступ Огляд особливостей використання алгоритмів мінінгу в рекомендаційних системах. Аналіз методів мінінгу у системах електронної комерції. Аналіз варіантів реалізації алгоритмів мінінгу на платформі СУБД MySQL. Реалізація алгоритмів мінінгу на стороні клієнта. Висновки.*
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій: _____
6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
	доцент каф. СТ Коваленко А. І.		
	доцент каф. СТ Коваленко А. І.		
	доцент каф. СТ Коваленко А. І.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	*Отримання завдання на виконання роботи*		Виконано
2	*Огляд особливостей використання алгоритмів мінінгу*		Виконано
3	*Аналіз методів мінінгу у системах електронної комерції*		Виконано
4	*Огляд літератури*		Виконано
5	*Реалізація алгоритмів мінінгу на платформі СУБД MySQL*		Виконано
6	*Аналіз варіантів реалізації алгоритмів мінінгу*		Виконано
7	*Реалізація алгоритмів мінінгу на стороні клієнта*		Виконано
8	*Оформлення пояснювальної записки*		Виконано
9	*Представлення на рецензування*		Виконано

Дата видачі завдання 16.10.2023 р.

Студент _____ *Гуляєв Д.С.*
(підпис) (прізвище, ініціали)

Керівник роботи _____ *доцент каф. СТ Коваленко А.І.*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Кваліфікаційна робота магістра містить: 57 с., 19 табл., 9 рис., 2 додатки, 1 джерело інформації.

АЛГОРИТМ МІНІНГУ, СУБД, ІНФОРМАЦІЙНА СИСТЕМА, ПОРІВНЯЛЬНИЙ АНАЛІЗ, ЕЛЕКТРОННА КОМЕРЦІЯ, DATA MINING

Об'єктом дослідження цієї роботи є алгоритми мінінгу для пошуку асоціацій.

Предметом дослідження є процес реалізації та аналізу алгоритмів мінінгу для пошуку асоціацій на платформі СУБД MySQL.

Метою дослідження є аналіз реалізації алгоритмів мінінгу для пошуку асоціацій для ефективного застосування їх в інформаційній системі електронної комерції.

Методи дослідження включають системний підхід, структурний аналіз, моделювання реляційних баз даних для інформаційної системи електронної. комерції, а також застосування алгоритмів мінінгу мінінгу для пошуку асоціацій на платформі СУБД MySQL для оптимізації пошуку асоціацій у контексті електронної комерції.

Робота включає теоретичний аналіз алгоритмів мінінгу, встановлення параметрів та критеріїв для їхньої оцінки та порівняння. Також передбачено проведення експериментального дослідження для перевірки реалізованих алгоритмів мінінгу для пошуку асоціацій.

Сфера застосування дослідження розширюється до практичного впровадження алгоритмів мінінгу с в реальних проектах електронної комерції.

ABSTRACT

Master's qualification work contains: 57 p., 19 tables, 9 figures,2 appendices, 1 source of information.

MINING ALGORITHM, SUBD, INFORMATION SYSTEM, COMPARATIVE ANALYSIS, E-COMMERCE, DATA MINING

The object of study of this work is the mining algorithms for finding associations.

The subject of the study is the process of implementing and analysing mining algorithms for finding associations on the MySQL database platform.

The aim of the study is to analyse the implementation of mining algorithms for finding associations for their effective application in an e-commerce information system.

The research methods include a systematic approach, structural analysis, modelling of relational databases for an e-commerce information system, as well as the application of association mining algorithms on the MySQL database platform to optimise association search in the context of e-commerce.

The work includes a theoretical analysis of mining algorithms, setting parameters and criteria for their evaluation and comparison. It also includes an experimental study to test the implemented mining algorithms for association retrieval.

The scope of the research extends to the practical implementation of c-mining algorithms in real e-commerce projects.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 8
ВСТУП 9
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ З ВИКОРИСТАННЯ АЛГОРИТМІВ МІНІНГУ 11
1.1 Особливості використання алгоритмів мінінгу в рекомендаційних системах 11
1.2 Огляд і аналіз сучасного стану реалізації алгоритмів мінінгу на різних платформах СУБД 14
1.3 Постановка задачі 17
2.1 Призначення алгоритмів мінінгу 18
2.2 Алгоритм AIS 19
2.3 Алгоритм SETM 22
2.4 Алгоритм Apriori 25
2.5 Висновок за аналізом алгоритмів мінінгу 31
3 АНАЛІЗ ВАРІАНТІВ РЕАЛІЗАЦІЇ АЛГОРИТМІВ МІНІНГУ НА ПЛАТФОРМІ СУБД MYSQL 32
3.1 Вибір предметної області для дослідження 32
3.2 Розробка бази даних системи е-комерції з продажу ліків 33
3.3 Аналіз реалізації алгоритму SETM для системи е-комерції з продажу ліків 48
3.4 Аналіз реалізації алгоритму AIS для системи е-комерції з продажу ліків 60
3.5 Порівняльний аналіз алгоритмів мінінгу для системи е-комерції з продажу ліків 67
ВИСНОВКИ 69
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 70
Додаток А Графічний матеріал кваліфікаційної роботи ..... **Ошибка! Закладка не определена.**
Додаток Б Сертифікат участі у конференції ..... **Ошибка! Закладка не определена.**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних;

ІС – інформаційна система;

СУБД – система керування базами даних;

Data Mining – інтелектуальний аналіз;

SQL (Structured Query Language) – мова структурованих запитів.

ВСТУП

З розвитком систем електронної комерції поширюється реалізація їх рекомендаційних функцій. Розуміння та застосування алгоритмів мінінгу, де величезні обсяги інформації переповнюють наше повсякденне життя, залишається суттєвим викликом в цифровому світі. Зокрема, у залежності від обраного товару, формується пропозиція для покупки супутніх для нього товарів. Ця рекомендаційна функція реалізується за допомогою методів Data Mining для пошуку асоціативних правил.

З вдосконаленням інфраструктури електронної комерції та розширенням її функціональності, використання алгоритмів мінінгу стає ще більш актуальним. Системи рекомендацій виробляють персоналізовані пропозиції, спрямовані на задоволення індивідуальних потреб кожного клієнта. В умовах надлишку інформації виникає необхідність вдосконалення алгоритмів, які визначають зв'язки між різними товарами та виводять з них змістовні рекомендації. Таким чином, використання методів Data Mining стає невід'ємною складовою цифрового магазину, допомагаючи покупцям знаходити не лише обрані товари, але й додаткові продукти, які можуть виявитися цікавими або важливими для них.

На ринку програмного забезпечення СУБД на сьогоднішній день тільки частина з них має ресурс для використання алгоритмів майнінгу для пошуку асоціацій, наприклад:

– на платформі СУБД Microsoft SQL Server реалізовано середовище SQL SSDT (Server Data Tools) і служба Analysis Services [1]. З використанням служби Analysis Services користувач отримує можливість використання алгоритму пошуку асоціацій «Microsoft»;

– на платформі СУБД Oracle Database [2] реалізована служба Oracle Data Mining Suite, що дозволяє, використовуючи дані БД, отримати вибірку (покриття) для заданих умов пошуку асоціацій.

Тому є практичний інтерес з розгляду особливостей реалізації алгоритмів мінінгу для пошуку асоціацій на платформі СУБД MySQL-server.

Ця дослідницька робота має за мету проаналізувати та реалізувати алгоритми мінінгу, спрямовані на оптимізацію та підвищення ефективності інформаційних систем електронної комерції. Глибокий аналіз цих алгоритмів та їх подальша реалізація розглядаються як ключові етапи, що сприятимуть розкриттю оптимальних підходів для ефективного пошуку асоціацій та здійснення рекомендацій.

Об'єктом дослідження цієї роботи є алгоритми мінінгу для пошуку асоціацій в інформаційних системах електронної комерції. В рамках даного об'єкта дослідження буде проведено більш детальний аналіз варіантів реалізації алгоритмів, спрямований на виявлення та оптимізацію їхніх можливостей у контексті пошуку асоціацій.

Предметом дослідження є процес реалізації та аналізу алгоритмів мінінгу для пошуку асоціацій на платформі СУБД MySQL. Докладне розглядання цього процесу спрямоване на визначення його оптимальності та можливостей у контексті використання даної бази даних.

Результати досліджень, які будуть отримані, мають на меті сприяти покращенню якості рекомендацій завдяки глибокому аналізу та оптимізації алгоритмів мінінгу. Дослідження в області алгоритмів мінінгу для інформаційних систем електронної комерції відіграє важливу роль у вдосконаленні цифрового середовища та розширенні можливостей електронної комерції.

Цей проект прагне сприяти розвитку більш ефективних та інноваційних рекомендаційних систем, відповідних сучасним потребам користувачів у всесвіті електронної комерції.

Результати цієї наукової роботи були представлені та опубліковані у міжнародній науково-практичній конференції «Distance education as the main problem of young people» IST-2023, що свідчить про визнання та цінність дослідження в академічній спільноті.

НАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ З ВИКОРИСТАННЯ АЛГОРИТМІВ МІНІНГУ

1.1 Особливості використання алгоритмів мінінгу в рекомендаційних системах

У сучасному цифровому світі електронні системи інформаційного обміну стають все більш важливими для задоволення різноманітних потреб користувачів. Рекомендаційні електронні системи виступають однією з ключових складових цього процесу, сприяючи покращенню користувацького досвіду та ефективності взаємодії з електронними платформами.

Алгоритми мінінгу даних знаходять широке застосування у різних сферах та індустріях. Наприклад:

– електронна комерція: В онлайн-торгівлі, алгоритми мінінгу використовуються для рекомендацій товарів, наприклад, коли сервіси, подібні до Amazon або Netflix, пропонують покупцям товари або фільми на основі їхніх попередніх покупок чи переглядів;

– маркетинг та реклама: У цій галузі алгоритми мінінгу використовуються для визначення цільової аудиторії та оптимізації рекламних кампаній. Наприклад, рекламні компанії використовують алгоритми для персоналізації рекламних оголошень на основі інтересів користувачів, показуючи їм релевантну рекламу;

– фінанси: У фінансовій сфері алгоритми мінінгу даних використовуються для прогнозування ринкових тенденцій та аналізу ризиків. Наприклад, банки використовують їх для виявлення шахрайства чи прогнозування змін курсів валют;

– охорона здоров'я: У медичній галузі алгоритми мінінгу даних застосовуються для аналізу клінічних даних та виявлення закономірностей у лікуванні захворювань, наприклад, для розробки нових методів діагностики чи лікування;

– телекомунікації: У цій галузі алгоритми мінінгу використовуються для аналізу даних про використання мережі, щоб оптимізувати мережеві ресурси та прогнозувати пікове навантаження;

– наука та дослідження: У наукових дослідженнях, алгоритми мінінгу використовуються для аналізу великих обсягів даних, щоб виявляти нові закономірності або робити прогнози у різних наукових дисциплінах;

– соціальні мережі: У соціальних мережах, алгоритми мінінгу даних використовуються для аналізу поведінки користувачів та рекомендацій контенту на основі їхніх інтересів.

Ці приклади демонструють широкий спектр використання алгоритмів мінінгу в різних галузях, які сприяють оптимізації та покращенню процесів у різних сферах діяльності.

Проблема полягає в тому, що існуючі рекомендаційні електронні системи часто стикаються з викликами, пов'язаними з обробкою великої кількості різноманітних даних, таких як інформація про користувачів, їхні вибори та попередні дії. Крім того, зростаюча складність та різноманітність контенту потребують постійного удосконалення алгоритмів мінінгу для забезпечення точних та значущих рекомендацій.

Вирішення цієї проблеми передбачає глибоке дослідження та аналіз різних варіантів реалізації алгоритмів мінінгу для рекомендаційних електронних систем. Це включає вивчення сучасних тенденцій у галузі мінінгу даних, визначення оптимальних стратегій аналізу клієнтської поведінки та розробку алгоритмів, які забезпечують високу якість рекомендацій при оптимальному використанні ресурсів.

Суть використання алгоритмів мінінгу полягає у визначенні наборів об'єктів, що часто зустрічаються, у великій множині таких наборів. Це завдання є окремим випадком завдання класифікації. Спочатку її розв'язували під час аналізу тенденцій у поведінці покупців у супермаркетах. Аналізу піддавалися дані про здійснювані ними покупки, які покупці складають у візок (кошик). Це послужило причиною другої назви, що часто зустрічається – аналіз ринкових кошиків (Basket Analysis).

Під час аналізу цих даних інтерес насамперед представляє інформація про те, які товари купують разом, у якій послідовності, яким категоріям споживачів яким товарам надають перевагу, у які періоди часу тощо. Така інформація дає змогу ефективніше планувати закупівлю товарів, проведення рекламної кампанії тощо. Наприклад, із набору покупок, що здійснюються в магазині, можна виокремити такі набори товарів, які купують разом: {чіпси, пиво}; {картопля, цибуля}. Отже, можна зробити висновок, що якщо купуються чіпси або цибуля, то, як правило, купуються і пиво або картопля відповідно. Володіючи такими знаннями, можна розмістити ці товари поруч, об'єднати їх в один пакет зі знижкою або здійснити інші дії, що стимулюють покупця придбати товар.

Завдання пошуку асоціативних правил актуальне не тільки у сфері торгівлі. Наприклад, у сфері обслуговування інтерес становить інформація про те, якими послугами клієнти воліють користуватися в сукупності. Для отримання цієї інформації завдання розв'язують стосовно даних про послуги, якими користується один клієнт протягом певного часу (місяця, року). Це допомагає визначити, наприклад, як найвигідніше скласти пакети послуг, пропонованих клієнту.

У медицині аналізу можуть піддаватися симптоми і хвороби, що спостерігаються у пацієнтів. У цьому разі знання про те, які поєднання хвороби і симптомів зустрічаються найчастіше, допомагають у майбутньому правильно ставити діагноз.

Під час аналізу часто викликає інтерес послідовність подій, що відбуваються. У разі виявлення закономірностей у таких послідовностях можна з певною часткою ймовірності передбачати появу подій у майбутньому, що дає змогу ухвалювати правильніші рішення. Таке завдання є різновидом завдання пошуку асоціативних правил і називається сиквенціальним аналізом.

Основною відмінністю завдання сиквенціального аналізу від пошуку асоціативних правил є встановлення відношення порядку між досліджуваними наборами. Це відношення може бути визначено різними способами. Під час аналізу послідовності подій, що відбуваються в часі, об'єктами таких наборів є події, а відношення порядку відповідає хронології їхньої появи.

1.2 Огляд і аналіз сучасного стану реалізації алгоритмів мінінгу на різних платформах СУБД

У світі, де обсяги даних зростають експоненційно, використання СУБД для реалізації алгоритмів мінінгу набуває особливої вагомості, і це стає надзвичайно актуально в умовах постійного збільшення обсягів інформації. Забезпечення ефективної обробки та аналізу даних в цьому контексті є ключовим завданням для підтримки рекомендаційних електронних систем. Важливо провести докладний аналіз сучасного стану реалізації алгоритмів мінінгу на різних платформах СУБД, оскільки це сприятиме оптимізації та вибору найефективніших підходів.

Наприклад, можна розглянути сценарій, де супермаркет використовує алгоритми мінінгу для аналізу покупок клієнтів, визначаючи, які товари часто купують разом. Ця інформація може бути використана для вдосконалення стратегій маркетингу та створення ефективних рекомендацій для покупців. Процес виявлення асоціацій, показаний на рисунку 1, демонструє, як дані з різних сховищ, таких як сховища даних, база даних, інформаційні сховища, реляційна база даних тощо, можуть використовуватися для створення цінної інформації через різноманітні операції, включаючи очищення даних, інтеграцію та перетворення.

Починаючи з розгляду платформ, таких як MySQL, PostgreSQL, MongoDB, тощо, важливо визначити, які можливості та функції кожна з них пропонує для оптимальної реалізації алгоритмів мінінгу. Наприклад, MySQL славиться своєю швидкодією та легкістю використання, PostgreSQL відзначається розширюваністю та стабільністю, а MongoDB, використовуючи концепцію NoSQL, забезпечує гнучкість у роботі з неструктурованими даними.

Дослідження сучасного стану також повинно охоплювати вивчення відкритих джерел та інструментів, які можуть служити підтримкою для реалізації алгоритмів мінінгу в різних СУБД. З урахуванням аспектів, таких як швидкодія, масштабованість, підтримка складних запитів та обробка великих обсягів даних, важливо обирати оптимальні рішення, які відповідають конкретним потребам рекомендаційної системи.

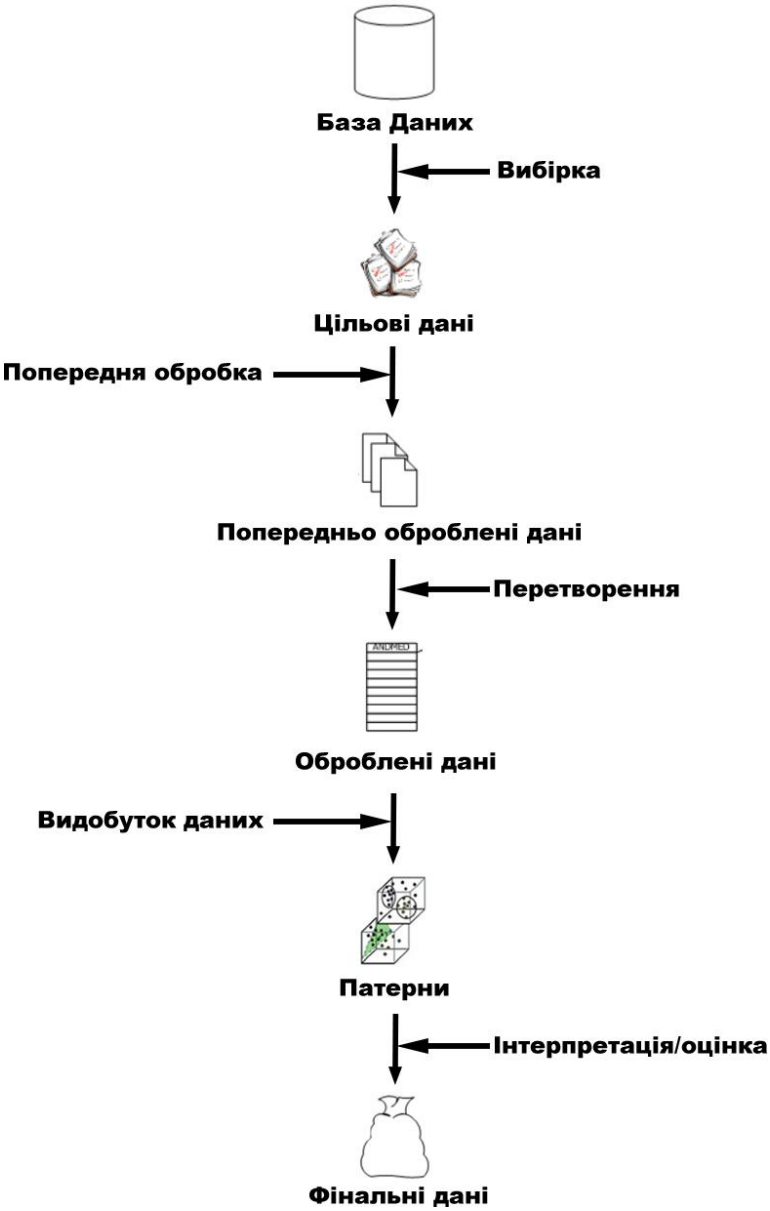

Рис. 1 – Процес виявлення асоціацій у базі даних

Підкреслюючи розмаїття архітектурних підходів, важливо встановити переваги та обмеження кожної платформи СУБД в контексті алгоритмів мінінгу. Наприклад, реляційні бази даних можуть виявитися ефективними для деяких аспектів мінінгу, тоді як NoSQL бази даних можуть стати більш підходящими для обробки великих обсягів неструктурованих даних. В статті «Fast Algorithms for Mining Association Rules» [6] представлено реалізацію алгоритмів мінінгу, але вони представлені на універсальній мові програмування, без вказівки в якій із платформ

вони реалізовані. Відомо, що реалізація цих алгоритмів є для MongoDB та PostgreSQL, але немає для MySQL.

Вихідними є дані БД, що містять інформацію купівельних транзакцій. Кожна транзакція – це набір товарів, куплених покупцем за один візит (ринковий кошик). Задача полягає в пошуку закономірностей у покупках товарів, яка виражається твердженням (правилом) $X \rightarrow Y$, що означає: «якщо купується товар X, то також купується товар Y».

Загальне визначення асоціативного правила формуються так [3–5]. Нехай $I=\{i_1, i_2, \ldots, i_n\}$ – множина товарів, а «T» – множина транзакцій, де кожна транзакція «$t \in T$» – це набір елементів з множини «I, $t \in I$». Кожна транзакція являє собою бінарний вектор, де «$t[k]=1$», якщо елемент «i_k» є присутнім у транзакції, а якщо ні, то «$t[k]=0$». Стверджують, що транзакція «t» містить «X» (деякий набір товарів з множини «I»). Асоціативним правилом є імплікація $X \rightarrow Y$, де $X \subset I$, $Y \subset I$.

Множина підтримки «T_X» набору елементів «X» складається з усіх транзакцій, які містять цей набір. Підтримка («*Support*», скорочено «*Supp*») набору елементів «X» обчислюється так

$$Supp(X) = \frac{|T_X|}{T}. \qquad (1)$$

де «T_X»– число транзакцій, що містять «X», а «T» – загальне число транзакцій.

Рівень підтримки («*Supp*») або імовірність правила $X \rightarrow Y$ визначається відносним числом транзакцій, для яких виконується це правило:

$$Supp(X \rightarrow Y) = \frac{T_{X \rightarrow Y}}{T}. \qquad (2)$$

де «$T_{X \rightarrow Y}$» – число транзакцій, що містять імплікацію (кон'юнкцію) «$X \vee Y$».

Проблема пошуку полягає в генерації всіх асоціативних правил $X \rightarrow Y$, які перевищують задані пороги підтримки («*minSupp*»), тобто мають рівень підтримки «$Support \geq minSuppt$».

Процес виводу правил асоціації складається із двох кроків. Перший – це знаходження покриттів («великих наборів», що перевищують задані пороги підтримки), а другий – оформлення правил.

Для знаходження покриттів, що є самою трудомісткою частиною генерації асоціативних правил, використовується метод обмеженого перебору, запропонований М.М. Бонгардом [3–5] в 1967 р., який дозволяє виключати з аналізу події з низкою логічною частотою. Обмеження накладаються на довжину комбінацій ознак вхідних в асоціативну залежність, на розмір запису, на частоту аналізованих логічних комбінацій залежно від типу й розміру даних і поставлених задач. Алгоритми, що реалізують цей метод, використовують різні варіанти штучного обмеження перебору. Такими алгоритмами є [3–5]: AIS, SETM, Apriori, AprioriTid. Розглянемо алгоритми AIS та SETM.

1.3 Постановка задачі

Метою атестаційної роботи є розглядання варіантів реалізації алгоритмів мінінгу для інформаційної системи електронної комерції. Головною метою є реалізація алгоритму мінінгу для СУБД MySQL-server.

Для досягнення визначеної мети необхідно:

– визначити особливості використання алгоритмів мінінгу в рекомендаційних системах;

– проаналізувати реалізації алгоритмів мінінгу на різних платформах СУБД;

– визначити призначення алгоритмів мінінгу;

– аналіз алгоритмів мінінгу;

– зробити порівняльний аналіз алгоритмів мінінгу;

– реалізація алгоритму мінінгу на основі СУБД MySQL;

– зробити висновки за результатами роботи.

2 АНАЛІЗ МЕТОДІВ МІНІНГУ ДЛЯ ВИЗНАЧЕННЯ РЕКОМЕНДАЦІЙ У СИСТЕМАХ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

2.1 Призначення алгоритмів мінінгу

Алгоритми мінінгу, застосовані в контексті систем електронної комерції, виконують ряд важливих функцій, спрямованих на максимізацію користі та забезпечення оптимального взаємодії між платформою та користувачами. Призначення цих алгоритмів можна поділити на кілька ключових аспектів:

– визначення потреб та персоналізація. Однією з ключових ролей алгоритмів мінінгу є визначення індивідуальних потреб користувачів. Шляхом аналізу їхнього попереднього поведінки, виборів та покупок, алгоритми можуть створити персоналізовані рекомендації, враховуючи унікальні характеристики та смаки кожного користувача;

– прогнозування тенденцій та попиту. Аналізуючи масиви даних, алгоритми мінінгу можуть прогнозувати тенденції ринку та зміни в споживчому попиті. Це дозволяє платформам електронної комерції адаптуватися до швидко змінного середовища та надавати актуальні та популярні пропозиції користувачам;

– покращення систем рекомендацій. Алгоритми мінінгу використовуються для постійного вдосконалення систем рекомендацій. Це включає в себе аналіз зворотного зв'язку від користувачів, врахування нових товарів та оновлення асортименту з метою надання більш точних та привабливих пропозицій;

– підвищення ефективності маркетингу та продажів. Алгоритми мінінгу також використовуються для оптимізації стратегій маркетингу та продажів. Вони допомагають ідентифікувати ефективні кампанії, прогнозувати результати акцій та персоналізувати пропозиції з метою збільшення конверсії;

– адаптація до змін у поведінці користувачів. Один з ключових аспектів призначення алгоритмів мінінгу - адаптація до змін у поведінці користувачів.

Шляхом постійного моніторингу та аналізу змін, алгоритми можуть швидко реагувати та пристосовуватися до нових уподобань та потреб споживачів;

Загальною метою використання алгоритмів мінінгу в контексті систем електронної комерції є створення інтелектуальних та динамічних систем, здатних ефективно адаптуватися до змін в споживчому ринку та надавати персоналізовані рекомендації, сприяючи тим самим покращенню користувацького досвіду та збільшенню конкурентоспроможності платформ електронної комерції.

2.2 Алгоритм AIS

Алгоритм AIS (Artificial Immune System) був вперше розроблений Агравалом (Agrawal), Імелінскі (Imielinski) та Свамі (Swami) з метою виявлення асоціативних правил для ефективного видобутку цінної інформації з баз даних. Основний акцент у цьому алгоритмі стався на підвищенні якості баз даних та їхньої функціональності для обробки запитів, спрямованих на підтримку прийняття рішень.

В основі алгоритму лежить генерація одноелементних асоціативних правил, де наслідком є лише один елемент. Наприклад, формуються правила вигляду $X \cap Y \Rightarrow Z$, але не $X \Rightarrow Y \cap Z$. Алгоритм сканує базу даних кілька разів для отримання наборів елементів, які найчастіше зустрічаються у контексті AIS.

Для підвищення ефективності роботи алгоритму було введено метод оцінки, що дозволяє відсікати набори-кандидати, які не мають потенціалу для стати значущими. Це дозволяє уникнути надмірних обчислень на непотрібних наборах.

У алгоритмі AIS, який передбачає збереження всіх наборів-кандидатів і наборів часто зустрічаються в оперативній пам'яті, також передбачено механізм керування пам'яттю для оптимізації використання ресурсів, особливо у випадку нестачі пам'яті.

Алгоритм AIS включає кілька етапів формування наборів-кандидатів та підрахунку їхньої підтримки під час сканування бази даних. Він генерує набори кандидатів різних розмірностей шляхом розширення вже існуючих наборів на нові

елементи, перевіряючи їхню підтримку та відповідність заданому порогу. Цей процес повторюється до тих пір, поки один з наборів не залишиться порожнім.

Серед недоліків цього алгоритму є те, що він породжує велику кількість наборів-кандидатів, які, в кінцевому підсумку, виявляються надто малими або несуттєвими. Це призводить до великого обсягу зайнятого місця та витрати значних обчислювальних ресурсів. Крім того, для виконання цього алгоритму потрібно проводити занадто багато проходів по всій базі даних, що збільшує час виконання та обчислювальні витрати.

Алгоритм AIS дозволяє будувати асоціативні правила, багаторазово скануючи базу даних *D*. Для використання алгоритму повинні бути задані: L_k – множина *k*-елементних частих наборів і C_k – множина кандидатів *k*-елементних наборів, сформованих з L_k, чия підтримка *Supp* не менше заданої *minSupp*. Алгоритм AIS можна представити у вигляді псевдокоду [3–5]:

Псевдокод алгоритму AIS:

```
1) L₁ = { Великі 1-елемети };
2) for ( k = 2; Lₖ₋₁ ≠ ∅; k++ ) do begin
3)    Cₖ ≠ ∅;
4)    forall transactions t ∈ D do begin
5)          Lₜ = subset(Lₖ₋₁, t); // Великі набори елементів, що містяться в t
6)          forall large itemsets lₜ ∈ Lₜ do begin
7)                Cₜ = 1-extensions of lₜ contained in t; //Кандидати, що містяться
в t
8)                forall candidates c ∈ Cₜ do
9)                      if (c ∈ Cₖ) then
                              add 1 to the count of c in the corresponding entry in
                        Cₖ
                        else
                              add c to Ck with a count of 1;
10)               end
11)         Lₖ = {c ∈ Cₖ | c.count  >= minsup}
12) end
13) Answer = Uₖ Lₖ;
```

Опишемо виконання алгоритму AIS за кроками.

Крок 1. Для «*k*=1» виконати сканування бази даних, сформувати всі можливі «1»-елементні набори – «L_1» і розрахувати рівень підтримки «*Supp*». Створити

множину «1»-елементних наборів кандидатів «$c \in C_k$», відібравши з «L_1» усі набори в яких підтримка більше «*minSupp*».

Крок 2. $k=k+1$.

Крок 3. Якщо не вдається створити «*k*»-елементні набори, то завершити операції, інакше виконати наступний крок.

Крок 4. Виконати сканування бази даних, сформувати всі можливі «*k*»-елементні набори – «L_k» і розрахувати рівень підтримки «*Supp*». Створити множину «*k*»-елементних наборів кандидатів «$c \in C_k$», відібравши з «L_k» усі набори в яких підтримка більше «*minSupp*». Повернутися до кроку 2.

Результатом роботи алгоритму є об'єднання всіх множин «L_k» для всіх «*k*».

Під час сканування бази даних генеруються та підраховуються набори елементів-кандидатів. Після зчитування транзакції визначається, які з наборів елементів, що були визнані великими під час попереднього проходу, містяться в цій транзакції (рядок 5). Нові набори елементів-кандидатів генеруються шляхом розширення великих наборів елементів іншими елементами в транзакції (рядок 7). Кандидати, згенеровані в результаті транзакції, додаються до множини наборів статей-кандидатів, що зберігаються для цього проходу, або збільшується кількість відповідних статей, якщо вони були створені попередньою транзакцією (рядок 9).

Приклад роботи алгоритму AIS показано на рисунку 2.

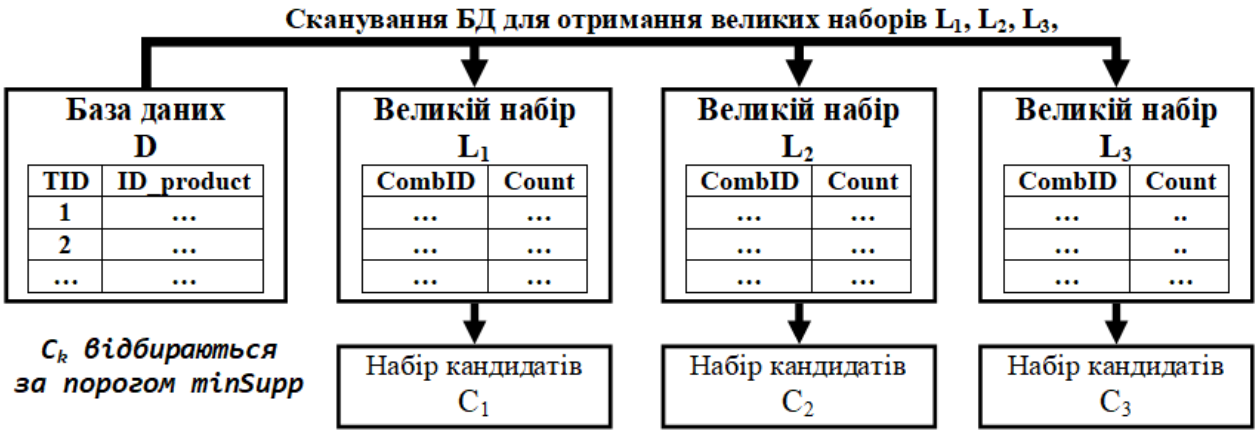

Рис. 2 – Приклад роботи алгоритму AIS

2.3 Алгоритм SETM

Алгоритм SETM (Sequential Evaluation of Traversals for Mining) спрямований на виявлення часто зустрічаються наборів об'єктів у великих обсягах даних. Він враховує послідовність обходу даних для виявлення цих наборів та має на меті покращення ефективності в порівнянні з іншими алгоритмами.

Алгоритм SETM використовує послідовність обходу даних для виявлення часто зустрічаються наборів об'єктів. Він розглядає різні маршрути та шляхи обходу даних з метою визначення наборів, що регулярно зустрічаються. Цей метод може бути корисним при аналізі великих обсягів даних, де звичайні методи перебору наборів стають неефективними.

SETM працює шляхом поетапного обходу даних з метою виявлення часто зустрічаються наборів об'єктів. Цей алгоритм може використовуватися для пошуку зв’язків між об'єктами в різних областях, де важливо виявити часті шаблони або зв'язки в даних для подальшого аналізу та прийняття рішень.

Алгоритм SETM був мотивований бажанням використовувати SQL для обчислення великих наборів елементів. C_k (L_k) в псевдокоді представляє множину кандидатів (великих) наборів елементів, в яких *TID* генеруючих транзакцій були пов'язані з наборами елементів. Кожен член цих наборів має вигляд < *TID*; набір елементів >.

Як і AIS, алгоритм SETM також генерує кандидатів на основі транзакцій, що зчитуються з бази даних. Таким чином, він генерує і підраховує кожен набір елементів-кандидатів, який генерує алгоритм AIS. Однак, щоб використовувати стандартну операцію приєднання SQL для генерації кандидатів, SETM відокремлює генерацію кандидатів від підрахунку. Він зберігає копію набору елементів-кандидатів разом з *TID* генеруючої транзакції у послідовній структурі (рядок 9). Наприкінці цього етапу підрахунок кількості наборів-кандидатів визначається шляхом сортування (рядок 12) та агрегування цієї послідовної структури (рядок 13). SETM запам'ятовує *TID* транзакцій-генерувальників з наборами-кандидатами. Щоб уникнути операції підмножини, він використовує цю

інформацію для визначення великих наборів елементів, що містяться у прочитаній транзакції (рядок 6). L_k C_k та отримує шляхом видалення тих кандидатів, які не мають мінімальної підтримки (рядок 13). Припускаючи, що база даних відсортована у порядку *TID*, SETM може легко знайти великі набори елементів, що містяться у транзакції, на наступному проході, відсортувавши L_k за *TID* (рядок 15).

Фактично, йому потрібно відвідати кожен член L_k лише один раз у порядку *TID*, а генерація кандидатів на рядках з 5 по 11 може бути виконана за допомогою операції реляційного злиття-об'єднання. Недолік цього підходу пов'язаний, головним чином, з розміром наборів-кандидатів C_k. Для кожного набору-кандидата набір-кандидат тепер має стільки записів, скільки транзакцій, в яких присутній цей набір-кандидат. Більше того, коли ми готові підрахувати підтримку для наборів-кандидатів в кінці проходу, C_k знаходиться у неправильному порядку і його потрібно відсортувати по наборам (рядок 12). Після підрахунку та відсікання малих наборів елементів-кандидатів, які не мають мінімальної підтримки, результуюча множина *L*k потребує ще одного сортування за *TID* (рядок 15), перш ніж її можна буде використати для генерації кандидатів у наступному проході.

За алгоритмом SETM (на відміну від алгоритму AIS) починаючи з формування третього великого набору «L_3» починається врахування результатів відбору кандидатів «C_2» на попередньому кроці. Для цього коригується великий набір «L_2» (на рис. 3. корегований великий набір «L_2» позначається як «L_{20}». Це коригування продовжується й за подальшими кроками, наприклад, для великого набору «L_3» створюється «L_{30}», який корегується за допомогою набору кандидатів «C_3».

Псевдокод алгоритму SETM:

```
1) L₁ = {великі 1-елемети};
2) L₁ = { Великі 1-елементні набори разом з TID, в яких вони з'являються, відсортовані за TID};
3) for ( k = 2; Lₖ₋₁ ≠ ∅; k++ ) do begin
4)     Cₖ ≠ ∅;
5)     forall transactions t ∈ D do begin
6)         Lₜ = {l ∈ Lₖ₋₁ | l.TID = t.TID}; // Великі (k-1)-елементні набори, що містяться в t
```

```
7)          forall large itemsets l_t ∈ L_t do begin
8)                C_t = 1-extensions of l_t contained in t; // Кандидати в t
9)                C_k += {< t.TID; c > | c ∈ C_t};
10)         end
11)   end
12)   sort C_k on itemsets;
13)   delete all itemsets c ∈ C_k for which c.count < minsup giving L_k;
14)   Lk = {< l.itemset, count of l in L_k > | l ∈ L_k}; // У поєднанні з кроком 13
15)   sort L_k on TID;
16) end
17) Answer = U_k L_k;
```

Продуктивність алгоритму SETM критично залежить від розміру множини C_k відносно обсягу пам'яті. Якщо C_k знаходиться в пам'яті, то два кроки сортування можна виконати, використовуючи сортування в пам'яті. Якщо C_k занадто великий для пам'яті, ми записуємо записи з C_k на диск у порядку FIFO, коли буфер, виділений для елементів-кандидатів, вичерпується, оскільки ці записи не будуть потрібні до кінця проходу. Однак, C_k тепер потребує двох зовнішніх сортувань.

Приклад роботи алгоритму SETM показано на рисунку 3.

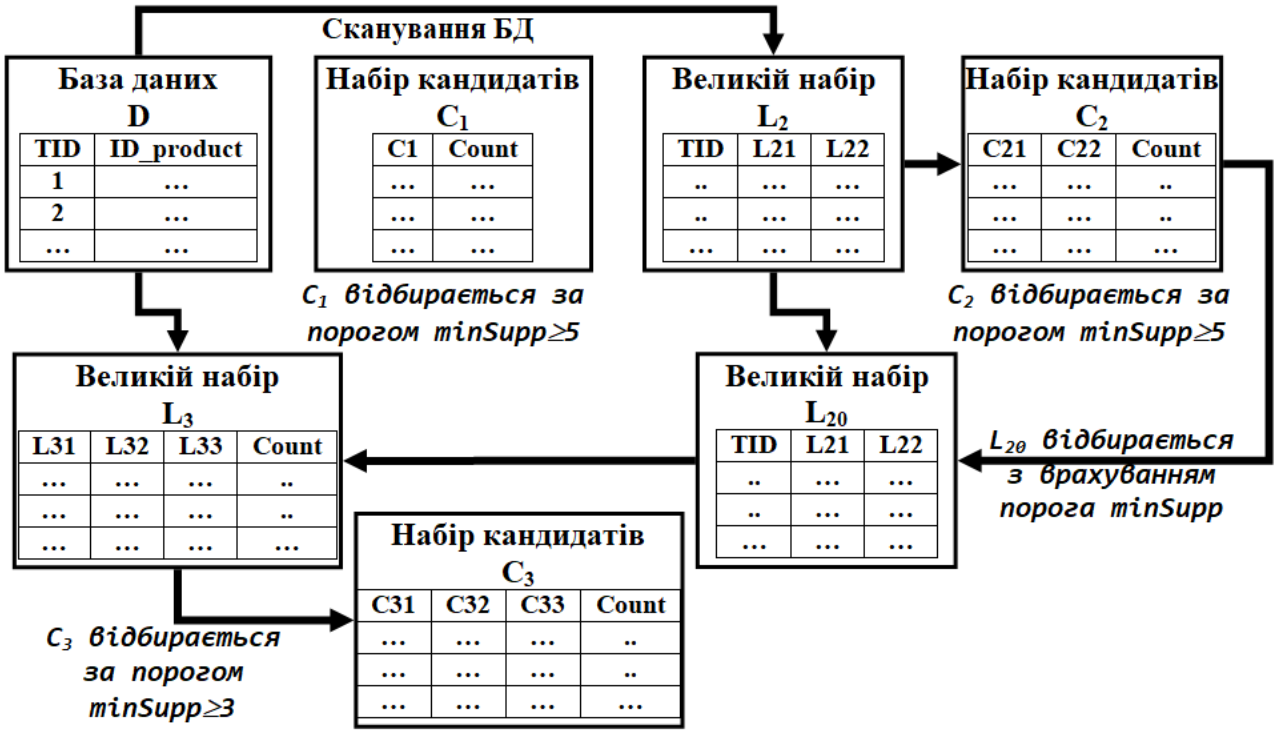

Рис. 3 – Приклад роботи алгоритму SETM

В алгоритмі SETM в базах даних набори елементів-кандидатів генеруються на льоту» під час сканування бази даних, але підраховуються в кінці проходу. Потім генеруються нові набори елементів-кандидатів так само, як і в алгоритмі AIS, але ідентифікатор транзакції *TID* генеруючої транзакції зберігається разом з набором елементів-кандидатів у послідовній структурі. Це відокремлює процес генерації кандидатів від підрахунку. Наприкінці проходу, шляхом агрегування послідовної структури, визначається кількість підтримуваних наборів елементів-кандидатів.

Алгоритм SETM має той самий недолік, що й алгоритм AIS. Іншим недоліком є те, що для кожного набору-кандидата існує стільки записів, скільки має його значення підтримки.

2.4 Алгоритм Apriori

Виявлення частих наборів об'єктів - завдання, яке вимагає значної обчислювальної потужності та відповідно, часу. Алгоритм, Apriori, був описаний у 1994 році Срікантом Рамакрішнаном (Ramakrishnan Srikant) та Ракешом Агравалом (Rakesh Agrawal). Цей алгоритм використовує властивість підтримки, що стверджує: підтримка будь-якого набору об'єктів не може перевищувати мінімальну підтримку будь-якої з його підмножини. Нерівність властивість підтримки виглядає так:

$$Supp_F \leq Supp_E \text{ при } E \subset F \qquad (2.1)$$

Алгоритм Apriori визначає набори, що часто зустрічаються, за кілька етапів. На і-му етапі визначаються всі часто зустрічаються і-елементні набори. Кожен етап складається з двох кроків: формування кандидатів (candidate generation) і підрахунку підтримки кандидатів (candidate counting).

Розглянемо і-й етап. На кроці формування кандидатів алгоритм створює множину кандидатів з і -елементних наборів, чия підтримка поки що не

обчислюється. На кроці підрахунку кандидатів алгоритм сканує безліч транзакцій, обчислюючи підтримку наборів-кандидатів. Після сканування відкидаються кандидати, підтримка яких менша від визначеного користувачем мінімуму, і зберігаються тільки часто зустрічаються і -елементні набори. Під час 1-го етапу обрана множина наборів кандидатів містить усі 1-елементні часті набори. Алгоритм обчислює їхню підтримку під час кроку підрахунку кандидатів.

Описаний алгоритм можна записати у вигляді такого псевдокоду:

```
= {найчастіші 1-елементи набори}
для (k=2; L_{k-1} <> φ; k++)
генерація кандидатів
      для всіх транзакцій t ∈ D виконати
      видалення надлишкових правил
            для всіх кандидатів c ∈ C_t виконати

            кінець для всіх
      кінець для всіх
∈ Ck | c.count >= Supp_min} // відбір кандидатів
кінець для всіх
Результат = ∪_k L_k
```

Позначення, що використовуються в алгоритмі:

- множина k-елементних частих наборів, чия підтримка не менша за задану користувачем. Кожен член множини має набір упорядкованих ($i_j < i_p$, якщо $j < p$) елементів F та значення підтримки набору $Supp_F > Supp_{min}$:

L

д

е

=

$_k$- множина кандидатів k-елементних наборів потенційно найчастіших. Кожен член множини має набір упорядкованих ($i_j < i_p$, якщо $j < p$) елементів F та значення підтримки набору Supp.

F

Опишемо цей алгоритм по кроках. ,

Крок 1. Присвоїти k = 1 і виконати відбір усіх 1-елементних наборів,

u

p

p

у яких підтримка більша за мінімально задану користувачем $Supp_{min}$.

Крок 2. k = k +1.

Крок 3. Якщо не вдається створювати k-елементні набори, то завершити алгоритм, інакше виконати наступний крок.

Крок 4. Створити множину k-елементних наборів кандидатів у найчастіші набори. Для цього необхідно об'єднати в k-елементні кандидати (k - 1) - елементні найчастіші набори. Кожен кандидат $c \in C_k$ формуватиметься шляхом додавання до (k - 1)-елементного частого набору - *p* елемента з іншого (k-1)-елементного частого набору - *q*. Причому додається останній елемент набору *q*, який за порядком вищий, ніж останній елемент набору *p* ($p.item_{k-1} < q.item_{k-1}$). При цьому перші всі (k - 2) елементи обох наборів однакові ($p.item_1 = q.item_1$, $p.item_2 = q.item_2$, ..., $p.item_{k-2} = q.item_{k-2}$).

Це може бути записано у вигляді такого SQL-подібного запиту:

```
insert into Ck
select p.item1, p.item2, ..., p.itemk-1, q.itemk-1
from Lk-1 p, Lk-1 q
where p.item1 = q.item1, p.item2 = q.item2, ..., p.itemk-2 = q.itemk-2,
p.itemk-1 < q.itemk-1
```

Крок 5. Для кожної транзакції *T* із множини *D* вибрати кандидатів C_t із множини C_k , присутніх у транзакції *T*. Для кожного набору з попобудованої множини C_k видалити набір, якщо хоча б одна з його (*k* - 1) підмножин не є такою, що часто зустрічається, тобто відсутня в множині L_{k-1}. Це можна записати у вигляді такого псевдокоду:

```
для всіх наборів c ∈ Ck виконати
      для всіх (k-1) - під наборів s з c виконати
            якщо (s ∉ Lk-1) то
                  видалити c із Ck
```

Крок 6. Для кожного кандидата з множини C_k збільшити значення підтримки на одиницю.

Крок 7. Вибрати тільки кандидатів L_k з множини C_k, у яких значення підтримки більше заданої користувачем $Supp_{min}$. Повернутися до кроку 2.

Результатом роботи алгоритму є об'єднання всіх множин L_k для всіх *k*.

Розглянемо роботу алгоритму на прикладі, наведеному в табл. 2.1, при $Supp_{min}$ = 0,5. На першому кроці маємо таку множину кандидатів C_1(вказуються ідентифікатори товарів) (табл. 2.2).

Таблиця 2.1 – Приклад даних для демонстрації роботи алгоритму

Ідентифікатор	**Назва товару**	**Ціна**
0	Шоколад	30
1	Чіпси	12
2	Кокоси	10
3	Вода	4
4	Пиво	14
5	Горіхи	15

Таблиця 2.2 – Роботу алгоритму на прикладі

№	**Набір**	**Supp**
1	{0}	0
2	{1}	0.5
3	{2}	0.75
4	{3}	0.25
5	{4}	0.75
6	{5}	0.75

Заданій мінімальній підтримці задовольняють тільки кандидати 2, 3, 5 і 6, отже:

$$L_1 = \{\{1\}, \{2\}, \{3\}, \{5\}\}.$$

На другому кроці збільшуємо значення k до двох. Оскільки можна побудувати 2-елементні набори, то отримуємо множину C_2 (табл. 2.3).

Таблиця 2.3 – Отримання множини C_2

№	Набір	Supp
1	{1, 2}	0.25
2	{1, 3}	0.5
3	{1, 5}	0.25
4	{2, 3}	0.5
5	{2, 5}	0.75
6	{3, 5}	0.5

Із побудованих кандидатів заданій мінімальній підтримці задовольняють тільки кандидати 2, 4, 5 і 6, отже:

$$L_2 = \{\{1, 3\}, \{2, 3\}, \{2, 5\}, \{3, 5\}\}.$$

На третьому кроці перейдемо до створення 3-елементних кандидатів і підрахунку їхньої підтримки. У результаті отримаємо таку множину C_3 (табл. 2.4).

Таблиця 2.4 – Отримання множини C_3

№	Набір	Supp
1	{2, 3, 5}	0.5

Цей набір задовольняє мінімальній підтримці, отже:

$$L_3 = \{\{2, 3, 5\}\}.$$

Оскільки 4-елементні набори створити не вдасться, то результатом роботи алгоритму є множина:

$$L = L_1 \cup L_2 \cup L_3 = \{\{1\}, \{2\}, \{3\}, \{5\}, \{1, 3\}, \{2, 3\}, \{2, 5\}, \{3, 5\}, \{2, 3, 5\}\}.$$

Для підрахунку підтримки кандидатів потрібно порівняти кожну транзакцію з кожним кандидатом. Очевидно, що кількість кандидатів може бути дуже великою і потрібен ефективний спосіб підрахунку. Набагато швидше й ефективніше використовувати підхід, заснований на зберіганні кандидатів у хеш-дереві. Внутрішні вузли дерева містять хеш-таблиці з покажчиками на нащадків, а листя - на кандидатів. Це дерево використовується під час швидкого підрахунку підтримки для кандидатів.

Хеш-дерево будують щоразу, коли формують кандидатів. Спочатку дерево складається тільки з кореня, який є листом, і не містить жодних кандидатів-наборів. Щоразу, коли формується новий кандидат, його заносять у корінь дерева, і так доти, доки кількість кандидатів у корені-листі не перевищить певного порога. Щойно це відбувається, корінь перетворюється на хеш-таблицю, тобто стає внутрішнім вузлом, і для нього створюються нащадки-листя. Усі кандидати розподіляються по вузлах-нащадках згідно з хеш-значеннями елементів, що входять до набору. Кожен новий кандидат хешується на внутрішніх вузлах, поки не досягне першого вузла-листа, де він і зберігатиметься, доки кількість наборів знову ж таки не перевищить порогу.

Після того як хеш-дерево з кандидатами-наборами побудовано, легко підрахувати підтримку для кожного кандидата. Для цього потрібно «пропустити» кожну транзакцію через дерево і збільшити лічильники для тих кандидатів, чиї елементи також містяться і в транзакції, C_k і $C_k \cap T_i = C_k$. На кореневому рівні хеш-функція застосовується до кожного об'єкта з транзакції. Далі, на другому рівні, хеш-функція застосовується до других об'єктів і т. д. На другому рівні хешується *k*-елемент, і так доти, доки не досягнемо листа. Якщо кандидат, що зберігається в листі, є підмножиною транзакції, що розглядається, збільшуємо лічильник підтримки цього кандидата на одиницю.

Після того як кожну транзакцію з вихідного набору даних «пропущено» через дерево, можна перевірити, чи задовольняють значення підтримки кандидатів мінімальному порогу. Кандидати, для яких ця умова виконується, переносяться в розряд тих, що часто зустрічаються. Крім того, слід запам'ятати і підтримку набору, яка стане в пригоді під час витягання правил. Ці ж дії застосовують для знаходження (k + 1)-елементних наборів і т. д.

2.5 Висновок за аналізом алгоритмів мінінгу

Було проведено глибокий аналіз різних алгоритмів мінінгу, спрямованих на виявлення рекомендаційних залежностей у системах електронної комерції. Під час аналізу було досліджено ключові алгоритми, такі як Apriori, SETM та AIS.

Apriori, як класичний алгоритм, забезпечує ефективну генерацію часто зустрічаючих наборів шляхом послідовного сканування бази даних. SETM, навпаки, пропонує альтернативний підхід, що базується на перевагах структурної інформації та використовує покращені методи генерації кандидатів. Зокрема, AIS намагається уникнути надмірного утворення наборів-кандидатів шляхом впровадження оцінювання їхньої значущості.

Після уважного аналізу можна визначити, що кожен з цих алгоритмів має свої переваги та обмеження. Apriori, хоч і ефективний у створенні часто зустрічаючих наборів, має обмеження щодо обробки великих обсягів даних. SETM, з своєю здатністю працювати зі структурною інформацією, може бути складним у використанні для великих і неструктурованих даних. Алгоритм AIS, хоч і намагається уникнути зайвого формування кандидатів, також потребує значних обчислювальних витрат. В цілому, результати аналізу свідчать про те, що вибір алгоритму мінінгу для рекомендаційних систем електронної комерції має бути здійснений з урахуванням специфіки даних, обсягів, потреб бізнесу та вимог до швидкості обробки. Кожен з алгоритмів має свої переваги та недоліки, які необхідно урахувати для досягнення оптимального результату в контексті конкретного застосування.

3 АНАЛІЗ ВАРІАНТІВ РЕАЛІЗАЦІЇ АЛГОРИТМІВ МІНІНГУ НА ПЛАТФОРМІ СУБД MYSQL

3.1 Вибір предметної області для дослідження

У кваліфікаційної роботі розглядається інформаційна система електронної комерції мережі аптек. Мережа аптек генерує величезні обсяги даних щоденно, реалізація алгоритмів мінінгу в таких умовах є важливою складовою аналізу продажів. Аптеки, зазвичай пропонують різні продукти та послуги. Застосування алгоритмів мінінгу для аналізу покупкового поведінки клієнтів може допомогти визначити їхні індивідуальні потреби та вибір, що надає можливість персоналізації рекомендацій та поліпшення обслуговування.

У сфері електронної комерції мережі аптек важливо вчасно виявляти та прогнозувати попит на різні товари. Алгоритми мінінгу можуть бути використані для аналізу історії замовлень, визначення популярних товарів, та управління запасами для ефективного вирішення питань постачання та попиту.

Застосування алгоритмів мінінгу може допомогти ідентифікувати ефективні маркетингові стратегії, прогнозувати результати рекламних кампаній, та забезпечити персоналізовані пропозиції для користувачів, що сприяє збільшенню конверсії та покращенню відповідності до ринкових умов. Сфера електронної комерції є динамічною, і зміни в попиті та трендах відбуваються швидко. Використання алгоритмів мінінгу дозволяє системі ефективно адаптуватися до змін у покупковому поведінці користувачів, що важливо для успішного функціонування мережі аптек.

Враховуючи вищенаведені аспекти, обрана предметна область «Інформаційна система електронної комерції мережі аптек» визначається як ідеальний контекст для тестування та оптимізації алгоритмів мінінгу на платформі СУБД MySQL. Дослідження в цьому напрямку має потенціал принести вагомий внесок у розвиток ефективних та інноваційних рекомендаційних систем для сфери

електронної комерції в мережі аптек. Важливо вибрати інфраструктуру для реалізації алгоритмів, що забезпечить швидкість обробки та аналізу даних, а саме мати наповнену інформацію в БД.

3.2 Розробка бази даних системи е-комерції з продажу ліків

Для реалізації БД інформаційної системи електронної комерції мережі аптек використовувалося середовище розробки Workbench [10]. Реалізована фізична модель (рис. 5) EER-моделі БД інформаційної системи електронної комерції мережі аптек подається на рис. 6

Логічне та фізичне моделювання даних інформаційної системи електронної комерції мережі аптек проводилося з використанням CASE-засобу «Allfusion ErWin Data Modeler» [11].

Розроблена логічна модель даних інформаційної системи електронної комерції мережі аптек, яка представлена на рис. 4

Предметною галуззю інформаційної системи електронної комерції мережі аптек є продаж ліків та препаратів лікарського призначення. Визначені такі сутності, атрибути та їх типи.

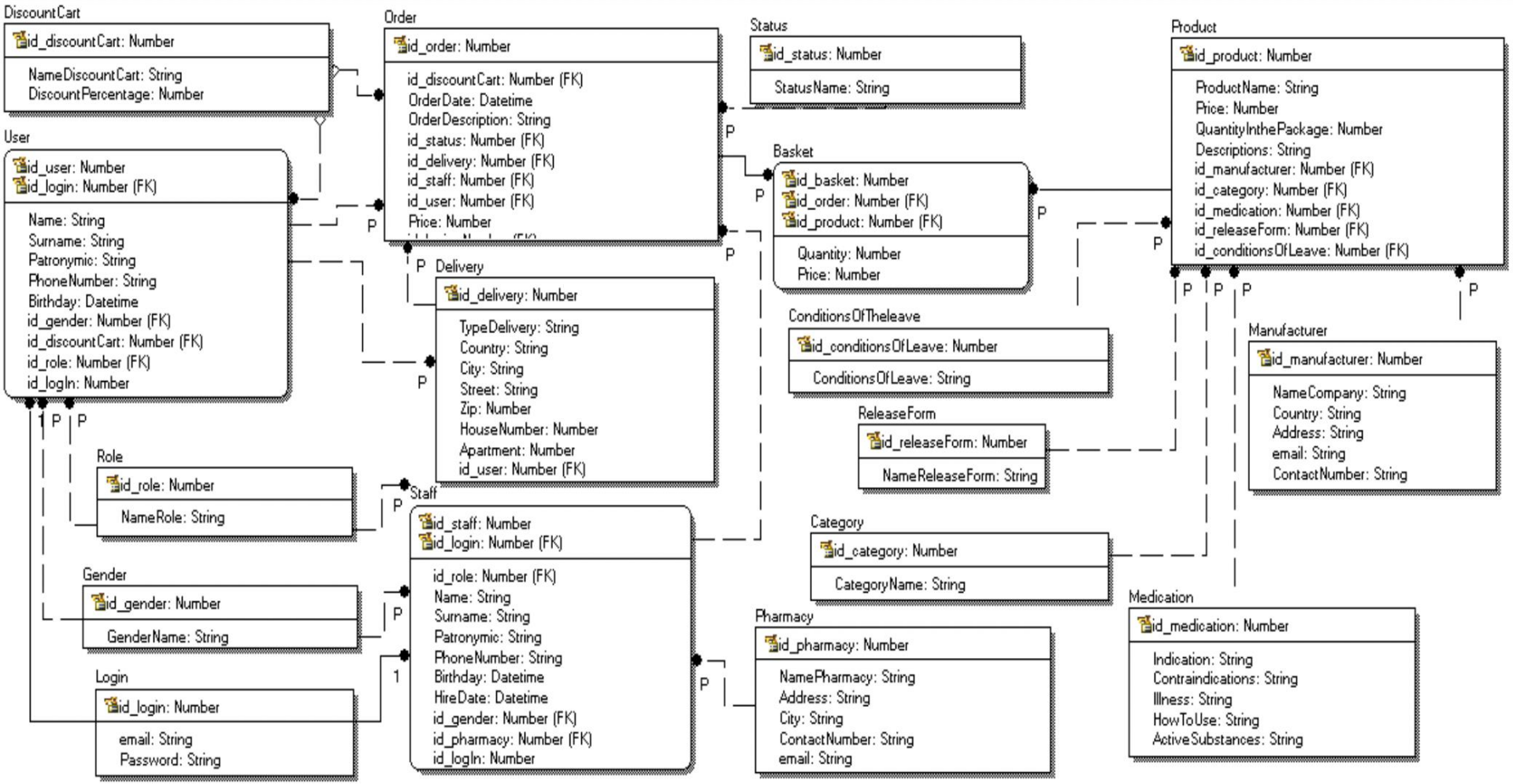

Рисунок 4 – Логічна модель даних інформаційної системи електронної комерції мережі аптек

Сутність «DiscountCart» зберігає дані дисконтних карток в інформаційній системі електронної комерції мережі аптек. Для сутності «DiscountCart» визначені такі атрибути:

– ключовий атрибут «id_discountCart» (домен «числовий», тип «INT») – первинний ключ сутності «DiscountCart» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «NameDiscountCart» (домен «рядковий», тип «VARCHAR») – назва дисконтної картки;

– атрибут «DiscountPercentage» (домен «числовий», тип «FLOAT») – відсоток знижки дисконтної картки.

Сутність «User» зберігає дані зареєстрованих користувачів в інформаційній системі електронної комерції мережі аптек. Для сутності «User» визначені такі атрибути:

– ключовий атрибут «id_user» (домен «числовий», тип «INT») – первинний ключ сутності «User» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «Name» (домен «рядковий», тип «VARCHAR») – Ім’я зареєстрованого користувача;

– атрибут «Surname» (домен «рядковий», тип «VARCHAR») – Прізвище зареєстрованого користувача;

– атрибут «Patronymic» (домен «рядковий», тип «VARCHAR») – Ім’я по батькові зареєстрованого користувача;

– атрибут «PhoneNumber» (домен «рядковий», тип «VARCHAR») – номер телефону зареєстрованого користувача;

– атрибут «Birthday» (домен «дата», тип «DATE») – дата народження зареєстрованого користувача;

– ключовий атрибут «id_gender» (домен «числовий», тип «INT») – зовнішній ключ сутності «Category» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_delivery» (домен «числовий», тип «INT») – зовнішній ключ сутності «Delivery» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_discountCart» (домен «числовий», тип «INT») – зовнішній ключ сутності «DiscountCart» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_role» (домен «числовий», тип «INT») – зовнішній ключ сутності «Role» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_login» (домен «числовий», тип «INT») – зовнішній ключ сутності «LogIn» в таблиці бази даних інформаційної системи електронної комерції мережі аптек.

Сутність «LogIn» зберігає дані логіну та паролю користувачів в інформаційній системі електронної комерції мережі аптек. Для сутності «LogIn» визначені такі атрибути:

– ключовий атрибут «id_login» (домен «числовий», тип «INT») – первинний ключ сутності «LogIn» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «email» (домен «рядковий», тип «VARCHAR») – поштова адреса користувача інформаційної системи електронної комерції мережі аптек;

– атрибут «Password» (домен «рядковий», тип «VARCHAR») – пароль користувача інформаційної системи електронної комерції мережі аптек.

Сутність «Gender» зберігає дані про стать користувачів в інформаційній системі електронної комерції мережі аптек. Для сутності «Gender» визначені такі атрибути:

– ключовий атрибут «id_gender» (домен «числовий», тип «INT») – первинний ключ сутності «Gender» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «GenderName» (домен «рядковий», тип «VARCHAR») – назва статі.

Сутність «Role» зберігає дані ролі користувачів в інформаційній системі електронної комерції мережі аптек. Для сутності «Role» визначені такі атрибути:

– ключовий атрибут «id_role» (домен «числовий», тип «INT») – первинний ключ сутності «Role» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «NameRole» (домен «рядковий», тип «VARCHAR») – назва ролі користувача в інформаційній системі електронної комерції мережі аптек.

Сутність «Delivery» зберігає дані адресів доставки а також тип доставки ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «Delivery» визначені такі атрибути:

– ключовий атрибут «id_delivery» (домен «числовий», тип «INT») – первинний ключ сутності «Delivery» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «TypeDelivery» (домен «рядковий», тип «VARCHAR») – тип доставки ліків та препаратів лікарського призначення;

– атрибут «Country» (домен «рядковий», тип «VARCHAR») – назва країни доставки ліків та препаратів лікарського призначення;

– атрибут «City» (домен «рядковий», тип «VARCHAR») – назва міста доставки ліків та препаратів лікарського призначення;

– атрибут «Street» (домен «рядковий», тип «VARCHAR») – назва вулиці доставки ліків та препаратів лікарського призначення;

– атрибут «Zip» (домен «числовий», тип «INT») – поштовий індекс доставки ліків та препаратів лікарського призначення;

– атрибут «HouseNumber» (домен «числовий», тип «INT») – номер будинку доставки ліків та препаратів лікарського призначення;

– атрибут «Apartment» (домен «числовий», тип «INT») – номер квартири доставки ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек.

Сутність «Order» зберігає дані всіх замовлень зареєстрованих користувачів в інформаційній системі електронної комерції мережі аптек. Для сутності «Order» визначені такі атрибути:

– ключовий атрибут «id_order» (домен «числовий», тип «INT») – первинний ключ сутності «Order» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «OrderDate» (домен «дата», тип «DATE») – дата оформлення замовлення в інформаційній системі електронної комерції мережі аптек;

– атрибут «OrderDescription» (домен «рядковий», тип «VARCHAR») – доповнення до замовлення зареєстрованого користувача інформаційної системи електронної комерції мережі аптек;

– атрибут «Price» (домен «числовий», тип «FLOAT») – загальна ціна замовлення в інформаційній системі електронної комерції мережі аптек;

– ключовий атрибут «id_status» (домен «числовий», тип «INT») – зовнішній ключ сутності «Status» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_delivery» (домен «числовий», тип «INT») – зовнішній ключ сутності «Delivery» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_staff» (домен «числовий», тип «INT») – зовнішній ключ сутності «Staff» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_user» (домен «числовий», тип «INT») – зовнішній ключ сутності «User» в таблиці бази даних інформаційної системи електронної комерції мережі аптек.

Сутність «Status» зберігає дані статусів замовлень в інформаційній системі електронної комерції мережі аптек. Для сутності «Status» визначені такі атрибути:

– ключовий атрибут «id_status» (домен «числовий», тип «INT») – первинний ключ сутності «Status» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «StatusName» (домен «рядковий», тип «VARCHAR») – назва статусу до замовлення в інформаційній системі електронної комерції мережі аптек.

Сутність «Basket» зберігає дані кошику зареєстрованого користувача в інформаційній системі електронної комерції мережі аптек. Для сутності «Basket» визначені такі атрибути:

– ключовий атрибут «id_basket» (домен «числовий», тип «INT») – первинний ключ сутності «Basket» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_order» (домен «числовий», тип «INT») – зовнішній ключ сутності «Order» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_product» (домен «числовий», тип «INT») – зовнішній ключ сутності «Product» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «Quantity» (домен «числовий», тип «INT») – кількість ліків та препаратів лікарського призначення в кошику зареєстрованого користувача інформаційної системи електронної комерції мережі аптек;

– атрибут «Price» (домен «числовий», тип «FLOAT») – ціна ліків та препаратів лікарського призначення з обліком кількості ліків та препаратів лікарського призначення.

Сутність «Staff» зберігає дані працівників інформаційної системи електронної комерції мережі аптек. Для сутності «Staff» визначені такі атрибути:

– ключовий атрибут «id_staff» (домен «числовий», тип «INT») – первинний ключ сутності «Staff» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «Name» (домен «рядковий», тип «VARCHAR») – Ім’я працівника інформаційної системи електронної комерції мережі аптек;

– атрибут «Surname» (домен «рядковий», тип «VARCHAR») – Прізвище працівника інформаційної системи електронної комерції мережі аптек;

– атрибут «Patronymic» (домен «рядковий», тип «VARCHAR») – Ім’я по батькові працівника інформаційної системи електронної комерції мережі аптек;

– атрибут «PhoneNumber» (домен «рядковий», тип «VARCHAR») – номер мобільного телефону працівника інформаційної системи електронної комерції мережі аптек;

– атрибут «Birthday» (домен «дата», тип «DATE») – дата народження працівника інформаційної системи електронної комерції мережі аптек;

– атрибут «HireDate» (домен «дата», тип «DATE») –дата прийому на роботу працівника інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_gender» (домен «числовий», тип «INT») – зовнішній ключ сутності «Gender» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_pharmacy» (домен «числовий», тип «INT») – зовнішній ключ сутності «Pharmacy» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_login» (домен «числовий», тип «INT») – зовнішній ключ сутності «LogIn» в таблиці бази даних інформаційної системи електронної комерції мережі аптек.

Сутність «ConditionsOfTheLeave» зберігає дані про умови відпуски ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «ConditionsOfTheLeave» визначені такі атрибути:

– ключовий атрибут «id_conditionsOfLeave» (домен «числовий», тип «INT») – первинний ключ сутності «ConditionsOfTheLeave» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «ConditionsOfLeave» (домен «рядковий», тип «VARCHAR») – умови відпуску ліків та препаратів лікарського призначення, які продаються в інформаційній системі електронної комерції мережі аптек.

Сутність «Product» зберігає дані ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «Product» визначені такі атрибути:

– ключовий атрибут «id_product» (домен «числовий», тип «INT») – первинний ключ сутності «Product» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «ProductName» (домен «рядковий», тип «VARCHAR») – назва лікарського препарату або препарату лікарського призначення, який продається в інформаційній системі електронної комерції мережі аптек;

– атрибут «Price» (домен «числовий», тип «FLOAT») – ціна лікарського препарату або препарату лікарського призначення, який продається в інформаційній системі електронної комерції мережі аптек;

– атрибут «QuantityInThePackage» (домен «числовий», тип «INT») – кількість в упаковці лікарського препарату або препарату лікарського призначення, який продається в інформаційній системі електронної комерції мережі аптек;

– атрибут «Descriptions» (домен «рядковий», тип «VARCHAR») – опис лікарського препарату або препарату лікарського призначення, який продається в інформаційній системі електронної комерції мережі аптек;

– ключовий атрибут «id_manufacturer» (домен «числовий», тип «INT») – зовнішній ключ сутності «Manufacturer» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_category» (домен «числовий», тип «INT») – зовнішній ключ сутності «Category» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_medication» (домен «числовий», тип «INT») – зовнішній ключ сутності «Medication» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_releaseForm» (домен «числовий», тип «INT») – зовнішній ключ сутності «ReleaseForm» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– ключовий атрибут «id_conditionsOfLeave» (домен «числовий», тип «INT») – зовнішній ключ сутності «ConditionsOfLeave» в таблиці бази даних інформаційної системи електронної комерції мережі аптек.

Сутність «ReleaseForm» зберігає дані форми випуку ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «ReleaseForm» визначені такі атрибути:

– ключовий атрибут «id_releaseForm» (домен «числовий», тип «INT») – первинний ключ сутності «ReleaseForm» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «NameReleaseForm» (домен «рядковий», тип «VARCHAR») – назва форми реалізації ліків або препарату лікарського призначення.

Сутність «Pharmacy» зберігає дані пунктів мережі аптек в інформаційній системі електронної комерції мережі аптек. Для сутності «Pharmacy» визначені такі атрибути:

– ключовий атрибут «id_pharmacy» (домен «числовий», тип «INT») – первинний ключ сутності «Pharmacy» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «NamePharmacy» (домен «рядковий», тип «VARCHAR») – назва одного з пунктів мережі аптек інформаційної системи електронної комерції мережі аптек;

– атрибут «Address» (домен «рядковий», тип «VARCHAR») – адреса одного з пунктів мережі аптек інформаційної системи електронної комерції мережі аптек;

– атрибут «City» (домен «рядковий», тип «VARCHAR») – місто в якому перебуває однин з пунктів мережі аптек інформаційної системи електронної комерції мережі аптек;

– атрибут «ContactNumber» (домен «рядковий», тип «VARCHAR») – контактний номер телефону одного з пунктів мережі аптек інформаційної системи електронної комерції мережі аптек;

Сутність «Category» зберігає дані категорії ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «Category» визначені такі атрибути:

– ключовий атрибут «id_category» (домен «числовий», тип «INT») – первинний ключ сутності «Category» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «CategoryName» (домен «рядковий», тип «VARCHAR») – назва категорії ліків та препаратів лікарського призначення, використовують пункти мережі аптек інформаційної системи електронної комерції.

Сутність «Manufacturer» зберігає дані виробників ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «Manufacturer» визначені такі атрибути:

– ключовий атрибут «id_manufacturer» (домен «числовий», тип «INT») – первинний ключ сутності «Manufacturer» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «NameCompany» (домен «рядковий», тип «VARCHAR») – назва компанії-постачальника, що виробляє ліки та препарати лікарського призначення, які використовують пункти мережі аптек інформаційної системи електронної комерції;

– атрибут «Country» (домен «рядковий», тип «VARCHAR») – країна виробник ліків та препаратів лікарського призначення, які використовують пункти мережі аптек інформаційної системи електронної комерції;

– атрибут «Address» (домен «рядковий», тип «VARCHAR») – адреса компанії-постачальника ліків та препаратів лікарського призначення;

– атрибут «email» (домен «рядковий», тип «VARCHAR») – поштова адреса компанії-постачальника ліків та препаратів лікарського призначення;

– атрибут «ContactNumber» (домен «рядковий», тип «VARCHAR») – контактний номер телефону компанії-постачальника ліків та препаратів лікарського призначення.

Сутність «Medication» зберігає дані про детальний опис ліків та препаратів лікарського призначення в інформаційній системі електронної комерції мережі аптек. Для сутності «Medication» визначені такі атрибути:

– ключовий атрибут «id_medication» (домен «числовий», тип «INT») – первинний ключ сутності «Medication» в таблиці бази даних інформаційної системи електронної комерції мережі аптек;

– атрибут «Indication» (домен «рядковий», тип «VARCHAR») – показання при яких можна приймати лікарський препарат або препарат лікарського призначення;

– атрибут «Contraindications» (домен «рядковий», тип «VARCHAR») – протипоказання при яких небажано приймати лікарський препарат або препарат лікарського призначення;

– атрибут «Illness» (домен «рядковий», тип «VARCHAR») – захворювання при якому треба приймати лікарський препарат або препарат лікарського призначення;

– атрибут «ActiveSubstances» (домен «рядковий», тип «VARCHAR») – активна речовина в лікарському препараті або препараті лікарського призначення.

Фізична модель бази даних інформаційної системи електронної комерції мережі аптек реалізовувалася на платформі СУБД MySql. Фізична модель даних подається на рис. 5

Для реалізації EER-моделі інформаційної системи електронної комерції мережі аптек у вигляді бази даних була визначена посилальна цілісність. Посилальна цілісність подається у табл. 3.1

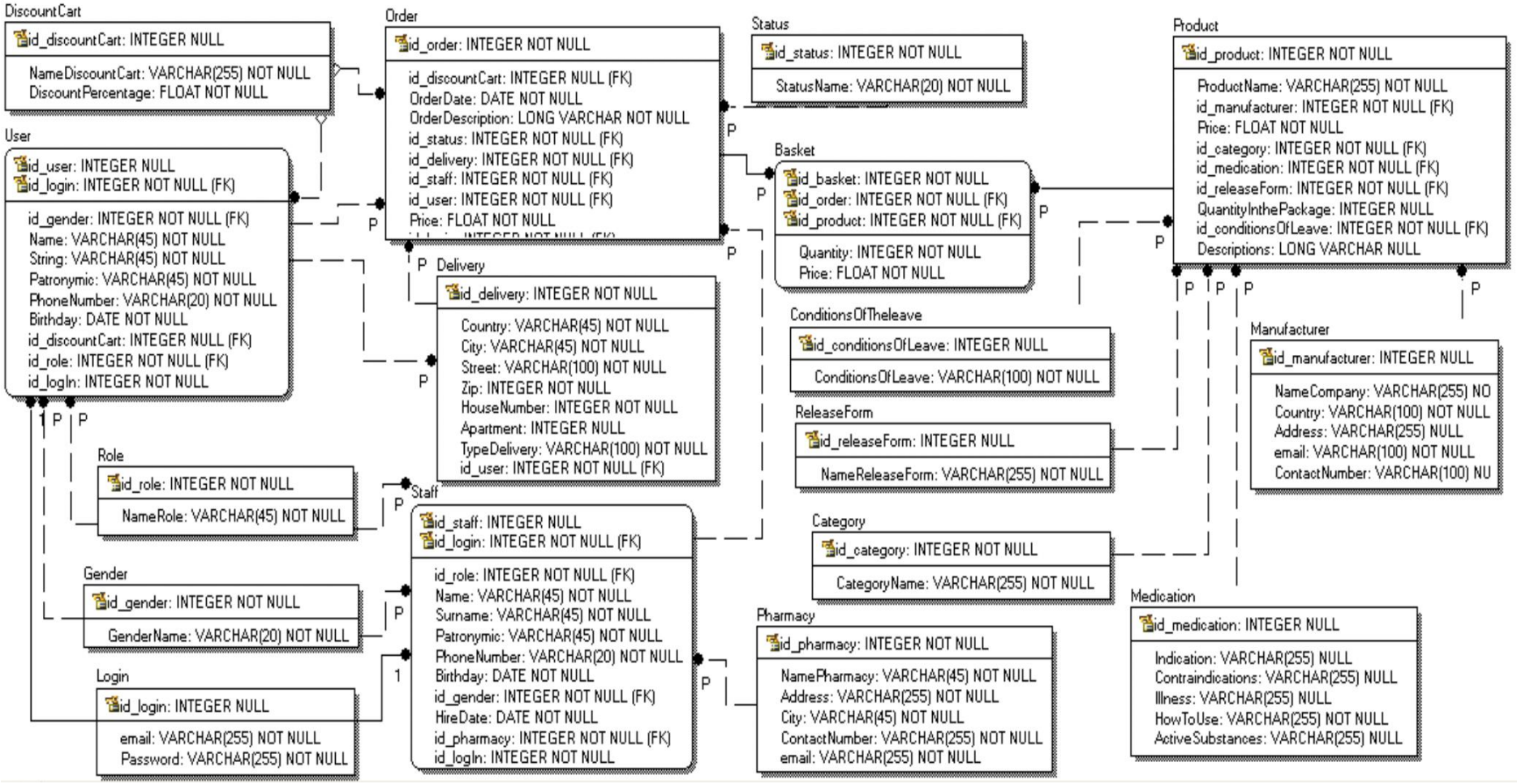

Рисунок 5 – Фізична модель даних інформаційної системи електронної комерції мережі аптек

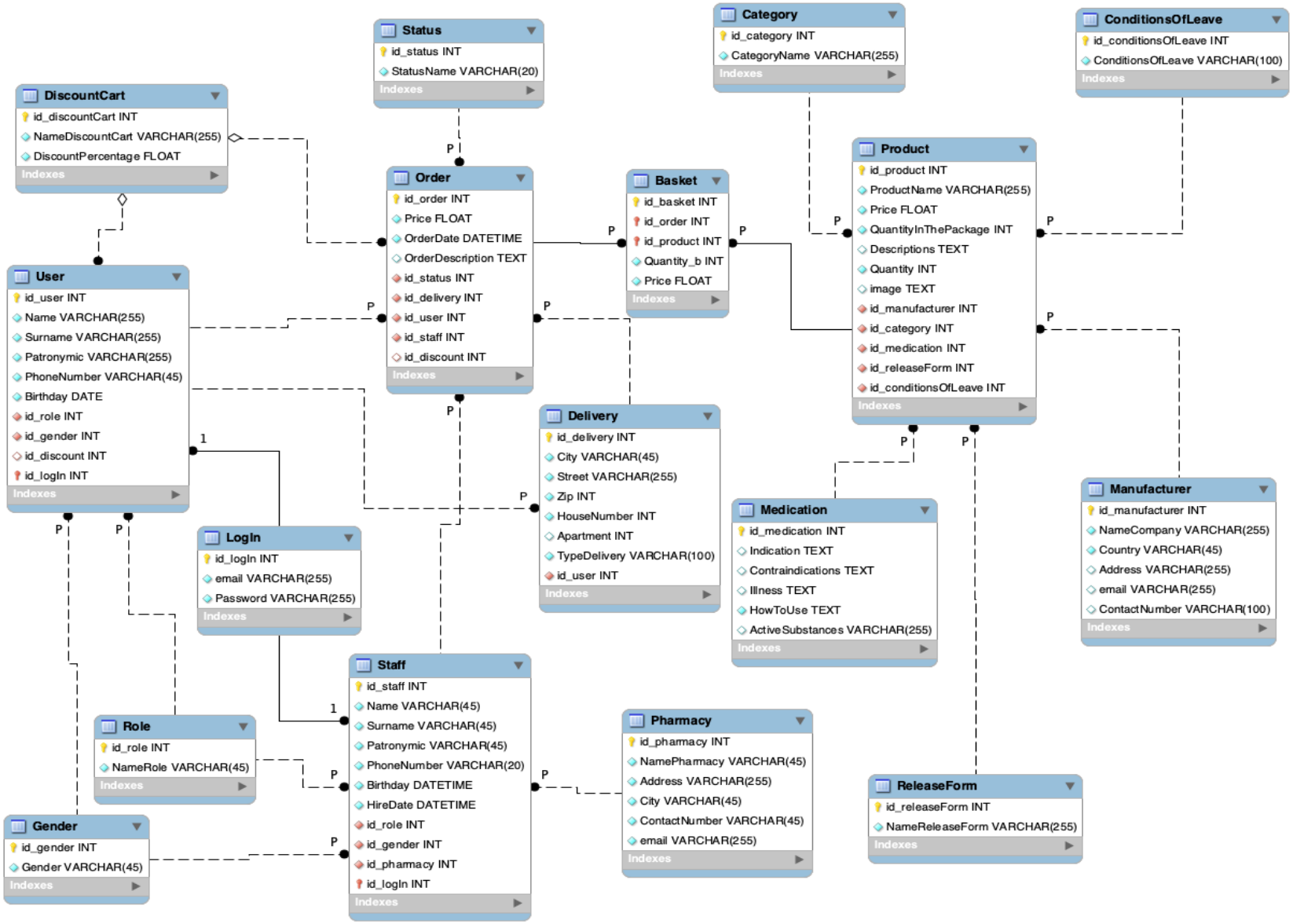

Рисунок 6 – EER- модель даних інформаційної системи електронної комерції мережі аптек

Таблиця 3.1 – Посилальна цілісність зав’язків між таблицями БД інформаційної системи електронної комерції мережі аптек

Таблиця 1	Зовнішній ключ	Таблиця 2	Первинний ключ	ON UPDATE	ON DELETE
User	fk_gender	Gender	id_gender	RESTRICT	RESTRICT
	fk_role	Role	id_role	RESTRICT	RESTRICT
	fk_discount	DiscountCart	id_discountCart	RESTRICT	RESTRICT
	fk_delivery	Delivery	id_delivery	RESTRICT	RESTRICT
LogIn	fk_user	User	id_user	RESTRICT	RESTRICT
	fk_staff	Staff	id_staff	RESTRICT	RESTRICT
Order	fk_staff	Staff	id_staff	RESTRICT	RESTRICT
	fk_user	User	id_user	RESTRICT	RESTRICT
	fk_status	Status	id_status	RESTRICT	RESTRICT
	fk_delivery	Delivery	id_delivery	RESTRICT	RESTRICT
	fk_discount	DiscountCart	id_discountCart	RESTRICT	RESTRICT
Staff	fk_gender	Gender	id_gender	RESTRICT	RESTRICT
	fk_role	Role	id_role	RESTRICT	RESTRICT
	fk_pharmacy	Pharmacy	id_pharmacy	RESTRICT	RESTRICT
Basket	fk_order	Order	id_order	RESTRICT	CASCADE

	fk_product	Product	id_product	RESTRICT	RESTRICT
Product	fk_manufacturer	Manufacturer	id_manufacture	RESTRICT	RESTRICT
	fk_category	Category	id_category	RESTRICT	RESTRICT
	fk_releaseForm	ReleaseForm	id_releaseForm	RESTRICT	RESTRICT
	fk_conditionsOfLeave	ConditionsOfLeave	id_	RESTRICT	RESTRICT
	fk_medication	Medication	id_medication	RESTRICT	RESTRICT

Для усіх зав’язків між таблицями для операцій «ON DELETE» та «ON UPDATE» встановлено тип посилальної цілісності «RESTRICT». Для зв’язку між таблицями «Order» («Замовлення») та «Basket» («Кошик») для операції «ON DELETE» вибрано тип посилальної цілісності «CASCADE». Дане рішення підтримує бізнес-функцію інформаційної системи електронної комерції мережі аптек для випадку коли адміністратор підтверджує замовлення, а користувач потім відмовляється від замовлення, в цьому випадку адміністратор видаляє замовлення з таблиці бази даних «Order» («Замовлення»), після видалення замовлення буде автоматично видалено всю інформацію до замовлення з таблиці бази даних «Basket» («Кошик») в інформаційній системі електронної комерції мережі аптек.

3.3 Аналіз реалізації алгоритму SETM для системи е-комерції з продажу ліків

Розглядається база даних інформаційної системи електронної комерції мережі аптек, EER-модель якої подається на рис. 6. Вибірка даних кошиків замовлень отримується за допомогою створеного уявлення «Order_Product_Products» до зв’язаних таблиць «Order» і «Basket»:

```
CREATE VIEW Order_Product_Products AS
SELECT
    Basket.id_order AS order_id,
    GROUP_CONCAT(Product.id_product ORDER BY Product.id_product) AS
id_products
FROM
    Basket
JOIN
    Product ON Basket.id_product = Product.id_product
GROUP BY
    Basket.id_order
HAVING
    COUNT(Basket.id_product) = 3
    OR COUNT(Basket.id_product) > 3;
```

Результат виконання цього запиту подається в табл. 3.2. Цей запит не використовується для реалізації алгоритму SETM. Це представлення буде містити інформацію про замовлення, де кількість продуктів рівна 3 або більше 3, а ідентифікатори продуктів будуть представлені у вигляді рядка.

Реалізація алгоритму SETM проводиться за схемою, що подається на рис. 3.

Таблиця 3.2 – Результат виконання запиту до уявлення «Order_Product_Products»

order_id	id_products
1	1,2,3
2	2,3,4
3	3,4,5
4	1,3,4
5	3,4,5
6	2,3,4
7	1,3,5
8	1,2,5
9	2,3,5
10	1,2,3
11	1,2,3,4

У якості унікального ідентифікатора транзакції (покупки) order_id використовуються первинний ключ «id_order» таблиці «Order», а для ідентифікації товарів у транзакції – зовнішній ключ «fk_product» таблиці «Basket».

Для реалізації цього запиту створимо представлення SETM_1 за допомогою такого лістингу:

```
CREATE VIEW SETM_1 AS
SELECT
    B1.id_order AS order_id,
    B1.id_product AS id_product_1,
    B2.id_product AS id_product_2,
    B3.id_product AS id_product_3
FROM
    (SELECT DISTINCT id_order FROM Basket) Orders
LEFT JOIN
    Basket B1 ON Orders.id_order = B1.id_order
LEFT JOIN
    Basket B2 ON Orders.id_order = B2.id_order AND B1.id_product < B2.id_product
LEFT JOIN
    Basket B3 ON Orders.id_order = B3.id_order AND B2.id_product < B3.id_product
WHERE
    B1.id_product IS NOT NULL AND B2.id_product IS NOT NULL AND B3.id_product IS NOT NULL;
```

Результат виконання цього запиту подається в табл. 3.3. Це представлення міститиме усі можливі комбінації трьох ідентифікаторів продуктів для кожного замовлення.

Таблиця 3.3 – Результат виконання запиту до уявлення «setm_1»

order_id	id_product_1	id_product_2	id_product_3
1	1	2	3
2	2	3	4
3	3	4	5
4	1	3	4
5	3	4	5

6	2	3	4
7	1	3	5
8	1	2	5
9	2	3	5
10	1	2	3
11	1	2	3
11	1	2	4
11	1	3	4
11	2	3	4

Наступним кроком є створення допоміжного уявлення «setm_2» до вже створенного уявлення «setm_1»:

```
CREATE VIEW SETM_2 AS
SELECT DISTINCT order_id, id_product
FROM (
    SELECT order_id, id_product_1 AS id_product FROM
Order_Product_Products
    UNION ALL
    SELECT order_id, id_product_2 AS id_product FROM
Order_Product_Products
    UNION ALL
    SELECT order_id, id_product_3 AS id_product FROM
Order_Product_Products
) AS UniqueProducts;
```

Результат виконання цього запиту подається в табл. 3.4. Уявлення «setm_2» містить унікальні комбінації замовлення та ідентифікаторів продуктів з усіх трьох стовпців «id_product_1», «id_product_2», «id_product_3» з попереднього представлення «Order_Product_Products».

Таблиця 3.4 – Результат виконання запиту до уявлення «setm_2»

order_id	id_product
1	1
2	2
3	3
4	1
5	3
6	2
7	1
8	1
9	2
10	1
11	1
11	2
1	2
2	3
3	4
4	3
5	4
6	3
7	3
8	2
9	3
10	2
11	3
1	3
2	4
3	5
4	4
5	5
6	4
7	5
8	5
9	5
10	3
11	4

Наступним кроком є створення уявлення «setm_3» до вже створенного уявлень:

```
CREATE VIEW setm_3 AS
SELECT id_product, COUNT(DISTINCT order_id) AS count_in_orders
FROM (
    SELECT order_id, id_product_1 AS id_product FROM SETM_1
    UNION ALL
    SELECT order_id, id_product_2 AS id_product FROM SETM_1
    UNION ALL
    SELECT order_id, id_product_3 AS id_product FROM SETM_1
) AS combined_products
GROUP BY id_product;
```

Результат виконання цього запиту подається в табл. 3.5. Уявлення «setm_3» містить унікальні ідентифікатори продуктів та кількість унікальних замовлень для кожного продукту.

Таблиця 3.5 – Результат виконання запиту до уявлення «setm_3»

id_product	count_in_orders
1	6
2	7
3	10
4	6
5	5

Наступним кроком є створення уявлення «setm_4» до вже створених уявлень:

```
CREATE VIEW setm_4 AS
SELECT
    B1.id_order AS order_id,
    B1.id_product AS product_1,
    B2.id_product AS product_2
FROM (
    SELECT id_order, COUNT(*) AS product_count
```

```
        FROM Basket
        GROUP BY id_order
        HAVING product_count >= 3
    ) Orders
    LEFT JOIN Basket B1 ON Orders.id_order = B1.id_order
    LEFT JOIN Basket B2 ON Orders.id_order = B2.id_order AND
B1.id_product < B2.id_product
    WHERE B1.id_product IS NOT NULL AND B2.id_product IS NOT NULL;
```

Результат виконання цього запиту подається в табл. 3.6. Уявлення «setm_4» містить усі можливі комбінації продуктів («product_1» та «product_2») для кожного «id_order», де кількість продуктів у замовленні дорівнює або більше 3.

Таблиця 3.6 – Результат виконання запиту до уявлення «setm_4»

order_id	product_1	product_2
1	1	2
1	1	3
1	2	3
2	2	3
2	2	4
2	3	4
3	3	4
3	3	5
3	4	5
4	1	3
4	1	4
4	3	4
5	3	4
5	3	5
5	4	5
6	2	3
6	2	4
6	3	4
7	1	3
7	1	5
7	3	5
8	1	2

8	1	5
8	2	5
9	2	3
9	2	5
9	3	5
10	1	2
10	1	3
10	2	3
11	1	2
11	1	3
11	1	4
11	2	3
11	2	4
11	3	4

Наступним кроком є створення уявлення «setm_5» до вже створених уявлень:

```
CREATE VIEW setm_5 AS
SELECT
    product_1,
    product_2,
    COUNT(*) AS count
FROM
    setm_4
GROUP BY
    product_1,
    product_2;
```

Результат виконання цього запиту подається в табл. 3.7. Уявлення «setm_5» містить інформацію про кількість кожної унікальної комбінації продуктів («product_1» та «product_2») з представлення «setm_4».

Таблиця 3.7 – Результат виконання запиту до уявлення «setm_5»

product_1	product_2	count
1	2	4

1	3	5
2	3	6
2	4	3
3	4	6
3	5	4
4	5	2
1	4	2
1	5	2
2	5	2

Наступним кроком є створення уявлення «setm_6» до вже створених уявлень:

```
CREATE VIEW setm_6 AS
SELECT s4.order_id, s4.product_1, s4.product_2
FROM setm_4 s4
JOIN setm_5 s5 ON (s4.product_1 = s5.product_1 AND s4.product_2 = s5.product_2) OR (s4.product_1 = s5.product_2 AND s4.product_2 = s5.product_1)
JOIN (
    SELECT MAX(count) * 0.7 AS threshold
    FROM setm_5
) s5_max ON s5.count >= s5_max.threshold;
```

Результат виконання цього запиту подається в табл. 3.8. Уявлення «setm_6» містить комбінації продуктів з «setm_4», які відповідають визначеному пороговому значенню, а саме в 70%.

Таблиця 3.8 – Результат виконання запиту до уявлення «setm_6»

order_id	product_1	product_2
11	1	3
10	1	3
7	1	3
4	1	3
1	1	3

11	2	3
10	2	3
9	2	3
6	2	3
2	2	3
1	2	3
11	3	4
6	3	4
5	3	4
4	3	4
3	3	4
2	3	4

Наступним кроком є створення уявлення «setm_7» до вже створених уявлень:

```
CREATE VIEW setm_7 AS
SELECT
    B1.id_order AS order_id,
    B1.id_product AS product_1,
    B2.id_product AS product_2,
    B3.id_product AS product_3
FROM
    (SELECT DISTINCT id_order FROM Basket) Orders
LEFT JOIN
    Basket B1 ON Orders.id_order = B1.id_order
LEFT JOIN
    Basket B2 ON Orders.id_order = B2.id_order AND B1.id_product < B2.id_product
LEFT JOIN
    Basket B3 ON Orders.id_order = B3.id_order AND B2.id_product < B3.id_product
WHERE
    B1.id_product IS NOT NULL AND B2.id_product IS NOT NULL AND B3.id_product IS NOT NULL;
```

Результат виконання цього запиту подається в табл. 3.9. Уявлення «setm_7» містить унікальні комбінації трьох продуктів для кожного унікального «id_order».

Таблиця 3.9 – Результат виконання запиту до уявлення «setm_7»

order_id	product_1	product_2	product_3
1	1	2	3
2	2	3	4
3	3	4	5
4	1	3	4
5	3	4	5
6	2	3	4
7	1	3	5
8	1	2	5
9	2	3	5
10	1	2	3
11	1	2	3
11	1	2	4
11	1	3	4
11	2	3	4

Наступним кроком є створення уявлення «setm_8» до вже створених уявлень:

```
CREATE VIEW setm_8 AS
SELECT s7.product_1, s7.product_2, s7.product_3, COUNT(*) AS count
FROM setm_7 s7
GROUP BY s7.product_1, s7.product_2, s7.product_3;
```

Результат виконання цього запиту подається в табл. 3.10. Уявлення «setm_8» містить унікальні комбінації трьох продуктів із «setm_7» та кількість їх входжень в усіх «id_order».

Таблиця 3.10 – Результат виконання запиту до уявлення «setm_8»

product_1	product_2	product_3	count
1	2	3	3

2	3	4	3
3	4	5	2
1	3	4	2
1	3	5	1
1	2	5	1
2	3	5	1
1	2	4	1

Останнім кроком є створення уявлення «setm_9» до вже створених уявлень:

```
CREATE VIEW setm_9 AS
SELECT s7.order_id, s7.product_1, s7.product_2, s7.product_3
FROM setm_7 s7
JOIN setm_8 s8
    ON (s7.product_1 = s8.product_1 OR s7.product_1 = s8.product_2 OR
s7.product_1 = s8.product_3)
    AND (s7.product_2 = s8.product_1 OR s7.product_2 = s8.product_2
OR s7.product_2 = s8.product_3)
    AND (s7.product_3 = s8.product_1 OR s7.product_3 = s8.product_2
OR s7.product_3 = s8.product_3)
JOIN (
    SELECT MAX(count) * 0.7 AS threshold
    FROM setm_8
) s8_max ON s8.count >= s8_max.threshold;
```

Результат виконання цього запиту подається в табл. 3.11. Уявлення «setm_9» містить унікальні комбінації трьох продуктів з «setm_7», які відповідають умові входження принаймні 70% від максимальної кількості входжень.

Таблиця 3.11 – Результат виконання запиту до уявлення «setm_9»

order_id	product_1	product_2	product_3
11	1	2	3
10	1	2	3
1	1	2	3
11	2	3	4

6	2	3	4
2	2	3	4

Для кращого розуміння роботи алгоритму SETM з БД інформаційної системи електронної комерції мережі аптек наведено послідовність виконання запитів до створених уявлень на рис. 7

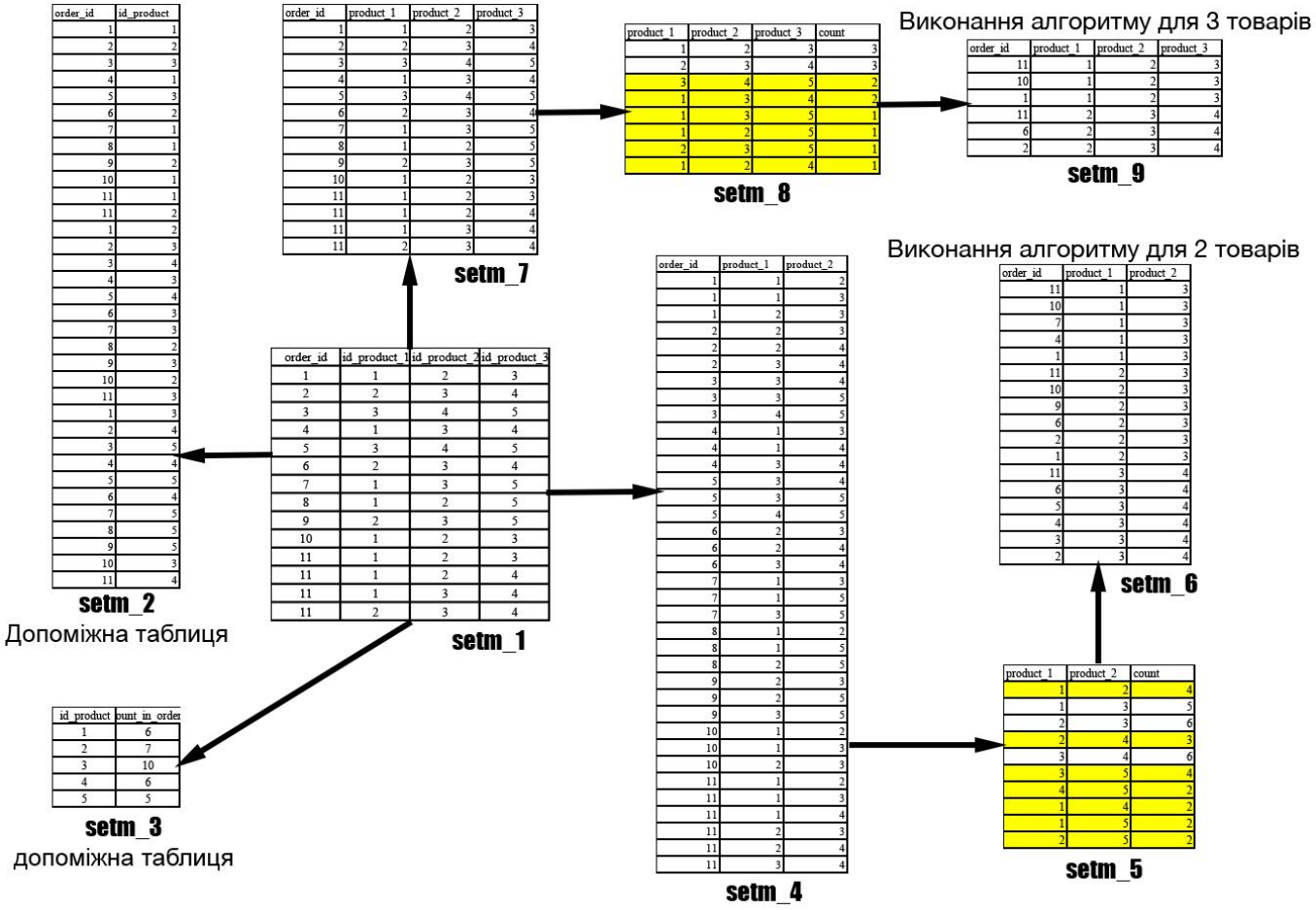

Рис. 7 – Реалізація алгоритму SETM

3.4 Аналіз реалізації алгоритму AIS для системи е-комерції з продажу ліків

Розглядається база даних інформаційної системи електронної комерції мережі аптек, EER-модель якої подається на рис. 6.

Вибірка даних кошиків замовлень отримується за допомогою створеного уявлення «Order_Product_Products» до зв’язаних таблиць «Order» і «Basket»:

```
CREATE VIEW Order_Product_Products AS
SELECT
    Basket.id_order AS order_id,
    GROUP_CONCAT(Product.id_product ORDER BY Product.id_product) AS
id_products
FROM
    Basket
JOIN
    Product ON Basket.id_product = Product.id_product
GROUP BY
    Basket.id_order
HAVING
    COUNT(Basket.id_product) = 3
    OR COUNT(Basket.id_product) > 3;
```

Результат виконання цього запиту подається в табл. 3.12. Цей запит не використовується для реалізації алгоритму AIS. Це представлення буде містити інформацію про замовлення, де кількість продуктів рівна 3 або більше 3, а ідентифікатори продуктів будуть представлені у вигляді рядка.

Реалізація алгоритму SETM проводиться за схемою, що подається на рис. 3.

Таблиця 3.12 – Результат виконання запиту до уявлення «Order_Product_Products»

order_id	id_products
1	1,2,3
2	2,3,4
3	3,4,5
4	1,3,4
5	3,4,5
6	2,3,4
7	1,3,5
8	1,2,5
9	2,3,5
10	1,2,3
11	1,2,3,4

У якості унікального ідентифікатора транзакції (покупки) order_id використовуються первинний ключ «id_order» таблиці «Order», а для ідентифікації товарів у транзакції – зовнішній ключ «fk_product» таблиці «Basket».

Для реалізації запиту створимо уявлення «ais_1»:

```
CREATE VIEW ais_1 AS
SELECT p.id_product, COUNT(*) AS product_count
FROM Product p
INNER JOIN order_product_products op ON FIND_IN_SET(p.id_product, op.id_products) > 0
GROUP BY p.id_product;
```

Результат виконання цього запиту подається в табл. 3.13. Уявлення «ais_1» містить унікальні ідентифікатори продуктів та кількість унікальних замовлень для кожного продукту.

Таблиця 3.5 – Результат виконання запиту до уявлення «ais_1»

id_product	product_count
3	10
2	7
1	6
4	6
5	5

Наступним кроком є створення уявлення «ais_2» до вже створених уявлень:

```
CREATE VIEW `ais_2` AS
SELECT
   CONCAT('(',
       LEAST(p1.id_product, p2.id_product),
       ', ',
```

```
            GREATEST(p1.id_product, p2.id_product), ')'
        ) AS product_pair,
        COUNT(*) AS occurrence_count,
        CONCAT(
            ROUND(COUNT(*) * 100 / (
                SELECT MAX(occurrence_count) FROM (
                    SELECT
                        CONCAT('(',
                            LEAST(p1.id_product, p2.id_product),
                            ', ',
                            GREATEST(p1.id_product, p2.id_product), ')'
                        ) AS product_pair,
                        COUNT(*) AS occurrence_count
                    FROM
                        Product p1
                    JOIN
                        Product p2
                    JOIN
                        order_product_products op1 ON
FIND_IN_SET(p1.id_product, op1.id_products) > 0
                    JOIN
                        order_product_products op2 ON
FIND_IN_SET(p2.id_product, op2.id_products) > 0 AND op1.order_id =
op2.order_id
                    WHERE
                        p1.id_product < p2.id_product
                    GROUP BY
                        product_pair
                ) AS max_occurrences
            ), 2),
            '%'
        ) AS support
    FROM
        Product p1
    JOIN
        Product p2
    JOIN
        order_product_products op1 ON FIND_IN_SET(p1.id_product,
op1.id_products) > 0
    JOIN
        order_product_products op2 ON FIND_IN_SET(p2.id_product,
op2.id_products) > 0 AND op1.order_id = op2.order_id
    WHERE
        p1.id_product < p2.id_product
    GROUP BY
```

```
product_pair;
```

Результат виконання цього запиту подається в табл. 3.14. Уявлення «ais_2» містить розрахунок кількості входжень комбінації: здійснює підрахунок кількості входжень кожної комбінації в таблиці. Здійснює розрахунок відсоткової підтримки кожної комбінації відносно максимальної кількості входжень комбінації в інших записах. Результат групується за комбінацією product_pair.

Таблиця 3.14 – Результат виконання запиту до уявлення «ais_2»

product_pair	occurrence_count	support
(1, 2)	4	66.67%
(1, 3)	5	83.33%
(1, 4)	2	33.33%
(1, 5)	2	33.33%
(2, 3)	6	100.00%
(2, 4)	3	50.00%
(2, 5)	2	33.33%
(3, 4)	6	100.00%
(3, 5)	4	66.67%
(4, 5)	2	33.33%

Наступним кроком є створення уявлення «ais_3» до вже створених уявлень:

```
CREATE VIEW ais_3 AS
SELECT
   CONCAT(
       '(',
       LEAST(p1.id_product, p2.id_product, p3.id_product),
       ', ',
       p1.id_product + p2.id_product + p3.id_product -
LEAST(p1.id_product, p2.id_product, p3.id_product) -
GREATEST(p1.id_product, p2.id_product, p3.id_product),
       ', ',
       GREATEST(p1.id_product, p2.id_product, p3.id_product),
       ')'
   ) AS product_combination,
```

```
        COUNT(*) AS occurrence_count,
        CONCAT(
            ROUND(COUNT(*) * 100 / (
                SELECT MAX(occurrence_count) FROM (
                    SELECT
                        CONCAT(
                            '(',
                            LEAST(p1.id_product, p2.id_product,
p3.id_product),
                            ', ',
                            p1.id_product + p2.id_product + p3.id_product
- LEAST(p1.id_product, p2.id_product, p3.id_product) -
GREATEST(p1.id_product, p2.id_product, p3.id_product),
                            ', ',
                            GREATEST(p1.id_product, p2.id_product,
p3.id_product),
                            ')'
                        ) AS product_combination,
                        COUNT(*) AS occurrence_count
                    FROM
                        order_product_products op
                    JOIN
                        Product p1 ON FIND_IN_SET(p1.id_product,
op.id_products) > 0
                    JOIN
                        Product p2 ON FIND_IN_SET(p2.id_product,
op.id_products) > 0
                    JOIN
                        Product p3 ON FIND_IN_SET(p3.id_product,
op.id_products) > 0
                    WHERE
                        p1.id_product < p2.id_product AND p2.id_product <
p3.id_product
                    GROUP BY
                        product_combination
                ) AS max_occurrences
            ), 2),
            '%'
        ) AS support
    FROM
        order_product_products op
    JOIN
        Product p1 ON FIND_IN_SET(p1.id_product, op.id_products) > 0
    JOIN
        Product p2 ON FIND_IN_SET(p2.id_product, op.id_products) > 0
```

```
JOIN
   Product p3 ON FIND_IN_SET(p3.id_product, op.id_products) > 0
WHERE
   p1.id_product < p2.id_product AND p2.id_product < p3.id_product
GROUP BY
   product_combination;
```

Результат виконання цього запиту подається в табл. 3.15. Уявлення «ais_3» містить комбінації трьох унікальних id_product. Також обчислюється кількість входжень кожної комбінації у таблиці order_product_products. Також обчислюється підтримка (support) у відсотках для кожної комбінації відносно найбільшої кількості входжень серед всіх комбінацій.

Таблиця 3.15 – Результат виконання запиту до уявлення «ais_2»

product_combination	occurrence_count	support
(1, 2, 3)	3	100.00%
(2, 3, 4)	3	100.00%
(3, 4, 5)	2	66.67%
(1, 3, 4)	2	66.67%
(1, 3, 5)	1	33.33%
(1, 2, 5)	1	33.33%
(2, 3, 5)	1	33.33%
(1, 2, 4)	1	33.33%

Для кращого розуміння роботи алгоритму AIS з БД інформаційної системи електронної комерції мережі аптек наведено послідовність виконання запитів до створених уявлень на рис. 8

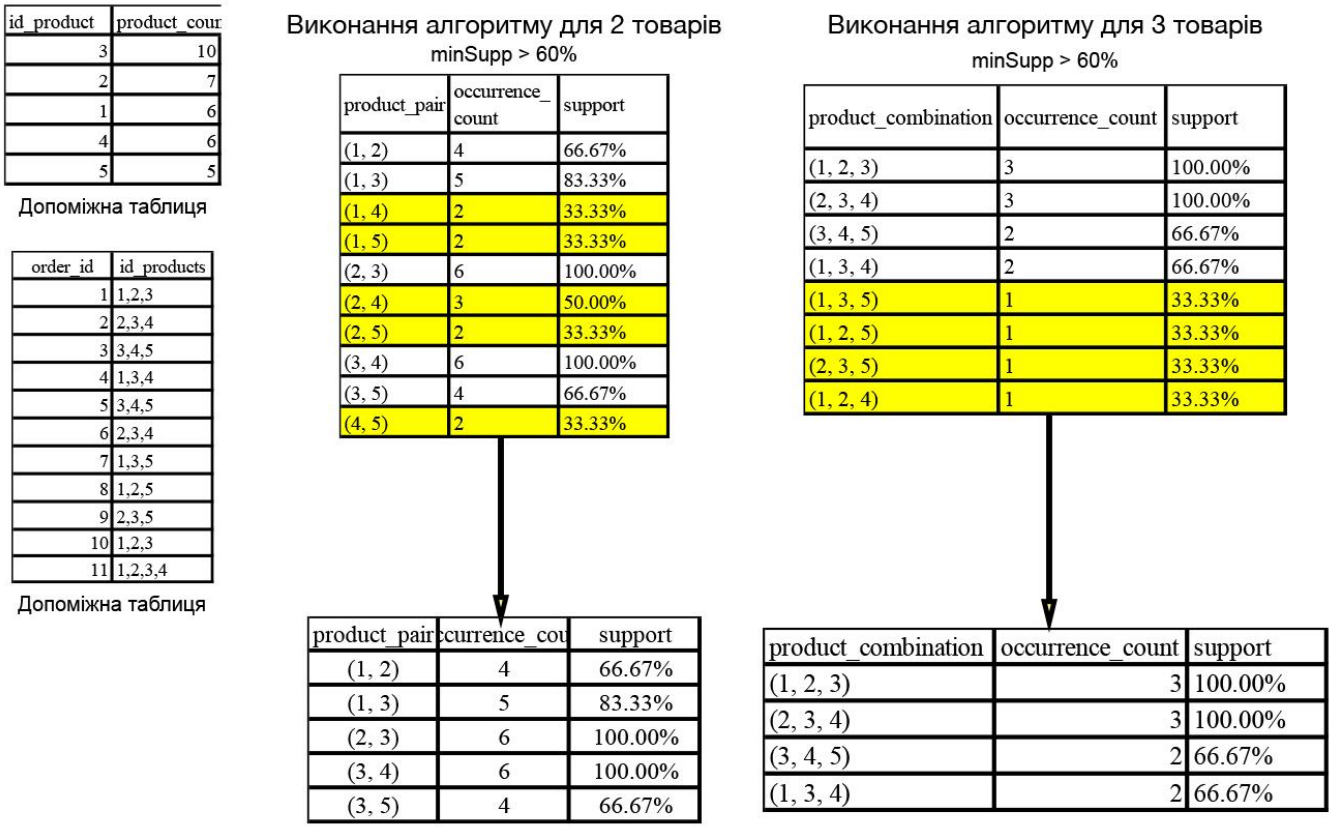

Рис. 8 – Реалізація алгоритму AIS

3.5 Порівняльний аналіз алгоритмів мінінгу для системи е-комерції з продажу ліків

Вибір між SETM та AIS залежить від конкретних вимог та характеристик системи е-комерції. Обидва алгоритми можуть бути використані для виявлення закономірностей у покупках та підвищення ефективності бізнес-процесів. У цьому контексті, застосування алгоритмів мінінгу даних, таких як SETM та AIS, може сприяти оптимізації процесів та покращенню конкурентоспроможності.

SETM є алгоритмом, спрямованим на виявлення послідовних шаблонів у транзакційних даних. В контексті системи е-комерції, SETM може бути використаний для аналізу послідовностей покупок клієнтів, допомагаючи підприємству прогнозувати та адаптувати свої стратегії продажу. Однією з переваг SETM є його здатність виявляти послідовні покупки, що є важливим для системи

електронної комерції з продажу ліків. Однак, можливі обмеження ефективності при обробці великого обсягу даних.

AIS використовує алгоритм Apriori для виявлення асоціативних правил між продуктами на основі їхнього входження у корзини покупок. У випадку е-комерції з продажу ліків, AIS може допомагати виявляти зв'язки між різними медичними препаратами та оптимізувати рекомендації для клієнтів. AIS, використовуючи принцип Apriori, дозволяє виявляти асоціативні правила, що вказують на зв'язки між різними продуктами. У випадку е-комерції з продажу ліків, AIS може ефективно оптимізувати рекомендації та управління запасами.

В порівнянні швидкості алгоритмів мінінгу даних, таких як SETM та AIS, існує кілька ключових факторів, які слід врахувати. SETM може виявитися менш ефективним, якщо об'єм транзакційних даних великий, оскільки він працює з послідовностями та вимагає багато операцій порівняння. Також він вимагає точної послідовності, що може бути складним у великих наборах даних. Оптимізація може бути складною. Може вимагати значних обсягів пам'яті та ресурсів, особливо при роботі з великими послідовностями. Алгоритм AIS також може зазнавати затримок при обробці великих об'ємів даних, але його ефективність покращується за допомогою покращених версій. Він добре підходить для асоціативного аналізу великих корзин товарів, де важливіше виявлення асоціацій між продуктами. Є можливість оптимізації шляхом використання покращених алгоритмів та структур даних для підтримки ефективного аналізу. Також алгоритм AIS можливість працювати ефективно з обмеженими ресурсами.

Якщо ж порівнювати алгоритми на одних з самих завантажених запитів до експериментальної БД інформаційної системи електронної комерції мережі аптек, яка наповнена малою кількістю даних, то маємо результати які показані на рис. 9.

	Time	Action	Response	Duration / Fetch Time
1	00:35:03	SELECT * FROM Pharmacy.setm_8 LIMIT 0, 1000	8 row(s) returned	0.00085 sec / 0.0000091 sec
2	00:35:05	SELECT * FROM Pharmacy.ais_3 LIMIT 0, 1000	8 row(s) returned	0.0035 sec / 0.0000072 sec

Рис. 9 – Порівняльний аналіз алгоритмів AIS та SETM на подібних запитах до БД

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було реалізовано та проаналізовано алгоритми мінінгу AIS та SETM для пошуку асоціацій на платформі СУБД MySQL для інформаційної системи електронної комерції мережі аптек.

Були розглянуті такі питання:

– аналіз предметної області з використанням алгоритмів мінінгу;

– призначення алгоритмів мінінгу;

– аналіз варіантів реалізації та вибір предметної області для реалізації алгоритмів мінінгу на платформі СУБД MySQL;

– постановка задачі на реалізацію алгоритмів мінінгу на платформі СУБД MySQL для інформаційної системи електронної комерції мережі аптек.

У першому розділі кваліфікаційної роботи «Аналіз предметної області» було вирішено реалізувати два алгоритми мінінгу на платформі СУБД MySQL шляхом аналізу розповсюджених алгоритмів мінінгу.

У другому розділі кваліфікаційної роботи «Призначення алгоритмів мінінгу» було розглянуто та проаналізовано три алгоритми мінінгу, а саме: AIS, SETM та Apriori. За висновками другого розділу вирішено реалізувати алгоритми AIS та SETM на платформі СУБД MySQL.

У третьому розділі кваліфікаційної роботи «Аналіз варіантів реалізації та вибір предметної області для реалізації алгоритмів мінінгу на платформі СУБД MySQL» в ході вибору предметної області вирішено обрати предметну область «Інформаційна система електронної комерції мережі аптек». Також проаналізовано реалізацію алгоритмів мінінгу AIS та SETM на платформі СУБД MySQL для обраної предметної області.

Отже, реалізовано та проаналізовано два алгоритми мінінгу, а саме, AIS та SETM на платформі СУБД MySQL для інформаційної системи електронної комерції мережі аптек.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Офіціальний сайт компанії Microsoft. Загальний огляд програмної платформи SQL Server Data Tools. URL: https://msdn.microsoft.com/ru-ru/library/hh272686(v=vs.103).aspx/

2. Офіціальний сайт компанії Oracle. Загальний огляд програмної платформи Oracle Advanced Analytics Machine Learning Algorithms SQL Functions. URL: https://www.oracle.com/technetwork/database/enterprise-edition/ odm-techniqu es-algorithms-097163.html

3. Jorge B. Bocca, Matthias Jarke, Carlo Zaniolo (Eds.). *Proceedings of the 20th International Conference on Very Large Data Bases, (VLDB'94), September 12–15*.1994. Santiago de Chile, Chile. Morgan Kaufmann, ISBN 1-55860-153-8. URL: https://www.vldb.org/conf/1994/P487.pdf

4. Margaret Dunham, Yongqiao Xiao Le Gruenwald, Zahid Hossain. A survey of association rules. 2001. URL: https://www.researchgate.net/publication/240737786_A_SURVEY_OF_ASSOCIATION_RULES#fullTextFileContent

5. Trupti A., Santosh V. An Overview of Association Rule Mining Algorithms. *International Journal of Computer Science and Information Technologies (IJCSIT)*.2014. Vol. 5(1). P. 927-930. ISSN:0975-9646. URL: https://www.ijcsit.com/docs/ Volume%205/vol5issue01/ijcsit20140501201.pdf

6. Fast Algorithms for Mining Association Rules. Rakesh Agrawal Ramakrishnan Srikant. IBM Almaden Research Center 650. Harry Road, San Jose, CA 95120

7. Барсегян, А. А. Б26 Анализ данных и процессов: учеб. пособие / А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2009. — 512 с.: ил. + CD-ROM — (Учебная литература для вузов) ISBN 978-5-9775-0368-6

8. Bei-Bei Cui. Design and Implementation of Movie Recommendation System Based on Knn Collaborative Filtering Algorithm. The 4th Annual International

Conference on Information Technology and Applications (ITA 2017.) Session 4: Information Theory and Information Systems –January 2017. 2017. Vol. 12(04008). P. 1–5. DOI: https://doi.org/10.1051/itmconf/20171204008

9. Gagandeep Kaur, Shruti Aggarwal , Performance Analysis of Association Rule Mining Algorithms, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, August 2013, ISSN: 2277 128X

10. Довідник Workbench [Електронний ресурс] - Режим доступу URL: https:// dev.mysql.com/doc/workbench/en/

11. Маклаков, С. В. BPwin та ERwin. CASE-засоби розробки інформаційних систем [Текст]/С. В. Маклаков. – М.: Діалог, 1999. – 256 с.: іл.;

12. Set-oriented mining for association rules in relational databases. University of Twente the Netherlands, IBM Almaden Research Center San Jose, CA 95120. Conference: Data Engineering, 1995. Proceedings of the Eleventh International Conference on. DOI: https://www.researchgate.net/publication/3597730_Set-oriented_mining_for_association_rules_in_relational_databases

13. Association Rule Mining Algorithm’s Variant Analysis. Prince Verma, Assistant Professor CSE Dept. CT Institute of Engg., Mgt. & Tech. Punjab, India. Dinesh Kumar, Associate Professor IT Dept. DAV Institute of Eng. & Tech. Punjab, India. International Journal of Computer Applications (0975 – 8887) Volume 78 – No.14, September 2013