Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Автоматики та комп'ютеризованих технологій*
(повна назва)
Кафедра *Системотехніки*
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти *другий (магістерський)*
(освітньо-кваліфікаційний рівень)

Розробка та дослідження підсистеми підтримки освітньої діяльності в навчальних закладах України І-ІІ ступенів
(тема)

Виконала: студентка *6* курсу, групи *КСУАм-18-1*
Булавіна А.М.
(прізвище, ініціали)

Спеціальність *151 – Автоматизація та комп'ютерно-інтегровані технології*
(код і повна назва спеціальності)

Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Комп'ютеризовані системи управління та автоматика*
(повна назва освітньої програми)

Керівник *проф. Колесник Л.В.*
(прізвище, ініціали)

Допускається до захисту

Зав. катедри СТ _____ *проф. Гребеннік І.В.*
(підпис) (прізвище, ініціали)

2019 р

Харківський національний університет радіоелектроніки

Факультет *Автоматики та комп'ютеризованих технологій*
(повна назва)

Катедра *Системотехніки*
(повна назва)

Рівень вищої освіти *другий (магістерський)*

Спеціальність *151 – Автоматизація та комп'ютерно-інтегровані технології*
(код і повна назва)

Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Комп'ютеризовані системи управління та автоматика*
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

" _____ " _____ *2019 р.*

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентці Булавіній Анастасії Миколаївні
(прізвище, ім'я, по батькові)

1. Тема роботи *Розробка та дослідження підсистеми підтримки освітньої діяльності в навчальних закладах України І-ІІ ступенів*
затверджена наказом по університету від " *04* " *листопада* 2019р. № *1629 Ст*

2. Термін подання студентом роботи _____

3. Вихідні дані до роботи (проекту) *Розробити підсистему підтримки освітнього процесу в рамках системи управління діяльністю навчальних закладів України І–ІІ ступенів. Розроблений компонент системи повинен представляти собою частину проекту клієнтського web-додатка з інтерфейсом доступу до бази даних. Перелік використовуваних програмних засобів: ОС Microsoft Windows 7,8,10, Visual Studio 2012, 2015, 2017. Технічне забезпечення: ІВМ-сумісний ПК з МП Core 2 Duo та вище*

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити)
4.1 Аналіз проблеми підтримки освітнього процесу в загальноосвітніх навчальних закладах. 4.2 Аналіз та огляд реалізованої інформаційної системи підтримання навчального процесу 4.3 Визначення сфери застосування підсистеми підтримки освітньої діяльності 4.4 Доповнення діаграми варіантів використання. 4.5 Обґрунтування вибору алгоритму пошуку асоціативних правил 4.6 Визначення вимог до апаратної частини інформаційної системи. 4.7. Обґрунтування вибору СУБД. 4.8 Опис структури розробленої інформаційної системи. 4.9 Опис апаратної частини інформаційної системи. 4.10 Логічне і фізичне моделювання даних інформаційної системи. 4.11 Створення бази даних для «Microsoft SQL Server. 4.12 Тестування розробленого програмного забезпечення 4.13 Висновки 4.14 Перелік посилань 4.15 Додатки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, плакатів)
5.1 *склад компонентів іс підтримки управління навчальним процесом (1 аркуш формату А4)* ***5.2*** *діаграма варіантів використання підсистеми (1 аркуш формату А4).* ***5.3*** *послідовність операцій алгоритму apriori.* ***5.4*** *розрахунок підтримки для 1-елементних кандидатів (1*

аркуш формату А4). ***5.5*** *1-елементні кандидати, що відповідають заданій умові* ***5.6*** *Рівень підтримки для 2-елементних кандидатів* ***5.7*** *База даних 3-елементних кандидатів* ***5.8*** *структура частини підсистеми, що відповідає за видачу рекомендацій стосовано корегування навчального процесу.* ***5.9*** *Діаграма класів.* ***5.10*** *Екранна форма «розклад занять».* ***5.11*** *Інформаційно логічна модель web-порталу*

6. Консультанти з роботи із зазначенням розділів роботи, що їх стосуються

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		(підпис)	(дата)
Спеціальна частина	*проф. Колесник Л.В.*		

7. Дата видачі завдання " 2 " вересня 2019 р.

Керівник роботи _____ проф. Колесник Л.В.
(підпис)

Завдання прийняла до виконання _____ Булавіна А.М.
(підпис студентки)

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів атестаційної роботи	Термін виконання етапів роботи	Примітка
1.	*Отримання завдання атестаційної роботи*	*1.09.2019*	
2.	*Аналіз завдання, літератури та аналогів з теми атестаційної роботи*	*2.09.2019 –14.09.2019*	
3.	*Вибір засобів для розробки технічних вимог до програми*	*15.09.2019 – 5.10.2019*	
4.	*Вибір середовища розробки програми*	*6.10.2019 – 9.10.2019*	
5.	*Аналіз вхідних даних. Побудова моделей*	*10.10.2019 – 24.10.2019*	
6.	*Розробка програми*	*25.10.2019 – 6.11.2019*	
7.	*Тестування програми*	*7.11.2019 – 14.11.2019*	
8.	*Розробка «Посібника користувача»*	*15.11.2019 – 17.11.2019*	
9.	*Оформлення пояснювальної записки та програмної документації*	*18.11.2019 – 5.12.2019*	
10.	*Оформлення графічної частини та презентаційних матеріалів комп'ютерного захисту*	*6.12.2019*	
11.	*Представлення на рецензування*	*16.12.2019*	
12.	*Представлення атестаційної роботи в ДЕК*	*18.12.2019*	

Студентка _____ Булавіна А.М.
(підпис)

Керівник роботи _____ проф. Колесник Л.В.
(підпис)

РЕФЕРАТ

Пояснювальна записка містить 91 аркушів, 21 рисунків, 22 таблиці, 11 додатків, 29 джерел; графічний матеріал – 12 аркушів.

АСОЦІАЦІЯ, НАВЧАЛЬНИЙ ПРОЦЕС, УПРАВЛІННЯ, АВТОМАТИЗАЦІЯ, WEB-СИСТЕМА, БАЗА ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗОЮ ДАНИХ, СЕРВЕР, SQL, C#

Об’єкт досліджень – підтримка освітньої діяльності в навчальних закладах.

Предмет досліджень – методи та засоби підтримки освітньої діяльності в навчальних закладах.

Метою роботи дослідження є застосування та реалізація алгоритму пошуку асоціативних правил для аналізу успішності учнів загальноосвітнього закладу.

Методи дослідження – системний підхід, методи пошуку асоціативних правил, методи структурованого моделювання реляційних баз даних та об'єктно-орієнтованого програмування на мові програмування C #.

Результати роботи – підсистеми, що представляють собою компонент системи, реалізованого у вигляді клієнтського web-додатка з інтерфейсом доступу до бази даних, що дозволяють проводити інтелектуальний аналіз даних та знаходити закономірності в результатах навчальної діяльності учнів.

Область застосування – навчальні заклади України I-II ступенів.

ABSTRACT

Explanatory note contains 91 sheets, 21 figures, 22 tables, 5 applications, 29 sources; graphic material – 12 sheets.

ASSOCIATION, EDUCATIONAL PROCESS, MANAGEMENT, AUTOMATION, WEB SYSTEM, DATABASE, DATABASE MANAGEMENT, SERVER, SQL, C #

The object of research is to support educational activities in educational institutions.

The subject of research is methods and means of supporting educational activities in educational institutions.

The purpose of the research is to apply and implement an algorithm for finding associative rules for analyzing the success of students of a comprehensive institution.

Research Methods – a systematic approach, methods for finding associative rules, methods for structured modeling of relational databases, and object-oriented programming in the C # programming language.

Outputs are subsystems that are a component of a system implemented as a client-based web-application with a database access interface that allow for data mining and finding patterns in the learning outcomes of students.

Scope – educational institutions of Ukraine I-II degrees.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БД – база даних;

ІС – інформаційна система

СУБД – система управління базами даних;

ASP – Active Server Pages (активна серверна сторінка);

CASE – Computer Aided Software Engineering (сукупність методів та засобів автоматизованого проектування інформаційних систем);

COM – Component Object Model (компонент, заснований на об'єктній моделі);

HTML – HyperText Markup Language (мова гипертекстової розмітки);

HTTP – HyperText Transfer Prorocol (протокол передачі гіпертекстових файлів);

IIS – Internet Information Services (web-сервер);

MS – Microsoft;

SQL – Structured Query Language (мова структуованих запитів);

URL – Uniform Resource Locator (уніфікований показчик інформаційного ресурсу).

ВСТУП

Актуальність роботи. Інтелектуалізація є одним з перспективних напрямів розвитку інформаційних систем. Інтелектуальні технології дозволяють вирішувати складні задачі, що погано формалізуються, можливість самонавчання, адаптивність та інше.

Використання інформаційних систем з елементами інтелектуальності в сфері освіти може допомогти зробити процес навчання більш ефективним за рахунок підвищення якості підтримки освітньої діяльності. Предметом інтелектуалізації можуть стати знання про навчальні дисципліни; знання про тих, хто навчається; методики навчання; практичні задачі; рішення, що приймаються в умовах невизначеності; рішення, що приймаються в умовах частково неспівпадаючих інтересів та цілей в освітньому процесі.

Предметом розробки є компонент інформаційної системи підтримки освітньої діяльності з елементами інтелектуальності, а саме – підсистеми інтелектуального аналізу даних навчального процесу, завдання якої – виявлення закономірностей в даних про навчальний процес які напряму пов’язані із залежністю успішності учнів від характеристик та особливостей навчального процесу в загальноосвітніх навчальних закладах України.

Актуальність роботи зумовлена з появою комп’ютерних технологій майже всіх сферах людської діяльності, на поточний момент кожна сучасна людина має доступ до Інтернету за допомогою комп’ютера або мобільного пристрою, що дає можливість працювати з інформацією в будь-який момент часу. Все частіше різні процеси, що виконувались вручну працівниками заміняються автоматичними або автоматизованими системами, що дозволяє скоротити витрати на ресурси, в тому числі і людські.

Застосування інформаційних технологій в навчальних закладах – це перспективний розвиток освітньої діяльності та її вдосконалення. Адже паперовий документообіг до цих пір використовується як основний, а

працівник сфери освіти витрачають багато часу на ведення рукописної документації, підрахунків для оцінювання знань учнів, складання звітів. Кожен вчитель повинен щодня обробляти вручну великі обсяги поточних оцінок, а що складніше – розрахунки підсумкових результатів. Ще одним недоліком такого виду ведення документації – це не можливість колективного користування даними, наприклад, класний керівник не може переглянути класний журнал в той час, поки йде урок і журнал знаходиться в іншого вчителя, або батьки не можуть переглянути оцінки свого учня за семестр поки від не принесе додому табель з оцінками. Звичайно, що в деяких випадках вчителі пристосувались проводити розрахунки підсумкових оцінок за допомогою табличних процесорів за допомогою комп'ютера, але нікуди не зникає потреба переносу даних з папіряного журналу в електронну таблицю.

Для розв'язання подібного роду труднощів існують веб-технології, що дозволяють реалізувати вирішення вищеперерахованих незручностей і полегшити навчальний процес принаймні усуненням рутинної роботи, що забирає багато ресурсів, а також об'єднати основні процеси навчальної діяльності в одному місці - інформаційній системі.

Мета та задачі досліджень. Метою роботи є розробка підсистеми підтримки освітньої діяльності із застосування методу пошуку асоціативних правил для аналізу успішності учнів загальноосвітнього закладу, для реалізації якої необхідне виконання наступних задач:

– дослідити існуючі алгоритми пошуку асоціативних правил та проаналізувати їх ефективність;

– обрати оптимальний алгоритм для поставленої задачі;

– розробити підсистему існуючої інформаційної системи автоматизації управління навчальним процесом.

Об'єкт досліджень – підтримка освітньої діяльності в навчальних закладах.

Предмет досліджень – методи та засоби підтримки освітньої діяльності в навчальних закладах.

Як *наукова новизна отриманих результатів* в роботі пропонується застосування елементів інтелектуалізації для полегшення обробки великих об’ємів даних про результати навчальної діяльності, а саме – алгоритму пошуку асоціативних правил на основі результатів оцінювання учнів.

Практичне значення отриманих результатів. Під час виконання атестаційної роботи були проаналізовані існуючі алгоритми пошуку асоціативних правил, н основні чого обраний найбільш підходящій для вирішення поставленої задачі та розроблені підсистема, яка на основі даних про оцінки, що вводяться в інформаційну систему управління навчальним процесом протягом здійснення освітньої діяльності, аналізує оцінки з дисциплін та знаходить асоціативні закономірності.

Результати атестаційної роботи. Результатом атестаційної роботи є пояснювальна записка до атестаційної роботи з теоретичним описом предметної області, порівняльний аналіз алгоритмів пошуку асоціативних правил, а також розроблену підсистему.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Аналіз проблеми підтримки освітнього процесу в загальноосвітніх навчальних закладах

Оцінювання якості знань учнів загальноосвітніх навчальних закладах України здійснюється на основі методичних рекомендацій, затверджених Міністерством освіти України. Об'єктам оцінювання учнів є процес навчання, а також результат навчальної діяльності на певному його етапі.

Виявлення рівня успішності учнів в навчанні відбувається в процесі контролю.

Функціями контролю досягнень учнів в навчанні є: мотиваційна, коригувальна, прогностична, діагностувальна, навчально-перевірювальна, виховна, розвивальна.

Об’єктами контролю у процесі навчання є складові компетентностей учнів з предметів: знання про предмети і явища навколишнього середовища, взаємозв’язки і відношення між ними; вміння застосовувати отримані знання; досвід творчої діяльності; цінності орієнтири.

Облік результатів контролю знань ведеться учителем/учителькою у класному журналі і табелях навчальних досягнень. Структурними компонентами контролю навчальних досягнень учнів є перевірка та оцінювання результатів навчальної діяльності.

Перевірка досягнень учнів у навчанні здійснюється з допомоги методів, вибір яких зумовлюється особливостями змісту навчального предмета, його обсягом, рівнем узагальнення, віковими можливостями учнів.

В процесі оцінювання рівня знань учнів вчителю доводиться проводити аналіз великих об’ємів даних (результатів навчальної діяльності учнів), що потребує значних витрат часу та ручних обчислень, адже в більшості випадків,

звітність під час навчального процесу в школі ведеться через папіряний документооблік, що підвищує ймовірність припущення помилки в розрахунках та в цілому вплив людського фактору на точність результатів.

На об’єктивність та вірність результатів аналізу навчальної діяльності вчителем можуть вливати наступні чинники:

– кваліфікація вчителя;

– кількість учнів в класі;

– кількість видів оцінювання.

Також на аналіз результатів освітнього процесу можуть мати вплив людські фактори, що мають суб’єктивний характер, але, нажаль, зустрічаються не часто в реальному житті:

– сукупність – учень має високі оцінки, і вчителеві «незручно» виставляти нижчу, ніж більшість його оцінок, або навпаки, особливо коли йдеться про підсумкові оцінки;

– контраст – клас в середньому має не високі результати, а один з учнів відрізняється рівнем знань, і йому нерідко виставляють завищені оцінки;

– соціального стану – на оцінці позначається дитина яких батьків відповідала;

– ідеалізація – вчителі часом керуються принципом, що «на оцінку 5 знає тільки вчитель», тому ставиться непідйомна учнями планка оцінки вищого рівня;

– особисті упередження щодо конкретного учня — роздратованість поведінковими особливостями чи рисами характеру конкретного учня;

– очікування – вчитель сподівався на кращі чи гірші результати роботи учня.

Такі суб'єктивні моменти перешкоджають об'єктивному аналізу навчальної діяльності учнів та визначення рівня знань [10].

1.2 Аналіз та огляд реалізованої інформаційної системи підтримання навчального процесу

Розроблена, на даний момент, інформаційна система підтримки освітньої діяльності загальноосвітніх закладів має структуру, що зображена на рис.1.1.

Рисунок 1.1 – Склад компонентів ІС підтримки управління навчальним процесом

У такому вигляді система дозволяє вести облік оцінок, відвідуваності учнів, надає єдиний простій для розміщення та перегляду домашніх завдань, зауважень з боку вчителів стосовно успішності та поведінки учня, надає доступ в будь-який момент часу всіх зареєстрованих в системі користувачів (учасників навчального процесу) до цієї інформації, надає зворотній зв’язок вчителів з батьками та дозволяє захистити персональні данні за допомогою реалізації рівнів доступу до ресурсу. Також перевагою даної ІС є доступ до системи з будь-якого комп’ютера або мобільного пристрою, підключеного до мережі Internet, без попередньої інсталяції додатка.

Але все ще існують компоненти (підсистеми), що все ще потребують розробки тому з метою розширення функціональності й підвищення ефективності існуючої ІС підтримки освітньої діяльності загальноосвітніх закладів України, доцільним є доповнення елементом інтелектуальності –

підсистемою аналізу результатів навчальної діяльності учнів, оскільки аналогів подібної підсистеми, яка б вже використовувалась в загальноосвітніх закладах – не існує. Така підсистема виявляє закономірності в даних, що накопичуються під час освітнього процесу, що пов'язані з залежністю успішності учнів від різних особливостей та характеристик навчального процесу. Для виявлення таких закономірностей необхідно розглянути та проаналізувати декілька існуючих методів пошуку асоціативних правил, та за результатами обрати для реалізації той, що дав найкращі результати.

1.3 Визначення сфери застосування підсистеми підтримки освітньої діяльності

Інформаційна система підтримки освітньої діяльності повинна бути реалізована як один з компонентів ІС управління навчальним процесом в загальноосвітніх закладів України, за допомогою якої можливе виконання наступних дій:

– навчальними дисциплінами;

– облік присутності учнів на заняттях та перегляд відвідуваності учнями занять;

– запис та перегляд домашнього завдання;

– редагування та перегляд розкладу занять по класах;

– зворотній зв'язок «вчитель – батьки» (вчитель має можливість написати батькам учня і навпаки за допомогою внутрішньої пошти школи або вести діалог в чаті просто на веб-сайті;

– доступ до даних відповідно по ролям: вчитель, учень, батьки, адміністрація школи, що дозволяє надавати доступ тільки до інформації що стосується конкретного учня (якщо це сам учень або його батьки), вчителя (має доступ лише до класів і дисциплін, які він викладає).

Діаграма варіантів використання (Use case diagram) описує функціональне призначення інформаційної системи [7]. Діаграма визначення предметну

область і описує функціональних можливості існуючої ІС. Діаграма варіантів використання інформаційної системи представлена на рис. 1.2.

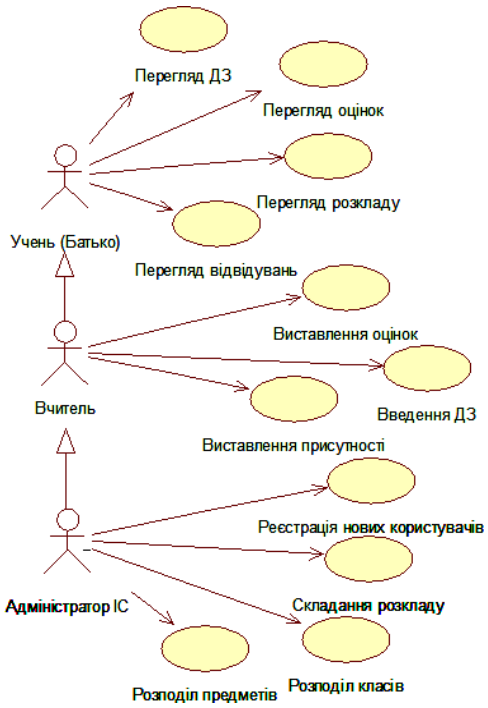


Рисунок 1.2 – Діаграма варіантів використання інформаційної системи

Інтелектуальна підсистема підтримки освітньої діяльності (ППОД) буде реалізована для користувачів як додаткова функція веб-додатку, що дозволить вчителям проводити аналіз результатів навчальної діяльності за допомогою алгоритма пошуку асоціативних правил та отримувати рекомендації щодо корегування навчального процесу на основі отриманого результату. Одночасно з цим, батьки та учні матимуть можливість:

– переглядати інформацію про рівень знань учня;

– отримувати звітність (шкільний табель) за період навчання.

1.4 Постановка завдання

Виходячи з вище проведеного аналізу предметної області, необхідним стає проаналізувати існуючі алгоритми пошуку асоціативних правил, що виявляють взаємозв’язки в великих об’ємах даних шляхом підрахунку елементів, які часто зустрічаються разом, визначити переваги та недоліки цих алгоритмів та виявити який з них найефективніше використовувати для

підсистеми підтримки освітньої діяльності в рамках інформаційної системи управління навчальним процесом: AIS, SETM, Apriori.

Розроблена підсистема підтримки освітньої діяльності буде реалізована в якості одного з компонентів інформаційної системи управління навчальним процесом в загальноосвітніх закладах України. Алгоритм роботи підсистеми буде обробляти дані про оцінки, що заносяться вчителем в ІС під час навчального процесу та виявляти взаємозв'язки між появою тих чи інших оцінок з предметів, що допоможе спростити аналіз результатів навчальної діяльності учнів.

Для реалізації запропонованої підсистеми необхідно виконати такі завдання:

– провести аналіз предметної області та оцінити існуючі підходи до вирішення поставленої задачі;

– провести аналіз реалізованої підсистеми підтримки освітньої діяльності;

– визначити сфери застосування підсистеми;

– розробити системні вимоги до підсистеми;

– розробити функціональні вимоги до підсистеми;

– описати структуру розробленої підсистеми;

– провести логічне і фізичне моделювання даних підсистеми;

– внести зміни до існуючої бази даних для Microsoft SQL-server;

– розробити SQL-запити для серверних елементів управління;

– провести тестування розробленого програмного забезпечення.

1.5 Доповнення діаграми варіантів використання

З урахуванням проведеного аналізу в підрозділах 1.2 – 1.3 та сформульованої постановки задачі (підрозділ 1.4) необхідно розробити діаграму варіантів використання підсистеми підтримки освітньої діяльності (рис. 1.3). Користувачами підсистеми є вчитель.

Доступ до підсистеми підтримки освітньої діяльності має вчитель,

оскільки він займається аналізом успішності учнів в процесі навчання та приймає рішення щодо корегування навчального процесу в залежності від отриманих результатів. Опис варіантів використання наведений у табл. 1.1.

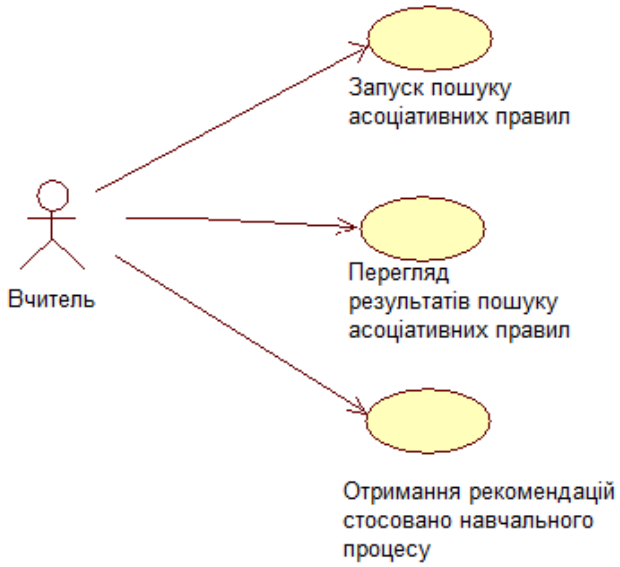


Рисунок 1.3 – Діаграма варіантів використання розробленої підсистеми

Вчитель має доступ до таких дій в підсистемі:

– запуск модуля пошуку асоціативних правил;

– перегляд результатів пошуку асоціативних правил;

– інтерпретація знайдених закономірностей.

Таблиця 1.1 – Опис варіантів використання

Варіант використання	Опис	Умови виконання	Результат виконання
1	2	3	4
Запуск модуля пошуку асоціативних правил	Запуск модулю ІС «підсистема підт-римки освітньої діяльності», що виконує алгоритм пошуку асоціа-тивних правил	Система та відповідні компоненти доступні, користувач має статус «вчитель» в системі, база даних з оцінками містить дані	Виконання роботи алгоритму пошуку асоціативних правил

Продовження таблиці 1.1

1	2	3	4
Перегляд результатів пошуку асоціативних правил	Відкрити та переглянути результати роботи алгоритму	Система та відповідні компоненти доступні, користувач має статус «вчитель» в системі, база даних з оцінками містить дані та робота алгоритма завершена без помилок	Перелік асоціативних правил, виведених на основі даних про оцінки
Отримання рекомендацій щодо навчального процесу	Натиснути відповідну кнопку «Вивід рекомендацій»	Система та відповідні компоненти доступні, користувач має статус «вчитель» в системі, база даних з оцінками містить дані та робота алгоритма завершена без помилок	Перелік рекомендацій щодо змін в навчальному процесі на основі результатів роботи алгоритму

1.6 Методи пошуку логічних залежностей

Одна з найбільш розповсюджених задач аналізу даних є визначення наборів множин, що часто зустрічаються в великій множині наборів. Задача, що розглядається, вперше виникла в маркетинговій сфері діяльності, коли постала необхідність виявляти товари, що зазвичай найчастіше всього купуються разом, вона мала назву – «продуктовий кошик» [1]. Пізніше цю задачу сформували в загальному вигляді як задачу пошуку асоціативних правил, які почали використовуватись як вхідні дані для створення баз знань після їх оцінки, аналізу даних та прийняття рішень. Задача виводу асоціативних правил та класифікаційних правил мають принципову відмінність одна від одної – класифікаційні правила інтерпретуються як прогноз та поділяються на тестові і навчальні, а асоціативні правила, відповідно – ні.

Визначення поняття асоціативного правила у загальному вигляді формулюється наступним чином: дискретні дані, що зберігаються в БД та містять інформацію про транзакції, здійснені покупцями, кожна з яких – це комбінація товарів, які купив один покупець за один похід в магазин (також ще має назву «ринковий кошик»), тож необхідно провести пошук закономірностей покупки товарів разом. Цю задачу можна виразити формулою: «якщо купується множина товарів *X* – також купується й множина товарів *Y*».

Основні положення знаходження асоціацій: нехай $I = \{i_1, i_1, \ldots, i_n\}$ – множина елементів (в нашому випадку – оцінок), а T – множина комбінацій оцінок, де кожна множина $t \in T$ – це набір елементів з I, $t \in I$. Кожна комбінація оцінок являє собою бінарний вектор, де $t[k] = 1$, якщо елемент i_k є присутнім у множині оцінок, отриманих учнем, а якщо ні, то $t[k] = 0$. Вважають, що комбінація оцінок t містить X (деякий набір елементів з I), якщо $X \subseteq t$. Асоціативним правилом називається імплікація $X \to Y$, де $X \subset I$, $Y \subset I$ і $X \cap Y = \varnothing$.

Множина підтримки T_X набору елементів X складається з усіх комбінацій оцінок, які містять цей набір, тобто $T_X = \{t \in T \mid X \subseteq t\}$. Підтримка (Support) набору елементів X обчислюється як

$$Supp(X) = \frac{|T_X|}{T}, \qquad (1.1)$$

де T_X – це число комбінацій оцінок (скільки раз зустрічалась ця комбінація), що містять X, а T – загальне число комбінацій.

Тобто підтримка показує скільки раз одна й та сама комбінація оцінок з предметів для учня зустрічається серед числа інших комбінацій.

Рівень підтримки (Support) або вірогідність правила $X \to Y$ визначається відносним числом комбінації оцінок, для якої виконується правило:

$$Supp(X \to Y) = \frac{T_{X \to Y}}{T} \cdot 100\%, \qquad (1.2)$$

де $T_{X \to Y}$ – число комбінації оцінок, що містять імплікацію $X \vee Y$.

У результаті такого аналізу можна встановити закономірність наступного вигляду: «Якщо в наборі оцінок для одного учня, зустрілась комбінація оцінок *X*, то можна зробити висновок, що цій же транзакції повинен з'явитись комбінація оцінок *Y*». Встановлення таких закономірностей дає нам можливість знаходити прості й зрозуміли правила, що мають назву асоціативні, основними характеристиками яких є підтримка та достовірність правила.

Достовірність (Confidence) визначає ймовірність, з якої з *X* випливає *Y*:

$$Conf(X \rightarrow Y) = \frac{Supp(X \rightarrow Y)}{Supp(X)} \cdot 100\%. \qquad (1.3)$$

Іншими словами, правило «Із *X* випливає *Y*» справедливо з достовірністю рівною $Conf(X \rightarrow Y)$, якщо з відсотка наборів оцінок, що дорівнює значенню достовірності, з усієї множини, що містить комбінацію оцінок *X*, також містять комбінацію оцінок *Y*.

За допомогою алгоритмів пошук асоціативних правил можна отримати всі можливі правила, що мають вигляд «Із *X* випливає *Y*», що будуть мати різні значення підтримки та достовірності. Проте в більшості, кількість правил потрібно обмежувати завчасно встановленими мінімальними та максимальними значеннями підтримки та достовірності. Якщо значення підтримки занадто велике, то знайдене правило занадто очевидне й, швидше за все, добре відоме, а з іншого боку, можливе знаходження великої кількості правил з низьким значенням підтримки, що будуть ніяк не обґрунтовані та невідомими для людини, що їх аналізує. Тому з'являється необхідність визначити такий інтервал, який і забезпечить знаходження неочевидних правил, і їх обґрунтованість.

У результаті роботи алгоритма, що описаний віще, формуються правила, на основі тих наборів елементів, які найчастіше трапляються у вхідних даних.

Оформлення асоціативних правил полягає в розбивці покриття на підмножини: якщо S – покриття, то для непустих $S_1 \subset S$ і $S_2 = S - S_1$ і

асоціативним правилом є $S_1 \to S_2$, якщо воно має потрібний рівень довіри (Confidence).

Для знаходження покриття існує метод обмеженого перебору для пошуку логічних закономірностей в даних, що був запропонований М.М. Бонгардом [15,16] в 1967 р. Цей метод визначає частоти комбінацій простих логічних подій в підгрупах даних, тобто дозволяє прибирати з аналізу події, маючих низьку логічну частоту. Обмеженням виступає довжина комбінації логічних подій, а на основі аналізу вирахуваних частот виноситься висновок про корисність комбінації для встановлення асоціацій в даних для класифікації чи прогнозування. Найбільш відомими є алгоритми [15-23] AIS, SETM, Apriori.

1.6.1 Алгоритм SETM

Поява алгоритму SETM (Set Oriented Mining) пов’язане з потребою застосування мови SQL для виявлення наборів товарів, що часто зустрічаються. Алгоритм SETM формує кандидатів (етап, під час якого, алгоритм створює *i*-елементного кандидата, де *i* – це номер етапа, та не розраховує під час нього підтримку кандидата) під час сканування бази даних. Для використання стандартної операції об’єднання мовою SQL щоб сформувати кандидата, алгоритм SETM відділяє формування кандидата від підрахування. В свою чергу підрахування кандидатів – це етап, на якому вираховується підтримка кожного *i*-елементного кандидата, де також відбувається і відсіювання кандидатів, підтримка яких менше, ніж визначений мінімум, а набори, що залишаються після цього етапу, називаються тими, що часто зустрічаються.

Алгоритм SETM можна представити у вигляді псевдокоду [3]:

```
L1 = {large 1-itemsets}      // 1-елементний набір з TID
for (k = 2; Lk-1≠∅; k++) do
begin
   C*k=∅
   forall transactions t∈D do          //перегляд транзакцій
   begin
      Lt = {l∈L*k-1|l.TID=t.TID}       //(k-1)-набори, що містять t
           forall large itemsets lt∈Lt do
```

```
            begin
                  Ct = 1-розширення l_t , що містять t;
                  C*_k += {t.TID,c | c∈Ct}          // Кандидати в t
            end tend
   sort C*_k on itemsets
   delete all itemsets c∈C*_k for which c.count < minsupp giving L*_k
   Lk = {c∈Ck | c.count >= minsupp}
   Lk = {<l.itemset.count of l in L*_k> | l∈L*_k}
   sort L*_k on TID;                                //Сортування за TID
end k
```

Повернення результату – *Return* $\bigcup_k L_k$.

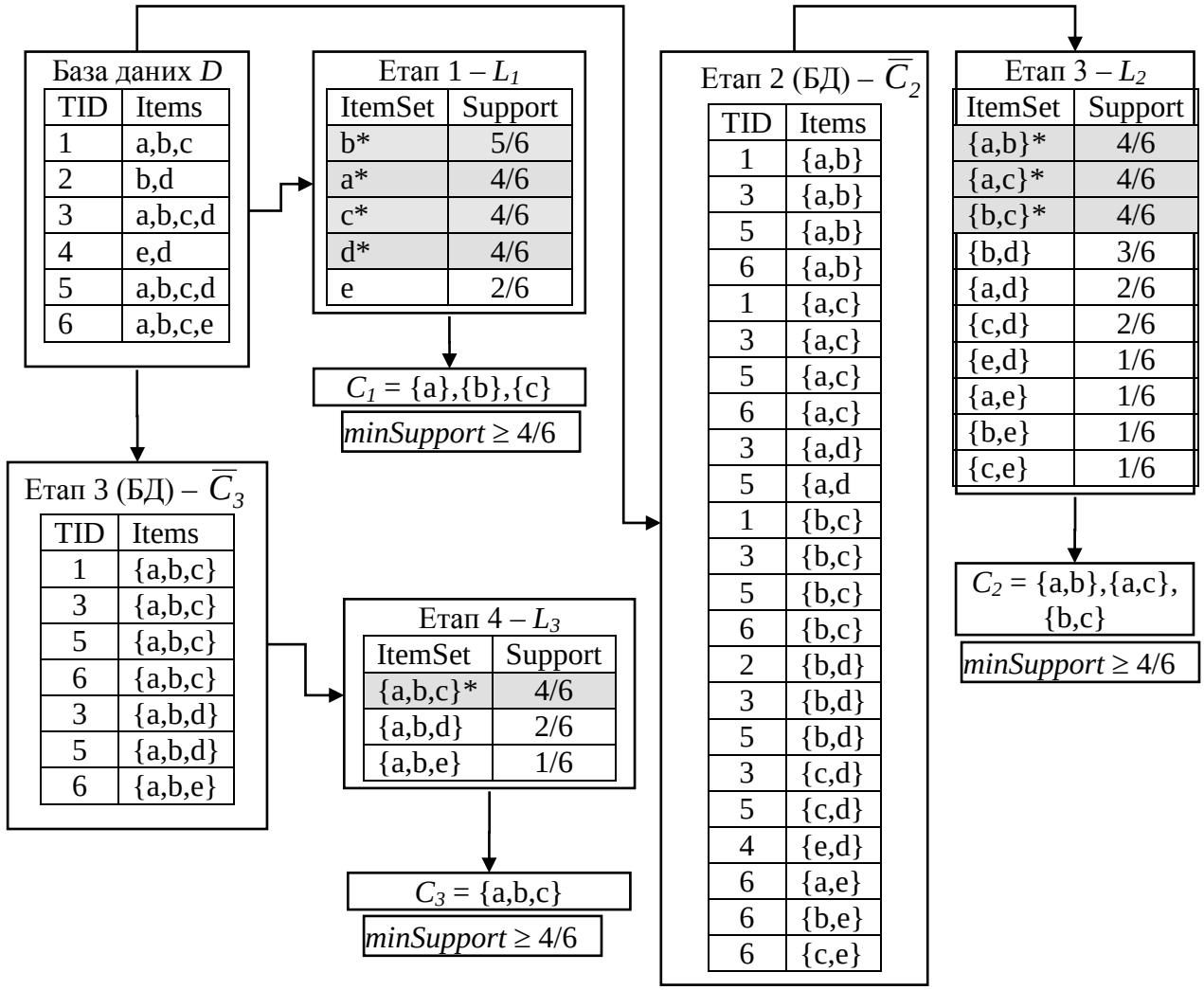


Рисунок 1.4 – Послідовність операцій за етапами алгоритму SETM

Нижче наведений опис роботи алгоритму SETM.

Перший етап. Для роботи алгоритму мають задаватися: L_k – множина *k*-частих наборів і $\overline{C}_k$ – множина кандидатів *k*-елементних наборів, сформованих з L_{k-1}, C_k – множина кандидатів *k*-елементних наборів, в яких підтримка (*Supp*) не

менше заданого мінімального значення (*minSupp)*. Для *k=1* просканувати БД , сформувати усі можливі 1-елементні набори – L_1 і розрахувати *Supp*. Створити множину одно елементних наборів кандидатів $c \in C_1$, відібравши з *L1* усі набори, що задовільняють умові *Supp* > *minSupp*.

Другий етап. Перехід до другої ітерації: *k=k+1*.

Третій етап. Якщо не виходить створити k-елементні набори, то завершити операції та вихід з циклу, інакше перейти до наступного кроку.

Четвертий етап. Сканування БД, формування всіх можливих k-елементарних наборів у вигляді кортежів <*t.TID, lt.itemset*> .

П’ятий етап. Формування всіх можливих *k*-елементарних наборів – L_k і розрахувати рівень підтримки.

Шостий етап. Створити множину *k*-елементних наборів кандидатів $c \in C_1$, при цьому треба відібрати з L_k всі набори в яких підтримка більше ніж мінімальна (minSupp) та повернутися до другого етапу. В результаті роботи алгоритму отримаємо об’єднання всіх множин L_k для всіх *k*.

Результаті роботи, що були отримані в результаті роботи алгоритму SETM потребують додаткової обробки покриття, яка включає в себе нагалудження всіх унікальних кортежів, розрахунки *Supp*, а також зменшення покриття *ItemSet* шляхом видалення записів C_k в яких *Supp* менше заданого мінімуму.

1.6.2 Приклад пошуку асоціативних правил за допомогою алгоритму SETM

Нижче розглядається приклад застосування алгоритму SETM для пошуку асоціативних правил у визначеній предметній області. Але перед розрахунками необхідно визначитись в якому вигляді будуть зберігатись оцінки у базі даних транзакцій.

Сформуємо таку класифікацію шкільних дисциплін за напрямами:

1) математичні дисципліни:

- математика: 1-6 класи;
- алгебра: 7-11 класи;
- геометрія: 7-11 класи;
- економіка: 10-11 класи;
- інформатика: 8-11 класи;

2) природньо-наукові дисципліни:
- оточуючий світ: 1- 4 класи;
- географія: 5-11 класи;
- біологія: 5-11 класи;
- астрономія: 11 клас;
- фізика: 7-11 класи;
- хімія: 8-11 класи;
- ОБЖ: 7-10 класи;
- екологія: 10 клас;

3) гуманітарні дисципліни:
- історія: 5-11 клас;
- право: 8-9 клас;
- суспільствознавство: 6-11 класи;
- етика: 5-6 класи;
- філософія: 11 клас;
- психологія: 5-7 класи;

4) філологічні дисципліни:
- українська мова: 2-11 класи;
- російська мова: 2-8 класи;
- читання: 1- класи;
- література: 5-11 класи;
- зарубіжна література: 6-11 класи;
- іноземна мова 1-11 класи;

5) трудове навчання:

- труд: 1-4 клас;
- технологія: 5-11 класи;
- креслення: 9-10 класи;

6) фізична культура: 1-11 класи;

7) зображувальне мистецтво:

- малювання: 1-7 класи;
- світова художня культура: 8 клас.

Кожен напрям дисциплін в базі даних буде мати свій ідентифікатор, що дозволить розрізняти їх та проводити аналіз тільки серед потрібної категорії предметів:

- математичні дисципліни (A);
- природньо-наукові дисципліни (B);
- гуманітарні дисципліни (C);
- філологічні дисципліни (D);
- трудове навчання (E);
- фізична культура (G);
- зображувальне мистецтво (H).

Розглянеться клас математичних дисциплін та підсумкові оцінки за семестр по ним, на яких застосуємо алгоритм пошуку асоціативних правил. Оцінки задані у вигляді одномірного масиву, де кожному індексу елемента відповідає предмет, оцінка за який знаходиться під цим індексом, в даному випадку використаєм наступний порядок: так як предметів, що входять до категорії математичних дисциплін входить 5 шкільних предметів, то і елементів матиці буде 5:

- математика: 1-6 класи (індекс – 0);
- алгебра: 7-11 класи (індекс – 1);
- геометрія: 7-11 класи (індекс – 2);
- економіка: 10-11 класи (індекс – 3);
- інформатика: 8-11 класи (індекс – 4).

Також для ідентифікації оцінок за дисциплінами, визначаємо позначення в вигляді якого оцінка буде відноситись до предмету: математика – m, алгебра – a, геометрія – g, економіка – e, інформатика – i.

Припустимими будуть випадки, якщо учень знаходиться в 1-6 класі і в його навчальній програмі ще немає таких предметів як економіка, алгебра та геометрія, або навпаки, учень знаходиться в 10 класі і в нього є в програмі все предмети, окрім математики, тому відсутні елементи будуть мати значення NULL, що означає те, що даної дисципліни немає в переліку навчальної програми. Алгоритм буде відпрацьовувати по кожному з видів дисциплін по всім учням, що є в базі даних з оцінками та шукати правила на основі існуючих даних про успішність учнів.

Таблиця 1.2 – Приклад зберігання оцінок в базі даних транзакцій (TID)

ID	ПІБ учня	Вид дисциплін	Оцінки
1	Учень 1	A	Null, a10, g10, e11, i10
2	Учень 2	A	Null, a9, g8, e10, i10
3	Учень 3	A	Null, a4, g6, e7, i8
4	Учень 4	A	Null, a10, g10, e10, i10
5	Учень 5	A	Null, a11, g9, e12, i10
6	Учень 6	A	Null, a6, g7, e8, i8
7	Учень 7	A	Null, a9, g9, e10, i11
8	Учень 8	A	m10, Null, Null, Null, i11
9	Учень 9	A	m6, Null, Null, Null, i8
10	Учень 10	A	m9, Null, Null, Null, i9
11	Учень 11	A	Null, a10, g12, e11, i10
12	Учень 12	A	Null, a9, g8, e10, i10

Продовження таблиці 1.2

ID	ПІБ учня	Вид дисциплін	Оцінки
13	Учень 13	A	Null, a4, g5, e6, i6
14	Учень 14	A	Null, a10, g9, e9, i10
15	Учень 15	A	Null, a11, g9, e12, i10
16	Учень 16	A	Null, a9, g7, e8, i8
17	Учень 17	A	Null, a9, g8, e10, i11
18	Учень 18	A	m10, Null, Null, Null, i11
19	Учень 19	A	m7, Null, Null, Null, i8
20	Учень 20	A	m8, Null, Null, Null, i9

Перед виконанням алгоритму потрібно задати: L_k – множина *k*-елементних частих наборів і $\overline{C}_k$ – множина кандидатів k-елементних наборів, сформованих з L_{k-1}, C_k – множина кандидатів k-елементних наборів чия підтримка *Supp* не менше заданої *minSupp*.

Крок 1. Для k=1 просканувати базу даних транзакцій, сформувати всі можливі 1-елементні набори – L_1 і розрахувати підтримку. Створити множину 1-елементних наборів кандидатів $c \in C_1$, відібравши з L_1 усі набори в яких підтримка більше *minSupp=4/20*.

Таблиця 1.3 – Результат розрахунку підтримки для 1-елементних кандидатів

ItemSet	Support
a4	2/20
a6	1/20
a9*	5/20
a10*	4/20
a11	2/20
g5	1/20
g6	1/20
g7	2/20
g8	3/20
g9*	4/20
g10	2/20
g12	1/20
e6	1/20

Продовження таблиці 1.3

ItemSet	Support
e7	1/20
e8	2/20
e9	1/20
e10*	5/20
e11	2/20
e12	1/20
i6	1/20
i8*	5/20
i9	2/20
i10*	8/20
i11*	4/20
m6	1/20
m7	1/20
m8	1/20
m9	1/20
m10	2/20

У результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 1-елементних комбінацій, отримали наступних кандидатів, чия пітримка не менша ніж задана в умові *minSupp=4*/20, тобто множина кандидатів C_1={a9},{a10}, {g9}, {e10}, {i8}, {i10}, {i11}.

Таблиця 1.4 – Отримані 1-елементні кандидати, що відповідають заданій умові

ItemSet	Support
a9*	5/20
a10*	4/20
g9*	4/20
e10*	5/20
i8*	5/20
i10*	8/20
i11*	4/20

Крок 2. На $k_2=k_1+1$ етапі знайдемо всі можливі 2-елементні пари та підрахуємо їх підтримку. Тому для початку необхідно сформувати базу даних що містить всі можливі 2-елементні комбінації.

Таблиця 1.5 – База даних сгенерованих 2-елементних комбінацій

TID	Items	TID	Items
1	{a4, g6}	34	{a11, e12}
2	{a4,g6}	35	{m7, i8}
3	{a4, e10}	36	{m8, i9}
4	{a4, i8}	37	{m9, i9}
5	{a4, g5}	38	{m10, i11}
6	{a4, g5}	39	{m10, i11}
7	{a4, g5}	40	{g10, e11}
8	{a4, g5}	41	{g10, e10}
9	{a4, e6}	42	{g9, e12}
7	{a4, i6}	43	{g9, e12}
8	{a6, g7}	44	{g9, e10}
9	{a6, e8}	45	{g8, e10}
10	{a6, i8}	46	{g8, e10}
11	{a9, g8}	47	{g8, e10}
12	{a9, g8}	48	{g7, e8}
13	{a9, g8}	49	{g7, e8}
14	{a9, g9}	50	{g7, i8}
15	{a9, g7}	51	{g7, i8}
17	{a9, e10}	52	{g6, e7}
18	{a9, e10}	53	{g6, i8}
19	{a9, e10}	54	{e11, i10}
20	{a9, e10}	55	{e11, i10}
15	{a9, e8}	56	{g5, e6}
16	{a9, i10}	57	{g5, i6}
17	{a9, i10}	58	{e10, i10}
18	{a9, i11}	59	{e10, i10}
19	{a9, i11}	60	{e10, i10}
20	{a9, i8}	61	{e7, i8}
21	{a10, g10}	62	{e12, i10}
22	{a10, g10}	63	{e12, i10}
23	{a10, g9}	64	{e8, i8}
24	{a10, g12}	65	{e8, i8}
25	{a10, e11}	66	{e6, i6}
26	{a10, e11}	67	{e10, i11}
27	{a10, e10}	68	{e10, i11}
28	{a10, e9}	69	{a11, i10}
29	{a10, i10}	70	{a11, i10}
30	{a10, i10}	71	{m10, i11}
31	{a10, i10}	72	{m10, i11}
32	{a10, i10}	73	{m6, i8}

Наступним етапом буде розрахування рівня підтримки для згенерованих записів бази даних 2-елементних кандидатів.

Таблиця 1.6 – Розрахований рівень підтримки для 2-елементних кандидатів

ItemSet	Support	ItemSet	Support
a4, g6	2/20	m7, i8	1/20
a4, e10	1/20	m8, i9	1/20
a4, i8	1/20	m9, i9	1/20
a4, g5	4/20	m10, i11	1/20
a4, e6	1/20	g10, e11	1/20
a4, i6	1/20	g10, e10	1/20
a6, g7	1/20	g9, e12	2/20
a6, e8	1/20	g9, e10	1/20
a6, i8	1/20	g8, e10	3/20
a9, g8	3/20	g7, e8	2/20
a9, g9	1/20	g7, i8	2/20
a9, g7	1/20	g6, e7	1/20
a9, e10*	4/20	g6, i8	1/20
a9, e8	1/20	g5/e6	1/20
a9, i10	2/20	g5/i6	1/20
a9, i11	2/20	e11, i10	2/20
a9, i8	1/20	e10, i10	3/20
a10, g10	2/20	e7, i8	1/20
a10, g9	1/20	e12, i10	2/20
a10, g12	1/20	e8, i8	2/20
a10, e11	2/20	e6, i6	1/20
a10, e10	1/20	e10, i11	2/20
a10, e9	1/20	a11, i10	2/20
a10, i10	4/20	m10, i11	2/20
a11, g9	2/20	m6, i8	1/20
a11, e12	2/20		

В результаті сканування БД транзакцій та підрахувавши підтримку виявлених 2-елементних комбінацій, знайдено наступних кандидатів, чия підтримка не менша *minSupp=4*/20, тобто C_2={a10, i10}, {a9, e10}, {a4,g5}.

Таблиця 1.6 – Отримані кандидати, що відповідають заданій умові

ItemSet	Support
a10, i10*	4/20
a9, e10*	4/20

a4, g5 *	4/20

Крок 3. На $k_3=k_1+1$ етапі знайдемо всі можливі 3-елементні пари та підрахуємо їх підтримку. Для цього проскануємо базу даних транзакцій та знайдемо всіх 3-елементних кандидатів.

Таблиця 1.7 – База даних 3-елементних кандидатів

TID	Item	TID	Item
1	a10, g10, e11	19	a9, g7, e8
2	a10, g10, e11	20	a10, g10, i10
3	a9, g8, e10	21	a9, g8, i10
4	a9, g8, e10	22	a10, g10, i10
5	a9, g8, e10	23	a4, g6, i8
6	a4, g6, e7	24	a9, g8, i10
7	a10, g10, e10	25	a11, g9, i12
8	a11, g9, e12	26	a6, g7, i8
9	a11, g9, e12	27	a9, g9, i11
10	a6, g7, e8	28	a10, g12, i10
11	a9, g9, e10	29	a4, g5, e6
12	a9, g9, e10	30	a10, g9, i9
13	a10, g12, e11	31	a9, g7, i8
14	a4, g5, e6	32	a9, g7, e8
15	a10, g9, e9	33	a10, g10, i10
16	a11, g9, e12	34	a10, g10, i10
17	a9, g7, e8		
18	a10, g10, i10		

Далі розрахуемо рівень підтримки для отриманих 3-елементних кандидатів та знайдемо тих, чия підтримка не менша, ніж задана умова *minSupp*=4/20.

Таблиця 1.8 – Розрахована підтримка для 3-елементних кандидатів

ItemSet	Support	ItemSet	Support
a10, g10, e11	2/20	a9, g8, i10	2/20
a9, g8, e10	3/10	a4, g6, i8	1/20
a4, g6, e7	1/10	a11, g9, i12	1/20
a10, g10, e10	1/10	a6, g7, i8	1/20
a11, g9, e12	2/10	a9, g9, i11	1/20
a6, g7, e8	1/10	a10, g12, i10	1/20
a9, g9, e10	2/10	a4, g5, e6	1/20
a10, g12, e11	1/10	a10, g9, i9	1/20
a4, g5, e6	1/10	a9, g7, i8	1/20

a10, g9, e9	1/10	a9, g7, e8	1/10

Продовження таблиці 1.8

ItemSet	Support	ItemSet	Support
a11, g9, e12	1/10	a10, g10, i10	2/10
a9, g7, e8	1/10		
a10, g10, i10	2/10		

У результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 3-елементних комбінацій, не було знайдно кандидатів, що відповідають заданій умові *minSupp*=20/5, тобто C_3=0.

Результатом роботи алгоритму є об'єднання всіх множин L_k для всіх k, що наведені на рисунку нижче.

Таблиця 1.9 – Об'єднання всіх множин L_k для всіх k

ItemSet	Support
a9*	5/20
a10*	4/20
g9*	4/20
e10*	5/20
i8*	5/20
i10*	8/20
i11*	4/20
a10, i10*	4/20
a9, e10*	4/20
a4, g5 *	4/20

На основі поданих кандидатів можна формалізувати закономірності у вигляді лінгвістичних правил:

– «25% учнів мають оцінки достатнього рівня з алгебри, інформатики»;

– «20% учнів мають оцінки достатнього рівня з геометрії»;

– «60% учнів мають оцінки високого рівня з інформатики»;

– «20% учнів мають оцінки високого рівня з алгебри»;

– «25% учнів мають оцінки високого рівня економіки»;

– «маючи оцінку високого або достатнього рівня з алгебри, зазвичай, учні мають оцінки високого рівня і з інформатики»;

– «маючи оцінки середнього рівня з алгебри, учні мають також оцінки середнього рівня і з геометрії».

1.6.3 Алгоритм AIS

Ще один алгоритм пошуку асоціативних правил AIS, що був розроблений в 1993 році співробітниками центру IBM Almaden, в якому кандидати множини наборів генеруються та підраховуються, так звано, «на льоту», під час сканування БД. Кожна транзакція перевіряється чи є наявність великих наборів, що виявляються при попередньому переході. Тож, нові набори формуються шляхом розширення наборів, що вже існують.

Алгоритм AIS можна представити у вигляді псевдокоду [16]:

```
L1 = {large 1-itemsets}
for (k = 2; Lk-1≠∅;; k++ ) do
begin
      Ck=∅
      forall transactions t∈D do
      begin
            Lt = subset(Lk-1, t); // Large itemsets contained in t
            forall large itemsets lt∈Lt do
            begin
                  Ct = 1-extensions of lt contained in t;
                  // Candidates contained in t
                  forall candidates c∈Ct do
                  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;
            end t
      Lk = {c∈Ck | c:count ≥ minsupp
end k
```

Повернення результату – *Return* $\bigcup_k L_k$.

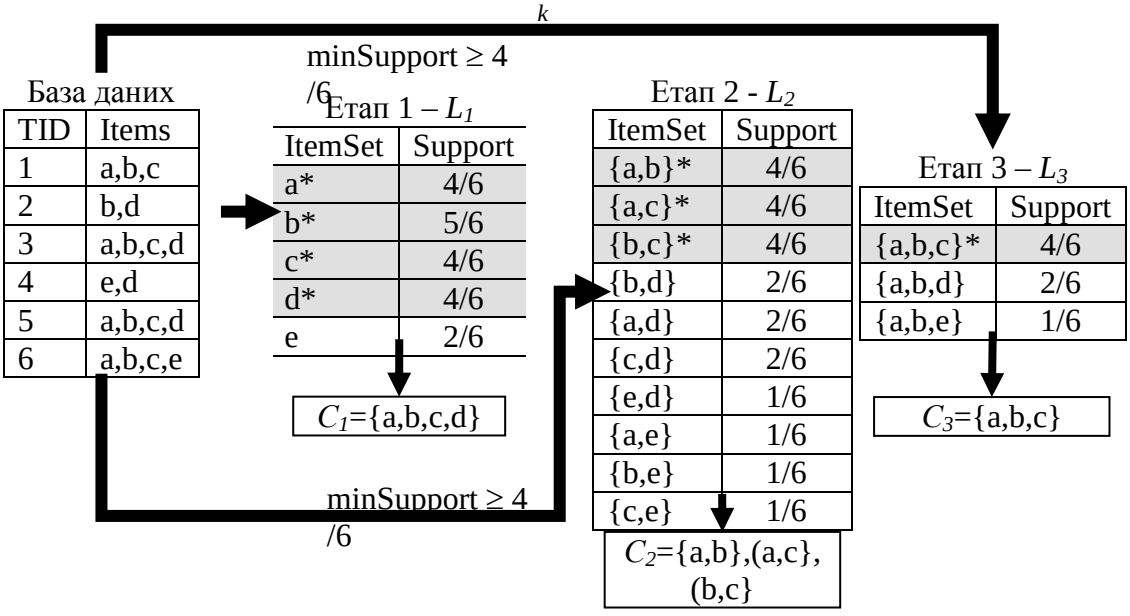


Рисунок 1.5 – Послідовність операцій за етапами алгоритму AIS

Опис кроків алгоритму AIS.

Перший етап. Для k=1 виконується сканування БД, сформувати всі можливі одноелементні набори – L_1 і розрахувати рівень підтримки *Support*. Створити множину одноелементних наборів кандидатів $c \in C_k$, виділити з L_1 всі набори, у яких підтримка більше *minSupp*.

Другий етап. Створення наборів з кількістю елементів $k=k+1$.

Третій етап. Якщо не вдається створити набори з k елементів, то завершити виконання алгоритму, інакше виконати наступний крок.

Четвертий етап. Сканування БД, формування всіх можливих k-елементних наборів – L_k та розрахувати рівень підтримки для них. Створення множини k-елементних наборів кандидатів $c \in C_k$, відібравши з L_k усі набори в яких підтримка більша ніж встановлена *minSupp*. Повернутися до етапу 2.

Проаналізуємо роботу алгоритму на прикладі, що використовувався у розрахунках для алгоритму SETM – оцінках , що описані в табл. 2.2.

Перший крок. Для k=1 проскануємо тестову БД, що була описана в таблиці 2.2 з оцінками по предметам математичного напряму, сформуємо всі можливі 1-елементні набори – L_1 і розрахувати рівень підтримки *Support*. Створимо множину 1-елементних наборів кандидатів $c \in C_k$, виділити з L_1 всі набори, у яких підтримка більше або дорівнює заданим обмеженням *minSupp=4/20*.

Таблиця 1.9 – Результат сканування БД транзакцій та розрахунку підтримки

ItemSet	Support	ItemSet	Support
a4	2/20	e10*	5/20
a6	1/20	e11	2/20
a9*	5/20	e12	1/20
a10*	4/20	i6	1/20
a11	2/20	i8*	5/20
g5	1/20	i9	2/20
g6	1/20	i10*	8/20
g7	2/20	i11*	4/20
g8	3/20	m6	1/20
g9*	4/20	m7	1/20

Продовження таблиці 1.9

ItemSet	Support	ItemSet	Support
g10	2/20	m8	1/20
g12	1/20	m9	1/20
e6	1/20	m10	2/20
e7	1/20		
e8	2/20		
e9	1/20		

У результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 1-елементних комбінацій, отримали наступних кандидатів, що мають підтримку, яка більша або дорівнює заданій *minSupp*=3/20 та маємо C_1={a9, a10, g9, e10, i8, i10, i11}. Алгоритм AIS відрізняється від SETM тим, що кандидати множини наборів генеруються та підраховуються «на льоту», під час сканування БД. Кожна транзакція перевіряється на наявність великих наборів, що виявляються при попередньому переході. Тож, нові набори формуються шляхом розширення наборів, що вже існують, що допомагає скоротити витрати на обчислення та зберігання проміжкових результатів.

Таблиця 1.10 – 2-елементні кандидати, підтримка яких відповідає заданій умові

ItemSet	Support
a9*	5/20
a10*	4/20
g9*	4/20
e10*	5/20
i8*	5/20
i10*	8/20
i11*	4/20

Крок 2. На $k_2=k_1+1$ етапі знайдемо всі можливі 2-елементні пари та підрахуємо їх підтримку (рисунок 1.7). У результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 2-елементних комбінацій, знайдено наступних кандидатів, підтримка яких є більша або дорівнює заданій умові *minSupp*=4/20: C_2={a10,i10}, {a9, e10}, {a4, g5} (таблиця 1.12).

Таблиця 1.11 – Розрахунок підтримки для 2-елементних кандидатів

ItemSet	Support	ItemSet	Support
a4, g6	2/20	m7, i8	1/20
a4, e10	1/20	m8, i9	1/20
a4, i8	1/20	m9, i9	1/20
a4, g5	4/20	m10, i11	1/20
a4, e6	1/20	g10, e11	1/20
a4, i6	1/20	g10, e10	1/20
a6, g7	1/20	g9, e12	2/20
a6, e8	1/20	g9, e10	1/20
a6, i8	1/20	g8, e10	3/20
a9, g8	3/20	g7, e8	2/20
a9, g9	1/20	g7, i8	2/20
a9, g7	1/20	g6, e7	1/20
a9, e10*	4/20	g6, i8	1/20
a9, e8	1/20	g5/e6	1/20
a9, i10	2/20	g5/i6	1/20
a9, i11	2/20	e11, i10	2/20
a9, i8	1/20	e10, i10	3/20
a10, g10	2/20	e7, i8	1/20
a10, g9	1/20	e12, i10	2/20
a10, g12	1/20	e8, i8	2/20
a10, e11	2/20	e6, i6	1/20
a10, e10	1/20	e8, i8	1/20
a10, e9	1/20	e10, i11	2/20
a10, i10	4/20	a11, i10	2/20
a11, g9	2/20	m10, i11	2/20
a11, e12	2/20	m6, i8	1/20

Таблиця 1.12 – Отримані кандидати, що відповідають заданій умові

ItemSet	Support
a10, i10*	4/20
a9, e10*	4/20
a4, g5 *	4/20

Крок 3. На $k_3=k_1+1$ етапі знайдемо всі можливі 3-елементні пари та підрахуємо їх підтримку (таблиця 1.13).

Таблиця 1.13 – Розрахунок підтримки для 3-елементних кандидатів

ItemSet	Support	ItemSet	Support
a10, g10, e11	2/20	a9, g8, i10	2/20
a9, g8, e10	3/10	a4, g6, i8	1/20
a4, g6, e7	1/10	a11, g9, i12	1/20
a10, g10, e10	1/10	a6, g7, i8	1/20
a11, g9, e12	2/10	a9, g9, i11	1/20
a6, g7, e8	1/10	a10, g12, i10	1/20
a9, g9, e10	2/10	a4, g5, e6	1/20
a10, g12, e11	1/10	a10, g9, i9	1/20
a4, g5, e6	1/10	a9, g7, i8	1/20
a10, g9, e9	1/10	a9, g7, e8	1/10
a11, g9, e12	1/10	a10, g10, i10	2/10
a9, g7, e8	1/10		
a10, g10, i10	2/10		

В результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 3-елементних комбінацій, не було знайдено кандидатів, підтримка яких більша або дорівнює заданій умові *minSupp*=4/20.

Результатом роботи алгоритму є об’єднання всіх множин L_k для всіх k, що наведені нижче (таблиця 1.14).

Таблиця 1.14 – Об’єднання всіх множин L_k для всіх k

ItemSet	Support
a9*	5/20
a10*	4/20
g9*	4/20
e10*	5/20
i8*	5/20
i10*	8/20
i11*	4/20
a10, i10*	4/20
a9, e10*	4/20
a4, g5 *	4/20

Об’єднання всіх множин L_k для всіх k є результатом роботи алгоритму AIS.

Головним недоліком як алгоритмів AIS так і SETM є надлишок генерації кандидатів в покриття. Для усунення недоліків вище розглянутих алгоритмів, був розроблений алгоритм Apriori.

1.6.4 Алгоритм Apriori

Алгоритм визначає набори, що часто зустрічаються, за декілька етапів, де на кожному і-му визначаються усі і-елементні набори, що часто зустрічаються, в свою чергу кожен етап складається з двох кроків:

– формування кандидатів (generation);

– підрахунок підтримки кандидатів (counting)

Схема алгоритму Apriori для k=3 представлена на рис. 1.6.

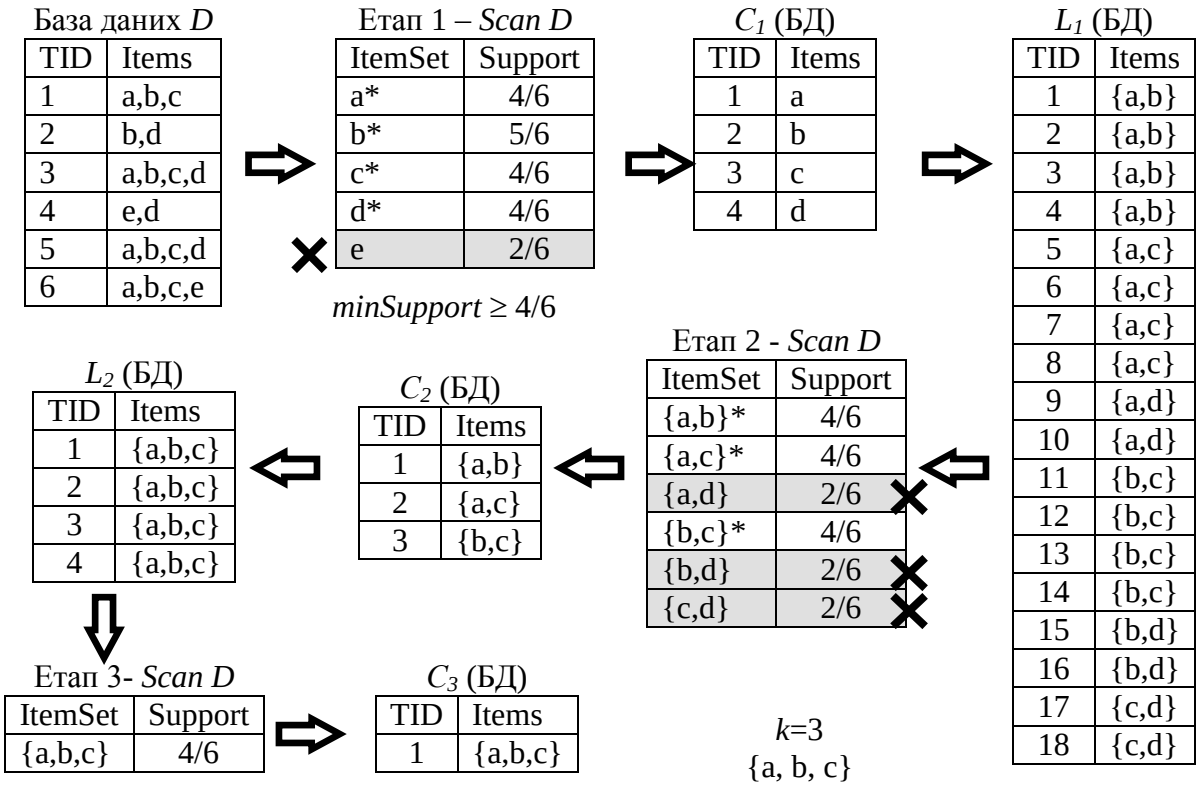


Рисунок 1.6 – Послідовність операцій алгоритму Apriori (k=3)

Першим етапом є формування одноелементних кандидатів та наступним кроком алгоритм підраховує підтримку одноелементних наборів, в результаті чого, відсікаються набори з рівнем підтримки менше заданого мінімуму (в даному прикладі 3). Далі відбувається формування кандидатів, що складаються з двох елементів, підрахування їх підтримки та відсічення наборів зі значенням підтримки менше ніж k=3. Набори з двох елементів, які залишились, називаються «двоелементні набори, що часто зустрічаються», вони приймають

участь в подальшій роботі алгоритму.

Проте, особливістю розглянутого вище алгоритма є те, що він відсікає – апріорі – тих кандидатів, які завідомо не можуть стати тими, що часто зустрічаються, на основі інформації про кандидатів, що прибралися на попередніх етапах алгоритму.

Відсічення відбувається на основі припущення про те, що у набора, що часто зустрічається, всі підмножини повинні бути теж такими, що часто зустрічаються, інакше цей набір виключається з наступного формування та підрахунку кандидатів.

Алгоритм Apriori розраховує також підтримку наборів, що не можуть бути відсічені апріорі. Це називається негативною областю (negative border), до якої належать набори, що зустрічаються рідко, але всі підмножини цих наборів є кандидатами, що часто зустрічаються.

Далі розглянемо роботу алгоритма Apriori на прикладі даних про оцінки з математичних дисциплін, які були описані в таблиці 2.2, сформуємо всі можливі 1-елементні набори – L_1 і розрахувати рівень підтримки *Support*.

Створимо множину 1-елементних наборів кандидатів $c \in C_k$, виділити з L_1 всі набори, у яких підтримка більше або дорівнює заданим обмеженням *minSupp=4/20* та відсічем результати, підтримка яких менша заданих обмежень, щоб уникнути утворення кандидатів, що апріорі не будуть в наступних розрахунках відповідати заданим умовам.

Таблиця 1.15 – Результат розрахунку підтримки для 1-елементних кандидатів

ItemSet	Support
a4	2/20
a6	1/20
a9*	5/20
a10*	4/20
a11	2/20
g5	1/20
g6	1/20
g7	2/20
g8	3/20

g9*	4/20
g10	2/20

Продовження таблиці 1.15

g12	1/20
e6	1/20
e7	1/20
e8	2/20
e9	1/20
e10*	5/20
e11	2/20
e12	1/20
i6	1/20
i8*	5/20
i9	2/20
i10*	8/20
i11*	4/20
m6	1/20
m7	1/20
m8	1/20
m9	1/20
m10	2/20

У результаті сканування БД транзакцій та підрахувавши підтримку виявлених 1-елементних комбінацій, отримали наступних кандидатів, чия підтримка не менша ніж задана в умові *minSupp=4*/20 та відсікли кандидатів, підтримка яких менша заданого мінімуму. В результаті отримуємо базу даних кандидатів, які будуть брати участь в подальших розрахунках (табл.1.15).

Крок 2. На $k_2=k_1+1$ етапі знайдемо всі можливі 2-елементні пари, підрахуємо їх підтримку та відсічем кандидатів, підтримка яких менша за задану. Тому необхідно сформувати базу даних що містить всі можливі 2-елементні комбінації на основі кандидатів, що були відібрані в попередньому кроці (табл. 1.15)

Таблиця 1.16 – Отримані кандидати, що відповідають заданій умові

ItemSet	Support
a9	5/20
a10	4/20
g9	4/20
e10	5/20
i8	5/20

i10	8/20
i11	4/20

Таблиця 1.17 – База даних сгенерованих 2-елементних кандидатів

TID	Items	TID	Items
1	{a4, e10}	38	{m7, i8}
2	{a4, i8}	39	{m10, i11}
3	{a6, i8}	40	{m10, i11}
4	{a9, g8}	41	{g10, e11}
5	{a9, g8}	42	{g10, e10}
6	{a9, g8}	43	{g9, e12}
7	{a9, g9}	44	{g9, e12}
8	{a9, g7}	45	{g9, e10}
9	{a9, e10}	46	{g8, e10}
10	{a9, e10}	47	{g8, e10}
11	{a9, e10}	48	{g8, e10}
12	{a9, e10}	49	{g7, i8}
13	{a9, e8}	50	{g7, i8}
14	{a9, i10}	51	{g6, e7}
15	{a9, i10}	52	{g6, i8}
16	{a9, i11}	53	{e11, i10}
17	{a9, i11}	54	{e11, i10}
18	{a9, i8}	55	{e10, i10}
19	{a10, g10}	56	{e10, i10}
20	{a10, g10}	57	{e10, i10}
21	{a10, g9}	58	{e7, i8}
22	{a10, g12}	59	{e12, i10}
23	{a10, e11}	60	{e12, i10}
24	{a10, e11}	61	{e10, i11}
25	{a10, e10}	62	{e10, i11}
26	{a10, e9}	63	{a11, i10}
27	{a10, i10}	64	{a11, i10}
28	{a10, i10}	65	{m10, i11}
29	{a10, i10}	66	{m10, i11}
30	{a10, i10}	67	{m6, i8}
31	{a10, g9}	68	{a10, i10}
32	{a10, g12}	69	{a10, i10}
33	{a10, e11}	70	{a10, i10}
34	{a10, e11}	71	{a11, g9}
35	{a10, e10}	72	{a4, g5}
36	{a10, e9}	73	{a4, g5}
37	{a10, i10}	74	{a4, g5}

Наступним етапом буде розрахування рівня підтримки для згенерованих записів бази даних 2-елементних кандидатів.

Таблиця 1.18 – Розрахований рівень підтримки для 2-елементних кандидатів

ItemSet	Support	ItemSet	Support
a11, g9	2/20	m7, i8	1/20
a4, e10	1/20	g10, e10	1/20
a4, i8	1/20	g9, e12	2/20
a4, g5	4/20	g9, e10	1/20
a6, i8	1/20	g8, e10	3/20
a9, g8	3/20	g7, i8	2/20
a9, g9	1/20	g6, i8	1/20
a9, g7	1/20	e11, i10	2/20
a9, e10*	4/20	e10, i10	3/20
a9, e8	1/20	e7, i8	1/20
a9, i10	2/20	e12, i10	2/20
a9, i11	2/20	e8, i8	2/20
a9, i8	1/20	e10, i11	2/20
a10, g10	2/20	a11, i10	2/20
a10, g9	1/20	m10, i11	2/20
a10, g12	1/20	m6, i8	1/20
a10, e11	2/20	a10, e9	1/20
a10, e10	1/20	a10, i10*	4/20

У результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 2-елементних комбінацій, знайдено наступних кандидатів, чия підтримка не менша ніж в заданій умові *minSupp=4*/20, тобто множина кандидатів C_2={a10, i10}, {a9, e10}, {a4,g5}. На наступному кроці база даних транзакцій буде створюватись на основі кандидатів, які задовольняють заданій умові.

Таблиця 1.19 – Отримані кандидати, що відповідають заданій умові

ItemSet	Support
a10, i10*	4/20
a9, e10*	4/20
a4, g5 *	4/20

Крок 3. На $k_3=k_1+1$ етапі знайдемо всі можливі 3-елементні пари та підрахуємо їх підтримку. Для цього проскануємо базу даних транзакцій на

основі відібраних в попередньому кроці кандидатів та знайдемо всіх 3-елементних кандидатів.

Таблиця 1.19 – База даних 3-елементних кандидатів

TID	Item	TID	Item
1	a9, g8, e10	7	a10, g10, i10
2	a9, g8, e10	8	a10, g10, i10
3	a9, g8, e10	9	a10, g12, i10
4	a9, g9, e10	10	a10, g10, i10
5	a9, g9, e10		
6	a4, g5, e6		

Далі розрахуємо рівень підтримки для отриманих 3-елементних кандидатів та знайдемо тих, чия підтримка не менша, ніж задана умова *minSupp*=4/20.

Таблиця 1.20 – Розрахована підтримка для 3-елементних кандидатів

ItemSet	Support
a9, g8, e10	3/20
a9, g9, e10	2/20
a4, g5, e6	1/20
a10, g10, i10	3/20
a10, g12, i10	1/20

У результаті сканування бази даних транзакцій та підрахувавши підтримку виявлених 3-елементних комбінацій, не було знайдено кандидатів, що відповідають заданій умові *minSupp*=4/20, тобто C_3=0.

Результатом роботи алгоритму є об’єднання всіх множин L_k для всіх k, що наведені на рисунку нижче.

Таблиця 1.21 – Об’єднання всіх множин L_k для всіх k

ItemSet	Support
a9*	5/20
a10*	4/20
g9*	4/20
e10*	5/20
i8*	5/20
i10*	8/20
i11*	4/20
a10, i10*	4/20
a9, e10*	4/20
a4, g5 *	4/20

Для відпрацювання розглянутого алгоритму було створено менше покриття, ніж для алгоритмів SETM та AIS за допомогою відсікання апріорі не підходящих кандидатів, підтримка яких менша ніж задана в умові, тому в кожному наступному кроці розглядалось все менша і менша кількість кандидатів та зменшувався розмір бази даних транзакцій.

На основі поданих кандидатів можна формалізувати закономірності у вигляді лінгвістичних правил:

– «25% учнів мають оцінки достатнього рівня з алгебри, інформатики»;

– «20% учнів мають оцінки достатнього рівня з геометрії»;

– «60% учнів мають оцінки високого рівня з інформатики»;

– «20% учнів мають оцінки високого рівня з алгебри»;

– «25% учнів мають оцінки високого рівня економіки»;

– «маючи оцінку високого або достатнього рівня з алгебри, зазвичай, учні мають оцінки високого рівня і з інформатики»;

– «маючи оцінки середнього рівня з алгебри, учні мають також оцінки середнього рівня і з геометрії».

1.7 Порівняльний аналіз алгоритмів пошуку асоціативних залежностей

Реальні бази даних транзакцій, що використовуються, наприклад, на основі ринкового кошика, зазвичай містять в собі тисячі артикулів, тож обчислювальні витрати, що йдуть на пошук асоціативних правил – значні. На прикладі вибірки, що складається всього зі 100 предметів, кількість асоціацій, що будуть утворені цими предметами, складе $100\times2^{99}=6{,}4\times10^{31}$. Пошук асоціативних правил за допомогою обчислення підтримки та вірогідності для всіх знайдених асоціативних правил та порівняння їх із заданим граничним значенням не відрізняється високою ефективністю через великі обчислювальні витрати.

Основною проблемою знаходження асоціативних правил в розглянутих алгоритмах обмеженого перебору (Apriori, AIS, SETM) є досить великий розмір покриття. Для нього можна виділити 3 види правил:

– корисні правила (такі, що містять дійсну інформацію, яка раніше не була відома, але має логічний опис);

– тривіальні правила (такі, що містять дійсну інформацію, що легко пояснюється та відображує вже відомі правила в досліджуваній предметній області, тому і не приносять користі);

– неясні правила (містять інформацію, що не може бути пояснена, їх отримають на основі аномальних вихідних даних, або вони мають глибоко сховані закономірності, а тому для інтерпретації таких правил потрібний додатковий аналіз).

Алгоритми Apriori, AIS, SETM, що розглядались вище, призначені для виявлення асоціативних правил та використовують для цього методи обмеженого перебору, що дозволяє зменшити об’єм покриття. До основних характеристикам, В якості критеріїв порівняльної оцінки алгоритмів із заданою кількістю наборів ознак *k*, можна відзначити:

- кількість операцій для виконання алгоритму;
- тимчасові витрати на всі операції алгоритму;
- кількість дій з даними, що зберігаються в базі даних;
- об’єм даних покриття, відібраного для аналізу;
- розмір ОЗУ комп’ютера для завантаження заданої вибірки даних та виконання операцій з ними.

Для проведення перевірки роботи алгоритмів пошуку асоціативних правил використовуються тестові бази даних, що знаходяться у вільному доступі. Параметри тестових БД наведені в табл. 1.22. Вони містять справжні набори даних про оцінки по предметам без указання особи учнів. Далі використані наступні позначення: T – середня кількість можливих оцінок у транзакції; C – середній розмір набору (комбінація оцінок на одного учня), D – кількість транзакцій (записів) бази даних.

Таблиця 1.22 – Характеристики тестових БД від IBM

Найменування БД	Кількість оцінок, T	Середній розмір набору, C	Кількість транзакцій, D	Розмір БД, Megabytes
T5.I2.D100K	5	2	100000	2.4
T10.I2.D100K T10.I4.D100K	10 10	2 4	100000 100000	4.4
T20.I2.D100K T20.I4.D100K T20.I6.D100K	20 20 20	2 4 6	100000 100000 100000	8.4

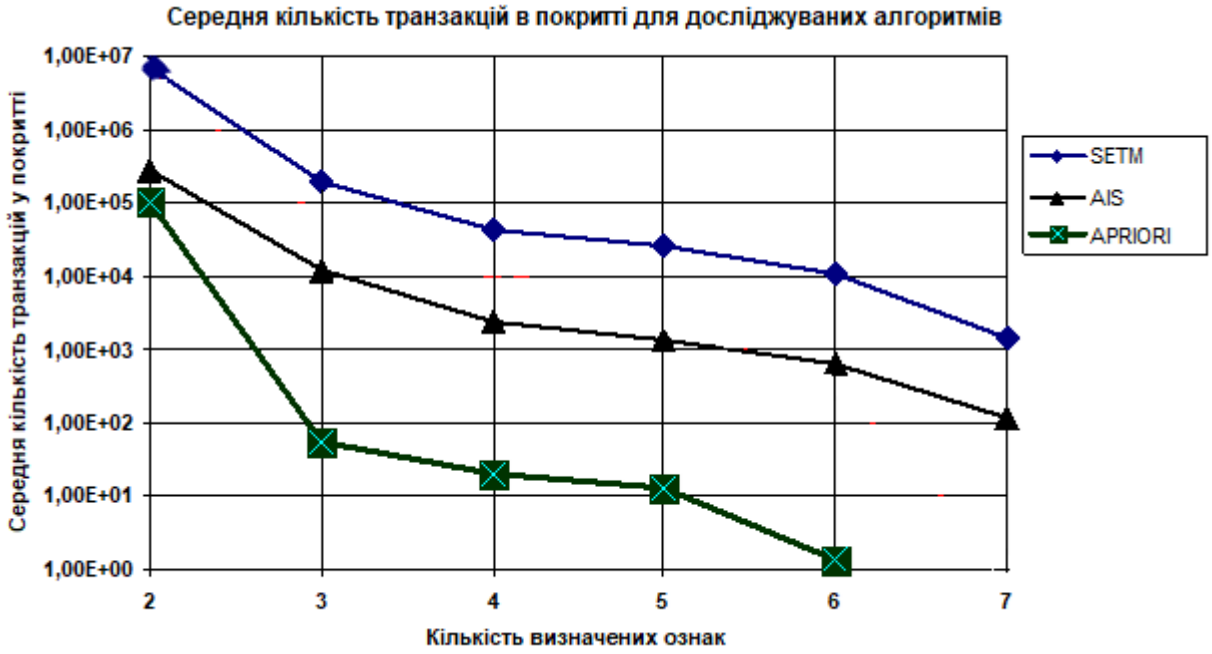
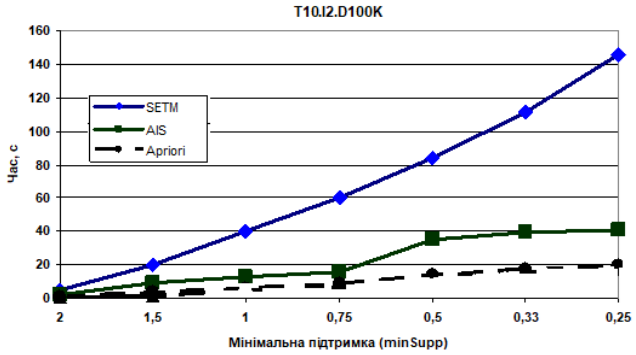


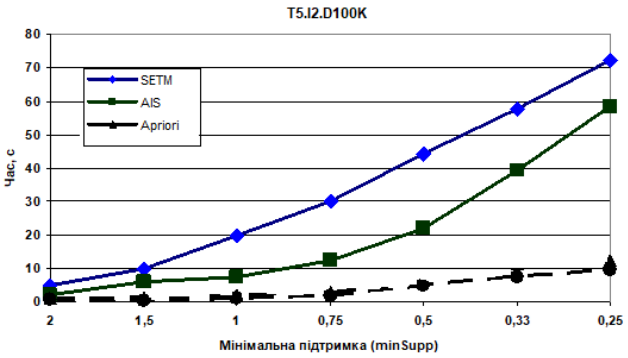
Рисунок 1.7 – Середній рівень транзакцій в покритті

На рис 1.7 зображено у вигляді діаграми усереднені значення кількості транзакцій у покритті для віще розглянутих алгоритмів, яка показує, що асоціації, що складається з двох умов розмір покриття досить великий.

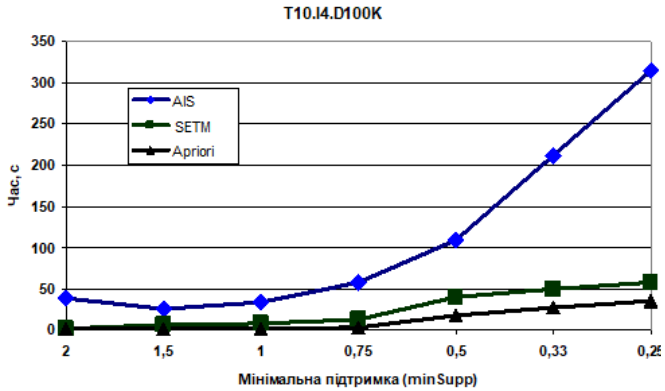
Залежності тимчасових витрат на виконання всіх операцій алгоритмів Apriori, AprioriTid, AIS, SETM [15-23] залежно від мінімальної підтримки (*minSupp*) для різних тестових баз даних, наведені на рис. 1.8, а)-е).



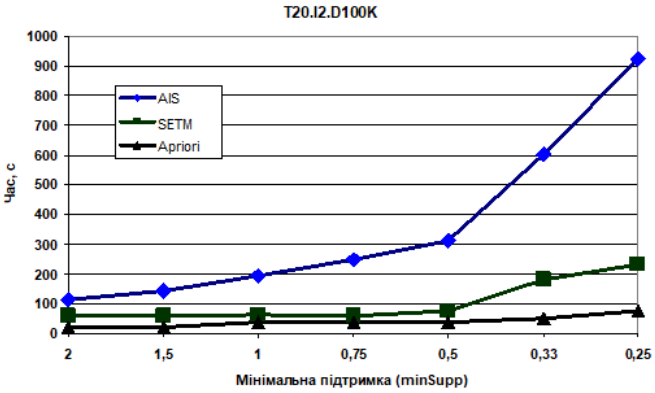
а)



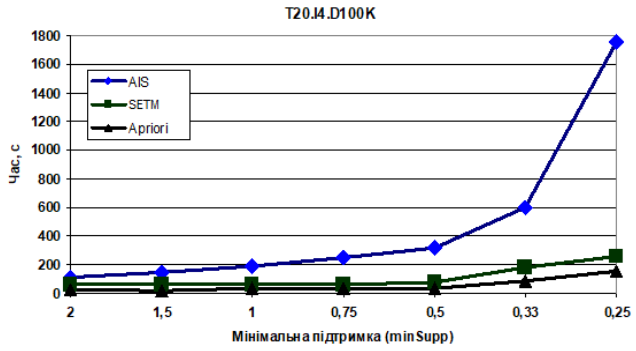
б)



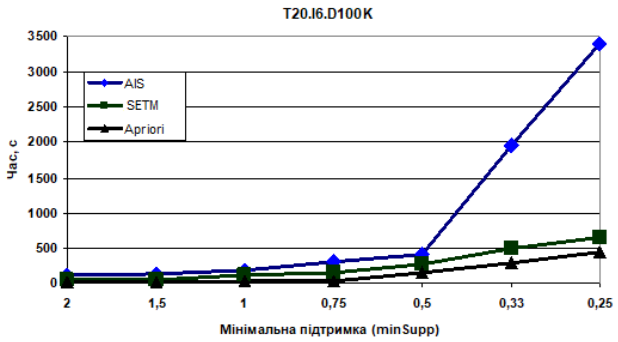
в)



г)



д)



е)

Рисунок 1.8 – Залежність тимчасових витрат на виконання операцій

Проаналізувавши залежності, що зображені на рис. 1.8, випливає, що тимчасові витрати на роботу алгоритма пошуку асоціацій AIS найбільш значні відносно інших, також для малих значень заданої мінімальної підтримки (*minSupp*) майже еквівалентні об’єму вибірки. Найкращі результати надав

Apriori алгоритм, витрати на виконання якого не виходять за рамки значення 100 с для заданої мінімальної підтримки *minSupp*=0,25.

Основний недолік розглянутих алгоритмів Apriori, AprioriTid, AIS, SETM є розмір покриття і обчислювальна складність оцінки отриманих правил. Нижче наведений аналіз параметрів, що можуть допомогти зменшити розмір покриття.

У постановці задачі виділення покриття відбувається за допомогою двох параметрів асоціації – *Support* і *Confidence*, яких мало для повноцінної характеристики залежностей. Отримання правил асоціації відбувається з допомоги алгоритмів знаходження покриття й оформлення правил.

Необхідно ввести додаткові характеристики, такі як «рівень поліпшення» (*Improvement*), «підйомна сила» або «ліфт» (*lift*), «посилення» (*leverage*), «очікуваний рівень імовірності», що можуть зменшити об'єм покриття та ухвалюються спроби розробки узагальненої міри асоціативного правила.

«Підйомна сила», або «ліфт» (*Lift*), що також має назву «рівень поліпшення» (*Improvement*), показує як підвищується ймовірність знаходження *Y* якщо в ньому вже є *X*. Ліфт (*Lift*) показує відношення частоти появи умови $X \to Y$ в транзакціях, що також мають і наслідок *Y*, до частоти появи наслідку взагалі:

$$Lift(X \to Y) = \frac{Conf\,(X \to Y)}{Supp(X)},$$

де $Conf\,(X \to Y), Supp(X)$ обчислюються за формулами (1.3), (1.2) відповідно.

Ліфт більше ніж 1 показує, що умова $X \to Y$ з'являється частіше в транзакціях, що містять *Y*, ніж в інших. При значеннях ліфта більше 1 ($Lift(X \to Y) > 1$) – зв'язок позитивний, при рівності ($Lift(X \to Y) = 1$) – зв'язок між *X* і *Y* відсутній, а при значеннях менше 1 ($Lift(X \to Y) < 1$) – негативний. Аналіз значень ліфта для двох правил $Lift(X \to Y) > 1$ і $Lift(Z \to Y) < 1$ дозволяє визначити, що в розглянутих правилах *Y* більше впливає на появу *X*, ніж *Z*.

Але вище розглянута міра не завжди виявляється вдалою. Правило $X \to Y$ з меншим значенням *Support* і більшим *Lift* може бути мало значимим, ніж правило $Z \to Y$ з більшим значенням *Support* і меншим *Lift*, тому що $Z \to Y$ застосовується для більшої частини транзакцій. Тож збільшення числа транзакцій приводить до збільшення зв'язку між умовою й наслідком.

Посилення (*Leverage*) – це різниця між значенням частоти, з якої умова і наслідок трапляються разом, і добутком підтримок появи умови й наслідку окремо:

$$Lev(X \to Y) = Supp(X \to Y) - Supp(X) \times Supp(Y).$$

Нехай $X \to Y$ й $Z \to Y$ з підтримкою $Supp(X \to Y) = Supp(Z \to Y) = 1$. Ліфти для них рівні, бо для їх обчислення задається підтримка $Supp(X)$: $Conf(X) = 0{,}8$. Звідти $Lift(X \to Y) = Lift(Z \to Y) = 10/8 = 1{,}25$. Якщо наприклад $Supp(X \to Y) = 0.4$, $Supp(Z \to Y) = 0.6$, $Supp(X) = 0.4$, $Supp(Y) = 0.6$, $Supp(Z) = 0.4$, то за *Leverage* можна визначити як:

$$Lev(X \to Y) = Supp(X \to Y) - Supp(X) \times Supp(Y) = 0.4 - 0.4 \times 0.6 = 0.04,$$
$$Lev(Z \to Y) = Supp(Z \to Y) - Supp(Z) \times Supp(Y) = 0.6 - 0.4 \times 0.4 = 0.24.$$

Остання асоціація має більшу значимість, оскільки вона трапляється частіше (застосовується для більшого числа транзакцій).

Чим асоціативні правила носять більш ймовірнісний характер, тим кориснішим є їх розгляд з точки зору теорії ймовірностей. Підтримка $Supp(X \to Y)$ визначається як відношення кількості записів у БД транзакцій, що задовольняють умові $X \to Y$ до кількості записів у БД. Відносно теорії ймовірностей підтримка визначається як ймовірність, що в транзакції наявні дві ознаки *X* і *Y*: $P(XY)$. Довіра $Conf(X \to Y)$ (імовірність правила) – це відношення

числа записів, що відповідні правилу, до числа записів, що відповідають лівої частини. Рівень довіри відображає умовну ймовірність

$$P_X(Y) = \frac{P(XY)}{P(X)}.$$

Для правила $X \to Y$ мають виконуватись наступні умови:

$$Supp(X \to Y) = P(XY) \geq \min Supp;$$
$$\frac{Supp(X \to Y)}{Supp(X)} = P_X(Y) \geq \min Conf,$$

де $Supp(X)=M(X)/M(БД)$ – значення відношення потужності множини транзакцій в БД, що мають таку ознаку: $X \cup Y = X \vee Y$, до кількості всіх записів транзакцій в БД;

$Sup(X \to Y)=Supp(XY)=M(X \cup Y)/M(БД)$ – значення відношення потужності множини транзакцій в базі даних, що володіють ознаками X і Y, до множини всіх транзакцій в базі даних.

Проте, цих 2 характеристик не вистачає для якісної оцінки одержаних асоціативних залежностей з точки зору теорії ймовірностей. Нижче наведена одна із додаткових характеристик – відношення ймовірності асоціативного правила до ймовірності результату – *Lift*, що виглядає як:

$$Lift(X \to Y) = \frac{P_X(Y)}{P(Y)} = \frac{Conf(X \to Y)}{Supp(Y)}.$$

Тож величина *Lift* характеризує ступінь збільшення ймовірності наставання події Y за умови появи X відносно ймовірності події Y.

Посилення (*Leverage*), відносно теорії ймовірності показує ймовірність суми залежних подій X та Y:

$$Lev(X \to Y) = Lev(XY) = Supp(XY) - Supp(X) \times Supp(Y).$$

Наведені віще характеристики асоціативних правил дають можливість надати повну характеристику асоціації і одночасно становлять систему показників, яка характеризує асоціативну залежність.

Система двох ознак, двох частин асоціативного правила, за теорією ймовірності описується наступними ймовірнісними значеннями: P_{00}, P_{01}, P_{10}, P_{11}. Для їхнього визначення потрібні, як мінімум три параметри. Маючи відомі три параметра, асоціативного правила $X \to Y$: *Support*, *Confidence* і *Lift*, необхідних для визначення значень ймовірностей, можна визначити P_{00}, P_{01}, P_{10}, і P_{11}.

Нехай *Support*, *Confidence* і *Lift* правила $X \to Y$ через ймовірності:

$$Sup(X \to Y) = P(XY) = P_{11};$$

$$Conf(X \to Y) = \frac{Supp(X \to Y)}{Supp(X)} = \frac{P(XY)}{P(X)} = \frac{P_{11}}{P_{11} + P_{10}};$$

$$Lift(X \to Y) = \frac{Sup(X \to Y)}{Sup(X)Sup(Y)} = \frac{P(XY)}{P(X)P(Y)} = \frac{P_{11}}{(P_{11} + P_{10})(P_{11} + P_{01})}.$$

Звідси одержимо значення ймовірностей:

$$P_{11} = Sup(X \to Y);$$

$$P_{10} = \frac{Sup(X \to Y)}{Conf(X \to Y)} - Sup(X \to Y) = Sup(X \to \overline{Y});$$

$$P_{01} = \frac{Conf(X \to Y)}{Lift(X \to Y)} - Sup(X \to Y) = Sup(\overline{X} \to Y);$$

$$P_{00} = 1 - P_{11} - P_{10} - P_{01} =$$

$$= 1 + Sup(X \to Y) - \frac{Sup(X \to Y)}{Conf(X \to Y)} - \frac{Conf(X \to Y)}{Lift(X \to Y)} = Sup(\overline{X} \to \overline{Y}).$$

Звідси, для визначення всіх значень P_{00}, P_{01}, P_{10}, P_{11} необхідні 3 характеристики асоціативного правила: *Supp*, *Conf* та *Lift*. Якщо принаймні один з ним буде невідомий, всі ймовірності P_{00}, P_{01}, P_{10}, P_{11} не можуть

отриматися (для оцінки асоціативного правила потрібні 3 параметри). Така система характеристик відповідає умовам:

– «повнота» – набір параметрів з достатньою повнотою характеризують асоціативне правило;

– «мінімальність» – набір повинен бути зменшений;

– «ненадлишковість» – різні характеристики враховують різні показники асоціативної залежності;

– «операційність» – кожна з характеристик мають однозначне формулювання й характеризує визначені якості;

– «вимірність» – кожна характеристика описується кількісно.

Для чисельних значень всіх 3 величин *Supp*, *Conf* та *Lift* враховуються обмеження:

$$0 < Supp(X \to Y) \le 1,$$
$$Supp(X \to Y) \le Conf(X \to Y) \le 1.$$

Другий спосіб генерації логічних правил заснований на теорії наближених множин, що розроблена З. Павлаком [2], зіграла роль бази для розвитку напрямку інтелектуального аналізу даних. Основне поняття теорії наближених множин – це «нерозрізненість» (або «нерозрізнене» відношення). Вважається, що однакова інформація може асоціюватись з різними елементами множини, що робить неможливим достеменне визначення приналежності даних елементів до деякої множини (наближеної множини). Вона характеризується за допомогою апроксимацій – нижньої, котра визначає елементи, що однозначно відносяться до цієї множині, та верхньої, що визначає елементи, належні даній множині.

Нехай Y – така множина, що є підмножиною універсума U і I – відношення «нерозрізненості». Нижньою апроксимацією Y є:

$$I_*(Y) = \{y \in U : I(y) \subseteq Y\},$$

а верхньою:

$$I^*(Y) = \{y \in U : I(y) \cap Y \neq 0\},$$

де *I(y)* визначає множину об'єктів, «нерозрізнених». Граничний регіон (*boundary region*) мається на увазі $BN_I = I^*(Y) - I_*(Y)$. Тому, *boundary region* містить елементи, що відносяться до верхньої апроксимації множини і не відносяться до нижньої. Якщо $BN_I \neq 0$, то множина Y – наближеною.

Для наближених множин вводиться поняття функції приналежності:

$$\mu_Y^I(y) = \frac{|Y \cap I(y)|}{|I(y)|},\ 0 \leq \mu_Y^I(y) \leq 1.$$

Функція приналежності може використовуватись для визначення верхньої та нижньої апроксимацій наближеної множини, а також граничного регіону:

$$I_*(Y) = \{y \in U : \mu_Y^I(y) = 1\};$$
$$I^*(Y) = \{y \in U : \mu_Y^I(y) > 0\};$$
$$BN_I(Y) = \{y \in U : 0 < \mu_Y^I(y) < 1\}.$$

Набори «нерозрізнених» елементів, в асоціацію з якими потрапила однакова інформація називається «гранулами». Також вводиться поняття редакта (*reduction* – зменшення, скорочення). Під цим терміном мається на увазі мінімальний набір ознак, що дозволяє розрізняти так звані «гранули». Мінімальність даного набору приймає неможливість скорочення цього набору без втрати на розрізнення «гранул», тобто редакт визначає набір ознак, якими описується ця наближена множина.

1.8 Обгрунтування вибору алгоритму пошуку асоціативних правил

Виходячи з попередньо проведеного аналізу, стало відомо, що найбільш оптимальним рішенням стане використовувати для розробляємої підсистеми алгоритм пошуку асоціативних правил – Apriori. Але поруч з цим, залишається проблема, яка полягає в тому, що багато з виявлених правил можуть бути або очевидними, або неінформативними для користувача. В рамках розробки підсистеми підтримки освітнього процесу, даний алгоритм був модифікований шляхом додавання показника інформативності, що характеризує знайдені правила з точки зору актуальності для користування. Це дозволить виводити правила, які будуть не тільки достовірними, але й актуальними.

Схема модифікованого Apriori алгоритму представлена на рис.1.9.

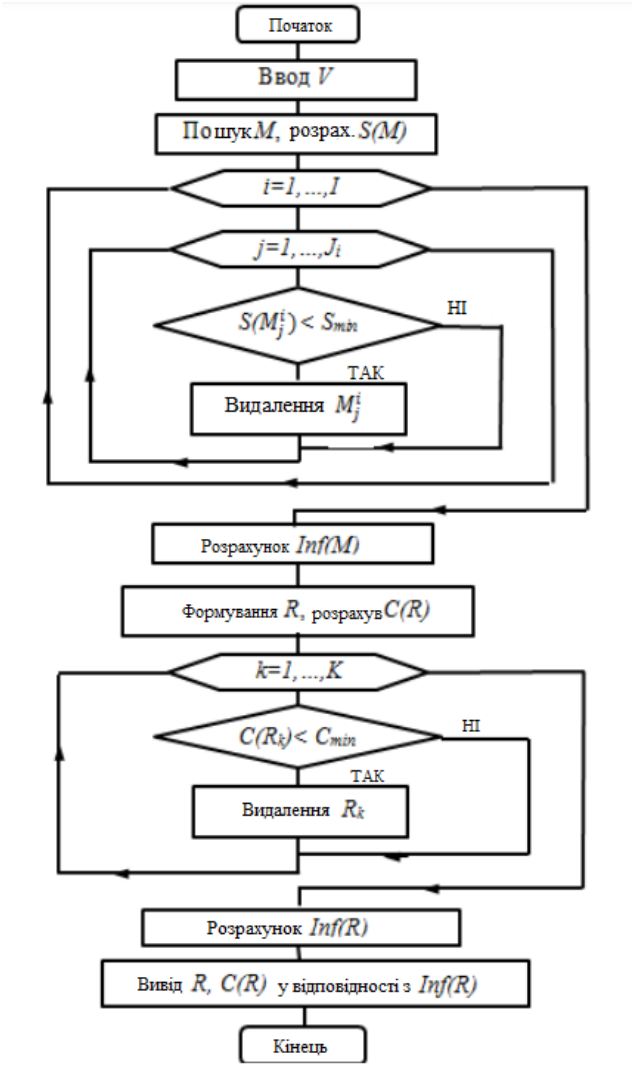


Рисунок 1.9 – Схема модифікованого Априорного алгоритму

Використання модифікованого Априорного модифікованого алгоритму в рамках підсистеми підтримки навчального процесу дозволить виявляти логічні залежності успішності учнів від різних характеристик навчального процесу.

На основі даних про успішність учнів за різними навчальними дисциплінами та виявлених випадків підсистемою підтримки навчального процесу, зниження успішності можуть бути сгенеровані рекомендації по корегуванню навчального процесу. Генерація рекомендацій здійснюється на основі логічного виводу. Структура підсистеми підтримки освітньої діяльності включає в себе компоненти вилучення вихідних даних, генерації рекомендацій, їх виводу, а також базу даних правил логічного виводу. Структура частини підсистеми, що відповідає за видачу рекомендацій стосовно корегування навчального процесу приведена на рис.1.10.

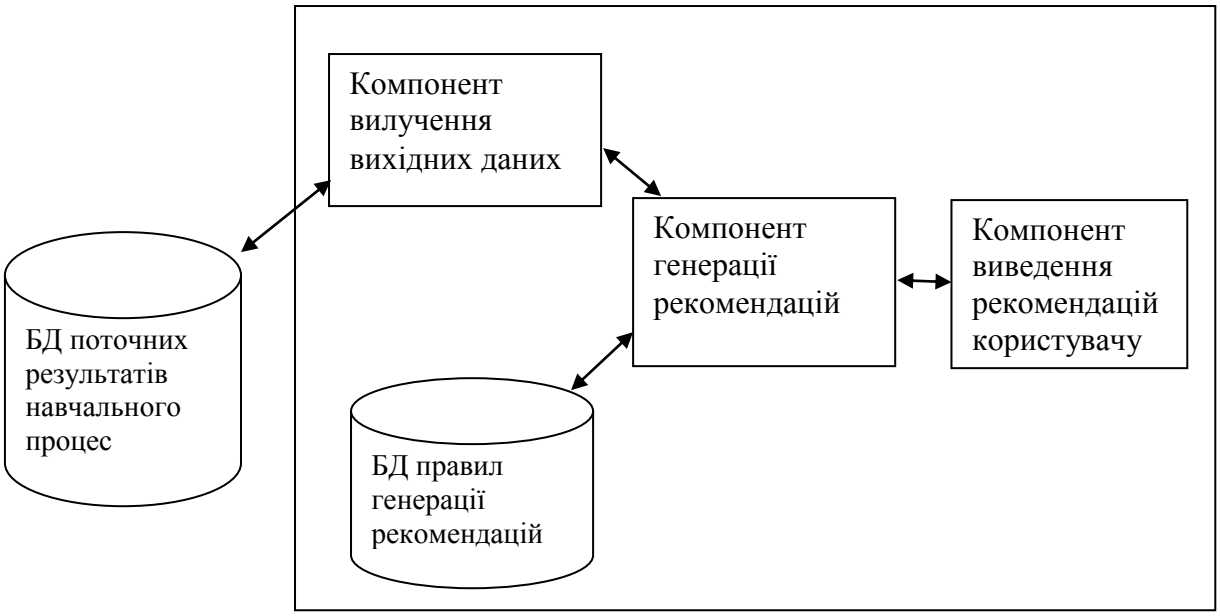


Рисунок 1.10 – Структура частини підсистеми, що відповідає за видачу рекомендацій стосовно корегування навчального процесу

Схема генерації рекомендацій стосовно корегування навчального процесу відбувається наступним чином: всі правила в БД розділені на декілька категорій у відповідності з областю їх застосування. Після процесу вилучення вхідних

даних відбувається виконання процедурі співвідношення їх з тими категоріями правил, що маються та вибір найбільш близької для них категорії.

Наступним кроком є здійснення вибору правила, що буде застосовуватись з вибраної категорії. Міра застосовності правила визначається на основі того, чи співпадає його посил поточним вхідним даним.

Якщо стається ситуація, що правил, які можна застосувати до поточних вхідних даних виявляється декілька, то їх називають «конфліктним набором». Для вирішення цієї проблеми використовується керуюча стратегія на основі вибору «найбільш спеціалізованого правила». Таким правилом вважається те, умови якого містять в собі найбільшу кількість вхідних даних, тобто вважається більш точним. Така схема генерації рекомендацій стосовно корегування навчального процесу дозволяє обрати найбільш підходяще правило з тих, що зберігаються в БД та на основі нього сформулювати рекомендації.

Оцінювання в школі відбувається за 12 бальною шкалою, де існують також розділення на рівні знань учнів, в залежності від того, які оцінки в нього і більшості.

Шкала оцінювання знань в школі (в балах):

- початковий рівень: 1, 2, 3;
- середній рівень: 4, 5, 6;
- достатній рівень; 7, 8, 9;
- високий рівень: 10, 11, 12.

Визначення рівня знань відбувається декілька разів в рік, де беруться підсумкові оцінки кожного з предметів та діє таким чином: щоб отримати високий рівень, необхідно мати всі оцінки в діапазоні від 10 до 12 балів включно, але при цьому, якщо учень отримує хоч одну підсумкову оцінку рівня, нижчого ніж високий, то його рівень знань оцінюється вже рівнем, найнижча оцінка якого є серед підсумкових оцінок учня.

1.5 Визначення вимог до апаратної частини ІС

Визначено вимоги, яким відповідає система підтримки навчального процесу:

– доступ локальну мережу до ресурсів Інтернет;

– сумісного доступу до даних працівників;

– організація електронного документообігу;

– зручний та швидкий обмін інформацією.

Управління роботою та забезпечення працездатності мережевих, обчислювальних, програмних та інформаційних ресурсів локальної мережі виконується системним адміністратором локальної мережі.

Локальна мережа має надійно функціонувати в програмно-апаратному середовищі, бути стійкою до збоїв і не здатною зруйнувати роботу операційної системи.

Вимоги до складу та параметрів технічних засобів:

а) кількісний склад мережі:

– 21 персональний комп’ютер (ПК);

– 1 серверний ПК;

– комутатори;

– мережевий кабель.

б) доступ до системи має здійснюватися через IBM-сумісний персональний комп’ютер. Мінімальні вимоги до робочого ПК:

– процесор 1,6 -2,4 ГГц;

– оперативна пам’ять для 32-розрядної ОС – 1 ГБ, для 64-розрядної ОС – 2 ГБ;

– монітор з мінімальною роздільною здатністю 1024x768 точок;

– клавіатура;

– маніпулятор миш;

в) доступ з мобільного пристрою за допомогою мобільної версії

веб-сайту.

Серверний комп’ютер працює під управлінням ОС MS Windows Server версії 2008 або вище;

Апаратні вимоги комп’ютера з серверною частиною (серверу СУБД) повинні забезпечувати наступну кількість працюючих користувачів:

– за мінімальним піковим навантаженням –150 користувачів;

– за навантаженням у середньому – 350 користувачів;

– за максимальним піковим навантаженням –750 користувачів.

Апаратні вимоги до серверного комп’ютера за мінімальним піковим навантаженням:

– процесор Intel/AMD-сумісний 64-розрядний 4 ядра з тактовою частотою 2 ГГц;

– оперативна пам’ять не менше ніж 12 ГБ;

– мережевий адаптер не менше ніж 100 Мбит/с;

– монітор з мінімальною роздільною здатністю 1024x768 точок;

– клавіатура, маніпулятор миша.

Апаратні вимоги до серверного комп’ютера за середнім навантаженням:

– процесор Intel/AMD-сумісний 64-розрядний, 8 ядер з частотою не менше ніж 2 ГГц;

– оперативна пам’ять не менше ніж 16 ГБ;

– мережевий адаптер не менше ніж 1 Гбіт/с;

– монітор з мінімальною роздільною здатністю 1024x768 точок;

– клавіатура, маніпулятор миша.

Апаратні вимоги до серверного комп’ютера за максимальним піковим навантаженням:

– процесор Intel/AMD-сумісний 64-розрядний, 8 ядер з частотою не менше ніж 3 ГГц;

– оперативна пам’ять не менше ніж 24 ГБ;

– мережевий адаптер не менше ніж 1 Гбіт/с;

– монітор з мінімальною роздільною здатністю 1024x768 точок;

– клавіатура, маніпулятор миша.

2 ОПИС ПРИЙНЯТИХ ПРОЕКТНИХ РІШЕНЬ

2.1 Обґрунтування вибору СУБД

Серед існуючих реляційних СУБД, що використовуються при створенні веб-додатків, найбільшої популярності набули:

– MySQL –швидка та гнучка реляційна СУБД, що рекомендується для розробки невеликих та середніх проектів. Підтримує типи таблиць: як найбільш розповсюджені MyISAM і InnoDB, так і HEAP та MERGE. З MySQL може працювати одночасно необмежена кількість користувачів, а число записів в таблицях може дорівнювати 50 млн.

– PostgreSQL – це СУБД, що відноситься до об’єктно-реляційного типу. Робота з PostgreSQL базується на мові SQL, однак PostgreSQL стандарт SQL-2011. Ця СУБД не має обмежень по максимальній кількості записів або індексів у таблицях. З переваг PostgreSQL: надійність транзакцій, наслідування та розширюваність. PostgreSQL підтримує різні розширення й варіанти мов програмування (PL/Perl, PL/Python, Java);

– Microsoft SQL Server – система управління реляційними базами даних, розроблена Microsoft. СУБД має широкий набір інтегрованих служб, що дозволяє складати запити, виконувати пошук, проводити синхронізацію, формувати звіти та аналізувати дані та забезпечує звертання до даних з будь-якого додатку, розробленого із застосуванням технології Microsoft .NET та Visual Studio.

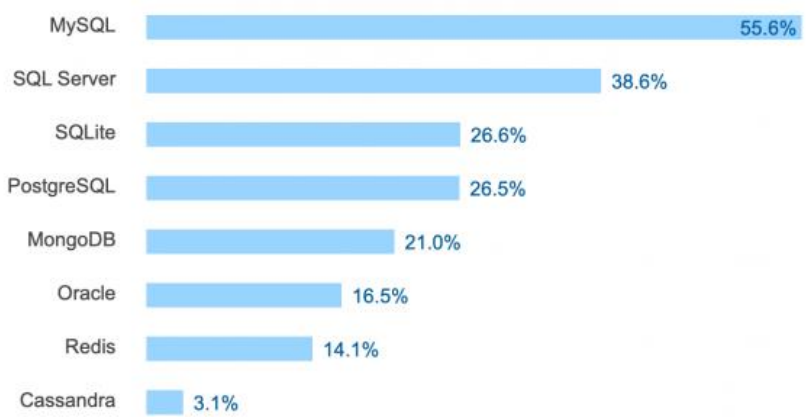


Рисунок 2.1 – Статистика популярності баз даних

Таким чином, проаналізувавши існуючі СУБД на основі обраної раніше технології проектування web-додатка ASP.NET, правильним рішенням буде обрати систему управління базами даних Microsoft SQL Server.

Також слід відмітити, що Microsoft SQL Server включає в себе також Common Language Runtime (CRL) .NET, що дозволяє реалізовувати збережені процедури та різні функції додаткам, що розробляються на мовах платформи .NET (наприклад VB.NET або C#).

2.2 Опис структури розробленої інформаційної системи

Для розробки системи була використана клієнт-серверна архітектура для глобальної мережі Інтернет. Структура розробленої інформаційної системи, що представлена на рис. 2.2, складається з 13 класів, кожен з яких визначає інтерфейс і реалізацію web-сторінки.

Реалізовані класи, представляються у виді ASPX-файлів web-сторінок клієнтської системи: клас «Ratings» – поточна успішность навчання; клас «Visiting_lessons» – облік відвідування занять; клас «Homejobs» – домашні завдання; клас «Rozklad_lessons» – розклад занять; клас «Edit_visits» – занесення даних відвідування занять; клас «Distr_objects» – розподіл предметів за викладачами; клас «Distr_teachers» – розподіл вчителів за класами; клас «Students» – особиста інформація учнів; клас «Teachers» – особиста інформація викладачів; клас «Rozklad» – занесення даних до розкладу занять, а також «Transaction» - транзакції, що є наборами оцінок учня.

Усі класи успадковані від шаблонового класу управління System.Web.UI.Page, який знаходиться в просторі імен System.Web.UI середовища розробки ASP .NET. Операція спадкування дозволяє одержати доступ до методів і властивостям класу Page для рішення завдань реалізації

функцій клієнтської частини при її роботі із сервером баз даних. У свою чергу клас Page, пов'язаний з інтерфейсом System.Web.IHttpHandler, представляє собою http-оброблювач.

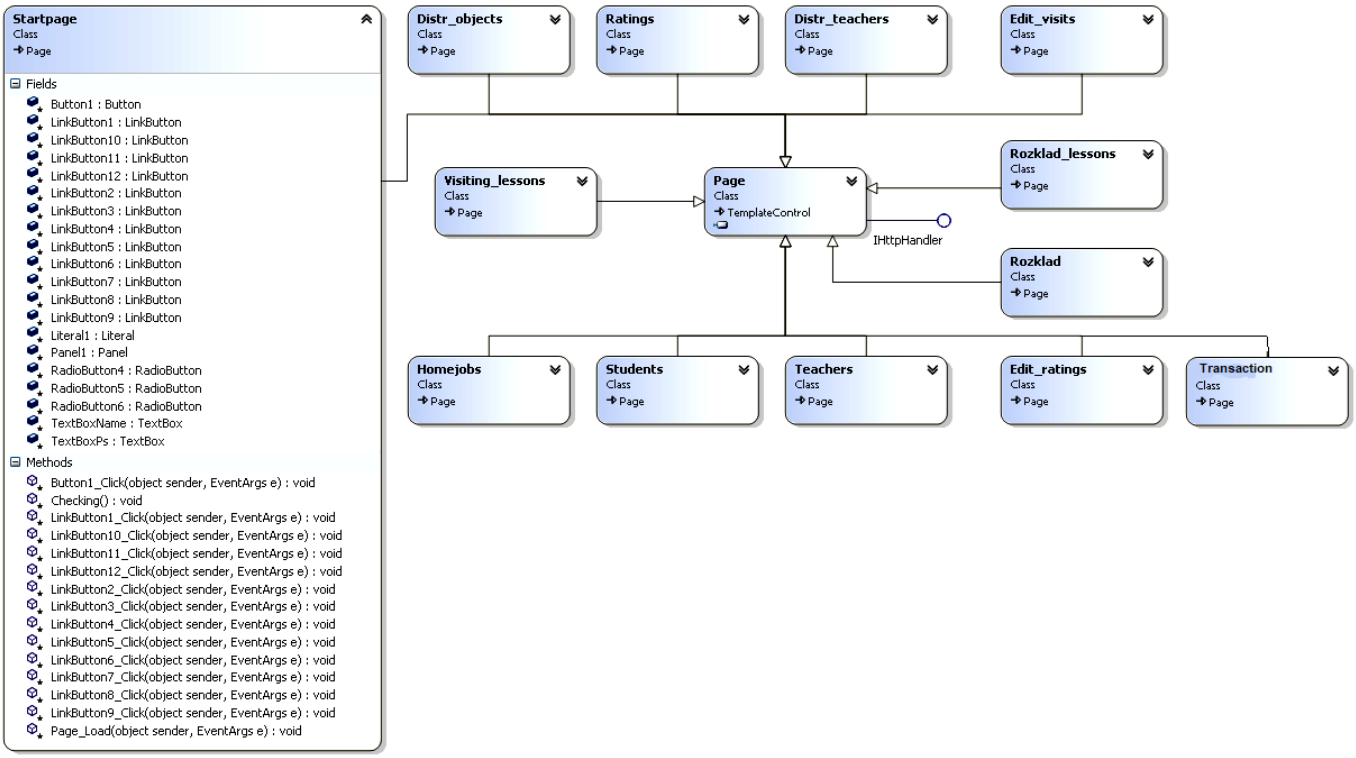


Рисунок 2.2 – Діаграма класів розробленої системи

Http-оброблювач використовується для обробки запитів і генерації вмісту відповіді на http-запит клієнта. Клас інтерфейсу IHttpHandler визначає один метод і одну властивість. Метод IHttpHandler.ProcessRequest(HttpContext) у якості параметра приймає об'єкт контексту запиту HttpContext і генерує відповідь клієнту. Властивість IHttpHandler.IsReusable указує, чи буде даний http-оброблювач використовуватися іншими запитами (true – так, false – ні).

2.3 Логічне і фізичне моделювання даних інформаційної системи

Логічне і фізичне моделювання даних інформаційної системи проводилось з використанням CASE-засобу «Allfusion ERWin Data Modeler» (Erwin). Процес моделювання в Erwin базується на методології проектування

реляційних баз даних – IDEF1X, що визначає стандарти термінології і графічного зображення типових елементів на ER-діаграмах.

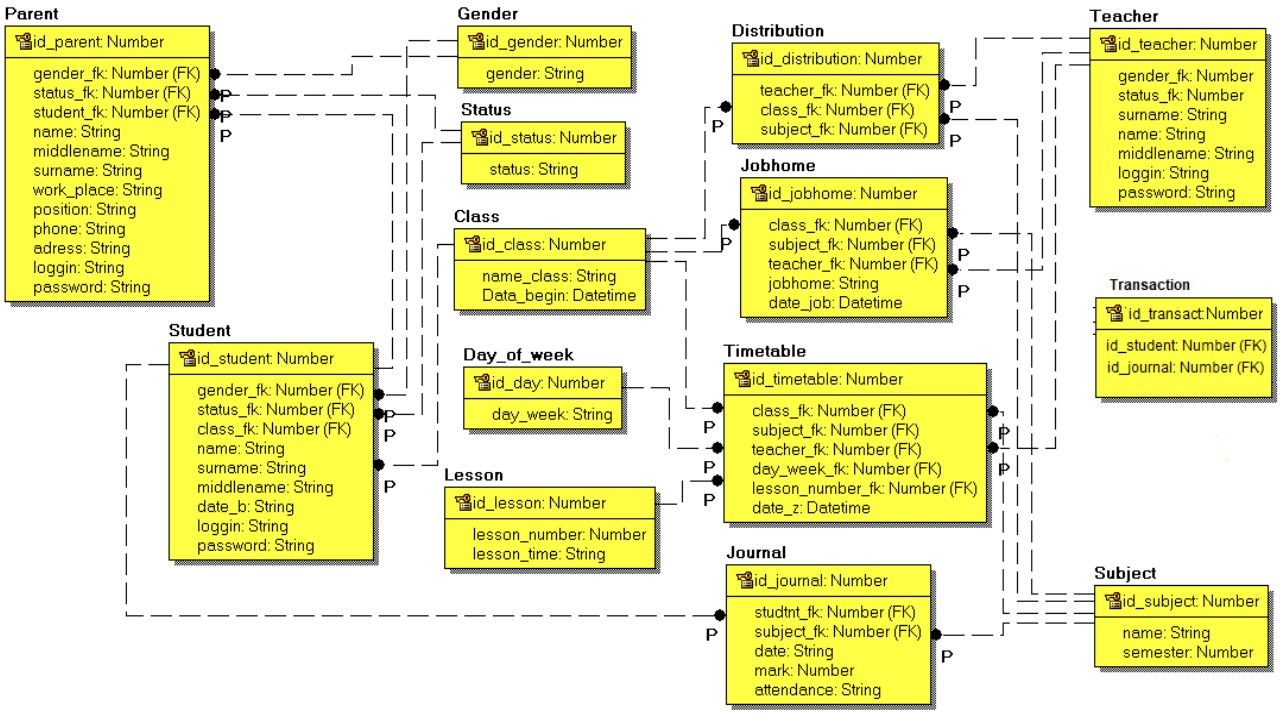


Рисунок 2.3 – Логічна модель даних інформаційної систем

Процес побудови інформаційної моделі складається з наступних етапів: визначення сутностей; визначення залежностей між сутностями; задання первинних та зовнішніх ключів; визначення атрибутів сутностей; приведення моделі до рівня нормальної форми; перехід до фізичного опису моделі (призначення відповідності ім’я сутності – ім’я таблиці, атрибут сутності – атрибут таблиці, задання тригерів, процедур і обмежень); генерація бази даних.

В Erwin існують два рівня представлення та моделювання – логічний (не залежить від платформи СУБД) та фізичний (відповідає обраної платформі СУБД). Логічна модель даних інформаційної системи приведена на рис. 3.3.

Предметною областю системи є навчальний процес навчальних закладів України I-II ступенів. Сутності, що увійшли в логічну модель подані у табл. Г.1 додатку Г.

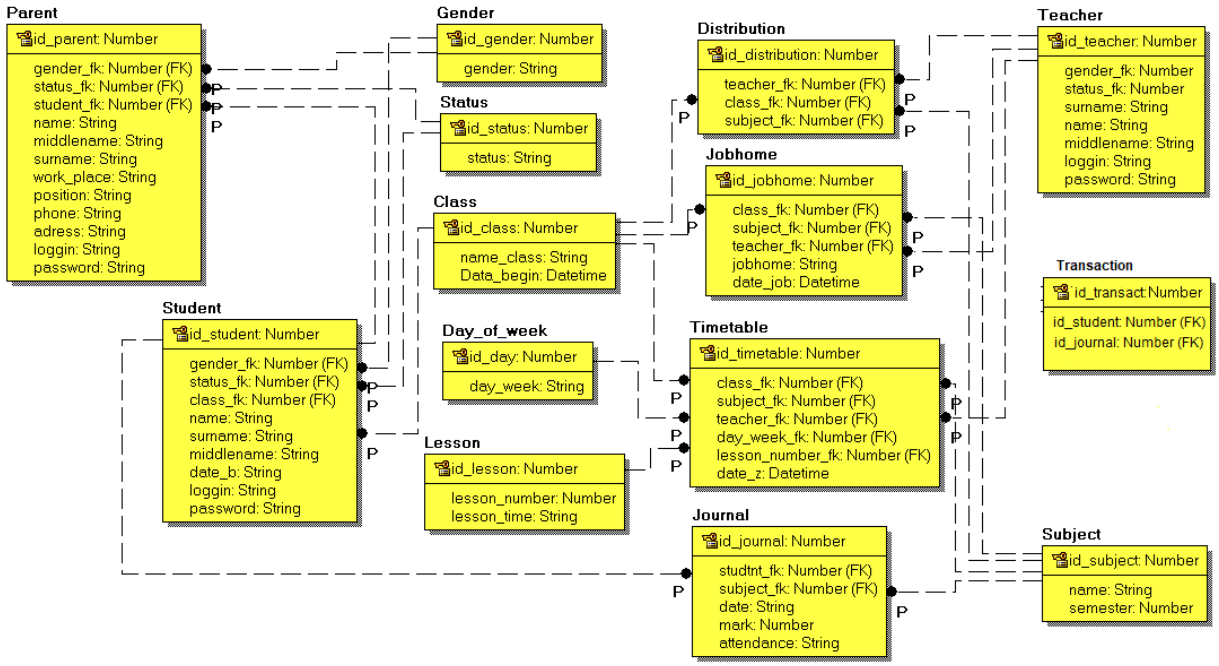


Рисунок 2.4 – Фізична модель даних інформаційної системи

Створення бази даних – це другий етап в її проектуванні. Для створення фізичної моделі даних необхідно дотриматись наступних кроків: вибір платформи СУБД; перенесення логічної моделі даних в середу СУБД; проектування основних відносин між сутностями; реалізація обмежень предметної області; проектування фізичного представлення БД.

Фізична модель даних інформаційної системи для обраної платформи – Microsoft SQL-сервер подана на рис. 3.4.

2.4 Створення бази даних для на платформі Microsoft SQL-server

Створення бази даних серверної частини інформаційної системи здійснювалось за допомогою CASE-засобу Allfusion ERwin Data Modeler.

На основі фізичної моделі ERwin (рис. 2.4) генерований SQL-код для створення бази даних системи. При виконанні SQL-коду створюються: таблиці, тригери, збережені процедури, індекси, обмеження цілісності та інші об’єкти, що підтримуються цільовою СУБД.

Вигляд схеми бази даних серверної частини інформаційної системи, реалізованої на платформі «Microsoft SQL Server Express 2008 R2», поданий на рис. 3.5. На схемі зображена структура реалізованої бази даних, що складається

з таблиць та зв’язків між ними.

2.5 Опис інформаційно-логічної моделі інформаційної системи

Для визначення логіки виклику web-сторінок, а також їх інформаційного змісту створюється інформаційно-логічна модель web-сайту, що фактично визначає карту сайту (рис. 3.6).

У ролі адміністратора інформаційної системи на даному етапі розробки виступає користувач зі статусом «Вчитель». В подальшій роботі над системою можливі зміни в присвоєні повноважень користувачам на розсуд адміністрації школи. Зміни полягають у відведенні окремої посади «Адміністратор інформаційної системи», що буде мати доступ до всіх можливостей інформаційної системи, а користувачі зі статусом «Вчитель» лише будуть мати доступ до функцій виставлення оцінок, відмічання присутності, введення домашнього завдання лише відповідно до тих дисциплін і тих класів, які до них прив’язані та доступ до функції аналізу оцінок учнів за допомогою розробленої підсистеми підтримки освітнього процесу, що дозволяє сформувати рекомендації щодо корегування навчального плану.

Стартова сторінка web-порталу «Startpage.aspx» при першому завантажені містить новинну інформацію. Ліве «Службове меню» web-сторінки «Startpage.aspx» містить пункти «Головна», що дозволяє завантажити web-сторінку «Startpage.aspx» та «Новини школи».

Окремо, для входу в систему, на сторінці знаходиться меню з полями «Логін» та «Пароль», радіо-кнопка з вибором статусу («Учень», «Батько», «Вчитель») і кнопка «Вхід». Логіка роботи web-сайту працює таким чином, що не авторизований користувач заходить на сайт, то йому автоматично присвоюється статус «User» з обмеженим доступом до ресурсів системи і йому доступна лише опція входу в систему ввівши логін і пароль у відповідні поля, вибравши статус і натиснувши кнопку «Вхід».

Зареєстрованим користувачам зі статусом «Admin» необхідно ввести логін та пароль у відповідні поля, вибрати статус «Вчитель» та натиснути

кнопку «Вхід», після чого здійснюється перевірка на стороні сервера введених даних і якщо вона завершується успішно, то завантажується стартова сторінка «Startpage.aspx».

Стартова сторінка «Startpage.aspx» для адміністраторів містить наступні пункти меню: «Головна»; «Оцінки»; «Відвідування занять»;«Домашні завдання»; «Розклад занять»; «Оцінки та завдання»; «Виставлення відвідувань»; «Розподіл предметів»; «Розподіл вчителів»; «Учні»; «Викладачі»; «Складання розкладу».

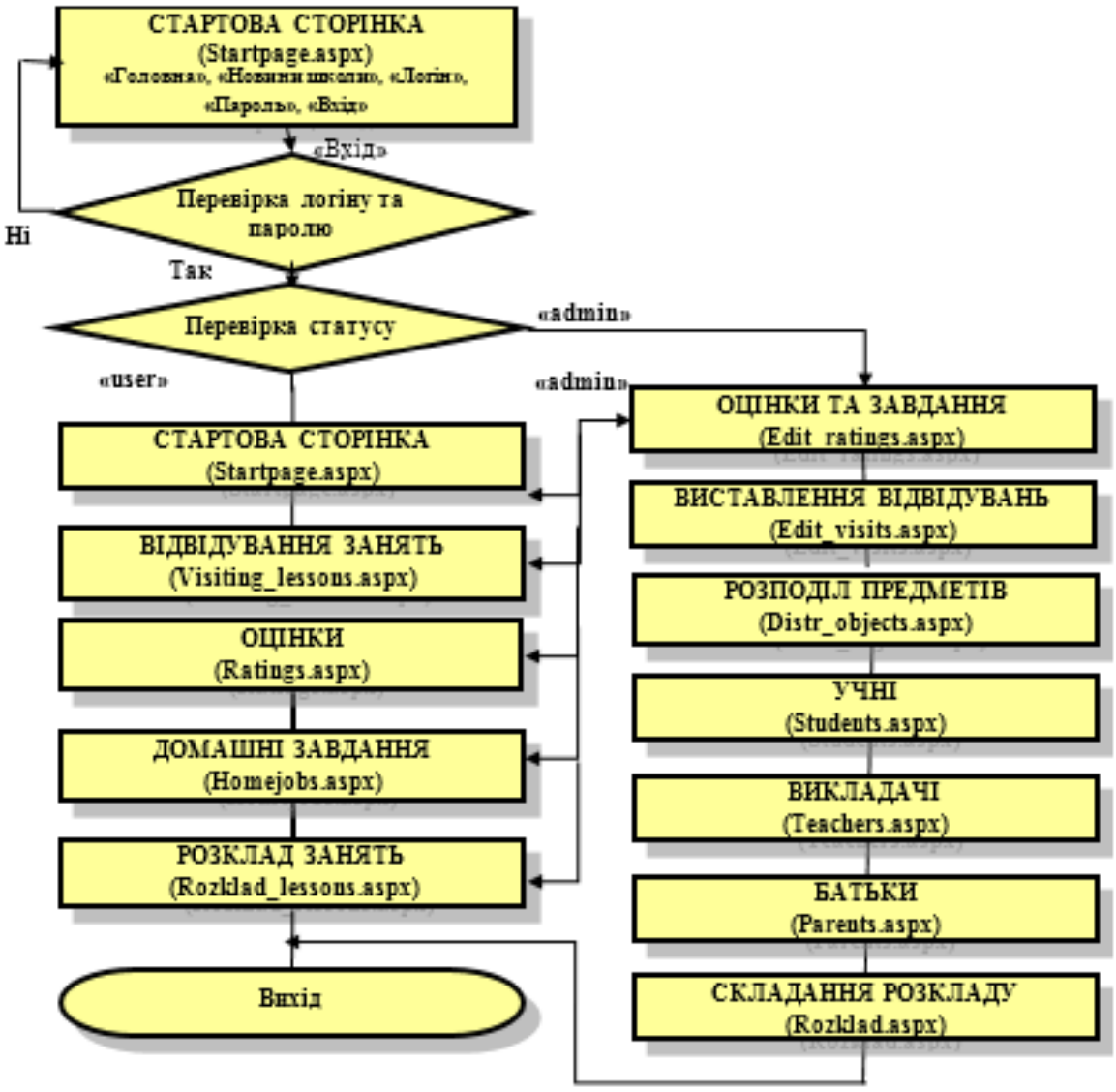


Рисунок 2.5 – Інформаційно логічна модель web-порталу

Зареєстрованим користувачам зі статусом «Учень» або «Батько» необхідно ввести логін та пароль у відповідні поля, вибрати відповідний статус

та натиснути кнопку «Вхід», після чого здійснюється перевірка на стороні сервера введених даних і якщо вона завершується успішно, то завантажується стартова сторінка «Startpage.aspx» з наступними пунктами меню: «Головна»; «Оцінки»; «Відвідування занять»; «Домашні завдання»; «Розклад занять».

Web-сторінка «Оцінки» (Ratings.aspx) містить інформацію щодо оцінок учнів за класами, для Адміністратора є доступними оцінки всіх учнів, для користувача «Учень» або «Батько» будуть доступні оцінки, що належать лише йому (учневі) або його дитині (батькові). Це буде здійснюватися за допомогою вибірки з бази даних за персональнім ідентифікатором користувача.

Web-сторінка «Відвідування» (Visiting_lessons.aspx) містить інформацію щодо відміток про відвідування занять учнями, для користувача із статусом «Вчитель» є доступними відмітки про присутність всіх учнів, для користувача «Учень» або «Батько» - тільки відмітки, що належать йому (його дитині), що здійснюється за допомогою вибірки з бази даних за персональним ідентифікатором користувача.

Web-сторінка «Домашні завдання» (Homejobs.aspx) містить інформацію щодо домашніх завдань відповідно до класу, предмету та дню тижня. Користувачу зі статусом «Вчитель» доступна інформація щодо домашніх завдань по всім класам, користувачам зі статусом «Учень» або «Батько» - інформація тільки щодо його класу (його дитини), що здійснюється за допомогою вибірки з бази даних за ідентифікатором класу, що прив’язаний до даного користувача.

Web-сторінка «Оцінки та завдання» (Edit_ratings.aspx) доступна користувачам зі статусом «Вчитель» та буде містити необхідну форму з полями для вводу домашніх завдань.

Web-сторінка «Розподіл предметів» (Distr_objects.aspx) доступна для завантаження тільки користувачам зі статусом «Вчитель» та містить необхідну форму для розподілу предметів за вчителями.

Web-сторінка «Розподіл вчителів» (Distr_teachers.aspx) доступна для завантаження тільки користувачам зі статусом «Вчитель» та містить необхідну форму для розподілу вчителів за класами.

Web-сторінка «Учні» (Students.aspx) доступна для завантаження тільки користувачам зі статусом «Вчитель» та містить необхідну форму з полями для занесення реєстраційної інформації про учнів.

Web-сторінка «Викладачі» (Teachers.aspx) доступна для завантаження тільки користувачам зі статусом «Вчитель» та містить необхідну форму з полями для занесення реєстраційної інформації про викладачів школи.

Web-сторінка «Батьки» (Parents.aspx) доступна для завантаження тільки користувачам зі статусом «Вчитель» та містить необхідну форму з полями для занесення реєстраційної інформації про батьків учнів.

Web-сторінка «Складання розкладу» (Rozklad.aspx) доступна для завантаження тільки користувачам зі статусом «Вчитель» та містить необхідну форму для складання розкладу занять.

2.6 Розробка SQL-запитів для серверних елементів управління

2.6.1 Розробка коду процедур та SQL-запитів для відображення поточної успішності

Функція відображення поточної успішності доступна авторизованим користувачам зі статусами «Учень», «Батько», «Вчитель». Для відображення поточної успішності за учнем, необхідно виконати вибірку полів прізвища, ім’я, по батькові, клас, дата, предмет, оцінка, вчитель, ідентифікатор учня за допомогою оператора мови SQL SELECT з таблиці «Journal». Оператор SELECT використовується сумісно з оператором INNER JOIN, що формує

таблицю з записів декількох зв’язаних таблиць. Програмний код запиту мовою SQL для відображення поточної успішності учня, поданий у лістингу 2.1.

Лістинг 2.1 – Реалізація коду процедур та SQL-запитів для відображення поточної успішності

```
SELECT Student.surname AS Прізвище, Student.name AS [Ім'я'],
Student.middlename AS [По батькові], class.name_class AS Клас,
Journal.date AS Дата, Subject.name AS Предмет, Journal.mark AS Оцінка,
Teacher.surname AS Вчитель, Student.id_student
FROM Journal
INNER JOIN Student ON Journal.studtnt_fk = Student.id_student
INNER JOIN Subject ON Journal.subject_fk = Subject.id_subject
INNER JOIN Distribution ON Subject.id_subject = Distribution.subject_fk
INNER JOIN Teacher ON Distribution.teacher_fk = Teacher.id_teacher INNER
JOIN class ON Student.class_fk = class.id_class
AND Distribution.class_fk = class.id_class
```

Для фільтрації по ID зареєстрованого учня використовується умова:

```
WHERE (Student.id_student = @id_student)
```

Для фильтрації за ID зареєстрованого учня та даті умова:

```
WHERE (Student.id_student = @id_student) AND (Journal.date = @date_a)
```

	Прізвище	ім'я'	По батькові	Клас	Дата	Предмет	Оцінка	Вчитель	id_student
▶	Іванов	Петро	Іванович	1А	03.09.2018	Малювання	11	Петров	2
	Іванов	Петро	Іванович	1А	03.09.2018	Письмо	12	Петров	2

Рисунок 2.6 – Перевірка SQL-запита на вибірку

Перевірка результатів роботи запиту на вибірку, наведеного в лістингу 2.1, подана на рис. 2.6.

2.6.2 Розробка коду процедур та SQL-запитів для відображення домашніх завдань

Функція відображення домашніх завдань доступна авторизованим користувачам зі статусами «Учень», «Батько», «Вчитель». Для відображення домашніх завдань для кожного з учнів, необхідно виконати вибірку полів: ідентифікатор учня (Student.id_student); клас (class.name_class); предмет (Subject.name); дата домашнього завдання (Jobhome.date_job); завдання

(Jobhome.jobhome); вчитель (Teacher.surname) за допомогою команди SELECT. Оператор SELECT використовується сумісно з оператором INNER JOIN, що дозволяє формувати таблицю з записів зв'язаних таблиць. Програмний код запиту мовою SQL для відображення домашнього завдання, поданий у лістингу 2.2.

Лістинг 2.2 – Реалізація коду процедур та SQL-запитів для відображення домашніх завдань

```
SELECT Student.id_student, class.name_class AS Клас, Subject.name AS
Предмет, Jobhome.date_job AS Дата, Jobhome.jobhome AS Завдання,
Teacher.surname AS Вчитель
FROM Jobhome INNER JOIN
Subject ON Jobhome.subject_fk = Subject.id_subject INNER JOIN
Teacher ON Jobhome.teacher_fk = Teacher.id_teacher INNER JOIN
Distribution ON Subject.id_subject = Distribution.subject_fk AND
Teacher.id_teacher = Distribution.teacher_fk INNER JOIN
class ON Jobhome.class_fk = class.id_class AND Distribution.class_fk =
class.id_class INNER JOIN
Student ON class.id_class = Student.class_fk
```

Для фільтрації по ID зареєстрованого учня використано оператор умови WHERE:

```
WHERE (Student.id_student = @id_student)
```

Для фільтрації по ID зареєстрованого учня та даті використано оператор умови WHERE:

```
WHERE (Student.id_student = @id_student) AND (Jobhome.date_job = @data_a)
```

Перевірка результатів роботи запиту на вибірку, наведеного в лістингу 2.2, представлений на рис. 2.7.

	id_student	Клас	Предмет	Дата	Завдання	Вчитель
▶	1	1А	Малювання	03.09.2018	Виконати завдання № ...	Петров
	1	1А	Письмо	03.09.2018	Виконати завдання № ...	Петров
	2	1А	Малювання	03.09.2018	Виконати завдання № ...	Петров
	2	1А	Письмо	03.09.2018	Виконати завдання № ...	Петров
	3	1Б	Малювання	03.09.2018	Виконати завдання № ...	Петров
	3	1Б	Письмо	03.09.2018	Виконати завдання № ...	Петров

1 of 32 Cell is Read Only.

Рисунок 2.7 – Перевірка SQL-запита на вибірку

2.6.3 Розробка коду процедур та SQL-запитів для відображення розкладу занять

Функція відображення розкладу занять доступна авторизованим користувачам зі статусами «Учень», «Батько», «Вчитель». Заняття повинні відображуватись для обраного найменування класу або по прізвищу та ініціалам учня. Для відображення розкладу занять для кожного з учнів, необхідно виконати вибірку полів: ідентифікатор розкладу (Timetable.id_timetable), день тижня (Day_of_week.day_week), клас (class.name_class), номер уроку (Lesson.lesson_number), завдання (Jobhome.jobhome), вчитель (Teacher.surname + ‘ ’ + Teacher.name + ‘ ’ + Teacher.middlename), час початку (lesson.lesson_time), назва предмету (Subject.name). Оператор SELECT використовується сумісно з оператором INNER JOIN, що формує таблицю даних з записів декількох зв’язаних таблиць. Лістинг програмного коду запиту мовою SQL для відображення розкладу по обраному класу приведений нижче.

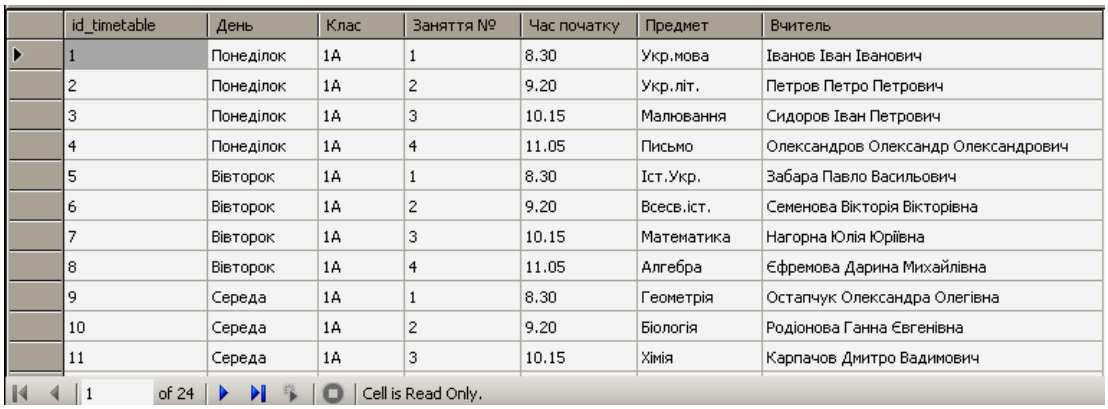
Лістинг 2.3 – Реалізація коду процедур та SQL-запитів для відображення розкладу по обраному класу

```
SELECT Timetable.id_timetable, Day_of_week.day_week AS День,
class.name_class AS Клас, Lesson.lesson_number AS [Заняття №],
Lesson.lesson_time AS [Час початку],
Subject.name AS Предмет, Teacher.surname + ' ' + Teacher.name + ' ' +
Teacher.middlename AS Вчитель
FROM Timetable INNER JOIN
class ON Timetable.class_fk = class.id_class INNER JOIN
Lesson ON Timetable.lesson_number_fk = Lesson.id_lesson INNER JOIN
Subject ON Timetable.subject_fk = Subject.id_subject INNER JOIN
Teacher ON Timetable.teacher_fk = Teacher.id_teacher INNER JOIN
Day_of_week ON Timetable.day_week_fk = Day_of_week.id_day
WHERE     (class.id_class = @id_class)
```

Якщо вибірка здійснюється за обраним викладачем, то умова буде мати вигляд:

```
WHERE (Teacher.id_teacher = @id_teacher)
```

Перевірка SQL-запита на вибірку розкладу за класом подана на рис. 2.8.

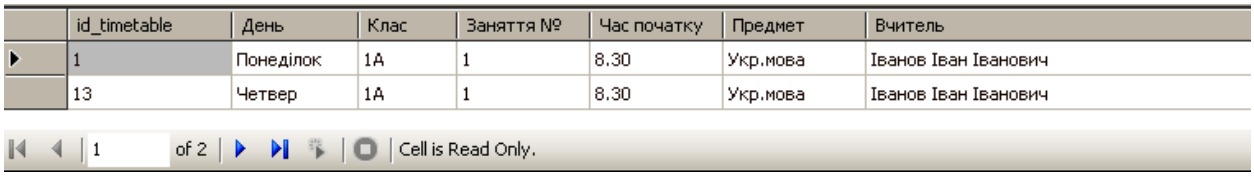


id_timetable	День	Клас	Заняття №	Час початку	Предмет	Вчитель
1	Понеділок	1А	1	8.30	Укр.мова	Іванов Іван Іванович
2	Понеділок	1А	2	9.20	Укр.літ.	Петров Петро Петрович
3	Понеділок	1А	3	10.15	Малювання	Сидоров Іван Петрович
4	Понеділок	1А	4	11.05	Письмо	Олександров Олександр Олександрович
5	Вівторок	1А	1	8.30	Іст.Укр.	Забара Павло Васильович
6	Вівторок	1А	2	9.20	Всесв.іст.	Семенова Вікторія Вікторівна
7	Вівторок	1А	3	10.15	Математика	Нагорна Юлія Юріївна
8	Вівторок	1А	4	11.05	Алгебра	Єфремова Дарина Михайлівна
9	Середа	1А	1	8.30	Геометрія	Остапчук Олександра Олегівна
10	Середа	1А	2	9.20	Біологія	Родіонова Ганна Євгенівна
11	Середа	1А	3	10.15	Хімія	Карпачов Дмитро Вадимович

1 of 24 | Cell is Read Only.

Рисунок 2.8 – Перевірка SQL-запита на вибірку розкладу за класом

Перевірка результатів роботи запиту на вибірку розкладу за вчителем, наведеного в лістингу 2.3, представлений на рис.2.9.



id_timetable	День	Клас	Заняття №	Час початку	Предмет	Вчитель
1	Понеділок	1А	1	8.30	Укр.мова	Іванов Іван Іванович
13	Четвер	1А	1	8.30	Укр.мова	Іванов Іван Іванович

1 of 2 | Cell is Read Only.

Рисунок 2.9 – Перевірка SQL-запита на вибірку розкладу за вчителем

2.6.4 Розробка коду процедур та SQL-запитів для внесення і редагування даних учнів

Для занесення та редагування даних учнів за допомогою елементів інтерфейсу необхідно створити запити на видалення, оновлення та занесення даних. Для видалення та оновлення записів таблиці необхідно, щоб запит був прив’язаний до ID учня.

Лістинг 2.4 – Реалізація коду процедур та SQL-запитів для внесення і редагування учнів

Для вибірки використовується оператор SELECT:

```
SELECT [id_student], [gender_fk], [status_fk], [class_fk], [name], [surname], [middlename], [date_b], [login], [password] FROM [Student]
```

Для оновлення даних щодо обраного учня, використовується оператор UPDATE.

```
UPDATE [Student] SET [gender_fk] = @gender_fk, [status_fk] = @status_fk, [class_fk] = @class_fk, [name] = @name, [surname] = @surname, [middlename] = @middlename, [date_b] = @date_b, [login] = @login, [password] = @password
WHERE [id_student] = @id_student
```

Для позначення параметричних змінних використовується знак – «@». Значення параметрів @gender_fk, @status_fk, @class_fk, @name, @surname, @middlename, @date_b, @login, @password, @id_student визначаються користувачем та вводяться з використанням інтерфейсу.

Для реалізації функції видалення запису учня, необхідно використати оператор DELETE, запит має вигляд:

```
DELETE FROM [Student] WHERE [id_student] = @id_student
```

2.6.5 Розробка коду процедур та SQL-запитів для внесення і редагування даних щодо вчителів

Для внесення і редагування даних про вчителів за допомогою елементів інтерфейсу необхідно створити запити на видалення, оновлення та занесення даних. Для видалення та оновлення записів таблиць необхідно, щоб запит був прив’язаний до ID вчителя. Для вибірки використовується оператор SELECT:

```
SELECT [id_teacher], [gender_fk], [status_fk], [surname], [name], [middlename], [login], [password] FROM [Teacher]
```

Для оновлення даних щодо обраного вчителя, використовується оператор UPDATE. Значення параметрів @gender_fk, @status_fk, @name, @surname,

@middlename, @login, @password, @id_teacher визначаються користувачем та вводяться з використанням інтерфейсу.

```
UPDATE [Teacher] SET [gender_fk] = @gender_fk, [status_fk] = @status_fk, [surname] = @surname, [name] = @name, [middlename] = @middlename, [login] = @login, [password] = @password WHERE [id_teacher] = @id_teacher
```

Для реалізації функції видалення запису щодо вчителя, необхідно використати оператор DELETE:

```
DELETE FROM [Teacher] WHERE [id_teacher] = @id_teacher
```

2.6.6 Розробка коду процедур та SQL-запитів для створення розкладу занять

Для створення розкладу використовується список «Клас» і таблиця Timetable (об'єкт інтерфейсу GridView), в якій всі поля, що відповідають зовнішнім ключам, реалізовані як списки.

Фильтрації даних таблиць відбуваються у відповідності з обраним значенням у списку «Клас». Для редагування і видалення даних таблиць розроблені наступні SQL-запити:

```
SELECT [id_timetable], [class_fk], [subject_fk], [teacher_fk], [day_week_fk], [lesson_number_fk], [date_z] FROM [Timetable] WHERE ([class_fk] = @class_fk)
```

```
UPDATE [Timetable] SET [class_fk] = @class_fk, [subject_fk] = @subject_fk, [teacher_fk] = @teacher_fk, [day_week_fk] = @day_week_fk, [lesson_number_fk] = @lesson_number_fk, [date_z] = @date_z WHERE [id_timetable] = @id_timetable
```

```
DELETE FROM [Timetable] WHERE [id_timetable] = @id_timetable
```

Параметри вставки даних: @id_timetable, @class_fk, @subject_fk, @teacher_fk, @day_week_fk, @lesson_number_fk, @date_z приймають значення, обрані в списках таблиці.

Так як інтерфейс таблиці GridView не дозволяє вставляти запис в таблицю Timetable, розроблена процедура. Код розробленої процедури:

```
protected void Button1_Click(object sender, EventArgs e)
{
string connectionString =
WebConfigurationManager.ConnectionStrings["ConnectionStringSchool"].Conne
ctionString;
SqlConnection con = new SqlConnection(connectionString);
string sql = "INSERT INTO Timetable (class_fk) ";
sql = sql + " VALUES (" + (string)DropDownList1.SelectedItem.Value + ")";
SqlCommand cmd = new SqlCommand(sql, con);
con.Open();
cmd.ExecuteNonQuery();
con.Close();
GridView1.DataBind();
}
```

Наведений вище код реалізує запит, в якому параметр @class_fk визначається обраним значенням елемента списку «Клас»:

```
INSERT INTO Timetable (class_fk) VALUES (@class_fk).
```

2.7 Тестування розробленого програмного забезпечення інформаційної системи

Для проведення тестування програмного забезпечення використовувався додаток «Microsoft Test Manager», що входить до складу інтегрованого середовища розробки починаючи з версії Microsoft Visual Studio 2010 (випуски Ultimate або Professional).

Для тестування розробленої інформаційної системи були створені

– модульний тест (Unit test) для тестування коду класів WebForm (їх властивостей і методів), створених шляхом спадкування від класу Page;

– тест продуктивності (функціональності) web-системи (Web Performance test);

– тест навантажувального тестування web-сторінок сайту (Load test)

Для тестування розробленої інформаційної системи використовувалися сторонні засоби:

– web-сервер «Visual Studio Development Server» (версія v.4.5), що входить до складу «Visual Web Developer 2012» з пакетом «Microsoft .NET Framework Version 4.5 SP1»;

– web-сервер «Microsoft Internet Information Services» v5.1, що входить до складу операційної системи «Microsoft Windows XP Professional SP3», із установленим пакетом «Microsoft .NET Framework 4 Extended»;

– web-сервер «Microsoft Internet Information Services Express v6.0 Resource Kit Tool», локалізований для операційної системи «Microsoft Windows 7, SP1», із установленими пакетами «Microsoft .NET Framework 4.0 Extended», «Microsoft .NET Framework 4.5»;

- СУБД SQL-сервера «Microsoft SQL Server 2008 R2 Express».

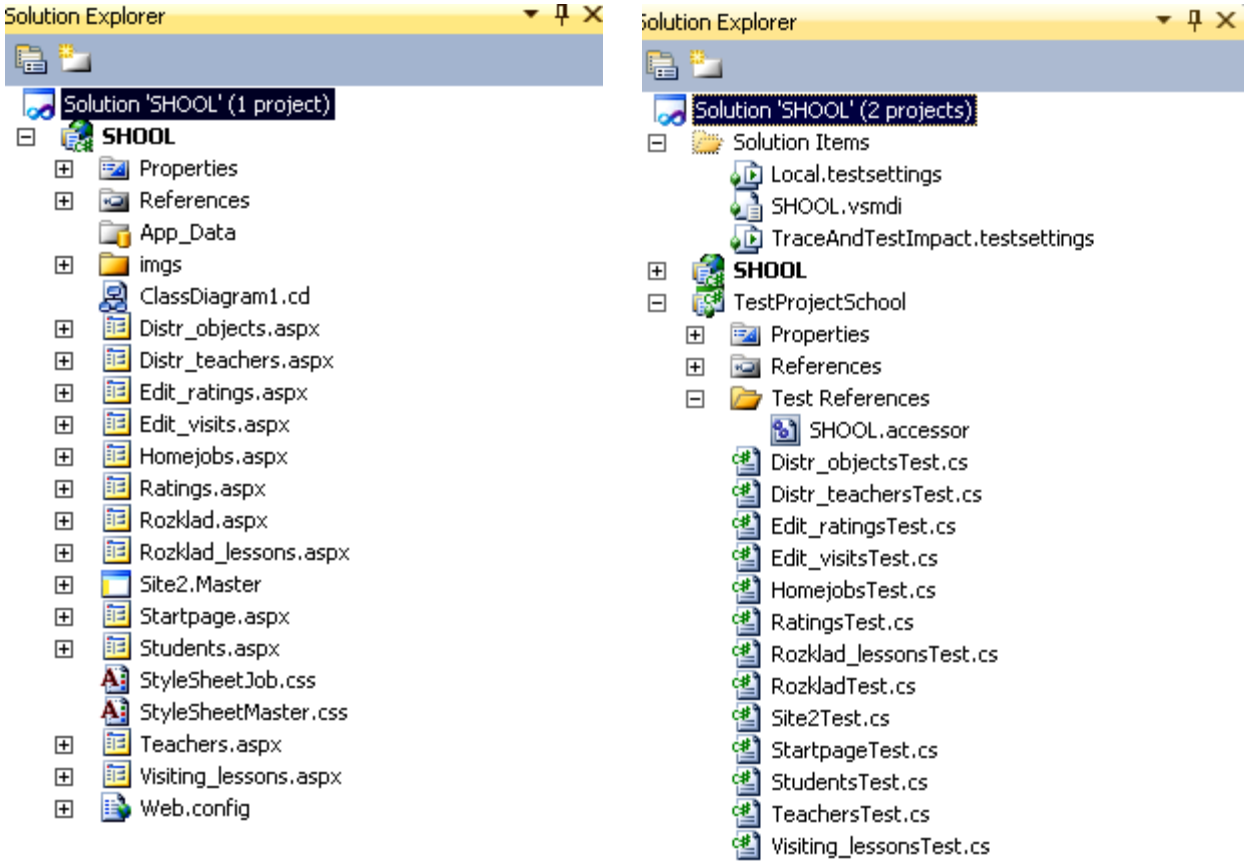
Для виконання тестування був створений план проведення тесту у вигляді впорядкованого тесту «Ordered test». Тестування проводилося в наступній послідовності.

На першому етапі проводилося модульне тестування. Так як розроблена web-система утримує 12 класів-спадкоємців класу «Page», яки містять інтерфейс доступу до бази даних, то при автоматизованому створенні модульних тестів усіх процедур (методів), що належать об'єктам web-сторінок, були створені класи «PageTest».

Для проведення модульного тестування у поточне рішення (solution) був доданий проект із ім'ям «TestProjectSchool». При автоматизованому створенні модульних тестів усіх процедур (методів) класів, були створені окреми класи до імени яких додалося ключове слово «Test». Фрагменти властивостей класів системи і властивостей класу модульного тестування «TestProjectSchool» представлені на рис. 3.11 а), б) відповідно.

При проведенні модульного тестування використовувався принцип «білого ящика», який ґрунтується на знанні внутрішньої структури класів. Модульне тестування проводилося на рівні всіх класів для всіх функцій інтерфейсу. Ціль модульного тестування полягала в незалежній перевірці окремих компонентів коду, пов'язаного з подіями використання інтерфейсу web-сторінок. Модульне тестування дозволило виявити локальні дефекти, такі як помилки, у реалізації алгоритму програми, некоректна реалізація інтерфейсів, невірні операції, невірні логічні і математичні вираження, цикли,

помилки у використанні локальних ресурсів для окремих локальних і глобальних змінних, полів об'єктів відкритих класів.



а) б)

Рисунок 2.11 – Фрагменти властивостей класів системи і властивостей класу модульного тестування

При виявленні помилок модульного тестування, проводилося регресивне тестування, що дозволяє виявляти збої в роботі реалізованих функцій web-сторінок, що викликані додаванням нових функцій. Тобто для конкретної помилки заново проводилися всі види перерахованих вище тестів для виявлення збоїв зміненого коду. Тест дав позитивні результати.

На другому етапі проводився тест функціональності (продуктивності). При проведенні тесту здійснювалися запити до сервера і аналізувалися його відповіді. Тестувалися операції відкриття сторінки по заданому URL, уведення коректних і некоректних логіну і пароля, перевірка роботи інтерфейсу web-сторінок для доступу до бази даних, перевірка правильності переходу до різних

web-сторінок інформаційної системи при використанні меню навігації. Усі виявлені помилки усунуті.

На третьому етапі проводилося навантажувальне тестування з використанням 150, 350 і 750 користувачів. У якості основного тесту при навантажувальнім тестуванні використовувався тест продуктивності з 2-ма стратегіями. Перша стратегія припускала використання тесту продуктивності для користувачів зі статусом «учень» та «батько», яким доступна тільки функція перегляду інформаційних web-сторінок системи. У другій стратегії імітувалися дії користувача зі статусом адміністратора. Тестувалися дії по авторизації, доступ до web-сторінок, призначених тільки для адміністратора, уведення і редагування даних. Результати проведеного навантажувального тесту дали позитивний результат.

Системне тестування проводиться над інформаційною web-системою в цілому за допомогою методу «чорного ящика». При тестуванні не приймалася в облік структура проекту, перевірялися тільки коди і інформація (вхідна, вихідна, відображувана), якої оперував користувач. У ході системного тестування були виявлені відсутні і некоректні функції, незручність використання інтерфейсу, непередбачувані дані і їх комбінації (у тому числі на зміну типів даних), непередбачувані або не підтримані сценарії роботи. Помилки усунуті. Тестування на платформі проводилося для операційних систем Windows XP і Windows 7. Здійснювалася перевірка локалізації і роботи програмного забезпечення web-сторінок системи на обраних цільових платформах. Результати позитивні.

Для оцінки якості розробленої інформаційної системи з web-інтерфейсом доступу до даних на всіх етапах тестування застосовувалися вимоги міжнародного стандарту ISO 9126, згідно з яким під «якістю» розумілася «ступінь відповідності розробленого програмного забезпечення вимогам замовника». Тестування розробленої інформаційної системи дало позитивний результат.

ВИСНОВКИ

У ході виконання атестаційної роботи було проведено аналіз предметної області, яка визначає діяльність існуючої системи управління навчальним процесом в навчальних закладах України I-II ступенів та реалізованої підсистеми підтримки освітньої діяльності, що робить пошук асоціативних правил стосовно успішності учнів та дозволяє вчителям отримувати рекомендації щодо корегування навчального процесу. Визначені недоліки та шляхи їх усунення. Розроблена підсистема допомагає робити логічні висновки щодо успішності учнів, обробляючи дані, що вже існують в базі даних школи, вирішаючи тим самим частину робот пов’язаних з організацією навчального процесу.

Головним завданням розробленої програмної підсистеми є автоматизація процесу інтелектуального аналізу даних. В роботі розроблені вимоги до підсистеми, внесені зміни до існуючої бази даних та інтерфейсу користувача. Реалізовані такі функції системи:

– пошук асоціативних правил;

– генерація рекомендацій щодо навчального процесу на основі виявлених закономірностей.

Отже, web-система, яка включає компоненти інформаційної системи управління діяльністю навчальних закладів України I–II ступенів, в тому числі розроблений компонент підсистеми, відповідає всім висунутим вимогам. Завдання роботи виконані, мета досягнута.

Розроблені компоненти інформаційної системи управління діяльністю навчальних закладів України готові для апаратного розгортання та експлуатації. Розроблені програмні компоненти відкрити для подальшого вдосконалення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Барсегян, А.А. Анализ данных и процессов [текст]: учебн. 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.

2. Ларичев, О.И. Выявление експертных знаний (процедуры и реализации) / О.И. Ларичев, А.И. Мечитов, Е.М. Мошкович, Е.М. Фурмес. – Наукаб 1989.-128 с. ISBN 5-02-006736-9;

3. Палкин Н.Б., Орешков В.И. Бизнес аналитика: от данных к знаниям: учебное пособие, 2-е изд., испр. – СПб. : Питер, 2013 – 704 с.: ил.;

4. Маклаков, С. В. BPwin и ERwin. CASE-средства разработки информационных систем [Текст] / С. В. Маклаков. – М.: Диалог, 1999. – 256 с.:

5. Дюк, В., Самойленко А. Data mining: учебный курс – СПб: Питер, 2001. – 368 с.: ил.

6. Кватрани, Т. Rational Rose 2000 и UML. Визуальное моделирование [Текст]/ Т. Кватрани. – М.: Пресс, 2001. – 176 с.: ил. – (Объектно-ориетированные технологии в программировании).;

7. Мюллер, Р. Базы данных и UML. Проектирование [Текст] / Р. Мюллер. – М.: Лори, 2002. – 423 с. – Библиогр. с. 417 – 420;

8. Трофимов, С. А. CASE-технологии: практическая работа в Rational Rose [Текст] / С. А. Трофимов. – 2-е изд. – М.: Бином-пресс, 2002. – 288 с.: ил. – Библиогр. с. 276;

9. Андреев, В. В. Требования к информационной системе управления учебным процессом в ВУЗе [Текст] / В. В. Андреев, Н. В. Герова // Программные продукты и системы. – 2010. – № 1. – С. 125 – 128.;

10. Бакулин, В. К. Единая информационная система управления учебным процессом [Электронный ресурс] / В. К. Бакулин. – Режим доступа: http://tandemservice.ru/web/guest/solutions/university. – Загол. с экрана.;

11. Буч, Г. Объектно-ориентированный анализ и проэктирование с примерами приложений [текст] / Г. Буч, Дж. Бобби, А. Роберт и др. – М.: Вильямс, 2008. – 720 с.;

12. Гайдамакин, Н. А. Автоматизированные информационные системы, базы и банки данных. Вводный курс [Текст]: учеб. пособие / Н. А. Гайдамакин. – М.: Гелиос АРВ, 2002. – 368 с.: ил. – Библиогр. с. 354-355;

13. Геркул, В.И. Проектирование информационных систем[текст]: Учебное пособие / В.И. Геркул. – М.: Интернет ун-т информ технологий, 2005. – 504 с.;

14. Долженкова М.Л. Использование CASE-средств для проектирования информационных систем: Учебное пособие / М.Л. Долженкова, О.В. Караваева - Киров: Изд-во ВятГУ, 2002.- 73 с.;

15. Диго С.М. Базы данных. Проектирование и создание: Учебно-методический комплекс. – М.: Изд. центр ЕАОИ. 2008. – 171 с.;

16. Емельянова, Н. З., Прытка, Т. Л., Попов, И. И. Основы проектирования автоматизированных информационных систем [текст]: Учебное пособие / Н.З. Емельянова, Т.Л. Прытка, И.И.Попов. – М.: ФОРУМ:ИНФРА-М, 2005. – 416 с.;

17. Золотухина, Е. Б. Пример описания предметной области с использованием UML при разработке программных систем [Электронный ресурс] / Е. Б. Золотухина, Л. Р. Алфимов. – Режим доступа: http://interface.ru/misc/uml1.htm. – Загол. с экрана.;

18. Котов, А. В. Автоматизированные системы управления ВУЗом: новые реалии в условиях рынка [Текст] / А. В. Котов // Информационные технологии. – 2009. – № 10. – С. 38 – 43.;

19. Кузнецов, С. Д. Концептуальное проектирование реляционных баз данных с использованием языка UML [Электронный ресурс] / С. Д. Кузнецов. – Режим доступа: http://www.citforum.ru/database/articles/ umlbases.shtml. – Загол. с экрана.;

20. Методичні рекомендації щодо розробки нових навчальних планів та програмно-методичної документації за кредитно-модульною системою [Текст] / Харк. держ. акад. культури; Уклад.: В. М. Шейко, М. М. Каністратенко, Н. М. Кушнаренко. – Х.: ХГАК, 2006. – 24 с.;

21. Райордан, Р. Основы реляционных баз данных [Текст] / Р. Райордан. – М.: Русская редакция, 2001. – 384 с.: ил.;

22. Румянцева, Е. Л. Информационные технологии [Текст]: учеб. пособие / Е. Л. Румянцева, В. В. Слюсарь. – М.: Инфра, 2007. – 256 с.: ил. – (Профессиональное образование);

23. Семакин, И. Г. Информационные системы и модели [Текст]: учеб. пособие / И. Г. Семакин, Е. К. Хеннер. – М.: Бином, 2005. – 303 с.: ил.;

24. Тоискин В.С., Красильников В.В., Малиатаки В.В. Автоматизация процессов проектирования на основе CASE технологий: Учебное пособие. – Ставрополь: Изд-во СГПИ, 2010. – 131 с.;

25. Трофимов, С. UML диаграммы в Rational Rose [Электронный ресурс] / С. Трофимов. – Режим доступа: http://www.caseclub.ru/articles/rose2.html?next=11. – Загол. с экрана;

26. Тяпкин, С. В. Преимущества и недостатки объектно-ориентированных баз данных [Электронный ресурс] / С. В. Тяпкин. – Режим доступа: http://www.jurnal.org/articles/2007/inf3.html. – Загол. с экрана;

27. Яблонский, С. А. Автоматизированная информационная управляющая система университета [Текст] / С. А. Яблонский, Петербургский государственный университет путей сообщения // Программные продукты и системы. – 2009. – № 4. – С. 87 – 95;

28. Токмаков,Г. П. Базы данных и знаний. Проектирование баз данных по технологии «клиент-сервер» и разработка клиентских приложений: Учебное пособие / Г.П. Токмаков.- Ульяновск; УлГТУ, 2005. - 143 с.;

29. Файзрахманов Р. А., Селезнев К. А. Учебное пособие к практическим занятиям «Структурно функциональный подход к

проектированию информационных технологий и автоматизированных систем с использованием CASE-средств» / Перм. гос. техн. ун-т. –Пермь, 2005. – 245 с.;

30. Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. CASE-технологии: Практикум. - М.: Горячая линия-Телеком, 2005.-160 с: ил.