Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження інтелектуальних методів синтезу розподілених баз даних
(тема)

Виконав:
студент 2 курсу, групи СШІм-20-3
Яковлєв Д.А.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

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

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник проф. ФілатовВ.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

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

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«» _____2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Яковлєву Денису Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження інтелектуальних методів синтезу розподілених баз даних

затверджена наказом університету від 24березня 2022 р. № 414 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 12травня2022 р.

3. Вихідні дані до роботи Теоретичні відомості з тематики кваліфікаційної роботи, науково-технічні публікації та дані Інтернет-джерел

4. Перелік питань, що потрібно опрацювати в роботі

Перспективи розвитку баз даних, їх роль та місце у розподілених інформаційних системах; методологічна основа інтеграції розподілених баз даних; розробка методу синтезу схеми реляційної моделі даних; особливості інтелектуального аналізу даних при схемній інтеграції

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)

Рисунок 1 – Методологія інтеграції розподіленої інформації

Рисунок 2 – Архітектура мультибази даних

Рисунок 3-9 – Об'єднання відношень варіант 1-7

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування Розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	д.т.н., проф.ФілатовВ.О.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Отримання завдання на кваліфікаційну роботу	28.03.2022	виконано
2.	Аналіз джерел інформації предметної галузі	29.03.2022–30.03.2022	виконано
3.	Постановка задачі	31.03.2022–05.04.2021	виконано
4.	Огляд перспектив розвитку баз даних, їх роль і	6.04.2022–10.04.2022	виконано
	місце у розподілених інформаційних системах		
5.	Розгляд методологічної основи інтеграції	11.04.2022–15.04.2022	виконано
	розподілених баз даних		
6.	Розробка методу синтезу схеми реляційноїмоделі	16.04.2022–20.04.2022	виконано
	Даних		
7.	Оформлення пояснювальної записки	21.04.2022–29.04.2022	виконано
8.	Оформлення графічного матеріалу	30.04.2022–02.05.2022	виконано
9.	Захист перед ЕК	12.05.2022	виконано

Дата видачі завдання 28березня 20 22 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 80с., 31 рис., 2 табл., 1 дод., 26 джерел.

БАЗА ДАНИХ, ІНФОРМАЦІЙНА СИСТЕМА, РЕЛЯЦІЙНА МОДЕЛЬ, СИНТЕЗ РОЗПОДІЛЕНИХ БАЗ ДАНИХ, СИНТЕЗ СХЕМИ РЕЛЯЦІЙНОЇ МОДЕЛІ ДАНИХ, СИСТЕМИ УПРАВЛІННЯ БАЗАМИ ДАНИХ.

Об'єктом дослідження є розподілені бази даних реляційних систем.

Метою даної роботи є розробка методу синтезу реляційної моделі даних, отриманої на основі інтеграції моделей даних різних незалежних предметних галузей.

Методами дослідження є системний аналіз, теорія реляційних баз даних, теорія побудови інформаційних систем, інтелектуальний аналіз даних.

ABSTRACT

Explanatory note:80 p., 31 fig., 2 tabl., 1 ann., 26 sources.

DATABASE, DATABASE MANAGEMENT SYSTEMS, INFORMATION SYSTEMS, RELATIONAL MODEL, SYNTHESIS OF DISTRIBUTED DATABASES, SYNTHESIS OF CIRCUITS RELATIONAL DATA MODEL.

The aim of this work is to develop a method for synthesizing a relational data model, obtained on the basis of the integration of data models of different independent subject areas.

Research methods are systems analysis, relational database theory, information systems theory, data mining.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

1НФ – перша нормальна форма;

2НФ – друга нормальна форма;

3НФ – третя нормальна форма;

БД – база даних;

ІРБД–інтегрована розподілена база даних;

ІС – інформаційна система;

ММД – мова маніпулювання даними;

РБД–розподілена база даних;

РМД – реляційна модель даних;

СС – семантична мережа;

СУБД – системи управління базами даних.

ВСТУП

Інтеграція розподілених баз даних є ключовою проблемою, вирішення якої залежить від рівня розвитку сучасних методів і засобів комп'ютеризованих інформаційних технологій. Як категорія, інтегрована розподілена база даних визначає клас складних та неоднозначних інформаційних об'єктів, механізми побудови та управління якими на сьогоднішній день є найбільш актуальними.

Багатоаспектність інтегрованих баз даних породжує необхідність компромісного використання цілого ряду методів та засобів комп'ютерних технологій для реалізації концепції інтегрованих розподілених баз даних.

Основна ідея інтеграції баз даних полягає у введенні предвідношення даних, в яке має бути відображена кожна з баз даних, що інтегруються. Цьому рівню уявлення відповідає цілком певна модель даних, яку ефективно могли б бути перетворені моделі даних довільних СУБД. Така модель даних називається концептуальною моделлю інтегрованої системи. Концептуальні моделі даних, підтримувані реальними СУБД, стосовно цієї загальної моделі, виступають як внутрішні моделі.

Слід зазначити, що з концептуальної моделлю даних інтегрованої системи можуть надбудовуватися моделі даних зовнішнього рівня. Тут досить думати, що включають мови програмування і прикладні програми спираються безпосередньо на концептуальний рівень системи інтеграції.

1 ПЕРСПЕКТИВИ РОЗВИТКУ БАЗ ДАНИХ, ЇХ РОЛЬ ТА МІСЦЕ У РОЗПОДІЛЕНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

1.1Мета роботи

Метою кваліфікаційної роботи є розробка та дослідження методів інтеграції розподілених баз даних інформаційних систем на основі синтезу моделей даних різних незалежних предметних областей. За допомогою розробленого методу на формальному рівні можна буде описати процес синтезу структурної складової реляційної моделі бази даних з безлічі схем баз даних, що окремо існують. Метод дозволить мінімізувати витрати на проектування загальної схеми БД, забезпечить зберігання даних та маніпулювання ними для всіх функціональних завдань урамках інтегрованої ІС. Використання інтегрованої ІС знизить ризики надання неактуальної, надмірної інформації, а також вартість її підтримки та супроводу.

1.2 Актуальність роботи

Існує кілька підходів до поняття «неоднорідні бази даних». Найбільшого інтересу заслуговують два з них, які інтенсивно обговорюються в даний час. З одного боку, бази даних, реалізовані засобами різних систем управління базами даних (СУБД), є неоднорідними за відповідними моделями даних. З іншого боку, бази даних, підтримувані однієї й тієї ж СУБД, але які визначаються різними концептуальними схемами, є інформаційно неоднорідними. Надалі поняття неоднорідних баз даних включає обидва види неоднорідності.

Основні цілі створення систем інтеграції неоднорідних баз даних можна сформулювати наступним чином: під інтеграцією баз даних розуміється досягнення можливості одночасного та спільного використання прикладної програми кількох баз даних як єдиного цілого.

Інтегрована сукупність різних баз даних стосовно прикладної програми має логічно виглядати як єдина база даних.

Ідея інтеграції баз даних полягає у запровадженні уявного предвідношення даних (віртуальної бази даних), у якому має бути відображена кожна з інтегрованих баз даних. Цьому рівню уявлення відповідає цілком певна модель даних, яку ефективно могли б бути перетворені моделі даних довільних СУБД. Така модель даних називається концептуальною моделлю інтегрованої системи. Концептуальні моделі даних, підтримувані реальними СУБД, стосовно цієї загальної моделі, виступають як внутрішні моделі.

Істотно, що для забезпечення умов інтеграції довільних баз даних концептуальна модель даних системи інтеграції повинна включати засоби, що дозволяють працювати одночасно як із структурованими, так і з неструктурованими даними. Слід зазначити, що з концептуальної моделлю даних інтегрованої системи можуть надбудовуватися моделі даних зовнішнього рівня. Тут досить думати, що включають мови програмування і прикладні програми спираються безпосередньо на концептуальний рівень системи інтеграції [1].

Поряд із здатністю до інтеграції неоднорідних баз даних іншим, не менш важливим аспектом архітектури (рисунок 1.1) є можливість досягнення високого ступеня незалежності та мобільності прикладних програм від типу СУБД:

Таким чином, моделі даних відіграють значну роль у системах управління базами даних:

- є ключовими компонентами архітектури СУБД;
- служать основою розробки сімейств мов високого рівня взаємодії з базами даних (мов програмування, запитних мов, мов діалогу);
- є основою розробки загальної методології проектування баз даних;
- є засобом забезпечення еволюції бази даних.

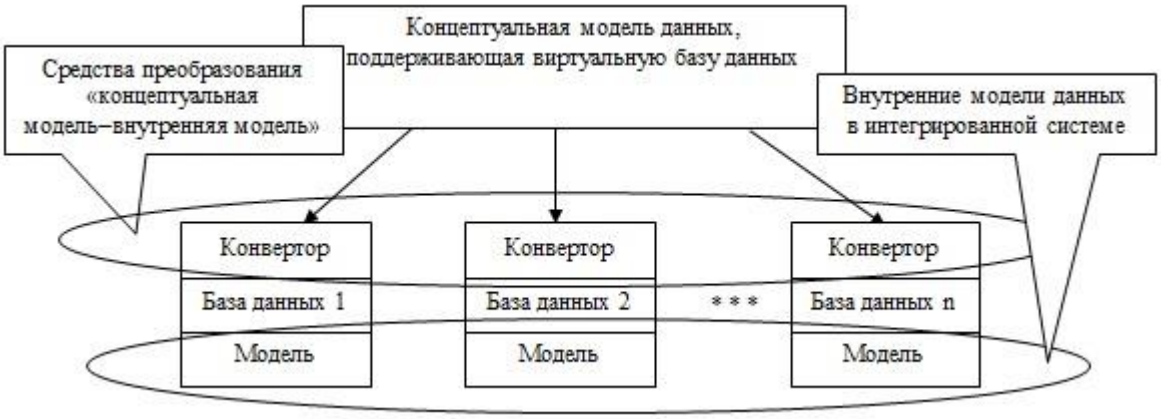


Рисунок 1.1 – Структура інтегрованої бази даних

Об'єктивні обставини, такі як: різні способи формального опису об'єктів у математиці; різноманітність структур даних та засобів маніпулювання даними, розвинених у мовах програмування; різноманітність предметних областей, що відображаються в базах даних, сприяли необмеженому зростанню кількості моделей даних та підтримують їх СУБД.

Однією з причин такого стану слід вважати відсутність методів формального опису та аналізу моделей даних, методів оперування моделями даних як самостійними об'єктами. З іншого боку, розвиток архітектур СУБД – таких як трирівнева архітектура чи архітектура систем інтеграції неоднорідних баз даних – свідчить у тому, що створення таких методів, передусім методу перетворення однієї моделі даних на іншу, є насущним завданням. Саме перетворювачі моделей даних становлять основу розглянутих архітектур.

При цьому перспективним вважається такий підхід:

− перетворення вихідної моделі даних у цільову виявляється у перетворенні схем та станів конкретних баз даних у схеми та стани баз даних у цільовій моделі даних;

− перетворення операторів мови маніпулювання даними (ЯМД) цільової моделі на послідовність операторів вихідної моделі даних.

Існуючі технології інтеграції баз даних засновані на використанні реляційної моделі даних (ODBC, BDE та ін.), Що забезпечує досягнення схемної однорідності за визначенням. Застосування цих технологій дозволяє говорити про можливість формування професійного рівня схемної інтеграції розподілених баз даних. На схемному інтеграційному рівні вирішуються завдання табличного предвідношення даних незалежно від специфіки визначення локальних баз даних у середовищах СУБД, наприклад, таких як Paradox, MySQL, Oracle, Access та ін.

На цьому рівні інтеграції принципово важливим є існування уявлень усіх розподілених даних у вигляді реляційних схем, що гарантує досягнення схемної модельної однорідності та вирішення цільової інтеграційної проблеми.

Іншим важливим аспектом побудови інтегрованих розподілених баз даних є та обставина, що серед нових інформаційних технологій бази даних повинні мати інфологічно і концептуально єдину основу. Схемна однорідність реляційних баз даних не торкається проблем забезпечення єдності смислового змісту розподілених даних. Будь-яка сучасна інформаційна технологія оперує системами локальних баз даних як технологічними інструментами, у своїй першому плані виходить проблема досягнення семантичної однорідності всіх інформаційних ресурсів.

Традиційно важливим аспектом побудови інтегрованих розподілених баз даних є визначення професійного інтерфейсу лише на рівні використання мережевих технологій доступу до розподілених баз даних. Очевидно, що інтеграція баз даних, що перетинаються, на користь декількох інформаційних технологій визначає необхідність розгляду методів і засобів управління та доставки однорідних інформаційних ресурсів на основі суперпозиції механізмів маршрутизації та комунікації інформаційних потоків [2].

Розподілені інформаційні системи стають структурою, що об'єднує засоби телекомунікації та обчислювальної техніки в єдину систему обміну, зберігання, обробки та управління інформацією.

2 МЕТОДОЛОГІЧНА ОСНОВА ІНТЕГРАЦІЇ РОЗПОДІЛЕНИХ БАЗ ДАНИХ

2.1 Основні поняття та визначення формованої інтеграційної методології

Основні поняття та визначення формованої інтеграційної методології представлені на рисунку2.1.

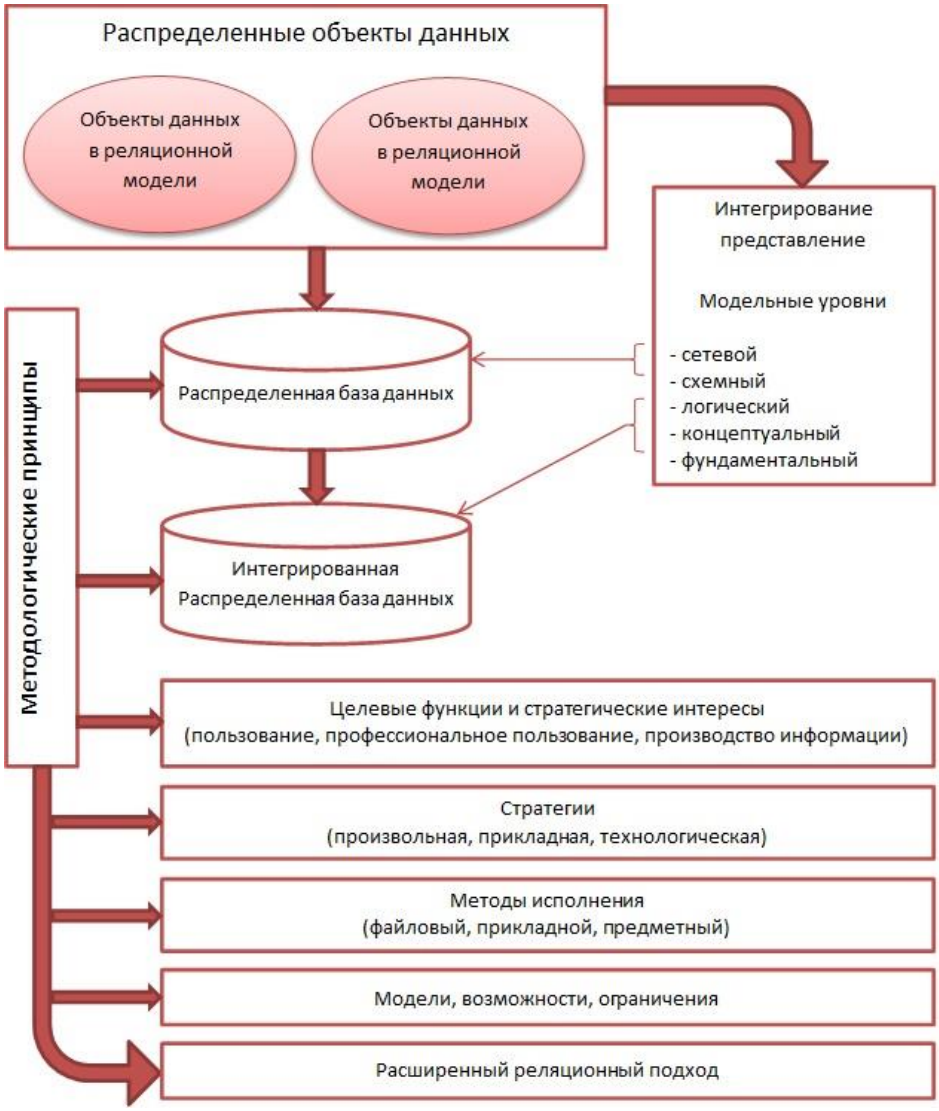


Рисунок 2.1 – Методологія інтеграції розподіленої інформації

Сформована таким чином архітектура інтеграційної методології дозволяє

сформулювати та визначити ключові поняття та принципи побудови об'єктів комп'ютерних технологій класу ІРБД [3].

2.2 Особливості інтеграції регіонально-розподілених інформаційних систем та баз даних

Сучасний рівень розвитку телекомунікаційних засобів з одного боку, та програмно-технічних з іншого, дозволяють нині впритул підійти до вирішення проблеми інтеграції територіально розподілених ІС та розподілених баз даних.

Перспектива впровадження розподілених інформаційних систем, що забезпечують прозорий доступ до всіх даних в організації або компанії, є виключно важливою, оскільки дозволяє користувачеві отримати якісно функціональні можливості. Нині існує низка підходів, які задовольняють інформаційним потребам бізнесу та суспільства, і в основі цих підходів є технології, пов'язані з базами даних – основою побудови сучасних регіонально-розподілених інформаційних систем.

Дослідження цього питання дозволило надати класифікацію систем, заснованих на розподілених базах даних (таблиця 2.1).

Таблиця 2.1 - Класифікація розподілених інформаційних систем

Тип інформаційної системи	Тип структур даних	Схема даних	Тип інтерфейсу	Рівень зв'язності
1	2	3	4	5
Розподілені бази даних	Однорідні	Глобальна схема	Внутрішні функції СУБД для забезпечення інтерфейсу між глобальним та локальним рівнями	Сильно пов'язаний

Продовження таблиці 2.1

Тип інформаційної системи	Тип структур даних	Схема даних	Тип інтерфейсу	Рівень зв'язності
1	2	3	4	5
Неоднорідні системи мультибаз даних із загальною мовою доступу	Неоднорідні	Функції мови доступу	Інтерфейс користувача СУБД для відображення між глобальним і локальним рівнями	Незв'язані
Однорідні системи мультибаз даних із загальною мовою доступу	Однорідні	Функції мови доступу	Інтерфейс користувача СУБД та внутрішні функції СУБД для відображення між глобальним та локальним рівнями	Слабо пов'язані
Інтероперабельні системи	Множина типів джерел даних	Відсутність глобальної інтеграції	Реалізація інтерфейсу між глобальним та локальним рівнями засобами додатків	Слабо пов'язані

У верхній частині таблиці 2.1 представлені системи, що відповідають максимальній інтеграції та взаємозв'язку компонентів. Однорідні розподілені бази даних функціонують під управлінням загальної глобальної схеми, яка, у свою чергу, відображається в схеми підтримують локальних баз даних. Для відображення інтерфейсів використовуються внутрішні функції самої СУБД. Ця модель отримала визнання в останні роки, і буде надалі користуватися популярністю головним чином у галузі паралельних систем баз даних.

Такого класу системи, як і раніше, застосовується і в середовищах LAN/WAN, однак такі архітектури навряд чи знайдуть широке поширення в корпоративних середовищах в основному через їхню нездатність справлятися з проблемою неоднорідності, коли необхідно мати справу з декількома різнорідними продуктами СУБД або моделями даних.

При аналізі наступного типу систем мультибаз із глобальною схемою даних, порівняно з розподіленими системами, додається неоднорідність, але, як і раніше, використовується глобальна схема. Звідси назва систем цього класу мультибази даних з глобальною схемою. Як і у випадку з однорідними розподіленими базами даних, клієнтські додатки оперують із глобальною схемою, ніби вона представляла одну велику централізовану базу даних. Усі відображення підтримуючі бази даних та їх вміст обробляються засобами глобального рівня.

На відміну від однорідних розподілених баз даних, мультибази даних з глобальною схемою не мають внутрішні функції СУБД, що дозволяють підтримувати відображення та інтерфейс між глобальним і локальним рівнями. Це зв'язки України з тим, що неоднорідність виключає можливість реалізації внутрішнього відображення, інтерфейси СУБД є зовнішніми щодо неї і мають створюватися і керуватися за необхідності. Відбувається це частково через те, що вже існуючі локальні СУБД і бази даних включаються в середовище, кероване глобальною схемою, без будь-яких змін на локальному рівні.

Підхід на основі глобальної схеми відносно простий у концептуальному відношенні: повний набір глобальних об'єктів відображається в фізичні реалізації, що підтримують їх, але на практиці він пов'язаний з низкою серйозних проблем. Зокрема створення всеосяжної глобальної схеми саме по собі – дуже нетривіальне завдання.

Складність розв'язання такого завдання пояснюється тим, що, по-перше, глобальна схема визначає повний універсум елементів даних середовища та, по-друге, всі зміни у складових базах даних (переміщення даних, послідовне тиражування) мають поширюватися і на глобальну схему, щоб забезпечити правильне відображення між глобальним та локальним рівнями. При детальному розгляді ця проблема виявляється значно складнішою, ніж видається на перший погляд. Концептуально класична архітектура систем мультибаз даних виглядає відносно просто, причому за відображення

глобального рівня локальний відповідають послуги довідників.

Розглянемо варіант, коли клієнтські програми самі розподілені на безлічі вузлів інформаційного простору. Це означає, що кожен додаток для здійснення будь-якої операції над локальною базою даних повинен мати доступ до глобальної схеми, і це має бути передбачено в архітектурі глобальної схеми. Один із можливих підходів до вирішення проблеми полягає в тому, що глобальна схема є централізованою та підтримується на одному вузлі. При цьому всі запити клієнтських додатків з будь-якого вузла повинні спочатку надходити на один з «вузлів довідника», для того щоб отримати доступ до глобальної схеми.

Недоліки такого підходу очевидні: основна причина для створення середовищ управління розподіленою інформацією – це виключення вузьких місць, а направлення запитів від усіх додатків розподіленого середовища на невелику кількість «розрахункових центрів», швидше за все, призведе до перевантаження в мережевому трафіку, особливо в організаціях з інтенсивною обробкою транзакцій.

Зазначимо два важливі фактори. По-перше, значна частка обробки, що запитується додатком, швидше за все, ставитиметься до локальних даних. По-друге, слідуючи цій логіці, слід визнати, що використання глобальної схеми для обробки локальних запитів – не найраціональніший підхід.

Для обробки таких запитів розумніше звертатися до локальної схеми «своєї» бази даних. Враховуючи цю парадигму доступу до даних, можна зробити висновок про те, що обов'язкове використання глобальної схеми для доступу до локальних даних створює не лише додаткові накладні витрати на відображення даних, але й надмірний мережевий трафік, якщо клієнтська програма виконується на вузлі, де відсутній екземпляр глобальної схеми.

Існує альтернативний підхід, при якому глобальна схема розподілена по всіх вузлах корпоративної мережі, де можуть виконуватися клієнтські додатки, щоб виключити необхідність віддаленого доступу додатків до сервісів

відображення та довідника, що надаються схемою. Негативна сторона цього підходу полягає в тому, що тепер усі зміни в глобальній схемі доведеться поширювати по багатьох вузлах мережі, що також не можна вважати очевидним завданням.

Залежно від частоти змін розподілу ресурсів інформаційному середовищу, яке може змінюватися від порівняно низького, з відносно рідкісними змінами в глобальній схемі, до дуже високої, коли схема змінюється дуже часто, підхід, що розглядається, може породжувати серйозні проблеми, пов'язані зі зниженням пропускної здатності мережі, а також із підтриманням цілісності даних. У цьому проблема підтримки цілісності виникає у ситуаціях, коли зміни, викликані перерозподілом даних чи коригуванням схеми, не встигають своєчасно поширитися однією чи більше вузлів.

Можна уявити таке середовище, де клієнтські додатки самі мають у своєму розпорядженні кошти, що дозволяють визначати, чи потрібен їм у тій чи іншій ситуації доступ до глобальної схеми. Інакше кажучи, підхід, заснований на глобальній схемі, можна було б використовувати для доступу клієнтських програм до віддалених даних. Але цей підхід призводить до зайвих накладних витрат і потенційного зниження продуктивності у разі доступу до локальних даних.

Середовище, що підтримує різні типи доступу у різний спосіб, дозволяє виключити зазначені недоліки. Таким чином, ми підходимо до ще одного класу систем мультибаз даних – регіональних баз даних.

Регіональна база даних, на відміну від мультибази даних з глобальною схемою, не має повної глобальної схеми, до якої обов'язково звертаються всі додатки. Натомість підтримується локальна схема типу імпорту/експорту. У регіональних базах даних застосовуються часткові глобальні схеми. На кожному вузлі підтримується часткова глобальна схема, що описує лише інформацію з віддалених джерел, яка необхідна виконання бізнес-функцій цьому вузлі.

Зазначимо, деякі складності, властиві мультибазам даних із глобальної схемою, у своїй однаково зберігаються. Так, як і раніше, необхідно поширювати зміни, що відбуваються в глобальній схемі, на відповідні вузли (необов'язково завжди в масштабі всієї інформаційної системи, можливо, тільки в межах деякої її частини). До того ж виникає додаткова проблема, тому що доводиться вирішувати, які саме дані, що є в системі, потрібні для додатків на кожному конкретному вузлі, і ця інформація повинна певним чином моделюватись і відповідно подаватись у схемі імпорту/експорту вузла.

Таким чином, у регіональній базі даних доводиться відмовлятися від підходу, при якому «все представлено у глобальній схемі, звернися туди і візьми це, коли тобі потрібно». В цьому випадку отримуємо деяку проміжну модель між управлінням розподіленою інформацією з архітектурою «клієнт-сервер» і мультибазами даних з глобальною схемою. Зрозуміло, у цій ситуації якщо на якомусь вузлі виникає потреба в доступі ще до якихось віддалених даних, або якщо метадані певного віддаленого вузла змінилися, то доводиться вносити зміни і в локальну схему імпорту/експорту.

Переваги цього підходу в наступному: на етапі проектування користувач визначає з якою інформацією він працюватиме, і відпаде необхідність здійснювати пошук у всьому масиві даних. У дуже великих розподілених середовищах підтримка загальної схеми може виявитися некерованим процесом. У такому разі регіональний підхід, можливо, буде легшим як для моделювання, так і для реалізації.

Наступний рівень, що відповідає ще більш пов'язаної організації середовища, ніж для регіональних баз даних, – це мультибази даних із загальною мовою доступу. Фактично, вони є розподілені середовища управління інформацією з технологією «клієнт-сервер».

Порівняємо два підходи до організації управління розподіленою інформацією: мультибази даних із загальною мовою доступу та мультибази даних із глобальною схемою. Тут зазначимо лише, що серед мультибази даних

із загальною мовою доступу – неоднорідної чи однорідної – глобальна схема взагалі відсутня, а запитів даних із віддалених джерел використовуються «функції мови доступу» – деякий інтерфейс прикладного програмного забезпечення. Як і в регіональних базах даних із частковою схемою, тут також необхідно заздалегідь визначити, які об'єкти даних із віддалених джерел потрібні клієнтським додаткам кожного вузла, та забезпечити ці можливості [4].

Нарешті, найбільшслабкопов'язанимиєінтероперабельныесистеми, уякихсамідодатки, виконуванісередтієїчиіншоїСУБД, відповідальнізаінтерфейсиміжрізнимисередовищамиуправлінняданими, незалежновідцього, єоднороднимичинеоднорідними. Інтероперабельні системи спрямовані, головним чином, на обмін даними, а чи не на обробку, типову баз даних.

Інтенсивно розвивалися останніми роками технології, засновані на стандарті CORBA, спрямовані на забезпечення інтеграції розподілених об'єктних систем. Розширення функціональних можливостей такого підходу в галузі управління даними (засобами ООСУБД та стандартних об'єктних сервісів OMG), зростання ринку комерційних програмних продуктів, що відповідають цим стандартам, а також поява нових і нових програмних продуктів підтверджують перспективність цього напряму.

На рисунку2.2 представлена концептуальна архітектура мультибаз даних, де рівень глобальної схеми відображається в середу СУБД та локальні схеми окремих підтримуючих інформаційних вузлів:

Рисунок2.2 – Архітектура мультибази даних

Крім складності та потенційних недоліків, які відзначалися вище, слід звернути увагу на деякі моменти, пов'язані з розглянутим вище підходом.

Глобальна схема теоретично може бути представлена в рамках будь-якої інформаційної моделі, що відповідає обраному середовищу. У більшості (в 11) з 16 прототипів та дослідницьких систем, перерахованих у роботі М. Брайта та ін., для предвідношення глобальної схеми використовувалася реляційна модель. В інших випадках для цієї мети використовувалися концептуальні моделі типу «сутність-зв'язок» або деякий «абстрактний концептуальний» підхід.

Донедавна застосування реляційної моделі для предвідношення глобальної схеми було виправдано, оскільки і підтримуючі бази даних були переважно реляційними. Навіть якщо в таке середовище були інтегровані нереляційні системи (наприклад, IMS або система типу CODASYL), то все одно можна було б використовувати відображення між глобальним реляційним рівнем і підтримуючими ієрархічною або мережевою моделлю таким чином, щоб програми могли програмуватися, начебто вони взаємодіяли виключно з реляційними базами даних.

Поява об'єктно-орієнтованих баз даних, які мають багатшими засобами предвідношення семантичної інформації, ставить під сумнів застосування реляційної моделі для предвідношення глобальних схем. В останні роки об'єктно-орієнтовані СУБД і бази даних все частіше використовуються як

компоненти мультибаз даних з глобальною схемою, для підтримки на концептуальному рівні значно більшої семантичної інформації, ніж це можливо в рамках базової реляційної моделі.

У цьому контексті залишається питання: чи буде використана як така семантично багата глобальна модель деяка розширена реляційна модель, заснована на добре опрацьованих методах концептуального моделювання, таких як «сутність-зв'язок», або якась інша сучасна об'єктно-орієнтована модель даних.

Мережевий рівень інтеграції розподілених баз даних пов'язані з вирішенням завдань управління та доставки схемно- і семантично однорідних інформаційних ресурсів користувачам та виробникам інформаційної продукції серед мереж інтегрального обслуговування. Очевидно, що інтеграція перетинаються або однієї і тієї ж бази даних уінтересах кількох інформаційних технологій зумовлює необхідність розгляду методів та засобів управління та доставки однорідних інформаційних ресурсів на основі своєрідної суперпозиції механізмів маршрутизації та комутації інформаційних потоків.

У міру розширення міжнародних ринків та інтернаціоналізація бізнесу зростають вимоги до продуктивності розподілених автоматизованих систем. З кожним роком відбувається кількісне збільшення трафіку у телекомунікаційних мережах та якісне ускладнення складу цього трафіку. Останнім часом активізувалися найбільш характерні види інформаційних потоків: комерційна, довідково-адресна, нормативно-правова, фінансово-біржова, рекламна, рейтингова, економіко-статистична інформація тощо.

Розподілені автоматизовані системи стають матеріальнимиосновою інформаційної структури, що об'єднує засоби телекомунікації та обчислювальної техніки в єдину систему обміну, зберігання, обробки та управління, що забезпечує інформатизацію суспільства.

На підставі проведеного аналізу найбільш популярних технологій організації баз даних для вирішення завдань інтеграції інформаційних ресурсів

розподілених систем можуть бути такі висновки:

– дослідження у сфері зміни та взаємодії розподілених баз даних залишаються областю інтенсивних досліджень, і розробок;

– підходи до організації даних регіональних інформаційних систем на основі схем глобальних моделей є громіздкими та технічно важко реалізованими;

– перспективним напрямом доцільно визнати комбінований спосіб, що поєднує ефективність розподілених однорідних баз даних із перевагами «клієнт-серверної» взаємодії.

2.3 Дослідження особливостей розвитку інформаційних технологій в економіці та бізнесі

Питання про те, що представлятиме інформаційна система майбутнього, вже давно обговорюється на сторінках спеціалізованих видань. Багато авторів небезпідставно прогнозують бурхливий розвиток аналітичних блоків та модулів управління. Цей напрямок – один із пріоритетних серед перспектив розвитку корпоративного програмного забезпечення. Однак не варто забувати і про ту частину корпоративної системи, яка забезпечує безпосередньо автоматизацію бізнес-процесів. Цей аспект автоматизації анітрохи не втратив своєї актуальності і, очевидно, залишиться важливим напрямком для розробників автоматизованих систем у майбутньому.

Історично розвиток автоматизованих інформаційних систем, орієнтованих працювати з фінансовою інформацією, йшло у двох основних напрямах.

Перше – автоматизація основних облікових функцій. Характерною ілюстрацією тут можуть бути системи типу «Операційний день» для кредитних установ та бухгалтерські програми для підприємств. За принципами і концепції побудови це були дуже схожі продукти, які дозволяли вводити проводки і

отримувати з них баланс і набір основних бухгалтерських звітів.

Другий напрямок – автоматизація окремих ділянок роботи, створення спеціалізованих робочих місць. При цьому відповідні системи були жорстко прив'язані до фактичної реалізації окремих завдань і рідко були комплексними рішеннями з автоматизації взаємопов'язаних технологічних операцій.

До якого напряму не ставилися подібні спеціалізовані програмні продукти, вони справлялися з автоматизацією бухгалтерського обліку, але з вирішували завдання комплексної автоматизації. Тому, незважаючи на здійснену «комп'ютеризацію», співробітникам банків, комерційних організацій та підприємств доводилося обмінюватися величезною кількістю паперових документів. Найчастіше вони змушені були по кілька разів вводити ту саму інформацію. Крім того, порядок обробки того чи іншого документа не був чітко формалізований, і, відповідно, при виконанні відповідальних фінансових операцій на чільному місці стояв людський фактор.

Так можна сформулювати першу вимогу: система повинна автоматизувати не лише бухгалтерський облік, а й документообіг організації. Автоматизована система документообігу повинна мати такі властивості:

– обробляти різноманітні типи документів (фінансових та інших), список яких можна змінювати та розширювати;

– мати вбудовані засоби визначення як набору типів оброблюваних системою документів, а й реквізитного складу кожного їх. При цьому екранні форми подання будь-якого типу документа повинні налаштовуватися під конкретного користувача або групу користувачів;

– забезпечувати формалізацію опису технології обробки документів: стадій обробки, правил переходу від однієї стадії до іншої, виконавця чи групу виконавців кожної стадії тощо;

– забезпечувати настроювання облікових операцій, що виконуються на тій чи іншій стадії;

– автоматично виконувати облікові дії, залучаючи бухгалтерські та

фінансові документи, що породжуються, до загального документообігу;

– вирішувати завдання однозначної ідентифікації користувачів, що виконують у системі ту чи іншу дію;

– гарантувати незмінність інформації, що вводиться користувачем. Від паперового документообігу, однією з цілей якого є юридичне обґрунтування, як можливості, так і факту виконання тієї чи іншої операції, можна відмовитися, лише маючи в системі вбудовані можливості щодо використання електронно-цифрового підпису;

– мати прозорий для кінцевого користувача інтерфейс із офісними програмами.

Тільки за наявності подібних властивостей системи можна перейти від автоматизації облікових операцій до автоматизації бізнес-завдань.

Якщо система вміє обробляти різні види документів, має гнучке налаштування і прозорий інтерфейс з іншими системами, то в автоматизований документообіг можна включити навіть документи, створені поза корпоративною мережею. Наприклад, до картки клієнта можуть бути приєднані установчі документи та бізнес-план організації, які клієнт надав банку. Крім того, застосування індивідуального налаштування форм та списків кожного робочого місця позначиться на підвищенні ергономічності роботи кінцевого користувача. А можливість описати технологічні процеси значно знизить кількість інструкцій щодо виконання бізнес-операцій та перекласти функцію контролю належного виконання бізнес-правил з людини на комп'ютер. Автоматичне формування облікових документів з одного боку, зменшить залежність від людського фактора.

Наступним важливим завданням може стати завдання стирання меж між спеціалізацією фінансових установ.Якщо взяти за основу законодавство, обов'язкову звітність, межу провести досить легко. Але якщо розглядати ці суб'єкти економіки з погляду організаційної структури, технологій ведення бізнесу, тобто. з позицій, притаманних інформаційних технологій, то провести

чіткий поділ сьогодні досить складно.

Напрошується висновок: системи для таких фінансових інститутів мають бути якщо не однаковими, то схожими за принципами своєї побудови. Це означає, що вимоги до подібних систем повинні бути універсальними за такими компонентами:

– облікове ядро системи (різноманітність планів рахунків, принцип подвійного проведення, переоцінка валютних рахунків, ведення рахунків аналітичного обліку, функції нарахування відсотків та взяття комісій за обслуговування тощо);

– система розмежування доступу;

– комплект основних звітів про виконані операції;

– комплект звітів для внутрішнього фінансового обліку;

– комплект звітів, які надаються клієнту.

Принципи побудови систем такого класу та набір функціональних можливостей мають бути спільними, налаштування системи – індивідуальне. Звідси виходить друга вимога: принципи побудови ядра системи мають бути універсальними. Адже побудова системи відповідно до концепції (або моделі) офісної системи закономірно заперечує жорстку орієнтацію продукту на форму організації суб'єкта, що автоматизується. Маючи можливість налаштувати документообіг і пов'язаний з ним бухгалтерський облік, ми не замислюватимемося, для організації якої юридичної форми це виконано. Різниця лише у специфічних реквізитах об'єктів обліку та організації документообігу.

Ще одна тенденція, яка явно позначилася останнім часом – розширення спектру завдань, що делегуються у віддалені офіси, та подальший розвиток мережі таких офісів.

Обробка інформації у такій системі виконується на центральному комп'ютері. У всіх офісах організації встановлені термінали, що взаємодіють із центральним комп'ютером. У віддаленому офісі можуть виконуватись будь-які

операції, доступ до яких дозволено конкретному виконавцю, як операції з обслуговування клієнтів, так і облікові операції. Причому ці функції можуть бути і розділені, і суміщені. Система у разі має максимальну гнучкість.

За такої побудови бізнес-процесів інформаційна система повинна мати вбудовані засоби обміну даними. Причому системних засобів реплікації промислових СУБД буде недостатньо, оскільки очевидна необхідність на прикладному рівні забезпечувати логіку синхронізації станів об'єктів системи.

У цьому випадку інформаційна система має забезпечувати:

– формальний розподіл кожної операції на технологічні фази (фактично це не що інше, як побудова системи на принципах документообігу);

– можливість обміну даними між територіально-розподіленими виконавцями у межах однієї технологічної операції.

Стає очевидним, що застосування тієї чи іншої технології взаємодії визначається доцільністю економічної, технологічної, адміністративної і ставить перед нами третю вимогу: інформаційна система повинна мати можливість організації розподіленого документообігу і дозволяти застосовувати і поєднувати всі можливі технології віддаленої взаємодії.

Таким чином, розширюючи межі завдання, пов'язаного з побудовою системи документообігу, можна сформулювати четверту важливу вимогу: інформаційна система повинна забезпечувати документообіг, що охоплює всі підрозділи корпорації: її відділення, філії, дистанційні офіси та представництва.

Особливо слід розглянути одну з найпопулярніших технологій типу «Клієнт-Банк», орієнтовану на передачу платіжних доручень до кредитної установи та надсилання виписок з банку в офіс клієнта. З розвитком цієї технології стрімко розширювалася і номенклатура повідомлень – заявки на конвертацію валюти, на зняття готівки, інформаційні листи тощо. У результаті система обміну фінансовими повідомленнями трансформувалася у комплексну систему взаємодії клієнта та банку. Багато клієнтів активно переміщаються не тільки по країні, але і по всьому світу, і як офіс у них виступає переносний

комп'ютер. Понад те, комп'ютер то, можливо не особистим (припустимо, комп'ютер, встановлений в Internet-кафе), проте взаємодія здійснюватися через глобальну мережу. Можна сказати, що в даному випадку "офіс" - це людина.

Якщо з боку банку, то цю тенденцію можна охарактеризувати як наближення до клієнта. Стирається «вододіл» між постачальником послуг та споживачем, банк входить до будинку клієнта, відвідує його офіс, потрапляє на його робочий стіл. І навпаки, клієнт із будь-якої точки земної кулі отримує багато послуг, доступних раніше тільки в офісі кредитної установи.

Можна назвати п'яту вимога: інформаційна система має забезпечити інтеграцію у єдиний інформаційний простір (єдиний документообіг із розподіленою обробкою даних) як всіх підрозділів банку, а й усіх клієнтів за повним переліком наданих услуг.

Підсумовуючи вищевикладене, можна стверджувати, що магістральна тенденція розвитку програмних засобів автоматизації фінансових операцій полягає в універсалізації цих коштів. Розробка інформаційних систем дедалі рідше орієнтуватиметься на вузькоспеціалізовану область діяльності [5].

В результаті проведеного аналізу можна зробити такі висновки:

– програмне забезпечення найближчого майбутнього буде не банківські або бухгалтерські програми, а корпоративні офісні системи;

– нині існує тенденція до децентралізації зберігання даних та бізнес-серверів, системи майбутнього – це корпоративні мережі;

– як базову технологію реалізації розглядається архітектура «клієнт-сервер»;

– технологія потоків завдань (бізнес-завдань) є визначальною серед різноманітних методів предвідношення інформації.

2.4 Постановка задач синтезу схеми розподіленої реляційної бази даних

В даний час ключова роль у досягненні успіху більшості

комп'ютеризованих систем належить програмному забезпеченню. В останнє десятиліття прикладні програми пройшли шлях від маленьких і простих додатків з кількох рядків коду до дуже великих і складних додатків, що складаються з кількох мільйонів рядків. Багато цих додатків вимагали постійного супроводу, включаючи виправлення виявлених помилок, реалізацію нових вимог користувачів, а також перенесення програмного забезпечення на нові або модернізовані обчислювальні платформи. Зусилля й ресурси, витрачаються нині супровід програмного забезпечення, зростають дуже високими темпами.

2.4.1 Основні проблеми проектування інформаційних систем

Розробка та реалізація багатьох сучасних великих інформаційних проектів має, як правило, затяжний характер, їхня вартість перевищує заплановану, а остаточний продукт виходить ненадійним і складним у супроводі. Все це призвело до ситуації, яка відома під назвою «криза програмного забезпечення». Хоча перші згадки про кризу були зроблені ще наприкінці 80-х років, навіть через 30 років його все ще не вдалося подолати.

Деякі причини загальної проблеми проектування складних інформаційних систем полягають у наступному:

– розробка близько 40% систем закінчується невдало чи припиняється до завершення робіт;

– раціонально інтегрувати інтереси бізнесу та використовуваної інформаційної технології вдається не більше ніж у 25% систем;

– лише 20-30% інформаційних систем відповідають усім критеріям досягнення успіху.

Основні невдачі при створенні програмного забезпечення були викликані відсутністю повної специфікації всіх вимог на етапі проектування, прийнятною методологією розробки або недостатнім ступенем поділу загального

глобального проекту на окремі компоненти, що піддаються ефективному контролю та управлінню.

У разі часткової реалізації вимог користувачів інформаційної системи (ІС) або зміни бізнес-процесу настільки, що система перестає відповідати вимогам користувачів, можливі кілька варіантів розвитку:

– розробка нової системи;

– модифікація (розвиток) існуючої (успадкованої – legacy system) системи;

– реінженіринг існуючої (успадкованої – legacy system) системи.

Перший варіант найпростіший і переважний для розробника, але це шлях найменше задовольняє вимогам користувачів, т.к. потрібні витрати додаткового часу та фінансових ресурсів, а також існують ризики розробки та ризики втрати накопиченої інформації за час існування ІС, що експлуатується.

Реінженіринг успадкованих інформаційних систем потребує залучення експертів у галузі інформаційних систем та технологій, що відповідно призводить до надзвичайної складності таких робіт.

Найчастіше існує така думка: простіше розробити систему наново, ніж вдатися до її реінженірингу. Це з кваліфікацією фахівців, яких необхідно залучити щодо робіт. Вона повинна бути досить високого рівня для вирішення комплексу завдань проектування та створення модифікованої інформаційної системи [6].

2.4.2 Постановка задачі

Метою проведених досліджень є розробка методу синтезу реляційної моделі даних, отриманої з урахуванням інтеграції моделей даних різних незалежних предметних областей.

Найважливішу роль реалізації проекту ІС займає інформаційне забезпечення, саме внутримашинное інформаційне забезпечення – бази даних і

банки даних. На сьогоднішній день переважна більшість ІС використовують системи управління базами даних, в основі яких закладено реляційну модель зберігання даних, запропоновану Е.Ф. Коддом [7]. Теоретично дана модель описана більше 40 років тому, проте, широке практичне застосування набула не більше ніж 15 років тому. На даному етапі розвитку інформаційних технологій широко відомий суворий математичний апарат, що дозволяє описати структуру бази даних, операційну складову, алгоритми пошуку функціональних залежностей та нормалізацію схем баз даних.

Існує класичний опис реляційної моделі даних, у якому виділяють три основні функціональні компоненти:

- структурний компонент;
- обмеження цілісності;
- операційна специфікація.

Структурна компонента реляційної моделі даних (SRM) – n-арне відношення:

$$SRM = \{R,D,A,dom\}, \quad (3.1)$$

де R – безліч імен відносин;

D – безліч доменів;

A – безліч імен атрибутів;

dom – відображення з A до D.

Перш ніж перейти до розробки методу синтезу інтегрованої схеми бази даних, коротко розглянемо основні поняття реляційного підходу.

Будь-який елемент D_i множини D називається доменом і дозволяє називати нескінченну множину елементарних даних або значень. Для елемента A_i множини A визначено безліч значень атрибуту, що збігається з одним із доменів. У реляційній моделі даних задається відображення $\mathrm{dom}A \to D$, пара $< \mathrm{A_i dom(A_i)} >$ називається атрибутом з ім'ям A_i та областю значень $\mathrm{dom}A_i = D_j$.

Вираз$R_i(A_i...A_n)$, в якому всі іменаA_iрізні, називається схемою відношення. Безліч імен атрибутів у схемі відношення називається носієм відношення. Кожній схемі відношенняR_iмодель ставить у відповідність безліч кортежів декартовоїмножини:$r_i = \mathrm{domA}_1 \times ... \times \mathrm{domA}_n$.

Схемоюреляційноїбазиданихназиваєтьсякінцевийнабірсхемвідносин $\mathrm{B} = \{\mathrm{R}_1,..., \mathrm{R}_\mathrm{p}\}$.Реляційноюбазоюданихзісхемою B називаєтьсябезлічреалізаційсхемR_1,…,R_pвідносин $\mathrm{b} = \{\mathrm{r}_1,..., \mathrm{r}_\mathrm{p}\}$.

Розглянемо інформаційний простір, у межах якого функціонують безліч інформаційних систем, спроектованих у час і вирішують різні завдання інформаційної підтримки. В основі кожної ІС використовується реляційна база даних та, відповідно, відома її реляційна модель. Метою проведених досліджень розглядається завдання побудови єдиної інтегрованої реляційної моделі бази даних, яка б по можливості поєднати схеми різних БД без втрати їх функціональності і тих даних, які в них зберігаються. В даний час методи та алгоритми вирішення такого класу завдань відомі тільки для окремих випадків та для певних предметних областей. Однак, єдиної методики та технології для вирішення такого роду завдань на формальному математичному рівні на сьогоднішній день немає.

2.5 Визначення класу інтегрованих розподілених баз даних

Класи баз даних, розподілених баз даних та інтегрованих розподілених баз даних утворюють ієрархію «узагальнення-спеціалізація», тому визначення інтегрованої РБД можна використовувати успадкування істотних властивостей об'єктів породжуючих класів.

Будь-яка база даних визначається через сукупність пов'язаних (інтегрованих у базу) даних і є інформаційним ресурсом, що розділяється, комп'ютеризованих технологій. Тоді будь-яка розподілена база даних також є інформаційним ресурсом (сукупністю локальних інформаційних

ресурсів) у вигляді пов'язаних розподілених даних (рисунок 2.3):

Рисунок 2.3 – Ієрархія «Узагальнення-спеціалізація»

Вочевидь, що визначення поняття «інтегрована розподілена база даних» насамперед необхідно уточнити суть поняття «розподілені дані».

Під розподіленістю даних зазвичай розуміється фізична роздільність їх розміщення, рознесення даних з різних комп'ютерів мережі.Проте використання природних ознак класифікування розподіленості даних виявляється недостатньо. Наприклад, зовсім не просто вирішити таке завдання у разі, коли дані зберігаються у двох таблицях, створених з використанням різних СУБД, але в одному директорії одного й того самого комп'ютера. Тому поняття «розподілені дані» слід сформулювати штучно, спираючись на техніку опису об'єктів даних у комп'ютерних технологіях.

Для вираження сутності розподілених даних можна використати та обставина, що у сучасних комп'ютерних технологіях бази даних реєструються в системах драйверів БД шляхом вказівкипсевдоніма, чи імені системного ресурсу.

Псевдонім грає роль розв'язки між логічним ідентифікатором БД та фізичними характеристиками її побудови.

Зазвичай псевдонім БД співвідноситься з її фізичного розміщення в дискової пам'яті, типом використаної СУБД, параметрами сторінкової

організації тощо. буд. Тоді будь-які дані БД може бути ідентифіковані через її псевдонім (для реляційних БД елементами даних є таблиці, стовпці таблиць і атрибути) (рисунок2.4):

Такий підхід до вираження сутності розподіленості даних БД дозволяє сформулювати такі визначення: якщо в сукупності даних існують хоча б два елементи даних із різними псевдонімами, то така сукупність є середовищем розподілених даних; розподіленими даними може бути будь-які елементи баз даних, ідентифіковані псевдонімами.

В умовно логічній формі дані визначення виражаються як: розподілений об'єкт даних → ідентифікований псевдонімом елемент бази даних → локальний інформаційний ресурс.

Рисунок 2.4Підхід вираження сутності розподілу даних БД

Прийняття сформульованого логічного правила у складі формованої інтеграційної методології забезпечує точність виділення в інформаційному середовищі розподілених об'єктів даних та чіткість подальшого оголошення сукупності таких об'єктів як розподілену базу даних. Поняття локального інформаційного ресурсу стосовно розподілених БД уточнюватиметься з розкриття змісту інших компонентів інтеграційної методології.

Ідентифікація розподілених даних за допомогою псевдоніму призводить до ще одного важливого визначення у складі формованої методології. Сучасні системи драйверів БД (реєстрації псевдонімів) збудовані в архітектурі клієнт-

сервер. Іншими словами, з'єднання додатків з розподіленими даними через псевдонім будується по мережній архітектурі, що дозволяє трактувати будь-яке середовище розподілених даних як мережеве, забезпечуючи тим самим типізацію способів її побудови та опису. Фактично псевдонім є мережевою адресою розподілених даних незалежно від того, як фізично реалізована мережева архітектура, з тим застереженням, що при цьому має враховуватись режим локального чи віддаленого доступу. Тоді дві різнотипні таблиці з наведеного вище прикладу, що зберігаються в одному директорії одного комп'ютера, але ідентифіковані різними псевдонімами, у сукупному розгляді утворюють мережеве розподілене інформаційне середовище, якщо такі таблиці задовольняють вимогу уявлення у вигляді локальних інформаційних ресурсів.

Узагальнюючим поняттям розподілених даних є розподілені об'єкти даних. Суть такого узагальнення у тому, що у сучасних комп'ютерних технологіях інформаційними ресурсами як об'єктів даних може бути як елементи БД, а й об'єкти інтеграційних технологій OLEDB, ActiveX та інших. На відміну від реляційних об'єктів даних, об'єкти інтеграційних технологій характеризуються неоднозначністю можливостей декомпозиції їхнього подання. Наприклад, таблиця Excel як об'єкта інтеграційної технології є неподільним інформаційним ресурсом (наприклад, як help-описания якогось об'єкта даних), але, перетворена на структурований файл формату .dbf із зазначенням псевдоніма, таблиця може перетворитися на повноцінний елемент БД. Подібні перетворення уявлень інформаційних ресурсів відносяться до ведення самих інтеграційних технологій. Тут же важливим є сам факт існування у розподіленому інформаційному середовищі об'єктів даних нереляційного типу [8]. Надалі поняття розподілених даних та розподілених об'єктів даних будуть вважатися синонімами.

Таким чином, сукупність розподілених об'єктів даних становить розподілене інформаційне середовище в мережевому поданні. Усі розподілені об'єкти даних характеризуються наявністю у явному (як визначень) чи

неявному (як обмежень і залежностей) вираженні локальної пов'язаності своїх даних. Розподілені дані відповідають сутності локальних інформаційних ресурсів, причому крім елементів реляційних БД у розподіленому інформаційному середовищі, допустимо оголошення нереляційних об'єктів інтеграційних технологій.

У узагальненому вигляді класифікаційне визначення розподіленої бази даних випливає з визначення БД і формулюється у вигляді: розподілена база даних – це розподілені дані та зв'язки між ними. Локальна пов'язаність розподілених даних притаманна визначенням локальних БД. Для розподіленої БД ключовим об'єктом визначення стають зв'язки між розподіленими даними. Тобто тепер необхідно сформулювати умови, за яких розподілене інформаційне середовище набуває статусу розподіленої бази даних.

Однією з головних переваг застосування реляційної моделі є забезпечення однорідності табличного уявлення будь-яких БД. У термінах розширеного реляційного підходу така однорідність сприймається як датологічна однорідність реляційних БД. Саме ця обставина супроводжується сьогодні формуванням стандарту доступу БД на основі мови SQL. Специфіка СУБД, що виявляється у тому числі й у існуванні діалектів мови SQL нівелюється шляхом застосування засобів логічного з'єднання з БД у вигляді механізму псевдонімів.

Реляційна модель забезпечує однорідність ппредвідношення розподілених даних. Механізм псевдонімів забезпечує однорідність середовища маніпулювання розподіленими даними (наприклад, система драйверів БД DAO використовує програму MicrosoftJetDatabaseEngine для однорідного маніпулювання базами даних MicrosoftAccess, а також MySQL та Excel). У сукупності, таким чином, реалізується однорідне середовище визначення та маніпулювання розподіленими даними. Назвемо уявлення такого середовища схемним модельним рівнем інтегрованого уявлення сукупності розподілених даних з огляду на загальноприйняте розуміння реалізації реляційних БД у

вигляді реляційних схем. Тоді можна сказати, що реалізація уявлення сукупності (інтегрованого уявлення) розподілених даних на рівні схемної моделі забезпечує задоволення представленої таким чином сукупності розподілених даних вимог реляційної моделі. Сукупність розподілених даних стає при цьому реляційною БД, і саме така реляційна БД і називатиметься розподіленою базою даних. Враховуючи інтеграційний характер об'єднання розподілених даних у сукупність, схемну однорідність РБД називатимемо схемною інтеграцією. Тоді визначення розподіленої БД формулюється так: розподілена база дачах – це щонайменше схемно інтегрована сукупність розподілених даних.

По суті, таке визначення розподіленої БД означає побудова сукупності локальних інформаційних ресурсів як однорідного середовища розподілених даних, причому однорідність розглядається з погляду визначення та маніпулювання розподіленими даними у межах реляційного підходу або лише на рівні даталогічних схем локальних елементів БД. Узагальнена даталогічна схема РБД у принципі доступна до виконання стандартних операцій узагальненої нормалізації, проте цьому необхідно враховувати мети побудови РБД, яку які завжди доцільно і можна сформувати як єдиної реляційної БД.

Розширення реляційного підходу передбачає уявлення баз даних з повної технологічної схемою, тобто. включаючи формалізовані описи інфологічної, концептуальної та фундаментальної моделей БД. Розподілені об'єкти даних цих рівнях уявлення різнорідні за визначенням, змістовно, а чи не технічно, з їхньої побудови у сфері локальних систем та додатків. Визначимо інфологічний, концептуальний та фундаментальний рівні семантичними рівнями предвідношення РБД. І якщо локальних баз даних семантична інтеграція виконується явно чи інтуїтивно проектувальником БД, то розподілених БД досягнення семантичної однорідності стає істотною проблемою.

Інакше кажучи, говорячи про інтеграцію РБД, завжди буде на увазі семантична інтеграція, досяжна у межах розширеного реляційного підходу.

Схемна інтеграція РБД однозначна (табличне уявлення розподілених даних незалежно від специфіки реалізації). Семантична інтеграція багатозначна і припускає безліч варіантів реалізації. Це означає, що визначення класу інтегрованих РБД постулює існування різновидів ІРБД, об'єднаних суттєвою властивістю семантичної однорідності (логічної, концептуальної чи фундаментальної) (рисунок2.5).

Сформульоване визначення розподіленої БД та її інтегрованих різновидів відповідає інтеграційній парадигмі. Особливістю такого підходу є апріорність наявності сукупності розподілених даних (локальних інформаційних ресурсів), яка формується як інтегрована РБД. Логічна формула інтеграційної парадигми записується як: розподілені дані → розподілена база даних.

Рисунок 2.5 – Схема семантичної інтеграції РБД

На відміну від інтеграційної, традиційна парадигма виражає інший підхід, що базується на наступній логіці міркувань: існує деяка база даних; далі з'являється можливість розподілу фрагментів початкової БД з урахуванням інформаційних потреб розподілених додатків. Насамперед, така можливість

обумовлюється технічно (наприклад, у результаті побудови мережевої архітектури). Розподілу, таким чином, підлягають фрагменти БД, котрим спочатку існує спільність сукупного уявлення.

Тоді логічна формула традиційної парадигми має вигляд:

База даних → розподілені фрагменти бази даних.

Традиційному підходу, заснованому на застосуванні способів декомпозиції спроектованої БД, присвячено безліч досліджень та розробок. Однією з основних завдань організації РБД у традиційному підході стає завдання вибору методів розподілу фрагментів БД. Механізми розподілу варіювалися від побудови централізованої РБД із забезпеченням розсилки необхідних копій фрагментів БД додатків до децентралізованої РБД. Традиційний підхід досліджувався теоретично реляційних БД, наприклад, шляхом розробки методів реляційного обчислення розподілених відносин. Ключовими завданнями стають мінімізація трафіку даних, оптимізація обчислень тощо. Нижче наведено приклади результатів таких досліджень.

Основною причиною зниження інтересу до тенденції розвитку традиційного способу визначення РБД стала поява технологій розробки розподілених програм в архітектурі клієнт-сервер. Завдання управління передачею розподілених даних перестали бути прерогативою систем управління РБД і в сферу відповідальності мережевих технологій доставки інформаційних ресурсів. В результаті розподілені БД як об'єкт досліджень значною мірою втратили статус самостійності.

Поява CASE-технологій перевизначили центр уваги проблем побудови РБД на організацію середовища розподілених додатків (рисунок 2.6).

Рисунок 2.6 – Етапи проектування розподілених додатків

Сформована таким чином прикладна парадигма характеризується реалізацією етапів проектування розподілених додатків, що супроводжуються відображенням даних предметних областей реляційні схеми баз даних. Логічна формула прикладної парадигми включає фонове визначення організації баз даних:прикладна постановка задачі → розподілене додаток на основі проектно-інтегрованої бази даних.

Логічний рівень проектування розподілених додатків у CASE-технологіях фактично реалізує елементи розширеного реляційного підходу, оскільки моделювання предметних областей здійснюється шляхом проектування логічних схем накопичувачів та потоків інформації, виділення їх властивостей, моделювання сутностей-зв'язків, перетворення отриманих логічних моделей на реляційні схеми та генерації. для створення та управління базами даних спроектованих розподілених додатків.

Побудова розподілених БД в архітектурі клієнт-сервер набула розвитку з появою технологій SQL-серверів, які вирішили багато проблем побудови РБД з використанням будь-яких підходів до їх визначення.

В аспекті застосування підходів до побудови розподілених БД можна слідувати такій класифікаційній схемі:

– інтеграційний підхід орієнтований на інтеграцію існуючих, локально створених та наповнених елементів баз даних;

– традиційний підхід спрямовано розподіл узагальнено представлених баз даних;

– прикладний підхід орієнтований створення нових систем розподілених додатків із забезпеченням побудови інтегрованих баз даних.

3 РОЗРОБКА МЕТОДУ СИНТЕЗУ СХЕМИ РЕЛЯЦІЙНОЇ МОДЕЛІ ДАНИХ

3.1 Стратегії побудови та інтеграції розподілених баз даних

Багато сучасних комп'ютерних технологій і підходів розглядаються в архітектурі клієнт-сервер, при цьому застосування такої архітектури стало сьогодні своєрідним методологічним підходом визначення комп'ютерних методів та засобів. Значною мірою популярність використання архітектури клієнт-сервер обумовлена орієнтацією такої архітектури на об'єктно-орієнтованого підходу.

Логіка міркувань при розгляді об'єктно-орієнтованого підходу у поняттях та термінах архітектури клієнт-сервер є наступною. Будь-який об'єкт, який використовує ресурси іншого об'єкта, визначається як клієнт, а об'єкт, що постачає ресурси, називається сервером. Поведінка об'єкта характеризується послугами, що надаються іншим об'єктом. Такий підхід зосереджує увагу на зовнішніх проявах об'єкта (абстрагування) і отримав назву контейнерної моделі. Зовнішнє прояв об'єкта реалізує договір взаємодії коїться з іншими об'єктами. Договір визначає відповідну відповідальність об'єкта. Контракт передбачає кінцеву множину передбачених операцій, яку називають протоколом. Для кожної операції задаються передумови та постумови. Зміна інваріанту завдання операції порушує договір. Таким чином, об'єктно-орієнтований підхід розглядається сьогодні як універсальна методологія аналізу, проектування та реалізації будь-яких складних систем у комп'ютеризованому виконанні.

3.2 Реляційна модель

Реляційна модель була розроблена професором Е. Ф. Коддом на початку

1970-х рр. н. З її створенням розпочався новий етап у еволюції СУБД. Простота та гнучкість моделі привернули до неї увагу розробників та здобули їй безліч прихильників. Незважаючи на деякі недоліки, реляційна модель стала домінуючою, а реляційні СУБД стали промисловим стандартом де-факто. Реляційна модель заснована на математичному понятті відносини, фізичним уявленням якого є двомірна таблиця, що складається з рядків однакової структури. Логічна структура даних є набором пов'язаних таблиць. Модель підтримує зв'язки «один до одного» та «один до багатьох». Зв'язок «багато до багатьох» реалізується з допомогою декомпозиції.

Розглянемо детальніше реляційну модель.

Як зазначалося, будь-яка база даних складається з описів об'єктів деякої предметної галузі, і навіть містить інформацію про взаємозв'язки між об'єктами. Тип об'єкта називається сутністю, а характеристики об'єктів – атрибутами. Таким чином, сутності відповідає певний набір атрибутів, кожному конкретному об'єкту відповідає набір значень атрибутів. Набір атрибутів, що однозначно визначає кожен об'єкт, називають ключем. Атрибут можна розглядати як змінну, що приймає значення з певної кількості значень, званого доменом атрибута.

Розглянемо об'єкт типу T, який має набір атрибутівA_1,A_2,...,A_n. АтрибутA_jможе набувати значень з галузі (домена)D_j,j = 1,2,...,n. Позначимо черезa_{ij}значення атрибутаA_jдля об'єкта i, тоді кожному конкретному об'єкту i типу T відповідає кортеж виду:

$$a_i = (a_{i1},a_{i2},...,a_{in}),a_{ij} \in D,i = 1,2,...,m;j = 1,2,...,n;, \quad (3.1)$$

dem –кількістьоб'єктівтипуT.

Усьомунаборуоб'єктівтипуTвідповідаєнабіркортежів:

$$R = \begin{pmatrix} a_{11}, & a_{12}, & ..., & a_{1n} \\ a_{21}, & a_{22}, & ..., & a_{2n} \\ ... & ... & ... & ... \\ a_{m1}, & a_{m2}, & ..., & a_{mn} \end{pmatrix}, \quad (3.2)$$

Зрозуміло, що$R \subseteq D_1 \times D_2 \times ... \times D_n$.

Багато кортежів R називають відношенням, а кількість атрибутів n – арністю відношення. Кількість кортежів, що містяться у відношенні, називається кардинальністю відношення. Зауважимо, що оскільки відношення – це безліч, то порядок прямування кортежів щодо несуттєвий; відношення не містить однакових елементів - кортежів і, отже, обов'язково має набір атрибутів, що є ключем.

Сукупність атрибутів$R = (A_1, A_2, ..., A_n)$називається схемою відношення. Якщо позначити $U = \{A_1, A_2, ..., A_n\}$,то схему відношення можна записати у вигляді$R = (U)$.Саме відношення Rназивається поточним значенням або екземпляром схеми відношення R. База даних зазвичай містить кілька відношень:$R_1, R_2, ..., R_k$.Сукупність їхніх схем$R_1 = (U_1), R_2 = (U_2), ..., R_k = (U_k)$називається схемою реляційної бази даних.

Відношення можна розглядати як двомірну таблицю, кожен стовпець якої має ім'я – атрибут, а кожен рядок містить дані по одному об'єкту або дані зв'язку між декількома конкретними об'єктами.

Таким чином, набір кортежів R можна записати у вигляді (рисунок 3.1).

A_1	A_2	...	A_n
a_{11}	a_{12}	...	a_{1n}
a_{21}	a_{22}	...	a_{2n}
...	...	...	...
a_{m1}	a_{m2}	...	a_{mn}

Рисунок 3.1 – Набір кортежів R

Для таблиці повинні виконуватись такі правила:

– таблиця має ім'я, відмінне від імен інших таблиць;

– кожна клітина містить лише атомарне значення;

– кожен стовпець має унікальне ім'я;

– дані для стовпця беруться з однієї множини значень;

– порядок проходження стовпців не має значення;

– таблиця немає повторюваних рядків;

– рядки немає імен;

– порядок слідування рядків немає значення.

Будь-яка таблиця має один або кілька стовпців, значення яких однозначно ідентифікують кожен її рядок. Такий стовпець (або сукупність стовпців) називається первинним ключем. Взаємозв'язки таблиць реляційної моделі підтримуються зовнішніми ключами. Зовнішній ключ – це стовпець (або сукупність стовпців), значення якого однозначно характеризують сутності, представлені рядками іншого відношення, тобто задають значення їх первинного ключа.

Для користувачів ІС необхідно, щоб база даних відображала предметну галузь однозначно та несуперечливо, тобто задовольняла умовам цілісності. Для забезпечення виконання умов цілісності на базу даних накладаються деякі обмеження, які називають обмеженнями цілісності. Виділяють два основних типи таких обмежень: цілісність сутностей та цілісність посилань. Обмеження першого типу означає, що будь-яке відношення має мати первинний ключ (в принципі для відношення ця властивість має виконуватися автоматично). Обмеження цілісності посилань полягає в тому, що зовнішній ключ не може бути покажчиком на неіснуючий рядок у таблиці.

Контроль цілісності здійснюється перевіркою обмежень цілісності:

– ключовий стовпець неспроможна містити невизначене значення (визначник NULL);

– для пов'язаних таблиць кожному рядку основної таблиці відповідає

нуль або більше рядків підпорядкованої таблиці;

– для пов'язаних таблиць у підпорядкованій таблиці немає рядків, які мають батьківських рядків у основній таблиці;

– для пов'язаних таблиць кожен рядок підпорядкованої таблиці має лише один батьківський рядок в основній таблиці.

Основними операціями маніпулювання даних є: додавання рядків, модифікація рядків, видалення рядків.

В основі операцій над відносинами лежать операції реляційної алгебри та реляційного обчислення.

Перевагами реляційної моделі є простота, наочність, незалежність від даних. До того ж, на відміну від мережевих та ієрархічних моделей, реляційні моделі для організації зв'язків між записами застосовують не внутрішні покажчики, а фактичні значення атрибуту, використовуючи загальний атрибут у кожному із записів.

Недоліки реляційної моделі пов'язані з однорідністю структури даних, семантичною перевантаженістю моделі, обмеженим набором операцій [9].

3.3Проектування реляційних баз даних на основі нормалізації

Як зазначалося, завдання проектування реляційної бази даних полягає у виборі схеми бази з безлічі альтернативних варіантів, тобто, по суті, потрібно визначити набір схем відносин бази даних. Для задоволення цих вимог необхідно визначити, з яких відносин має складатися база даних та які атрибути повинні входити до цих відносин. Основний підхід ось у чому. Передбачається існування деякого універсального відношення, що містить усі атрибути бази даних, потім на основі аналізу зв'язків між атрибутами намагаються здійснити декомпозицію універсального відношення, тобто перейти до кількох відносин меншої розмірності, що задовольняють певним умовам, причому вихідне відношення має відновлюватися за допомогою операції реляційної алгебри.

Перш ніж перейти до розгляду методики побудови оптимальної схеми бази даних, розглянемо недоліки схем відносин, що найчастіше зустрічаються, або аномалії відносин. Наприклад, розглянемо схему відношення:

Виробники = (назв_вигот, адреса_вигот, виріб, кільк_за_рік, ціна).

Ця схема має низку недоліків:

– надмірність– адреса виробника повторюється для кожного виробу;

– потенційна суперечливість (аномалії поновлення). Внаслідок надмірності можливе оновлення адреси виробника в одному кортежі, залишаючи його незмінним в іншому, тобто можлива ситуація, коли база даних містить різні адреси для одного виробника;

– аномалії включення. В базу даних не може бути записана адреса виробника, якщо не відомо, які вироби та в якій кількості він виготовляє;

– видалення аномалії. Зворотна проблема виникає при необхідності видалення всіх виробів, що виготовляються певним виробником, внаслідок чого ми втрачаємо його адресу, що завжди бажано.

У цьому прикладі всі ці недоліки відсутні, якщо замінити вихідне відношення двома відносинами (рисунок 3.2).

Изг_адр = (назв_изгот, адрес_изгот),
Изг_изд = (назв_изгот, изделие, колич_за_год, цена).

Рисунок 3.2 – Заміна вихідних відносин

Однак наведена декомпозиція має істотний недолік: щоб отримати адреси виробників, потрібно виконати операцію природного з'єднання, яка працює порівняно повільно, проте, наведена декомпозиція явно краще вихідної схеми відношення.

Теорія реляційних баз даних має потужний інструмент, який здатний допомогти розробнику оптимальним чином спроектувати структуру відносин

бази даних. Цей інструмент – метод нормалізації відносин. Нормалізація відносин – покроковий процес розкладання (декомпозиції) вихідних відносин бази даних більш прості. Кожен ступінь цього процесу наводить схему відносин бази даних у послідовні нормальні форми. Кожна наступна нормальна форма має «кращі» властивості, ніж попередня. Кожній нормальній формі відповідає певний набір обмежень. Відношення знаходиться у певній нормальній формі, якщо воно задовольняє набору обмежень цієї форми [10].

Процес нормалізації грунтується на понятті функціональної залежності атрибутів.

Розглянемо докладно функціональні залежності та їх властивості.Нехай$R = (U)$ – схема відношення, $\mathrm{U} = \{\mathrm{A}_1, \mathrm{A}_2, ..., \mathrm{A}_\mathrm{n}\}$ - безліч атрибутів,X,Y $\subseteq U$. Говорять, що X функціонально визначає Y або що Y функціонально залежить від X, і позначають це$X \to Y$. Якщо у будь-якому відношенні R, що є поточним значенням схеми R, не можуть утримуватися два кортежі, компоненти яких збігаються по всіх атрибутах, що належать множині X, але не збігаються хоча б по одному атрибуту, що належить множині Y.

Функціональні залежності виникають різним чином, наприклад, якщо R містить опис набору об'єктів іA_1,A_2,...,A_n–атрибути цього типу об'єктів, а X – безліч атрибутів, що утворюють його ключ, можна стверджувати, що$X \to Y$для будь-кого $\mathrm{Y} \subseteq \{\mathrm{A}_1, \mathrm{A}_2, ..., \mathrm{A}_\mathrm{n}\}$. Це випливає з того, що кортежі R представляють об'єкти, а об'єкти однозначно ідентифікуються значеннями атрибутів ключа, отже, два кортежі, що збігаються за атрибутами, що належать X, повинні представляти той самий об'єкт і тому є одним і тим самим кортежем.

Слід зазначити, що функціональні залежності є твердженнями про всі відносини, які можуть бути значеннями схеми R, тобто функціональна залежність – це властивість схеми, а не конкретного екземпляра відношення. Отже, неможливо, аналізуючи конкретне поточне значення схеми R, визначити, які залежності мають місце для R, у кращому випадку можна стверджувати про деякі залежності, що вони не мають місця у схемі R. Єдиний спосіб визначення

функціональних залежностей для схеми відношення полягає в тому, щоб проаналізувати семантику атрибутів. У цьому сенсі залежності є фактично висловлюваннями про предметнігалузі, де вони можуть бути доведені формальними засобами проектування схем баз даних.

Нехай задано безліч атрибутів $U = \{A_1, A_2, ..., A_n\}$ тадеяка кількість функціональних залежностей F, записаних у вигляді пар підмножин(X, Y), $X, Y \subseteq U$, таких, що$X \to Y$; відповідну схему відносини записуватимемо у вигляді$R = (U,F)$. Говорять, що залежність$A \to B$логічно випливає з F, якщо для кожного екземпляра відношення R зі схемою R, що задовольняє залежностям F, виконується також залежність$A \to B$. Наприклад, легко показати, що якщо$X \to Y$і$Y \to Z$, то$X \to Z$.

НехайF^+позначає замикання F, тобто безліч всіх функціональних залежностей, які логічно випливають з F (включаючи, звичайно, саме F); F при цьому називають системою, утворюючою структури функціональних залежностей. Якщо$F^+ = F$, F називають замкнутим безліччю залежностей. Тепер розглянемо задачу пошуку залежностей, що логічно випливають із заданого набору F.

Як показав Армстронг, сукупність усіх пар(X, Y), таких, що$X, Y \subseteq U$і$X \to Y$, утворює структуру функціональних залежностей відношення R, яка характеризується наступним набором аксіом, званих аксіомами Армстронга:

а)якщо$X \supseteq Y$, то$X \to Y$(рефлексивність);

б) якщо$X \to Y$і$W \supseteq Z$, то$X \cup W \to Y \cup Z$(продовження);

в) якщо$X \to Y$і$Y \to Z$, то$X \to Z$(транзитивність).

Легко бачити, що аксіоми б) та в) можуть бути об'єднані в одну аксіому:

г) якщо$X \to Y$і$X \cup W \to Y \cup Z$, то$X \cup W \to Z$(псевдотранзитивність).

Корисні також такі властивості, що з а) – в):

д) якщо$X \to Y$і$X \to Z$, то$X \to Y \cup Z$(адитивність);

е) якщо$X \to Y$і$Z \subseteq Y$, то$X \to Z$(декомпозиція).

Аксіоми Армстронга можна як правила виведення, дозволяють виводити

функціональні залежності, логічно наступні із заданого набору залежностей.

Аксіоми Армстронга є надійними та повними, іншими словами, якщо залежність$X \to Y$виведена з F за цими аксіомами, то вона виконується в будь-якому відношенні, в якому виконуються всі залежності з F, і навпаки, якщо деяка залежність$X \to Y$не виводиться з F за аксіомами Армстронга, то можна побудувати екземпляр відносини, для якого виконуються всі залежності з F і не виконується залежність$X \to Y$.

Нехай задана схема відношення$R = (\mathrm{U,F}), \mathrm{X} \subseteq U$, замиканням безлічі атрибутів X щодо набору залежностей F називається безліч всіх таких атрибутів$A_i \in U$, що залежність$X \to A_i$виводиться з F за аксіомами Армстронга; замикання X позначаютьX^+.

Таким чином,X^+– максимальна за включенням підмножина множини U, функціонально залежна від X при заданому наборі функцій F. Зрозуміло, що залежність$(X \to Y) \in F$тоді і лише тоді, коли$X^+ \supseteq Y$.

Зазначимо такі властивості замикань:

а)$X^+ \supseteq X$;

б)$X \supseteq Y \Rightarrow X^+ \supseteq Y^+$;

в)$(X^+)^+ = X^+$.

Аналогічні властивості мають замикання наборів функцій.

ОбчисленняF^+– досить трудомістка задача, оскількиF^+може бути досить великим, навіть якщо F мало, і може містити близько 2n елементів, де n – кількість атрибутів. Фактично обчисленняF^+зводиться до обчислення замикань підмножини атрибутів.

Аксіоми Армстронга є досить зручними для безпосереднього застосування при обчисленні замикань підмножин атрибутів, і зазвичай використовується описаний нижче алгоритм, його правильність легко обґрунтовується тими ж аксіомами Армстронга. Цей алгоритм вимагає часу пропорційно до довжини всіх виписаних залежностей з набору F.

Нехай задана схема$R = (\mathrm{U,F}), \mathrm{X} \subset U$, потрібно вирахувати$X^+$.

Крок 1: покласти$X^{(0)} := \mathrm{X}$,i: $= 0$;

Крок 2: знайти$\Delta X^{(i+1)}$- багато таких атрибутів$y \in \mathrm{U}\backslash \mathrm{X}^{(i)}$, що існує певна залежність$V \to W \in F$, така що$X^{(i)} \supseteq V$і$y \in W$;

Крок 3: якщо$\Delta X^{(i+1)} = \emptyset$, то$\Delta X^{(i)} = X^{+}$і кінець роботи, інакше$X^{(i+1)} := X^{(i)} \cup \Delta X^{(i+1)}$,i: $= i + 1$та перейти до кроку 2.

Оскільки безліч атрибутів U звичайно, алгоритм завжди закінчує роботу за кінцеве число ітерацій.

Розглянемо приклад.

Нехай$R = (\mathrm{U,F})$– схема відношення, де$\mathrm{U} = \{\mathrm{A}_1, \mathrm{A}_2, ..., \mathrm{A}_7\}$

$\mathrm{F} = \{\mathrm{A}_1, \mathrm{A}_4 \to \mathrm{A}_2; \mathrm{A}_2 \to \mathrm{A}_3; \mathrm{A}_3 \to \mathrm{A}_4; \mathrm{A}_4, \mathrm{A}_7 \to \mathrm{A}_5; \mathrm{A}_5 \to \mathrm{A}_6; \mathrm{A}_6 \to \mathrm{A}_7\}$
,множина $\mathrm{X} = \{\mathrm{A}_3, \mathrm{A}_5\}$.

Потрібно вирахуватиX^{+}.

Випишемо послідовність дій кожного етапу.

$X^{(0)} = (A_3, \mathrm{A}_5)$;

$\Delta X^{(1)} = \{A_4, \mathrm{A}_6\}, \mathrm{X}^{(1)} = \{A_3, \mathrm{A}_4, \mathrm{A}_5, \mathrm{A}_6\}$;

$\Delta X^{(2)} = \{A_7\}, \mathrm{X}^{(2)} = \{A_3, \mathrm{A}_4, \mathrm{A}_5, \mathrm{A}_6, \mathrm{A}_7\}$;

$\Delta X^{(3)} = \emptyset$, отже, $X^{+} = X^{(2)} = \{A_3, \mathrm{A}_4, \mathrm{A}_5, \mathrm{A}_6, \mathrm{A}_7\}$.

Далі розглянемо ключі схеми відносин. Розглядаючи набори об'єктів, ми припускали, що є ключ – набір атрибутів, однозначно визначальний об'єкт. Аналогічне поняття існує і для відношення із заданим безліччю функціональних залежностей, тобто набір атрибутів, що однозначно визначає кортеж відношення.

Нехай задана схема відношення$R = (\mathrm{U,F}), \mathrm{X} \subseteq U$. Безліч X називають надключом схеми відношення R, якщо$X^{+} = U$, тобто залежність$\mathrm{X} \to \mathrm{U}$ належитьF^{+}.

Мінімальний за включенням надключ називається ключем, тобто Xключ схеми R, якщо:

а)$X^{+} = U$;

б) ні для якої власної підмножини$Y \subset \mathrm{X}, \mathrm{Y}^{+} \neq U$.

У літературі часто такий ключ називають можливим ключем, а власне ключем називають певний можливий позначений ключ; у свою чергу, такий позначений ключ іноді називають первинним, а решта – просто ключами. Тут ці терміни (можливий ключ, первинний ключ) не використовуються, оскільки вони пов'язані з семантикою бази.

У принципі, відношення може мати кілька ключів. Якщо атрибут міститься у певному ключі, його називають ключовим чи первинним атрибутом відносини, інакше атрибут називають непервинним.

Завдання знаходження довільного «першого-ліпшого» ключа відношення вирішується досить просто. Це можна зробити наступним чином: розглядається вся множина атрибутів U, з нього викреслюється будь-який атрибут, отримуємо множину V; якщо$V^{+} = U$то викреслений атрибут так і залишається викресленим, інакше повертаємо його на місце. Далі намагаємося викреслити наступний атрибут, так продовжуємо доти, доки жоден атрибут не можна буде викреслити. Набір атрибутів, що залишився, є ключем відношення.

Обчислення всіх ключів відносини – завдання досить трудомістке, оскільки ключів може бути експоненційно багато. Завдання знаходження мінімального за кількістю атрибутів ключа, і навіть перевірки, чи є конкретний атрибут ключовим (первинним), ставляться до класу NP-важких, тобто немає алгоритмів рішення з поліноміальної оцінкою трудомісткості.

Розглянемо приклад.

Для схеми відношення $\mathrm{R} = (\mathrm{U}, \mathrm{F})$, де $\mathrm{U} = \{\mathrm{A}_1, \mathrm{A}_2, ..., \mathrm{A}_8\}$,

$$\mathrm{F} = \{\mathrm{A}_1 \to \mathrm{A}_2; \mathrm{A}_2 \to \mathrm{A}_3; \mathrm{A}_3 \to \mathrm{A}_1; \mathrm{A}_2, \mathrm{A}_3 \to \mathrm{A}_4; \mathrm{A}_4, \mathrm{A}_7 \to \mathrm{A}_6; \mathrm{A}_6, \mathrm{A}_5 \to \mathrm{A}_7; \mathrm{A}_7 \to \mathrm{A}_8\}.$$

Потрібно знайти всі ключі, набори первинних та непервинних атрибутів та надключ.

В результаті рішення отримуємо:

а) всі ключі відношення:

$K_1 = \{A_1, A_5, A_6\}$, $K_2 = \{A_1, A_5, A_7\}$, $K_3 = \{A_2, A_5, A_6\}$,

$K_4 = \{A_2, A_5, A_7\}$, $K_5 = \{A_3, A_5, A_6\}$, $K_6 = \{A_3, A_5, A_7\}$;

б) безліч первинних атрибутів: $\{A_1, A_2, A_3, A_4, A_5, A_6, A_7\}$;

в) безліч непервинних атрибутів: $\{A_4, A_8\}$.

Ясно, що надключом є будь-яка безліч атрибутів, що містить ключ. У наведеному прикладі надключом є безліч.

Тепер перейдемо до нормальних форм відносин. Відношення нормалізоване або знаходиться в першій нормальній формі (1НФ), якщо домен кожного атрибуту складається з неподільних значень, а не множин або кортежів з елементарного домену або доменів, тобто значення атрибутів – це деякі елементарні величини, які не мають структури. Надалі розглядатимемо лише нормалізовані відносини.

Друга нормальна форма (2НФ)схеми відношення - це або дана схема, якщо вона нормалізована і містить тільки повні функціональні залежності непервинних атрибутів від ключів, або декомпозиція схеми, кожна з підсхем якої має зазначені властивості, а сама декомпозиція має властивість з'єднання без втрат.

Зауважимо, що якщо у схемі відношення всі атрибути первинні або будь-який ключ складається тільки з одного атрибута, то схема свідомо знаходиться у другій нормальній формі.

Друга нормальна форма існує для будь-якої схеми відношення.

Поняття третьої нормальної форми схеми відношення тісно пов'язане з поняттям транзитивної залежності, що має вигляд ланцюжка функціональних залежностей з певними обмеженнями.

Нехай задана схема відношення$R = (U,F), A,C \subseteq U$. Функціональна залежність$A \to C$називається транзитивною, якщо існує підмножина атрибутів$B \subset U$таке, що $A \not\supseteq B$, $B \not\supseteq C$, $A \to B$, $B \not\to A$ та $B \to C$.

Зазначимо, що наявність чи відсутність залежності$C \to B$значення немає. Якщо такої залежності немає, то транзитивна залежність називається строгою

транзитивною залежністю. Наприклад, $R = (\{A_1, A_2, A_3, A_4, A_5\},$

$\{A_1 \to A_2; A_2 \to A_3; A_1 \to A_4; A_4 \to A_5; A_5 \to A_1\})$.

Залежність$A_1 \to A_3$є транзитивною, тому що маємо ланцюжок$A_1 \to A_2 \to A_3$та $A_2 \nrightarrow A_1$.

Водночас залежність$A_1 \to A_5$не є транзитивною, хоча і є ланцюжок$A_1 \to A_4 \to A_5$, але тут$(A_4 \to A_1) \in F^+$, що суперечить визначенню транзитивної залежності.

Присутність у схемі транзитивної залежності$A \to B \to C$призводить до надмірності даних про зв'язок атрибутів B і C і, отже, надмірності значень атрибутів C, що може призвести до порушення цілісності бази даних при зміні значень атрибутів з множин C і B. Надмірність значень C зростає, якщо додатково $B \nleftarrow C$. Заміна схеми відносини, у якій є транзитивна залежність, декомпозицією, яка містить залежностей такого типу, позбавляє базу даних від зазначених недоліків. Така декомпозиція під час виконання певних умов називається третьою нормальною формою відносини.

Третя нормальна форма(3НФ) схеми відношення – це або дана схема, якщо вона знаходиться у другій нормальній формі і не містить транзитивної залежності непервинних атрибутів від ключів, або декомпозиція вихідної схеми, кожна підсхема якої задовольняє цим самим вимогам, а сама декомпозиція має властивість з'єднання без втрат. Третя нормальна форма існує для будь-якої схеми відносин [11].

3.4 Розробка методу синтезу схеми реляційної моделі даних

Як було зазначено вище, в інформаційному просторі організації, установи, підприємства можуть функціонувати різні системи, спроектовані на основі реляційного підходу, предметні галузі яких довільним чином можуть співвідноситися між собою. У кваліфікаційній роботі розглядаються питання

інтеграції розподілених баз даних, які можуть бути лише у вигляді універсального відношення. Якщо ПЗ характеризується схемою БД, в якій присутні відносини та зв'язки між ними, у цьому випадку шляхом використання запитів на об'єднання, вихідну схему необхідно представити у вигляді універсального відношення.

Введемо деякі позначення: нехай S_n– схема бази даних ІС, яка забезпечує інформаційну підтримку n-ої предметної галузі [12]. Розглянемо можливі випадки взаємного впливу різних ІС з прикладу двох систем, які представлені рисунку 3.3.

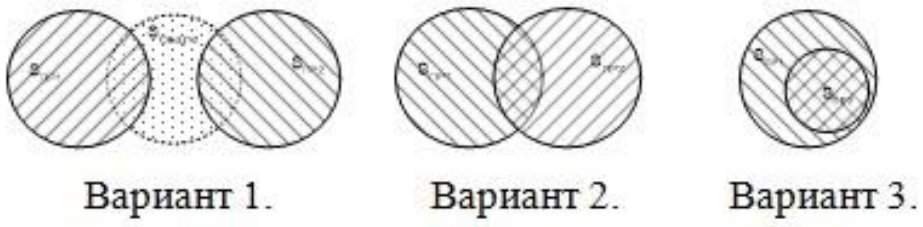


Рисунок 3.3 – Варіанти інтеграції двох предметних галузей

Варіант 1. Предметні галузі не перетинаються, схеми БД не містять даних про один і той же об'єкт реального світу. У разі прийняття рішення про інтеграцію таких схем БД вводиться додаткова підсхема ($S_{общ}$), яка дозволить зв'язати представлені схеми в єдину схему.

РозглянемовідносиниR_1 ($\underline{A_1}$, A_2, A_3) іR_2 ($\underline{B_1}$, B_2, B_3). Атрибути A1 та B1 єключовимивідповідноR_1іR_2 (рисунок 3.4).

R_1

A_1	A_2	A_3
a	d	g
b	e	h
c	f	i

R_2

B_1	B_2	B_3
k_1	k_4	k_7
k_2	k_5	k_8
k_3	k_6	k_9

Рисунок 3.4 – Відносини з різними атрибутами

Для інтеграції відносин R1 (A1, A2, A3) та R2 (B1, B2, B3) необхідно ці відносини перетворити на універсальні. Таким чином, отримуємо $R_1 \rightarrow R_1^u$ і $R_2 \rightarrow R_2^u$. Далі поєднуємо уявлення предметних галузей R_1^u U R_2^u, отримуємо універсальне відношення та послідовну декомпозицію проводимо на основі нормалізації до 3НФ (рисунок 3.5).

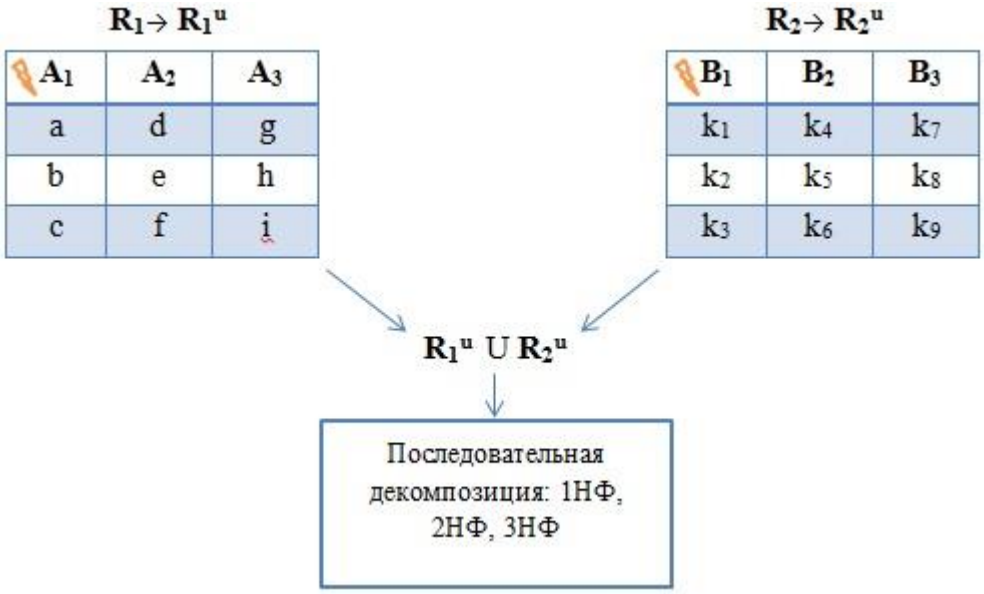


Рисунок 3.5 – Об'єднання відносин із різними атрибутами

Варіант 2. Предметні галузі частково перетинаються.

У такому разі, коли відносини мають атрибути, які частково перетинаються, можливі три випадки:

– відносини R_1 та R_2 мають однакові дані;

– відносини R_1 та R_2 мають загальні дані;

– відносини R_1 та R_2 мають різні дані.

На рисунку 3.6 представлені відносини R_1 ($\underline{A_1}$, A_2, A_3) иR_2 ($\underline{A_1}$, A_2, C_3). Атрибути цих стосунків частково перетинаються.

R_1

A_1	A_2	A_3
a	d	g
b	e	h
c	f	i

R_2

A_1	A_2	C_3
a	d	g
b	e	h
c	f	i

Рисунок 3.6 – Відносини з атрибутами, що частково перетинаються, які містять однакові дані

Відносини R_1 ($\underline{A_1}$, A_2, A_3) іR_2 ($\underline{A_1}$, A_2, C_3) містять однакові дані, з цього можна дійти невтішного висновку, що A_3= C_3. При об'єднанні цих двох відносин ми отримаємо схему ідентичну відносин R_1и R_2 (рисунок 3.7).

R_1

A_1	A_2	A_3
a	d	g
b	e	h
c	f	i

R_2

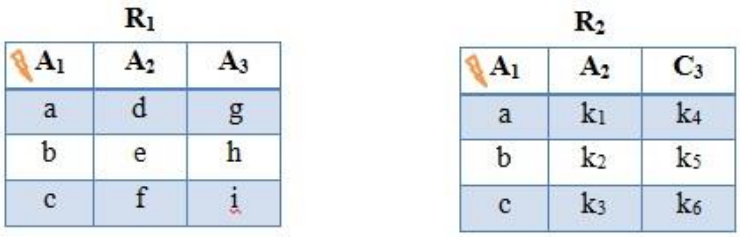
A_1	A_2	C_3
a	d	g
b	e	h
c	f	i

$R_1 \cup R_2$

A_1	A_2	A_3
a	d	g
b	e	h
c	f	i

Рисунок 3.7 – Об'єднання відносин з атрибутами, що частково перетинаються, які містять однакові дані

Розглянемо відносини R1 (A1, A2, A3) та R2 (A1, A2, C3). АтрибутиA_1увідносинахєключовимиатрибутами. Відносини містять загальні дані (рисунок 3.8).



R_1

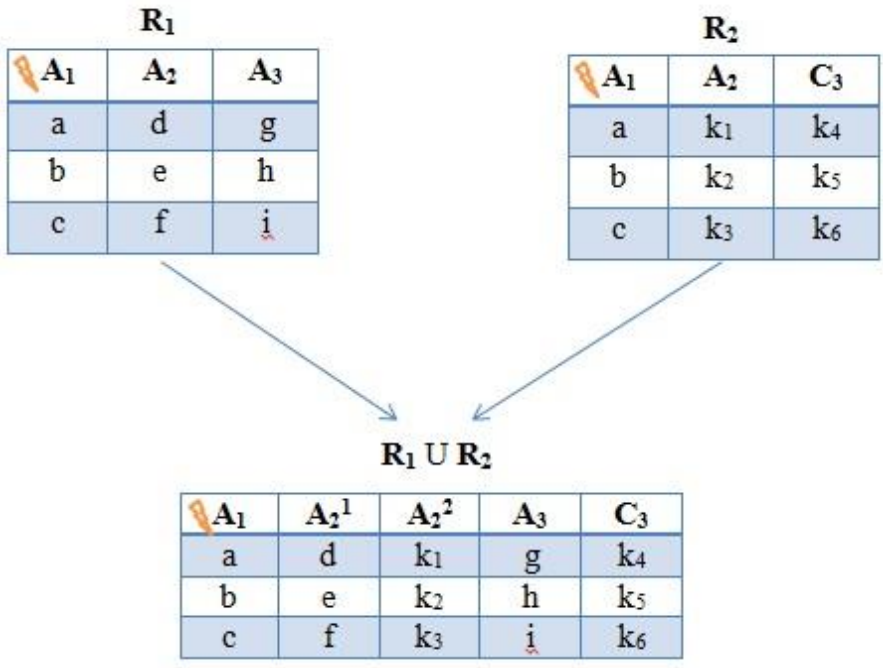
A_1	A_2	A_3
a	d	g
b	e	h
c	f	i

R_2

A_1	A_2	C_3
a	k_1	k_4
b	k_2	k_5
c	k_3	k_6

Рисунок 3.8 – Відносини з атрибутами, що частково перетинаються, які містять загальні дані

Для того, щоб поєднати схеми R_1і R_2, визначимо які з атрибутів містять однакові дані. У разі це атрибут A_1. Таким чином, при інтеграції схем, атрибут A_1 буде доданий до загальної таблиці і залишиться ключовим. Атрибут A_2 знаходиться в обох відносинах, але містить різні дані, тому для відношення R_1 перейменуємо $A_2=A_2^1$, а щодо R_2 атрибут $A_2 =A_2^2$ ідалі додаємо в таблицю атрибути A_2^1и A_2^2. Атрибути A_3 і C_3 переносимо в таблицю без змін (рисунок 3.9).



R_1

A_1	A_2	A_3
a	d	g
b	e	h
c	f	i

R_2

A_1	A_2	C_3
a	k_1	k_4
b	k_2	k_5
c	k_3	k_6

$R_1 \cup R_2$

A_1	A_2^1	A_2^2	A_3	C_3
a	d	k_1	g	k_4
b	e	k_2	h	k_5
c	f	k_3	i	k_6

Рисунок 3.9 – Об'єднання відносин з атрибутами, що частково перетинаються, які містять загальні дані

На рисунку 3.10 представлені відносини R_1 ($\underline{A_1}$, A_2, A_3) іR_2 ($\underline{A_1}$, A_2, C_3). Атрибути А1 у відносинах є ключовими атрибутами. Атрибути цих відносин містять різні дані.

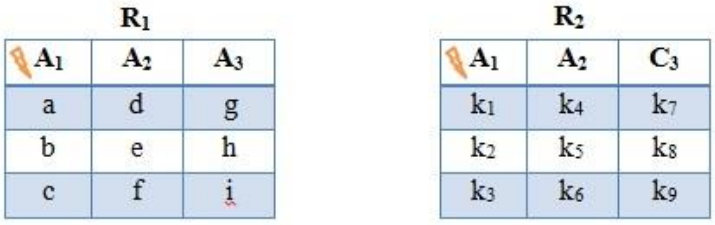


Рисунок 3.10 – Відносини з атрибутами, що частково перетинаються, які містять різні дані

Для того, щоб об'єднати два відносини, атрибути, які збігаються, в даному випадку A_1 іA_2додаємо в загальне відношення, а атрибути, в яких дані відрізняються не переносимо в загальну таблицю. Дані відносин R_1і R_2поєднуються за атрибутами A_1 іA_2(рисунок 3.11).

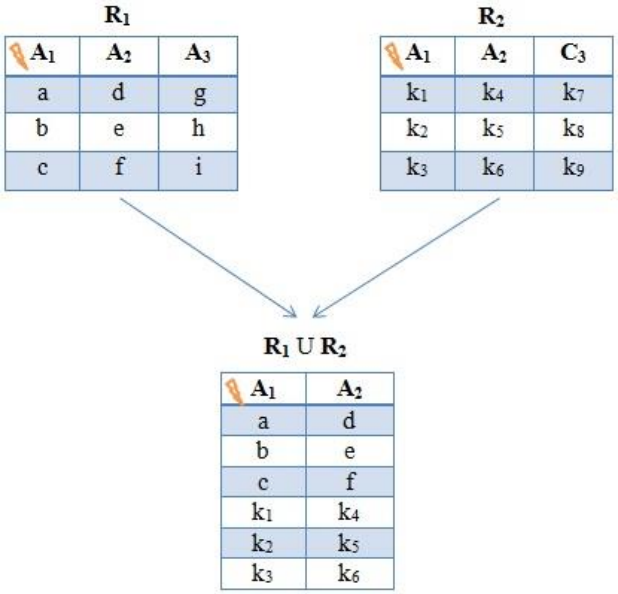
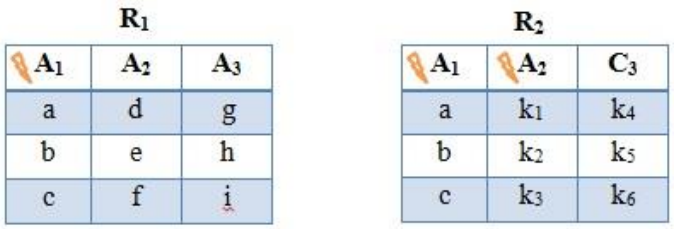


Рисунок 3.11 – Об'єднання відносин з атрибутами, що частково перетинаються, які містять різні дані

РозглянемовідносиниR$_1$ (A$_1$, A$_2$, A$_3$) iR$_2$ (A$_1$, A$_2$, C$_3$). Відношення R$_2$ містить подвійний складовий ключ (рисунок 3.12). У такому разі необхідно перевірити наявність зв'язку між атрибутами цих відносин:

а) за наявності зв'язку, інтеграція не потрібна;

б) за відсутності зв'язку, інтеграція проводиться лише експертом.



R$_1$

A$_1$	A$_2$	A$_3$
a	d	g
b	e	h
c	f	i

R$_2$

A$_1$	A$_2$	C$_3$
a	k$_1$	k$_4$
b	k$_2$	k$_5$
c	k$_3$	k$_6$

Рисунок 3.12 – Відношення з подвійним складовим ключем

Варіант 3. Предметні галузі перетинаються, всі відносини однієї БД містять дані про той самий об'єкт, що і відносини іншої (усі або деякі).

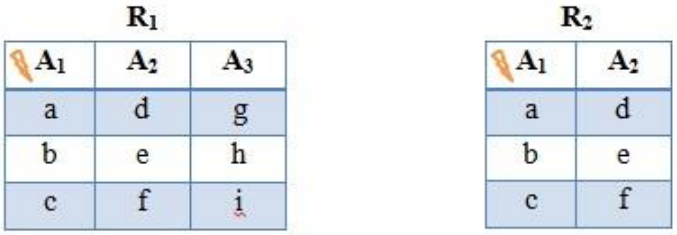
Як і в другому варіанті, можливі три випадки:

а) відносини R$_1$ та R$_2$ мають однакові дані;

б) відносини R$_1$ та R$_2$ мають загальні дані;

в) відносини R$_1$ та R$_2$ мають різні дані.

На рисунку 3.13 зображені відносини R$_1$ (A$_1$, A$_2$, A$_3$) таR$_2$ (A$_1$, A$_2$). Атрибут A$_1$є ключовим в обох стосунках.



R$_1$

A$_1$	A$_2$	A$_3$
a	d	g
b	e	h
c	f	i

R$_2$

A$_1$	A$_2$
a	d
b	e
c	f

Рисунок 3.13 – Відносини з однаковими атрибутами, які містять однакові дані

Для даного випадку введемо перевірку на обчислення потужності відносин R_1і R_2: Dim (R_1)→($\underline{A_1}$, A_2, A_3), Dim (R_2) ($\underline{A_1}$, A_2).. На рисунку 3.14 наведено приклад, коли потужність відношення R_1>R_2. Тоді при об'єднанні цих двох відносин отримуємо таку ж схему як і R_1.

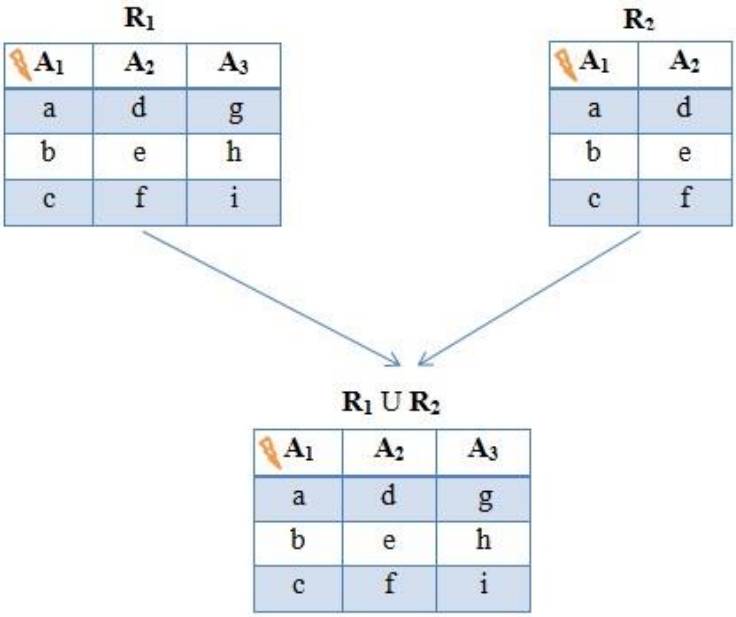


Рисунок 3.14 – Об'єднання відносин з однаковими атрибутами, що містять однакові дані

РозглянемовідносиниR$_1$ ($\underline{A_1}$, A_2, A_3) iR$_2$ ($\underline{A_1}$, A_2)..АтрибутиA$_1$увідносинахєключовимиатрибутами. Відносинимістятьзагальнідані (рисунок 3.15).

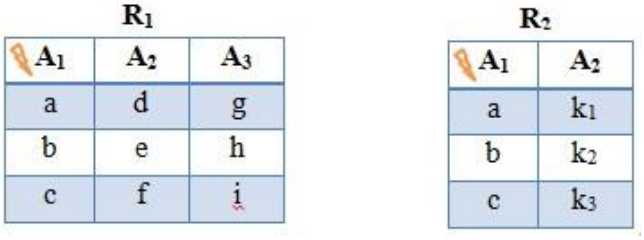


Рисунок 3.15 – Відносини з однаковими атрибутами, які містять загальні дані

Для того, щоб поєднати схеми R_1 іR_2, визначимо які з атрибутів містять однакові дані. У разі це атрибут A_1. Таким чином, при інтеграції схем, атрибут A_1 буде доданий до загальної таблиці і залишиться ключовим. Атрибут A_2 знаходиться в обох відносинах, але містить різні дані, тому для відношення R_1 перейменуємо $A_2=A_2^1$, а щодо R_2, атрибут $A_2 =A_2^2$, і далі додаємо в таблицю атрибути A_2^1 и A_2^2. Атрибут A_3 переносимо до таблиці без змін (рисунок 3.16).

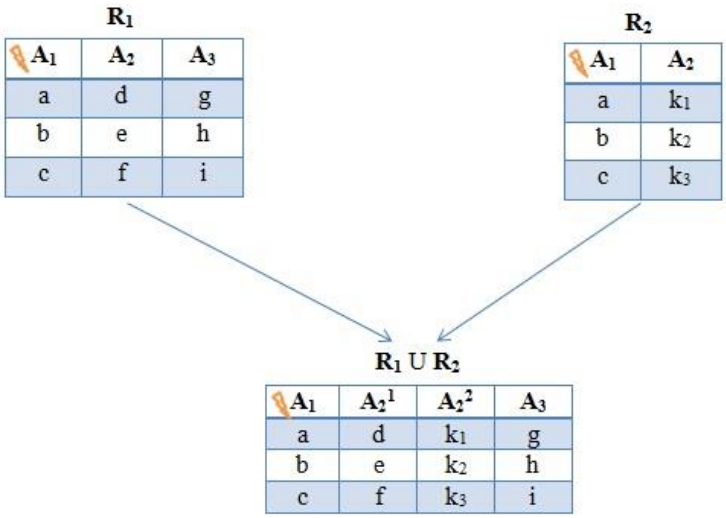


Рисунок 3.16 – Об'єднаннявідносинзоднаковимиатрибутами, щомістятьзагальнідані

Нарисунку 3.17зображенівідносиниR1 (A1, A2, A3) iR2 (A1, A2). АтрибутиA1увідносинахєключовимиатрибутами. Атрибутицихвідносинмістятьрізнідані.

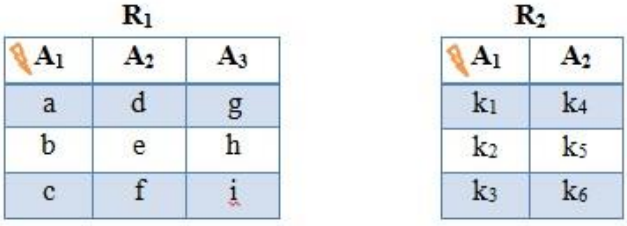


Рисунок 3.17 – Відносини з однаковими атрибутами, які містять різні дані

Для об'єднання даних у цьому випадку підсумкове відношення буде містити атрибути A_1іA_2 , а дані відносин R_1іR_2об'єднуються (рисунок 3.18).

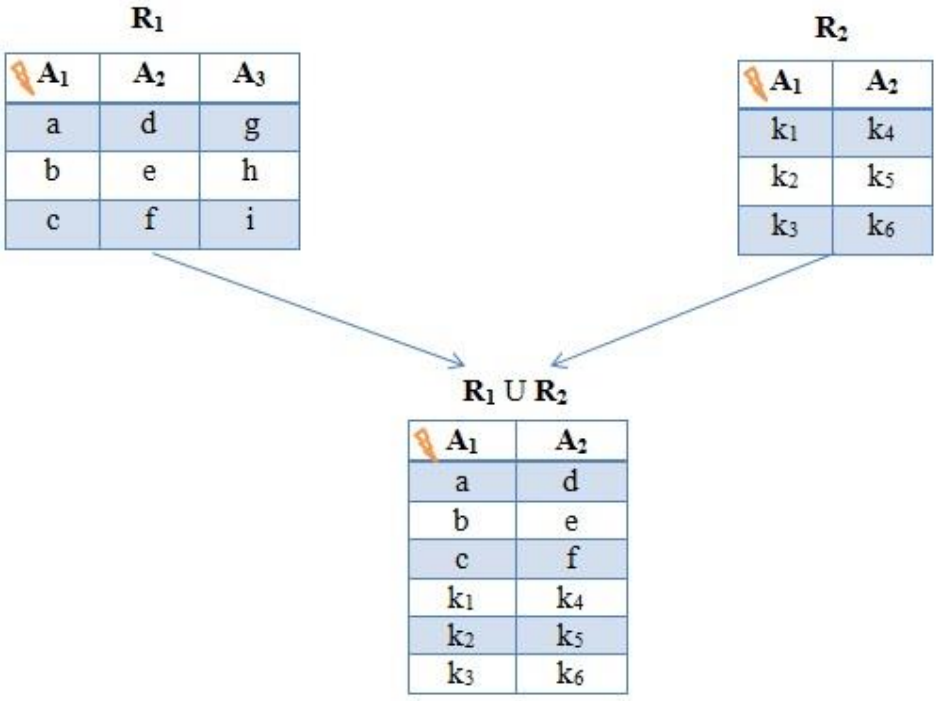


Рисунок 3.18 – Об'єднання відносин з однаковими атрибутами, що містять різні дані

Приватним випадком третього варіанта може бути повна відповідність $S_{ПР1}$ $S_{ПР2}$. Таке можливе, наприклад, у разі впровадження нового варіанта системи [13]. У такому разі питання про інтеграцію даних є особливо важливим.

Для інтеграції різних схем баз даних різнорідних ІС, необхідно перетворити їх схеми на одну, із збереженням даних. Для цього пропонується метод синтезу інтегрованої реляційної моделі даних:$SRM_s = \{R_s, D_s, A_s, dom_s\}$з безлічі окремих схем баз даних – $SRM_1 = \{R_1, D_1, A_1, dom_1\}, SRM_2 = \{R_2, D_2, A_2, dom_2\}, ..., SRM_n = \{R_n, D_n, A_n, dom_n\}$.

Даний метод розроблено для другого та третього варіантів розглянутих вище. Метод дає змогу синтезувати структуру інтегрованої реляційної моделі

даних [14].

Етапи реалізації методу:

а) визначити рівень нормалізації існуючих схем БД;

б) якщо необхідно, привести їх до другої нормальної форми, використовуючи відомі алгоритми нормалізації;

в) визначити множини відносин$R_1, R_2 ... \mathrm{R}_N$, встановити відповідність$R_1 \leftrightarrow R_2$,$\mathrm{R}_1 \leftrightarrow R_n$,$\mathrm{R}_2 \leftrightarrow R_n$...(які з відносин перетинаються чи можуть розцінюватися як тотожні), якщо необхідно, зробити перейменування імен відносин. Сформувати безліч$R_s = R_1 \cup R_2 \cup ... \cup R_n$. Визначення повної відповідності відбувається на підставі експертних оцінок. Експерт повинен добре знати предметну область, існуючий бізнес-процес та схеми БД, з якими доведеться працювати. Необхідно визначити семантичне навантаження імен відносин і, за необхідності, зробити їх транслітерацію;

г) визначити безліч доменів$D_1, D_2 ... \mathrm{D}_N$та встановити відповідність$D_1 \leftrightarrow D_2$,$\mathrm{D}_1 \leftrightarrow D_n$,$\mathrm{D}_2 \leftrightarrow D_n$...Визначити можливі сторонні елементи D_{is} множин$D_1, D_2 ... \mathrm{D}_N$для їх подальшого виключення з D_s. На даному етапі також необхідно визначити семантичне навантаження множини доменів$D_1, D_2 ... \mathrm{D}_N$, провести їх транслітерацію;

д) визначити «застарілі» елементи множин$D_1, D_2 ... \mathrm{D}_N$(які не можуть бути в поточному бізнес-процесі, але були присутніми в минулому), перетворити їх у безліч D* - в SRM_s безліч доменів, що визначає «історичну» складову в безлічі$D_1, D_2 ... \mathrm{D}_N$.

Виділити єдину множину $\mathrm{D}^{i'} = \mathrm{D}_1 \cup \mathrm{D}_2 \cup ... \cup \mathrm{D}_n \mid \mathrm{D}^* \mid \mathrm{D}_{is}$.

Результуюча безліч доменів складається з об'єднання безлічі фактичних доменів усіх схем та безлічі «історичних» доменів $\mathrm{D}_s = \mathrm{D}^i \cup \mathrm{D}^*$.

е) Провести оцінку безлічі схем відносин R_1 іR_2, визначити множини атрибутів$A_1, A_2 ... \mathrm{A}_N$, з урахуванням проведеного аналізу множини доменів, при необхідності, виключити надлишкові атрибути в ході формування підсумкової

множини$A_s = A_1 \cup A_2 \cup ... \cup A_n | A_{\text{is}}$.

На даному етапі необхідно визначити семантичне навантаження множини імен атрибутів $A_1, A_2 ... \text{A}_N$, і, при необхідності, провести їхню транслітерацію;

ж) Провести оцінку безлічі схем відносин, визначити можливі галузі значень атрибутів:$\text{dom}_1 : A_1 \to D_1, \text{dom}_2 : A_2 \to D_2, \ldots, \text{dom}_n : A_n \to D_n$з урахуванням сформованих множин D*, Ds, As. З проведеного аналізу сформувати dom_s;

з) отримані результати перетворити на такий вид:$\text{SRM}_s = \{R_s, \text{D}_s, \text{A}_s, \text{dom}_s\}$.

3.5 Особливості інтелектуального аналізу даних під час схемної інтеграції

В основі всіх сучасних інформаційних як локальних, і розподілених систем перебуває база даних. Досвід розробки, впровадження та експлуатації таких систем показав, що найефективнішою структурою, яка відповідає вимогам, як розробника, так і користувача, є реляційна модель даних. Теоретичні основи проектування реляційних баз даних (РБД), запропоновані Коддом, з 1970 року по сьогодні практично не змінилися.

Неодноразово робилися спроби ревізії цієї моделі, заміни її на різні варіації з об'єктно-орієнтованим підходом, проте досвід розробників і час показали – за сукупністю виразних засобів та математичної основи РМД ще довго буде основною моделлю під час проектування БД. При цьому для програмної реалізації та підтримки РБД за останні 10 років були вкладені великі фінансові ресурси: створені та неодноразово модифікувалися системи управління реляційними базами даних (СУБД) різного класу від FoxProта MS ACCESS до ORACLE-8i.

Таким чином, до теперішнього часу склалася така ситуація. Бази даних стали невід'ємною частиною інформаційної підтримки від локальних систем в офісі, до територіально розподілених систем підтримки прийняття рішень, у

яких на сьогодні накопичені терабайти інформації. Цей факт не залишився поза увагою фахівців у галузі інформаційних технологій. Абсолютно логічно у такій ситуації виникла проблема ефективного використання роками накопичуваних та добре структурованих даних з метою отримання якісно нової інформації – знань: закономірностей розвитку предметної галузі, виражених у вигляді законів, тенденцій, таблиць, графіків тощо, що дозволяють планувати, прогнозувати її розвиток [15]. З таких передумов виник сучасний напрямок - вилучення знань з даних (data mining).

Ця область досліджень, динамічно розвиваючись, виділилася в окремий напрямок широкого спектру інформаційних технологій, проте ще раз слід зазначити важливий факт: data mining досліджує методи та технології обробки баз даних, у яких зберігається накопичена внаслідок тривалої експлуатації інформація.

Розглянемо моделі пошуку функціональних асоціативних правил.

Нехай є БД, для доступу до якої реалізований набіртранзакцій $T = \{T_1, ..., T_n\}, D = \{d_1, ..., d_n\}$ – безліч елементів з яких складається з транзакції T, тобто $T_i \subseteq D$ и $\Omega = \{COUNT, SUM, MAX, MIN, AVG, ...\}$ – набір агрегатних функцій. Кожна транзакція є бінарним вектором,де$T_i = 1$, якщо елементd_iприсутній у транзакції та$T_i = 0$в іншому випадку. ТранзакціяT_iмістить набір елементів$X \subseteq D$, якщо$X \subset T_i$. Тоді продукцію виду (3.3) називатимемо функціональним асоціативним правилом, якщо $X \subset D, Y \subset D, X \cap Y = \varnothing$ і $\varpi \in \Omega$.

$$\{P;\ X \Rightarrow \varpi\ (Y)\}, \tag{3.3}$$

деP–умова активізації ядра правила.

Метою аналізу БД є встановлення таких залежностей: якщо у транзакції зустрівся певний набір елементівX, то на підставі цього можна зробити висновок про те, що інший набір елементівYтакож має з'явитися в цій

транзакції. Встановлення таких залежностей дає змогу знаходити прості та інтуїтивно зрозумілі правила.

У загальному вигляді формування асоціативних правил можна у вигляді двох етапів:

– виділення всіх необхідних наборів елементів;

– генерація правил із наборів елементів із застосуванням необхідних функцій.

З іншого боку, якщо схема БД містить кілька зв'язних відносин, то можлива ситуація, коли значень зв'язного атрибуту в даний момент часу невизначено.

Побудуємо систему продукції у вигляді функціональних асоціативних правил (3.4):

$$\begin{gathered} 1.\ \{c \neq \varnothing;\ a\ \&\ b \Rightarrow SUM(c), \\ 2.\ d \neq \varnothing;\ b \Rightarrow COUNT(e), \\ 3.\ e \neq \varnothing;\ c\ \&\ d \Rightarrow AVG(e), \\ 4.\ e \neq \varnothing \&\ d > \text{‘10’};\ d \Rightarrow MIN(e)\} \end{gathered} \qquad (3.4)$$

Покажемо, що правила (3.3) та (3.4) можуть не активізуватися, тобто існує ситуація, коли не виконатись умови продукції.

Зв'язок типу «один до багатьох» визначає однозначну відповідність одному елементу однієї множини нулю, одному або більше елементам іншої множини. Таким чином, якщо функціональне асоціативне правило будується між інформаційними одиницями, що належать різним таблицям, то правило може бути побудоване, але можлива ситуація, коли умова активізації ядра не виконається.

Нехай задані множини $A = \{a_1,...,a_n\}, B = \{b_1,...,b_m\}$ і $C = \{c_1,...,c_k\}$ нехай задані відносини $R_1 \subseteq A \times B$ таких, що b_i не повторюються і $R_2 \subseteq B \times C$, де не повторюються c_i (згідно з ключовими атрибутами, визначеними у схемі вихідної

БД). Запишемо кортежі множин у такому вигляді (3.5).

$$
\begin{array}{ll}
R_1 = \{<(a_1,...,a_n),b_1>\} & R_2 = \{<(b_1,...,b_m),c_1>\} \\
\{<(a_1,...,a_n),b_2>\} & \{<(b_1,...,b_m),c_2>\} \\
\dots & \dots \\
\{<(a_1,...,a_n),b_m>\} & \{<(b_1,...,b_m),c_k>\}
\end{array} \qquad (3.5)
$$

Такий запис показує, що вR_1кожному значенню з$\{b_1,...,b_m\}$може відповідати одне будь-яке значення з$\{a_1,...,a_n\}$, а також уR_2кожному значенню з$\{c_1,...,c_k\}$може відповідати одне будь-яке значення з$\{b_1,...,b_m\}$.

На відміну від логічних моделей уявлення знань, семантичні мережі (СМ) дозволяють успішно структурувати інформацію. Такі моделі дозволяють наочно уявити коло вирішуваних проблем більшою мірою, ніж безліч правил, що належать до них.

Використовуючи позначення функціональних асоціативних правил, під семантичною мережею розумітимемо вираз (3.6)

$$S = (T,\Omega), \qquad (3.6)$$

деT–набіртранзакцій, елементи якого виступають у ролі вершин СМ;

Ω– безліч агрегатних функцій, що представляють відношення між вершинами (дуги графа).

Розглянемо приклад. Нехай дана система функціональних асоціативних правил:

{Quantity ≠∅; Product ⇒ SUM(Quantity)

Price ≠∅; Product ⇒ MIN(Price)

Product ≠∅; Name_Customer ⇒ COUNT(Product)

Quantity ≠∅; Name_Customer, Product ⇒ MAX(Quantity)

Price ≠∅; Material ⇒ MIN(Price)

Quantity≠∅; Price⇒MIN(Quantity)}

Для системи збудуємо семантичну мережу S, представлену рисунку 3.19. Для ідентифікації складових лівих частин ядра правила в мережі використовується позначення ϖ^i (i = $\overline{1,\infty}$), де i – показує відношення складових вершин.

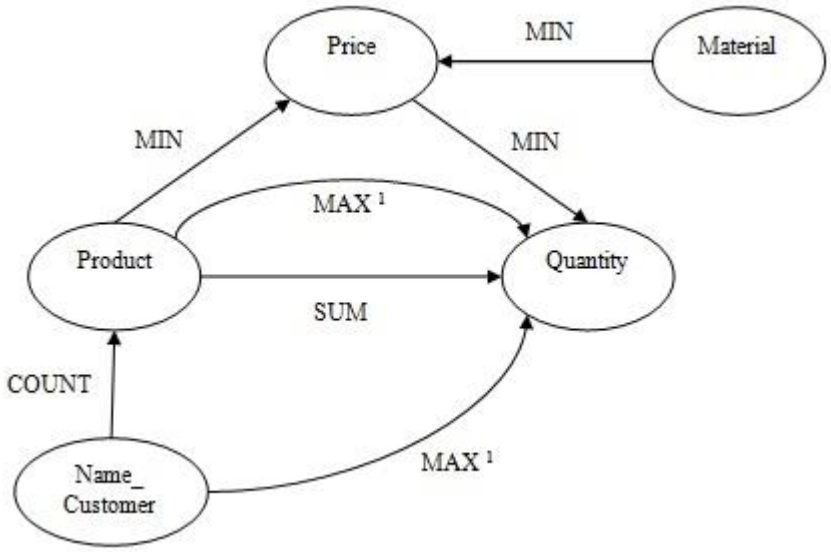


Рисунок 3.19Семантична мережа системи функціональних асоціативних правил

З відомих нині методів аналізу даних слід виділити методи оперативного аналізу баз даних – OLAP-системы. Такі системи базуються на багатовимірних структурах зберігання даних. За типом бази даних OLAP-системи, що використовуються, можуть розділятися на кілька класів залежно від структури зберігання даних:

– системи оперативної аналітичної обробки багатовимірних баз даних (або MOLAP-системи), в яких дані організовані як упорядковані багатовимірні масиви гіперкубів або полікубів;

– системи оперативної аналітичної обробки реляційних баз даних (або ROLAP-системи), які дозволяють подавати дані у багатовимірній формі,

забезпечуючи перетворення інформації на багатовимірну модель через проміжний шар метаданих.

– гібридні системи оперативної аналітичної обробки даних (HOLAP-системи) розроблені з метою поєднання переваг та мінімізації недоліків попередніх систем. Вони поєднують аналітичну гнучкість та швидкість відповіді MOLAP-систем з постійним доступом до реальних даних, властивих ROLAP-систем.

До недоліків розглянутих вище методів оперативного аналізу даних слід віднести головний – практично багатомірність даних, зазвичай, закінчується використанням функцій виділення з атрибутів типу «дата/час» розширень: рік, півріччя, квартал, місяць, тиждень. Всі подальші дії зводяться до послідовності виконання спеціалізованих запитів: «вибірка» та «угруповання» з використанням функцій «CONT», «SUM», «AVG» або інших аналогічних.

Метою наукових досліджень, що проводяться, є розробка методів і моделей інтелектуального аналізу реляційних баз даних. На першому етапі розглядається завдання: на основі формального аналізу схеми бази даних представити підмножину всіх залежних атрибутів відносин моделі даних типу «багато» до «багатьох» N:M, і таким чином відповісти на запитання: які атрибути можуть потенційно зберігати знання про предметну галузь, дані про яку зберігає РБД [16].

Загальний підхід до проектування реляційної бази даних наведено у розділі 3.1.

Концептуально реляційна база є інформаційно-логічною моделлю деякої предметної галузі, такою, що кожне розширення відповідає деякому стану даної галузі в певний момент дискретно поточного часу. Кожен стан моделюється впорядкованою сукупністю значень елементів даних, відповідних значення властивостей об'єктів предметної галузі.

Об'єкт певного типу відповідає кортежу відношення певного типу. Зауважимо, що реляційна модель даних передбачає сильну типізацію об'єктів,

використання цілком певних категорій, таких як тип об'єкта, атрибут (властивість) об'єкта, домен та віднесення кожного значення та впорядкованої сукупності значень однієї з цих категорій. Об'єкти певного типу мають певний набір властивостей, що задається в реляційній моделі схемою відношення.

На етапі проектування розробник виконує обов'язкову процедуру – нормалізацію схеми реляційної бази даних до однієї з нормальних форм, зазвичай це третя нормальна форма (3НФ). Наведемо алгоритм нормалізації універсального відношення.

Етап 1. На першому етапі задається одне або кілька універсальних відносин, що відображають поняття предметної галузі. Відповідно до предметної галузі виділяються функціональні залежності. Якщо відносини задовольняють умовам: немає однакових кортежів, кортежі не впорядковані і значення атрибутів є атомарними, всі відносини автоматично перебувають у 1НФ.

Етап 2. Якщо в деяких відносинах виявлено залежність атрибутів від частини складного ключа, то проводиться декомпозицію цих відносин на кілька відносин таким чином: атрибути, які залежать від частини складного ключа виносяться в окреме відношення разом з цією частиною ключа. У вихідному відношенні залишаються всі ключові атрибути:

Вихідне відношення $\mathrm{R}(\mathrm{K}_1, \mathrm{K}_2, \mathrm{A}_1,..., \mathrm{A}_\mathrm{n}, \mathrm{B}_1,..., \mathrm{B}_\mathrm{m})$;

Ключ$\{K_1,\mathrm{K}_2\}$– складний.

Функціональні залежності:

$\{K_1,\mathrm{K}_2\} \to (A_1,...,\mathrm{A}_n,\mathrm{B}_1,...,\mathrm{B}_m)$ –залежність всіх атрибутів від ключа відношення;

$\{K_1\} \to (A_1,...,\mathrm{A}_n)$ –залежність деяких атрибутів від частини складного ключа.

Декомпозовані відносини:

$R_1(K_1,\mathrm{K}_2,\mathrm{B}_1,...,\mathrm{B}_m)$ –залишок від вихідного відношення. Ключ$\{K_1,\mathrm{K}_2\}$.

$R_2(K_1,A_1,...,A_n)$– атрибути, винесені із вихідного відношення разом із частиною складного ключа. Ключ$\{K_1\}$.

Етап 3. Якщо у відносинах виявлено залежність неключових атрибутів від інших неключових атрибутів, то проводиться декомпозиція цих відносин: ті неключові атрибути, які залежать від інших неключових атрибутів, виносяться в окреме відношення. У новому відношенні ключем стає детермінант функціональної залежності:

Вихідневідношення$R(K,A_1,...,A_n,B_1,...,B_m)$;

Ключ$\{K\}$.

Функціональні залежності:

$\{K\} \to (A_1,...,A_n,B_1,...,B_m)$ –залежність всіх атрибутів від ключа відношення;

$\{A_1,...,A_n\} \to \{B_1,...,B_m\}$залежність неключових атрибутів інших неключових атрибутів.

Декомпозовані відносини:

$R_1(K,A_1,...,A_n)$ – залишок від вихідного відношення. Ключ$\{K\}$.

$R_2(A_1,...,A_n,B_1,...,B_m)$– атрибути, винесені із вихідного відношення разом із детермінантом функціональної залежності. Ключ$\{A_1,...,A_n\}$.

Технологія нормалізації дозволяє вирішити кілька завдань: позбутися надмірності даних і уникнути проблем відомих як «аномалії» – оновлення, видалення та додавання. У цьому теорія функціональних залежностей у процесі нормалізації використовується як критерій ступеня нормалізації. Чим більше функціонально-залежних атрибутів щодо тем, з погляду, теорії проектування реляційних баз даних, «гірше» за проект. Проте, з погляду інтелектуального аналізу даних, дуже нормалізоване відношення не становить цінності як джерело знань. Для підтвердження цього твердження розглянемо визначення функціонально залежних атрибутів.

НехайR– відношення. Безліч атрибутівYфункціонально в залежності від

безлічі атрибутівX(Xфункціонально визначаєY) тоді і лише тоді, коли для будь-якого стану відношення R для будь-яких кортежівr_1,$r_2 \in R$з того, що$r_1 X = r_2 X$випливає, що$r_1 Y = r_2 Y$. Символічно функціональна залежність записується$X \to Y$.

З визначення слідує: у всіх кортежах, що мають однакові значення атрибутів X, значення атрибутівYтакож збігаються у будь-якому стані відносиниR. Таким чином, наявність функціонально-залежних атрибутів щодо (у тому числі і транзитивно-залежних, тобто повторюваних груп даних) – формальна основа пошуку закономірностей групових властивостей даних.

Сформулюємо завдання інтелектуального аналізу даних стосовно реляційної моделі [17].

Під завданням інтелектуального аналізу даних розумітимемо процедуру пошуку всіх пар атрибутів реляційної бази даних, що задовольняють умові$X \to Y$для яких здійснені групові операції.

Вирішити поставлене завдання можна принаймні двома способами:

а) на основі правил теорії нормалізації, аналізуючи другу та третю нормальні форми, вибрати всі пари функціонально-залежних атрибутів$X \to Y$з відношення$R(K_1,K_2,A_1,...,A_n,B_1,...,B_m)$. До списку пар також увійдуть атрибути, транзитивно залежні між собою через відносини, в яких як ключові атрибути використовується складовий ключ.

б) з урахуванням аналізу схеми бази даних, нормалізованої до третьої нормальної форми [18].

Цей варіант формування функціонально-залежних атрибутів для розв'язання задач інтелектуального аналізу даних розглянемо на прикладі фрагмента реляційної моделі даних ІС факультету закладу вищої освіти.

Нехай надано універсальне відношення:Студенти_Групи_ Спеціальності» (рисунок 3.20).

Відношення «Студенти_Групи_Спеціальності» знаходиться в 1НФ, тому що немає однакових кортежів, кортежі не впорядковані і значення атрибутів є

атомарними. Як потенційний ключ відношення можна вибрати атрибути «Код_Студента», «Н_Групи». Розглянуте відношення не знаходиться у 2НФ, тому що. існують атрибути, що залежать від частини складного ключа: «Код_Студента» → «Прізвище», «Код_Студента» → «Група», «Код_Студента» → «Факультет», «Н_Групи» → «Спеціальність».

			Студенты_Группы_Специальности			
Код Студента	Фамилия	Факульте	Телефон	**Н Группы**	Специальность	Код Спец
1	Иванов	КН	380503305445	ИТШИ-20-1	Компьютерные науки	122
1	Иванов	ИТМ	380503405445	ПМ-20-1	Прикладная математика	113
2	Александров	КН	380503565445	ИТШИ-20-1	Компьютерные науки	122
3	Борисов	КИУ	380503305478	КИУКИ-20-1	Компьютерная инженер	123
3	Борисов	ИТМ	380503359445	ПМ-20-1	Прикладная математика	113

Рисунок 3.20 – Універсальне відношення «Студенти_Групи_ Спеціальності»

Декомпозуємо відношення «Студенти_Групи_Спеціальності» на три відносини – «Студенти_Факультети_Групи», «Студенти», «Групи_Спеціальності» (рисунок 3.21).

Студенты	
Код Студента	Фамилия
1	Иванов
2	Александров
3	Борисов

	Группы_Специальности	
Н Группы	Специальность	Код Спец
ИТШИ-20-1	Компьютерные науки	122
ПМ-20-1	Прикладная математи	113
КИУКИ-20-1	Компьютерная инжене	123

		Студенты_Факультеты_Группы		
Код Студента	Фамилия	Факультет	Телефон	**Н Группы**
1	Иванов	КН	380503305445	ИТШИ-20-1
1	Иванов	ИТМ	380503405445	ПМ-20-1
2	Александров	КН	380503565445	ИТШИ-20-1
3	Борисов	КИУ	380503305478	КИУКИ-20-1
3	Борисов	ИТМ	380503359445	ПМ-20-1

Рисунок 3.21 – Декомпозиція відносини «Студенти_Групи_Спеціальності»

Відношення «Студенти_Факультети_Групи» не знаходиться в 3НФ, тому

що є функціональна залежність неключових атрибутів «Факультет» → «Телефон».

Декомпозуємо відношення «Студенти_Факультети_Групи» на два відносини – «Студенти_Групи», «Телефони» (рисунок 3.22).

Студенти_Группы

Код Студента	Факультет	**Н Группы**
1	КН	ИТШИ-20-1
1	ИТМ	ПМ-20-1
2	КН	ИТШИ-20-1
3	КИУ	КИУКИ-20-1
3	ИТМ	ПМ-20-1

Телефоны

Факультет	Телефон
КН	380503305445
ИТМ	380503405445
КН	380503565445
КИУ	380503305478
ИТМ	380503359445

Рисунок 3.22 – Декомпозиція відносини «Студенти_Факультети_Групи»

Схема реляційної бази даних «Студенти_Групи_Спеціальності» представлена на рисунку 3.23.

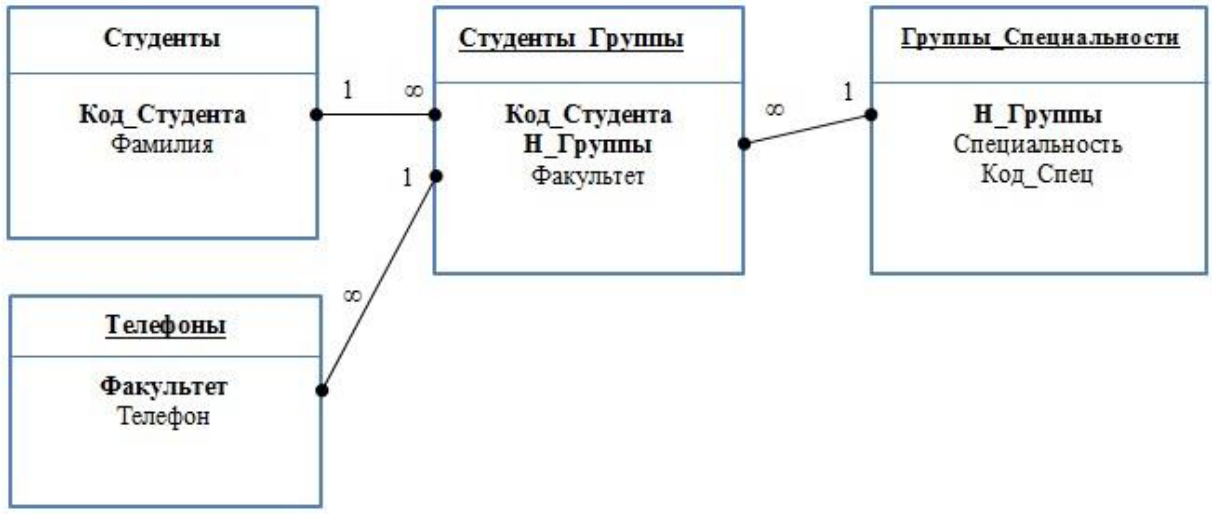


Рисунок 3.23 – Схема РБД «Студенти_Групи_ Спеціальності»

В результаті аналізу представленої вище схеми бази даних «Студенти_Групи», можна виділити наступні залежності та вказати їхній інформаційний зміст:

а) «Код_ Студента» → «Н_Групи» – у якій кількості груп присутній кожен студент;

б) «Код_ Студента» → «Факутьтет» – скільки студентів навчаються у кожному факультеті;

в) «Код_ Студента» → «Н_Групи» – скільки студентів у кожній групі;

г) «Код_ Студента» → «Код_Спеціальності» (через запит на об'єднання) – скільки студентів навчаються за кожною спеціальністю;

д) «Факультети» → «Н_Групи» – скільки навчальних груп на кожному факультеті.

Вибрані залежності та зв'язки дозволяють здійснювати оперативний аналіз даних та отримувати нові знання про предметну область [19].

ВИСНОВКИ

У ході виконання кваліфікаційної роботи запропоновано метод структурної інтеграції схем незалежно спроектованих та функціонуючих баз даних. Для повного перетворення декількох різних реляційних моделей даних в єдину, розглянуто операційну складову та компоненти обмежень цілісності. У розглянутому випадку досліджено технологію перетворення декількох моделей в одну, операційна складова не розглядалася, зважаючи на те, що всі операції синтезу обмежені реляційною моделлю даних. Синтезована реляційна модель даних володітиме тим же набором операцій, що і кожна з вихідних моделей. Прикладами таких операцій можуть бути базові операції реляційної алгебри та похідні від них.

Запропонований метод на формальному рівні дозволяє описати процес синтезу структурної складової реляційної моделі бази даних з безлічі схем баз даних, що окремо існують. Метод дозволяє мінімізувати витрати на проектування загальної схеми БД, забезпечити зберігання даних та маніпулювання ними для всіх функціональних завдань у рамках інтегрованої ІС. Використання інтегрованої ІС знижує ризики подання неактуальної, надмірної інформації, а також вартість її підтримки та супроводу.

На основі аналізу особливостей проектування реляційної моделі даних розглянуто основні проблеми побудови інтегрованих інформаційних систем, що об'єднують у собі реляційні бази даних як джерело первинних даних та засоби інтелектуального аналізу, що дозволяють отримувати знання з даних, прогнозувати та планувати розвиток предметної галузі.

У роботі досліджено функціональні залежності атрибутів та запропоновано варіанти вибору атрибутів, інформаційна значущість яких може використовуватися для інтелектуального аналізу даних.Наведено приклад формування функціонально залежних атрибутів для вирішення задачі інтелектуального аналізу даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бази даних: досягнення та перспективи на порозі 21 століття Електрон. текст. дано. Режим доступу: http://citforum.ru/database/classics/nfs_report - Загл. з екрану.(Дата звернення 02.05.2022)
2. Дейт, К. Введення в системи баз даних К. Дейт. М.: Видавничий дім «Вільямс», 2001. 1072 с.
3. Арсеньєв, Б.П. Інтеграція розподілених баз даних Б.П. Арсеньєв. СПб.: Лань, 2001. 464с.
4. Розподілені БД. Інтегровані та мультибази даних Електрон. текст. дано. Режим доступу: http://edu-knigi.ru/tikunov/geoinformatika.php?id=29 - Загл. з екрану.(Дата звернення 02.05.2022)
5. Інформаційні технології економіки Электрон. текст. дано. Режим доступу: http://abc.vvsu.ru/Books/inform_tehnolog/page0006.asp Загл. з екрану.(Дата звернення 02.05.2022)
6. Stein, RE Re-Engineering the Manufacturing System: Використання Theory of Constraints RE Stein. Marcel Dekker, 2003. 325p.
7. Codd EF A Relational Model of Data for LargeShared Data Banks EFCodd. Communications of the ACM, 1983. Vol.38, №1. С. 17-36.
8. Роб, П. Системи баз даних: проектування, реалізація управління П. Роб, К. Коронел. СПб. : Вид-во БХВ, 2004. 1040 с.
9. Мейєр, Д. Теорія реляційних баз даних Д. Мейєр. М.: Світ, 1987. 608 с.
10. Харрінгтон, Дж.Л. Проектування реляційних БД Дж.Л.Харрінгтон. М.: Лорі, 2000. 230 с.
11. Теорія нормалізації відносинЕлектрон. текст. дано. – режим доступу: http://globalteka.ru/news/1-latest-news/185-2009-02-05-18-02-22.html Загл. з екрану.(Дата звернення 02.05.2022)
12. Козлова, А.Є. Інтеграція баз даних у завданнях реінжинірингу розподілених інформаційних систем О.Є. Козлова 19-й Міжнародний

форум «Радіоелектроніка та молодь у XXI столітті». Зб. матеріалів форуму. Т.6. Харків: ХНУРЕ, 2015. 430.с.

13. Пономаренко, Л.А. Побудова оптимальної послідовності з'єднання відносин у запитах реляційної бази даних Л.А. Пономаренко, С.С. Танянський, В.А. Філатов. Системні дослідження та інформаційні технології. Міжнародний науково-технічний журнал. 2003. - № 2. С. 53-58.
14. Цикрітзіс, Д. Моделі данихД. Цикрітзіс, Ф. Лоховські. М.: Фінанси та статистика, 1985.344 с.
15. Саймон, А.Р. Стратегічні технології баз данихА.Р. Саймон.М.: Фінанси та статистика, 1999.479 с.
16. Коннолі, Т. Бази даних: проектування, реалізація та супровідТ. Коннолі, К. Бегг.М: «Вільямс», 2000.1120 с.
17. Data Mining – інтелектуальний аналіз данихЕлектрон. текст. дано. Режим доступу: http://www.inftech.webservis.ru/it/database/datamining/ar2.html - Загл. з екрану.(Дата звернення 02.05.2022)
18. Корнєєв, В.В Бази даних. Інтелектуальна обробка інформації В.В Корнєєв, А.Ф. Гарєєв, С.В. Васютін, В.В. Райх. М.: Нолідж, 2001.496 с.
19. Яковлєв Д.А. Методи інтеграції розподілених баз даних інтелектуальних систем / Д. А. Яковлєв, В. О. Філатов. // Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління : тези доп. 12-ї міжнар. наук.-техн. конф., 27-28 квітня 2022 р., Баку–Харків–Жиліна : Т. 2 – Харків : Петров В.В., 2022. – С 20.
20. Филатов, В.А. Методы и средства проектирования информационных систем и распределенных баз данных / В.А. Филатов, Р.В. Семенец. // Вестник Херсонского национального технического университета № 4(27) – 2007. – С. 203-207.
21. Filatov, V. Model of semantic integration of information systems properties in relay database reengineering problems / O. Avrunin, O. Vlasov, V. Filatov. //

Innovative Technologies and Scientific Solutions for Industries, 2020 - (4 (14)), 5-12. doi:10.30837/itssi.2020.14.005

22. Filatov V., Semenets V. Methods for Synthesis of Relational Data Model in Information Systems Reengineering Problems // International Scientific-Practical Conference «Problems of Infocommunications. Science and Technology» (PIC S&T-2018) Kharkiv, 2018. doi: 10.1109/INFOCOMMST.2018.8632144
23. Filatov V., Voloshchuk O., Spivak N. Implementation and support fuzzy systems by means the relational data model, «European Cooperation», Vol 4, No 11 (2016). – P. 49-61.
24. Doskalenko S. N. On the Approach to Searching for Functional Dependences of Data in Relational Systems / S. N. Doskalenko, V. A. Filatov. // Innovative technologies and scientific solutions for industries. Kharkiv. 2018. No. 3 (1). P. 54–58. doi:10.30837/2522-9818.2018.3.054
25. Filatov V. Fuzzy models presentation and realization by means of relational systems // Econtechmod : an international quarterly journal on economics in technology, new technologies and modelling processes. – Lublin ; Rzeszow, 2014. – Vol.(3), № 3. – P. 99-102.
26. Филатов В.А. Модель мультиагентной системы автономного администрирования информационных систем и распределенных баз данных / В.А. Филатов, Е.Е. Цыбульник, Л.Э. Чалая // Новости искусственного интеллекта. – 2002. – № 4. – С. 620-627.