

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

Т.О. Паромова

ТЕКСТИ (конспект) лекцій з дисципліни

Організація баз даних
для студентів спеціальності 123 “Комп’ютерна інженерія”
усіх форм навчання
частина I

2019

Тексти (конспект) лекцій з дисципліни “Організація баз даних” для студентів спеціальності 123 “Комп’ютерна інженерія” усіх форм навчання частина I /Укл.: Т.О. Паромова. - Запоріжжя: ЗНТУ, 2019. – 46 с.

Укладачі: Т.О. Паромова, ст. викладач

Рецензент: С.Д. Точилін , доцент, к.ф.-м.н

Відповідальний
за випуск: Т.О. Паромова, ст. викладач

Затверджено
на засіданні кафедри
“Комп’ютерні системи
та мережі ”
Протокол № 7
від “21” січня 2019.

Рекомендовано до видання
НМК факультету комп’ютерних
наук і технологій
Протокол № 6
від “31” січня 2019.

ЗМІСТ

Вступ	4
Історія виникнення баз даних та систем управління базами даних	4
Основні поняття і визначення	5
Архітектура бази даних	6
ТЕМА 1. Моделі організації даних	9
Основні поняття і визначення	9
Ієрархічна модель даних.....	9
Мережева модель даних	13
ТЕМА 2. Реляційна модель	16
Основні поняття і визначення	17
Ключі відношення	18
Види відносин.....	20
Реляційна схема даних	21
Схема бази даних.....	21
Цілісність даних.....	22
ТЕМА 3. Нормалізація відношень	24
Загальні поняття нормалізації	24
Надмірність даних і аномалії модифікації	24
Функціональні залежності.....	26
Багатозначна залежність.....	27
Перша нормальна форма	28
Друга нормальна форма.	36
Третя нормальна форма.....	39
Нормальна форма Бойса-Кодда.....	40
Четверта нормальна форма.....	42
П'ята нормальна форма.....	44
Послідовність проведення нормалізації.....	45
Перелік джерел посилання	45

ВСТУП

Історія виникнення баз даних та систем управління базами даних

З самого початку розвитку обчислювальної техніки утворилися два основні напрями її використання. Перший напрям - застосування обчислювальної техніки для виконання чисельних розрахунків, які дуже довго або взагалі неможливо проводити уручну. Становлення цього напрямку сприяло інтенсифікації розвитку методів чисельного рішення складних математичних задач, розвитку мов програмування, орієнтованих на зручний запис чисельних алгоритмів.

Другий напрям - це використання засобів комп'ютерної техніки в автоматизованих інформаційних системах. Автоматизована система є апаратно- програмний комплекс, призначений для накопичення і колективного використання інформації. У інформаційних системах сукупність взаємозв'язаних інформаційних об'єктів фактично відображає модель об'єктів реального миру.

Конкретні завдання, які вирішуються інформаційною системою, залежать від тієї прикладної області, для якої призначена система. Але всі інформаційні системи повинні забезпечити:

- накопичення і надійне довготривале зберігання інформації, яке в значній мірі залежить від призначення самої системи;
- зберігання даних, що мають різну структуру;
- зручний інтерфейс користувача;
- забезпечення доступу до даних, що розрахований на багато користувачів;

– розподілену обробку даних в мережі в корпоративних систем.

У розвитку АІС можна виділити два покоління:

1-е покоління - інформаційні системи, які базуються на автономних файлах. Це системи простої архітектури і обмеженим набором можливостей. Вони складаються з набору автономних файлів і комплексу прикладних програм, призначених для обробки цих файлів та створення документів. Такі системи мають ряд недоліків, які обмежують їх широке застосування:

- високу надмірність даних;
- сильний зв'язок файлів з програмним забезпеченням - файлова система не визначає внутрішній формат інформації у файлі, а визначає тільки її розміщення у внутрішній пам'яті;

- тривалість розробки;
- труднощі організації доступу до даним (час і швидкість вибірки);
- обмеженість сумісного доступу до даним;
- складність ведення і загальної обробки файлів;
- ізольованість даних (багато файлів і форматів).
- складність синхронізації внесення змін в дані - немає підтримки інформаційної цілісності даних;
- труднощі захисту даних від несанкціонованого доступу.

Області застосування таких систем - зберігання неструктурованих даних:

- текстовій інформації (документів, текстів програм, об'єктних модулів);
- графіки;
- мультимедіа і т.д.

2-е покоління - банки даних.

Це системи з високим ступенем інтеграції даних і автоматизації управління ними. Таки системи орієнтовані на зберігання, вибір і модифікацію постійно існуючої інформації. Для зберігання і обробки великих об'ємів інформації необхідні великі об'єми зовнішньої пам'яті, тому розвиток інформаційних систем став можливим після появи пристроїв, здатних зберігати великі об'єми даних, - жорстких дисків і т.д.

У інформаційних системах необхідно забезпечити:

- підтримку логічно зв'язаного набору файлів;
- можливість використання мови маніпулювання даними;
- відновлення інформації після збоїв;
- паралельну роботу декількох користувачів.

Основні поняття і визначення

Банк даних - система організованих даних, призначених для централізованого накопичення і колективного використання

До складу банку даних входять:

- одна або декілька баз даних;

- довідник баз даних, СУБД;
- бібліотеки запитів і прикладних програм.

Дані - сукупність конкретних значень, параметрів, що характеризують об'єкт, явище, інформація про які зберігається в базі даних.

Предметна область - область або явища реального миру, інформація про яких зберігається в базі даних, є джерелом зв'язаних даних при проектуванні. Наприклад, деканат, склад, лікарня.

База даних (БД) - іменована сукупність даних, що відображає стан об'єктів і їх взаємозв'язків в деякій предметній області.

Визначення ДЖ. Мартіна[1], база даних - це сукупність даних і зв'язків між ними. Тоді інформація - дані, витягнуті з БД на основі зв'язків [2].

Система управління базами даних (СУБД) - сукупність мовних і програмних засобів, які використовуються для створення, ведення і сумісного використання БД багатьма користувачами.

Таким чином, база даних є сукупність формалізованої інформації, а СУБД - середовище для зберігання та обробки цих даних.

Архітектура бази даних

Трирівнева модель бази даних запропонована американським комітетом із стандартизації - ANSI.

Зовнішній рівень - додатки, що відображають потреби конкретних користувачів.

Зовнішній рівень складається з декількох різних представлень бази даних. Кожен користувач має справу з уявленням «реального миру» в зручній для нього формі. Зовнішнє уявлення містить тільки ту інформацію, яка необхідна саме цьому користувачеві.

Різні уявлення можуть по-різному відображати одні і ті ж дані з використанням різних форматів, включати похідні і обчислювані дані декількох об'єктів.

Уявлення окремого користувача називається зовнішнім уявленням - зміст БД, який бачить окремий користувач.

Цей рівень визначає уявлення БД окремими додатками. Кожному додатку доступні для відображення та обробки тільки ті дані, які необхідні саме цьому додатку. Наприклад, система розподілу робіт використовує зведення про кваліфікацію співробітника, але в її неї не відображаються відомості про оклад, домашню адресу і телефон

співробітника, і навпаки, саме ці відомості використовуються в підсистемі відділу кадрів.

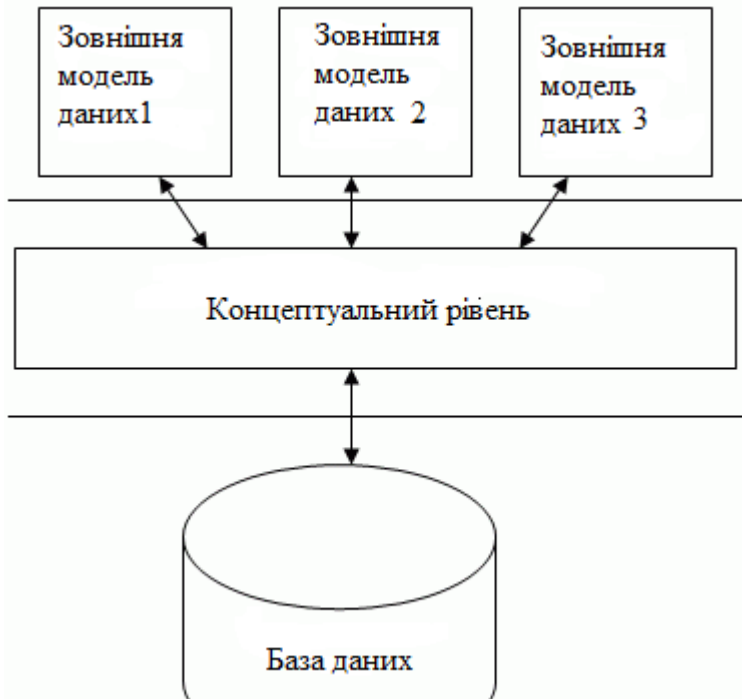


Рисунок 1 Архітектура бази даних

Кожен користувач має свою мову спілкування з БД. Для прикладного програміста це або спеціальна мова даної системи або одна з мов високого рівня. Для кінцевого користувача це спеціальна мова запитів або мова спеціального призначення, заснована на формах, меню, створювана спеціально з урахуванням вимог користувача. На сьогоднішній день мова SQL підтримується всіма СУБД.

Концептуальний рівень - узагальнення уявлень окремих користувачів - узагальнена модель предметної області.

На цьому рівні описуються дані, які взагалі зберігаються в базі даних, і які зв'язки існують між ними. Це повне уявлення вимог до даним, яке не залежить від способу їх зберігання.

На цьому рівні представлені наступні компоненти:

- всі сутності, їхні атрибути та зв'язки між ними;
- обмеження, що накладаються на дані;
- семантична (смислова) інформація про даних;
- інформація про заходи забезпечення безпеки і підтримки цілісності даних.

Виділення концептуального рівня дозволило розробити апарат централізованого управління базою даних.

Внутрішній рівень (фізичний рівень) - дані, збережені на зовнішньому носіїві. На цьому рівні відображається фізичне представлення бази даних в комп'ютері.

Цей рівень описує, як інформація зберігається в базі даних, містить опис структур даних і опис окремих файлів, використовуваних для зберігання даних в пристроях, що запам'ятовують.

На цьому рівні зберігається наступна інформація:

- розподіл дискового простору для зберігання даних і індексів;
- опис подробиць збереження записів з вказівкою реальних розмірів;
- зведення про розміщення записів;
- відомості про стиснення даних і вибраних методах шифрування.

Ця архітектура дозволяє забезпечити логічну (між рівнями 1 і 2) і фізичну (між рівнями 2 і 3) незалежність при роботі з даними.

Логічна незалежність припускає можливість зміни одного додатку без корегування інших додатків, що працюють з цією ж базою даних.

Фізична незалежність припускає можливість перенесення інформації, що зберігається, з одних носіїв на інші при збереженні працездатності всіх додатків, що працюють з даною базою даних.

ТЕМА 1. МОДЕЛІ ОРГАНІЗАЦІЇ ДАНИХ

Основні поняття і визначення

Кожна БД є якоюсь моделлю конкретної предметної області.

Концептуальне (інфологічне) моделювання - перехід від неформалізованого опису предметної області до її формального викладу за допомогою спеціальних мовних засобів.

Така модель є універсальною і ніяк не пов'язана з конкретною реалізацією СУБД.

Модель даних - формалізоване уявлення даних деякої предметної області, яке дозволяє трактувати дані як якусь інформацію.

Моделлю даних (МД) - називається спосіб відображення предметної області на структурі даних.

Модель даних можна розглядати як поєднання трьох компонентів:

- структурна частина, тобто набір правил, по яких може бути побудована база даних;
- маніпуляційна частина, яка визначає типи припустимих операцій з даними;
- набір обмежень, які накладаються на дані та гарантують коректність використовуваних даних.

Ієрархія модель даних

Ієрархічні системи є основою старого покоління систем баз даних. Вони розроблялися для великих ЕОМ.

Ієрархічні СУБД використовуються до цих пір із завдяки великим інвестиціям у вже існуючі системи, високим витратам, необхідним на їх модернізацію, крім того вони до цих пір зберігають працездатність.

Поява ієрархічної моделі пов'язана з тим, що в реальному світі дуже багато зв'язків відповідають ієрархії, коли один об'єкт виступає як батьківський, а з ним може бути пов'язана безліч підлеглих об'єктів. Ієрархія проста і природна у відображенні взаємозв'язку між класами об'єктів.

Ієрархічна БД складається з впорядкованого набору дерев. Такі бази підтримують відношення типу "один-до-багатьох".

У ієрархічній моделі об'єднуються записи, що зберігаються в загальній деревовидній структурі з одним кореневим типом запису,

який має декілька підлеглих записів. Кожен підлеглий тип запису у свою чергу може мати декілька підлеглих типів записів.

Схема ієрархічної БД є сукупністю окремих дерев, кожне дерево в рамках моделі називається фізичною базою даних. Кожна фізична БД задовольняє наступним ієрархічним обмеженням:

- у кожній фізичній БД існує один кореневий вузол, у якого немає логічне початкового (батьківського) типу вузла;
- кожен логічне початковий вузол може бути пов'язаний з довільним числом логічно підлеглих вузлів;
- кожен логічне підлеглий сегмент може бути пов'язаний тільки з одним логічне початковим (батьківським) сегментом;
- між двома вузлами може бути тільки один зв'язок;
- зв'язки в конкретній структурі можуть мати тільки певне впорядкування.

Пошук даних в ієрархічній системі завжди починається з кореня. Потім проводиться спуск з одного рівня на інший, поки не буде досягнутий необхідний рівень. Переміщення по системі від одного запису до іншої здійснюються за допомогою посилань.

Групове відношення - ієрархічне відношення між записами двох типів. Батьківський запис (власник групового відношення) називається початковим записом, а дочірні записи (члени групового відношення) - підлеглими.

Кореневий запис кожного дерева обов'язково повинен містити ключ з унікальним значенням. Ключі некоренових записів повинні мати унікальне значення тільки в рамках групового відношення. Кожен запис ідентифікується повним складеним (зчепленим) ключем, під яким розуміється сукупність ключів всіх записів від кореневої по ієрархічному шляху.

Для групових відносин в ієрархічній моделі забезпечується автоматичний режим включення і фіксоване членство. Це означає, що для запам'ятовування будь-якого некоренового запису в БД повинен існувати її батьківський запис. При видаленні батьківського запису автоматично віддаляються всі підлегли.

Набір всіх екземплярів сегментів, підлеглих одному екземпляру кореневого сегменту, називається фізичним записом. Кількість екземплярів-нащадків може бути різним для різних екземплярів

батьківських сегментів, тому в загальному випадку фізичні записи мають різну довжину.

У узагальненій структурі дерева типи записів зазвичай впорядковані усередині структури зліва направо. Екземпляри записів також впорядковані.

В рамках ієрархічної моделі виділяють мовні кошти опису даних (DDL, Data Definition Language) і засобу маніпулювання даними (DML, Data Manipulation Language).

Типовим представником сімейства баз даних, заснованих на ієрархічній моделі, є Information Management System (IMS) фірми IBM, перша версія якої з'явилася в 1968 р.

Приклад

Підприємство складається з відділів. У кожному відділі може працювати декілька співробітників, але співробітник не може працювати більш ніж в одному відділі.

Тому, для інформаційної системи управління персоналом необхідно створити групове відношення, що складається з батьківського запису ВІДДІЛ (Найменування_відділу, кількість_працівників) і дочірньому запису СПІВРОБІТНИК (прізвище, посада, оклад). Це відношення показане на рисунку 1(а) Для спрощення вважається, що є тільки два дочірні записи.

Для автоматизації обліку контрактів із замовниками необхідне створення ще однієї ієрархічної структури: Замовник - Контракти з ним - Співробітники, задіяні в роботі над контрактом. Це дерево включатиме записи ЗАМОВНИК(Найменування_замовника, адреса), КОНТРАКТ (номер, дата, сума), ВИКОНАВЕЦЬ (прізвище, посада, Найменування_відділу) (рисунку 1(в)).

З цього прикладу видно недоліки ієрархічних БД:

- частково дублюється інформація між записами СПІВРОБІТНИК і ВИКОНАВЕЦЬ (такі записи називають парними), причому в ієрархічній моделі даних не передбачена підтримка відповідності між парними записами.

- ієрархічна модель реалізує відношення між початковим і дочірнім записом по схемі 1:N. Допустимо, що виконавець може брати участь більш ніж в одному контракті (виникає зв'язок типу M:N). В цьому випадку в БД необхідно ввести ще одне групове відношення, в

якому ВИКОНАВЕЦЬ буде початковим записом, а КОНТРАКТ - дочерней (рисунку 1(с)), тобто знову необхідно дублювати інформацію.

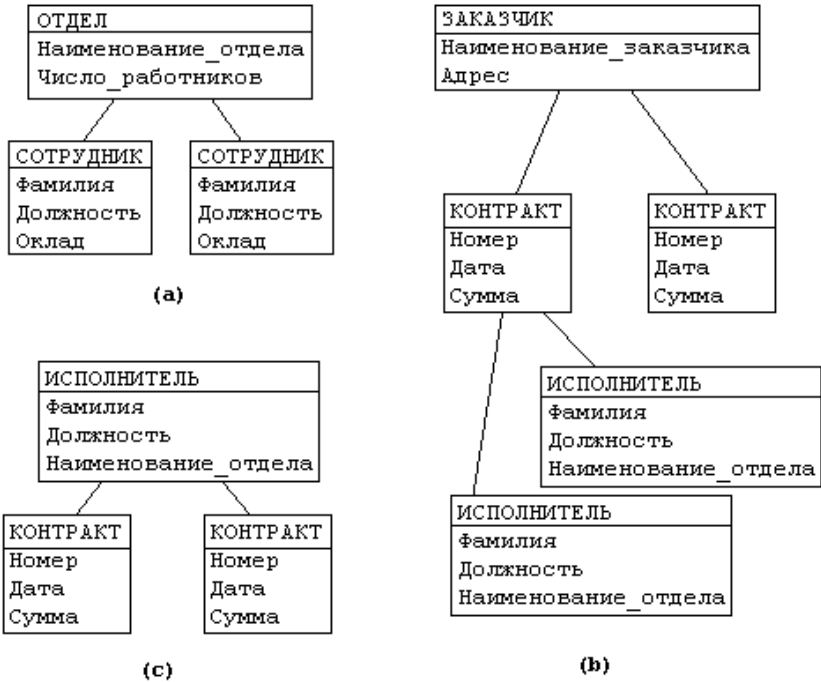


Рисунок. 1.1 — Приклад бази даних за ієрархічною моделлю

Операції над даними, визначені в ієрархічній моделі:

ДОДАТИ в базу даних новий запис. Для кореневого запису обов'язкове формування значення ключа.

ЗМІНИТИ значення даних заздалегідь витягнутого запису. Ключові дані не повинні піддаватися змін.

ВИДАЛИТИ деякий запис і всі підлеглі їй записи.

ВИТЯГТИ:

- витягти кореневий запис по ключовому значенню, допускається також послідовний перегляд корневих записів;
- витягти наступний запис (наступний запис витягується в порядку лівобічного обходу дерева).

У операції **ВИТЯГТИ** допускається завдання умов вибірки (наприклад, витягнути співробітників з окладом більше 1 тисячі крб.)

Обмеження цілісності.

Підтримується тільки цілісність зв'язків між власниками і членами групового відношення (ніякий нащадок не може існувати без батька). Не забезпечується автоматична підтримка відповідності парних записів, що входять в різні ієрархії.

Основні переваги ієрархічної моделі:

- простота опису ієрархічних структур реального миру;
- швидке виконання запитів, відповідних структурі даних;
- цілісність на рівні посилань забезпечується автоматично, де екземпляри записів повністю залежать від інших екземплярів записів.

Основні недоліки ієрархічної моделі

- надмірність даних;
- не підтримуються зв'язки типу "багато - до багатьох", що приводить до необхідності дублювання даних;
- не завжди зручно кожного разу починати пошук потрібних даних з кореня.

Мережева модель даних

Концепція мережевої моделі даних пов'язана з ім'ям Ч. Бахмана. Мережевий підхід до організації даних є розвитком ієрархічного.

У мережевій структурі даних нащадок може мати будь-яке число предків, тобто мережева модель дозволяє відображати записи типу М:М. В результаті можна сформувати модель бази даних з довільними зв'язками між записами різного типу. Мережева БД складається з набору записів і набору зв'язків між цими записами, точніше, з набору екземплярів записів заданих типів (з допустимого набору типів) і набору екземплярів із заданого набору типів зв'язку.

Мережеві структури можна описувати за допомогою батьківських і дочірніх елементів. При розгляді мережевих структур розглядаються рівні як в ієрархічній моделі. Якщо в моделі присутні зв'язки 1:М або М:1, то така структура називається простою. Якщо в моделі є зв'язки М:М, то структура називається складною.

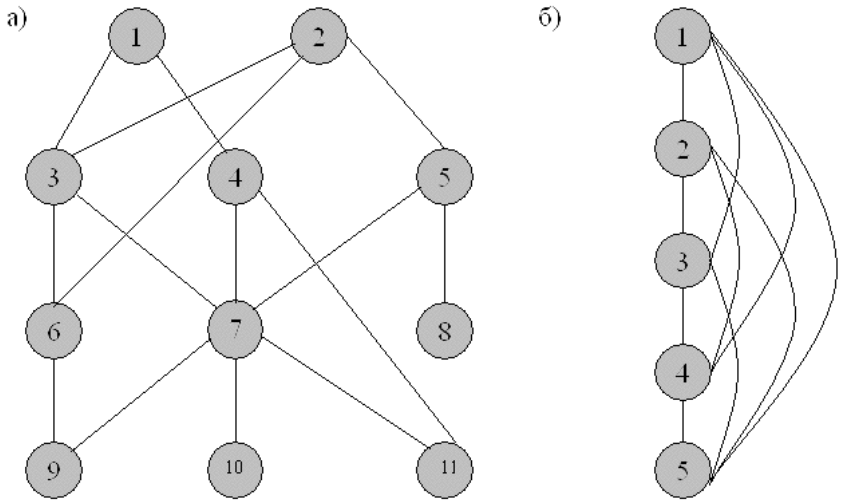


Рисунок 1.2 — Приклад мережевої моделі даних

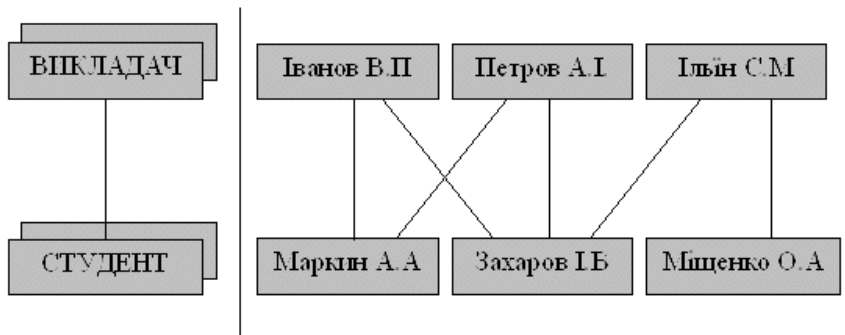


Рисунок 1.3 —Приклад складної мережевої структури даних

Викладач працює з декількома студентами, але з кожним студентом працює декілька викладачів.

Ще приклад мережевої бази даних.

Така база даних складається з двох зв'язаних дерев інформації

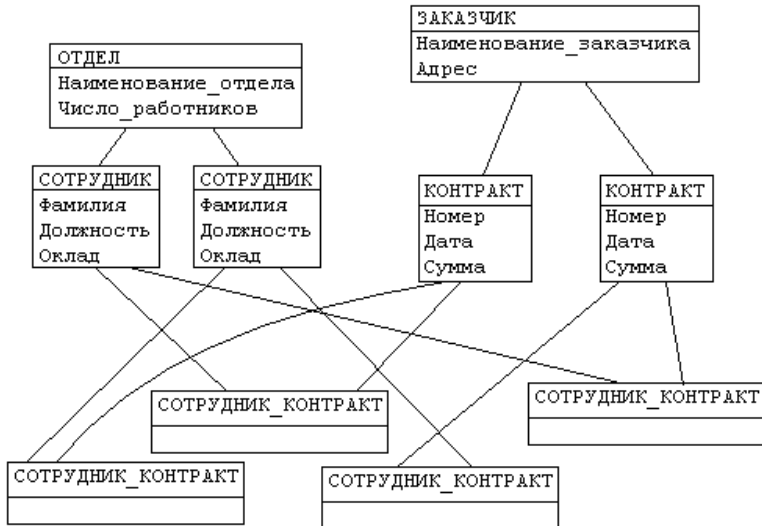


Рисунок 1.4 — Приклад мереженої БД

Умови ті ж, що і в попередньому прикладі, але в даному випадку підтримуються зв'язки М:М, тобто один співробітник може брати участь в багатьох контрактах.

Переваги мереженої моделі даних:

- наявність СУБД, які успішно реалізують таку організацію даних;
- простота реалізації зв'язків "багато до багатьом".

Недоліки мереженої моделі даних

- складність структур у порівнянні з ієрархічною структурою;
- прикладному програмістові часто необхідно знати логічну структуру БД;
- при реорганізації бази даних у ряді випадків можлива втрата незалежності даних.

Прикладом системи управління даними з мереженою організацією є Integrated Database Management System (IDMS) компанії Cullinet Software Inc., розроблена в середині 70-х років. Вона призначена для використання на "великих" обчислювальних машинах.

Переваги СУБД, заснованих на ієрархічній або мережевій моделях:

- компактність;
- як правило, висока швидкодія.

Недоліки СУБД, заснованих на ієрархічній або мережевій моделях:

- призначення кожної БД для використання у конкретній предметній області.
- високий ступінь залежності від конкретних даних;
- складність програмного забезпечення.

ТЕМА 2. РЕЛЯЦІЙНА МОДЕЛЬ

Загальні відомості

Концепції реляційної моделі вперше були сформульовані в роботах американського ученого Е. Ф. Кодда.

Математик Е.Кодд запропонував використовувати для обробки даних апарат теорії множин (об'єднання, перетин, різниця, декартове множення).

Реляційна модель заснована на математичному понятті відношення, фізичним представленням якого є таблиця.

Реляційна база даних є сукупністю відносин, що містять всю інформацію, яка повинна зберігатися в базі даних.

Реляційна модель даних - це модель, в якій всі дані представлені у вигляді таблиць, а всі операції над даними зводяться до операцій над таблицями. При цьому таблиці мають бути тільки двовимірними, в кожному стовпці таблиці повинні розміщуватися тільки однотипні дані.

У загальному виді таблиці взагалі можуть мати абсолютно довільну структуру, наприклад, як у відношенні Відомості про співробітників Така таблиця не відповідає вимогам реляційної моделі і повинна бути перетворена

Відношення Відомості про співробітників

Відділ	Співробітники	Діти співробітників (зацікавленість)		
Цех	Іванов І.І.	Маша	ЛЕГО	
		Петро	Книги	Відео
		Саша	Комп'ютери	
		Діма	Спорт	
	Петров П.П.	Артур	Не має зацікавленість	
	Сидоров С.С.	Сергій	Комп'ютери Книги	
		Валерій	Книги	
		Станіслав	Відео	
	Бухгалтерія	...	...	

Основні поняття і визначення

Відношення - це двовимірна таблиця, що складається із стовпців і рядків. Відношення - це абстрактний об'єкт, таблиця - це конкретне зображення цього об'єкту.

Відношення складається з двох частин - заголовка і тіла. **Заголовок** містить фіксована безліч атрибутів, при цьому кожен атрибут визначений тільки на одному з доменів, які лежать в основі відношення.

Кортеж - рядок у відношенні. Кортеж є сукупність значень всіх атрибутів відношення, відповідних одному рядку.

Сукупність кортежів утворюють **тіло відношення**, яке постійно змінюється. Кортежі можуть розташовуватися у будь-якому порядку, при цьому відношення залишається одним і тим же.

Ступінь відношення визначається кількістю атрибутів, яку воно містить.

Кардинальність відношення (кардинальне число або потужність відношення) характеризує кількість кортежів, яка містить відношення.

Кардинальність постійно змінюється унаслідок додавання і видалення кортежів (записів) у відношенні, на відміну від його ступеня.

Домен - сукупність допустимих значень, яка може приймати атрибут. На одному домені може бути визначені декілька атрибутів.

Наприклад, домен "Імена" визначений на базовому типі «Рядок символів», але в число його значень можуть входити тільки ті рядки, які можуть зображати ім'я (зокрема, такі рядки не можуть починатися з м'якого знаку).

Дані вважаються порівнянними тільки у тому випадку, коли вони відносяться до одного домену. Кожен домен в базі даних має унікальне ім'я.

Атрибут - поименований стовпець відношення, використовується для зберігання інформації одного типу. Атрибут має унікальне для даного відношення ім'я і тип даних. При цьому атрибути можуть розташовуватися у відношенні у будь-якому порядку (якщо поміняти місцями стовпці відношення не поміняється).

Кожен атрибут визначається на деякому домені. Але на одному домені можуть визначатися декілька атрибутів. Наприклад, атрибути `Номер_телефону` і `Номер_факсу` визначаються на домені `Номера_тел_факс`.

Атомарне значення - найменша одиниця даних реляційної моделі - це окреме нерозкладне для даної моделі значення даних. Так, в одній предметній області прізвище, ім'я і по батькові можуть розглядатися як єдине значення, а в іншій - як три різні значення. Конкретне значення атрибуту, відповідне конкретному кортежу, повинне бути атомарно.

Ключі відношення

Серед атрибутів виділяється один атрибут або декілька атрибутів, які є **ключами**. При цьому кожне відношення в реляційній БД обов'язково має хоч би один ключ.

Ключ - мінімальний набір атрибутів, який однозначно ідентифікує кожен кортеж відношення, тобто у будь-який момент часу не існує двох рядків, що містять однакове значення цих атрибутів.

У реляційній моделі розрізняють декілька видів ключів.

Потенційний ключ - атрибут або декілька атрибутів, які єдиним чином ідентифікують кожен кортеж даного відношення, тобто однозначно визначають значення решти всіх атрибутів.

Первинний ключ - будь-який з потенційних ключів, краще якщо до складу ключа входить яку можна менша кількість атрибутів та значення яких є цілим числом.

Зовнішній ключ - атрибут одного відношення, який є ключем іншого відношення. За допомогою зовнішніх ключів встановлюється зв'язку між відносинами



Рисунок 2.1 – Зв'язок між відносинами за допомогою зовнішнього ключа

Ключ відношення Відділ N_відділу є зовнішнім ключем відношення Службовець, для якого первинним ключем є N_службовця відношення Відділ N_відділу є зовнішнім ключем відношення Службовець, для якого первинним ключем є N_службовця.

Багаторазове використання зовнішніх ключів дозволяє усунути надмірність даних

Складений ключ (зчеплений) - ключ відношення, що складається з декількох атрибутів відношення. При цьому один з цих атрибутів може бути зовнішнім ключем.

Властивості ключів:

- унікальність: у будь який момент часу ніякі два різні кортежі не мають одного і того ж значення ключа;
- мінімальність: виключення з ключового набору будь-якого атрибуту не дозволяє однозначно ідентифікувати сутність по атрибутах, що залишилися;
- ключовий атрибут не може приймати значення NULL або невизначене;

Кожне відношення володіє хоч би одним можливим ключем, оскільки хоч би комбінація всіх його атрибутів задовольняє умові унікальності. Один з можливих ключів (вибраний довільним чином) береться за його первинний ключ. Решта потенційних ключів, якщо вони є, називається альтернативними або можливими ключами.

Види відносин.

Базове відношення - відношення, яке зберігається на зовнішньому носіїві, є частиною БД і не може бути виражене через інші відносини.

Похідне відношення визначається через інші відносини. Похідне відношення можна отримати з набору відносин через реляційний вираз.

Властивості відносин:

- відношення має унікальне ім'я, яке відрізняється від імен всіх інших відносин;
- кожна комірка відношення містить атомарне значення;
- у кожній позиції перетину стовпця і рядка розташовано в точності одне значення, а не набір значень, тобто відношення не містить груп, які повторюються. Відносини, що задовольняють цій умові, називаються нормалізованими. Нормалізовані відносини мають простішу структуру, тому робота з такими відносинами простіше;
- кожен атрибут має унікальне ім'я та унікальний в межах даного відношення;
- значення всіх атрибутів беруться з одного і того ж кортежу, тобто значення атрибутів зв'язані.
- у відношення не може бути однакових кортежів - кожен кортеж є унікальним.
- ніякі два кортежі відношення не можуть бути дублікатами один одного в будь-який довільно-заданий момент часу, оскільки відношення - це множина, а множини за визначенням не містять співпадаючих елементів. У цьому виявляється різниця між відношенням і таблицею: таблиця допускає однакові рядки. Важливим слідством того, що у відношенні не може бути однакових кортежів, є те, що завжди існує первинний ключ.
- кортежі не впорядковані зверху вниз - порядок проходження кортежів не має значення

– два співвідношення з різним порядком записів будуть однакові, тому що тіло відношення – це математична множина, а прості множини не впорядковані.

– атрибути не впорядковані зліва на право - порядок проходження кортежів не має значення.

– кожне відношення має хоча би один ключ, оскільки у відношенні не може бути однакових кортежів, то набір атрибутів відношення може бути ключем.

Для зручності роботи з БД слід уникати однакових імен атрибутів в різних відносинах, якщо ці атрибути неключові. Для посилання на значення атрибуту в кортежі відношення завжди використовується ім'я атрибуту. Ця властивість теоретично дозволяє, наприклад, модифікувати схеми існуючих відносин не тільки шляхом додавання нових атрибутів, але і шляхом видалення існуючих атрибутів. Проте в більшості існуючих систем така можливість не допускається, і хоча впорядкованість набору атрибутів відношення явно не потрібна, часто як неявний порядок атрибутів використовується їх порядок в лінійній формі визначення схеми відношення.

Реляційна схема даних

Реляційна схема включає ім'я відношення, за яким в дужках розташовуються імена атрибутів, при цьому першим указується первинний ключ, який підкреслюється.

Наприклад:

Студент (Кодстуд, Прізвище, Ім'я, По батькові, годрожд, Адреса, Телефон)

Склад (Кодтовара, Дата, Кількість, Ціна, Постачальник)

Схема бази даних

Концептуальною моделлю, концептуальною схемою БД називається безліч всіх реляційних схем відносин.

Схемою реляційної бази даних називається набір сукупність схем відносин, що входять в базу даних.

ПРИКЛАД: База даних Деканат

Студенти (Код_Студента, група, Прізвище, Ім'я, По батькові, Курс, Дата народження);

Гуртожиток (Код Студента, Гуртожиток, Кімната);

Іспити (Код_Іспиту, Дисципліна);
 Викладачі (Код_Викладача, Прізвище, Ім'я, По батькові,
 Кафедра, Посада);
 Сесія (Код_Студента, Код_Викладача, Оцінка, Дата_Іспиту);

Цілісність даних

У реляційній моделі об'єкти реального миру представлені у вигляді сукупності взаємозв'язаних відносин. Під цілісністю розуміється відповідність інформаційній моделі предметній області, що зберігається в базі даних, об'єктам реального миру і їх взаємозв'язкам в кожен момент часу. Будь-яка зміна в предметній області, значуща для побудованої моделі, повинна відображатися в базі даних, і при цьому повинна зберігатися однозначна інтерпретація інформаційної моделі в термінах наочної області.

Згідно Дейту, реляційна модель складається з трьох частин, що описують різні аспекти реляційного підходу:

Підтримка цілісності включає:

- структурну цілісність - схема бази даних може складатися тільки з нормалізованих таблиць;
- цілісність сутності - жоден атрибут первинного ключа не може містити відсутніх значень, що позначаються визначником NULL;
- посилальна цілісність - якщо у відношенні є зовнішній ключ, то він не може указувати на неіснуючий кортеж;
- корпоративні обмеження цілісності - додаткові правила обмеження цілісності, які визначаються користувачем або адміністратором бази даних.

Наприклад:

У відділі не може працювати більше 10 чоловік.

Абітурієнт не може бути старше 35 років.

У одному кабінеті не може працювати одночасно два лікарі.

Один викладач не може читати лекцію одночасно в двох аудиторіях.

Кожна СУБД повинна забезпечувати підтримку цілісності даних.

Визначник NULL

Визначник NULL означає, що значення атрибуту або невідомо в даний момент або неприйнятно для даного кортежу. Ключове слово NULL є спосіб обробки неповних або незвичайних даних, позначає

відсутність якого-небудь значення. Наприклад, студент внесений до списку групи, але ще не склав іспит, тоді в таблиці по обліку успішності проти його прізвища слід поставити NULL. Людина поки не має мобільного телефону.

Цілісність сутності

Перше обмеження цілісності стосується первинних ключів базових відносин. Базове відношення визначається як відношення деякої суті в концептуальній схемі.

У базовому відношенні жоден атрибут первинного ключа не може містити відсутніх значень, що позначаються визначником NULL. Це означає, що ніяка підмножина первинного ключа не може бути достатньою для унікальної ідентифікації кортежів.

Посилальна цілісність

Якщо у відношенні існує зовнішній ключ, то значення зовнішнього ключа повинне або відповідати значенню потенційного ключа деякого кортежу в його базовому відношенні, або задаватися визначником NULL.

Наприклад атрибут кодПостачальника є первинним ключем відношення Постачальник та зовнішнім ключем відношення Надходження товарів. За допомогою цього атрибуту здійснюється зв'язок між зазначеними відношеннями. СУБД повинна перешкоджати внесенню запису до відношення Надходження товарів, поки не буде внесений запис в Відношення Постачальник.

Відношення Надходження товарів

код	Товар	Код_Поставника
		Зовнішній ключ

Відношення Постачальник

кодПостачальника	Постачальник
Первинний ключ	

Не можна внести записи про постачальника в звіт по постачаннях, якщо постачальник ще не внесений до бази даних.

ТЕМА 3. НОРМАЛІЗАЦІЯ ВІДНОШЕНЬ

Загальні поняття нормалізації

Реляційна БД - сукупність нормалізованих відносин. Процес нормалізації вперше був запропонований Е.Ф. Коддом 1976 р.

Нормалізація - це розбиття таблиці на дві або більш, що володіють кращими властивостями при включенні, зміні і видаленні даних. Кінцева мета нормалізації - усунути надмірність даних, тобто кожен факт з'являється лише в одному місці. Надмірність даних можна визначити як повторення одних і тих же даних в декількох кортежах відношення або в декількох відносинах.

Нормалізація виконується для того, щоб скоротити об'єм інформації, що зберігається на диску, і усунути можливу при надмірності суперечливість даних.

Основою нормалізації є аналіз відносин на основі їх первинного ключа і існуючих функціональних залежностей. У результаті проведення нормалізації відношення повинні відповідати ряду правил (1НФ, 2НФ, 3НФ і т.д.), які можуть бути використані для перевірки окремих відносин, щоб вся база даних була нормалізована до бажаного ступеня нормалізації. Якщо вимоги нормалізації порушуються, то відношення, яке порушує ці вимоги, повинне бути декомпазоване на відносини, що відповідають вимогам нормалізації.

Надмірність даних і аномалії модифікації

Надмірність означає, що інформація в багатьох стовпцях багаторазово повторюється. Чим більше даних зберігатиметься в базі даних, тим більше інформації буде дублюватися і тим вище непродуктивні витрати. Надмірність даних зазвичай виникає при неправильній побудові схеми відношення, тобто коли в одне

відношення включається дуже багато атрибутів або одні і ті ж значення атрибутів повторюються в одному відношенні.

В цьому випадку виникають аномалії модифікації.

Надмірність даних приводить до аномалії оновлення, вставки, видалення.

Аномалії оновлення - інформація про цю сутність буде оновлена тільки в одному з відносин і залишиться незмінною в іншому відношенні або в межах одного відношення при зміні даних, що повторюються, в одному кортежі необхідно змінити їх в решті всіх кортежів.

Наприклад, якщо на одному з факультативів міняється викладач, то необхідно змінити інформацію у всіх рядках, що відносяться до цього факультативу. Інакше справа виглядатиме так, що один факультатив ведуть два викладачі

Приклад: ключ - код_студента

Відношення Спортивний гурток

Код_студента	Секція	Плата
100	Лижі	200
150	Плавання	50
175	Теніс	150
200	Плавання	50

У відношенні Спортивний гурток при зміні вартості абонементу конкретної секції її треба відновити для всіх студентів, які займаються в цій секції

- аномалії вставки: неможливо вставити нову інформацію в відношення, поки не буде модифіковано інше відношення.

Приклад: ключ - код_студента

У дане відношення Спортивний гурток неможливо вставити запис про нову секцію, поки в неї не буде записаний хоч один студент, оскільки ключ - код_студента не може приймати значення Null.

- аномалії видалення: - інформація про сутність буде видалена тільки в одному відношенні або при видаленні інформації про конкретний екземпляр суті віддаляється інформація про іншу сутність.

Якщо тільки в цьому відношенні Спортивний гурток зберігається інформація про плату за абонемент, то при видаленні запису про

студента віддаляється також увесь кортеж тобто віддаляється не тільки інформація про те, в якій секції він займається, але і вартість абонементу. Тобто при видаленні інформації про одну сутність (Студент), віддаляється інформація про іншу сутність (Секція).

Функціональні залежності

Теорія нормалізації базується на наявності тій або іншій залежності між атрибутами відношення. Функціональна залежність описує зв'язок між атрибутами відношення і є одним з основних понять нормалізації. Визначено два види таких залежностей: функціональні і багатозначні.

Функціональна залежність: якщо у відношенні R атрибут B пов'язаний функціонально з атрибутом A , то кожному значенню атрибуту A відповідає тільки одне значення атрибуту B .

Практично, якщо B функціонально пов'язаний з A , то кожний з кортежів, що має одне і теж значення A , повинен мати одне і теж значення B . Значення A і B можуть змінюватися з часом, але вони повинні змінюватися одночасно, тобто одне значення B відповідає одному значенню A .

Математичне позначення $A \rightarrow B$

Детермінантом функціональної залежності називається атрибут або група атрибутів, від якої залежать інші атрибути.

Повна функціональна залежність: атрибут B знаходиться в повній функціональній залежності від деяких атрибутів K_1 та K_2 , якщо він функціонально залежить від всіх атрибутів відразу і не залежить функціонально тільки від деяких з них. Тобто якщо атрибут B залежить від атрибутів A, B, C , то при повній функціональній залежності він не може залежати ні від одного з атрибутів окремо, ні від їх комбінації.

Функціональна залежність атрибуту B від набору атрибутів K є повною, якщо при видаленні якого-небудь атрибуту з набору K залежність не зберігається. Атрибут B повністю залежить від ключових атрибутів K_1 і K_2 - повна функціональна залежність

Частковою функціональною залежністю називається така залежність, коли при видаленні деякого атрибуту з набору ключових атрибутів K залежність не зберігається. Тобто якщо атрибут A частково залежить від атрибутів K_1 та K_2 , то він може залежати тільки від одного або декількох атрибутів з цього набору.

Часткова ФЗ - це залежність не ключового атрибуту тільки від окремих ключових атрибутів, а не від всіх ключових в цілому. Часткова функціональна залежність може виникати у відношенні з складеним ключем.

Приклад 1

Ціна ліків залежить від того, в якій аптеці продається.

Ліки	Термін придатності	Ціна	Аптека
------	--------------------	------	--------

Функціональні залежності цього відношення:

Ліки, Аптека $\rightarrow$ Ціна

Ліки $\rightarrow$ Термін придатності

ЯУ якості первинного ключа вибираються атрибути Ліки, Аптека. Тоді атрибут Термін придатності залежить тільки від частки ключових атрибутів, тобто має місце часткова залежність. А атрибут Ціна залежить від обох ключових атрибутів, тобто має місце повна функціональна залежність.

Транзитивна залежність: якщо для атрибутів А, В, С деякого відношення мають місце залежності $A \rightarrow B$ і $B \rightarrow C$, то атрибут С транзитивно залежить від атрибуту А через атрибут В. При цьому атрибут А функціонально не залежить ні від атрибуту В, ні від атрибуту С. Транзитивна залежність - це функціональна залежність між не ключовими атрибутами.

Багатозначна залежність

Відносно R існує багатозначна залежність $R.A \twoheadrightarrow R.B$ у тому і лише у тому випадку, коли безліч значень В, відповідне парі значень А і С, залежить тільки від А і не залежить від С.

У разі багатозначної залежності, що існує між атрибутами А, В, С деякого відношення, для кожного значення А є набір значень атрибуту В і набір значень атрибуту С, що входять в ці набори значень атрибутів В і С незалежні один від одного.

$A \twoheadrightarrow B$

$A \twoheadrightarrow C$

Приклад 3

У відношення Навчання зберігається інформація про те, який викладач читає яку дисципліну, і яка література рекомендується при цьому для вивчення дисципліни.

Відношення Навчання

Дисципліна	Викладач	Підручник
Інформатика	Шипилов П.А.	Форсайт Р. Паскаль для всіх
Інформатика	Шипилов П.А.	Уейт М. і ін. Мова Сі
Інформатика	Голованевский Г.Л.	Форсайт Р. Паскаль для всіх
Інформатика	Голованевский Г.Л.	Уейт М. і ін. Мова Сі

У цьому відношенні можна виділити наступні функціональні залежності. При цьому атрибути Викладач та Підручник не залежать один від одного

Дисципліна→Викладач

Дисципліна→Підручник

Перша нормальна форма

Перша нормальна форма - відношення, в якому на перетині кожного рядка і кожного стовпа міститься тільки одне значення.

Ненормалізована таблиця - таблиця, що містить одну або декілька груп даних, що повторюються.

Група, що повторюється, - це група записів, що складається з одного або декількох атрибутів верб якої можливі декілька значень для одного значення ключового атрибуту.

У даному прикладі група, що повторюється, складається з атрибутів Дисципліна і Оцінка, оскільки одному значенню ключа Код_Студента відповідає декілька записів атрибутів Дисципліна і Оцінка.

Для перетворення ненормалізованої таблиці в 1 нормалізовану форму необхідно в початковій таблиці знайти і усунути всі повторювальні групи даних.

Відношення Успішність

Код_Студента	ФІО	Дисципліни	Оцінка
01	Петренко А.Т.	Основи екології	4
		Програмування	4
02	Сидорко А.В.	Основи екології	5
		Програмування	5
03	Кулик А.О.	Основи екології	3
		Програмування	5

Існує два способи виключення повторювальних груп з ненормалізованої таблиці.

1 спосіб

Групи, що повторюються, усуваються шляхом введення відповідних даних в порожні стовпці рядків з даними, що повторюються, тобто заповнюються дублікатами даних, які не повторюються.

Приклад: у відношенні Успішність порожні осередки заповнені копіями записів, що не повторюються.

При такому способі нормалізації зберігається висока надмірність даних.

Відношення Успішність

Код Студента	ФІО	дисципліна	оцінка
01	Петренко А.Т.	Основи екології	4
01	Петренко А.Т.	Програмування	4
01	Петренко А.Т.	Фізика	5
02	Сидорко А.В.	Основи екології	5
02	Сидорко А.В.	Програмування	5
02	Сидорко А.В.	Фізика	5
03	Кулик А.О.	Основи екології	3
03	Кулик А.О.	Програмування	5
03	Кулик А.О.	Фізика	4

2 спосіб

Один або декілька атрибутів призначаються ключем ненормалізованої таблиці, а потім групи, що повторюються, вилучаються і поміщаються в окреме відношення разом з копіями ключа початкової таблиці. Далі в нових відношеннях встановлюються первинні ключі.

Приклад

Дані в приведеній вище таблиці повторюються в атрибутах Дисципліна і ФІО. Ці атрибути переносяться в нові відношення, а в вихідному залишаються копії ключів.

Створюються відносини

Дисципліна (Код_дисципліни, дисципліна)

Студент (Код_студента, Студент)

У вихідному відношенні залишаються тільки копії ключів. Т.ч. в вихідному відношенні атрибут ФІО просто вилучається, оскільки вже є атрибут Код_студента, атрибут Дисципліна замінюється атрибутом Код_дисципліни, в якому замість назви конкретної дисципліни проставляється відповідне значення ключового атрибуту відношення Дисципліна:

Відношення Студент

Код_студента	ФІО
01	Петренко А.Т.
02	Сидорко А.В.
03	Кулик А.О.

Відношення Дисципліна

Код_дисципліни	Дисципліна
01	Основи екології
02	Програмування
03	Фізика

Таким чином відношення Сесія містить тільки посилання з інших відношень та інформація, що зберігається у відношенні має цілий тип.

Обидва способи коректні, але другий спосіб дозволяє отримати відношення в 1НФ з меншою надмірністю даних. При нормалізації 1 способом надалі виконується декомпозиція таблиці на ті ж відносини, які можуть бути отримані при проведенні нормалізації іншим способом.

Відношення Сесія

Код_Студента	Код_дисципліни	оцінка
01	01	4
01	02	4
01	03	5
02	01	5
02	02	5
02	03	5
03	01	3
03	02	5
03	03	4

Приклад 3 Вихідне відношення має вигляд

Відношення Аптека

Код_ліків	Найменування ліків	Термін придатності	Ціна	Аптека
01	Аспірин	2006	2,50	Жива вода
			3,0	Матері і дитини
			3,1	№ 256
02	Колдфлю	2007	10,5	Жива вода
			11,50	Матері і дитини
			12	№ 256
03	Колдрекс	2005	14,50	Жива вода
			12,50	Матері і дитини

Я

Якщо проводити нормалізацію заданого відношення другим способом, вибравши як ключ атрибут Код_ліків, слід перенести в нове відношення атрибути Код_ліків, Найменування ліків, Термін придатності, а в вихідному відношенні залишити копії ключів.

Відношення Продаж ліків

Код_ліків	Ціна	Аптека
01	2,50	Жива вода
01	3,0	Матері і дитини
02	10,5	Жива вода
02	11,50	Матері і дитини
02	12	№256
03	14,50	Жива вода
03	12,50	Матері і дитини
03	15	№256

Приклад 4 Дато відношення Агентство нерухомості

Це відношення є ненормалізованим, тому що у відношення є незаповнені (пусті) комірки.

Нормалізація 1 способом

Як ключ приймемо атрибут Код_клієнта. Для кожного ключового атрибуту і атрибуту Клієнт повторюються записи в атрибутах

Групи, що повторюються = (Код_нерухомості, Адреса, Початок оренди, Кінець оренди, Плата, Код_власника, Власник)

Група, що в даному випадку повторюється, усувається введенням в кожен рядок атрибутів Код_клієнта і Клієнт відповідних записів.

Таким чином у відношенні не існує пустих комірок.

Це відношення знаходиться в 1 НФ, але характеризується надмірністю даних, що при фізичній реалізації приведе до аномалій оновлення, видалення даних. Тому отримане відношення необхідне перетворити в 2 НФ.

Наприклад 4 Відношення Оренда нерухомості

02		01	Код клієнта
Сич А.В.		Петров А.Т.	Клієнт
K10	K12	K10	Код_нерух
Перемоги 17-5	Струмяна 14-7	Перемоги 17-5	Адреса
01.06.03	02.06.03	01.02.03	Початок оренди
31.12.03	31.12.03	01.06.03	Кінець оренди
100	150	100	Плата
123	124	123	Код Влад.
Сирко А.А.	Петрік В.В	Сирко А.А.	Власник

K12	K5	
Перемоги 17-5	Гоголя 3-8	
24.02.02	01.02.03	
31.01.03	31.05.03	
150	90	
124	125	
Петрік В.В	Кирпа А.А.	

Нормалізоване відношення Оренда нерухомості яке відповідає

1 НФ.

02	01	01	Код клі-
Сич А.В.	Петров А.Т.	Петров А.Т.	Клієнт
K10	K12	K10	Код _ нерух
Перемоги 17-5	Струмяна 14-7	Перемоги 17-5	Адреса
01.06.03	02.06.03	01.02.03	Початок оренди
31.12.03	31.12.03	01.06.03	Кінець оренди
100	150	100	Плата
123	124	123	Код Влад.
Сирко А.А.	Петрік В.В	Сирко А.А.	Власник

02	02	Сич А.В.	К5	Гоголя 3-8	01.02.03	31.05.03	90	125	Кирпа А.А.
02	Сич А.В.	К12	Перемоги 17-5	24.02.02	31.01.03	150	124	Петрік В.В	

Нормалізація 2 способом

При використанні 2 способи група, що повторюється, вилучається з вихідного відношення і поміщається в окреме відношення разом з копією початкового ключового атрибуту. Тобто початкове відношення Оренда нерухомості розбивається на два відношення Клієнти і Нерухомість

Відношення Клієнти

Код клієнта	клієнт
01	Петров А.Т.
02	Сич А.В.

Відношення Власник

Код клієнта	Власник
01	Сирко А.А.
02	Петрік В.В
03	Кирпа А.А

Відношення Нерухомість

Код клієнта	Код_н ерух	Адреса	Початок оренди	Кінець оренди	Плата	Код Влад.
-------------	---------------	--------	----------------	---------------	-------	-----------

01	K10	Перемог и 17-5	01.02.03	01.06.03	100	123
01	K12	Струмян а 14-7	02.06.03	31.12.03	150	124
02	K10	Перемог и 17-5	01.06.03	31.12.03	100	123
02	K5	Гоголя 3-8	01.02.03	31.05.03	90	125
02	K12	Струмян а 14-7	24.02.02	31.01.03	150	124

Таблиця Нерухомість має надмірність даних, що також приведе до аномалій оновлення. Тому слід продовжити її нормалізацію.

Друга нормальна форма.

Друга нормальна форма заснована на понятті повної функціональної залежності.

Друга нормальна форма застосовується до відносин з складеними ключами (первинний ключ складається з двох або більш атрибутів).

2 нормальна форма: відношення, яке знаходиться в 1 НФ, і кожний атрибут якого, що не входить до складу первинного ключа, характеризується повною функціональною залежністю від цього ключа.

Нормалізація 1НФ-відносин з перетворенням в 2НФ-відношення включає усунення часткових залежностей, тобто якщо у відношенні між атрибутами існує частково-функціональна залежність, то частково - залежні атрибути вилучаються і переносяться в нове відношення разом з копією їх детермінантів.

Приклад

Задано відношення “Рецепти”.

Складаються рецепти блюд. Кожне блюдо може складатися з будь-якої кількості продуктів, кількість якої на порцію завжди постійно. Кожен постачальник може поставляти будь-яку кількість продуктів, але незалежно від постачальника ціна продукту залишається незмінною.

Код блюда	Назва блюда	Код_про дукту	Проду к-ти	Кількість 1 порцію	Ціна за кг	Загальна вартість	Поста- чальник
--------------	----------------	------------------	---------------	-----------------------	---------------	----------------------	-------------------

При нормалізації приймаємо такі умови, що обмежують значення і зв'язки між атрибутами.

Кожен Постачальник може поставляти будь-яку кількість найменувань продуктів і за будь-яку ціну.

Атрибутами, в яких можуть повторюватися значення, є атрибути Назва блюда, Постачальник і Продукт. Тому їх виносимо в окремі відносини, а в результатному залишаємо копії ключів.

Блюдо (Код блюда, Назва блюда)

Постачальник (Код_постачальника, Постачальник)

Продукти (Код_продукта, Продукти)

Тоді початкова таблиця має вигляд

Код блюда	Код Продук- ту	Кількість на 1 порцію	Ціна за кг	Загаль- на вартість	Код_Поста чальника

У початковому відношенні як ключ приймаються атрибути Код_блюда і Код продукту. Атрибут Загальна вартість є обчислювальним, тому його можна взагалі виключити з таблиці і розраховувати в запиті. Для приведення до 2НФ і вище виявляються функціональні завіси-мости між атрибутами відношення.

Функціональні залежності відношення Рецепті

Детермінант	Залежні атрибути	Вид залежності
Код блюда Код_продукт а	Кількість_порцію	первинний ключ
Код продукта	Код_постачальника	Часткова залежність $\Rightarrow$ залежність не від обох атрибутів ключа, а тільки от одного

Код продукту Код_постачальника	Ціна за кг	транзитивна залежність ⇒ залежність між неключовими атрибутами
-----------------------------------	------------	---

Залежність 2 – часткова, оскільки атрибут Код_постачальника залежить тільки від одного ключового атрибуту.

Залежність 3 є частковою (Ціна залежить тільки від одного ключового атрибуту Код продукту) і транзитивною (Код_постачальника і Ціна за кг - не ключові атрибути).

Приклад Відношення Нерухомість

Якщо як ключ відношення Нерухомість використовувати атрибути Кодклієнта і Код_нерухомості, виявляються наступні залежності

код залежності	Залежність	Вид залежності
1фз	КодКлієнта, КодНерухомості -> ПочОренди, КінецьОренди	Первинний ключ
2фз	КодКлієнта -> Клієнт	Часткова залежність (від частки ключа)
3фз	КодНерухомості -> Адреса, Платня, КодВласника, Власник	Часткова залежність (від частки ключа)
4фз	КодВласника-> Власник	Транзитивна залежність
5фз	КодКлієнта, ПочОренди,-> КодНерухомості, Адреса, КінецьОренди, КодВласника, Власник	Потенційний ключ
6фз	КодНерухомості, ПочОренди-> КодКлієнта, КінецьОренди	Потенційний ключ

Після виявлення функціональних залежностей процес нормалізації відношення продовжується перевіркою його відповідності

2НФ. Для цього потрібно знайти хоч би один випадок часткової залежності від первинного ключа.

Такими залежностями є

2фз	КодКлієнта -> Клієнт	Часткова залежність (від частки ключа)
3фз	КодНерухомості -> Адреса, Платня, КодВласника, Власник	Часткова залежність (від частки ключа)

Таким чином, відношення Нерухомість потрібно розбити на три відношення

Клієнт (Кодклієнта, Клієнт)
 Оренда (Кодклієнта, КодНерухомості, ПочОренди, КінецьОренди)
 Власник (КодНерухомості, Адреса, Плата, КодВласника, Власник)

Створені таблиці знаходяться в 2НФ, оскільки кожен атрибут, який не входить в первинний ключ повністю функціонально залежить від всіх атрибутів цього ключа.

Третя нормальна форма

Хоча відносини в 2НФ характеризуються меншою надмірністю даних, відношення в 1НФ, все ж таки вони можуть страждати аномаліями оновлення. Наприклад, якщо два об'єкти нерухомості належать одному власникові, то при зміні прізвища власника об'єкту необхідно змінювати декілька записів в таблиці Власник, в якій існує транзитивна залежність між атрибутами КодВласника і Власник.

Третя нормальна форма: відношення, яке знаходиться в 1НФ і 2НФ і яке не має атрибутів, що не входять в первинний ключ і не які знаходяться в транзитивній функціональній залежності від цього первинного ключа. (у таблиці немає транзитивних залежностей)

Якщо у відношенні існує транзитивна залежність між атрибутами, то транзитивне - залежні атрибути вилучаються з нього у нове відношення разом з копією їх детермінантів.

Приклад

У відношенні Власник існують наступні функціональні залежності:

КодНерухомості -> Адреса, Платня, КодВласника, Власник
КодВласника -> Власник

Атрибут Власник залежить від ключа КодНерухомості і від атрибуту КодВласника, тобто має місце транзитивна залежність, для усунення якої таблицю Власник слід розбити на дві нові:

Нерухомість(Код нерухомості, Адреса, Платня, Код власника)

Власник (Код власника, Власник)

Т.ч. вихідне відношення Агенція нерухомості було перетворене в 4 відношення

Клієнт (КодКлієнта, Клієнт)

Нерухомість (КодНерухомості, Адреса, Плата, КодВласника)

Власник(КодВласника, Власник)

Оренда (КодКлієнта, Почоренди, КодНерухомості, КінецьОренди)

Вихідне відношення може бути відновлене шляхом з'єднання зазначених відносин за допомогою первинних і зовнішніх ключів. Т.ч. процес нормалізації полягає в декомпозиції вихідного відношення шляхом послідовного виконання декількох операцій проєкції реляційної алгебри. Отримані в результаті декомпозиції відношення забезпечують їх з'єднання без втрат інформації.

Нормальна форма Бойса-Кодда

НФБК враховує функціональні залежності, в яких беруть участь потенційні ключі, а не тільки його первинні ключі. Для відношення з єдиним потенційним ключем його ЗНФ і НФБК є еквівалентними.

НФБК- відношення знаходиться в **НФБК** тоді і тільки тоді, кожен детермінант є потенційними ключем.

Для перевірки відповідності таблиці НФБК необхідно знайти всі детермінанти таблиці і переконатися в тому, що вони є ключами. Детермінант - атрибут, від якого повністю функціонально залежить інший атрибут.

Відмінність ЗНФ і НФБК полягає в тому, що функціональна залежність $A \rightarrow B$ припускається в ЗНФ навіть у тому випадку, коли B є первинним ключем, а A необов'язково є потенційним ключем. У НФБК ця залежність припускається тільки тоді, коли атрибут A є потенційним ключем.

Для наведених прикладів всі таблиці знаходяться в НФБК, оскільки всі детермінанти є ключами.

Порушення вимог НФБК відбувається у край рідко і лише в тих випадках, коли:

- є два або більш складених ключа;
- коли потенційні ключі перекриваються, тобто сумісно використовується хоча би один атрибут.

Приклад

Для проведення співбесіди того дня, коли призначена зустріч з клієнтом, співробітникамі виділяється спеціальна кімната. Протягом дня ця кімната може використовуватися різними співробітниками. Кожен клієнт бере участь у співбесіді один раз в день, але декілька в різні дні.

Відношення Співбесіда з клієнтом

Кодклієнта	Дата	Час	Код співробітника	Кімната
01	02.10.03	10.30	C2	101
02	02.10.03	12.00	C2	101
03	02.10.03	12.00	C1	102
02	03.10.03	10.30	C2	102

Потенційні ключі

(КодКлієнта, Дата)

(Співробітник, Дата, Час)

(Кімната, Дата, Час)

Якщо у якості первинного ключа вибрати атрибути КодКлієнта, Дата, то відношення буде мати вигляд:

Співбесіда з клієнтом (КодКлієнта, Дата, Співробітник, Час, Кімната)

Функціональні залежності відношення Співбесіда з клієнтом

	ключ	Залежні атрибути	Вид ключа
Ф31	КодКлієнта, Дата	Співробітник, Час, Кімната	первинний
Ф32	Співробітник, Дата	КодКлієнта, Час, Кімната	потенційний
Ф33	Кімната, дата, Час	КодКлієнта, Співробітник	потенційний
Ф34	Співробітник, Дата	Кімната	

Відношення Співбесіда з клієнтом не відповідає НФБК, оскільки Ф34 -> Співробітник, Дата→Кімната не є потенційним ключем (см ключі), що порушує вимогу НФБК- всі детермінанти - ключі. Тому його треба декомпонувати на два відношення:

Співбесіда (КодКлієнта, Дата, Час, Співробітник)

Зайнятість кімнати (Співробітник, Дата, Кімната)

Декомпонувати відношення, що не відповідають НФБК, не завжди рекомендується, оскільки при цьому втрачаються деякі функціональні залежності. В даному випадку втрачена функціональна залежність Ф33->Кімната, Дата, Час → КодКлієнта, Співробітник оскільки детермінант цієї залежності не знаходиться в тому ж відношенні, що і атрибути, які він визначає.

Четверта нормальна форма

Перша НФ забороняє мати в таблиці неатомарні значення атрибутів. Подібне обмеження породжує багатозначну залежність, що приводить до надмірності інформації. Але якщо необхідно моделювати багатозначні атрибути, то використовується поняття багатозначної залежності.

Нагадаємо, що атрибут А багатозначно визначає атрибут В тієї ж таблиці, якщо для кожного значення атрибуту А існує добре певна множина відповідних значень атрибуту В або умова, що забезпечує взаємну незалежність багатозначних атрибутів.

Теорема Фейджіна : відношення R(A,B,C) можна проєциувати без втрат в тому і лише у тому випадку, коли існує багатозначна залежність $MVD A \twoheadrightarrow B|C$. Під проєциуванням без втрат розуміють

такій спосіб декомпозиції, при якому початкове відношення відновлюється повністю і без надмірності шляхом природного з'єднання отриманих відносин.

Четверта нормальна форма: відношення відповідає НФБК і не має багатозначних залежностей.

Приклад

Філіал	Співробітник	Клієнт

Філіал→Сотрудник

Філіал→Клієнт

Розбити на два відношення Співробітник(Філіал, Співробітник)

Клієнт(Філіал, Клієнт)

Ще приклад

Відношення Спеціальність

Кодстудента	Група(Спеціальність)	Дисципліна

Спеціальність → Кодстудента (кожен студент вчиться на конкретній спеціальності)

Спеціальність → Дисципліна (спеціальність визначає перелік дисциплін, що вивчаються)

Між атрибутами Кодстудента і Дисципліна немає функціональною залежності. Тому це відношення треба декомпонувати на два

Спеціальність (Спеціальність, кодстудента)

Спеціальність (Спеціальність, дисципліна).

Ще приклад

Відношення Навчання

Дисципліна	Викладач	Підручник
Інформатика	Шипилов П.А.	Форсайт Р. Паскаль для всіх

Інформатика	Шипилов П.А.	Уейт М. і ін. Мова Сі
Інформатика	Голованевский Г.Л.	Форсайт Р. Паскаль для всіх
Інформатика	Голованевский Г.Л.	Уейт М. і ін. Мова Сі

У ній є багатозначна залежність "Дисципліна-Викладач": дисципліна (у прикладі Інформатика) може читатися декількома викладачами (у прикладі Шипиловим і Голованевським). Є і інша багатозначна залежність "Дисципліна-Підручник": при вивченні Інформатики використовуються підручники "Паскаль для всіх" і "Мова Сі". При цьому атрибути Викладач і Підручник не зв'язані функціональною залежністю, що призводить до появи надмірності інформації (для додавання ще одного підручника доведеться ввести в таблицю два нові рядки). Для усунення багатозначною залежності відношення Навчання слід розбити на два (Дисципліна-Викладач і Дисципліна-Підручник).

П'ята нормальна форма

При декомпозиції відношення на два інших отримані відносини мають властивість з'єднання без втрат, що означає, що отримані відношення можна з'єднати і отримати вихідне відношення. У кожне отримане відношення у разі потреби може бути і далі декомпозувало. У таких випадках необхідно враховувати залежність з'єднання, яка усувається за допомогою 5НФ.

Залежність з'єднання: властивість декомпозиції, яка викликає генерацію помилкових рядків при зворотному з'єднанні декомпозованих відношень за допомогою операції природного з'єднання.

При декомпозиції відносин за допомогою операції проєкції необхідно добитися того, щоб при зворотному з'єднанні отриманих відносин можна було відновити вихідне з'єднання - декомпозиція із з'єднанням без втрат (зберігаються всі дані початкового відношення і не з'являються додаткові рядки). У попередньому прикладі результатом декомпозиції відношення Результати іспитів (КодСтудента, Спеціальність, Дисципліна) є два відношення, які при з'єднанні дадуть в точності відношення Результати іспитів.

П'ята нормальна форма: відношення без залежності з'єднання.

5НФ називається також продуктивне-сполучною НФ, що означає, що в такій формі немає залежності з'єднання.

Послідовність проведення нормалізації

Нормалізація відношень дозволяє зменшити надмірність даних у базі даних. Але після її проведення збільшується кількість таблиць та зменшується швидкість пошуку потрібних даних у базі.

1. Видалення груп, що повторюються, - 1НФ.
2. Видалення залежності від частини ключа - 2НФ.
3. Видалення транзитивних залежностей - 3НФ.
4. Видалення із залежностей, що залишилися, аномалій, що залишилися, - НФБК.
5. Видалення нетривіальних багатозначних залежностей - 4НФ.
6. Видалення залежностей з'єднання - 5НФ.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Конноли Т., Бегг К., Страчан А. Базы данных. Проектирование, реализация и сопровождение. Теория и практика.- Москва-Санкт-Петербург- Киев: Вильямс, 2013.-1719 с.
2. Дейт К. ДЖ. Введение в системы баз данных.- Киев-Москва: Диалектика, 2011. – 782 с.
3. Хансен Г., Хансен В. Базы данных. Разработка и управление. - Москва: Бином, 2012. – 356 с.
4. Г. Гарсия-Молина, Дж. Ульман, Дж. Уидом. Системы баз данных. Полный курс.- Москва-Санкт-Петербург -Киев: Вильямс, 2013.- 1182 с.
5. Д.Кренке. Теория и практика построения баз данных. Москва: Питер, 2012: - 850 с.
6. Швецов В.И., Визгунов А.Н., Мееров И.Б. Базы данных. Учебное пособие. Нижний Новгород: Изд-во ННГУ, 2010.- 217 с.
7. Хомопенко А. Д. , Цыганков В. М., Мальцев М. Г. Базы данных: Учебник для высших учебных заведений / Под ред. А. Д. Хомопенко. — 6-е изд., доп. - СПб.: КОРОНА-Век, 2009. - 736 с.
8. Туманов В.Е., Маклаков С.В., Проектирование реляционных хранилищ данных.-Москва: ДиалогМифи, 2012.-333с.
9. www.citforum.ru
10. www.os.ru

11. www.olap.ru
12. www.technet.microsoft.com/ru-ru/library/gg592949.aspx
13. www.sql.ru
14. www.interface.ru
15. www.proklondike.com