МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

М. В. Добролюбова

ПРОГРАМУВАННЯ БАЗ ДАНИХ

КОНСПЕКТ ЛЕКЦІЙ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра за освітньою програмою
«Інформаційні вимірювальні технології»
спеціальності 152 «Метрологія та інформаційно-вимірювальна техніка»

Київ
КПІ ім. Ігоря Сікорського
2021

Рецензент *Попов, А. О.*, к.т.н., доцент, доцент, кафедра електронної інженерії, ФЕЛ, КПІ ім. Ігоря Сікорського
Помазун, О. М., к.е.н., доцент, доцент, кафедра інформаційних систем в економіці, ІІТЕ, КНЕУ імені Вадима Гетьмана

Відповідальний редактор *Богомазов, С. А.*, к.т.н., доцент

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 1 від 16.09.2021 р.) за поданням Вченої ради Приладобудівного факультету (протокол № 6/21 від 29.06.2021 р.)

Електронне мережне навчальне видання

Добролюбова Марина Валеріївна, канд. техн. наук, доц.

ПРОГРАМУВАННЯ БАЗ ДАНИХ
КОНСПЕКТ ЛЕКЦІЙ

Програмування баз даних: конспект лекцій [Електронний ресурс] : навч. посіб. для студ. спеціальності 152 «Метрологія та інформаційно-вимірювальна техніка» / М. В. Добролюбова ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 22,7 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2021. – 275 с.

Навчальний посібник забезпечує лекційний курс з дисципліни «Програмування баз даних».

Навчальний посібник знайомить студентів з фундаментальними поняттями в області баз даних, які необхідні при проєктуванні та експлуатації інформаційно-вимірювальних систем і приладів. Чітка структурованість лекцій допоможе студентові швидше і легше оволодіти теоретичними положеннями для кращого розуміння будови і роботи баз даних та застосовувати набуті уміння при розв’язанні за допомогою ПК реальних науково-технічних задач різного ступеня складності.

ПЕРЕДМОВА

Курс лекцій з дисципліни «Програмування баз даних» сформовано на основі багаторічного досвіду розробки баз даних та викладання автором принципів їх проєктування, реалізації і супроводу.

Курс лекцій відповідає освітній програмі «Інформаційні вимірювальні технології» за спеціальністю 152 «Метрологія та інформаційно-вимірювальна техніка».

Курс лекцій допомагає організувати самостійну роботу студента для вивчення і систематизації знань за темами, пов’язаними із поняттями та термінами, що використовуються в області розробки баз даних, з розглядом реляційної, об’єктно-орієнтованої, об’єктно-реляційної, розподіленої моделей та принципами їх розробки і функціонування, зв’язком Web-технологій та баз даних, організацією сховищ даних та концепціями аналізу даних.

Курс лекцій базується на чіткому викладі теоретичних матеріалів, містить презентації до кожного лекційного заняття та контрольні питання з відповідних тем, що необхідні для закріплення здобутих знань і навичок.

ЗМІСТ

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.1 ПОНЯТТЯ ТА ТЕРМІНИ. СЕРЕДОВИЩЕ БД

Лекція 1

ВСТУП ДО БАЗ ДАНИХ

Мета лекції: розгляд основних понять, пов’язаних з базами даних.

Зміст лекції

1. Поняття бази даних.
2. Файлові системи.
3. Системи з базами даних.
    3.1. СУБД. Користувачі СУБД. Розподіл обов’язків.
    3.2. Історія розвитку СУБД.
    3.3. Переваги та недоліки СУБД.
4. Практичний приклад.

Поняття бази даних

- Досягнення в дослідженнях баз даних стали основою фундаментальних розробок комунікаційних систем, транспорту і логістики, фінансового менеджменту, систем з базами знань, методів доступу до наукової літератури, а також великої кількості цивільних і військових програм.
- Поява технології баз даних стала каталізатором багатьох значних досягнень в області програмного забезпечення.
- Бази даних стали невід'ємною частиною нашого повсякденного життя.

Файлові системи

- ***Файлові системи*** – набір програм, які виконують для користувачів деякі операції, при цьому кожна програма визначає свої власні дані та управляє ними.
- Обмеження, властиві файловим системам:

  - розділення та ізольованість даних;
  - повторюваність даних;
  - несумісність файлів;
  - залежність від даних;
  - фіксовані запити / швидке збільшення кількості додатків.
- Фактори, що впливають на обмеження файлових систем:
  - визначення даних міститься всередині додатків, не зберігається окремо та незалежно від них;
  - крім додатків не передбачено ніяких інших інструментів доступу до даних та їх обробки.

Системи з базами даних

- ***База даних*** – набір логічно пов'язаних даних спільного використання (і опис цих даних), призначений для задоволення інформаційних потреб.
- Характерні ознаки БД:
  - єдине сховище даних, яке визначається одноразово, а потім одночасно використовується багатьма користувачами;
  - загальний корпоративний ресурс;
  - зберігає не лише дані, а і їх опис.

СУБД. Користувачі СУБД. Розподіл обов'язків

- ***СУБД*** – це програмне забезпечення, за допомогою якого користувачі можуть визначати, створювати, підтримувати базу даних, а також здійснювати над нею контрольований доступ.
- Можливості СУБД:
  - дозволяє визначати базу даних за допомогою мови визначення даних;
  - дозволяє вставляти, оновлювати, видаляти, витягувати інформацію з бази даних за допомогою мови управління даними;

- надає контрольований доступ до бази даних за допомогою системи забезпечення безпеки та системи підтримки цілісності даних;
- забезпечує додатковий рівень безпеки;
- надає механізм налаштування зовнішнього інтерфейсу бази даних;
- дозволяє зберігати зовнішній інтерфейс бази даних несуперечливим і незмінним навіть при внесенні змін до її структуру.

- Основні компоненти СУБД:
  - апаратне забезпечення;
  - програмне забезпечення;
  - дані;
  - процедури;
  - користувачі.

Історія розвитку СУБД

- Етапи розвитку СУБД:
  - програмне забезпечення під назвою GUAM (Generalized Update Access Method);
  - система IMS (Information Management System);
  - система IDS (Integrated Data Store);
  - розробка реляційної моделі даних;
  - розробка структурованої мови запитів SQL;
  - модель «сутність-зв'язок» (Entity-Relationship model - ER-модель);
  - об'єктно-орієнтовані СУБД і об'єктно-реляційні СУБД.
- Основні складові стандарту баз даних:
  - мережева схема – це логічна організація всієї бази даних в цілому, яка включає визначення імені бази даних, типу кожного запису та компонентів записів кожного типу;
  - підсхема – це частина бази даних (як її бачуть користувачі або додатки);

- мова керування даними – інструмент для визначення характеристик структури даних і управління ними.

Переваги і недоліки СУБД

- Переваги:
  - контроль за надмірністю даних;
  - несуперечність даних;
  - більше корисної інформації при тому ж обсязі збережених даних;
  - спільне використання даних;
  - підтримка цілісності даних;
  - підвищена безпека;
  - застосування стандартів;
  - підвищення ефективності зростанням масштабів системи;
  - можливість знаходження компромісу при суперечливих вимогах;
  - підвищення готовності даних до роботи;
  - підвищення доступності даних;
  - покращення показників продуктивності;
  - спрощення супроводу системи за рахунок незалежності від даних;
  - покращене керування паралельністю;
  - розвинені служби резервного копіювання та відновлення.
- Недоліки:
  - складність;
  - розмір;
  - вартість СУБД;
  - додаткові витрати на апаратне забезпечення;
  - витрати на перетворення;
  - продуктивність;
  - серйозні наслідки при виході системи з ладу.

Практичний приклад

- В якості практичного прикладу в курсі лекцій розгладається навчальний проєкт DreamHome, який описує роботу компанії на території Великобританії, що займається здачею в оренду об'єктів нерухомості за дорученням їх власників. Компанія пропонує повний комплекс послуг власникам, які бажають здати оренду мебльовану нерухомість. Пропоновані компанією DreamHome послуги включають:
  - рекламу нерухомості;
  - опитування передбачуваних орендарів;
  - організацію перегляду об'єктів, що здаються в оренду потенційними орендарями;
  - складання договорів на оренду.
- Після здачі нерухомості в оренду на компанію DreamHome покладається відповідальність за неї, тобто співробітники DreamHome повинні регулярно перевіряти поточний стан об'єктів.
-

Закріплення матеріалу

1. Роз’яснити значення наступних термінів: «дані», «база даних», «система управління базами даних», «незалежність від даних», «представлення», «цілісність» та «безпека».
2. Описати підхід, що використовується в файлових системах. Вказати основні недоліки даного підходу.
3. Описати основні характеристики підходу, який базується на використанні бази даних..
4. Які основні компоненти СУБД вам відомі?
5. Пояснити ролі різних користувачів СУБД.
6. Назвати основні переваги СУБД.
7. Назвати основні недоліки СУБД.

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.1 ПОНЯТТЯ ТА ТЕРМІНИ. СЕРЕДОВИЩЕ БД

Лекція 2

СЕРЕДОВИЩЕ БАЗИ ДАНИХ

Мета лекції: розгляд архітектури СУБД, типів схем баз даних, типів незалежності від даних, видів моделей даних, функцій та основних компонентів СУБД, типових архітектурних рішень багатокористувальницьких СУБД.

Зміст лекції

1. Трирівнева архітектура.
2. Мови баз даних.
3. Моделі даних та концептуальне моделювання.
4. Функції та компоненти СУБД.
5. Архітектура багатокористувальницьких СУБД.

Трирівнева архітектура

- Перша спроба створення стандартної термінології загальної архітектури СУБД була зроблена у 1971 році групою DBTG – дворівневий підхід, побудований на основі використання системного представлення, тобто схеми та уявлень користувача (підсхем). 1975 році Комітет ANSI / SPARC визнав необхідність використання трирівневого підходу – додався незалежний рівень для ізоляції програм від особливостей подання даних на більш низькому рівні.
- Причини розділення на три рівні:

    1. Кожен користувач повинен мати можливість звертатися до одних й тих самих даних, використовуючи своє власне уявлення про них та мати можливість змінювати своє уявлення про дані, не впливаючи на інших користувачів.

2. Користувачі не повинні безпосередньо мати справу з індексуванням та хешуванням.

3. Адміністратор бази даних повинен мати можливість змінювати структуру зберігання даних в базі, не впливаючи на уявлення користувачів.

4. Внутрішня структура бази даних не повинна залежати від змін фізичних аспектів зберігання інформації.

5. Адміністратор бази даних повинен мати можливість змінювати концептуальну або глобальну структуру бази даних без будь-якого впливу на всіх користувачів.

- ***Зовнішній рівень*** – подання бази даних точки зору користувачів.
- ***Концептуальний рівень*** – узагальнююче представлення бази даних.
- ***Внутрішній рівень*** – фізичне представлення бази даних в комп'ютері.
- ***Фізичний рівень*** – рівень, що контролюється операційною системою, але під керівництвом СУБД.

Схеми відображення та екземпляри

- ***Схема бази даних*** (вміст) – це загальний опис БД.
- ***Стан бази даних*** (деталізація) – сукупність інформації, що зберігається в базі даних у будь-який визначений момент часу.
- ***Зовнішня схема (підсхема)*** – знаходиться на найвищому рівні та відповідає різним представленням даних.
- ***Концептуальна схема*** – знаходиться на концептуальному рівні та описує всі елементи даних і зв'язки між ними, з зазначенням необхідних обмежень підтримки цілісності даних.
- ***Внутрішня схема*** – знаходиться на найнижчому рівні абстракції і є повним описом внутрішньої моделі даних; містить визначення збережених записів, методи представлення, опису полів даних, відомості про індекси та обрані схеми хешування.

Незалежність від даних

- ***Логічна незалежність*** – незалежність від даних, що означає повну захищеність зовнішніх схем від змін, які вносяться в концептуальну схему.
- ***Фізична незалежність*** – незалежність від даних, що означає захищеність концептуальної схеми від змін, які вносяться у внутрішню схему.

Мови баз даних

- ***Мова визначення даних*** (Data Definition Language – DDL) використовується для визначення схеми бази даних.
- ***Мова управління даними*** (Data Manipulation Language – DML) використовується для читання та оновлення даних, що зберігаються базі.
- ***Метадані*** – дані, які описують об'єкти бази даних та дозволяють спростити спосіб доступу до них і управління ними.
- ***Процедурна мова DML*** – мова, яка дозволяє повідомити системі те, які дані необхідні, і точно вказати, як їх можна витягти.
- ***Непроцедурна (декларативна) мова DML*** – мова, яка дозволяє вказати лише те, які дані необхідні, але не те, як їх слід витягувати.
- Мови останніх поколінь: мови представлення інформації, спеціалізовані мови, генератори додатків, мови дуже високого рівня.

Моделі даних та концептуальне моделювання

- ***Модель даних*** – інтегрований набір понять для опису даних, зв'язків між ними та обмежень, що накладаються на дані.
- Складові моделі даних:
  - структурна частина;
  - керуюча частина;
  - набір обмежень підтримки цілісності даних (необов'язково).

- ***Структурна частина*** – це складова моделі даних, яка представляє собою набір правил, за якими будується база даних.
- ***Керуюча частина*** – це складова моделі даних, яка визначає типи припустимих операцій з даними.
- ***Набір обмежень підтримки цілісності даних*** – це складова моделі даних, яка гарантує коректність даних, що використовуються.
- Види моделей даних:
  - зовнішня модель даних (предметна область);
  - концептуальна модель даних;
  - внутрішня модель даних.
- Категорії моделей даних: об'єктні, на основі записів, фізичні.
- Типи об'єктних моделей даних:
  - модель типу "сутність-зв'язок", або ER-модель;
  - семантична модель;
  - функціональна модель;
  - об'єктно-орієнтована модель.
- Типи логічних моделей даних на основі записів:
  - реляційна модель даних;
  - мережева модель даних;
  - ієрархічна модель даних.
- Типи фізичних моделей даних:
  - узагальнююча;
  - модель пам'яті кадрів.

Функції та компоненти СУБД

- Функції СУБД:
  - зберігання, витягування та оновлення даних;
  - каталог, доступний кінцевим користувачам;
  - підтримка транзакцій;

  - сервіси управління паралельністю;
  - сервіси відновлення;
  - сервіси контролю доступу до даних;
  - підтримка обміну даними;
  - служби підтримки цілісності даних;
  - служби підтримки незалежності від даних;
  - допоміжні служби.
- Програмні компоненти середовища СУБД:
  - процесор запитів: перетворює запити у послідовність низькорівневих інструкцій для контролера бази даних;
  - контролер бази даних: взаємодіє з запущеними користувачами прикладними програмами і запитами;
  - контролер файлів: маніпулює призначеними для зберігання даних файлами та відповідає за розподіл доступного дискового простору, створює і підтримує список структур та індексів, визначених у внутрішній схемі, викликає функції хешування для генерації адрес записів, але не керує фізичним вводом та виводом даних безпосередньо;
  - препроцесор мови DML: перетворює впроваджені в прикладні програми DML-оператори у виклики стандартних функцій базової мови;
  - компілятор мови DDL: перетворює команди в набір таблиць, що містять метадані;
  - контролер словника: управляє доступом до системного каталогу і забезпечує роботу з ним.
- Основні програмні компоненти, що входять до складу контролера бази даних:
  - контроль прав доступу: перевіряє наявність у даного користувача повноважень для виконання затребуваної операції;

- процесор команд: приймає управління для виконання затребуваної операції після перевірки повноважень користувача;
- засоби контролю цілісності: виконують перевірку того, чи задовольняє затребувана операція всім встановленим обмеженням підтримки цілісності даних;
- оптимізатор запитів: визначає оптимальну стратегію виконання запиту.
- контролер транзакцій: здійснює необхідну обробку операцій, що надходять в процесі виконання транзакцій.
- Планувальник: відповідає за безконфліктне виконання паралельних операцій з базою даних, керує відносним порядком виконання операцій, викликаних в окремих транзакціях;
- контролер відновлення: гарантує відновлення бази даних до несуперечливого стану при виникненні збоїв;
- контролер буферів: відповідає за перенесення даних між оперативною пам'яттю та вторинним запам'ятовуючим пристроєм.

Архітектура багатокористувальницьких СУБД

- ***Телеобробка*** – це архітектура, при якій один комп'ютер з єдиним процесором з'єднаний з декількома терміналами.
- ***Файловий сервер***. У середовищі файлового сервера обробка даних зазвичай розподілена в локальній обчислювальній мережі.
- ***"Клієнт / сервер"*** – такий спосіб взаємодії програмних компонентів, при якому вони утворюють єдину систему. Тобто існує деякий клієнтський процес, що вимагає певних ресурсів, а також серверний процес, який ці ресурси надає. В контексті бази даних клієнт управляє призначеним для користувача інтерфейсом і логікою додатків. Сервер приймає та обробляє запити до бази даних, потім передає отримані результати назад клієнтові.

- Web-середовище складається з мережі комп'ютерів, які можуть діяти або як сервери, що надають інформацію, або як клієнти (які зазвичай називаються браузерами), що запитують інформацію.
- Бізнес-додатки для інтенсивної роботи з даними складаються з чотирьох основних компонентів: бази даних, логіки транзакцій, логіки додатка та інтерфейсу користувача.
- В середовищі Web традиційна дворівнева модель "клієнт / сервер" (товстий клієнт – відповідає за інтерфейс користувача та основну логіку обробки даних) замінена трирівневою моделлю, що складається з рівня інтерфейсу користувача (тонкий клієнт – відповідає за інтерфейс користувача), рівня бізнес-логіки та обробки даних (сервер додатку) і рівня СУБД (сервер баз даних), які розподілені між різними комп'ютерами.

Закріплення матеріалу

1. Назвати причини розділення архітектури СУБД на три рівні.
2. Пояснити поняття «схема БД» та описати її типи.
3. Назвати типи незалежності від даних та пояснити їх суть.
4. Пояснити поняття «модель даних», описати її складові, види та категорії.
5. Назвати основні функції СУБД.
6. Назвати основні програмні компоненти середовища СУБД та пояснити їх призначення.
7. Назвати типові архітектурні рішення багатокористувальницьких СУБД.
8. Описати функції та загальне значення системного каталогу.
9. Описати типи сервісів, які повинна надавати типова багатокористувальницька СУБД.

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.2 РЕЛЯЦІЙНА МОДЕЛЬ. ПЛАНУВАННЯ, ПРОЄКТУВАННЯ, АДМІНІСТРУВАННЯ БД

Лекція 3

РЕЛЯЦІЙНІ БАЗИ ДАНИХ

Мета лекції: розгляд термінології, що застосовується при роботі з реляційною моделлю; розгляд поняття реляційної цілісності; опанування аспектів роботи основних та додаткових операцій реляційної алгебри; формування розуміння вимог, що висуваються до РСУБД.

Зміст лекції

1. Термінологія.
    1.1. Структура реляційних даних.
    1.2. Властивості відношень.
    1.3. Реляційні ключі.
2. Реляційна цілісність.
3. Реляційна алгебра.
4. Вимоги до реляційної СУБД.

Термінологія

- Реляційна модель вперше була запропонована Едгаром Коддом у 1970 році.
- Цілі створення реляційної моделі:

    1) Забезпечення більш високого ступеня незалежності від даних.

    2) Створення міцного фундаменту для вирішення семантичних питань, проблем несуперечності та надмірності даних, нормалізованих відношень.

3) Розширення мов управління даними за рахунок включення операцій над множинами.

Структура реляційних даних

- Реляційна модель заснована на математичному понятті відношення, фізичним поданням якого є таблиця.
- ***Відношення*** – це двовимірна таблиця, яка складається із стовпців та рядків. В будь-якій реляційній СУБД береться за основу сприйняття бази даних користувачем, як набору таблиць.
- ***Атрибут*** – це поіменований стовпець відношення. Атрибути можуть розміщуватись у відношенні в довільному порядку. Навіть якщо вони пере впорядковуються, відношення залишається тим самим.
- ***Домен*** – це набір допустимих значень для одного або декількох атрибутів. Для кожного з атрибутів домени можуть відрізнятися, але два та більше атрибутів можуть визначатись на одному домені.
- ***Кортеж*** – це рядок відношення. Кортежі можуть розміщуватись у відношенні в довільному порядку. При цьому сенс відношення не змінюється.
- ***Ступінь відношення*** визначається кількістю атрибутів, яку воно містить.
- ***Унарне відношення*** – це відношення лише з одним атрибутом.
- ***Бінарне відношення*** – це відношення лише з двома атрибутами.
- ***Тернарне відношення*** – це відношення лише з трьома атрибутами.
- ***N-арне відношення*** – це відношення лише з великою кількістю атрибутів.
- ***Кардинальність*** – це кількість кортежів, яку містить відношення.
- ***Реляційна база даних*** – це набір нормалізованих відношень.

- ***Базове відношення*** – це поіменоване відношення, яке відповідає сутності в концептуальній схемі, кортежі якого фізично зберігаються в базі даних.
- ***Представлення*** – це динамічний результат однієї або декількох реляційних операцій над базовими відношеннями для створення деякого іншого відношення.
- Причини використання механізму представлень:
  - надає потужний гнучкий механізм захисту за рахунок приховування деяких частин бази даних від певних користувачів;
  - організує найбільш зручним для них чином доступ користувачів до даних;
  - дозволяє спрощувати складні операції з базовими відношеннями.
- Умови для перевірки допустимості поновлення даних через деяке представлення:
  - оновлення допускаються через представлення, яке визначено на основі простого запиту до єдиного базового відношення та містить первинний або потенційний ключ даного базового відношення;
  - оновлення не допускаються в будь-яких представленнях, визначених на базі декількох базових відношень;
  - оновлення не допускаються в будь-яких представленнях, що включають виконання операцій узагальнення або групування.
- Класи представлень:
  - теоретично непоновлювані;
  - теоретично поновлювані;
  - частково поновлювані.

Властивості відношень

- Характеристики відношень:

- відношення має ім'я, яке відрізняється від імен всіх інших відношень;
- кожна комірка відношення містить тільки атомарне (неподільне) значення;
- кожний атрибут має унікальне ім'я;
- значення атрибута беруться з одного і того самого домену;
- порядок проходження атрибутів не має ніякого значення;
- кожен кортеж є унікальним, тобто дублікатів кортежів бути не може;
- порядок слідування кортежів у відношенні не має ніякого значення (теоретично).

- Порядок слідування заголовків стовпців у заголовку відношення є несуттєвим. Однак, якщо структура відношення вже визначена, то порядок елементів в кортежах тіла відношення повинен відповідати порядку імен атрибутів.

Реляційні ключі

- ***Суперключ*** (superkey) – це атрибут або множина атрибутів, яка єдиним чином ідентифікує кортеж даного відношення.
- ***Потенційний ключ*** – це суперключ, який не містить підмножини, яка також є суперключем даного відношення.
- Властивості потенційного ключа:

  1) унікальність;

  2) незвідність.
- ***Первинний ключ*** – це потенційний ключ, який обраний для унікальної ідентифікації кортежів всередині відношення.
- ***Зовнішній ключ*** – це атрибут або множина атрибутів всередині відношення, яка відповідає потенційному ключу деякого відношення.

Реляційна цілісність

- ***Визначник NULL*** – вказує на той факт, що значення атрибута на теперішній час невідоме чи неприйнятне для цього кортежу.
- ***Цілісність сутностей*** свідчить про те, що у базовому відношенні жоден атрибут первинного ключа не може містити відсутніх значень, які позначаються визначником NULL.
- ***Цілісність посилання*** свідчить про те, що якщо у відношенні існує зовнішній ключ, то значення цього ключа повинно або відповідати значенням потенційного ключа деякого кортежу в його базовому відношенні, або ж задаватися визначником NULL.
- ***Корпоративні обмеження цілісності*** – це додаткові правила підтримки цілісності даних, що визначаються користувачами або адміністраторами бази даних.

Реляційна алгебра

- ***Реляційна алгебра*** – це теоретична мова операцій, які на основі одного або декількох відношень дозволяють створювати інше відношення без зміни вихідних відношень.
- П'ять основних операцій реляційної алгебри:
  - вибірка (selection);
  - проєкція (projection);
  - декартовий добуток (cartesian product);
  - об'єднання (union);
  - різниця (set difference).
- Додаткові операції:
  - операції поєднання (join);
  - операції перетину (intersection);
  - операції ділення (division).

- ***Операція вибірки*** (selection) – це операція, яка виконується над кортежами одного відношення. Результатом операції є нове відношення, яке складається з кортежів вихідного відношення, що задовольняють заданій умові.
- ***Операція проєкції*** (projection) – це операція, яка виконується над кортежами одного відношення. Результатом операції є нове відношення, яке містить лише задані атрибути вихідного відношення.
- ***Декартовий добуток*** (cartesian product) – це операція, яка виконується над двома відношеннями. Результатом операції є нове відношення, що складається з усіх можливих кортежів, які є попарними поєднанням кортежів вихідних відношень.
- ***Об'єднання*** (union) – це операція, яка виконується над двома відношеннями. Результатом операції є нове відношення, що складається з усіх кортежів вихідних відношень, при цьому спільні для вихідних відношень кортежі в новому відношенні зустрічаються лише по одному разу.
- ***Різниця*** (set difference) – це операція, яка виконується над двома відношеннями. Результатом операції є нове відношення, що складається з кортежів, які належать першому відношенню і не належать другому.
- ***Поєднання*** (join) – це операція, яка виконується над двома відношеннями, що мають спільні атрибути. Результатом операції є нове відношення, що складається з усіх атрибутів вихідних відношень і об’єднує лише ті кортежі вихідних відношень, в яких значення спільних атрибутів співпадають.
- Типи операцій поєднання:
  - тета-поєднання (Θ-join);
  - поєднання за еквівалентністю (equi-join), яке є приватним видом тета-поєднання;
  - природнє поєднання (natural join);

- зовнішнє поєднання (outer join);
- напівпоєднання (semi-join).

- ***Перетин*** (intersection) – це операція, яка виконується над двома відношеннями. Результатом операції є нове відношення, що складається з кортежів, які належать обом вихідним відношенням.
- Результатом ***ділення*** (division) відношення A(A1, A2, …, An) на відношення B(B1, B2, …, Bn) по відношенню C(A1, A2, …, An, B1, B2, …, Bn) називається нове відношення, що містить всі такі кортежі *x* з відношення A, для яких виконується умова: для будь-якого *y*, що належить множині кортежів B, відношення C містить кортеж *x*:*y*.
- В *реляційному численні кортежів* завдання полягає у знаходженні таких кортежів, для яких предикат є істинним. Це числення засноване на змінних кортежу. ***Змінними кортежу*** є такі змінні, областю визначення яких є вказане відношення.
- В *реляційному численні доменів* використовуються змінні, значення яких беруться з доменів, а не з кортежів відношень.

Вимоги до реляційної СУБД

- *ФУНДАМЕНТАЛЬНІ ПРАВИЛА (ПРАВИЛА 0 ТА 12)*
- **Правило 0 – фундаментальне правило:** будь-яка система, яка представляється або рекламується як реляційна СУБД, повинна бути здатна управляти базами даних виключно за допомогою її реляційних функцій.
- **Правило 12 – правило заборони обхідних шляхів:** якщо реляційна система має низькорівневу мову, вона не може бути використана для скасування або обходу правил обмежень цілісності, складених реляційною мовою більш високого рівня.
- *СТРУКТУРНІ ПРАВИЛА (ПРАВИЛА 1 ТА 6)*

- **Правило 1 – представлення інформації:** вся інформація реляційної базі даних представляється у явному вигляді на логічному рівні тільки одним способом – у вигляді значень в таблицях.
- **Правило 6 – поновлення представлення:** всі представлення, які є теоретично поновлюваними, повинні бути поновлюваними і в даній системі.
- *ПРАВИЛА ЦІЛІСНОСТІ (ПРАВИЛА 3 ТА 10)*
- **Правило 3 – систематична обробка невизначених значень (NULL):** невизначені значення (NULL) підтримуються для систематичного представлення відсутньої або ж неприйнятної інформації незалежно від типу даних.
- **Правило 10 – незалежність обмежень цілісності:** обмеження цілісності, які є специфічними для конкретної реляційної СУБД, повинні визначатися на підмові реляційних даних та зберігатися не в прикладних програмах, а в системному каталозі.
- *ПРАВИЛА МАНІПУЛЮВАННЯ ДАНИМИ (ПРАВИЛА 2, 4, 5 ТА 7)*
- **Правило 2 – гарантований доступ:** для всіх та кожного елемента даних реляційної бази даних повинен бути гарантований логічний доступ на основі комбінації імені таблиці, значення первинного ключа і значення імені стовпця.
- **Правило 4 – динамічний інтерактивний каталог, побудований за правилами реляційної моделі: о**пис бази даних повинен подаватися на логічному рівні таким само, як і звичайні дані, що дозволить санкціонованим користувачам використовувати для звернень до цього опису ту саму реляційну мову, що і при зверненні до даних.
- **Правило 5 – вичерпна підмова даних:** реляційна СУБД може підтримувати кілька мов і різні режими роботи, однак повинна існувати принаймні одна мова, оператори якої дозволяли б виражати такі конструкції, як: 1) визначення даних; 2) визначення представлень;

3) команди маніпулювання даними; 4) обмеження цілісності; 5) авторизація користувачів; 6) організація транзакцій.

- **Правило 7 – високорівневі операції вставки, оновлення та видалення:** здатність обробляти базові або похідні відношення як єдиний операнд повинна відноситися не тільки до процедур вилучення даних, але і до операцій вставки, оновлення та видалення даних.
- *ПРАВИЛА НЕЗАЛЕЖНОСТІ ВІД ДАНИХ (ПРАВИЛА 8, 9 ТА 11)*
- **Правило 8 – фізична незалежність від даних:** прикладні програми та засоби роботи з терміналами при внесенні будь-яких змін у способи зберігання даних або методи доступу до них повинні залишатися логічно непорушними.
- **Правило 9 – логічна незалежність від даних:** прикладні програми та засоби роботи з терміналами при внесенні в базові таблиці будь-яких змін, які не змінюють інформацію та теоретично не зачіпають прикладне програмне забезпечення, повинні залишатися логічно непорушними.
- **Правило 11 – незалежність від розподілу даних:** підмова маніпулювання даними в реляційній СУБД повинна дозволяти прикладним програмам та запитам залишатися логічно незмінними, незалежно від способу збереження даних: фізично, централізовано або у розподіленому вигляді.

Закріплення матеріалу

1. Пояснити основні поняття реляційної моделі: «відношення», «атрибут», «кортеж».
2. Пояснити поняття «представлення» та причини його використання.
3. Пояснити зміст основних і додаткових операцій реляційної алгебри.
4. Перелічити вимоги, що висуваються до РСУБД.

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.3 МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО, ЛОГІЧНОГО ТА ФІЗИЧНОГО ПРОЄКТУВАННЯ БД

Лекція 4

МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО ПРОЄКТУВАННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Мета лекції: розгляд концепції методології проєктування бази даних загалом та фази концептуального проєктування зокрема.

Зміст лекції

1. Вступ в методологію проєктування БД.
2. Загальний огляд процедури проєктування БД.
3. Методологія концептуального проєктування БД.

- ***Концептуальне проєктування*** – створення концептуального представлення бази даних, що включає визначення типів найважливіших сутностей та існуючих між ними зв'язків. Це перша фаза процеса проєктування, яка передбачає створення концептуальної моделі даних для аналізуємої частини предметної області. Модель створюється на основі інформації, що записана в специфікаціях вимог користувачів.
- Концептуальне проєктування не залежить від типу обраної цільової СУБД, набору прикладних програм, обраних мов програмування, типу платформи та інших особливостей фізичної реалізації.
- ***Логічне проєктування*** – перетворення концептуального представлення в логічну структуру бази даних, включаючи проєктування відношень.
- Логічна модель даних враховує особливості обраної моделі організації даних в цільовій СУБД, але при цьому ігноруються всі інші аспекти обраної СУБД (будь-які особливості фізичної організації її структур збереження даних та побудови індексів).

- Для перевірки коректності логічної моделі використовується метод нормалізації.
- Метод нормалізації – це метод, який гарантує, що виведені з існуючої моделі даних відношення не будуть володіти збитковістю даних, яка здатна викликати аномалії оновлення після їх фізичної реалізації.
- ***Фізичне проєктування*** – прийняття рішення про те, як логічна модель буде фізично реалізована в базі даних, що створюється за допомогою обраної СУБД.

Вступ в методологію проєктування БД

- ***Методологія проєктування*** – структурований підхід, який передбачає використання спеціалізованих процедур, технічних прийомів, інструментів, документації та націлений на підтримку спрощення процесу проєктування.
- Найважливіші фактори успішного завершення проєктування бази даних:
  - підтримувати постійний активний зв'язок з потенційними користувачами додатку;
  - дотримуватись при проведенні процедур моделювання даних рекомендацій, наведених під час обговорення методології;
  - розробляти систему, виходячи з існуючих характеристик даних;
  - створювати модель даних з урахуванням вимог підтримки їх структурної цілісності і узгодженості;
  - доповнювати процедури, що пропонуються даною методологією, технологічними прийомами концептуалізації, нормалізації та перевірки цілісності транзакцій;
  - використовувати діаграми для представлення моделі даних якомога ширше;
  - використовувати засоби мови DBDL (Database Design Language) для опису додаткових семантичних вимог до даних;
  - розробити словник опису даних;

– повертатися до вже виконаних раніше етапів, якщо це потрібно для досягнення оптимальних результатів.

Загальний огляд процедури проєктування бази даних

- Структура процедури проєктування бази даних:

Фаза 1 **Концептуальне проєктування бази даних**

Етап 1. Створення локальної концептуальної моделі даних, виходячи з уявлень предметної області кожного з типів користувачів.

Завдання 1.1. Визначення типів сутностей.

Завдання 1.2. Визначення типів зв'язків.

Завдання 1.3. Визначення атрибутів і зв'язування їх з типами сутностей та зв'язків.

Завдання 1.4. Визначення доменів атрибутів.

Завдання 1.5. Визначення атрибутів, які є потенційними і первинними ключами.

Завдання 1.6. Спеціалізація або генералізація типів сутностей (необов'язковий етап).

Завдання 1.7. Створення діаграми "сутність-зв'язок".

Завдання 1.8. Обговорення локальних концептуальних моделей даних з кінцевими користувачами.

Фаза 2 **Логічне проєктування бази даних (для реляційної моделі)**

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів.

Завдання 2.1. Перетворення локальної концептуальної моделі даних в локальну логічну модель.

Завдання 2.2. Визначення набору відношень, виходячи із структури локальної логічної моделі даних.

Завдання 2.3. Перевірка моделі за допомогою правил нормалізації.

Завдання 2.4. Перевірка моделі щодо транзакцій користувачів.

Завдання 2.5. Створення діаграм «сутність-зв'язок».

Завдання 2.6. Визначення вимог підтримки цілісності даних.

Завдання 2.7. Обговорення з кінцевими користувачами розроблених локальних логічних моделей даних.

Етап 3. Створення та перевірка глобальної логічної моделі даних.

Завдання 3.1. Злиття локальних логічних моделей даних в єдину глобальну модель даних.

Завдання 3.2. Перевірка глобальної логічної моделі даних.

Завдання 3.3. Перевірка можливостей розширення моделі у майбутньому.

Завдання 3.4. Створення остаточного варіанту діаграми «сутність-зв'язок».

Завдання 3.5. Обговорення глобальної логічної моделі даних з користувачами.

Фаза 3 **Фізичне проєктування бази даних (з використанням реляційної СУБД)**

Етап 4. Перенесення глобальної логічної моделі даних в середовище цільової СУБД.

Завдання 4.1. Проєктування основних таблиць в середовищі цільової СУБД.

Завдання 4.2. Реалізація бізнес-правил підприємства в середовищі цільової СУБД.

Етап 5. Проєктування фізичного представлення бази даних.

Завдання 5.1. Аналіз транзакцій.

Завдання 5.2. Вибір файлової структури.

Завдання 5.3. Визначення вторинних індексів.

Завдання 5.4. Аналіз необхідності введення контрольованої надлишковості даних.

Завдання 5.5. Визначення вимог дискової пам'яті.

Етап 6. Розробка механізмів захисту.

Завдання 6.1. Розробка користувальницьких представлень (видів).

Завдання 6.2. Визначення прав доступу.

Етап 7. Організація моніторингу та налаштування функціонування системи.

Методологія концептуального проєктування БД

- ***Етап 1.*** Створення локальної концептуальної моделі даних, виходячи з уявлень предметної області кожного з типів користувачів.

  Мета – створення локальної концептуальної моделі даних на основі уявлення предметної області кожного окремого типу користувачів.

  – *Завдання 1.1.* Визначення типів сутностей.

    Мета – визначення основних типів сутностей, що присутні у представленні даного користувача про предметну область додатку.

  – *Завдання 1.2.* Визначення типів зв'язків.

    Мета – визначення найважливіших типів зв'язків, що існують між сутностями, виділеними на попередньому етапі.

  – *Завдання 1.3.* Визначення атрибутів і зв'язування їх з типами сутностей та зв'язків.

    Мета – зв'язування атрибутів з відповідними типами сутностей або зв'язків.

  – *Завдання 1.4.* Визначення доменів атрибутів.

    Мета – визначення доменів для всіх атрибутів, присутніх в кожній локальній концептуальній моделі даних.

  – *Завдання 1.5.* Визначення атрибутів, які є потенційними і первинними ключами.

    Мета – визначення всіх потенційних ключів для кожного типу сутності і, якщо таких ключів виявиться кілька, вибір серед них первинного ключа.

- *Завдання 1.6.* Спеціалізація або генералізація типів сутностей (необов'язковий етап).

  Мета – визначення суперкласів і підкласів для типів сутностей (якщо це необхідно).

- *Завдання 1.7.* Створення діаграми «сутність-зв'язок».

  Мета – розробка діаграм «сутність-зв'язок» (ER-діаграм), що містять концептуальне відображення представлень користувача про предметну галузь додатку.

- *Завдання 1.8.* Обговорення локальних концептуальних моделей даних з кінцевими користувачами.

  Мета – обговорення локальних концептуальних моделей даних з кінцевими користувачами з метою отримання підтверджень, що дана модель коректно відображає уявлення користувача про додаток.

Закріплення матеріалу

1. Пояснити концепцію методології проєктування БД.
2. Пояснити поняття концептуального, логічного та фізичного проєктування БД.
3. Перелічити найважливіші фактори успішного завершення проєктування бази даних.
4. Опишіть мету етапу концептуального проектування.
5. З яких етапів складається фаза концептуального проєктування БД та які завдання виконуються на кожному з них?
6. Пояснити поняття домену.
7. Пояснити поняття потенційного та альтернативного ключів.
8. Які рекомендації надаються при виборі первинного ключа?
9. В чому полягають процедури спеціалізації та генералізації?

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.3 МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО, ЛОГІЧНОГО ТА ФІЗИЧНОГО ПРОЄКТУВАННЯ БД

Лекція 5

МЕТОДОЛОГІЯ ЛОГІЧНОГО ПРОЄКТУВАННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Мета лекції: розгляд фази логічного проєктування реляційних баз даних.

Зміст лекції

1. Загальний огляд методології логічного проєктування.
2. Етапи логічного проєктування.

- ***Логічне проєктування баз даних*** – процес конструювання загальної інформаційної моделі конкретної предметної області на основі окремих моделей даних користувачів, яка є незалежною від особливостей СУБД, що реально використовується, та інших фізичних умов.

Етапи логічного проєктування

- ***Етап 2.*** Побудова та перевірка локальної логічної моделі даних для окремих представлень кожного з типів користувачів.
- ***Етап 3.*** Створення та перевірка глобальної логічної моделі даних.

Методологія логічного проєктування БД

- ***Етап 2.*** Побудова та перевірка локальної логічної моделі даних для окремих представлень кожного з типів користувачів.
- ***Мета*** – побудова логічної моделі даних на основі концептуальної моделі даних, що відображає уявлення окремого користувача про предметну галузь застосування, перевірка отриманої моделі за допомогою методів нормалізації та контролю виконання транзакцій.
  - *Завдання 2.1.* Перетворення локальної концептуальної моделі даних в локальну логічну модель. Дії, що передбачаються даним

завданням: видалення зв’язків типу M:N; видалення рекурсивних зв’язків, видалення складних зв’язків, видалення зв’язків з атрибутами, видалення множинних атрибутів, повторна перевірка зв’язків типу 1:1, видалення надлишкових зв’язків.

- *Мета* – доопрацювання локальних концептуальних моделей з метою видалення з них небажаних елементів; перетворення отриманих моделей у локальні логічні моделі даних.
- *Завдання* 2.2. Визначення набору відношень, виходячи із структури локальної логічної моделі даних.
- *Мета* – визначення набору відношень на основі локальної логічної моделі даних.
- *Завдання* 2.3. Перевірка моделі за допомогою правил нормалізації. Аналізуються коректність об’єднання атрибутів в кожному з відношень.
- *Мета* – перевірка локальної логічної моделі даних з використанням технології нормалізації.
- *Завдання* 2.4. Перевірка моделі щодо транзакцій користувачів. Використання двох підходів перевірки локальної логічної моделі: 1) виконання перевірки того, що дана логічна модель надає всю інформацію (сутності, зв'язку та їх атрибути), необхідну для виконання кожної з транзакцій (практично це реалізується при підготовці опису вимог, висунутих кожною з транзакцій); 2) нанесення безпосередньо на ER-діаграми всіх шляхів, які будуть потрібні для виконання кожної з транзакцій.
- *Мета* – переконатися в тому, що локальна логічна модель даних дозволяє виконати всі транзакції, передбачені цим представленням користувача.
- *Завдання* 2.5. Створення діаграм "сутність-зв'язок".

- *Мета* – створення остаточного варіанту діаграм "сутність-зв'язок" (ER-діаграм), які є локальним логічним представленням даних, що використовуються окремими користувачами програми.
- *Завдання* 2.6. Визначення вимог підтримки цілісності даних. Типи обмеження цілісності даних: обов'язкові дані, обмеження для доменів атрибутів, цілісність сутностей, цілісність посилань, вимоги предметної області.
- *Мета* – визначення обмежень, що накладаються в представленнях користувачів вимогою збереження цілісності даних.
- *Завдання* 2.7. Обговорення розроблених локальних логічних моделей даних з кінцевими користувачами.
- *Мета* – переконатися, що створені локальні моделі даних точно відображають уявлення користувачів про предметну область додатку.

- ***Етап 3.*** Створення та перевірка глобальної логічної моделі даних.
- ***Мета*** – об'єднання окремих локальних логічних моделей даних в єдину глобальну логічну модель даних, що представляє ту частину підприємства, яка охоплюється даним додатком.
  - *Завдання 3.1.* Злиття локальних логічних моделей даних в єдину глобальну модель даних.
  - *Мета* – об'єднати окремі локальні логічні моделі даних в єдину глобальну логічну модель даних (наприклад, підприємства).
  - *Завдання 3.2.* Перевірка глобальної логічної моделі даних.
  - *Мета* – перевірка глобальної логічної моделі даних за допомогою методів нормалізації та контроль можливості виконання необхідних транзакцій.
  - *Завдання 3.3.* Перевірка можливостей розширення моделі у майбутньому.

- *Мета* – визначення ймовірності внесення будь-яких істотних змін у створену модель даних у майбутньому та оцінка того, наскільки дана модель пристосована для цього.
- *Завдання 3.4.* Створення остаточного варіанту діаграми "сутність-зв’язок".
- *Мета* – створення остаточного варіанту діаграми "сутність-зв'язок", що відображає глобальну логічну модель даних підприємства.
- *Завдання 3.5.* Обговорення глобальної логічної моделі даних з користувачами.
- *Мета* – переконатися, що створена глобальна логічна модель даних адекватно відображає моделюєму частину інформаційної структури підприємства.

Закріплення матеріалу

1. Назвати три основні фази процесу розробки баз даних.
2. Описати прзначення концептуального, логічного та фізичного проєктування БД.
3. З яких етапів складається фаза логічного проєктування БД та які завдання виконуються на кожному з них?
4. Які основні етапи включає процес нормалізації?
5. Пояснити, як методи нормалізації можуть використовуватись для перевірки логічної моделі даних та набору відношень, створених на основі цієї моделі.
6. Які існують підходи щодо впевненості в тому, що локальна логічна модель даних дозволяє виконати всі необхідні транзакції?
7. Пояснити призначення обмежень цілісності.
8. Які типи обмежень цілісності даних вам відомі?
9. Які типові задачі вирішуються при злитті локальних логічних моделей в єдину глобальну логічну модель?

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.3 МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО, ЛОГІЧНОГО ТА ФІЗИЧНОГО ПРОЄКТУВАННЯ БД

Лекція 6

МЕТОДОЛОГІЯ ФІЗИЧНОГО ПРОЄКТУВАННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Мета лекції: розгляд фази фізичного проєктування реляційних баз даних.

Зміст лекції

1. Загальний огляд методології фізичного проєктування БД.
2. Детальний розгляд етапів фізичного проєктування БД реляційного типу.

- Фаза фізичного проєктування бази даних не є ізольованою від інших – між логічним та фізичним проєктуванням є постійний зворотний зв'язок, який часто охоплює і розробку додатків користувача.

Етапи фізичного проєктування

- ***Етап 4.*** Перенесення глобальної логічної моделі даних в середовище цільової СУБД.
- ***Етап 5.*** Проєктування фізичного представлення бази даних.
- ***Етап 6.*** Розробка механізмів захисту.
- ***Етап 7.*** Організація моніторингу та налаштування функціонування системи.

Методологія фізичного проєктування реляційних БД

- ***Етап 4.*** Перенесення глобальної логічної моделі даних в середовище цільової СУБД.
- ***Мета*** – створення базової функціональної схеми реляційної бази даних на основі глобальної логічної моделі даних.
  - *Завдання 4.1.* Проєктування таблиць бази даних у середовищі цільової СУБД. Аналіз інформації про відношення, зібраної на

попередньому етапі, яка може міститись в словнику даних та у визначеннях відношень. Елементи, що включає кожне відношення, виділене в логічній моделі: ім'я відношення; список простих атрибутів, укладених в дужки; визначення первинного, альтернативних і зовнішніх ключів; визначення вимог цілісності посилань для будь-яких зовнішніх ключів. Інформація, яка має бути присутня в словнику даних для кожного атрибуту: визначення домену атрибуту; значення атрибуту за замовчуванням; припустимість значення NULL; визначення факту чи є атрибут похідним та способу обчислення його значення.

- *Мета* – визначення способу представлення в цільовій СУБД відношень, виокремлених у глобальній логічній моделі даних.
- *Завдання 4.2.* Реалізація бізнес-правил підприємства у середовищі цільової СУБД.
- *Мета* – реалізація бізнес-правил у середовищі цільової СУБД.

- ***Етап 5.*** Проєктування фізичного представлення бази даних.
- ***Мета*** – визначення файлової структури методів доступу, які будуть використовуватись для подання таблиць бази даних.
  - *Завдання 5.1.* Аналіз транзакцій. З'ясування інформації про кожну заплановану транзакцію: очікувана частота виконання; перелік відношень та атрибутів, що використовуються для даної транзакції; тип доступу до відношень та атрибутів; атрибути, які використовуються хоча б в одному з предикатів; атрибути, які використовуються для з'єднання відношень при виконанні запитів; обмеження, встановлені на час виконання транзакції.
  - *Мета* – визначення функціональних характеристик транзакцій, які будуть виконуватися в базі даних, що проєктується, та виділення найбільш важливих з них.
  - *Завдання 5.2.* Вибір файлової структури.

- *Мета* – визначення найбільш ефективного файлового представлення для кожної з таблиць бази даних.
- *Завдання 5.3.* Визначення вторинних індексів.
- *Мета* – визначення того, чи буде додавання вторинних індексів сприяти підвищенню продуктивності системи.
- *Завдання 5.4.* Аналіз необхідності введення контрольованої надмірності даних. Проведення, в разі необхідності, процедури денормалізації, враховуючи те, що вона ускладнює реалізацію, знижує гнучкість системи та прискорює вибірку даних, одночасно уповільнюючи їх оновлення.
- *Мета* – визначення того, чи сприятиме підвищенню продуктивності системи внесення контрольованої надмірності даних, що здійснюється за рахунок зниження вимог до рівня їх нормалізації.
  - ✓ *Завдання 5.4.1.* Аналіз доцільності використання похідних даних.
  - ✓ *Завдання 5.4.2.* Аналіз доцільності дублювання атрибутів або об’єднання окремих відношень.
- *Завдання 5.5.* Визначення вимог до дискової пам'яті. Оцінка об’єму дискового простору, який знадобиться для розміщення бази даних.
- *Мета* – визначення обсягу дискового простору, необхідного для розміщення бази даних.

- ***Етап 6.*** Розробка механізмів захисту.
- ***Мета*** – розробка механізмів захисту бази даних відповідно до вимог користувачів.
  - *Завдання 6.1.* Розробка представлень користувача (видів).
  - *Мета* – розробка представлень користувача (видів), які були виокремленні на першому етапі концептуального проєктування бази даних.
  - *Завдання 6.2.* Визначення прав доступу.

- *Мета* – визначення прав доступу до таблиць бази даних для кожного з представлень користувачів.

- ***Етап 7.*** Організація моніторингу та налаштування функціонування системи. Це дозволяє уникнути придбання додаткового обладнання та знизити вимоги до його конфігурації, підвищити рівень продуктивності за рахунок зменшення часу відповіді системи.
- ***Мета*** – моніторинг функціонування операційної системи та досягнутого рівня продуктивності бази даних з метою усунення помилкових проєктних рішень або відображення зміни вимог до системи.

Закріплення матеріалу

1. Пояснити концепцію методології проєктування БД.
2. Пояснити поняття концептуального, логічного та фізичного проєктування БД.
3. З яких етапів складається фаза фізичного проєктування БД та які завдання виконуються на кожному з них?
4. Які способи реалізації та вимог цілісності за посиланнями викладенні в Завданні 4.1?
5. Які показники можуть бути використані для оцінки досягнутої ефективності?
6. Які варіанти типів файлів вам відомі?
7. Які рекомендації надаються при визначенні набору необхідних вторинних індексів?
8. Яких переваг дозволяє домогтися виконання налаштування бази даних?

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.4 МОДЕЛЬ «СУТНІСТЬ-ЗВ'ЯЗОК». НОРМАЛІЗАЦІЯ

Лекція 7

ER-МОДЕЛЬ

Мета лекції: розгляд особливостей побудови ER-моделі.

Зміст лекції

1. Концепції ER-моделі.
2. Структурні обмеження.
3. Проблеми ER-моделювання.

- ***Модель "сутність-зв'язок"*** (Entity-Relationship model, або ER-модель) – високорівнева концептуальна модель даних для спрощення задачі проєктування баз даних.
- Основна мета розробки високорівневої моделі даних – створення моделі сприйняття даних користувачем та узгодження технічних аспектів, пов'язаних з проєктуванням бази даних.
- ***Розширена ER-модель*** (EER) – модель, яка включає всі концепції ER-моделі та додаткові концепції серіалізації / генералізації та категоризації.
- ***Серіалізація*** – це процес збільшення розходжень між окремими членами типу сутності за рахунок виділення їх відмінних характеристик.
- ***Генералізація*** – процес приведення відмінностей між сутностями до мінімуму шляхом виділення їх загальних характеристик.
- ***Категоризація*** – моделювання одного підкласу із зв'язком, який охоплює декілька різних суперкласі.
  - ***Суперклас*** – тип сутності, що включає різні підкласи, які необхідно подати в моделі даних.

- ***Підклас*** – член суперкласу, що виконує окрему роль.

Концепції ER-моделі

- ***Типи сутностей*** – об'єкт або концепція, які характеризуються на даному підприємстві як такі, що існують незалежно.
- ***Сутність*** – екземпляр типу сутності, який може бути ідентифікований унікальним чином.
- ***Слабкий тип сутності*** (дочірні, залежні, підлеглі) – тип сутності, існування якого залежить від якогось іншого типу сутності.
- ***Сильний тип сутності*** (батьківські, сутності-власники, домінантні) – тип сутності, існування якого не залежить від якогось іншого типу сутності.
- ***Атрибут*** – властивість типу сутності або типу зв'язку.
- ***Домен атрибута*** – набір значень, які можуть бути присвоєні атрибуту. Домен визначає всі потенційні значення, які можуть бути присвоєні атрибуту.
- ***Простий атрибут (атомарний)*** – це атрибут, що складається з одного компонента з незалежним існуванням. Прості атрибути не можуть бути розділеними на більш дрібні компоненти.
- ***Складовий атрибут*** – це атрибут, що складається з декількох компонентів, кожен з яких характеризується незалежним існуванням.
- ***Однозначний атрибут*** – це атрибут, який містить одне значення для однієї сутності.
- ***Багатозначний атрибут*** – це атрибут, який містить кілька значень для однієї сутності.
- ***Похідний атрибут*** – це атрибут, який представляє значення, похідне від значення пов'язаного з ним атрибута або деякої множини атрибутів, що належать деякому (не обов'язково даному) типу сутності.
- ***Ключ*** – елемент даних, який дозволяє унікально ідентифікувати окремі екземпляри деякого типу сутності.

- ***Потенційний ключ*** – атрибут або набір атрибутів, який унікально ідентифікує окремі екземпляри типу сутності.
- ***Первинний ключ*** – потенційний ключ, який обраний як первинний ключ.
- ***Складовий ключ*** – потенційний ключ, який складається з двох або більше атрибутів.
- ***Тип зв'язку*** – осмислена асоціація між сутностями різних типів. Тип зв’язку є набором асоціацій між двома та більше типами сутностей-учасників.
- ***Зв'язок*** – асоціація між сутностями, яка включає по одній сутності з кожного типу сутності, що бере участь у зв'язку.
- ***Ступінь зв'язку*** – це кількість сутностей, які охоплені даним зв'язком.
- ***Бінарний (binary) зв'язок*** – зв'язок зі ступенем два.
- ***Тернарний (ternary) зв'язок*** – зв'язок зі ступенем три.
- ***Кватернарний (quaternary) зв'язок*** – зв'язок зі ступенем чотири.
- ***Рекурсивний зв'язок (unary)*** – це зв'язок, в якому одні й ті самі сутності беруть участь декілька разів і в різних ролях.
- ***Рольові імена*** – це імена, які призначаються зв’язкам для визначення призначення кожної сутності, що бере в них участь.

Структурні обмеження

- ***Показник кардинальності*** – показник, який описує кількість можливих зв'язків для кожної з сутностей-учасниць.
- Найбільш поширеними є бінарні зв'язки з показниками кардинальності "один до одного" (1 : 1), "один до багатьох" (1 : M) "багато до багатьох" (M : N).
- ***Бізнес-правила (business rules) організації*** – правила, що визначають показники кардинальності.

- ***Ступінь участі*** – визначає, чи залежить існування деякої сутності від участі в зв'язку деякої іншої сутності.
- Ступінь участі є повною, якщо для існування деякої сутності потрібно існування іншої сутності, пов'язаної з нею певним зв'язком. В іншому випадку ступінь участі є частковою.
- ***Обов'язкова участь*** – це повна участь.
- ***Необов'язкова участь*** – це часткова участь.

Проблеми ER-моделювання

- ***Пастка розгалуження*** має місце в тому випадку, коли модель відображає зв'язок між типами сутностей, але шлях між окремими сутностями цього типу визначено неоднозначно. Пастка розгалуження виникає в тому випадку, коли два або більше зв'язків типу 1:M розгалужуються з однієї сутності.
- ***Пастка розриву*** з'являється в тому випадку, коли в моделі передбачається наявність зв'язку між типами сутностей, але не існує шляху між окремими сутностями цих типів. Пастка розриву може виникнути за наявності зв'язку з частковою участю, який утворює частину шляху між зв'язаними сутностями.

Закріплення матеріалу

1. В чому полягає основна мета розробки ER-моделі?
2. Що включають основні концепції ER-моделі?
3. Що таке типи сутності та як вони класифікуються?
4. Що таке атрибут та домен атрибуту?
5. Які різновиди атрибутів та ключів вам відомі?
6. Що таке зв'язок та ступінь зв'язку?
7. Назвати основні типи обмежень, що накладаються на зв'язки. Пояснити їх суть.
8. Які проблеми ER-моделювання вам відомі?

РОЗДІЛ 1
Основні відомості про бази даних та СУБД

Тема 1.4 МОДЕЛЬ «СУТНІСТЬ-ЗВ'ЯЗОК». НОРМАЛІЗАЦІЯ

Лекція 8

ПОНЯТТЯ НОРМАЛІЗАЦІЇ БАЗИ ДАНИХ

Мета лекції: розгляд процесу нормалізації баз даних.

Зміст лекції

1. Мета нормалізації.
2. Надмірність даних та аномалії оновлення.
3. Процес та форми нормалізації.

Мета нормалізації

- ***Нормалізація*** – це метод створення на основі вимог до даних набору відношень із заданими властивостями.
- Нормалізація зазвичай виконується у вигляді послідовності тестів для деякого відношення задля перевірки його відповідності (або невідповідності) вимогам заданої нормальної форми.
- Процес нормалізації – це формальний метод, який дозволяє ідентифікувати відношення на основі їх первинних ключів і функціональних залежностей, існуючих між їх атрибутами.

Надмірність даних та аномалії оновлення

- ***Аномалії оновлення*** – це проблеми, які можуть виникати при роботі з відношеннями, що містять надлишкові дані. Аномалії оновлення поділяються на аномалії вставки, видалення та модифікації.
- ***Функціональна залежність*** (functional dependency) – це одне з основних понять нормалізації, яке описує зв'язок між атрибутами.

- ***Детермінант функціональної залежності*** – це атрибут або група атрибутів, розміщена на діаграмі функціональної залежності зліва від символу стрілки.

Процес та форми нормалізації

- ***Нормалізація*** – це формальний метод аналізу відношень на основі їх первинного ключа (або потенційних ключів, як у випадку НФБК) та існуючих функціональних залежностей. Він включає ряд правил, які можуть використовуватися для перевірки окремих відношень таким чином, щоб вся база даних могла бути нормалізована до бажаного ступеня нормалізації. Якщо деяка вимога не задовольняється, то відношення, що порушує цю вимогу має бути декомпозовано на відношення, кожне з яких (окремо) задовольняє всім вимогам нормалізації.
- ***Ненормалізована форма*** (ННФ) – це таблиця, що містить одну або кілька повторюваних груп даних.
- ***Перша нормальна форма*** (1НФ) – це відношення, в якому на перетині кожного рядка та кожного стовпця міститься тільки одне значення.
- ***"Вирівнювання" ("flattening") таблиці*** – це підхід виключення повторюваних груп з ненормалізованих таблиць.
- ***Друга нормальна форма*** (2НФ) – це відношення, яке знаходиться в першій нормальній формі і кожен атрибут якого, що не входить до складу первинного ключа, характеризується повною функціональною залежністю від цього первинного ключа.
- ***Повна функціональна залежність*** A→B – це функціональна залежність, при якій видалення будь-якого атрибуту з A призводить до втрати цієї залежності.
- ***Часткова функціональна залежність*** A→B – це функціональна залежність, яка зберігається при видаленні в A довільного атрибуту.

- ***Третя нормальна форма*** (3НФ) – це відношення, яке знаходиться в першій та другій нормальних формах та не має атрибутів, що не входять у первинний ключ, та які б знаходились у транзитивній функціональній залежності від цього первинного ключа.
- ***Транзитивна залежність.*** Якщо для атрибутів A, B та C деякого відношення існують залежності виду A→B та B→C, то кажуть, що атрибут C транзитивно залежить від атрибута A через атрибут B (за умови, що атрибут A функціонально не залежить ні від атрибута B, ні від атрибута C).
- ***Нормальна форма Бойса-Кодда*** (НФБК) – це відношення, в якому кожен його детермінант є потенційним ключем.
- ***Четверта нормальна форма*** (4НФ) – це відношення, яке знаходиться в нормальній формі Бойса-Кодда та не містить нетривіальних багатозначних залежностей.
- ***Багатозначна залежність*** – це залежність між атрибутами A, B та C деякого відношення, за якої для кожного значення атрибута A існують відповідні набори значень атрибутів B та C, причому обидва цих набори не залежать один від одного.
- ***П'ята нормальна форма*** (5НФ) – це відношення, яке не містить залежностей з'єднання.
- ***Залежність з'єднання*** – це ситуація, за якої декомпозиція відношення може супроводжуватися генерацією помилкових рядків при зворотному з'єднанні декомпозованих відношень за допомогою операції природного з'єднання.

Закріплення матеріалу

1. В чому полягає мета нормалізації?
2. Які нормальні форми вам відомі і в чому особливості кожної з них?

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.1 ЗАХИСТ БД, КЕРУВАННЯ ТРАНЗАКЦІЯМИ, ОБРОБКА ЗАПИТІВ

Лекція 9

ЗАХИСТ БАЗ ДАНИХ

Мета лекції: розгляд засобів захисту баз даних.

Зміст лекції

1. Комп’ютерні засоби контролю.
2. Некомп’ютерні засоби контролю.
3. Захист ПК.
4. СУБД та захист у Web.

- ***Захист бази даних*** – забезпечення захищеності бази даних проти будь-яких навмисних або ненавмисних загроз за допомогою різних комп'ютерних та некомп'ютерних засобів.
- ***Небезпека*** – будь-яка ситуація або подія, навмисна або ненавмисна, яка здатна несприятливо вплинути на систему і, як наслідок, на всю організацію.
- Потенційні небезпеки:
  - викрадення та фальсифікація даних;
  - втрата конфіденційності;
  - порушення недоторканості особових даних;
  - втрата цілісності;
  - втрата доступності.

Комп’ютерні засоби контролю

- Комп’ютерні засоби контролю:
  - авторизація користувачів;

- представлення;
- створення резервних копій та відновлення;
- підтримка цілісності;
- шифрування;
- допоміжні процедури.

- ***Авторизація*** – надання прав (або привілеїв), що дозволяють їх власнику мати законний доступ до системи або її об'єктів.
- ***Аутентифікація*** – механізм визначення того, чи є користувач тим, за кого себе видає.
- ***Представлення*** – це динамічний результат однієї або декількох реляційних операцій з базовими відношеннями з метою створення деякого іншого відношення.
- ***Резервне копіювання*** – періодично виконувана процедура отримання копії бази даних і її файлу журналу на носії, збереженому окремо від системи.
- ***Ведення журналу*** – процедура створення та обслуговування файлу журналу, що містить відомості про всі зміни, внесені в базу даних з моменту створення останньої резервної копії, та призначеного для забезпечення ефективного відновлення системи у випадку її відмови.
- ***Контрольна точка*** – момент синхронізації між станом бази даних та станом журналу виконання транзакцій.
- ***Підтримка цілісності*** – запобігання переходу даних у неузгоджений стан.
- ***Шифрування*** – кодування даних з використанням спеціального алгоритму, внаслідок чого дані стають недоступними для читання будь-якою програмою, яка не має ключа дешифрування.
- ***Допоміжні процедури*** для захисту даних у середовищі СУБД:
  - аудит;
  - встановлення нового прикладного програмного забезпечення;

- встановлення або модернізація системного програмного забезпечення.

Некомп’ютерні засоби контролю

- Некомп'ютерні засоби контролю:
  - заходи забезпечення безпеки та планування захисту від непередбачених обставин;
  - контроль за персоналом;
  - захист приміщень та сховищ;
  - гарантійні угоди;
  - договори про супровід;
  - контроль за фізичним доступом.

Захист ПК

- Способи організації захисту персонального комп’ютера:
  - використання індивідуальних ідентифікаторів користувачів та відповідних особистих паролів;
  - спеціальні сховища для захищеного зберігання будь-яких необхідних даних, програмного забезпечення та комп'ютерного обладнання;
  - особиста відповідальність кожного користувача за створення копій даних та зберігання програмного забезпечення і комп'ютерного обладнання;
  - чіткі вказівки про процедури, які повинні застосовуватися до будь-якого програмного забезпечення, перед тим як воно буде допущено до встановлення системи.

СУБД та захист у Web

- Заходи захисту, пов'язані з використанням СУБД у середовищі Web:
  - проксі-сервери;
  - брандмауери;

- використання цифрового підпису;
- алгоритми створення дайджесту повідомлень цифрового підпису;
- цифрові сертифікати;
- використання церберів;
- використання технологій SSL та SHTTP;
- концепції мов програмування для захисту Web-додатків;
- засоби захисту технології ActiveX.

- ***Проксі-сервер*** – це комп'ютер, який розміщується між Web-браузером та Web-сервером. У загальному випадку проксі-сервери призначені для вирішення задач підвищення продуктивності та фільтрації запитів.
- ***Брандмауер*** – це система, призначена для захисту від несанкціонованого доступу до (або з) закритої мережі. Брандмауери можуть бути програмними, апаратними або комбінованими.
- Основні типи брандмауерів:
  - пакетний фільтр;
  - шлюз додатка;
  - шлюз на рівні ланцюга;
  - проксі-сервер.
- ***Цифровий підпис*** використовується для перевірки істинності походження даних та складається з двох частин: рядка бітів, який обчислений на основі "підписаних" даних, та закритого ключа приватної особи або організації, яка "підписала" ці дані.
- ***Цифровий сертифікат*** – це доповнення електронного повідомлення, яке використовується з метою забезпечення безпеки, – найчастіше для того, щоб довести, що відправник повідомлення дійсно є тим, за кого себе видає.
- ***Цербер*** – це сервер захищених облікових імен та паролів користувачів. Цербер ідентифікує і встановлює справжність користувача.

- ***Протокол Secure Sockets Layer*** (SSL) – це протокол, який створює безпечне з'єднання між клієнтом та сервером, по якому може безпечно передаватися будь-яка кількість даних.
- ***Протокол Secure HTTP*** (S-HTTP) – це протокол, який призначений для безпечної передачі окремих повідомлень.
- ***Концепція "пісочниця"*** гарантує заборону на доступ до системних ресурсів несанкціонованого і, можливо, зловмисного додатку. Для реалізації концепції "пісочниці" використовуються завантажувач класів, верифікатор байт-коду та менеджер захисту.
- ***ActiveX*** – це фреймворк для визначення програмних компонентів, придатних до використання з програм, написаних на різних мовах програмування.

Закріплення матеріалу

1. Пояснити призначення та область застосування поняття «захист бази даних».
2. Перелічити шість типів небезпек, яким можуть піддаватися системи з базами даних, і вказати для кожної з них можливі засоби контролю та протидії.
3. Навести приклади зв’язку між можливими варіантами порушень системи захисту та їх наслідками.
4. Пояснити наступні поняття: авторизація користувачів, резервне копіювання, шифрування, захист від непередбачених обставин, контроль за персоналом, гарантійні угоди, недоторканість особових даних, захист особистих даних.
5. Які засоби контролю належать до комп’ютерних?
6. Які засоби контролю належать до некомп’ютерних?
7. Які заходи захисту, пов'язані з використанням СУБД у середовищі Web, вам відомі?

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.1 ЗАХИСТ БД, КЕРУВАННЯ ТРАНЗАКЦІЯМИ, ОБРОБКА ЗАПИТІВ

Лекція 10

УПРАВЛІННЯ ТРАНЗАКЦІЯМИ

Мета лекції: розгляд трьох основних функціональних можливостей, які повинна надавати сучасна СУБД.

Зміст лекції

1. Підтримка транзакцій.
2. Керування паралельністю.
3. Впорядкованість і відновлення БД.
4. Покращенні моделі транзакцій.

- Основні функції сучасної СУБД:
  - механізми підтримки транзакцій;
  - служби управління паралельністю;
  - засоби відновлення баз даних.
- Засоби відновлення та механізм управління паралельністю доступу призначені для захисту баз даних від переходу даних в неузгоджений стан або їх втрати.

Підтримка транзакцій

- ***Транзакція*** – це дія або серія дій, що виконуються одним користувачем або прикладною програмою і здійснюють доступ бази даних або змінюють її вміст.
- Транзакція є логічною одиницею роботи, що виконується в базі даних і може бути представлена окремою програмою, частиною алгоритму

програми або окремою командою та включати довільну кількість операцій, які виконуються в базі даних.

- Будь-яка транзакція завжди повинна переводити базу даних з одного узгодженого стану в інший, хоча допускається, що узгодженість стану бази буде порушуватися під час виконання транзакції.
- Будь-яка транзакція завершується одним з двох можливих способів: COMMIT (фіксація) або ROLLBACK (відкат).
- Жодна СУБД не володіє внутрішньою можливістю встановити, які саме зміни повинні бути сприйняті як єдине ціле, що утворить одну логічну транзакцію.
- ***COMMIT*** – це процес фіксації результатів транзакції в базі даних у разі її успішного завершення.
- ***ROLLBACK*** – це процес відновлення бази даних, тобто приведення її в той узгоджений стан, в якому вона перебувала до початку виконання транзакції, що відбулося невдало.
- ***Компенсуюча транзакція*** – це транзакція, яка скасовує дії, виконані зафіксованою помилковою транзакцією.
- Для позначення границь окремих транзакцій використовуються наступні оператори: BEGIN TRANSACTION, COMMIT і ROLLBACK (або їх еквіваленти). Якщо ці обмежувачі не були використані, вся виконувана програма розцінюється як єдина транзакція і СУБД автоматично виконає команду COMMIT при нормальному завершенні цієї програми. Аналогічно, у разі її аварійного завершення в базі даних автоматично буде виконана команда ROLLBACK.
- Основні властивості транзакцій:
  - *Атомарність* – будь-яка транзакція є неподільною одиницею роботи, яка може бути або виконана вся цілком, або не виконана зовсім;

- *Узгодженість* – кожна транзакція повинна переводити базу даних з одного узгодженого стану в інший узгоджений стан;
- *Ізольованість* – усі транзакції виконуються незалежно одна від одної;
- *Тривалість* – результати зафіксованої транзакції повинні зберігатися в базі даних постійно і не повинні бути втрачені в результаті наступних збоїв.

Управління паралельністю

- ***Управління паралельністю*** – це процес організації одночасного виконання в базі даних різних операцій, що гарантує виключення їх взаємного впливу один на одного.
- ***Протокол управління паралельністю*** – це протокол, який реалізується в СУБД для запобігання небажаного впливу призначених для користувача процесів один на одного.
- Потенційні проблеми, які можуть мати місце при паралельному виконанні транзакцій:
  - проблема втраченого оновлення;
  - проблема залежності від нефіксованих результатів;
  - проблема неузгодженої обробки.

Впорядкованість і відновлення БД

- ***Впорядкованість*** – засіб, здатний допомогти у виявленні тих транзакцій, які гарантовано не викличуть порушення узгодженості даних при одночасному виконанні.
- ***Графік*** – послідовність запуску операцій багатьох паралельно виконуваних транзакцій, що зберігає черговість виконання операцій в кожній окремій транзакції.

- ***Послідовний графік*** – графік, в якому операції кожної з транзакцій виконуються строго послідовно і не можуть чергуватися з операціями, що виконуються в інших транзакціях.
- ***Непослідовний графік*** – графік, в якому чергуються операції з деякого набору одночасно виконуваних транзакцій.
- Суть впорядкування полягає у знаходженні таких непослідовних графіків, які дозволять транзакціям виконуватися паралельно, але без надання взаємовпливу одна на одну, і привести базу даних в стан, який може бути досягнуто при використанні послідовного графіка.
- Порядок виконання операцій читання і запису даних задля досягнення впорядкованості:
  - якщо дві транзакції тільки зчитують деякий елемент даних, вони не будуть конфліктувати між собою і порядок їх виконання не має значення;
  - якщо дві транзакції зчитують або записують абсолютно незалежні елементи даних, вони не будуть конфліктувати між собою і порядок їх виконання не має значення;
  - якщо одна транзакція записує елемент даних, а інша транзакція цей самий елемент даних зчитує або записує, порядок їх виконання має істотне значення.
- Типи впорядкування:
  - конфліктне впорядкування;
  - впорядкування за переглядом.
- ***Конфліктно впорядкований графік*** – це графік, в якому порядок виконання будь-яких конфліктуючих операцій відповідає їх розміщенню в послідовному графіку. Для перевірки конфліктної впорядкованості можна використовувати граф передування.
- ***Впорядкований за переглядом графік*** – це графік, еквівалентний за переглядом деякому послідовному графіку.

- Кожен конфліктно впорядкований графік в той же час є впорядкованим за переглядом, проте зворотнє твердження невірне.
- ***Впорядковані графіки*** – це графіки, які дозволяють зберегти узгодженість бази даних в припущенні, що жодна з транзакцій цього графіка не буде скасована.
- ***Відновлюваний графік*** – графік, в якому для кожної пари транзакцій T_i і T_j виконується правило: якщо транзакція T_i зчитує елемент даних, попередньо записаний транзакцією T_j, то фіксація результатів транзакції T_i повинна виконуватися до фіксації результатів транзакції T_j.
- Методи управління паралельністю підтримують консервативні та оптимістичні підходи.
- ***Консервативні (песимістичні) методи*** – це методи, які відкладають виконання транзакцій, здатних в майбутньому в той чи інший момент часу увійти в конфлікт з іншими транзакціями.
- ***Методи блокування*** – це методи, які зазвичай усувають можливі конфлікти за допомогою переведення транзакцій в стан очікування.
- ***Блокування*** – процедура, яка використовується для управління паралельним доступом до даних і дозволяє відхилити спроби отримання доступу до даних з боку інших транзакцій у разі, коли деяка транзакція вже отримала до них доступ.
- Фундаментальний принцип методів блокування: транзакція повинна вимагати виконати блокування для читання або для запису деякого елемента даних перед тим, як вона зможе виконати в базі даних відповідну операцію читання або запису.
- ***Протокол двофазного блокування (2PL)*** – це протокол, який визначає моменти встановлення та зняття блокування для кожної з транзакцій і використовується для забезпечення впорядкованості.

- Фундаментальний принцип двофазного блокування: транзакція виконується по протоколу двофазного блокування, якщо в ній всі операції блокування передують першій операції розблокування.
- ***Методи з використанням часових міток*** – це методи, які не передбачають будь-якого очікування – залучені в конфлікт транзакції просто скасовуються, після чого запускаються заново.
- ***Часова мітка*** – унікальний ідентифікатор, який створюється СУБД з метою позначення відносного моменту часу запуску транзакції.
- ***Метод використання часових міток*** – протокол управління паралельністю, основна мета якого – встановлення глобальної черговості виконання транзакцій, при якій транзакції з меншим значенням часової мітки мають більш високий пріоритет при розв'язанні виникаючих конфліктів.
- ***Правило запису Томаса*** – це модернізований базовий протокол впорядкування за часовими мітками, який дозволяє досягти менш жорсткої конфліктної впорядкованості за рахунок скасування застарілих операцій запису.
- ***Правило ігнорування застарілого запису*** – це ситуація, коли новіша транзакція вже перезаписала значення елемента даних і значення, яке старіша транзакція збирається записати в даний елемент, було обчислено на основі застарілого вихідного значення даного елемента, тому операція запису цілком безпечно може бути проігнорована.
- ***Оптимістичні методи*** – це методи, які будуються на припущенні, що ймовірність конфлікту невисока, і тому допускають асинхронне виконання транзакцій, а перевірка на наявність конфлікту відкладається на момент їх завершення і фіксації в базі даних.
- Фази оптимістичного підходу: читання, перевірки, запису.

- Оптимістичні технології ефективні в разі незначної кількості конфліктів в системі, але можуть викликати відкат окремих транзакцій, при цьому виникнення каскадного відкату виключається.
- ***Відновлення бази даних*** – це процедура повернення бази даних до деякого коректного (узгодженого) стану, що вживається в разі руйнування системи.
- Причини руйнування системи:
  - відмови обладнання;
  - програмні помилки;
  - збої носіїв інформації.
- Функції відновлення бази даних:
  - механізм резервного копіювання;
  - засоби ведення журналу;
  - створення контрольних точок;
  - менеджер відновлення.
- ***Контрольна точка*** – момент синхронізації між базою даних і журналом реєстрації транзакцій.
- Методи відновлення:
  - метод відновлення з використанням відкладеного поновлення;
  - метод відновлення з використанням негайного поновлення;
  - метод тіньових сторінок.

Покращені моделі транзакцій

- ***Модель вкладених транзакцій*** – це модель, яка розглядає транзакцію, як набір пов'язаних підзадач (субтранзакції), кожна з яких також може складатися з довільної кількості субтранзакцій.
- ***Хроніка*** – це послідовність (пласких) транзакцій, які можуть чергуватися з іншими транзакціями.

- ***Модель багаторівневих транзакцій*** – це модель, яка вимагає, щоб завершення роботи субтранзакцій виконувалося від низу до верху, в напрямку транзакції верхнього рівня.
- ***Модель відкритих вкладених транзакцій*** – це модель, в якій атомарность порушується і часткові результати виконання субтранзакцій можуть бути доступні поза транзакцією.
- ***Динамічна реструктуризація*** – це модель, яка об’єднує операції розбиття і об'єднання транзакцій
- ***Робочий потік*** – це вид діяльності, який передбачає координоване виконання множини завдань, що здійснюються різними суб'єктами обробки.

Закріплення матеріалу

1. Що таке транзакція?
2. Які властивості транзакцій вам відомі?
3. Що таке протокол управління паралельністю?
4. Які потенційні проблеми можуть мати місце при паралельному виконанні транзакцій?
5. Що таке впорядкованість?
6. Яким має бути порядок виконання операцій читання і запису даних задля досягнення впорядкованості?
7. Які типи впорядкування вам відомі? В чому їх особливості?
8. Які методи управління паралельністю вам відомі?
9. Що таке відновлення бази даних?
10. Які функції відновлення типової СУБД вам відомі?
11. Які вам відомі покращені моделі транзакцій? В чому їх особливості?

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.1 ЗАХИСТ БД, КЕРУВАННЯ ТРАНЗАКЦІЯМИ, ОБРОБКА ЗАПИТІВ

Лекція 11

ОБРОБКА ЗАПИТІВ

Мета лекції: розгляд підходів щодо обробки та оптимізації запитів .

Зміст лекції

1. Огляд методів обробки запитів.
2. Декомпозиція запитів.
3. Методи оптимізації.

Огляд методів обробки запитів

- ***Обробка запитів*** – це дії, які необхідні для отримання потрібної інформації з бази даних шляхом перетворення запиту, записаного мовою високого рівня, в коректну та ефективну послідовність операцій (план запиту), записану мовою низького рівня, що реалізує операції реляційної алгебри.
- ***План виконання запиту*** – це конкретна та ефективна послідовність операцій.
- ***Оптимізація запиту*** – це процедура вибору найбільш ефективного плану виконання запиту.
- Основні етапи процесу обробки запитів:
  - декомпозиція;
  - оптимізація;
  - генерація коду;
  - виконання.

- ***Динамічна оптимізація запитів*** – це виконання декомпозиції та оптимізації при кожному виклику запиту.
- Перевага динамічної оптимізації: вся інформація, яка використовується в процесі вибору оптимальної стратегії, є актуальною саме на поточний момент часу.
- Основний недолік динамічної оптимізації: відносне зниження ефективності обробки запитів через виконання сканування, перевірки та оптимізації при обробці кожного із запитів.
- ***Статична оптимізація запитів*** – це одноразове виконання етапів сканування, перевірки та оптимізації.
- Основна перевага статичної оптимізації: усунення додаткового навантаження через виконання процедур оптимізації в процесі виконання запиту.
- Основний недолік статичної оптимізації: обрана стратегія виконання запиту, яка була оптимальною при компіляції запиту, на момент виконання запиту може виявитися не найоптимальнішою.

Декомпозиція запитів

- ***Декомпозиція*** – це перетворенні запиту, який написаний мовою високого рівня у вираз реляційної алгебри з наступною перевіркою його синтаксичної семантичної коректності.
- Стадії декомпозиції:
  - аналіз;
  - нормалізація;
  - семантичний аналіз;
  - спрощення;
  - реструктуризація запиту.

Аналіз

- Операції стадії аналізу:

- виконання лексичного і синтаксичного аналізу запиту з використанням методів, що застосовуються в компіляторах мов програмування високого рівня;
- уточнення наявності у системному каталозі визначень для зазначених в запиті відношень та атрибутів;
- контроль спроможності затребуваних в запиті операцій над різними об’єктами бази даних бути застосованими до об'єктів відповідного типу;
- перетворення запиту, написаного мовою високого рівня, до внутрішнього надання у формі дерева запиту для виконання подальшої обробки.

Нормалізація

- Операції стадії нормалізації:
  - перетворення запита в нормалізовану форму з метою спрощення маніпулювання ним.
- ***Кон'юнктивна нормальна форма*** (AND, $\wedge$) – це послідовність кон'юнкцій, пов'язаних між собою операторами диз'юнкції. Кожна кон'юнкція містить один або більше термів, з'єднаних операторами кон'юнкції. Вибірка, виконана на основі кон'юнктивного предиката, містить тільки ті кортежі, які задовольняють всім кон'юнкціям, що входять у предикат.
- ***Диз'юнктивна нормальна форма*** (OR, $\vee$) – це послідовність диз'юнкцій, з'єднаних між собою операторами кон'юнкції. Кожна диз'юнкція містить один або більше термів, з'єднаних операторами кон'юнкції. Вибірка, виконана на основі диз'юнктивного предиката, містить об'єднання всіх кортежів, які задовольняють будь-якій з кон'юнкцій, що входять у предикат.

Семантичний аналіз

- Операції стадії семантичного аналізу:

- аналіз та відхилення нормалізованих запитів, які некоректно сформульовані або містять суперечливі вимоги.

- ***Некоректно сформульований запит*** – це запит, виконання якого ніколи не зможе призвести до створення результуючого набору даних.
- ***Суперечливий запит*** – це запит, предикат якого ніколи не зможе бути задоволений будь-яким кортежем.

Спрощення

- Операції стадії спрощення:
  - розгляд існуючих обмежень доступу;
  - виявлення надлишкових кваліфікаторів;
  - розкриття визначень представлень, що використовуються;
  - виключення загальних підвиразів;
  - врахування вимог підтримки цілісності даних;
  - отримання користувачем необхідних прав доступу;
  - перетворення запиту в семантично еквівалентну форму.

Реструктуризація

- Операції стадії реструктуризації:
  - перетворення запиту задля забезпечення найбільш ефективного його виконання.

Методи оптимізації

- ***Евристичний підхід до оптимізації запитів*** – це метод оптимізації запитів, яким передбачене використання правил трансформації для перетворення виразів реляційної алгебри в еквівалентну форму, обробка якої буде більш ефективною.
- ***Підхід до оптимізації запитів на основі оцінки вартості операцій реляційної алгебри*** – це метод порівняння показників різних стратегій, який виконується на основі їх відносної вартості, причому вибирається

той з варіантів, який дозволяє мінімізувати використання системних ресурсів.

- ***Матеріалізація*** – це підхід, який полягає в тому, що результати однієї операції до початку наступної операції зберігаються у тимчасових відношеннях.
- ***Конвеєрна обробка*** – це метод для підвищення ефективності запитів, при якому передача результатів однієї операції на обробку іншої операції здійснюється без створення тимчасових відношень, призначених для зберігання проміжних результатів.
- Механізм конвеєрної обробки реалізується за допомогою групи окремих процесів або потоків, які виконуються в рамках обчислювального середовища СУБД.
- Основний недолік методу конвеєрної обробки: вхідні дані не можуть бути доступні операціям одночасно у всьому об'ємі, що обмежує можливості вибору алгоритмів обробки.

Закріплення матеріалу

1. Що таке обробка запиту?
2. Які основні етапи процесу обробки запитів вам відомі? Опишіть кожен з них.
3. Які основні стадії етапу декомпозиції вам відомі?
4. Пояснити різницю між кон’юктивною та диз’юнктивною нормальними формами.
5. Що таке оптимізація запиту?
6. В чому полягає різниця між динамічною та статичною оптимізацією запитів?
7. Які методи оптимізації вам відомі? Опишіть кожен з них.

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.2 НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ

Лекція 12

КОНЦЕПЦІЇ ТА РОЗРОБКА РОЗПОДІЛЕНИХ СУБД

Мета лекції: розгляд основних концепцій розподіленої обробки даних.

Зміст лекції

1. Огляд основних концепцій розробки розподілених СУБД.
2. Розгляд організації та роботи комп’ютерних мереж.
3. Огляд функцій та архітектури розподілених СУБД.
   3.1 Функції розподілених СУБД.
   3.2. Архітектура СУБД.
4. Дванадцять правил Дейта для розподілених СУБД.

Огляд основних концепцій розробки розподілених СУБД

- ***Розподілена база даних*** – це набір логічно пов'язаних між собою даних (і їх описів), які фізично розподілені у деякій комп'ютерній мережі.
- ***Розподілена СУБД*** – це програмний комплекс, призначений для управління розподіленими базами даних, що дозволяє зробити розподіл інформації прозорим для кінцевого користувача.
- ***Локальні додатки*** – це додатки, які не вимагають доступу до даних на інших сайтах.
- ***Глобальні додатки*** – це додатки, які вимагають доступу до даних на інших сайтах. В розподіленій СУБД повинен існувати хоча б один глобальний додаток.
- ***Реплікація даних*** – це процес, під яким розуміють копіювання даних з одного джерела в інше та навпаки.

- ***Розподілена обробка даних*** – це обробка з використанням централізованої бази даних, доступ до якої може здійснюватися з різних комп'ютерів мережі.
- ***Паралельна СУБД*** – це система управління базою даних, яка функціонує з використанням декількох процесорів та пристроїв жорстких дисків, що дозволяє їй (за можливості) з метою підвищення загальної продуктивності обробки розпаралелювати виконання деяких операцій.
- Основні типи архітектури паралельних СУБД:
  - системи з поділом пам'яті;
  - системи з поділом дисків;
  - системи без поділу.
- ***Системи з поділом пам'яті*** – це системи, які складаються з компонентів, що тісно пов'язані між собою, в число яких входить кілька процесорів, що поділяють загальну системну пам'ять. Ця архітектура ще називається симетричною багатопроцесорною обробкою.
- ***Системи без поділу*** (масова паралельна обробка) – це системи, які використовують схему, де кожен процесор, що є частиною системи, має свою власну оперативну дискову пам'ять. Ця архітектура ще називається масовою паралельною обробкою.
- ***Системи з поділом дисків*** – це системи, які будуються з менш тісно пов'язаних між собою компонентів.
- ***Гомогенні розподілені СУБД*** – це системи, в яких всі сайти використовують один і той самий тип СУБД.
- ***Гетерогенні розподілені СУБД*** – це системи, в яких на сайтах можуть функціонувати різні типи СУБД, що використовують різні моделі даних.
- ***Мультибазова система*** – це розподілена система управління базами даних, в якій управління кожним з сайтів здійснюється абсолютно автономно, тобто кінцеві користувачі різних сайтів можуть отримувати

доступ та спільно використовувати дані без необхідності фізичної інтеграції існуючих баз даних. Мультибазова СУБД працює тільки з глобальною схемою.

- ***Схема експорту*** – це схема, за допомогою якої системний адміністратор локальної бази даних надає дозвіл зовнішнім користувачам на доступ до певних елементів своєї бази даних.
- ***Нефедеральна мультибазова система*** – це мультибазова система, що не має локальних користувачів.
- ***Федеральна мультибазова система*** – це мультибазова система, що виглядає як розподілена система для віддалених користувачів та як централізована система для локальних.

Розгляд організації та роботи комп'ютерних мереж

- ***Комп'ютерна мережа*** – це сукупність автономних комп'ютерів, з'єднаних між собою і здатних обмінюватися інформацією.
- ***Локальна мережа*** (local area network, LAN) – це мережа, яка утворюється при з'єднанні комп'ютерів, що належать одному і тому самому сайту.
- ***Глобальна мережа*** (wide area network, WAN) – це мережа, яка використовується для з'єднання комп'ютерів різних локальних мереж, що знаходяться на віддаленій відстані один від одного.
- ***Одноадресна мережа*** – це мережа, в якій при необхідності розіслати повідомлення всім її вузлам знадобиться відправити кілька окремих повідомлень.
- ***Широкомовна мережа*** – це мережа, в якій всі вузли отримують всі повідомлення, проте в кожному з повідомлень присутній префікс, що ідентифікує вузол-одержувач, а всі інші вузли мережі просто проігнорують повідомлення, яке надійшло.

Огляд функцій та архітектури розподілених СУБД

Функції розподілених СУБД

- Функціональні можливості СУРБД (окрім притаманних централізованим):
  - розширені служби установки з'єднань;
  - розширені засоби ведення каталогу;
  - засоби обробки розподілених запитів;
  - розширені функції управління паралельністю;
  - розширені функції відновлення.

Архітектура СУБД

- Рекомендована архітектура розподілених СУБД:
  - набір глобальних зовнішніх схем;
  - глобальна концептуальна схема;
  - схема фрагментації і схема розподілу;
  - набір схем для кожної локальної СУБД, що відповідають вимогам трирівневої архітектури ANSI-SPARC.
- Рекомендована архітектура федеральних мультибазових СУБД має враховувати рівень локальної автономності, який надається користувачеві.
- ***Слабко пов'язана федеральна мультибазова система*** – це федеральна мультибазова система, яка не використовує глобальні концептуальні схеми. В такій системі зовнішні схеми складаються з однієї або декількох локальних концептуальних схем.
- ***Тісно пов'язана федеральна мультибазова система*** – це федеральна мультибазова система, яка використовує глобальні концептуальні схеми, які є інтегрованим представленням або елементів локальних концептуальних схем, або локальних зовнішніх схем.
- Компонентна архітектура СУРБД:

- локальна СУБД;
- компонент передачі даних;
- глобальний системний каталог;
- розподілена СУБД (СУРБД).

Дванадцять правил Дейта для розподілених СУБД

- ***Правило 0.*** Основний принцип: розподілена система з точки зору кінцевого користувача повинна виглядати так само, як звичайна, нерозподілена система.
- ***Правило 1.*** Локальна автономність: сайти розподіленої системи повинні бути автономними.
- ***Правило 2.*** Відсутність опори на центральний сайт: в системі не повинно бути жодного сайту, без якого вона не зможе функціонувати.
- ***Правило 3.*** Безперервне функціонування: в системі ніколи не повинна виникати потреба у плановому припиненні її функціонування для виконання операцій додавання/видалення сайту з системи, динамічного створення/видалення фрагментів з одного або декількох сайтів.
- ***Правило 4.*** Незалежність від розташування: користувач повинен отримувати доступ до бази даних з будь-якого сайту до будь-яких даних.
- ***Правило 5.*** Незалежність від фрагментації: користувач повинен отримувати доступ до даних незалежно від способу їх фрагментації.
- ***Правило 6.*** Незалежність від реплікації: користувач не повинен потребувати відомостей про наявність реплікації даних.
- ***Правило 7.*** Обробка розподілених запитів: система повинна підтримувати обробку запитів, що посилаються на дані, розподілені на більш ніж одному сайті.
- ***Правило 8.*** Обробка розподілених транзакцій: система повинна підтримувати виконання транзакцій, як одиниці відновлення.

- ***Правило 9.*** Незалежність від типу обладнання: система повинна бути здатною функціонувати на обладнанні з різними обчислювальними платформами.
- ***Правило 10.*** Незалежність від операційної системи: система повинна бути здатною функціонувати під управлінням різних операційних систем.
- ***Правило 11.*** Незалежність від мережевої архітектури: система повинна бути здатною функціонувати в мережах з різною архітектурою та типами носія.
- ***Правило 12.*** Незалежність від типу СУБД: система повинна підтримувати гетерогенність.

Закріплення матеріалу

1. Що таке розподілена база даних?
2. Що таке розподілена СУБД?
3. Що таке розподілена обробка даних?
4. Пояснити відмінності між СУРБД та системами з розподіленою обробкою.
5. Що таке локальний та глобальний додатки?
6. Що таке реплікація даних?
7. Які основні типи архітектури паралельних СУБД вам відомі?
8. Які функції розподілених СУБД вам відомі?
9. Що таке мультибазова система?
10. В чому відмінність між гомогенною та гетерогенною розподіленими СУБД?
11. Охарактеризуйте рекомендовані архітектури для розподілених та мультибазових СУБД.
12. Які правила сформульовані Дейтом для розподілених СУБД?

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.2 НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ

Лекція 13

ОБ'ЄКТНО-ОРІЄНТОВАНІ СУБД

Мета лекції: розгляд основних концепцій об'єктно-орієнтованої обробки даних.

Зміст лекції

1. Основні концепції об'єктно-орієнтованого підходу.
2. Аспекти функціонування.
3. Маніфест, переваги та недоліки ООСУБД.

Основні концепції об'єктно-орієнтованого підходу

- ***Абстракція*** – це процес ідентифікації найбільш важливих аспектів сутності та ігнорування всіх інших її малозначних властивостей. В контексті створення програмного забезпечення це означає концентрацію уваги на те, що представляє собою об'єкт та що він може робити ще до вибору методу його реалізації.
- Основні аспекти абстракції:
  - інкапсуляція;
  - приховування інформації.
- ***Інкапсуляція*** означає, що об'єкт містить як структуру даних, так і набір операцій, за допомогою яких цією структурою можна маніпулювати.
- ***Приховування інформації*** означає, що всі зовнішні аспекти об'єкта відокремлюються від подробиць його внутрішнього устрою, прихованих від зовнішнього світу.
- Представлення про інкапсуляцію:
  - представлення об'єктно-орієнтованої мови програмування (ООМП);

  - адаптація представлення об’єктно-орієнтованої мови програмування для потреб баз даних.
- ***Об'єкт*** – це сутність, яка унікально ідентифікується та містить атрибути, що описують стан об'єктів реального світу і пов'язані з ними дії.
- ***Атрибут*** – це елемент для опису поточного стану об’єкту.
- ***Простий атрибут*** – це атрибут, який може мати примітивний тип та приймати літеральне значення.
- ***Складовий атрибут*** – це атрибут, який може містити колекції і/або посилання.
- ***Складовий об'єкт*** – це об'єкт, який містить один або декілька складових атрибутів; виглядає як єдиний об'єкт в реальному світі, але містить в собі інші об'єкти у вигляді набору складових зв'язків типу a-part-of.
- Властивості ідентифікатора об'єкта:
  - генерується системою;
  - унікально позначає об'єкт;
  - інваріантний;
  - не залежить від значень атрибутів об’єкта;
  - прихований від користувача.
- Переваги використання ідентифікаторів об'єктів:
  - ефективність;
  - швидкість;
  - неможливість зміни користувачем;
  - незалежність від змісту даних.
- ***Метод*** – це функція або процедура, яка визначає поведінку об'єкта та складається з імені і тіла. В об'єктно-орієнтованих мовах програмування тіло складається з блоку програмного коду, який виконує необхідні функції.

- ***Повідомлення*** – це запит, спрямований одним об'єктом (відправником) на адресу іншого об'єкта (одержувача) та який вимагає, щоб об'єкт-одержувач виконав один зі своїх методів.
- ***Клас*** – це шаблон для визначення набору подібних об'єктів.
- ***Екземпляр*** – це об'єкт класу. В об'єктно-орієнтованих мовах новий екземпляр зазвичай створюється за допомогою команди new.
- ***Конструктор*** – це метод створення екземплярів.
- ***Деструктор*** – це метод видалення екземплярів з вивільненням зайнятого ними простору пам'яті.
- ***Метаклас*** – це екземпляром класу більш високого рівня.
- ***Наслідування*** – це парадигма об’єктно-орієнтованого програмування, яка дозволяє визначати один клас на основі більш загального класу.
- ***Підклас*** – це менш загальний клас.
- ***Суперклас*** – це більш загальний клас.
- ***Узагальнення*** – це процес утворення суперкласу.
- ***Спеціалізація*** – це процес утворення підкласу.
- ***Зв'язок типу АКО*** (A Kind Of) – це зв'язок між підкласом і суперкласом.
- Види наслідування:
  - одиничне – підкласи наслідують не більше ніж від одного суперкласу;
  - множинне – підкласи наслідують властивості суперкласів;
  - повторне – суперкласи походять від загального суперкласу;
  - вибіркове – підкласи наслідують обмежену кількість властивостей їх суперкласу.
- ***Перевизначення*** – це визначення властивості суперкласу у підкласі.
- ***Перезавантаження*** – це окремий випадок поліморфізму – дозволяє повторно використовувати ім'я методу всередині визначення класу і визначень декількох класів.

- Типи поліморфізму:
  - робочий поліморфізм (перезавантаження);
  - поліморфізм включення (метод, визначений в суперкласі та успадкований в його підкласі);
  - параметричний поліморфізм (використання типів в якості параметрів в оголошеннях універсального типу або класу).
- ***Зв'язування*** (binding) – це процес вибору відповідного методу, заснований на типі об'єкта.
- ***Динамічне*** (dynamic) або ***пізнє*** (late) ***зв'язування*** – це відкладання процесу визначення типу об'єкта до настання часу виконання (а не під час компіляції).

Аспекти функціонування

- ***Об'єктно-орієнтована модель даних*** – це логічна модель даних, що враховує семантику об'єктів, яка застосовується в об'єктно-орієнтованому програмуванні.
- ***Об'єктно-орієнтована бази даних*** – це перманентний набір (колекція) об'єктів для сумісного використання, визначений властивостями об'єктно-орієнтованої моделі даних.
- ***Об'єктно-орієнтована СУБД*** – це система управління об'єктно-орієнтованими базами даних.
- Мінімальна модель для об'єктно-орієнтованої СУБД:
  1. Забезпечення функціональності бази даних.
  2. Підтримка ідентичності об'єкта.
  3. Забезпечення інкапсуляції.
  4. Підтримка об'єктів із складним станом.
- ***Об'єктно-орієнтована СУБД:***
  1. "об'єктно-орієнтований підхід" = "абстрактні типи даних" + "наслідування" + "ідентичність об'єктів".

2. "об'єктно-орієнтована СУБД" = "об'єктно-орієнтований підхід" + "можливості бази даних".

- ***Об'єктно-орієнтована СУБД:***

  1. Високорівнева мова запитів із засобами оптимізації, реалізованими в базовій системі.

  2. Підтримка перманентності і збереження атомарності транзакцій, що забезпечуються механізмами управління паралельністю і відновленням.

  3. Підтримка складних об'єктних сховищ, індексів і методів доступу, призначених для швидкого і ефективного отримання даних.

  4. "Об'єктно-орієнтована СУБД" = "об'єктно-орієнтована система" + «умови пунктів 1, 2 і 3".

- ***Перманентна мова програмування*** – це мова, яка дозволяє користувачам зберігати дані безпосередньо при послідовному виконанні програми з наступним використанням цих даних іншими програмами (за необхідності).
- ***Мова програмування баз даних*** – це мова, в якій інтегруються деякі ідеї, взяті як з моделі програмування баз даних, так і з концепцій традиційних мов програмування.
- Підходи для розробки ООСУБД:

  1. Розширення існуючої об’єктно-орієнтованої мови програмування можливостями роботи з базою даних.

  2. Надання розширюваних об’єктно-орієнтованих бібліотек СУБД.

  3. Вставка конструкцій об’єктно-орієнтованої мови бази даних в звичайну базову мову.

  4. Розширення існуючої мови бази даних об’єктно-орієнтованими функціями.

  5. Розробка нової мови бази даних або моделі даних.

- Схеми забезпечення перманентності мов програмування:
    1. Створення контрольних точок.
    2. Серіалізація.
    3. Явна підкачка сторінок.
- ***Транзакція*** – це логічна одиниця роботи, яка завжди повинна переводити базу даних з одного несуперечливого стану в інший.
- В ООСУБД логічною одиницею управління паралельністю та відновленням є об'єкт. Протоколи на основі блокування є найбільш поширеним механізмом управління паралельністю, що використовується в ООСУБД для запобігання виникнення конфліктів.
- Підходи для вирішення проблем з довготривалими транзакціями:
    - використання версій;
    - удосконалення моделей обробки транзакцій.
- ***Управління версіями*** – це процес супроводу еволюції об'єктів.
- Способи управління версіями:
    - тимчасові версії;
    - робочі версії;
    - остаточні версії.
- Типові зміни, які можуть вноситись в схему бази даних:

    1. Зміна визначення класу:
    - зміна атрибутів;
    - зміна методів.

    2. Зміна ієрархії наслідування:
    - визначення класу *A* як суперкласу для класу *B*;
    - видалення класу *A* зі списку суперкласів для класу *B*;
    - зміна порядку суперкласів для класу *B*.

    3. Зміни набору класів.
- Аспекти архітектурних рішень ООСУБД:

  - оптимальне застосування архітектури клієнт/сервер;
  - способи збереження методів.
- Основні типи моделі "клієнт/сервер":
  - сервер об'єктів;
  - сервер сторінок;
  - сервер бази даних.
- Підходи до управління методами:
  - збереження методів у зовнішніх файлах;
  - збереження методів в базі даних.
- Переваги підходу із збереженням методів в базі даних:
  - виключається надлишковий код;
  - спрощуються модифікації;
  - методи захищені більшою мірою;
  - методи можуть сумісно використовуватися у паралельному режимі;
  - підвищена цілісність.

Маніфест, переваги та недоліки ООСУБД

- Маніфест об'єктно-орієнтованих СУБД:

  Правило 1. Підтримка складених об'єктів.

  Правило 2. Підтримка ідентичності об'єктів.

  Правило 3. Підтримка інкапсуляції.

  Правило 4. Підтримка типів або класів.

  Правило 5. Підтримка наслідування типів або класів від їх предків.

  Правило 6. Підтримка динамічного зв'язування.

  Правило 7. Мова DML повинна володіти обчислювальною повнотою.

  Правило 8. Набір типів даних має бути розширюваним.

  Правило 9. Підтримка перманентності даних.

  Правило 10. Підтримка дуже великих баз даних.

  Правило 11. Підтримка паралельної роботи багатьох користувачів.

Правило 12. Забезпечення відновлення після збоїв апаратного та програмного забезпечення.

Правило 13. Надання простих способів створення запитів до даних.

- Переваги ООСУБД:
  - збільшення можливості моделювання;
  - можливість розширення;
  - усунення проблеми невідповідності;
  - розвинена мова запитів;
  - підтримка еволюції схеми;
  - підтримка довготривалих транзакцій;
  - застосування для складних спеціалізованих додатків баз даних;
  - підвищена продуктивність.
- Недоліки ООСУБД:
  - відсутність універсальної моделі даних;
  - вплив оптимізації запитів на інкапсуляцію;
  - недостатність досвіду експлуатації;
  - відсутність підтримки представлень;
  - вплив на продуктивність блокування на рівні об'єкта;
  - складність;
  - недостатність коштів забезпечення безпеки.

Закріплення матеріалу

1. Що таке абстракція?
2. Що таке інкапсуляція, поліморфізм та наслідування?
3. Що таке реплікація даних?
4. Що таке об’єктно-орієнтована СУБД?
5. Які підходи для розробки ООСУБД вам відомі?
6. Які основні типи моделі "клієнт/сервер" вам відомі?
7. Назвіть переваги та недоліки ООСУБД.

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.2 НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ

Лекція 14

ОБ'ЄКТНО-РЕЛЯЦІЙНІ СУБД

Мета лекції: розгляд підходів щодо розширення функціональних можливостей об'єктно-реляційних СУБД.

Зміст лекції

1. Вступ до об'єктно-реляційних СУБД.
2. Маніфести новітніх БД.
3. Порівняльна характеристика ОРСУБД та ООСУБД.

Вступ до об'єктно-реляційних СУБД

- ***Об'єктно-реляційна СУБД*** – це СУБД з розширеною реляційною моделлю даних такими об'єктно-орієнтованими компонентами, як система типів, що розширюється користувачем, інкапсуляція, наслідування, поліморфізм, динамічне зв'язування методів, використання складових об'єктів підтримка ідентичності об'єктів.
- Переваги ОРСУБД:
  - повторне і спільне використання компонентів;
  - підвищення продуктивність праці розробників і кінцевих користувачів;
  - великий об'єм накопичених знань і досвіду, пов'язаних з розробкою реляційних додатків.
- Повторне використання компонентів ґрунтується на можливості розширення сервера СУБД стандартними функціями, що виконуються централізовано і не входить у вигляді виконуваного коду до складу кожного додатку.

- Недоліки:
  - складність;
  - підвищені витрати.
- Основне завдання розробників ОРСУБД:
  - повне розкриття особливостей оптимізатора запитів – важливого та складного в налаштуванні компонента СУБД;
  - навчання сторонніх розробників технологіям оптимізації.
- В традиційних РСУБД використовуються індекси на основі структури B-дерева для прискорення доступу до даних. В ОРСУБД використовуються спеціалізовані індексні структури: універсальні B-дерева, R-дерева (регіональні дерева) тощо.

Маніфести БД

Маніфест баз даних, опублікований комітетом CADF

- Функції, які повинна підтримувати будь-яка сучасна СУБД:

  1. Велика система типів.

  2. Підтримка механізму наслідування.

  3. Підтримка функцій, процедур, методів та механізму інкапсуляції.

  4. Присвоєння засобам СУБД унікальних ідентифікаторів для записів тільки в тому випадку, коли не можна використовувати первинні ключі, визначені користувачем.

  5. Правила (тригери обмеження) не повинні бути пов'язані з якоюсь конкретною функцією або колекцією.

  6. Здійснення всіх програмних способів доступу до бази даних за допомогою непроцедурної мови високого рівня.

  7. Існування принаймні двох способів визначення колекцій: один – на основі перерахування членів колекції, інший – з використанням мови запитів для визначення членства в колекції.

  8. Наявність представлень, які оновлюються.

9. Індикатори продуктивності не повинні мати ніякого відношення до моделей даних та не повинні бути присутніми в них.

10. СУБД повинні бути доступні з багатьох мов високого рівня.

11. Наявність у мов програмування перманентних форм.

12. Мова SQL є та залишається "міжгалактичною мовою роботи з даними".

13. Запити та відповіді на них повинні складати найнижчий рівень взаємодії клієнта і сервера.

Маніфест Дарвена та Дейта на захист реляційної моделі даних. Вимоги до мови D

- ***Реляційна модель***
- Приписи:

1. Домени.
2. Типізовані скаляри.
3. Скалярні оператори.
4. Фактичне подання.
5. Значення істинності.
6. Тип-конструктор TUPLE.
7. Тип-конструктор RELATION.
8. Оператор еквівалентності.
9. Кортежі.
10. Відношення.
11. Скалярні змінні.
12. Кортежні змінні.
13. Змінні відношень (relvar).
14. Базові або похідні змінні відношень.
15. Змінні баз даних (dbvar).
16. Транзакції та змінні баз даних.
17. Оператори створення/видалення.

18. Реляційна алгебра.
19. Імена змінних відношень та значення явних відношень.
20. Функції відношень.
21. Присвоєння відношення кортежу.
22. Порівняння.
23. Обмеження цілісності.
24. Предикати відношень баз даних.
25. Каталог.
26. Макет мови.

- Заборони:

1. Відсутність впорядкування атрибутів.
2. Відсутність впорядкування кортежів.
3. Відсутність дублікатів кортежів.
4. Відсутність невизначених значень (NULL).
5. Відсутність NULL-логічних помилок.
6. Відсутність конструкцій на внутрішньому рівні.
7. Відсутність операторів на рівні кортежів.
8. Відсутність складових стовпців.
9. Відсутність підміни перевірки відповідності домену.
10. Відсутність мови SQL.

- ***Інші вимоги, що не мають відношення до реляційної моделі***
- Приписи

1. Контроль типів під час компіляції.
2. Одиничне наслідування (умовно).
3. Множинне наслідування (умовно).
4. Обчислювальна повнота.
5. Явне розмежування транзакцій.
6. Вкладені транзакції.
7. Агреговані значення та порожні множини.

- Заборони
  1. Змінні відношень (relvar) не є доменами.
  2. Відсутність ідентифікаторів об'єктів.
  3. Відсутність "відкритих змінних екземпляра".
  4. Відсутність "закритих змінних екземпляра" або дружніх їм.

Дуже суворі вимоги до мови D

- Реляційна модель:
  1. Потенційні ключі для похідних змінних відношення.
  2. Генеровані системою ключі.
  3. Цілісність посилань.
  4. Виведення потенційних ключів.
  5. Часткові запити.
  6. Транзитивне замикання (для відношень).
  7. Параметри кортежу відношення.
  8. Значення, що приймаються за замовчуванням.
  9. SQL-міграція.
- Інші вимоги
  1. Наслідування типів.
  2. Типи-конструктори колекцій.
  3. Перетворення відношення назад.
  4. Однорівневе сховище.
- ***Домен*** (первинний об'єкт в даній пропозиції) – це іменована множина інкапсульованих значень довільної складності.
- Змінні, визначені в мові D:
  - скалярна змінна типу V – змінна, для якої допустимими значеннями є скаляри із зазначеного домену V;
  - змінна кортежу типу H – змінна, для якої допустимими значеннями є кортежі із зазначеним заголовком кортежу H;

  - змінна відношення (relvar) типу H – змінна, для якої припустимими значеннями є відношення із зазначеним заголовком відношення H;
  - змінна бази даних (dbvar) – іменований набір змінних відношення.
- Обмеження, що накладається на транзакцію: взаємодія тільки з однією змінною бази даних, тоді як змінні відношення з цієї змінної бази даних можуть додаватися або віддалятися динамічно. Додатково повинні підтримуватись вкладені транзакції.
- Функції мови D:
  - представляє реляційну алгебру "без зайвих надмірностей";
  - містить оператори для створення / видалення іменованих функцій, значеннями яких є відношення, що визначаються за допомогою заданих реляційних виразів;
  - підтримує наступні оператори порівняння:
    - (= та ≠) для кортежів;
    - (=, ≠, "є підмножиною", ∈ для перевірки приналежності кортежу до даного відношення) для відношень.
  - побудована у відповідності до принципів правильного проєктування мови програмування.

Порівняльна характеристика ОРСУБД та ООСУБД

- Об'єктно-реляційні та об'єктно-орієнтовані СУБД порівнюються з трьох точок зору: моделювання даних, доступ до даних та спільне використання даних. При цьому припускається, що майбутні ОРСУБД будуть задовольняти вимогам нових стандартів.

Закріплення матеріалу

1. Що таке об’єктно-реляційна СУБД?
2. Які переваги та недоліки ОРСУБД вам відомі?
3. Наведіть порівняльну характеристику ОРСУБД та ООСУБД.

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.2 НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ

Лекція 15

WEB-ТЕХНОЛОГІЇ ТА СУБД

Мета лекції: розгляд основних понять Internet та Web; опанування підходів щодо інтеграції баз даних у Web-середовище.

Зміст лекції

1. Короткий опис основних понять Internet та Web.
2. Web-середовище, як платформа для додатків баз даних.
3. Підходи інтеграції баз даних у Web-середовище.

Короткий опис основних понять Internet та Web

- ***Протокол TCP*** – це протокол, який регламентує коректну доставку повідомлень з одного комп'ютера на інший.
- ***Протокол IP*** – це протокол, який регламентує відправку і отримання пакетів даних при передачі їх між різними комп'ютерами на основі чотирибайтової адреси призначення (IP-адреси).
- ***Intranet*** – це єдиний Web-сайт або група Web-сайтів, що належать одній організації і є доступними тільки членам цієї організації.
- ***Брандмауер*** – це захисний екран між глобальним Internet і локальною комп'ютерною мережею організації, який виконує функцію перевірки і фільтрації даних, що надходять з Internet, тобто в залежності від налаштувань може пропустити їх або заблокувати.
- ***Extranet*** – це різновид мережі Intranet, яка частково доступна санкціонованим зовнішнім користувачам.

- ***EDI (Electronic Data Exchange)*** – це специфікація електронного обміну даними, яка дозволяє організаціям зв'язати воєдино системи обліку товарів і обліку покупок / замовлень.
- ***Аплет*** – це несамостійний компонент програмного забезпечення, який працює в контексті повноцінного додатка, призначений для однієї вузької задачі та не має цінності у відриві від базової програми.
- ***World Wide Web*** – це гіпермедіа система, яка надає прості засоби для перегляду інформації в мережі Internet за допомогою механізму гіперпосилань.
- ***Протокол HTTP*** (Hyper Text Transfer Protocol) – це загальний, об'єктно-орієнтований протокол передачі інформації між серверами і клієнтами в мережі Internet.
- ***URL-адреса*** (Uniform Resource Locator) – це спеціальна адреса, яка унікальним чином визначає місце розташування документа (або ресурсу) в мережі Internet.
- ***URI-ідентифікатори*** (Uniform Resource Identifiers) – це загальний набір всіх імен / адрес, які належать до ресурсів Internet.
- ***URN-імена*** (Uniform Resource Names) – позначають певний ресурс Internet, роблячи це на постійній основі, яка не залежить від поточного розташування; мають дуже загальний характер і ґрунтуються на сервісах підстановки імен.
- Етапи HTTP-транзакції:
  - підключення – клієнт встановлює з’єднання з Web-сервером;
  - запит – клієнт надсилає Web-серверу повідомлення-запит;
  - відгук – Web-сервер надсилає клієнтові відповідь (HTML-документ);
  - закриття – з'єднання з Web-сервером закривається.

- ***Мова HTML*** (HyperText Markup Language) – це система розмітки або внесення спеціальних дескрипторів в документи, призначені для публікації у Web.
- ***Мова SGML*** (Standardized Generalized Markup Language) – це система визначення типів структурованих документів і мов розмітки для подання екземплярів цих типів документів.
- ***Статична Web-сторінка*** – це HTML-документ, що зберігається в окремому файлі, вміст якого не змінюється до тих пір, поки не зміниться сам файл.
- ***Динамічна Web-сторінка*** – це сторінка, вміст якої генерується всякий раз при доступі до неї.

Web-середовище, як платформа для додатків баз даних

- Вимоги до інтеграції СУБД в середовище Web:
  - можливість захищеного доступу до цінних корпоративних даних;
  - спосіб підключення, що не залежить від даних і розробника програмного забезпечення;
  - можливість взаємодії з базою даних, незалежно від конкретного типу використовуваного Web-браузера або Web-сервера;
  - наявність такого підключення, яке дозволяє отримати всі переваги всіх компонентів СУБД, що використовується в даній організації;
  - відкритість архітектури, що дозволяє взаємодіяти з різноманітними системами та технологіями;
  - економічно ефективне рішення, що допускає масштабованість, зростання, зміну стратегічних напрямків та сприяє скороченню витрат на розробку і супровід додатків;
  - підтримка транзакцій, які покривають кілька HTTP-запитів;
  - підтримка сеансів на основі ідентифікації користувачів засобами СУБД і додатків;

  - прийнятна продуктивність;
  - мінімальний рівень адміністрування;
  - набір високорівневих інструментів розробки.
- Переваги інтеграції СУБД в середовище Web:
  - переваги використання функцій СУБД;
  - простота реалізації;
  - незалежність від платформи;
  - графічний інтерфейс користувача;
  - стандартизація;
  - міжплатформена підтримка;
  - прозорий мережевий доступ;
  - масштабованість розгортання;
  - інноваційність.
- Недоліки інтеграції СУБД в середовище Web:
  - недостатня надійність;
  - слабка захищеність;
  - висока вартість;
  - труднощі з визначенням масштабу;
  - обмежена функціональність мови HTML;
  - відсутність запам'ятовування стану;
  - високі вимоги до пропускної здатності мережі;
  - недостатня продуктивність;
  - недосконалість інструментів розробки.

Підходи інтеграції баз даних у Web-середовище

- Деякі методи інтеграції баз даних у Web-середовище:
  - CGI-сценарії;
  - серверні вставки Server-Side Includes;
  - файли cookies мови HTTP;

- розширення Web-сервера;
- мови C# та Java і їх розширення;
- мови сценаріїв JavaScript і VBScript;
- платформа Active Platform фірми Microsoft і її трансформація в .Net.

- ***Інтерфейс CGI*** (Common Gateway Interface) – це специфікації на передачу інформації між Web-сервером і CGI-програмою.
- ***CGI-сценарій*** – це будь-який сценарій, який може приймати і передавати дані відповідно до специфікації CGI.
- Основні методи передачі інформації від браузера до CGI-сценарію:
  - передача параметрів в командному рядку;
  - передача значень змінних оточення;
  - передача даних через стандартний вхідний потік;
  - використання додаткової інформації про шлях.
- ***Серверна вставка*** (SSI-вставка) – це процес синтаксичного розбору документа перед відправкою його браузеру.
- ***Файл cookies*** – це маленький текстовий файли, який створюється CGI-сценарієм, пересилається Web-сервером браузеру клієнта та зберігається на боці Web-клієнта.
- ***API*** (Application Programming Interface) – це опис способів, за допомогою яких одна програма може взаємодіяти з іншою програмою. Зазвичай API входить до опису Internet-протоколів, програмних каркасів або стандарту викликів функцій операційної системи, реалізується окремою програмною бібліотекою або сервісом операційної системи та використовується при написанні різноманітних додатків.
- ***RFC*** (Request for Comments) – документ з серії пронумерованих інформаційних документів Інтернету, що містять технічні специфікації і стандарти, які широко застосовуються в Internet.

- ***Мова сценаріїв*** (Scripting language) – це мова програмування, розроблена для запису сценаріїв (послідовностей операцій), які користувач може виконувати на комп'ютері.
- ***JavaScript*** – мультипарадигмена мова програмування, яка підтримує об'єктно-орієнтований, імперативний і функціональний стилі та зазвичай використовується як вбудована мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності Web-сторінкам.
- ***Технологія Active Server Pages*** (ASP) – це модель програмування, яка дозволяє створювати на Web-сервері динамічні інтерактивні Web-сторінки.
- ***Технологія Active Data Objects*** (ADO) – це програмне розширення технології ASP, яке реалізовано у Web-сервері з метою організації підключень до баз даних.
- ***XML*** (eXtensible Markup Language) – розширювана мова розмітки, вимовляється – текстовий формат, призначений для зберігання структурованих даних (замість існуючих файлів баз даних), для обміну інформацією між програмами та для створення на його основі більш спеціалізованих мов розмітки.

Закріплення матеріалу

1. Що таке протокол TCP?
2. Що таке протокол IP?
3. Поясніть різницю між Intranet та Extranet.
4. Що таке протокол HTTP?
5. Що таке URL-адреса, URI-ідентифікатори та URN-імена?
6. Поясніть призначення мов SGML, HTML, JavaScript та XML.
7. Які методи інтеграції баз даних у Web-середовище вам відомі?

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.2 НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ

Лекція 16

СХОВИЩА ДАНИХ

Мета лекції: розгляд основних понять, пов’язаних з концепцією сховищ даних.

Зміст лекції

1. Поняття сховищ даних.
2. Архітектура сховищ даних.
3. Інформаційні потоки в сховищі.
4. Інструменти та технології сховищ.
5. Проєктування сховищ.

Поняття сховищ даних

- ***Сховище даних*** – це предметно-орієнтований, інтегрований, прив'язаний до часу незмінний набір даних, призначений для підтримки та прийняття рішень.
- Характеристики сховища даних:
  - *предметна орієнтованість* – властивість, яка відображає необхідність збереження даних, призначених для підтримки прийняття рішень, а не звичайних оперативно-прикладних даних;
  - *інтегрованість* – властивість, яка відображає необхідність створення інтегрованого джерела, яке забезпечує узгодженість інформації, що зберігається, задля надання користувачеві єдиного узагальненого представлення даних.
  - *прив'язка до часу* – властивість, яка відображає необхідність прив’язки даних до деякого моменту чи проміжку часу задля їх коректності та точності.

  - незмінюваність – властивість, яка відображає регулярне поповнення даних за рахунок інформації з оперативних систем обробки, а не їх оновлення в оперативному режимі.

- ***Технологія сховищ даних*** – це технологія управління даними і їх аналізу.
- Переваги технології сховищ даних:
  - потенційно висока віддача від інвестицій;
  - підвищення конкурентоспроможності;
  - підвищення ефективності праці осіб, відповідальних за прийняття рішень.
- Проблеми сховищ даних:
  - недооцінювання ресурсів, необхідних для завантаження даних;
  - приховані проблеми джерел даних;
  - відсутність необхідних даних у наявних архівах;
  - підвищення вимог кінцевих користувачів;
  - гомогенизація даних;
  - високі вимоги ресурсів;
  - володіння даними;
  - складний супровід;
  - довготривалий характер проєктів;
  - складнощі інтеграції.

Архітектура сховищ даних

- Основні компоненти сховища даних:
  - оперативні дані;
  - менеджер завантаження;
  - менеджер сховища;
  - менеджер запитів;
  - детальні дані;

  - частково та глибоко узагальнені дані;
  - архівні та резервні копії;
  - метадані;
  - засоби доступу до даних кінцевого користувача.
- ***Оперативні дані*** – дані, які розміщуються в сховищі.
- ***Менеджер завантаження*** – це зовнішній компонент, який виконує всі операції, пов'язані з отриманням і завантаженням даних в сховище.
- ***Менеджер сховища*** – це компонент, який виконує всі операції, пов'язані з управлінням інформацією, розміщеною в сховищі даних.
- Операції менеджера сховища:
  - аналіз несуперечності даних;
  - перетворення і переміщення вихідних даних з тимчасового сховища в основні таблиці сховища даних;
  - створення індексів і представлень для базових таблиць;
  - денормалізація даних (в разі необхідності);
  - узагальнення даних (в разі необхідності);
  - резервне копіювання та архівування даних.
- ***Менеджер запитів*** – це внутрішній компонент, який виконує всі операції, пов'язані з управлінням призначеними для користувача запитами.
- Основні групи інструментів користувачів для доступу до даних:
  - інструменти створення звітів і запитів;
  - інструменти розробки додатків;
  - інструменти інформаційної системи керівника;
  - інструменти оперативної аналітичної обробки (OLAP-інструменти);
  - інструменти розробки даних.
- ***Детальні дані*** – дані, що описані в схемі бази даних.

- ***Частково та глибоко узагальнені дані*** – всі дані, що попередньо оброблені менеджером сховища з метою їх часткового або глибокого узагальнення.
- ***Архівні та резервні копії*** – компонент сховища, який відповідає за підготовку детальної і узагальненої інформації для розміщення в резервні та архівні копії.
- ***Метадані*** – компонент сховища, в якому зберігаються всі метадані, що використовуються будь-якими процесами сховища.
- Основні групи інструментів доступу користувача до даних:
  - інструменти створення звітів та запитів;
  - інструменти розробки додатків;
  - інструменти інформаційної системи;
  - інструменти оперативної аналітичної обробки;
  - інструменти розробки даних.
- ***OLAP-інструменти*** – це інструменти оперативної аналітичної обробки даних, які створюються на основі концепції багатовимірної бази даних та дозволяють кваліфікованим користувачам аналізувати дані за допомогою складних багатовимірних представлень.
- ***Розробка даних*** – це процес відкриття нових усвідомлених кореляцій, розподілів і тенденцій шляхом переробки величезної кількості інформації, витягнутої із сховища або магазину даних, з використанням статистичних і математичних методів, а також методів штучного інтелекту.

Інформаційні потоки в сховищі

- ***Вхідний потік*** – це процеси, пов'язані з отриманням, очищенням та завантаженням інформації з джерел даних у сховище даних.
- ***Висхідний потік*** – це процеси, пов'язані з підвищенням цінності даних, що зберігаю у сховищі, за допомогою узагальнення, упаковки та розподілення вихідних даних.

- ***Низхідний потік*** – це процеси, пов'язані з архівуванням та резервним копіюванням інформації у сховищі даних.
- ***Вихідний потік*** – це процеси, пов'язані з наданням даних користувачам.
- ***Метапотік*** – це процеси, пов'язані з управлінням метаданими.

Інструменти та технології сховищ

- Інструменти і технології, які використовуються при створенні і супроводі сховищ даних:
  - інструменти отримання, очищення та перетворення даних;
  - СУБД для сховища даних;
  - метадані сховища даних;
  - інструменти управління та адміністрування.
- Інтегровані рішення для отримання, очищення та перетворення даних з наступним завантаженням у конкретну систему:
  - генератори коду;
  - інструменти реплікації інформації бази даних;
  - механізми динамічного перетворення.
- Вимоги до СУБД для сховища даних:
  - висока продуктивність завантаження даних;
  - можливість обробки даних під час завантаження;
  - наявність засобів управління якістю даних;
  - висока продуктивність запитів;
  - масштабованість за розміром;
  - масштабованість за кількістю користувачів;
  - можливість організації мережі сховищ даних;
  - наявність засобів адміністрування сховища;
  - підтримка інтегрованого багатовимірного аналізу;
  - розширений набір функціональних засобів запитів.

- Основне призначення метаданих – це збереження інформації про виникнення даних з метою надання можливості адміністраторам сховища знати історію будь-якого елементу даних, який розміщений у сховищі.
- Метадані, пов'язані з перетворенням та завантаженням даних, повинні описувати джерело даних та будь-які зміни, внесені в ці дані.
- Метадані, пов'язані з управлінням даними, повинні описувати спосіб збереження даних у сховищі.
- Операції, які повинні дозволяти виконувати інструменти управління і адміністрування сховищ даних:
  - моніторинг завантаження даних з декількох джерел;
  - перевірка якості і цілісності даних;
  - управління та оновлення метаданих;
  - моніторинг поточної продуктивності бази даних з метою забезпечення швидкої реакції на запит та ефективного використання ресурсів;
  - аудит процесів використання сховища даних з метою збору інформації про вартість виконаної кожним користувачем роботи;
  - реплікація, розбиття та розподіл даних;
  - підтримка ефективного управління сховищем даних;
  - очищення даних;
  - архівування і резервне копіювання даних;
  - засоби відновлення після збою;
  - управління засобами захисту.

Проєктування сховищ

- ***Схема "зірка"*** – це логічна структура, в центрі якої знаходиться таблиця фактів (з детальними даними), оточена таблицями розмірності (з даними-посиланнями).

- Чинники, необхідні для оптимального проєктування бази даних:
  - визначення необхідного часу відгуку для кожного додатку підтримки прийняття рішення;
  - пошук компромісу між необхідністю використання статистичних вибірок підмножин даних та необхідністю обробки детальних відомостей;
  - визначення стовпців, що видаляються;
  - скорочення розміру стовпців таблиці фактів;
  - визначення найкращого способу застосування зовнішніх ключів, які налаштовуються та не налаштовуються;
  - визначення оптимального підходу для введення в таблицю фактів розмірності "час";
  - секціонування таблиць фактів для поліпшення їх керованості.
- ***Схема "сніжинка"*** – це варіант схеми "зірка", в якому кожна розмірність може мати свої власні розмірності.
- ***Схема "зірка-сніжинка"*** – це гібридна структура, яка включає комбінацію денормалізованої схеми "зірка" і нормалізованої схеми "сніжинка".

Закріплення матеріалу

1. Що таке сховище даних та технологія сховищ даних?
2. Які характеристики сховища даних вам відомі?
3. Назвіть переваги та проблеми, пов’язані із сховищами даних.
4. Які інформаційні потоки виділяють в сховищі даних?
5. Які інструменти і технології, які використовуються при створенні і супроводі сховищ даних?
6. Які висуваються вимоги до СУБД для сховища даних?

РОЗДІЛ 2
Деякі аспекти експлуатації баз даних

Тема 2.2 НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ

Лекція 17

OLAP ТА РОЗРОБКА ДАНИХ

Мета лекції: ознайомлення і огляд основних характеристик інструментів користувачів для доступу до даних в сховищах даних.

Зміст лекції

1. Інтерактивна аналітична обробка даних (OLAP).
2. Технологія розробки даних.

Інтерактивна аналітична обробка даних (OLAP)

- ***Оперативна аналітична обробка*** (OLAP) – це динамічний синтез, аналіз та консолідація великих об'ємів багатовимірних даних.
- Основні аналітичні операції серверів багатовимірних баз даних на основі OLAP:
  - *Консолідація*. Операція включає такі узагальнюючі операції, як просте підсумовування значень ("згортка") або розрахунок використанням складних виразів, які включають інші пов'язані дані.
  - *Низхідний аналіз* ("drill-down"). Операція, зворотна консолідації, яка включає відображення докладних відомостей для розглянутих консолідованих даних.
  - *Розбиття з поворотом* (slicing and dicing) або зведена таблиця. Операція дозволяє отримати представлення даних з різних точок зору.
- Правила для OLAP-систем:

  1. Багатовимірне концептуальне представлення даних.
  2. Прозорість.

3. Доступність.

4. Незмінна продуктивність підготовки звітів.

5. Архітектура "клієнт / сервер".

6. Універсальність вимірювань.

7. Динамічне управління розрідженістю матриць.

8. Розрахована на багато користувачів підтримка.

9. Необмежені перехресні операції між розмірностями.

10. Підтримка інтуїтивно зрозумілого маніпулювання даними.

11. Гнучкість засобів формування звітів.

12. Необмежена кількість вимірювань та рівнів узагальнення.

- Основні категорії OLAP-інструментів:
  - багатовимірні OLAP-інструменти;
  - реляційні OLAP-інструменти;
  - кероване середовище запитів.

Технологія розробки даних

- ***Розробка даних*** **–** процес вилучення з великих баз даних достовірної, попередньо невідомої, комплексної та значимої інформації і використання її для прийняття відповідальних бізнес-рішень.
- Основні операції методів розробки даних:
  - прогнозуюче моделювання;
  - сегментування баз даних;
  - аналіз зв'язків;
  - виявлення відхилень.
- Методи прогнозуючого моделювання:
  - класифікація;
  - прогнозування значення.

- ***Класифікація*** використовується для встановлення спеціального зумовленого класу для кожного запису в базі даних на основі обмеженого набору можливих значень класу.
- Спеціалізації методу класифікації:
  - деревовидна індукція;
  - нейронна індукція.
- Метою ***сегментування бази*** даних є її розбиття на деяку, заздалегідь невідому кількість сегментів, або кластерів, що складаються з подібних записів, тобто записів, які мають деякі загальні властивості, що дозволяє вважати їх однорідними.
- Метод сегментування бази даних також пов'язаний з методами демографічної та нейронної кластеризації, які відрізняються допустимими вхідними даними, методами розрахунку відстані між записами і представленням результуючих сегментів.
- Метою ***аналізу зв'язків*** є встановлення зв'язків, або асоціацій, між окремими записами або наборами записів в базі даних.
- Спеціалізації методу аналізу зв’язків:
  - пошук асоціацій;
  - пошук послідовних закономірностей;
  - пошук аналогічних часових послідовностей.
- Метод ***виявлення відхилень*** передбачає аналіз викидів, які висловлюють ступінь відхилення від деяких попередньо відомих значень математичного очікування і нормального середнього.
- Важливі характеристики інструментів розробки даних:
  - наявність інструментів підготовки даних;
  - можливість вибору операцій розробки даних (алгоритмів);
  - масштабованість продукту і показники його продуктивності;
  - наявність інструментів візуалізації даних.

Закріплення матеріалу

1. Що таке OLAP-інструменти?
2. Що таке розробка даних?
3. Назвіть правила та поясніть для OLAP-систем.
4. Назвіть основні категорії OLAP-інструментів.
5. Які основні операції методів розробки даних?
6. Які основні категорії OLAP-інструментів вам відомі?
7. Які важливі характеристики інструментів розробки даних вам відомі?

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. *Дейт К.* Введение в системы баз данных : пер. с англ. / К. Дейт. – 8-е изд. – М., СПб.: Вильямс, 2005. – 1328 с.

2. *Мейер Д.* Теория реляционных баз данных / Д. Мейер. – М., Мир, 1987. – 608 с.

3. *Кузнецов С.Д.* Основы баз данных / С.Д. Кузнецов. – 2-е изд. – Москва: Бином, 2007. – 251 с.

4. *Кодд Э.Ф.* Реляционная модель данных для больших совместно используемых банков данных / Э.Ф. Кодд // СУБД. – 1995. – № 1.

5. *Чемберлин Д.* Анатомия объектно-реляционных баз данных / Д. Чемберлин // СУБД. – 1998. – №. 1-2.

6. *Бойко В.В.* Проектирование баз данных информационных систем / В.В. Бойко, В.М. Савинков. – М.: Финансы и статистика, 1989. – 351 с.

7. *Васкевич Д.* Стратегии клиент/сервер / Д. Васкевич. – Киев: Диалектика, 1996. – 384 с.

8. *Кузнецов С.Д.* Стандарты языка реляционных баз данных SQL: краткий обзор / С.Д. Кузнецов // СУБД. – 1996. – №2. – С.6-36.

9. *Date C. J.* Databases, Types, and the Relational Model. The Third Manifesto / C. J. Date, Hugh Darwen. – 3th edition. – Addison Wesley, 2006. – 572 p.

10. *Чен П.* Модель "сущность-связь" - шаг к единому представлению о данных / П. Чен //СУБД. – 1995. – №3. – С.137-158.

11. *Диго С.М.* Проектирование и использование баз данных / С.М. Диго. – М.: Финансы и статистика, 1995. – 208 с.

12. Справочник по Transact-SQL (Transact-SQL). [Електронний ресурс] – Режим доступу до ресурсу: https://msdn.microsoft.com/ru-ru/library/ms189826(v=sql.90).aspx – Дата доступу: 17.03.2021.

ДОДАТОК А
ПРЕЗЕНТАЦІЇ ДО ЛЕКЦІЙ


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 1. ПОНЯТТЯ ТА ТЕРМІНИ. СЕРЕДОВИЩЕ БД
Лекція 1
«Вступ до баз даних»
@ М.В.Добролюбова

Поняття бази даних

БАЗА ДАНИХ

© М.В.Добролюбова

2

Файлові системи

Файлові системи – набір програм, які виконують для користувачів деякі операції, при цьому кожна програма визначає свої власні дані та управляє ними.

Схема обробки даних в файловій системі

Обмеження, властиві файловим системам:

1. Розділення та ізоляція даних.
2. Дублювання даних.
3. Залежність від даних.
4. Несумісність файлів.
5. Фіксовані запити / швидке збільшення кількості додатків.

Фактори, що впливають на обмеження файлових систем:

1. Визначення даних міститься всередині додатків, не зберігається окремо та незалежно від них.
2. Крім додатків не передбачено ніяких інших інструментів доступу до даних та їх обробки.

© М.В.Добролюбова 3

Системи з базами даних

База даних – це сукупність взаємозалежних даних певної предметної галузі, збережених у пам'яті комп'ютера і організованих таким чином, що ці дані можуть бути використані для розв'язання різних завдань багатьма користувачами.

База даних – набір логічно пов'язаних даних спільного використання (і опис цих даних), призначений для задоволення інформаційних потреб.

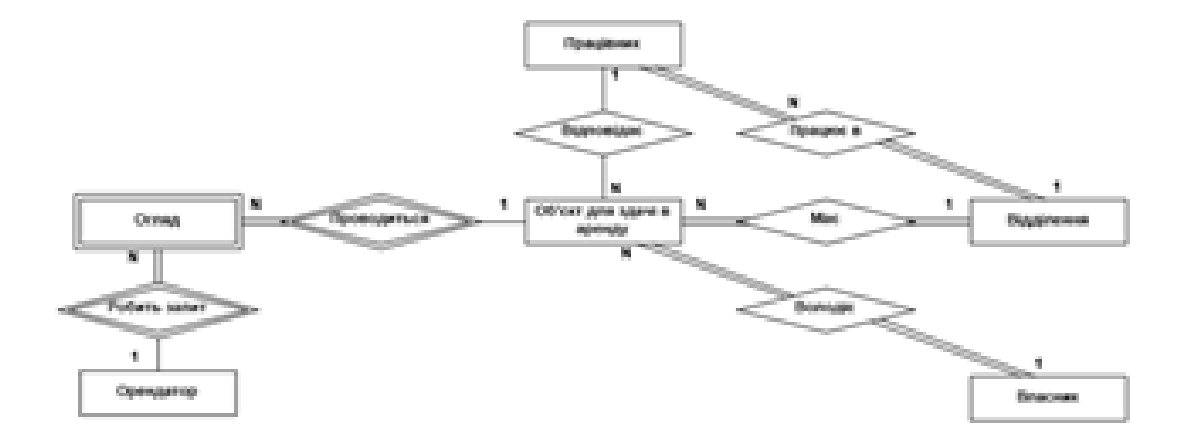
Характерні ознаки БД:

1. Єдине сховище даних.
2. Загальний корпоративний ресурс.
3. Збереження даних і їх опису.

Сутність (entity) – окремий тип об'єкту організації (людина, місце або річ, поняття або подія), який потрібно представити в базі даних.

Атрибут (attribute) – властивість, яка описує деяку характеристику об'єкта.

Зв'язок (relationship) – це те, що об'єднує кілька сутностей.



© М.В.Добролюбова 4

Системи з базами даних

СУБД – це програмне забезпечення, за допомогою якого користувачі можуть визначати, створювати підтримувати базу даних, а також здійснювати над нею контрольований доступ.

Можливості СУБД:

1) Дозволяє визначати базу даних за допомогою мови визначення даних.
2) Дозволяє вставляти, оновлювати, видаляти, витягувати інформацію з бази даних за допомогою мови управління даними.
3) Надає контрольований доступ до бази даних за допомогою системи забезпечення безпеки та системи підтримки цілісності даних.
4) Забезпечує додатковий рівень безпеки.
5) Надає механізм налаштування зовнішнього інтерфейсу бази даних.
6) Дозволяє зберігати зовнішній інтерфейс бази даних несуперечливим і незмінним навіть при внесенні змін до її структуру.

Схема обробки даних за допомогою СУБД

© М.В.Добролюбова 5

Системи з базами даних

Основні компоненти СУБД:

1. Апаратне забезпечення
2. Програмне забезпечення
3. Дані
4. Процедури
5. Користувачі

Групи користувачів:

- адміністратори даних баз даних
- розробники баз даних
- прикладні програмісти
- кінцеві користувачі

Схема обробки даних за допомогою СУБД

@ М.В.Добролюбова

6

Системи з базами даних

Історія розвитку СУБД

1. GUAM (Generalized Update Access Method) – фірма North American Aviation.
2. IMS (Information Management System) – фірми North American Aviation та IBM.
3. IDS (Integrated Data Store) – фірма General Electric.

Едгар Кодд

Пітер Чен

IBM DB2

ORACLE

DB DB.io Database of Databases SQL/DS

@ М.В.Добролюбова 7

Системи з базами даних

Переваги СУБД:

1. Контроль за надмірністю даних.
2. Несуперечність даних.
3. Більше корисної інформації при тому ж обсязі збережених даних.
4. Спільне використання даних.
5. Підтримка цілісності даних.
6. Підвищена безпека.
7. Застосування стандартів.
8. Підвищення ефективності зростанням масштабів системи.
9. Можливість знаходження компромісу при суперечливих вимогах.
10. Підвищення доступності даних і їх готовності до роботи.
11. Покращення показників продуктивності.
12. Спрощення супроводу системи за рахунок незалежності від даних.
13. Покращене керування паралельністю.
14. Розвинені служби резервного копіювання та відновлення.

Недоліки СУБД:

1. Складність.
2. Розмір.
3. Вартість.
4. Додаткові витрати на апаратне забезпечення.
5. Витрати на перетворення.
6. Продуктивність.
7. Серйозні наслідки при виході системи з ладу.

© М.В.Добролюбова

8

Практичний приклад

НАВЧАЛЬНИЙ ПРОЕКТ DreamHome

Вимоги до даних

Відділення компанії

- унікальний номер;
- адреса;
- номер телефону, факсу;
- власний штат співробітників.

Об'єкти нерухомості, що здаються в оренду

- унікальний персональний номер;
- повна адреса;
- тип об'єкта;
- кількість кімнат;
- місячна орендна плата (щорічно переглядається).

Персонал компанії

- унікальний персональний номер;
- ім'я та прізвище;
- адреса;
- номер телефону;
- стать;
- дата народження;
- номер соціальної страховки;
- посада;
- річна зарплата;
- дата вступу на посаду;
- відомості про найближчих родичів співробітників;
- окремо для секретаря – швидкості друку на ПК.

@ М.В.Добролюбова 9

Практичний приклад

НАВЧАЛЬНИЙ ПРОЕКТ DreamHome

Вимоги до даних

Власники нерухомості (фізична особа)

- власний номер, унікальний для всіх відділень компанії;
- ім'я та прізвище;
- адреса;
- номер телефону.

(юридична особа)

- назва компанії;
- тип діяльності;
- адреса представництва;
- телефон та ім'я контактної особи.

Клієнти / орендарі

- власний номер, унікальний для всіх відділень компанії;
- ім'я та прізвище;
- адреса;
- номер телефону;
- бажаний тип житла;
- дата співбесіди;
- ім'я співробітника, який проводив співбесіду з клієнтом;
- коментарі співробітника щодо потенційного орендаря.

Огляд нерухомості

- дата огляду об'єкта;
- коментарі потенційного орендаря щодо цього об'єкта.

© М.В.Добролюбова 10

Практичний приклад

НАВЧАЛЬНИЙ ПРОЕКТ DreamHome

Вимоги до даних

Реклама нерухомості

- номер рекламованого об'єкта;
- дата;
- вартість розміщення реклами;
- назва газети;
- адреса представництва;
- номер телефону та факсу;
- ім'я контактної особи.

Інспекція орендованого об'єкта

- відомості про об'єкт;
- дата проведення інспекції;
- будь-які зауваження.

Договір про оренду

- номер;
- відомості про орендаря;
- відомості про орендований об'єкт нерухомості;
- місячна орендна плата;
- спосіб оплати;
- сума завдатку;
- позначка про сплату завдатку;
- дата початку та закінчення оренди;
- термін дії договору про оренду;
- відомості про співробітника, який склав договір про оренду.

© М.В.Добролюбова 11

Практичний приклад

НАВЧАЛЬНИЙ ПРОЕКТ DreamHome

Транзакції, які виконуються в кожному відділенні компанії DreamHome

Транзакція 1. Створення та коригування записів з даними про співробітників та їх найближчих родичів для кожного відділення (менеджер).

Транзакція 2. Створення звіту з відомостями про співробітників кожного відділення (менеджер).

Транзакція 3. Створення списку співробітників, що працюють під керівництвом даного інспектора (менеджер і цей інспектор).

Транзакція 4. Створення списку інспекторів, які працюють в кожному з відділень компанії (менеджер і інспектор).

Транзакція 5. Створення та коригування записів з даними про об'єкти нерухомості (і її власників), що здаються оренду, у конкретному відділенні компанії (інспектор).

Транзакція 6. Створення звіту з даними про об'єкти нерухомості, що здаються в оренду, в даному відділенні компанії (всі співробітники).

Транзакція 7. Створення списку об'єктів, що здаються оренду, за які відповідає певний співробітник (інспектор).

Транзакція 8. Створення та коригування записів з описом потенційних орендарів по кожному з відділень компанії (інспектор).

Транзакція 9. Створення списку потенційних орендарів, зареєстрованих у кожному з відділень компанії (всі співробітники).

Транзакція 10. Пошук всіх об'єктів, що здаються в оренду, які відповідають вимогам потенційного орендаря (всі співробітники).

 12

Практичний приклад

НАВЧАЛЬНИЙ ПРОЕКТ DreamHome

Транзакції, які виконуються в кожному відділенні компанії DreamHome

Транзакція 11. Створення та коригування записів з відомостями про огляд об'єктів, що здаються в оренду, потенційними орендарями (всі співробітники).

Транзакція 12. Створення звіту із зауваженнями потенційних орендарів щодо конкретного об'єкта, який здається оренду (все співробітники).

Транзакція 13. Створення та коригування записів з відомостями про опубліковані в газетах рекламні оголошення про здачу в оренду об'єктів нерухомості (всі співробітники).

Транзакція 14. Створення списку всіх оголошень, зроблених щодо певного об'єкта, який здається в оренду (інспектор).

Транзакція 15. Створення списку всіх оголошень про здачу об'єктів в оренду, опублікованих в певній газеті (інспектор).

Транзакція 16. Створення та коригування записів з відомостями про укладені договори щодо оренди (менеджер та інспектор).

Транзакція 17. Роздруківка відомостей про умови укладеного договору про оренду для заданого об'єкта (менеджер та інспектор).

Транзакція 18. Створення та коригування записів з відомостями про виконані перевірки стану нерухомості, яка здається в оренду (всі співробітники).

Транзакція 19. Створення списку всіх проведених перевірок стану заданого об'єкта нерухомості (інспектор).

13

Практичний приклад

НАВЧАЛЬНИЙ ПРОЕКТ DreamHome

DreamHome
Staff Details

Staff Number: SL21

Personal Details

First Name	John	Last Name	White
Address	19 Taylor St, Cranford, London	Sex	Male
		Date of Birth	1-Oct-65
Tel No	0171-884-1112	NIN	

Next-of-Kin Details

Full Name	Mary White	Address	19 Taylor St, Cranford, London
Relationship	Wife		
Tel No	0171-884-1114		

General Work Details		Managerial Staff	
Position	Manager	Date Start	19-Sep-06
Allocated to Branch Office	London	Salary	30,000
Branch Number	1-Oct-65	Car Allowance	7,000
Joined Company	1-Oct-65	Bonus Payment	360
Secretarial Staff		Typing Speed	

Форма з даними про співробітника компанії DreamHome

Page 1 — Date 12-May-19

DreamHome
Property for Rent

Branch Number	John	Telephone Number	0141-339-2178
Branch Office Address	163 Main St, Partick, Glasgow G119QX	Fax Number	0141-339-4439

Property Number	Street	Area	City	Postcode	Type	No of Rooms	Monthly Rent
PG4	6 Lawrence St	Partick	Glasgow	G119QX	Flat	3	350
PG36	2 Manor Rd		Glasgow	G329QX	Flat	3	375
PG21	18 Dale Rd	Hyndland	Glasgow	G12	House	5	600
PG16	5 Novar Dr	Hyndland	Glasgow	G129AX	Flat	4	450

Звіт з перерахуванням об'єктів нерухомості, що здаються в оренду відділенням компанії DreamHome, яке розташоване в Глазго

@ М.В.Добролюбова

14


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 1. ПОНЯТТЯ ТА ТЕРМІНИ. СЕРЕДОВИЩЕ БД
Лекція 2
«Середовище бази даних»
@ М.В.Добролюбова

Трирівнева архітектура

© М.В.Добролюбова

2

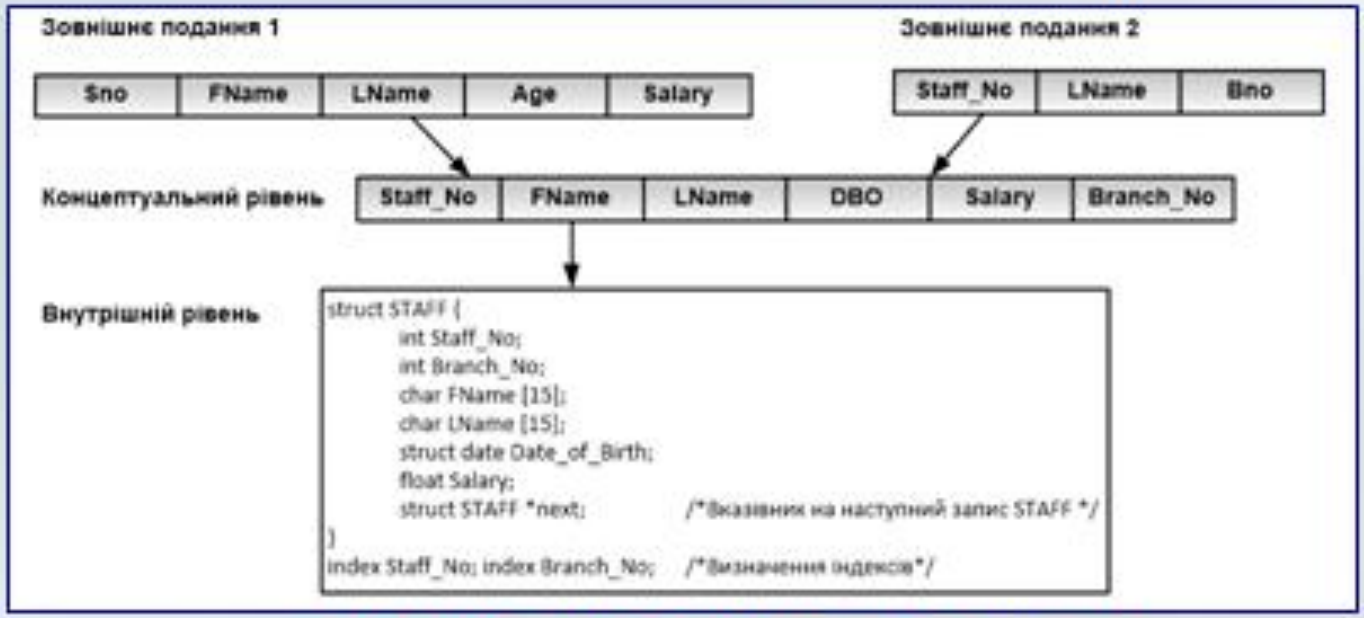
Трирівнева архітектура

Схеми відображення та екземпляри

Схема бази даних (вміст) – це загальний опис БД.
Стан бази даних (деталізація) – сукупність інформації, що зберігається в базі даних у будь-який визначений момент часу.

Типи схем:

Зовнішня схема (підсхема) Концептуальна схема Внутрішня схема



Відмінності між трьома рівнями подання даних

@ М.В.Добролюбова 3


Мови баз даних
DDL (Data Definition Language)
Vs
DML (Data Manipulation Language)
T-SQL
Vs
PL-SQL
Мова визначення даних (Data Definition Language – DDL) використовується для визначення схеми бази даних.
Мова управління даними (Data Manipulation Language – DML) використовується для читання та оновлення даних, що зберігаються базі.
Метадані – дані, які описують об'єкти бази даних та дозволяють спростити спосіб доступу до них і управління ними.
Процедурна мова DML – мова, яка дозволяє повідомити системі те, які дані необхідні, і точно вказати, як їх можна витягти.
Непроцедурна (декларативна) мова DML – мова, яка дозволяє вказати лише те, які дані необхідні, але не те, як їх слід витягувати.
Мови останніх поколінь: мови представлення інформації, спеціалізовані мови, генератори додатків, мови дуже високого рівня.
@ М.В.Добролюбова
4

Моделі даних та концептуальне моделювання

Модель даних – інтегрований набір понять для опису даних, зв'язків між ними та обмежень, що накладаються на дані.

Складові моделі даних:

структурна частина;

керуюча частина;

набір обмежень підтримки цілісності даних (необов'язково.

Види моделей даних:

зовнішня модель даних (предметна область);

концептуальна модель даних;

внутрішня модель даних.

Категорії моделей даних: об'єктні, на основі записів, фізичні.

Типи об'єктних моделей даних:

модель типу "сутність-зв'язок";

семантична модель;

функціональна модель;

об'єктно-орієнтована модель.

Типи логічних моделей даних на основі записів:

реляційна модель даних;

мережева модель даних;

ієрархічна модель даних.

Типи фізичних моделей даних:

узагальнююча;

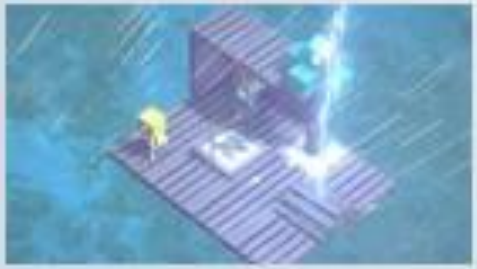
модель пам'яті кадрів.

© М.В.Добролюбова

5

Функції СУБД

- Зберігання, витягування та оновлення даних
- Каталог, доступний кінцевим користувачам
- Підтримка транзакцій
- Сервіси управління паралельністю
- Сервіси відновлення
- Сервіси контролю доступу до даних
- Підтримка обміну даними
- Служби підтримки цілісності даних
- Служби підтримки незалежності від даних
- Допоміжні служби

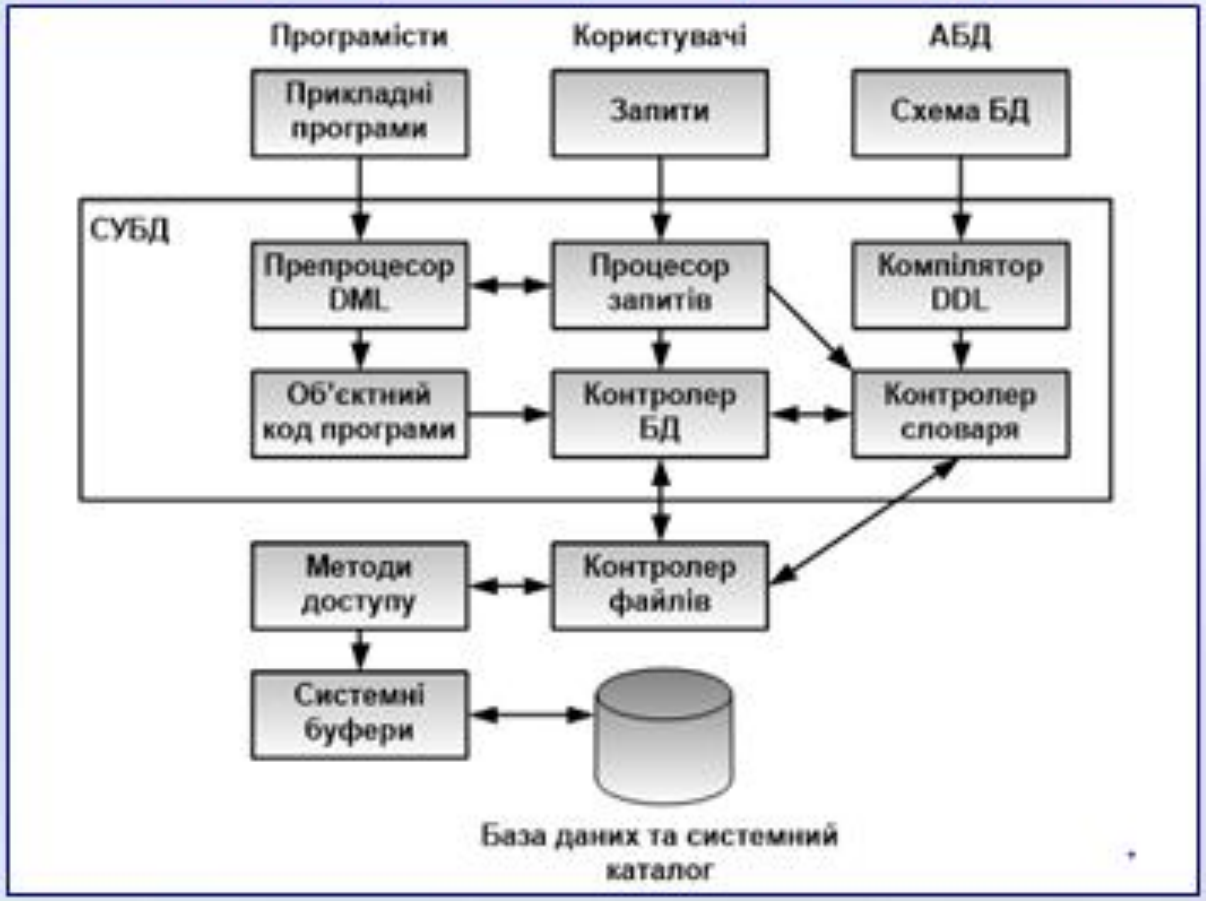


@ М.В.Добролюбова

6

Компоненти СУБД

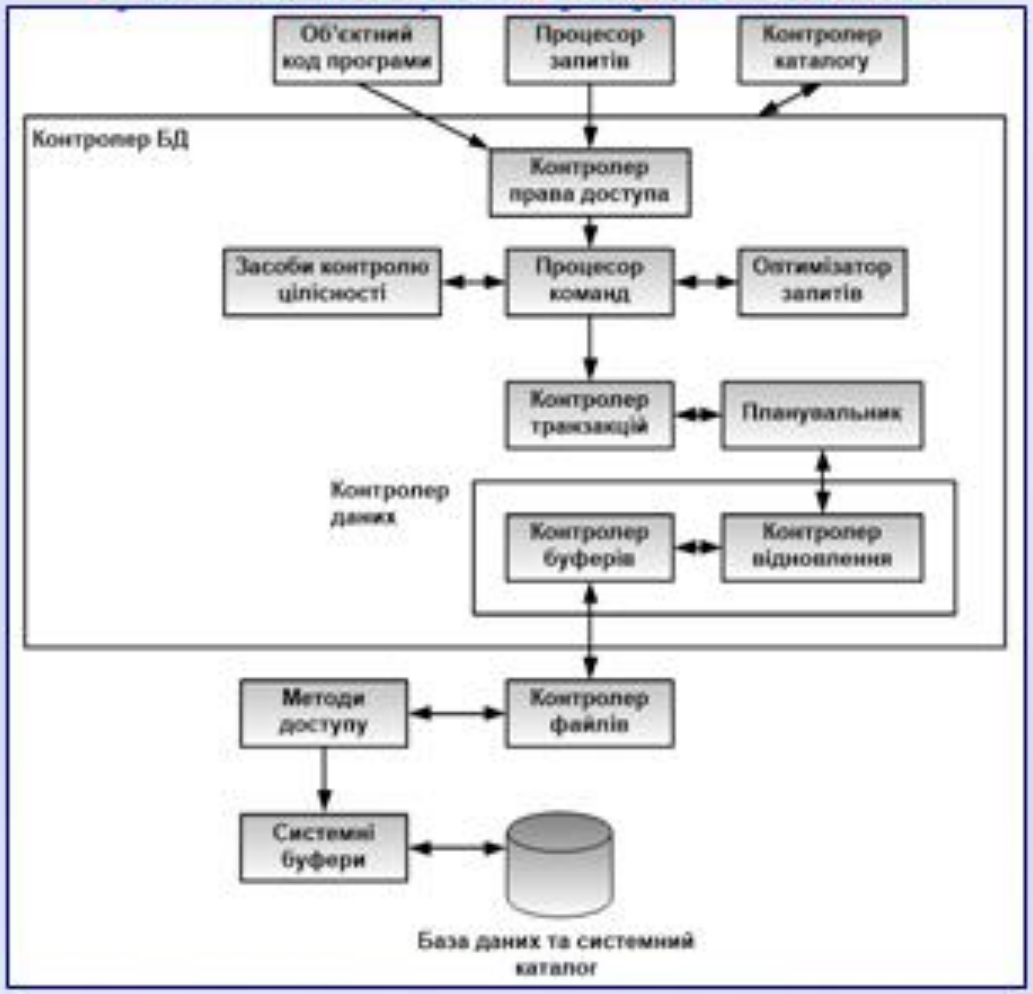
Основні компоненти типової системи управління базами даних



© М.В.Добролюбова 7

Компоненти СУБД

Компоненти контролера бази даних



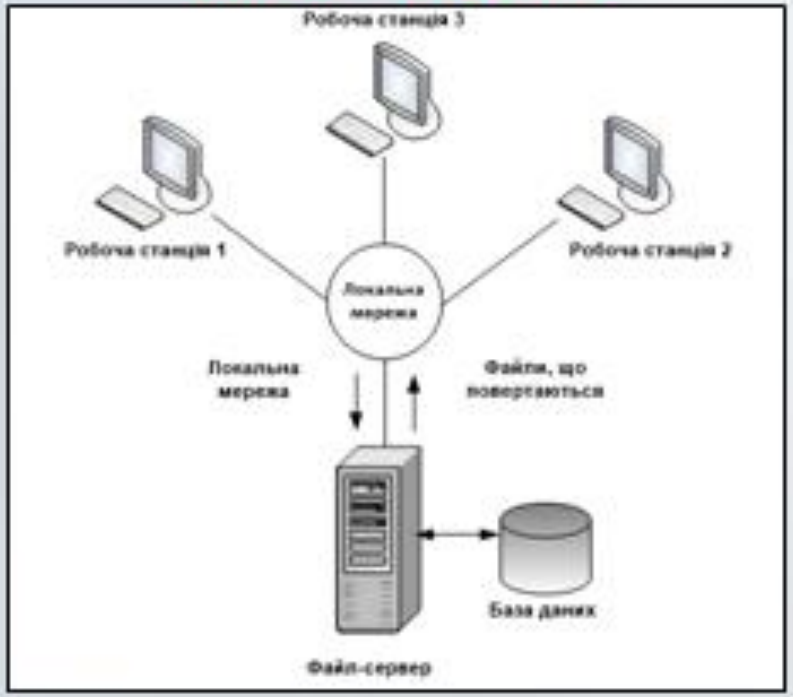
@ М.В.Добролюбова

8

Архітектура багатокористувальницьких СУБД

Топологія архітектури телеобробки

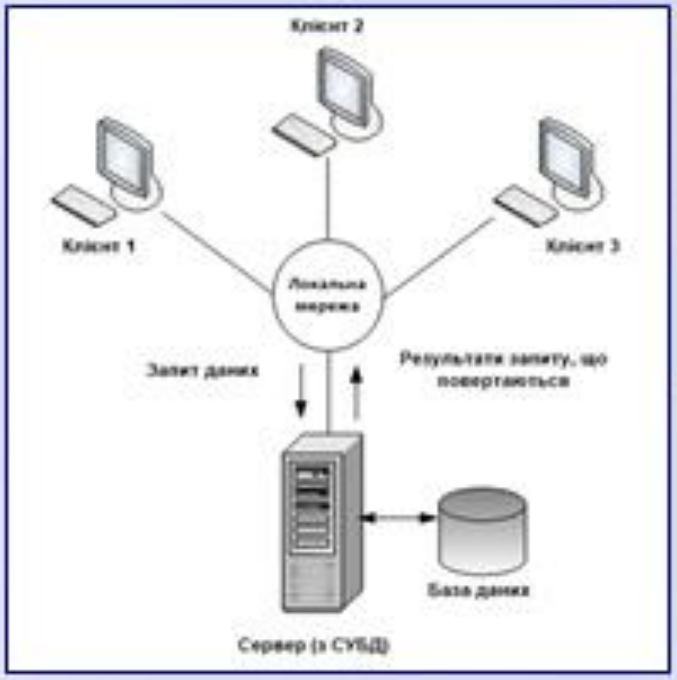
Архітектура з використанням файлового сервера



9

Архітектура багатокористувальницьких СУБД

Загальна схема побудови систем з архітектурою "клієнт / сервер"



Функції, які виконуються учасниками взаємодії у середовищі "клієнт / сервер"

Клієнт	Сервер
Управляє інтерфейсом користувача	Приймає та обробляє запити до бази даних з боку клієнтів
Приймає та перевіряє синтаксис введеного користувачем запиту	Перевіряє повноваження користувачів
Виконує додаток	Гарантує дотримання обмежень цілісності
Генерує запит до бази даних та передає його серверу	Виконує запити / оновлення та повертає результати клієнтові
Відображає отримані дані користувачеві	Підтримує системний каталог Забезпечує паралельний доступ до бази даних Забезпечує управління відновленням

Переваги:

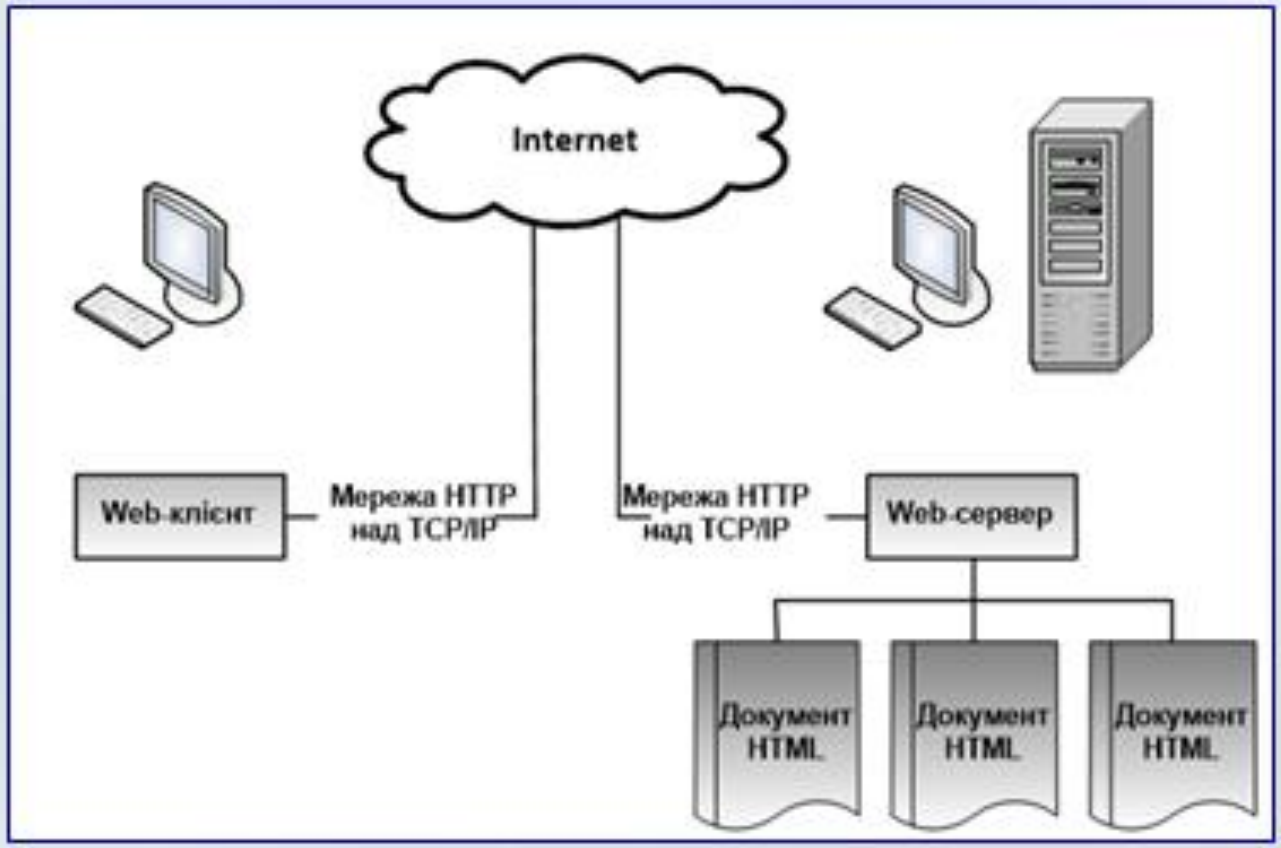
1) Забезпечується більш широкий доступ до існуючих баз даних.
2) Підвищується загальна продуктивність системи.
3) Вартість апаратного забезпечення знижується.
4) Скорочуються комунікаційні витрати.
5) Підвищується рівень несуперечливості даних.
6) Дана архітектура природно відображається на архітектуру відкритих систем.

@ М.В.Добролюбова

10

Web-технології

Основні компоненти середовища World Wide Web



© М.В.Добролюбова 11


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 2. РЕЛЯЦІЙНА МОДЕЛЬ. ПЛАНУВАННЯ, ПРОЄКТУВАННЯ, АДМІНІСТРУВАННЯ БД
Лекція 3
«Реляційні бази даних»
© М.В.Добролюбова

Термінологія

Едгар Кодд

Цілі створення реляційної моделі:

1) Забезпечення більш високого ступеня незалежності від даних.
2) Створення міцного фундаменту для вирішення семантичних питань, проблем несуперечності та надмірності даних, нормалізованих відношень.
3) Розширення мов управління даними за рахунок включення операцій над множинами.

Проєкт System R – дослідна лабораторія корпорації IBM в місті Сан-Хосе, штат Каліфорнія.

Проєкт INGRES (INteractive GRaphics REtrieval System) – Каліфорнійський університет (місто Берклі).

Проєкт PRTV (Peterlee Relational Test Vehicle) – науковий центр корпорації IBM (місто Петерлі, Великобританія).

@ М.В.Добролюбова 2

Термінологія

Відношення – це плоска таблиця, що складається із стовпців та рядків.

Атрибут – це поіменований стовпець відношення.

Домен – це набір допустимих значень для одного або декількох атрибутів.

Кортеж – це рядок відношення.

Ступінь відношення визначається кількістю атрибутів, яку воно містить.

Кардинальність – це кількість кортежів, яку містить відношення.

Реляційна база даних – це набір нормалізованих відношень.

Офіційні терміни	Альтернативний варіант 1	Альтернативний варіант 2
Відношення	Таблиця	Файл
Кортеж	Рядок	Запис
Атрибут	Стовпець	Поле

Термінологія

Базове відношення – це поіменоване відношення, яке відповідає сутності в концептуальній схемі, кортежі якого фізично зберігаються в базі даних.

Представлення – це динамічний результат однієї або декількох реляційних операцій над базовими відношеннями з метою створення деякого іншого відношення.

Причини використання механізму представлень:

- надає потужний гнучкий механізм захисту;
- дозволяє організувати доступ користувачів до даних зручним чином;
- дозволяє спрощувати складні операції з базовими відношеннями.

Умови для перевірки допустимості поновлення даних через деякі представлення:

- оновлення допускаються через представлення, яке визначено на основі простого запиту до єдиного базового відношення та містить первинний або потенційний ключ цього базового відношення;
- оновлення не допускаються в будь-яких представленнях, визначених на базі декількох базових відношень;
- оновлення не допускаються в будь-яких представленнях, що включають виконання операцій узагальнення або групування.

Класи представлень:

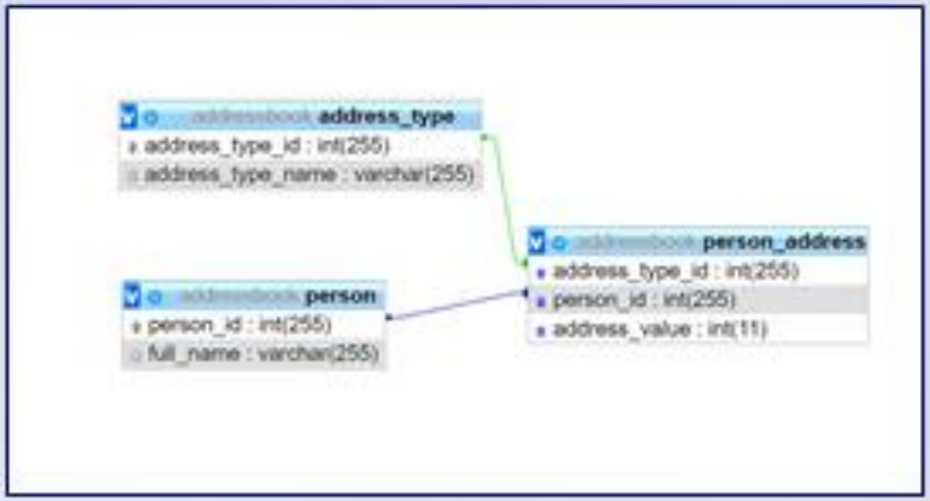
теоретично непоновлювані | теоретично поновлювані | частково поновлювані

@ М.В.Добролюбова 4

Властивості відношень

Характеристики відношень:

- відношення має ім'я, яке відрізняється від імен всіх інших відношень
- кожна комірка відношення містить тільки атомарне (неподільне) значення
- кожний атрибут має унікальне ім'я
- значення атрибута беруться з одного і того самого домену
- порядок проходження атрибутів не має ніякого значення
- кожен кортеж є унікальним, тобто дублікатів кортежів бути не може
- теоретично порядок слідування кортежів у відношенні не має ніякого значення



Реляційні ключі та реляційна цілісність

Реляційні ключі

Суперключ (superkey) – це атрибут або множина атрибутів, яка єдиним чином ідентифікує кортеж даного відношення.

Потенційний ключ – це суперключ, який не містить підмножини, що також є суперключем даного відношення.

Властивості потенційного ключа:

1) унікальність;
2) незвідність.

Первинний ключ – це потенційний ключ, який обраний для унікальної ідентифікації кортежів всередині відношення.

Зовнішній ключ – це атрибут або множина атрибутів всередині відношення, яка відповідає потенційному ключу деякого відношення.

Реляційна цілісність

Визначник NULL – вказує, що значення атрибута в даний момент невідоме чи неприйнятне для цього кортежу.

Цілісність сутностей свідчить про те, що у базовому відношенні жоден атрибут первинного ключа не може містити відсутніх значень, які позначаються визначником NULL.

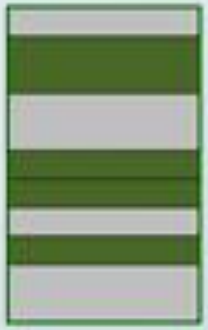
Цілісність посилання свідчить про те, що якщо у відношенні існує зовнішній ключ, то значення зовнішнього ключа повинно або відповідати значенням потенційного ключа деякого кортежу в його базовому відношенні, або задаватися визначником NULL.

Корпоративні обмеження цілісності – це додаткові правила підтримки цілісності даних, що визначаються користувачами або адміністраторами бази даних.

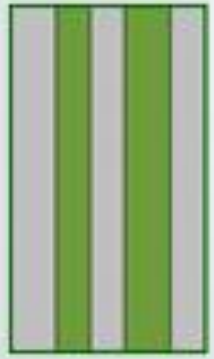
@ М.В.Добролюбова 6

Реляційна алгебра

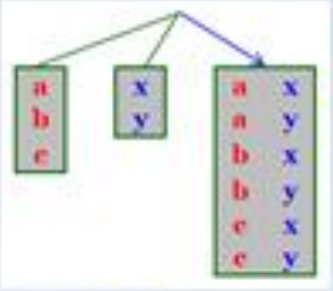
Основні операції реляційної алгебри



Операція вибірки (selection) проводиться над кортежами одного відношення. Результат вибірки – нове відношення, яке складається з кортежів вихідного відношення, що задовольняють заданій умові.



Операція проєкції (projection) проводиться над кортежами одного відношення. Результат проєкції – нове відношення, яке містить лише задані атрибути вихідного відношення.



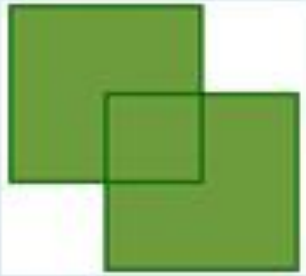
Декартовий добуток (cartesian product) – операція над двома відношеннями, в результаті якої отримується нове відношення, що складається з усіх можливих кортежів, які є попарними поєднанням кортежів вихідних відношень.

© М.В.Добролюбова

7

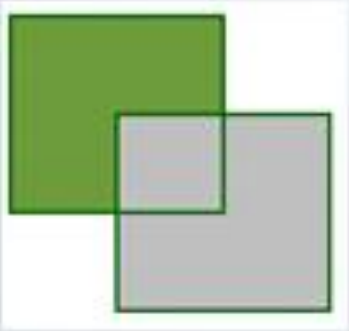
Реляційна алгебра

Основні операції реляційної алгебри



Об'єднання (union) – операція над двома відношеннями, в результаті якої отримується нове відношення, що складається з усіх кортежів вихідних відношень. Спільні для вихідних відношень кортежі в новому відношенні зустрічаються лише по одному разу.

Різниця (set difference) – операція над двома відношеннями, в результаті якої отримується нове відношення, що складається з кортежів, які належать першому відношенню і не належать другому.

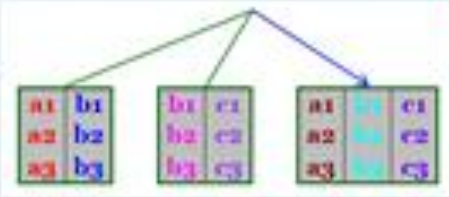


© М.В.Добролюбова

8

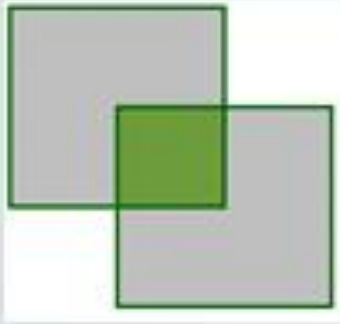
Реляційна алгебра

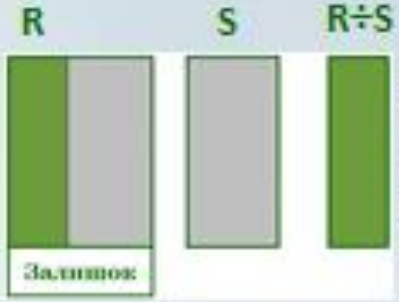
Додаткові операції реляційної алгебри



Поєднання (join) – операція над двома відношеннями, які мають спільні атрибути, в результаті якої отримується нове відношення, що складається з усіх атрибутів вихідних відношень і об'єднує лише ті кортежі вихідних відношень, в яких значення спільних атрибутів співпадають.

Перетин (intersection) – операція над двома відношеннями, в результаті якої отримується нове відношення, що складається з кортежів, які належать обом вихідним відношенням.





Результатом ***ділення (division)*** відношення A(A1, A2, ..., An) на відношення B(B1, B2, ..., Bn) по відношенню C(A1, A2, ..., An, B1, B2, ..., Bn) називається нове відношення, що містить всі такі кортежі x з відношення A, для яких виконується умова: для будь-якого y, що належить множині кортежів B, відношення C містить кортеж x:y.

© М.В.Добролюбова

9

Вимоги до реляційної СУБД

ФУНДАМЕНТАЛЬНІ ПРАВИЛА (ПРАВИЛА 0 ТА 12)

Правило 0 – фундаментальне правило
Правило 12 – правило заборони обхідних шляхів

СТРУКТУРНІ ПРАВИЛА (ПРАВИЛА 1 ТА 6)

Правило 1 – представлення інформації
Правило 6 – поновлення представлення

ПРАВИЛА ЦІЛІСНОСТІ (ПРАВИЛА 3 ТА 10)

Правило 3 – систематична обробка невизначених значень (NULL)
Правило 10 – незалежність обмежень цілісності

ПРАВИЛА МАНІПУЛЮВАННЯ ДАНИМИ (ПРАВИЛА 2, 4, 5 ТА 7)

Правило 2 – гарантований доступ
Правило 4 – динамічний інтерактивний каталог, побудований за правилами реляційної моделі
Правило 5 – вичерпна підмова даних
Правило 7 – високорівневі операції вставки, оновлення та видалення

ПРАВИЛА НЕЗАЛЕЖНОСТІ ВІД ДАНИХ (ПРАВИЛА 8, 9 ТА 11)

Правило 8 – фізична незалежність від даних
Правило 9 – логічна незалежність від даних
Правило 11 – незалежність від розподілу даних

@ М.В.Добролюбова 10

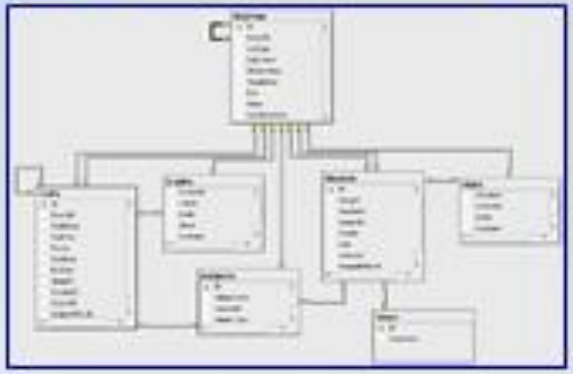

ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 3. МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО, ЛОГІЧНОГО ТА ФІЗИЧНОГО ПРОЄКТУВАННЯ БД
Лекція 4
«Методологія концептуального проєктування реляційних баз даних»
@ М.В.Добролюбова

Вступ в методологію проєктування БД

Концептуальне проєктування – створення концептуального представлення бази даних, що включає визначення типів найважливіших сутностей та існуючих між ними зв'язків.

Логічне проєктування – перетворення концептуального представлення в логічну структуру бази даних, включаючи проєктування відношень.

Фізичне проєктування – прийняття рішення про те, як логічна модель буде фізично реалізована в базі даних, що створюється за допомогою обраної СУБД.



@ М.В.Добролюбова

2

Вступ в методологію проєктування БД

Методологія проєктування – структурований підхід, який передбачає використання спеціалізованих процедур, технічних прийомів, інструментів, документації та націлений на підтримку спрощення процесу проєктування.

Фактори успішного завершення проєктування бази даних:

1) Підтримувати постійний активний зв'язок з потенційними користувачами додатку.
2) Дотримуватись при проведенні процедур моделювання даних рекомендацій, наведених під час обговорення методології.
3) Розробляти систему, виходячи з існуючих характеристик даних.
4) Створювати модель даних з урахуванням вимог підтримки їх структурної цілісності і узгодженості.
5) Доповнювати процедури, що пропонуються даною методологією, технологічними прийомами концептуалізації, нормалізації та перевірки цілісності транзакцій.
6) Використовувати діаграми для представлення моделі даних якомога ширше.
7) Використовувати засоби мови DBDL (Database Design Language) для опису додаткових семантичних вимог до даних.
8) Розробити словник опису даних.
9) Повертатися до вже виконаних раніше етапів, якщо це потрібно для досягнення оптимальних результатів.

© М.В.Добролюбова 3

Структура процедури проєктування БД

Фаза 1 Концептуальне проєктування бази даних

Етап 1. Створення локальної концептуальної моделі даних, виходячи з уявлень предметної області кожного з типів користувачів

Завдання 1.1. Визначення типів сутностей.
Завдання 1.2. Визначення типів зв'язків.
Завдання 1.3. Визначення атрибутів і зв'язування їх з типами сутностей та зв'язків.
Завдання 1.4. Визначення доменів атрибутів.
Завдання 1.5. Визначення атрибутів, які є потенційними і первинними ключами.
Завдання 1.6. Спеціалізація або генералізація типів сутностей (необов'язковий етап).
Завдання 1.7. Створення діаграми "сутність-зв'язок".
Завдання 1.8. Обговорення локальних концептуальних моделей даних з кінцевими користувачами.

© М.В.Добролюбова 4

Структура процедури проєктування БД

Фаза 2 Логічне проєктування бази даних (для реляційної моделі)

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Завдання 2.1. Перетворення локальної концептуальної моделі даних в локальну логічну модель.
Завдання 2.2. Визначення набору відношень виходячи зі структури локальної логічної моделі даних.
Завдання 2.3. Перевірка моделі за допомогою правил нормалізації.
Завдання 2.4. Перевірка моделі щодо транзакцій користувачів.
Завдання 2.5. Створення діаграм «сутність-зв'язок».
Завдання 2.6. Визначення вимог підтримки цілісності даних.
Завдання 2.7. Обговорення розроблених локальних логічних моделей даних з кінцевими користувачами.

Етап 3. Створення та перевірка глобальної логічної моделі даних

Завдання 3.1. Злиття локальних логічних моделей даних в єдину глобальну модель даних.
Завдання 3.2. Перевірка глобальної логічної моделі даних.
Завдання 3.3. Перевірка можливостей розширення моделі у майбутньому.
Завдання 3.4. Створення остаточного варіанту діаграми «сутність-зв'язок».
Завдання 3.5. Обговорення глобальної логічної моделі даних з користувачами.

© М.Е.Добролюбова 5

Структура процедури проєктування БД

Фаза 3 Фізичне проєктування бази даних (з використанням реляційної СУБД)

Етап 4. Перенесення глобальної логічної моделі даних в середовище цільової СУБД

Завдання 4.1. Проєктування основних таблиць в середовищі цільової СУБД.
Завдання 4.2. Реалізація бізнес-правил підприємства в середовищі цільової СУБД.

Етап 5. Проєктування фізичного представлення бази даних

Завдання 5.1. Аналіз транзакцій.
Завдання 5.2. Вибір файлової структури.
Завдання 5.3. Визначення вторинних індексів.
Завдання 5.4. Аналіз необхідності введення контрольованої надлишковості даних.
Завдання 5.5. Визначення вимог дискової пам'яті.

Етап 6. Розробка механізмів захисту

Завдання 6.1. Розробка користувальницьких представлень (видів).
Завдання 6.2. Визначення прав доступу.

Етап 7. Організація моніторингу та налаштування функціонування системи

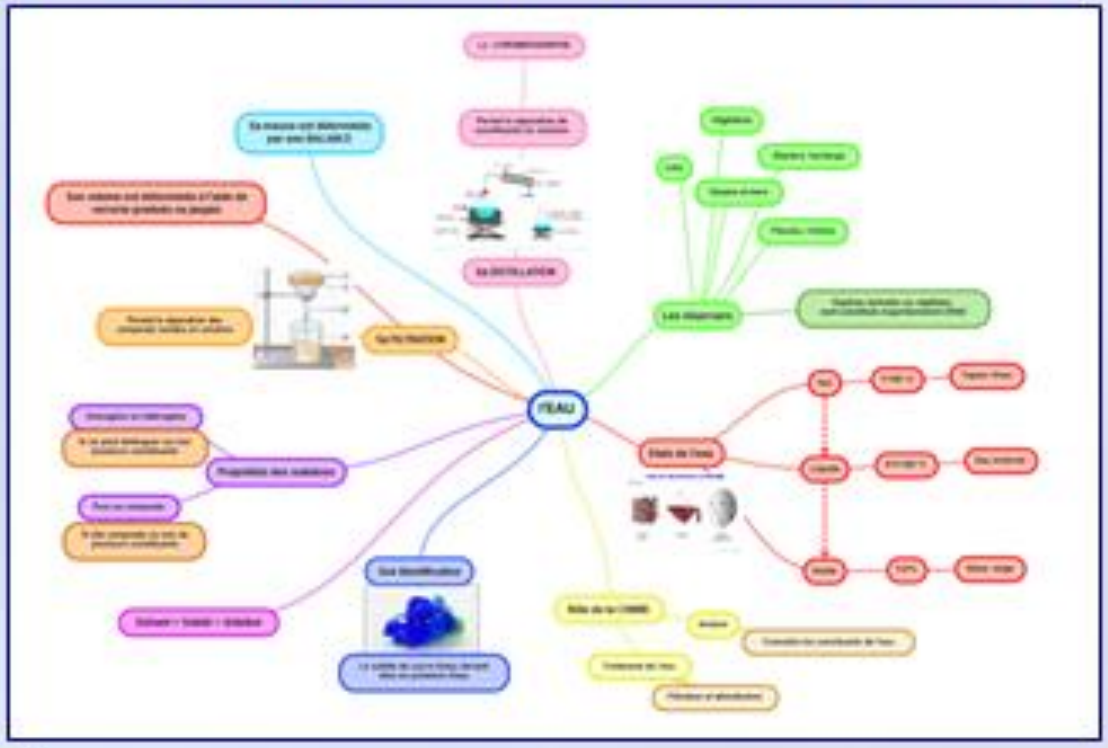
@ М.В.Добролюбова 6


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 3. МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО, ЛОГІЧНОГО ТА ФІЗИЧНОГО ПРОЄКТУВАННЯ БД
Лекція 5
«Методологія логічного проєктування реляційних баз даних»
@ М.В.Добролюбова

Фаза логічного проєктування РБД

Логічне проєктування – перетворення концептуального представлення в логічну структуру бази даних, включаючи проєктування відношень.

Логічне проєктування баз даних – процес конструювання загальної інформаційної моделі конкретної предметної області на основі окремих моделей даних користувачів, яка є незалежною від особливостей СУБД, що реально використовується, та інших фізичних умов.



@ М.В.Добролюбова

2

Структура процедури проєктування БД

Фаза 1 Концептуальне проєктування бази даних

Етап 1. Створення локальної концептуальної моделі даних, виходячи з уявлень предметної області кожного з типів користувачів

Фаза 2 Логічне проєктування бази даних (для реляційної моделі)

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Етап 3. Створення та перевірка глобальної логічної моделі даних

Фаза 3 Фізичне проєктування бази даних (з використанням реляційної СУБД)

Етап 4. Перенесення глобальної логічної моделі даних в середовище цільової СУБД

Етап 5. Проектування фізичного представлення бази даних

Етап 6. Розробка механізмів захисту

Етап 7. Організація моніторингу та налаштування функціонування системи

© М.В.Добролюбова 3

Структура процедури проєктування БД

Фаза 2 Логічне проєктування бази даних (для реляційної моделі)

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Завдання 2.1. Перетворення локальної концептуальної моделі даних в локальну логічну модель.
Завдання 2.2. Визначення набору відношень виходячи зі структури локальної логічної моделі даних.
Завдання 2.3. Перевірка моделі за допомогою правил нормалізації.
Завдання 2.4. Перевірка моделі щодо транзакцій користувачів.
Завдання 2.5. Створення діаграм «сутність-зв'язок».
Завдання 2.6. Визначення вимог підтримки цілісності даних.
Завдання 2.7. Обговорення розроблених локальних логічних моделей даних з кінцевими користувачами.

Етап 3. Створення та перевірка глобальної логічної моделі даних

Завдання 3.1. Злиття локальних логічних моделей даних в єдину глобальну модель даних.
Завдання 3.2. Перевірка глобальної логічної моделі даних.
Завдання 3.3. Перевірка можливостей розширення моделі у майбутньому.
Завдання 3.4. Створення остаточного варіанту діаграми «сутність-зв'язок».
Завдання 3.5. Обговорення глобальної логічної моделі даних з користувачами.

© М.В.Добролюбова 4

Структура процедури проєктування БД

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Мета – побудова логічної моделі даних на основі концептуальної моделі даних, що відображає уявлення окремого користувача про предметну галузь застосування, перевірка отриманої моделі за допомогою методів нормалізації та контролю виконання транзакцій.

Завдання 2.1. Перетворення локальної концептуальної моделі даних в локальну логічну модель.

Мета – доопрацювання локальних концептуальних моделей з метою видалення з них небажаних елементів; перетворення отриманих моделей у локальні логічні моделі даних.

Дії, які виконуються в рамках завдання:

1. Видалення зв'язків типу ∞ : ∞.
2. Видалення складних зв'язків.
3. Видалення рекурсивних зв'язків.
4. Видалення зв'язків з атрибутами.
5. Видалення множинних атрибутів.
6. Повторна перевірка зв'язків типу 1 : 1.
7. Видалення надлишкових зв'язків.

Поняття, що використовуються при виконанні завдання:

Складний зв'язок – це зв'язок, що існує між трьома та більше типами сутностей.

Рекурсивні зв'язки – це зв'язки, в яких сутність деякого типу взаємодіє сама з собою.

Множинні атрибути – це атрибути, які можуть мати водночас декілька значень для одного й того самого екземпляра сутності.

Надлишковий зв'язок – це зв'язок, при якому одна і та ж інформація може бути отримана не тільки через нього, але і за допомогою іншого зв'язку.

© М.В.Добролюбова 5

Структура процедури проєктування БД

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Завдання 2.2. Визначення набору відношень виходячи зі структури локальної логічної моделі даних.

Мета – визначення набору відношень на основі локальної логічної моделі даних.

Дії, які виконуються в рамках завдання:

1. Створення відношень для сильних типів сутностей.
2. Створення відношень для слабких типів сутностей.
3. Визначення батьківських і дочірніх сутностей в бінарних зв'язках типу «1:1».
4. Визначення батьківських і дочірніх сутностей в бінарних зв'язках типу «1: ∞».
5. Визначення батьківських і дочірніх сутностей в зв'язках типу «суперклас/підклас».

Поняття, що використовуються при виконанні завдання:

Батьківська сутність – це сутність , яка передає копію набору значень свого первинного ключа у відношення, що представляє дочірню сутність, де ці значення будуть грати роль зовнішнього ключа.

@ М.В.Добролюбова 6

Структура процедури проєктування БД

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Завдання 2.3. Перевірка моделі за допомогою правил нормалізації.

Мета – перевірка локальної логічної моделі даних з використанням технології нормалізації.

Основні етапи процесу нормалізації:

1. Приведення до першої нормальної форми (1НФ), що дозволяє видалити з відношень повторювані групи атрибутів.
2. Приведення до другої нормальної форми (2НФ), що дозволяє усунути часткову залежність атрибутів від первинного ключа.
3. Приведення до третьої нормальної форми (3НФ), що дозволяє усунути транзитивну залежність атрибутів від первинного ключа.
4. Приведення до нормальної форми Бойса-Кодда (НФБК), що дозволяє видалити з функціональних залежностей аномалії, які залишилися.

Поняття, що використовуються при виконанні завдання:

Нормалізація – це процедура прийняття рішень про те, які саме атрибути повинні бути об'єднані для представлення сутностей кожного типу.

@ М.В.Добролюбова 7

Структура процедури проєктування БД

Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів

Завдання 2.4. Перевірка моделі щодо транзакцій користувачів.

Мета – переконатися в тому, що локальна логічна модель даних дозволяє виконати всі транзакції, передбачені цим представленням користувача.

Підходи до перевірки моделей даних на відповідність необхідним транзакціям:

1. Виконання перевірки того, що дана логічна модель надає всю інформацію, необхідну для виконання кожної з транзакцій.
2. Нанесення безпосередньо на ER-діаграми всіх шляхів, які будуть потрібні для виконання кожної з транзакцій.

Завдання 2.5. Створення діаграм "сутність-зв'язок".

Мета – створення остаточного варіанту діаграм "сутність-зв'язок" (ER-діаграм), які є локальним логічним представленням даних, що використовуються окремими користувачами програми.

Завдання 2.6. Визначення вимог підтримки цілісності даних.

Мета – визначення обмежень, що накладаються в представленнях користувачів вимогою збереження цілісності даних.

Типи обмежень цілісності даних:

1. Обов'язкові дані. 2. Обмеження для доменів атрибутів. 3. Цілісність сутностей. 4. Довідкова цілісність. 5. Бізнес-правила.

Завдання 2.7. Обговорення розроблених локальних логічних моделей даних з кінцевими користувачами.

Мета – переконатися, що створені локальні моделі даних точно відображають уявлення користувачів про предметну область додатку.

@ М.В.Добролюбова 8

Структура процедури проєктування БД

Етап 3. Створення та перевірка глобальної логічної моделі даних

Мета – об'єднання окремих локальних логічних моделей даних в єдину глобальну логічну модель даних, що представляє ту частину підприємства, яка охоплюється даним додатком.

Завдання 3.1. Злиття локальних логічних моделей даних в єдину глобальну модель даних.

Мета – об'єднати окремі локальні логічні моделі даних в єдину глобальну логічну модель даних (наприклад, підприємства).

Дії, які виконуються в рамках завдання:

1. Аналіз імен сутностей та їх первинних ключів.
2. Аналіз імен зв'язків.
3. Злиття загальних сутностей з окремих локальних моделей.
4. Включення (без злиття) сутностей, унікальних для кожного локального представлення.
5. Злиття загальних зв'язків з окремих локальних моделей.
6. Включення (без злиття) зв'язків, унікальних для кожного локального представлення.
7. Перевірка на наявність пропущених сутностей та зв'язків.
8. Перевірка коректності зовнішніх ключів.
9. Перевірка дотримання обмежень цілісності.
10. Виконання креслення глобальної логічної моделі даних.
11. Оновлення документації.

Структура процедури проєктування БД

Етап 3. Створення та перевірка глобальної логічної моделі даних

Завдання 3.2. Перевірка глобальної логічної моделі даних.

Мета – перевірка глобальної логічної моделі даних за допомогою методів нормалізації та контроль можливості виконання необхідних транзакцій.

Завдання 3.3. Перевірка можливостей розширення моделі у майбутньому.

Мета – визначення ймовірності внесення будь-яких істотних змін у створену модель даних у майбутньому та оцінка того, наскільки дана модель пристосована для цього.

Завдання 3.4. Створення остаточного варіанту діаграми "сутність-зв'язок".

Мета – створення остаточного варіанту діаграми "сутність-зв'язок", що відображає глобальну логічну модель даних підприємства.

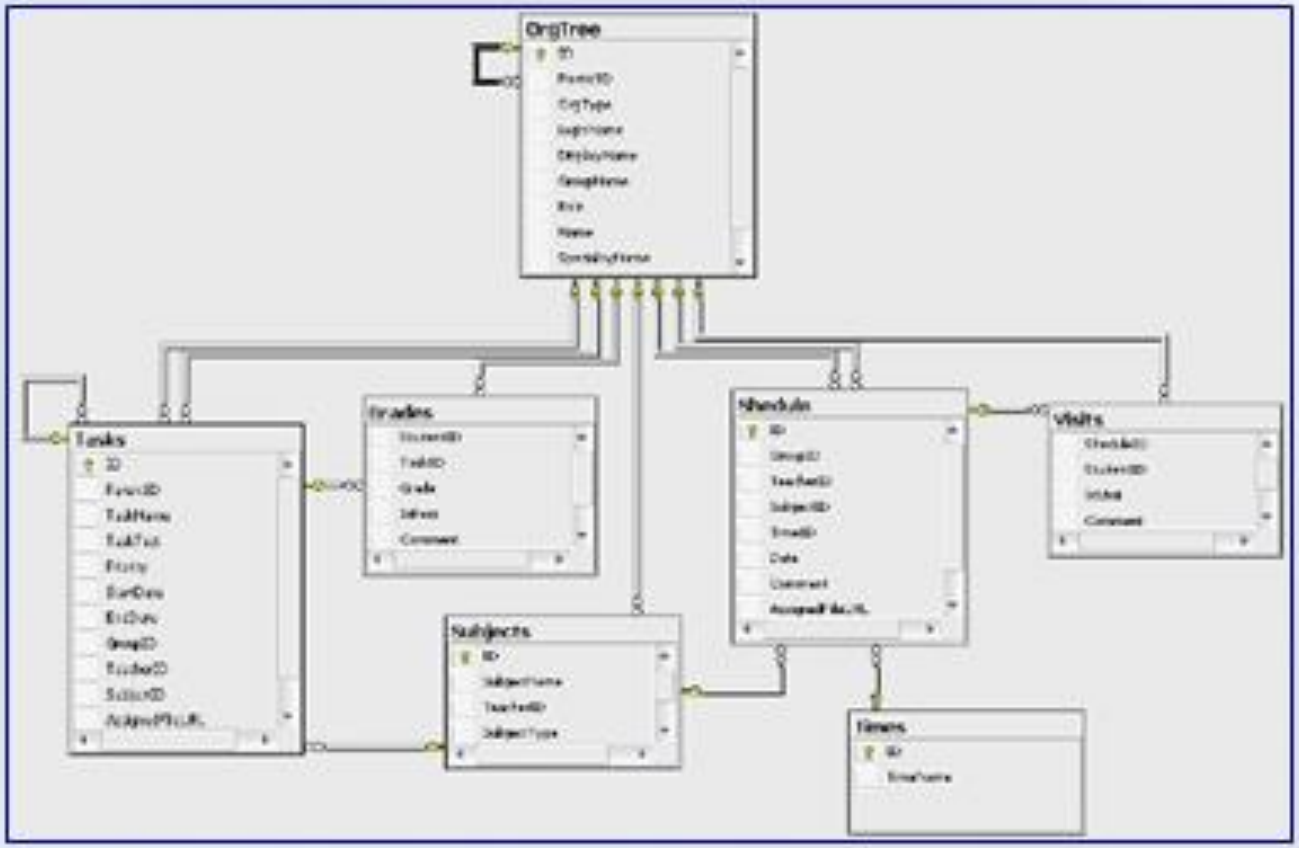
Завдання 3.5. Обговорення глобальної логічної моделі даних з користувачами.

Мета – переконатися, що створена глобальна логічна модель даних адекватно відображає моделюєму частину інформаційної структури підприємства.

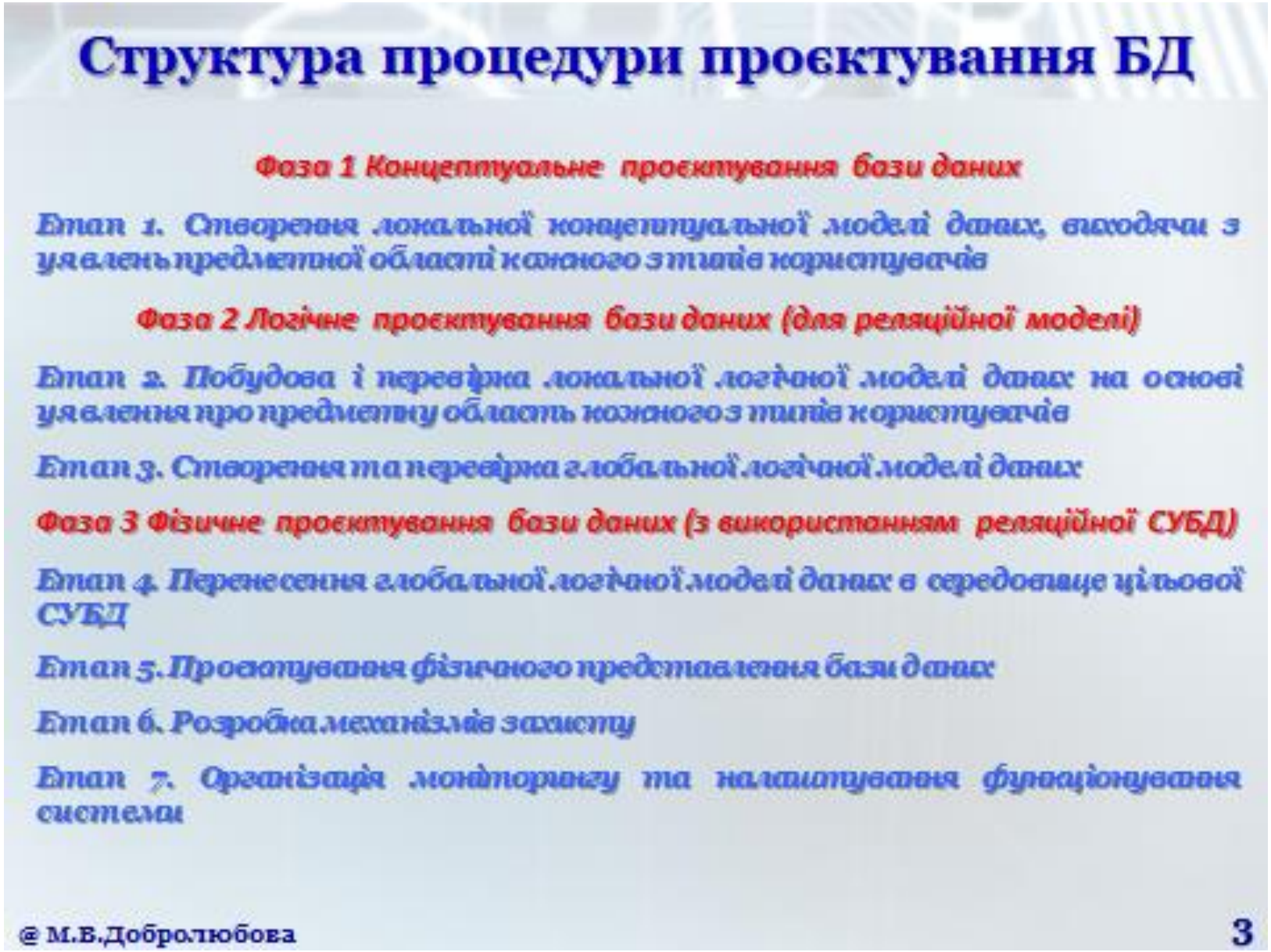

ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 3. МЕТОДОЛОГІЯ КОНЦЕПТУАЛЬНОГО, ЛОГІЧНОГО ТА ФІЗИЧНОГО ПРОЄКТУВАННЯ БД
Лекція 6
«Методологія фізичного проєктування реляційних баз даних»
@ М.В.Добролюбова

Фаза фізичного проєктування РБД

Фізичне проєктування – прийняття рішення про те, як логічна модель буде фізично реалізована в базі даних, що створюється за допомогою обраної СУБД.



@ М.В.Добролюбова 2


Структура процедури проєктування БД
Фаза 1 Концептуальне проєктування бази даних
Етап 1. Створення локальної концептуальної моделі даних, виходячи з уявлень предметної області кожного з типів користувачів
Фаза 2 Логічне проєктування бази даних (для реляційної моделі)
Етап 2. Побудова і перевірка локальної логічної моделі даних на основі уявлення про предметну область кожного з типів користувачів
Етап 3. Створення та перевірка глобальної логічної моделі даних
Фаза 3 Фізичне проєктування бази даних (з використанням реляційної СУБД)
Етап 4. Перенесення глобальної логічної моделі даних в середовище цільової СУБД
Етап 5. Проектування фізичного представлення бази даних
Етап 6. Розробка механізмів захисту
Етап 7. Організація моніторингу та налаштування функціонування системи
@ М.В.Добролюбова
3

Структура процедури проєктування БД

Фаза 3 Фізичне проєктування бази даних (з використанням реляційної СУБД)

Етап 4. Перенесення глобальної логічної моделі даних в середовище цільової СУБД

Завдання 4.1. Проєктування основних таблиць в середовищі цільової СУБД.
Завдання 4.2. Реалізація бізнес-правил підприємства в середовищі цільової СУБД.

Етап 5. Проектування фізичного представлення бази даних

Завдання 5.1. Аналіз транзакцій.
Завдання 5.2. Вибір файлової структури.
Завдання 5.3. Визначення вторинних індексів.
Завдання 5.4. Аналіз необхідності введення контрольованої надлишковості даних.
Завдання 5.5. Визначення вимог дискової пам'яті.

Етап 6. Розробка механізмів захисту

Завдання 6.1. Розробка користувальницьких представлень (видів).
Завдання 6.2. Визначення прав доступу.

Етап 7. Організація моніторингу та налаштування функціонування системи

@ М.В.Добролюбова 4

Структура процедури проєктування БД

Етап 4. Перенесення глобальної логічної моделі даних в середовище цільової СУБД

Мета – створення базової функціональної схеми реляційної бази даних на основі глобальної логічної моделі даних.

Завдання 4.1. Проектування таблиць бази даних у середовищі цільової СУБД.

Мета – визначення способу представлення в цільовій СУБД відношень, виокремлених у глобальній логічній моделі даних.

Дії, які виконуються в рамках завдання:

1. Визначення кожного виділеного в глобальній логічній моделі даних відношення.
2. Прийняття рішення про способи реалізації таблиць бази даних.

Способи реалізації таблиць та вимог цілісності за посиланнями:

1. Опис стандартизованою мовою SQL.
2. Реалізація з використанням тригерів.
3. Реалізація у конкретному середовищі СУБД.
4. Реалізація з використанням унікальних індексів.

Завдання 4.2. Реалізація бізнес-правил підприємства у середовищі цільової СУБД.

Мета – реалізація бізнес-правил у середовищі цільової СУБД.

5

Структура процедури проєктування БД

Етап 5. Проектування фізичного представлення бази даних

Мета – визначення файлової структури методів доступу, які будуть використуватись для подання таблиць бази даних.

Показники, що використовуються для оцінки досягнутої ефективності:

1. Пропускна здатність транзакцій.
2. Час відповіді.
3. Дискова пам'ять.

Основні компоненти обладнання:

1. Оперативна пам'ять.
2. Процесор.
3. Дисковий ввод / вивід.
4. Мережа.

Завдання 5.1. Аналіз транзакцій.

Мета – визначення функціональних характеристик транзакцій, які будуть виконуватися в базі даних, що проєктується, та виділення найбільш важливих з них.

@ М.В.Добролюбова 6

Структура процедури проєктування БД

Етап 5. Проектування фізичного представлення бази даних

Завдання 5.2. Вибір файлової структури.

Мета – визначення найбільш ефективного файлового представлення для кожної з таблиць бази даних.

Типи організації файлів:

1. Послідовні файли (heap).
2. Хешовані файли (hash).
3. Індексного-послідовні файли (ISAM).
4. Двійкові дерева (B+-Tree).

Поняття, що використовуються при виконанні завдання:

Послідовна невпорядкована організація файла означає довільне невпорядковане розміщення записів на диску.

Послідовна впорядкована організація файла передбачає розміщення записів у відповідності із значенням вказаного поля.

Хешовані записи зберігаються у відповідності із значенням деякої хеш-функції.

Хеш-функція – функція, яка перетворює вхідні дані будь-якого (як правило великого) розміру в дані фіксованого розміру.

Хешування – перетворення вхідного масиву даних довільної довжини у вихідний бітовий рядок фіксованої довжини.

Для кожного типу організації файлів використовується відповідний набір ***методів доступу*** – дій, які виконуються при збереженні або вилученні записів з файлу.

@ М.В.Добролюбова 7

Структура процедури проєктування БД

Етап 5. Проектування фізичного представлення бази даних

Завдання 5.2. Вибір файлової структури.

Поняття, що використовуються при виконанні завдання:

Індекс – структура даних, яка допомагає СУБД швидше виявити окремі записи в файлі, а тому дозволяє скоротити час виконання запитів користувача. Структура індекса пов'язана з певним ключем пошуку і містить записи, що складаються з ключового значення і адреси логічного запису в файлі, яка містить це ключове значення.

Індексний файл – це файл, який містить індексні записи.

Файл даних – це файл, який містить логічні записи.

Первинний індекс – це поле індексування, яке є ключовим полем послідовно впорядкованого файлу даних.

Індекс кластеризації – це поле індексування, яке не є ключовим полем файлу і з цієї причини одному індексному значенню може відповідати декілька записів файлу.

Вторинний індекс – це індекс, який визначений на полі файлу, що відмінне від поля його впорядкування.

Розріджений індекс – це індекс, що містить індексні записи лише для деяких ключових значень пошуку в даному файлі.

Щільний індекс – це індекс, що містить індексні записи для всіх ключових значень пошуку в даному файлі.

Індексно-послідовний файл (індексований послідовний файл) – це відсортований файл даних з первинним індексом.

8

Структура процедури проєктування БД

Етап 5. Проектування фізичного представлення бази даних

Завдання 5.2. Вибір файлової структури.

Поняття, що використовуються при виконанні завдання:

Дерево – структура, що використовується в СУБД для збереження даних і складається з ієрархії вузлів, в якій кожний вузол, за виключенням кореня, має батьківський вузол, а також один, декілька або жодного дочірнього вузла.

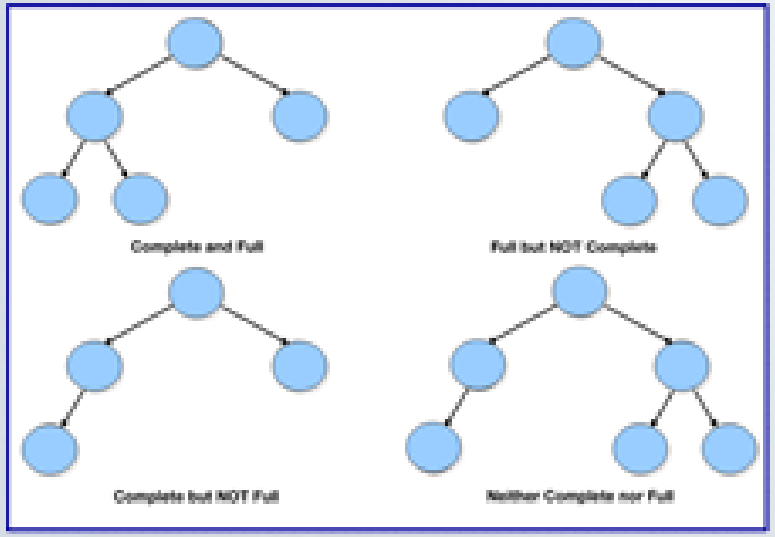
Лист – це вузол, який не має дочірніх вузлів.

Глибина дерева – максимальна кількість рівнів між коренем і листом.

Збалансоване дерево (B-дерево) – це дерево, глибина якого однакова для всіх листів.

Ступінь (порядок) дерева – це максимально припустима кількість дочірніх вузлів для кожного батьківського вузла.

Бінарне дерево (B⁺-дерево) – це дерево порядку 2, в якому кожен вузол має не більше двох дочірніх вузлів.



Структура процедури проєктування БД

Етап 3. Проектування фізичного представлення бази даних

Завдання 3.3. Визначення вторинних індексів.

Мета – визначення того, чи буде додавання вторинних індексів сприяти підвищенню продуктивності системи.

Поняття, що використовуються при виконанні завдання:

Вторинні індекси – це механізм визначення в таблицях бази даних додаткових ключів, які призначені для підвищення ефективності вибірки даних.

Завдання 3.4. Аналіз необхідності введення контрольованої надмірності даних.

Мета – визначення того, чи сприятиме підвищенню продуктивності системи внесення контрольованої надмірності даних, що здійснюється за рахунок зниження вимог до рівня їх нормалізації.

Завдання 3.4.1. Аналіз доцільності використання похідних даних.

Завдання 3.4.2. Аналіз доцільності дублювання атрибутів або об'єднання окремих відношень.

Завдання 3.5. Визначення вимог до дискової пам'яті.

Мета – визначення обсягу дискового простору, необхідного для розміщення бази даних.

Структура процедури проєктування БД

Етап 6. Розробка механізмів захисту

Мета – розробка механізмів захисту бази даних відповідно до вимог користувачів.

Завдання 6.1. Розробка представлень користувача (видів).

Мета – розробка представлень користувача (видів), які були виокремлені на першому етапі концептуального проєктування бази даних.

Завдання 6.2. Визначення прав доступу.

Мета – визначення прав доступу до таблиць бази даних для кожного з представлень користувачів.

Поняття, що використовуються при виконанні завдання:

Привілеї (права доступу) – це дії, які користувачу дозволено виконувати відносно конкретної таблиці або представлення.

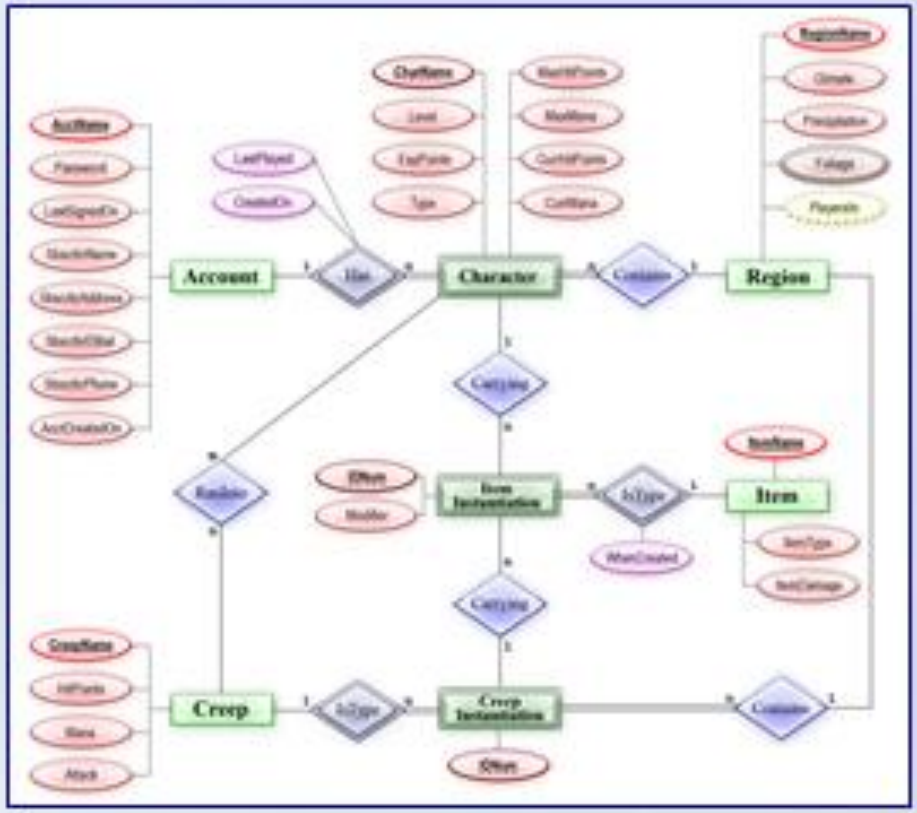
Етап 7. Організація моніторингу та налаштування функціонування системи

Мета – моніторинг функціонування операційної системи та досягнутого рівня продуктивності бази даних з метою усунення помилкових проектних рішень або відображення зміни вимог до системи.


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 4. МОДЕЛЬ «СУТНІСТЬ-ЗВ'ЯЗОК». НОРМАЛІЗАЦІЯ
Лекція 7
«ER-модель»
© М.В.Добролюбова

Концепції ER-моделі

Модель "сутність-зв'язок" (Entity-Relationship model, або ER-модель) – високорівнева концептуальна модель даних для спрощення задачі проєктування баз даних.



Пітер Пін-Шен Чен
1976 рік

© М.В.Добролюбова

2

Концепції ER-моделі

Основні концепції ER-моделі

- типи сутностей
- атрибути
- типи зв'язків

Типи сутностей – об'єкт або концепція, які характеризуються на даному підприємстві як такі, що існують незалежно.

Приклади сутностей з фізичним та концептуальним існуванням

Фізичне існування	Концептуальне існування
Працівник	Огляд об'єкта нерухомості
Об'єкт нерухомості	Інспекція об'єкта нерухомості
Клієнт	Продаж об'єкта нерухомості
Деталь	Робочий стаж
Постачальник	
Виріб	

Сутність – екземпляр типу сутності, який може бути ідентифікований унікальним чином.

Слабкий тип сутності – тип сутності, існування якого залежить від якогось іншого типу сутності.

Сильний тип сутності – тип сутності, існування якого не залежить від якогось іншого типу сутності.

Слабкі і сильні типи сутностей на ER-діаграмі

@ М.В.Добролюбова 3

Концепції ER-моделі

Основні концепції ER-моделі

- типи сутностей
- атрибути
- типи зв'язків

Атрибут – властивість типу сутності або типу зв'язку.

Домен атрибута – набір значень, які можуть бути присвоєні атрибуту.

Простий атрибут (атомарний) – це атрибут, що складається з одного компонента з незалежним існуванням.

Складовий атрибут – це атрибут, що складається з декількох компонентів, кожен з яких характеризується незалежним існуванням.

Однозначний атрибут – це атрибут, який містить одне значення для однієї сутності.

Багатозначний атрибут – це атрибут, який містить кілька значень для однієї сутності.

Похідний атрибут – це атрибут, який представляє значення, похідне від значення пов'язаного з ним атрибута або деякої множини атрибутів, що належать деякому (не обов'язково даному) типу сутності.

© М.В.Добролюбова 4

Концепції ER-моделі

Основні концепції ER-моделі

- типи сутностей
- атрибути
- типи зв'язків

Ключ – елемент даних, який дозволяє унікально ідентифікувати окремі екземпляри деякого типу сутності.

Потенційний ключ – атрибут або набір атрибутів, який унікально ідентифікує окремі екземпляри типу сутності.

Первинний ключ – потенційний ключ, який обраний як первинний ключ.

Складовий ключ – потенційний ключ, який складається з двох або більше атрибутів.

@ М.В.Добролюбова 5

Концепції ER-моделі

Основні концепції ER-моделі

- типи сутностей
- атрибути
- типи зв'язків

Приклади сутностей на діаграмі

Приклад визначення атрибутів, пов'язаних з типами сутностей

Staff(Staff_No, FName, LName, Address, Tel_No, Sex, DOB, Position, NIN, Salary)	
Первинний ключ:	Staff_No
Альтернативний ключ:	FName, LName, DOB
Альтернативний ключ:	NIN
Складовий атрибут:	Name(FName, LName)
Похідний атрибут:	Total_Staff

© М.В.Добролюбова 6

Концепції ER-моделі

Основні концепції ER-моделі

- типи сутностей
- атрибути
- типи зв'язків

Тип зв'язку – осмислена асоціація між сутностями різнихтипів.

Зв'язок – асоціація між сутностями, яка включає по одній сутності з кожного типу сутності, що бере участь у зв'язку.

© М.В.Добролюбова 7

Концепції ER-моделі

Основні концепції ER-моделі

- типи сутностей
- атрибути
- типи зв'язків

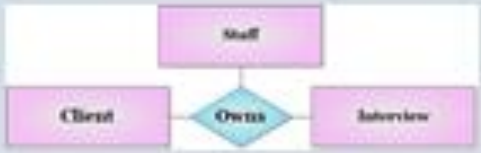
Ступінь зв'язку – це кількість сутностей, які охоплені даним зв'язком.

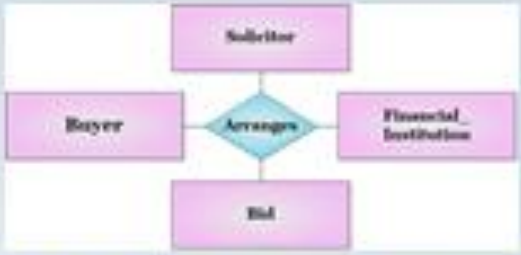
Бінарний (binary) зв'язок – зв'язок зі ступенем два.

Тернарний (ternary) зв'язок – зв'язок зі ступенем три.

Кватернарний (quaternary) зв'язок – зв'язок зі ступенем чотири.

Рекурсивний зв'язок (unary) – це зв'язок, в якому одні й ті самі сутності беруть участь декілька разів і в різних ролях.





Приклади зв'язків з різними ступенями

Приклад рекурсивного зв'язку з ролевими іменами Інспектор та Підлеглий

Приклад сутностей, пов'язаних різними зв'язками, із зазначенням ролевих імен

@ М.В.Добролюбова

8

Структурні обмеження

Основні типи обмежень

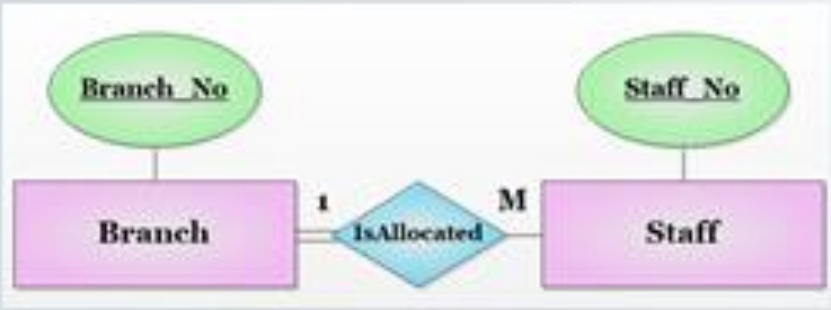
- кардинальність
- ступінь участі

Показник кардинальності – показник, який описує кількість можливих зв'язків для кожної з сутностей-учасниць.

Бізнес-правила (business rules) організації – правила, що визначають показники кардинальності.

Ступінь участі – визначає, чи залежить існування деякої сутності від участі в зв'язку деякої іншої сутності.

Приклади зв'язків



Приклад ступеня участі

@ М.В.Добролюбова 9

Проблеми ER-моделювання

Основні типи пасток

- пастка розгалуження
- пастка розриву

Пастка розгалуження має місце в тому випадку, коли модель відображає зв'язок між типами сутностей, але шлях між окремими сутностями цього типу визначено неоднозначно.

Пастка розриву з'являється в тому випадку, коли в моделі передбачається наявність зв'язку між типами сутностей, але не існує шляху між окремими сутностями цих типів.

Приклад пастки розгалуження

Приклад пастки розриву

© М.В.Добролюбова

10


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 1
Основні відомості про бази даних та СУБД
Тема 4. МОДЕЛЬ «СУТНІСТЬ-ЗВ'ЯЗОК». НОРМАЛІЗАЦІЯ
Лекція 8
«Поняття нормалізації бази даних»
@ М.В.Добролюбова

Мета нормалізації

Нормалізація – метод створення набору відношень із заданими властивостями на основі вимог до даних.

Едгар Кодд

1НФ	
2НФ	1972
3НФ	
НФБК	1974
4НФ	1977
5НФ	1979

Реймонд Бойс

© М.В.Добролюбова 2

Надмірність даних та аномалії оновлення

Мета проектування РБД – групування атрибутів відношення задля мінімізації надмірності даних, тобто скорочення об'єму пам'яті, необхідного для фізичного збереження відношень, представлених у вигляді таблиць.

```
Staff          (Staff_No, SName, SAddress, Position, Salary, Branch_No)
Branch         (Branch_No, BAddress, Tel_No)
Staff_Branch   (Staff_No, SName, SAddress, Position, Salary, Branch_No, BAddress, Tel_No)
```

Staff_No	SName	SAddress	Position	Salary	Branch_No
SL21	John White	19 Taylor St, London	Manager	30000	B5
SG37	Anna Beech	81 George St, Glasgow	Snr Asst	12000	B3
SG14	David Ford	63 Ashby St, Glasgow	Deputy	18000	B3
SA9	Mary Howe	2 Elm Pl, Aberdeen	Assistant	9000	B7
SG5	Susan Brand	5 Gt Western Rd, Glasgow	Manager	24000	B3
SL41	Julie Lee	28 Malvern St, Kilburn	Assistant	9000	B5

Branch_No	BAddress	Tel_No
B5	22 Deer Rd, London	0171-8861212
B7	16 Argyll St, Aberdeen	0122-6772771
B3	163 Main St, Glasgow	0141-6654432

Staff_No	SName	SAddress	Position	Salary	Branch_No	BAddress	Tel_No
SL21	John White	19 Taylor St, London	Manager	30000	B5	22 Deer Rd, London	0171-8861212
SG37	Anna Beech	81 George St, Glasgow	Snr Asst	12000	B3	163 Main St, Glasgow	0141-6654432
SG14	David Ford	63 Ashby St, Glasgow	Deputy	18000	B3	163 Main St, Glasgow	0141-6654432
SA9	Mary Howe	2 Elm Pl, Aberdeen	Assistant	9000	B7	16 Argyll St, Aberdeen	0122-6772771
SG5	Susan Brand	5 Gt Western Rd, Glasgow	Manager	24000	B3	163 Main St, Glasgow	0141-6654432
SL41	Julie Lee	28 Malvern St, Kilburn	Assistant	9000	B5	22 Deer Rd, London	0171-8861212

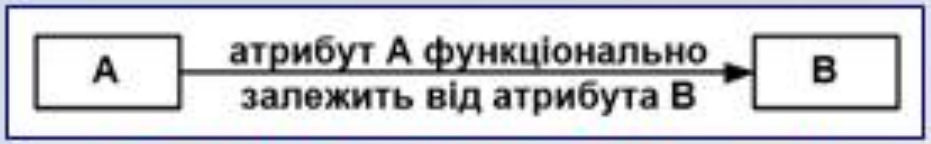
Надмірність даних та аномалії оновлення

Властивість з'єднання без втрат (lossless-join) – це властивість, яка дозволяє відновити будь-який кортеж вихідного відношення, використовуючи відповідні кортежі менших відношень, що отримані в результаті декомпозиції.

Властивість збереження залежності (dependency preservation) – це властивість, яка дозволяє зберегти обмеження, накладені на вихідне відношення, за допомогою накладення деяких обмежень на кожне з менших відношень, отриманих після декомпозиції.

Властивість збереження залежності (dependency preservation) – це властивість, яка дозволяє зберегти обмеження, накладені на вихідне відношення, за допомогою накладення деяких обмежень на кожне з менших відношень, отриманих після декомпозиції.

Функціональна залежність описує зв'язок між атрибутами відношення. Наприклад, якщо у відношенні *R*, що містить атрибути *A* та *B*, атрибут *B* функціонально залежить від атрибуту *A* (*A* → *B*), то кожне значення атрибута *A* пов'язано тільки з одним значенням атрибута *B*.



Діаграма функціональної залежності

Детермінант функціональної залежності – це атрибут або група атрибутів, розміщена на діаграмі функціональної залежності зліва від символу стрілки.

Процес нормалізації

Нормалізація – це формальний метод аналізу відношень на основі їх первинного ключа (або потенційних ключів, як у випадку НФБК) та існуючих функціональних залежностей.

Ненормалізована форма (ННФ) – таблиця, що містить одну або кілька повторюваних груп даних.

Перша нормальна форма (1НФ) – відношення, в якому на перетині кожного рядка та кожного стовпця міститься тільки одне значення.

Друга нормальна форма (2НФ) – це відношення, яке знаходиться в першій нормальній формі і кожен атрибут якого, що не входить до складу первинного ключа, характеризується повною функціональною залежністю від цього первинного ключа.

Повна функціональна залежність $A \rightarrow B$ – це функціональна залежність, при якій видалення будь-якого атрибуту з A призводить до втрати цієї залежності.

Часткова функціональна залежність $A \rightarrow B$ – це функціональна залежність, яка зберігається при видаленні в A довільного атрибуту.

Staff_No, SName → Branch_No

Третя нормальна форма (3НФ) – це відношення, яке знаходиться в першій та другій нормальних формах та не має атрибутів, що не входять у первинний ключ, та які б знаходились у транзитивній функціональній залежності від цього первинного ключа.

Транзитивна залежність. Якщо для атрибутів A, B та C деякого відношення існують залежності виду $A \rightarrow B$ та $B \rightarrow C$, то кажуть, що атрибут C транзитивно залежить від атрибута A через атрибут B (за умови, що атрибут A функціонально не залежить ні від атрибута B, ні від атрибута C).

Staff_No → Branch_No *та* *Branch_No → BAddress*

© М.В.Добролюбова 5

Процес нормалізації

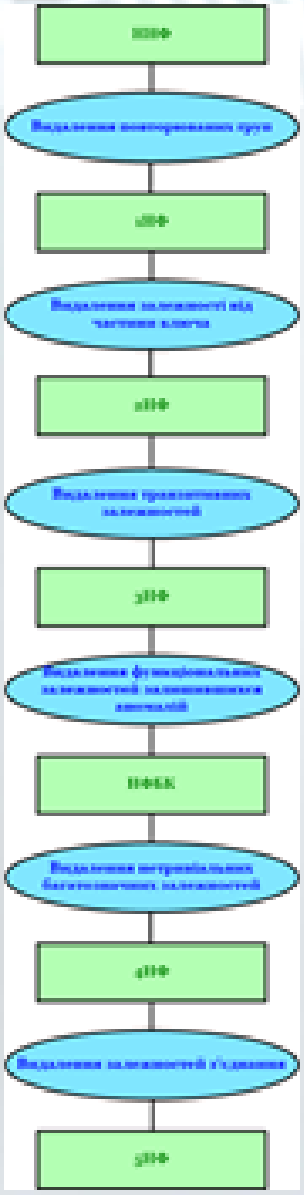
Нормальна форма Бойса-Кодда (НФБК) – це відношення, в якому кожен його детермінант є потенційним ключем.

Четверта нормальна форма (4НФ) – це відношення, яке знаходиться в нормальній формі Бойса-Кодда та не містить нетривіальних багатозначних залежностей.

Багатозначна залежність – це така залежність між атрибутами *A*, *B* та *C* деякого відношення, при якій для кожного значення атрибута *A* існують відповідні набори значень атрибутів *B* та *C*, причому обидва цих набори не залежать один від одного.

П'ята нормальна форма (5НФ) – це відношення, яке не містить залежностей з'єднання.

Залежність з'єднання – це така ситуація, при якій декомпозиція відношення може супроводжуватися генерацією помилкових рядків при зворотному з'єднанні декомпозованих відношень за допомогою операції природного з'єднання.



© М.В.Добролюбова

6

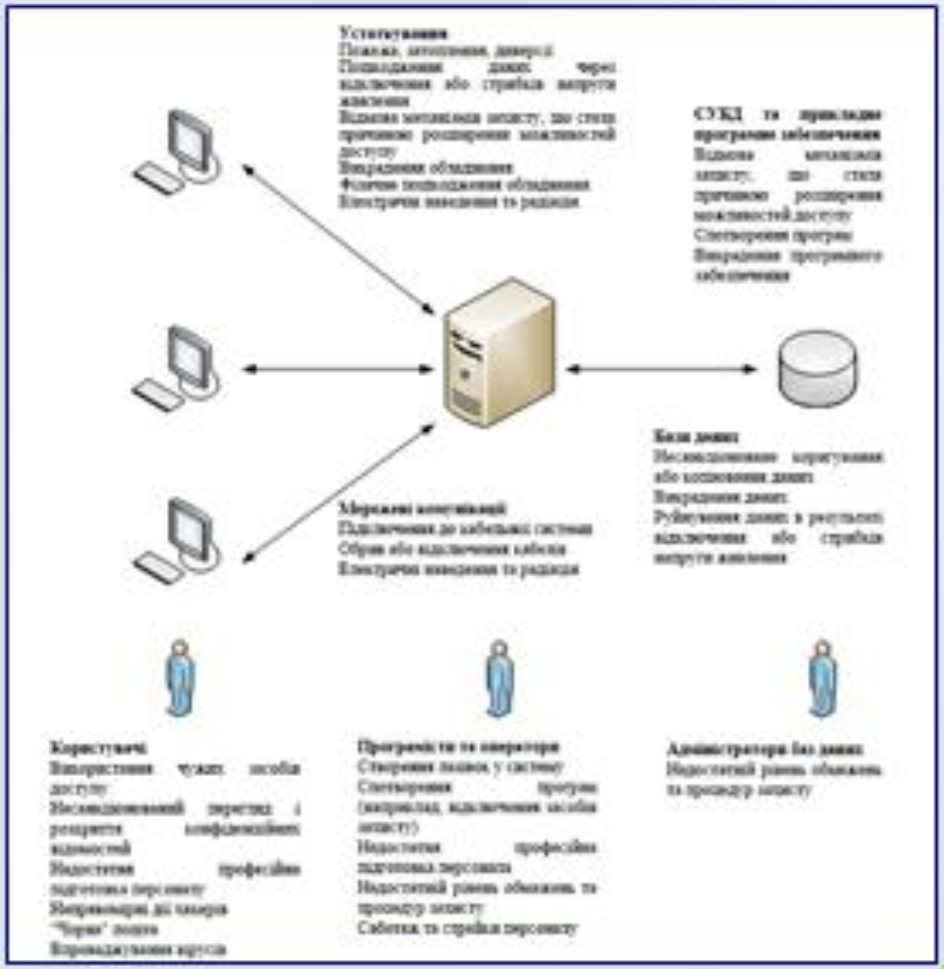

ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 1. ЗАХИСТ БД, КЕРУВАННЯ ТРАНЗАКЦІЯМИ, ОБРОБКА ЗАПИТІВ
Лекція 9
«Захист баз даних»
@ М.В.Добролюбова

Типи небезпеки

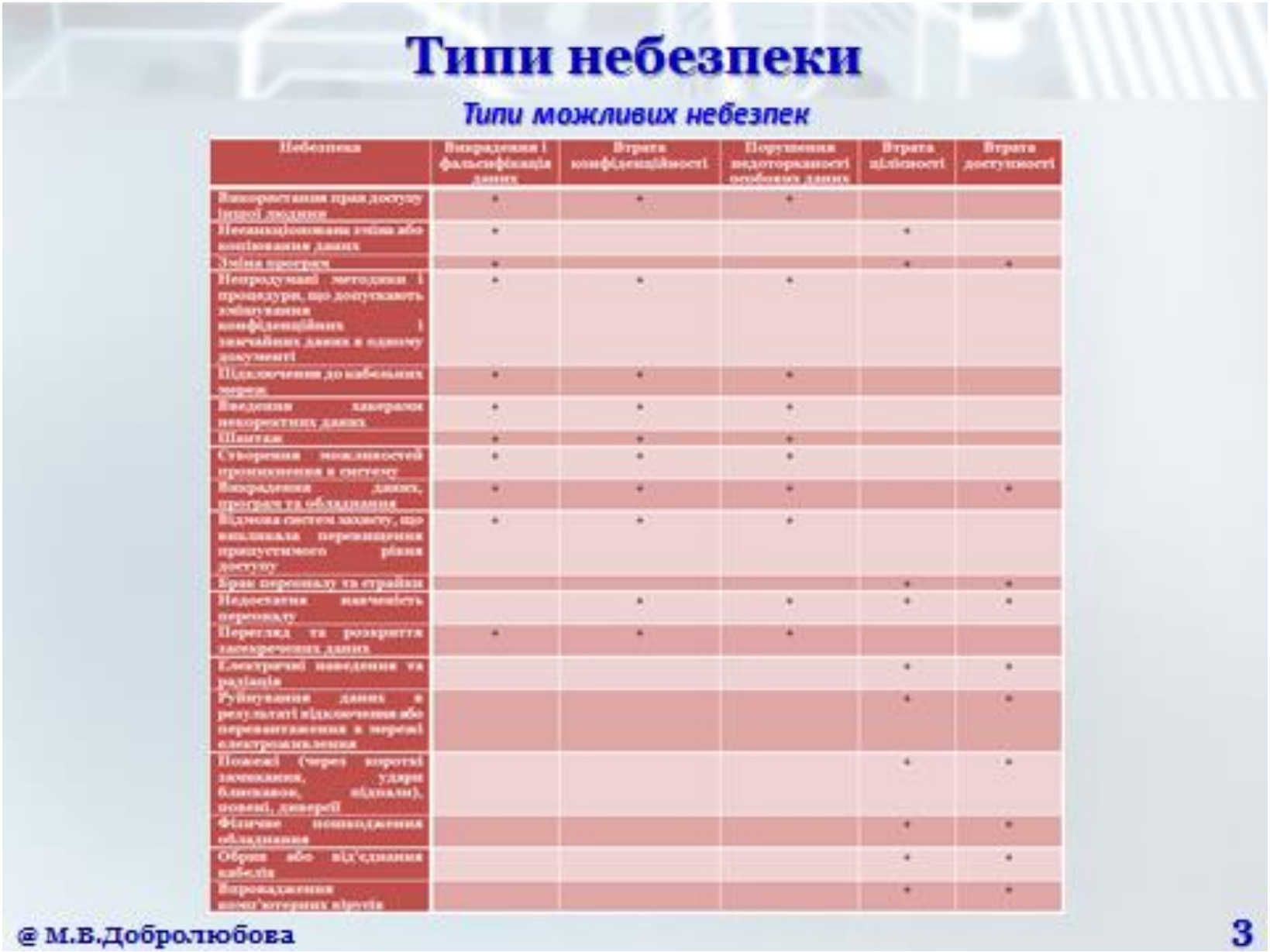
Захист бази даних – забезпечення захищеності бази даних проти будь-яких навмисних або ненавмисних загроз за допомогою різних комп'ютерних та некомп'ютерних засобів.

Небезпека – будь-яка ситуація або подія, навмисна або ненавмисна, яка здатна несприятливо вплинути на систему і, як наслідок, на всю організацію.

Узагальнена схема потенційних загроз, на які наражаються комп'ютерні системи

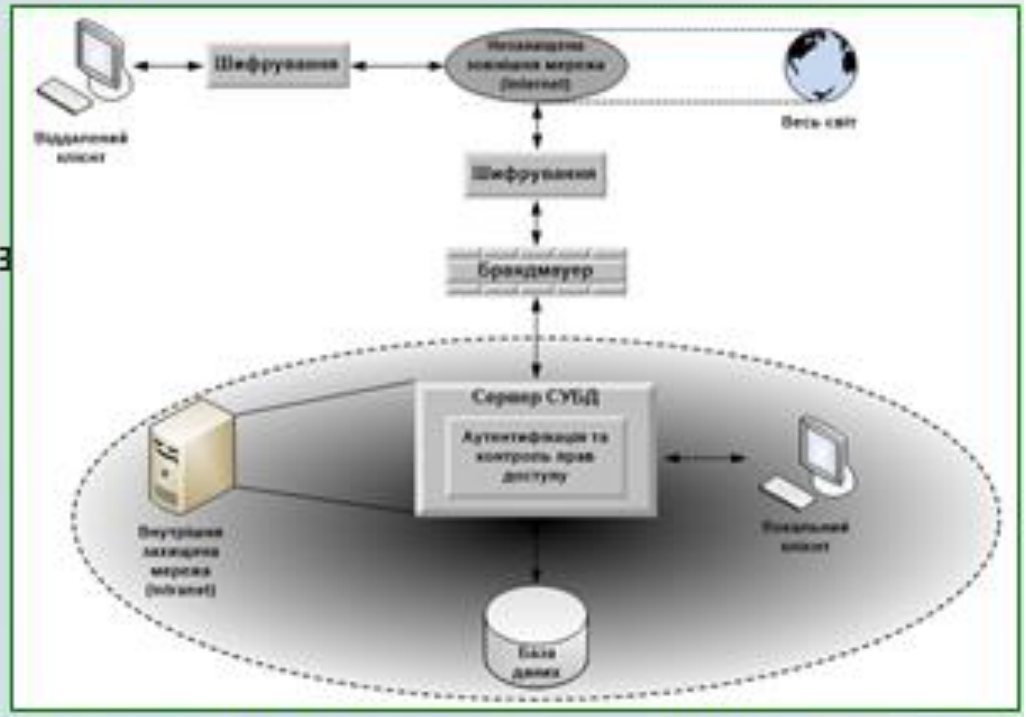


© М.В.Добролюбова


Типи небезпеки
Типи можливих небезпек
Небезпека
Використання прав доступу іншої людини
Несанкціонована зміна або копіювання даних
Зміна програм
Непродумані методики і процедури, що допускають змішування конфіденційних і звичайних даних в одному документі
Підключення до кабельних мереж
Введення хибних або некоректних даних
Шантаж
Створення можливостей проникнення в систему
Викрадення даних, програм та обладнання
Відмова систем захисту, що викликала перевищення припустимого рівня доступу
Брак персоналу та страйки
Недостатня кваліфікація персоналу
Перегляд та розкриття засекречених даних
Електричні наведення та радіація
Руйнування даних в результаті відключення або перевантаження в мережі електроживлення
Пожежі (через короткі замикання, удари блискавок, вибухи), повені, диверсії
Фізичне пошкодження обладнання
Обрив або від'єднання кабелів
Перехоплення електромагнітних відгуків
@ М.В.Добролюбова
3

Комп'ютерні засоби контролю

1. Авторизація користувачів
2. Представлення
3. Створення резервних копій та відновлення
4. Підтримка цілісності
5. Шифрування
6. Допоміжні процедури:
   - аудит
   - встановлення нового ППЗ
   - встановлення/модернізація СПЗ



Загальна схема типової, багатокористувальницької, комп'ютерної системи

@ М.В.Добролюбова 4

Некомп'ютерні засоби контролю

1. Заходи забезпечення безпеки та планування захисту від непередбачених обставин
2. Контроль за персоналом
3. Захист приміщень та сховищ
4. Гарантійні угоди
5. Договори про супровід
6. Контроль за фізичним доступом

@ М.В.Добролюбова

5

Захист ПК

1. Використання індивідуальних ідентифікаторів користувачів та відповідних особистих паролів
2. Спеціальні сховища для захищеного зберігання будь-яких необхідних даних, програмного забезпечення та комп'ютерного обладнання
3. Особиста відповідальність кожного користувача за створення копій даних та зберігання програмного забезпечення і комп'ютерного обладнання
4. Чіткі вказівки про процедури, які повинні застосовуватися до будь-якого програмного забезпечення, перед тим як воно буде допущено до встановлення системи

@ М.В.Добролюбова

6

СУБД та захист у Web

1. Проксі-сервери

Проксі-сервер – це комп'ютер, який розміщується між Web-броузером та Web-сервером. У загальному випадку проксі-сервери призначені для вирішення задач підвищення продуктивності та фільтрації запитів.

2. Брандмауери

Брандмауер – це система, призначена для захисту від несанкціонованого доступу до (або з) закритої мережі.

3. Використання цифрового підпису

Цифровий підпис використовується для перевірки істинності походження даних та складається з двох частин: рядка бітів, який обчислений на основі "підписаних" даних, та закритого ключа приватної особи або організації, яка "підписала" ці дані.

4. Алгоритми створення дайджесту повідомлень цифрового підпису

Хеш-функція – це функція, яка здійснює перетворення масиву вхідних даних довільної довжини в (вихідний) бітовий рядок встановленої довжини, що виконується певним алгоритмом.

© М.В.Добролюбова 7

СУБД та захист у Web

5. Цифрові сертифікати

Цифровий сертифікат – це доповнення електронного повідомлення, яке використовується з метою забезпечення безпеки, – найчастіше для того, щоб довести, що відправник повідомлення дійсно є тим, за кого себе видає.

6. Використання церберів

Цербер – це сервер захищених облікових імен та паролів користувачів.

7. Використання технологій SSL та SHTTP

Протокол Secure Sockets Layer (SSL) – це протокол, який створює безпечне з'єднання між клієнтом та сервером, по якому може безпечно передаватися будь-яка кількість даних.

Протокол Secure HTTP (S-HTTP) – це протокол, який призначений для безпечної передачі окремих повідомлень.

8. Концепції мов програмування для захисту Web-додатків

Концепція "пісочниця" гарантує заборону на доступ до системних ресурсів несанкціонованого і, можливо, зловмисного додатку.

9. Засоби захисту технології ActiveX

ActiveX – це фреймворк для визначення програмних компонентів, придатних до використання з програм, написаних на різних мовах програмування.

@ М.В.Добролюбова 8


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 1. ЗАХИСТ БД, КЕРУВАННЯ ТРАНЗАКЦІЯМИ, ОБРОБКА ЗАПИТІВ
Лекція 10
«Управління транзакціями»
@ М.В.Добролюбова

Основні функції сучасної СУБД

Основні функції сучасної СУБД

1. Механізми підтримки транзакцій
2. Служби управління паралельністю
3. Засоби відновлення баз даних

Засоби відновлення та механізм управління паралельністю доступу призначені для захисту баз даних від переходу даних в неузгоджений стан або їх втрати

Відновлення бази даних – це процедура повернення бази даних до деякого коректного стану, що вживається в разі руйнування системи.

Причини руйнування системи

- відмови обладнання
- програмні помилки
- збої носіїв інформації

@ М.В.Добролюбова

2

Підтримка транзакцій

Транзакція – це дія або серія дій, що виконуються одним користувачем або прикладною програмою і здійснюють доступ бази даних або змінюють її вміст.

Різновиди транзакцій

- окрема програма
- частина алгоритму
- окрема команда

Ілюстрація концепції транзакцій

Staff	(Sno, FName, LName, Address, Tel_No, Position, Sex, DOB, Salary, NIN, Bno)
Property_for_Rent	(Pno, Street, Area, City, Pcode, Type, Rooms, Rent, Ono, Sno, Bno)

Приклад виконання транзакцій

Варіант А	*Варіант Б*

`read (Sno = x, salary)`
`salary = salary * 1.1`
`write (Sno = x, new_salary)`
	`delete (Sno = x)`
	`for all Property_for_Rent, pno`
	`begin`
	`  read (Pno = pno, sno)`
	`  if (sno = x) then`
	`  begin`
	`  sno = new_sno`
	`  write (Pno = pno, sno)`
	`  end`
	`end`

© М.В.Добролюбова 3

Підтримка транзакцій

Вимоги до транзакцій

1. Будь-яка транзакція завжди повинна переводити базу даних з одного узгодженого стану в інший, хоча допускається, що узгодженість стану бази буде порушуватися під час виконання транзакції
2. Будь-яка транзакція завершується одним з двох можливих способів: COMMIT (фіксація) або ROLLBACK (відкат)
3. Жодна СУБД не володіє внутрішньою можливістю встановити, які саме зміни повинні бути сприйняті як єдине ціле, що утворить одну логічну транзакцію

Обмежувальні оператори

- BEGIN TRANSACTION
- COMMIT
- ROLLBACK

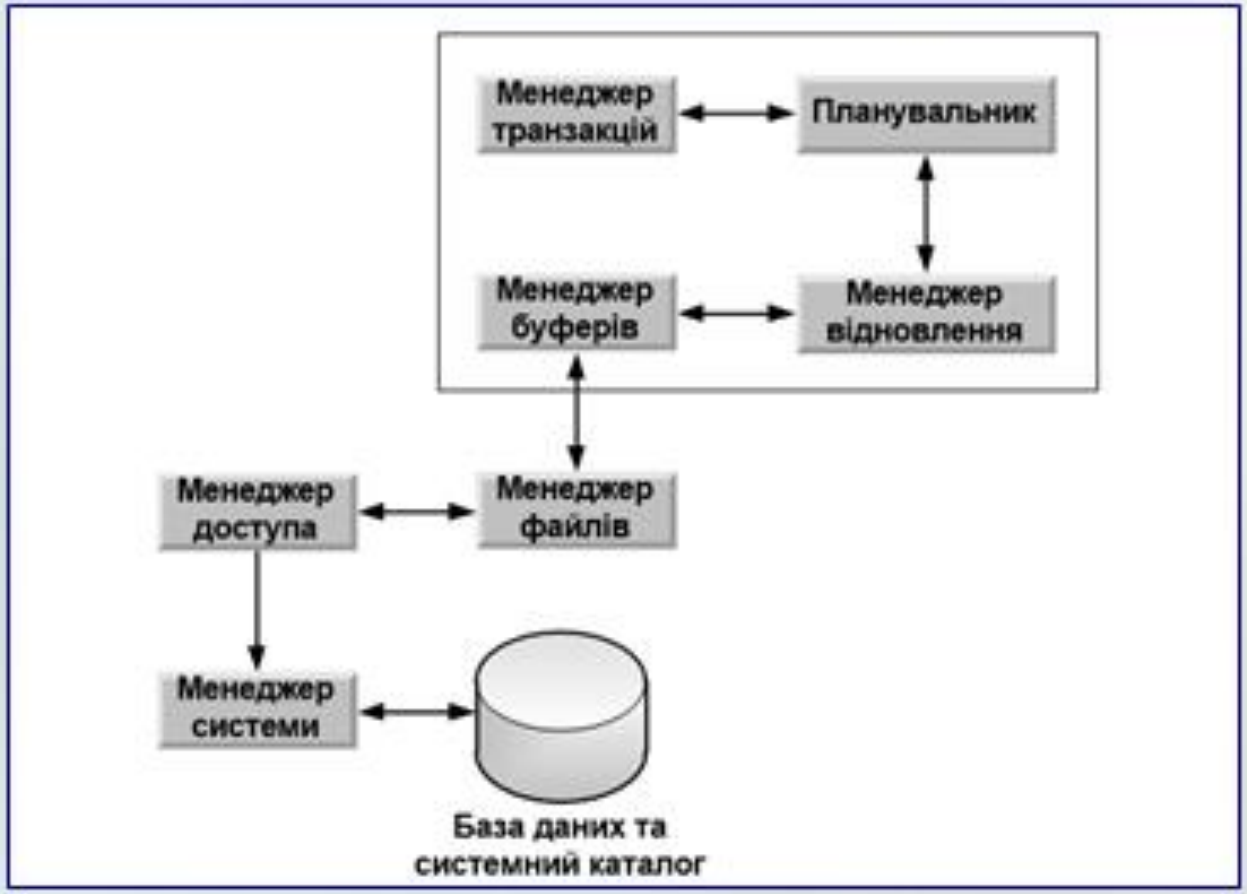
Основні властивості транзакцій

1. Атомарність
2. Узгодженість
3. Ізольованість
4. Тривалість

@ М.В.Добролюбова 4

Підтримка транзакцій

Підсистема обробки транзакцій типовою СУБД



© М.В.Добролюбова

5

Управління паралельністю

Управління паралельністю – це процес організації одночасного виконання в базі даних різних операцій, що гарантує виключення їх взаємного впливу один на одного.

Потенційні проблеми при паралельному виконанні транзакцій:

- проблема втраченого оновлення
- проблема залежності від нефіксованих результатів
- проблема неузгодженої обробки

Проблема втраченого оновлення

Результати цілком успішно завершеної операції оновлення однієї транзакції можуть бути перекриті результатами виконання іншої транзакції.

Час	*Транзакція T_1*	*Транзакція T_2*	*Поле bal_x*
t_1		begin_transaction	100
t_2	begin_transaction	read(bal_x)	100
t_3	read(bal_x)	$bal_x = bal_x + 100$	100
t_4	$bal_x = bal_x - 10$	write(bal_x)	200
t_5	write(bal_x)	commit	90
t_6	commit		90

© М.В.Добролюбова 6

Управління паралельністю

Проблема залежності від нефіксованих результатів

Одна з транзакцій отримує доступ до проміжних результатів виконання іншої транзакції до того, як вони будуть зафіксовані в базі даних.

Час	Транзакція T_3	Транзакція T_4	Поле bal_x
t_1		begin_transaction	100
t_2		read(bal_x)	100
t_3		$bal_x = bal_x + 100$	100
t_4	begin_transaction	write(bal_x)	200
t_5	read(bal_x)		200
t_6	$bal_x = bal_x - 10$	rollback	100
t_7	write(bal_x)		190
t_8	commit		190

Проблема неузгодженої обробки

Транзакціям, які лише зчитують інформацію, доступні для читання проміжні результати одночасно виконуємих і ще не завершених транзакцій, які оновлюють інформацію в базі даних.

Час	Транзакція T_5	Транзакція T_6	Поле bal_x	Поле bal_y	Поле bal_z	Поле sum
t_1		begin_transaction	100	50	25	
t_2		read(bal_x)	100	50	25	0
t_3		$bal_x = bal_x + 100$	100	50	25	0
t_4	begin_transaction	write(bal_x)	100	50	25	100
t_5	read(bal_x)		90	50	25	100
t_6	$bal_x = bal_x - 10$	rollback	90	50	25	150
t_7	write(bal_x)		90	50	25	150
t_8	commit		90	50	35	150
t_9			90	50	35	150
t_{10}			90	50	35	185
t_{11}			90	50	35	185

@ М.В.Добролюбова 7

Впорядкованість і відновлення

Впорядкованість – засіб, здатний допомогти у виявленні тих транзакцій, які гарантовано не викличуть порушення узгодженості даних при одночасному виконанні.

Графік – послідовність запуску операцій багатьох паралельно виконуваних транзакцій, що зберігає черговість виконання операцій в кожній окремій транзакції.

Послідовний графік – графік, в якому операції кожної з транзакцій виконуються строго послідовно і не можуть чергуватися з операціями, що виконуються в інших транзакціях.

Непослідовний графік – графік, в якому чергуються операції з деякого набору одночасно виконуваних транзакцій.

Порядок виконання операцій читання і запису даних задля досягнення впорядкованості

- якщо дві транзакції тільки зчитують деякий елемент даних, вони не будуть конфліктувати між собою і порядок їх виконання не має значення
- якщо дві транзакції зчитують або записують абсолютно незалежні елементи даних, вони не будуть конфліктувати між собою і порядок їх виконання не має значення
- якщо одна транзакція записує елемент даних, а інша транзакція цей самий елемент даних зчитує або записує, порядок їх виконання має істотне значення

© М.В.Добролюбова 8

Впорядкованість і відновлення

Типи впорядкування

1. Конфліктне впорядкування

Конфліктно впорядкований графік – це графік, в якому порядок виконання будь-яких конфліктуючих операцій відповідає їх розміщенню в послідовному графіку.

Складові графу передування:

- вершини, що відповідають кожній з транзакцій
- спрямовані ребра $Ti \rightarrow Tj$, де транзакція Tj зчитує значення елемента, записаного транзакцією Ti
- спрямовані ребра $Ti \rightarrow Tj$, де транзакція Tj записує значення в елемент даних після того, як він був зчитаний транзакцією Ti

2. Впорядкування за переглядом

Впорядкований за переглядом графік – це графік, еквівалентний за переглядом деякому послідовному графіку.

Два графіка *S1* та *S2*, що складаються з одних й тих самих операцій, які входять до складу *n* транзакцій *T1, T2, ... , Tn* є еквівалентними за переглядом, якщо виконуються наступні умови:

1. *Для кожного елемента даних x:* якщо транзакція Ti прочитала вихідне значення x в графіку *S1*, ця ж транзакція Ti повинна прочитати вихідне значення x і в графіку *S2*.
2. *Для кожної операції читання елемента даних x транзакцією Ti в графіку S1:* якщо зчитане значення елемента x було записано транзакцією Tj, то і в графіку *S2* транзакція Ti повинна зчитувати значення елемента x, записане транзакцією Tj.
3. *Для кожного елемента даних x:* якщо в графіку *S1* остання операція запису значення x була виконана транзакцією Tj, ця ж сама транзакція повинна виконувати останній запис значення елемента даних x і в графіку *S2*.

@ М.В.Добролюбова 9

Впорядкованість і відновлення

Типи впорядкування

1. Конфліктне впорядкування

Конфліктно впорядкований графік – це графік, в якому порядок виконання будь-яких конфліктуючих операцій відповідає їх розміщенню в послідовному графіку.

Складові графу передування:

- вершини, що відповідають кожній з транзакцій
- спрямовані ребра $Ti \rightarrow Tj$, де транзакція Tj зчитує значення елемента, записаного транзакцією Ti
- спрямовані ребра $Ti \rightarrow Tj$, де транзакція Tj записує значення в елемент даних після того, як він був зчитаний транзакцією Ti

2. Впорядкування за переглядом

Впорядкований за переглядом графік – це графік, еквівалентний за переглядом деякому послідовному графіку.

Два графіка *S1* та *S2*, що складаються з одних й тих самих операцій, які входять до складу *n* транзакцій *T1, T2, ... , Ti* є еквівалентними за переглядом, якщо виконуються наступні умови:

1. *Для кожного елемента даних x:* якщо транзакція *Ti* прочитала вихідне значення *x* в графіку *S1*, ця ж транзакція *Ti* повинна прочитати вихідне значення *x* і в графіку *S2*.
2. *Для кожної операції читання елемента даних x транзакцією Ti в графіку S1:* якщо зчитане значення елемента *x* було записано транзакцією *Tj*, то і в графіку *S2* транзакція *Ti* повинна зчитувати значення елемента *x*, записане транзакцією *Tj*.
3. *Для кожного елемента даних x:* якщо в графіку *S1* остання операція запису значення *x* була виконана транзакцією *Tj*, ця ж сама транзакція повинна виконувати останній запис значення елемента даних *x* і в графіку *S2*.

@ М.В.Добролюбова 9

Впорядкованість і відновлення

Методи управління паралельністю

Консервативні (песимістичні) підходи

1. Метод блокування (базовий протокол)

Блокування – процедура, яка використовується для управління паралельним доступом до даних і дозволяє відхилити спроби отримання доступу до даних з боку інших транзакцій у разі, коли деяка транзакція вже отримала до них доступ.

Фундаментальний принцип методів блокування: транзакція повинна вимагати виконати блокування для читання або для запису деякого елемента даних перед тим, як вона зможе виконати в базі даних відповідну операцію читання або запису.

Основні правила методу блокування:

1. Якщо транзакція встановила блокування елемента даних для читання, вона зможе зчитати його, але не зможе оновити.
2. Якщо транзакція встановила блокування елемента даних для запису, вона може як зчитувати, так і оновлювати цей елемент.
3. Будь-яка транзакція, якій необхідно отримати доступ до елементу даних, повинна спочатку виконати блокування цього об'єкта.
4. Якщо елемент ще не заблокований будь-якою іншою транзакцією, блокування елемента буде виконано успішно.
5. Якщо елемент даних вже заблокований, СУБД проаналізує, чи є тип отриманого запиту сумісним з типом вже існуючого блоку. Якщо робиться доступ на зчитування до елементу, який заблокований для зчитування, доступ до нього буде дозволений. В іншому випадку транзакція буде переведена в стан очікування, який триватиме доти, доки існуючий блок не буде знятий.
6. Транзакція продовжує утримувати блокування елемента даних доти, доки вона явно не звільнить його – або в ході виконання транзакції, або по її закінченні. Тільки після того як з елемента даних буде зняте блокування для запису, інші транзакції зможуть "побачити" результати проведеної операції запису.

@ М.В.Добролюбова 11

Впорядкованість і відновлення

Методи управління паралельністю

Консервативні (песимістичні) підходи

2. Протокол двофазного блокування

Протокол двофазного блокування (2PL) – це протокол, який визначає моменти встановлення та зняття блокування для кожної з транзакцій і використовується для забезпечення впорядкованості.

Фундаментальний принцип двофазного блокування: транзакція виконується по протоколу двофазного блокування, якщо в ній всі операції блокування передують першій операції розблокування. При цьому кожна транзакція може бути розділена на дві фази: *фазу нарощання*, в якій виконуються всі необхідні блокування і не звільняється жодного з елементів даних; і *фазу стиснення*, в якій звільняються всі виконані раніше блокування і не може бути затребувано жодного нового.

Проблеми двофазного блокування:

1. **Каскадний відкат.** Ситуація, в якій скасування єдиної транзакції призводить до цілої серії відкатів транзакцій, які від неї залежать. Варіант уникнути такої ситуації – доповнення 2PL вимогою відкладати виконання скасування всіх встановлених блокувань до кінця транзакції.
2. **Взаємне блокування.** Ситуація, в якій при очікуванні двома транзакціями звільнення елементів, заблокованих іншою транзакцією з цієї ж пари. Виявлення взаємних блокувань проводиться за допомогою *графу очікувань*, що відображає залежність транзакцій одна від одної: транзакція *Ti* залежить від транзакції *Tj* в тому випадку, якщо транзакція *Tj* заблокувала елемент даних, необхідний для продовження роботи транзакції *Ti*.

Побудова графа очікувань:

- для кожної виконуваної транзакції створюється окрема вершина
- окреме спрямоване ребро *Ti* → *Tj* створюється для кожного випадку, коли транзакція *Ti* чекає звільнення елемента даних, заблокованого транзакцією *Tj*.

Взаємне блокування має місце в тому і тільки в тому випадку, якщо граф очікувань містить петлю.

@ М.В.Добролюбова 12

Впорядкованість і відновлення

Методи управління паралельністю

Консервативні (песимістичні) підходи

2. Протокол двофазного блокування

Протокол двофазного блокування (2PL) – це протокол, який визначає моменти встановлення та зняття блокування для кожної з транзакцій і використовується для забезпечення впорядкованості.

Фундаментальний принцип двофазного блокування: транзакція виконується по протоколу двофазного блокування, якщо в ній всі операції блокування передують першій операції розблокування. При цьому кожна транзакція може бути розділена на дві фази: *фазу наростання*, в якій виконуються всі необхідні блокування і не звільняється жодного з елементів даних; і *фазу стиснення*, в якій звільняються всі виконані раніше блокування і не може бути затребувано жодного нового.

Проблеми двофазного блокування:

1. **Каскадний відкат.** Ситуація, в якій скасування єдиної транзакції призводить до цілої серії відкатів транзакцій, які від неї залежать. Варіант уникнути такої ситуації – доповнення 2PL вимогою відкладати виконання скасування всіх встановлених блокувань до кінця транзакції.
2. **Взаємне блокування.** Ситуація, в якій при очікуванні двома транзакціями звільнення елементів, заблокованих іншою транзакцією з цієї ж пари. Виявлення взаємних блокувань проводиться за допомогою *графу очікувань*, що відображає залежність транзакцій одна від одної: транзакція *Ti* залежить від транзакції *Tj* в тому випадку, якщо транзакція *Tj* заблокувала елемент даних, необхідний для продовження роботи транзакції *Ti*.

Побудова графа очікувань:

- для кожної виконуваної транзакції створюється окрема вершина
- окреме спрямоване ребро *Ti* → *Tj* створюється для кожного випадку, коли транзакція *Ti* чекає звільнення елемента даних, заблокованого транзакцією *Tj*.

Взаємне блокування має місце в тому і тільки в тому випадку, якщо граф очікувань містить петлю.

@ М.В.Добролюбова 12

Впорядкованість і відновлення

Методи управління паралельністю

Консервативні (песимістичні) підходи

4. Часові мітки (модернізований базовий протокол)

Правило запису Томаса – це модернізований базовий протокол впорядкування за часовими мітками, який дозволяє досягти менш жорсткої конфліктної впорядкованості за рахунок скасування застарілих операцій запису.

Правило виконання транзакцією T операції запису

- Транзакція *T* запитує запис елементу даних (*x*), значення якого вже було зчитане більш новою транзакцією:

$$ts(T) < read\_timestamp(x)$$

Транзакція *T* відкочується та перезапускається з новою часовою міткою.

- Транзакція запитує запис елемента даних (*x*), значення якого вже було перезаписане більш новою транзакцією:

$$ts(T) < write\_timestamp(x)$$

Це означає, що новіша транзакція вже перезаписала значення цього елемента даних і значення, яке старіша транзакція збирається записати в даний елемент, було обчислено на основі застарілого вихідного значення даного елемента. В подібному випадку операція запису цілком безпечно може бути проігнорована.

- В іншому випадку операція запису може бути виконана, а часовій мітці запису оновлюваного елемента даних має бути присвоєно нове значення:

$$write\_timestamp(x) = ts(T)$$

Правило запису Томаса дозволяє генерувати графіки, які не можливо створити при використанні будь-яких інших протоколів управління паралельністю.

Впорядкованість і відновлення

Методи управління паралельністю

Оптимістичні підходи

Фази оптимістичного протоколу управління

1. Фаза читання

Охоплює транзакцію від її початку до моменту безпосереднього виконання фіксації результатів. Транзакція зчитує значення всіх необхідних їй елементів даних і поміщає їх в локальні змінні. Будь-які оновлення застосовуються тільки до локальної копії даних, а не до інформації, що зберігається в самій базі даних.

2. Фаза перевірки

Слідує за фазою читання. Виконуються перевірки, необхідні для отримання гарантій відсутності порушення впорядкованості в разі перенесення в базу даних змін, виконаних транзакцією.

Для транзакцій, що включають тільки операції читання, перевірка полягає в підтвердженні того, що використані транзакцією значення, як і раніше, залишаються поточними значеннями відповідних елементів даних. Якщо порушень не виявлено, транзакція завершує своє виконання. Якщо знайдені змінені значення, транзакція скасовується і перезапускається. Якщо в транзакції виконувалося оновлення даних, то перевірка включає виконання контролю – чи збережеться база даних в узгодженому стані після внесення в неї результатів даної транзакції, а також чи не буде порушено умови впорядкованості. У разі виявлення помилки транзакція скасовується і перезапускається.

3. Фаза запису

Виконується після успішного завершення фази перевірки, але тільки щодо транзакцій, які включають операції оновлення. В цій фазі всі зміни, внесені в локальні копії даних, переносяться в базу даних.

@ М.В.Добролюбова 15

Впорядкованість і відновлення

Відновлення бази даних – процес повернення бази даних в коректний стан, втрачений в результаті збою або відмови.

Причини, які викликають відмови

- аварійне припинення роботи системи
- відмова носіїв інформації
- помилки прикладних програм
- стихійні лиха
- недбале або легковажне поводження

Функції відновлення типової СУБД

1. Механізм резервного копіювання
2. Засоби ведення журналу

Інформація, що вноситься до файлу журналу:

1) записи про транзакції (ідентифікатор транзакції; початок транзакції, операції вставки, оновлення або видалення, скасування або фіксація транзакції; ідентифікатор елемента даних, залученого до операції обробки бази даних; копія (значення) елемента даних до та після операції (зміни); службова інформація файлу журналу)

2) записи контрольних точок

3. Функція створення контрольних точок

Контрольна точка – момент синхронізації між базою даних і журналом реєстрації транзакцій.

4. Менеджер відновлення

Методи відновлення:

- метод відновлення з використанням відкладеного поновлення
- метод відновлення з використанням негайного поновлення
- метод тіньових сторінок

@ М.В.Добролюбова 16

Покращені моделі транзакцій

Модель вкладених транзакцій

Модель вкладених транзакцій (Еліот Мосс, 1981 р.) – це модель, яка розглядає транзакцію, як набір пов'язаних підзадач (субтранзакції), кожна з яких також може складатися з довільної кількості субтранзакцій.

Основні переваги моделі вкладених транзакцій:

- модульність
- додатковий рівень деталізації в механізмах управління паралельною і відновленням
- паралельність обробки в межах транзакції
- можливість управління з відновленням в межах транзакції

Допускає довільний рівень вкладення.

Метод хронік

Хроніка (Гарсія-Моліна та Салем, 1987 р.) – це послідовність (пласких) транзакцій, які можуть чергуватися з іншими транзакціями. При цьому СУБД гарантує, що або всі транзакції, що входять в хроніку, будуть успішно завершені, або будуть запущені компенсуючі транзакції, необхідні для усунення досягнутих часткових результатів.

Дозволяє наявність єдиного рівня вкладення.

Модель багаторівневих транзакцій

Модель багаторівневих транзакцій (модель закритих вкладених транзакцій) – це модель, яка вимагає, щоб завершення роботи субтранзакцій виконувалося від низу до верху, в напрямку транзакції верхнього рівня. Тут властивість атомарності в транзакціях зберігається до самого верхнього рівня.

Модель відкритих вкладених транзакцій – це модель, в якій атомарность порушується і часткові результати виконання субтранзакцій можуть бути доступні поза транзакцією.

17

Покращені моделі транзакцій

Динамічна реструктуризація

Динамічна реструктуризація (П'ютл, 1988 р.) – це модель, яка об'єднує операції розбиття і об'єднання транзакцій.

Принцип, покладений в основу операції розбиття транзакції: поділ активної транзакції на дві впорядковані транзакції і розподіл між ними виконуваних дій і використовуваних ресурсів.

Підхід дозволяє зробити проміжні результати транзакції доступними іншим транзакцій з повним збереженням їх семантики.

Операція поділу транзакції може застосовуватися тільки в тому випадку, якщо можливо створити дві транзакції, що будуть впорядковані одна з одною і з усіма іншими транзакціями, які виконуються в даний момент.

Моделі робочих потоків

Робочий потік – це вид діяльності, який передбачає координоване виконання множини завдань, що здійснюються різними суб'єктами обробки.

© М.В.Добролюбова 18

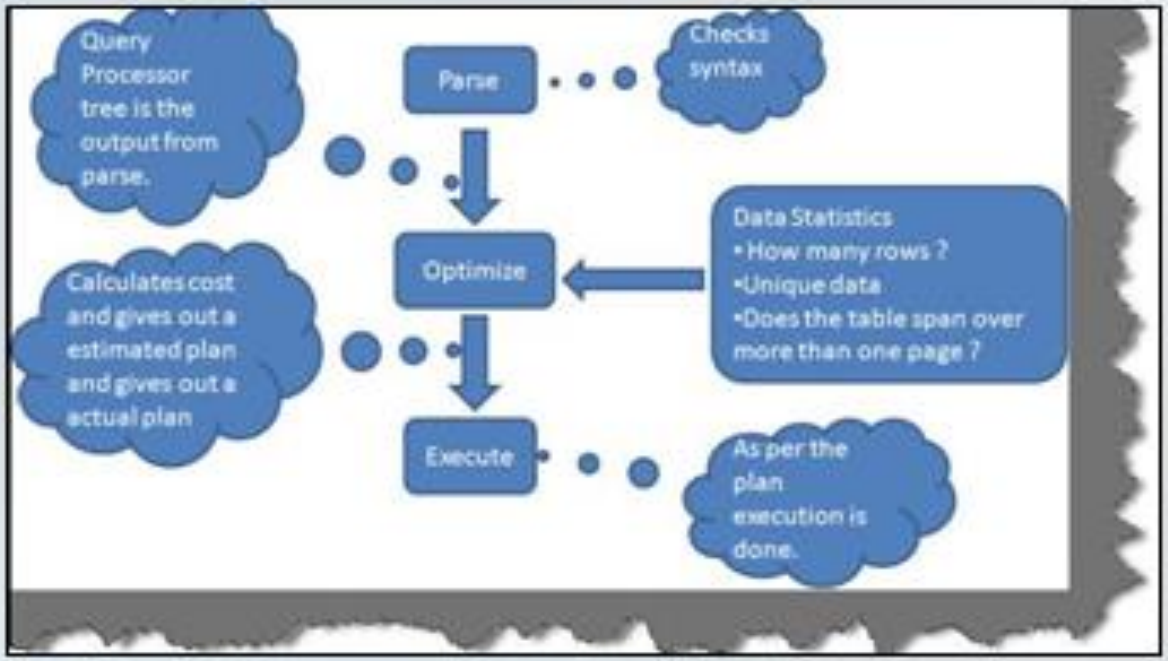

ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 1. ЗАХИСТ БД, КЕРУВАННЯ ТРАНЗАКЦІЯМИ, ОБРОБКА ЗАПИТІВ
Лекція 11
«Обробка запитів»
© М.В.Добролюбова

Загальний огляд методів обробки запитів

Обробка запитів – це дії, які необхідні для отримання потрібної інформації з бази даних шляхом перетворення запиту, записаного мовою високого рівня, в коректну та ефективну послідовність операцій (план запиту), записану мовою низького рівня, що реалізує операції реляційної алгебри.

План виконання запиту – це конкретна та ефективна послідовність операцій.

Оптимізація запиту – це процедура вибору найбільш ефективного плану виконання запиту.



© М.В.Добролюбова 2

Загальний огляд методів обробки запитів

Ілюстрація результатів застосування різних стратегій оптимізації запитів з точки зору використання системних ресурсів

ПРИКЛАД

Визначити імена всіх менеджерів, які працюють в Лондонських відділеннях компанії DreamHome

Припустимо, що таблиця Staff містить 1000 кортежів, а таблиця Branch – 50.

```
SELECT *
FROM staff s, branch b
WHERE s.bno = b.bno AND
    (s.position = 'Manager' AND b.city = 'London');
```

Варіанти запису запиту за допомогою виразів реляційної алгебри та загальна вартість їх обробки:

1. $\sigma_{(position = 'Manager') \wedge (city = 'London') \wedge (staff.bno = branch.bno)}(Staff \times Branch)$

   *(1000 + 50) + 2 * (1000 * 50) = 101 050 дискових операцій*

2. $\sigma_{(position = 'Manager') \wedge (city = 'London')}(Staff \bowtie_{staff.bno = branch.bno} Branch)$

   *2 * 1000 + (1000 + 50) = 3 050 дискових операцій*

3. $(\sigma_{position = 'Manager'}(Staff)) \bowtie_{staff.bno = branch.bno} (\sigma_{city = 'London'}(Branch))$

   *1000 + 2 * 50 + 5 + (50 + 5) = 1 160 дискових операцій*

@ М.В.Добролюбова 3

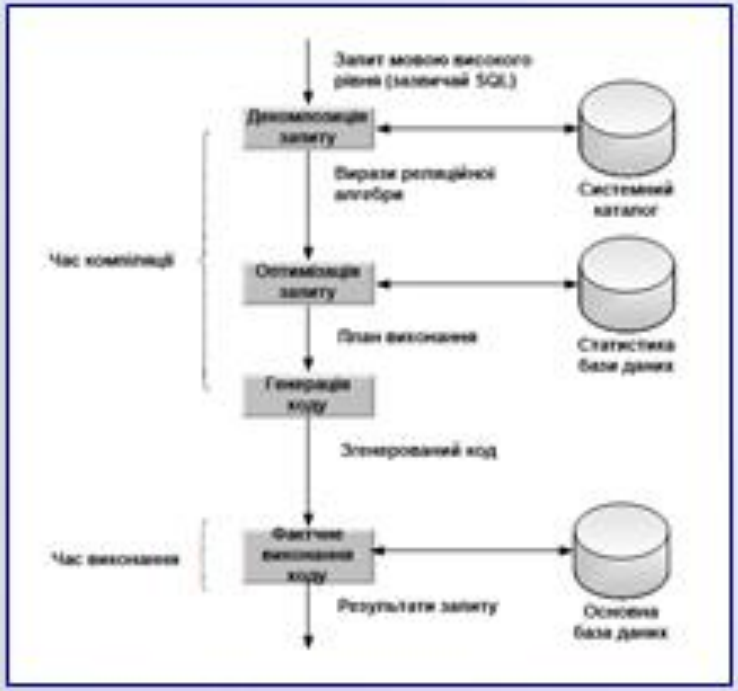
Загальний огляд методів обробки запитів

Основні етапи процесу обробки запитів:

- декомпозиція
- оптимізація
- генерацію коду
- виконання

Динамічна оптимізація запитів – це виконання декомпозиції та оптимізації при кожному виклику запиту.

Статична оптимізація запитів – це одноразове виконання етапів сканування, перевірки та оптимізації.



@ М.В.Добролюбова

4

Декомпозиція запитів

Декомпозиція – це перетворенні запиту, який написаний мовою високого рівня у вираз реляційної алгебри з наступною перевіркою його синтаксичної семантичної коректності.

Стадії декомпозиції:

- аналіз
- нормалізація
- семантичний аналіз
- спрощення
- реструктуризація запиту

АНАЛІЗ

ПРИКЛАД

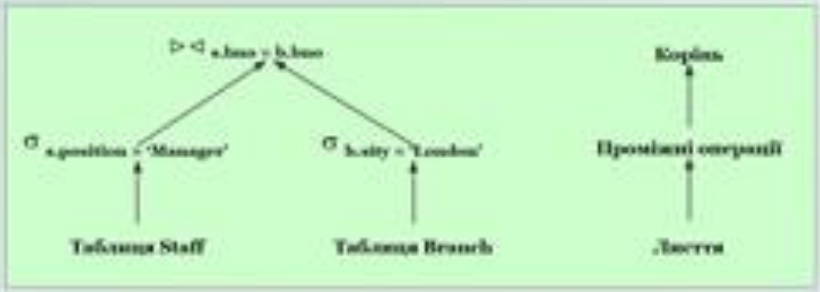
```
SELECT staff_no
FROM staff
WHERE position > 10;
```

Причини відхилення виконання запиту:

- В списку вибірки запиту вказаний атрибут *staff_no*, відсутній у визначенні відношення *Staff* (його ім'я *sno*)
- В інструкції *WHERE* операція порівняння «>10» несумісна з типом атрибуту *position*, який записаний у відношенні як рядок символів

Конструювання дерева запиту (дерева реляційної алгебри):

1. Листкові вузли створюються для кожного базового відношення, що використовується в запиті.
2. Нелистковi вузли створюються для кожного проміжного відношення, що створюється в результаті виконання деякої операції реляційної алгебри.
3. Корінь дерева – це результуючий набір даних.
4. Операції виконуються у напрямку від листкових вузлів до кореня дерева.



@ М.В.Добролюбова 5

Декомпозиція запитів

НОРМАЛІЗАЦІЯ

Кон'юнктивна нормальна форма (AND,$\wedge$) – це послідовність кон'юнкцій, пов'язаних між собою операторами диз'юнкції. Кожна кон'юнкція містить один або більше термів, з'єднаних операторами кон'юнкції.

$$(\text{position} = \text{'Manager'} \vee \text{salary} > 20000) \wedge \text{bno} = \text{'B3'}$$

Диз'юнктивна нормальна форма (OR,$\vee$) – це послідовність диз'юнкцій, з'єднаних між собою операторами кон'юнкції. Кожна диз'юнкція містить один або більше термів, з'єднаних операторами кон'юнкції.

$$(\text{position} = \text{'Manager'} \wedge \text{bno} = \text{'B3'}) \vee (\text{salary} > 20000 \wedge \text{bno} = \text{'B3'})$$

@ М.В.Добролюбова 6

Декомпозиція запитів

СЕМАНТИЧНИЙ АНАЛІЗ

Некоректно сформульований запит – це запит, виконання якого ніколи не зможе призвести до створення результуючого набору даних.

Суперечливий запит – це запит, предикат якого ніколи не зможе бути задоволений будь-яким кортежем.

$$(\text{position} = \text{'Manager'} \wedge \text{position} = \text{'Assistant'}) \vee \text{salary} > 20000$$

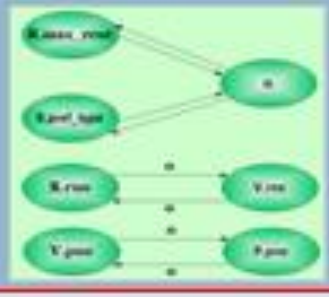
Способи визначення коректності:

1. Побудова *графу з'єднань відношень*. Якщо елементи графу виявляються роз'єднаними, формулювання запиту є не правильним.
2. Побудова *графу з'єднань нормалізованих атрибутів*. Якщо граф включає петлю, для якої сума оцінок від'ємна, то запит містить суперечливі умови.

ПРИКЛАД *Перевірка семантичної коректності запиту*

```
SELECT p.pno, p.street
FROM renter r, viewing v, property_for_rent p
WHERE r.rno = v.rno AND
      r.max_rent >= 500 AND r.pref_type = 'Flat' AND p.ono = 'CO93';
```

```
SELECT p.pno, p.street
FROM renter r, viewing v, property_for_rent p
WHERE r.max_rent >= 500 AND r.rno = v.rno AND v.pno = p.pno AND
      r.pref_type = 'Flat' AND r.max_rent < 200;
```



© М.В.Добролюбова 7

Декомпозиція запитів

СПРОЩЕННЯ

Мета виконання процедури спрощення – виявлення збиткових кваліфікаторів, виключення загальних підвиразів та перетворення запиту в семантично еквівалентну форму.

РЕСТРУКТУРИЗАЦІЯ

Мета виконання реструктуризації запитів – перетворення запиту таким чином, щоб забезпечити найбільш ефективне його виконання.

@ М.В.Добролюбова

8

Методи оптимізації запитів

Евристичний підхід до оптимізації запитів – це метод оптимізації запитів, яким передбачене використання правил трансформації для перетворення виразів реляційної алгебри в еквівалентну форму, обробка якої буде більш ефективною.

Підхід до оптимізації запитів на основі оцінки вартості операцій реляційної алгебри – це метод порівняння показників різних стратегій, який виконується на основі їх відносної вартості, причому вибирається той з варіантів, який дозволяє мінімізувати використання системних ресурсів.

Конвеєрна обробка – це метод для підвищення ефективності запитів, при якому передача результатів однієї операції на обробку іншої операції здійснюється без створення тимчасових відношень, призначених для зберігання проміжних результатів.

ЕВРИСТИЧНИЙ ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ

Правила перетворення операцій реляційної алгебри

1. Операція вибірки з кон'юктивним предикатом може бути перетворена в послідовність операцій вибірки по членам кон'юнкції (і навпаки).
2. Правило комутативності операції вибірки.
3. В послідовності операцій проекції необхідна лише остання з цих операцій.
4. Правило комутативності операцій вибірки і проекції.
5. Правило комутативності операції тета-з'єднання (і декартового добутку).
6. Правило комутативності операцій вибірки та тета-з'єднання (або декартового добутку).
7. Правило комутативності операцій проекції та тета-з'єднання (або декартового добутку).
8. Правило комутативності операцій об'єднання та перетину (але не різниці множин).
9. Правило комутативності операції вибірки і операцій над множинами (об'єднання, перетин та різниця множин).
10. Правило комутативності операцій проекції та об'єднання.
11. Правило асоціативності операції тета-з'єднання (і декартового добутку).
12. Правило асоціативності операцій об'єднання і перетину (але не операції різниці множин).

@ М.В.Добролюбова 9

Методи оптимізації запитів

ЕВРИСТИЧНИЙ ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ

Стратегії евристичної обробки запитів

1. Виконання операції вибірки на ранніх етапах.
2. Об'єднання в одну операцію з'єднання операції декартового добутку і операції вибірки, предикат якої є умовою з'єднання.
3. Використання властивості асоціативності бінарних операцій для перевпорядкування листкових вузлів таким чином, щоб листкові вузли з найбільш обмежуючими операціями вибірки виконувалися б в першу чергу.
4. Якомога скоріше виконання операцій проекції .
5. Одноразове обчислення загальних виразів.

Методи оптимізації запитів

ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ НА ОСНОВІ ОЦІНКИ ВАРТОСТІ ОПЕРАЦІЙ РЕЛЯЦІЙНОЇ АЛГЕБРИ

Статистичні дані, що зберігаються типовою СУБД в системному каталозі:

- кількість кортежів у відношенні (тобто його кардинальність) – $ntuples(R)$
- показник блокування відношення (тобто кількість кортежів відношення, які розміщені в одному блоці) – $ntuples(R)$
- кількість блоків вторинної пам'яті, необхідних для збереження відношення – $nblocks(R)$
- кількість унікальних значень атрибута у відношенні – $ndistinct_A(R)$
- мінімальна та максимальна кількість можливих значень атрибута у відношенні – $min_A(R), max_A(R)$
- кардинальність вибірки атрибута у відношенні – $SC_A(R)$
- кількість рівнів індексів – $nlevels_A(I)$
- кількість листкових блоків індексу – $nlfblocks_A(I)$

Методи оптимізації запитів

ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ НА ОСНОВІ ОЦІНКИ ВАРТОСТІ ОПЕРАЦІЙ РЕЛЯЦІЙНОЇ АЛГЕБРИ

Основні типи стратегій операції вибірки:

- лінійний пошук (файл не відсортований, індекс відсутній)
- двійковий пошук (впорядкований файл, індекс відсутній)
- пошук по рівності ключа перемішування
- пошук по рівності первинного ключа
- пошук по нерівності первинного ключа
- пошук по рівності значення кластерного (вторинного) індексу
- пошук по рівності значення некластерного (вторинного) індексу
- пошук по нерівності значення вторинного індексу типу B⁺-Tree

Зведені дані по оцінці вартості операцій вводу/виводу для різних стратегій виконання операцій вибірки

Стратегія	Оцінки вартості
лінійний пошук (файл не відсортований, індекс відсутній)	[nblocks(R)/2] – у випадку пошуку за рівністю ключового атрибуту; nblocks(R) – в протилежному випадку
двійковий пошук (впорядкований файл, індекс відсутній)	$[\log_2(nblocks(R))]$ – у випадку пошуку за рівністю ключового атрибуту; $[\log_2(nblocks(R))]+[SC_A(R)/bfactor(R)]-1$ – в протилежному випадку
пошук по рівності ключа перемішування	1 – за відсутності перемішування
пошук по рівності первинного ключа	$nlevels_A(I)+1$
пошук по нерівності первинного ключа	$nlevels_A(I)+[nblocks(R)/2]$
пошук по рівності значення кластерного (вторинного) індексу	$nlevels_A(I)+[SC_A(R)/bfactor(R)]$
пошук по рівності значення некластерного (вторинного) індексу	$nlevels_A(I)+[SC_A(R)]$
пошук по нерівності значення вторинного індексу типу B⁺-Tree	$nlevels_A(I)+[nlfblocks_A(I)/2+ntuples(R)/2]$

@ М.В.Добролюбова 12

Методи оптимізації запитів

ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ НА ОСНОВІ ОЦІНКИ ВАРТОСТІ ОПЕРАЦІЙ РЕЛЯЦІЙНОЇ АЛГЕБРИ

Основні типи стратегій операцій з'єднання:

- з'єднання блоками з використанням вкладених циклів
- індексоване з'єднання блоками з використанням вкладених циклів
- з'єднання через сортування-злиття
- пошук по рівності первинного ключа
- з'єднання через перемішування

Зведені дані по оцінці вартості операцій вводу/виводу для різних стратегій виконання операцій з'єднання

Стратегія	Вартість	Умова
з'єднання блоками з використанням вкладених циклів	nblocks(R)+(nblocks(R)*nblocks(S)) nblocks(R)+(⌈nblocks(R)/(nbuffer-2)⌉*nblocks(S)) nblocks(R)+nblocks(S)	буфер містить лише один блок для кожної множини відношень R та S буфер містить (nbuffer-2) блоків для R у випадку, якщо всі блоки R можуть бути прочитані в буфер
індексоване з'єднання блоками з використанням вкладених циклів	залежить від методу індексування, наприклад: nblocks(R)+ ntuples(R)* (nlevels_A(I)+1) nblocks(R)+ ntuples(R)* (nlevels_A(I)+⌈SC_A(R)/bfactor(R)⌉)	у випадку, якщо з'єднуваний атрибут A є у відношенні S первинним ключем якщо індекс I – кластеризований індекс атрибута A
з'єднання через сортування-злиття	nblocks(R)*⌈log_2(nblocks(R))⌉+ nblocks(S)*⌈log_2(nblocks(S))⌉ nblocks(R)+nblocks(S)	у випадку сортування у випадку злиття
з'єднання через перемішування	3(nblocks(R)+nblocks(S))+2*max_partitions 2(nblocks(R)+nblocks(S))* ⌈log_(max_buffer-1)(nblocks(S)-1)⌉+ nblocks(R)+nblocks(S)	у випадку, якщо індекс перемішування міститься в пам'яті в інших випадках

@ М.В.Добролюбова 13

Методи оптимізації запитів

ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ НА ОСНОВІ ОЦІНКИ ВАРТОСТІ ОПЕРАЦІЙ РЕЛЯЦІЙНОЇ АЛГЕБРИ

Послідовність дій для реалізації операції проекції:

- видалення з відношення зайвих атрибутів
- виключення з результуючої таблиці будь-яких продубльованих кортежів, що виникли в результаті виконання попереднього етапу

ПРИКЛАД

Загальна вартість першого етапу виконання операції проекції

$$nblocks(R)+nblocks(R)*(\log_2 nblocks(R))$$

Загальна вартість другого етапу виконання операції проекції

nblocks(R)+nb, де nb – кількість блоків, необхідних для розміщення тимчасової таблиці, яка містить результати виконання операції проекції для відношення R до видалення дублів

Операції реляційної алгебри над множинами:

- об'єднання R∪S
- перетин R∩S
- різниця R–S

Загальна вартість для бінарних операцій над множинами

$$nblocks(R)+nblocks(S)+nblocks(R)*[\log_2(nblocks(R))]+nblocks(S)*[\log_2(nblocks(S))]$$

© М.В.Добролюбова 14

Методи оптимізації запитів

ПІДХІД ДО ОПТИМІЗАЦІЇ ЗАПИТІВ НА ОСНОВІ ОЦІНКИ ВАРТОСТІ ОПЕРАЦІЙ РЕЛЯЦІЙНОЇ АЛГЕБРИ

Узагальнюючі операції:

ПРИКЛАД

```
SELECT AVG(salary)
FROM staff;
```

```
SELECT AVG(salary)
FROM staff
GROUP BY bno;
```

КОНВЕЄРНА ОБРОБКА

Матеріалізація – це підхід, який полягає в тому, що результати однієї операції до початку наступної операції зберігаються у тимчасових відношеннях.

Конвеєрна обробка – це метод для підвищення ефективності запитів, при якому передача результатів однієї операції на обробку іншої операції здійснюється без створення тимчасових відношень, призначених для зберігання проміжних результатів.

@ М.В.Добролюбова 15


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 2. НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ
Лекція 12
«Концепції та розробка розподілених СУБД»
@ М.В.Добролюбова

Основні концепції розробки СУРБД

Розподілена база даних – це набір логічно пов'язаних між собою даних (і їх описів), які фізично розподілені у деякій комп'ютерній мережі.

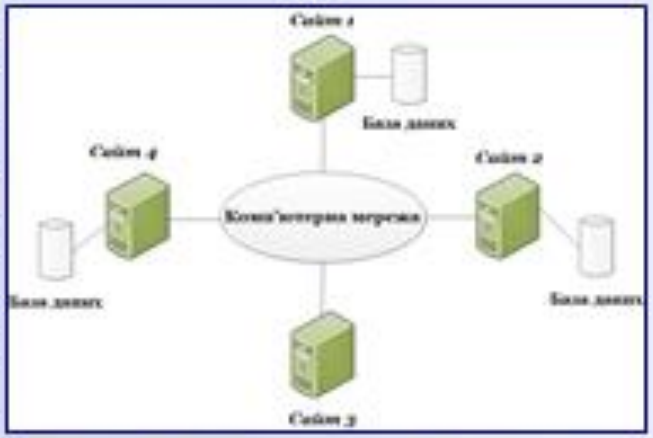
Розподілена СУБД – це програмний комплекс, призначений для управління розподіленими базами даних, що дозволяє зробити розподіл інформації прозорим для кінцевого користувача.

Локальні додатки – це додатки, які не вимагають доступу до даних на інших сайтах.

Глобальні додатки – це додатки, які вимагають доступу до даних на інших сайтах. В розподіленій СУБД повинен існувати хоча б один глобальний додаток.

Реплікація даних – це процес, під яким розуміють копіювання даних з одного джерела в інше та навпаки.

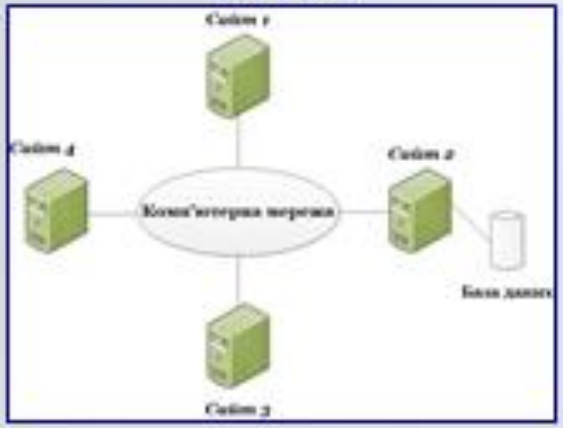
Розподілена обробка даних – це обробка з використанням централізованої бази даних, доступ до якої може здійснюватися з різних комп'ютерів мережі.



Топологія системи управління розподіленою базою даних

Особливості систем управління розподіленими базами даних

- набір логічно пов'язаних поділюваних даних
- збережені дані розбиті на декілька фрагментів
- між фрагментами може бути організована реплікація даних
- фрагменти та їх репліки розподілені по різних сайтах
- сайти пов'язані між собою мережевими з'єднаннями
- робота з даними на кожному сайті управляється СУБД
- СУБД на кожному сайті здатна підтримувати автономну роботу локальних додатків
- СУБД кожного сайту підтримує хоча б один глобальний додаток



Топологія системи з розподіленою обробкою

@ М.В.Добролюбова

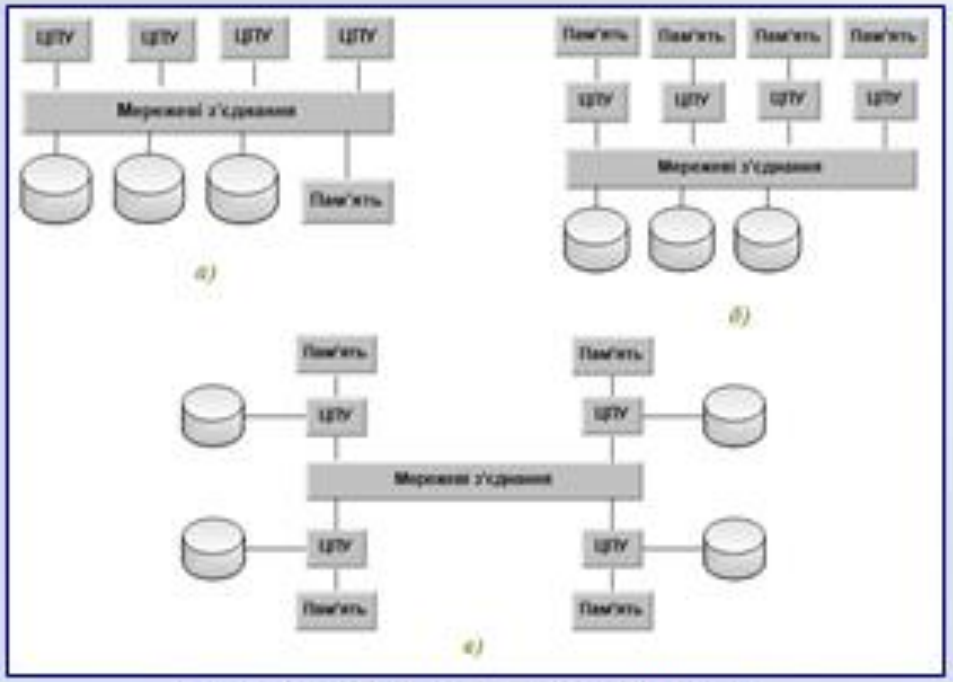
2

Основні концепції розробки СУРБД

Паралельна СУБД – це система управління базою даних, яка функціонує з використанням декількох процесорів та пристроїв жорстких дисків, що дозволяє їй (якщо це можливо) розпаралелювати виконання деяких операцій з метою підвищення загальної продуктивності обробки.

Основні типи архітектури паралельних СУБД

- системи з поділом пам'яті
- системи з поділом дисків
- системи без поділу



Архітектура систем з паралельною обробкою:
а) з поділом пам'яті; б) з поділом диска; в) без поділу

@ М.В.Добролюбова

3

Основні концепції розробки СУРБД

Переваги та недоліки розподілених СУБД

Переваги	Недоліки
Відображення структури організації	Підвищення складності
Розподіляємість та локальна автономність	Збільшення вартості
Підвищення доступності даних	Проблеми захисту
Підвищення надійності	Ускладнення контролю за цілісністю даних
Підвищення продуктивності	Відсутність стандартів
Економічні вигоди	Ускладнення процедури розробки бази даних
Модульність системи	

Гомогенні розподілені СУБД – це системи, в яких всі сайти використовують один і той самий тип СУБД.

Гетерогенні розподілені СУБД – це системи, в яких на сайтах можуть функціонувати різні типи СУБД, що використовують різні моделі даних.

Мультибазова система – це розподілена система управління базами даних, в якій управління кожним з сайтів здійснюється абсолютно автономно, тобто кінцеві користувачі різних сайтів можуть отримувати доступ та спільно використовувати дані без необхідності фізичної інтеграції існуючих баз даних.

Схема експорту – це схема, за допомогою якої системний адміністратор локальної бази даних надає дозвіл зовнішнім користувачам на доступ до певних елементів своєї бази даних.

Нефедеральна мультибазова система – це мультибазова система, що не має локальних користувачів.

Федеральна мультибазова система – це мультибазова система, що виглядає як розподілена система для віддалених користувачів та як централізована система для локальних.

© М.В.Добролюбова 4

Організація та робота комп'ютерних мереж

Комп'ютерна мережа – це сукупність автономних комп'ютерів, з'єднаних між собою і здатних обмінюватися інформацією.

Локальна мережа (local area network, LAN) – це мережа, яка утворюється при з'єднанні комп'ютерів, що належать одному і тому самому сайту.

Глобальна мережа (wide area network, WAN) – це мережа, яка використовується для з'єднання комп'ютерів різних локальних мереж, що знаходяться на віддаленій відстані один від одного.

Одноадресна мережа – це мережа, в якій при необхідності розіслати повідомлення всім її вузлам знадобиться відправити кілька окремих повідомлень.

Широкомовна мережа – це мережа, в якій всі вузли отримують всі повідомлення, проте в кожному з повідомлень присутній префікс, що ідентифікує вузол-одержувач, а всі інші вузли мережі просто проігнорують повідомлення, яке надійшло.

Порівняння характеристик глобальних і локальних мереж

Глобальні мережі	*Локальні мережі*
Відстань вузлів до тисяч кілометрів	Відстань вузлів до декількох кілометрів
Пов'язують автономні комп'ютери	Пов'язують комп'ютери, які спільно використовують розподілені додатки
Мережа управляється незалежною організацією (використовуються телефонні або супутникові канали зв'язку)	Мережа управляється її користувачами (використовуються особисті кабельні з'єднання)
Складні протоколи	Простіші протоколи
Використовується одноадресна маршрутизація	Використовується широкомовна маршрутизація
Використовується нерегулярна топологія	Використовується топологія шини або зірки
Імовірність помилки порядку $1:10^6$	Імовірність помилки порядку $1:10^9$

@ М.В.Добролюбова

5

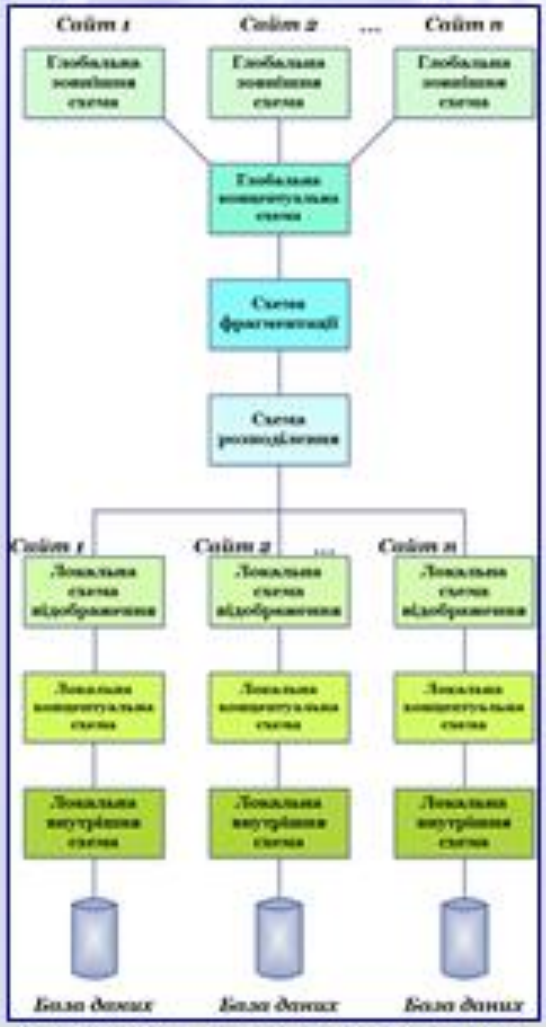
Функції та архітектура розподілених СУБД

Функціональні можливості СУРБД

- розширені служби установки з'єднань
- розширені засоби ведення каталогу
- засоби обробки розподілених запитів
- розширені функції управління паралельністю
- розширені функції відновлення

Елементи рекомендованої архітектури СУРБД

- набір глобальних зовнішніх схем
- глобальна концептуальна схема
- схема фрагментації і схема розподілу
- набір схем для кожної локальної СУБД, що відповідають вимогам трирівневої архітектури ANSI-SPARC



Архітектура, рекомендована для СУРБД

@ М.В.Добролюбова

6

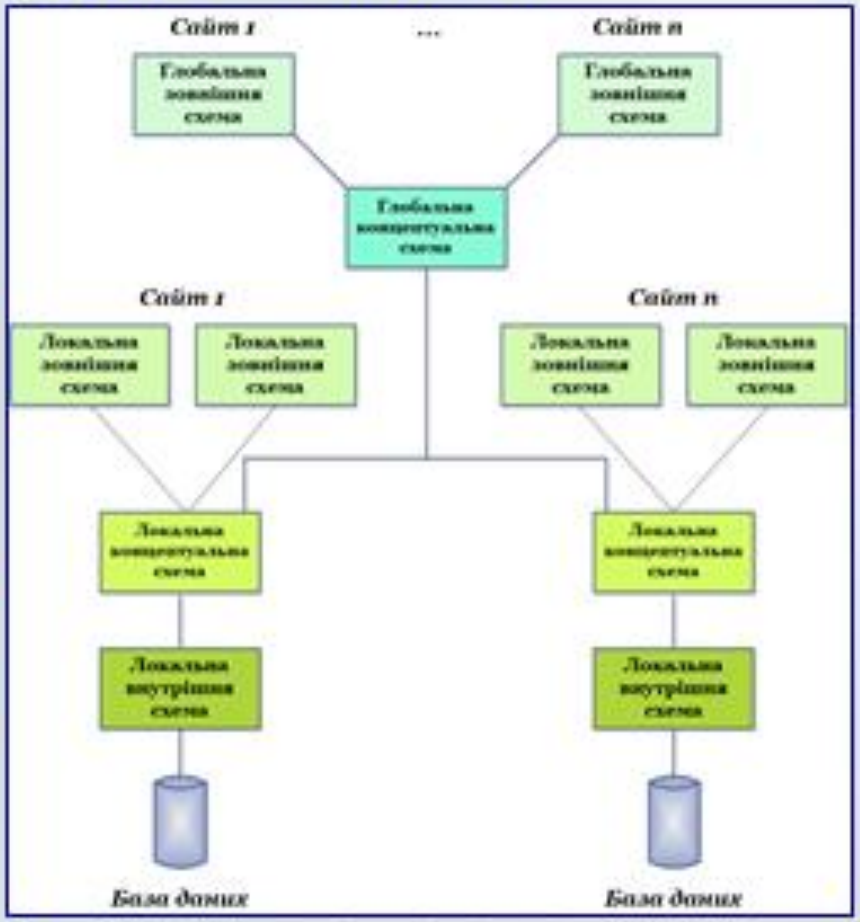
Функції та архітектура розподілених СУБД

Рекомендована архітектура мультибазових систем

Рекомендована архітектура федеральних мультибазових СУБД має враховувати рівень локальної автономності, який надається користувачеві.

Слабко пов'язана федеральна мультибазова система – це федеральна мультибазова система, яка не використовує глобальні концептуальні схеми. В такій системі зовнішні схеми складаються з однієї або декількох локальних концептуальних схем.

Тісно пов'язана федеральна мультибазова система – це федеральна мультибазова система, яка використовує глобальні концептуальні схеми, які є інтегрованим представленням або елементів локальних концептуальних схем, або локальних зовнішніх схем.



Архітектура, рекомендована для тісно зв'язаних федеральних мультибазових систем

@ М.В.Добролюбова

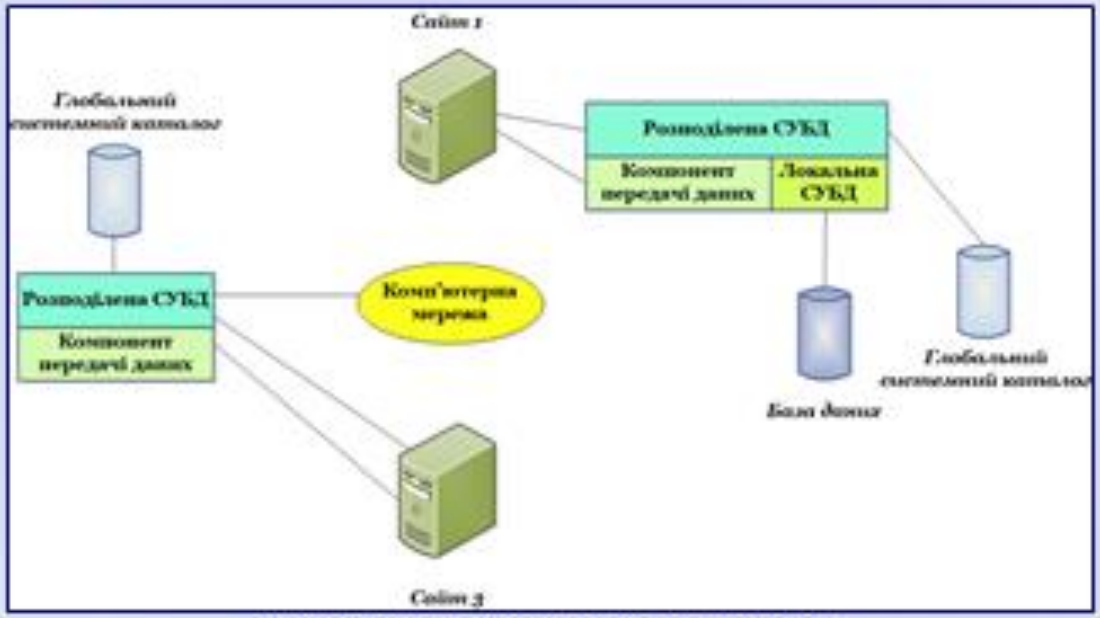
7

Функції та архітектура розподілених СУБД

Компонентна архітектура СУРБД

Елементи компонентної архітектури СУРБД

- локальна СУБД
- компонент передачі даних
- глобальний системний каталог
- розподілена СУБД (СУРБД)



Компонентна архітектура розподіленої СУБД

@ М.В.Добролюбова

8

Функції та архітектура розподілених СУБД

Правила Дейта для розподілених СУБД

Правило 0. Основний принцип: з точки зору кінцевого користувача розподілена система повинна виглядати так само, як звичайна, нерозподілена система.

Правило 1. Локальна автономність: сайти розподіленої системи повинні бути автономними.

Правило 2. Відсутність опори на центральний сайт: в системі не повинно бути жодного сайту, без якого вона не зможе функціонувати.

Правило 3. Безперервне функціонування: в системі ніколи не повинна виникати потреба у плановому припиненні її функціонування для виконання операцій додавання/видалення сайту з системи, динамічного створення/видалення фрагментів з одного або декількох сайтів.

Правило 4. Незалежність від розташування: користувач повинен отримувати доступ до бази даних з будь-якого сайту до будь-яких даних.

Правило 5. Незалежність від фрагментації: користувач повинен отримувати доступ до даних незалежно від способу їх фрагментації.

Правило 6. Незалежність від реплікації: користувач не повинен потребувати відомостей про наявність реплікації даних.

Правило 7. Обробка розподілених запитів: система повинна підтримувати обробку запитів, що посилаються на дані, розподілені на більш ніж одному сайті.

Правило 8. Обробка розподілених транзакцій: система повинна підтримувати виконання транзакцій, як одиниці відновлення.

Правило 9. Незалежність від типу обладнання: система повинна бути здатною функціонувати на обладнанні з різними обчислювальними платформами.

Правило 10. Незалежність від операційної системи: система повинна бути здатною функціонувати під управлінням різних операційних систем.

Правило 11. Незалежність від мережевої архітектури: система повинна бути здатною функціонувати в мережах з різною архітектурою та типами носія.

Правило 12. Незалежність від типу СУБД: система повинна підтримувати гетерогенність.

@ М.В.Добролюбова 9


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 2. НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ
Лекція 13
«Об'єктно-орієнтовані СУБД»
@ М.В.Добролюбова

Спеціалізовані додатки та недоліки РСУБД

Спеціалізовані додатки

- автоматизоване проєктування
- автоматизоване виробництво
- автоматизована розробка програмного забезпечення
- офісні інформаційні системи і мультимедіа системи
- цифрова видавнича справа
- геоінформаційні системи

Недоліки реляційних СУБД

- слабке представлення сутностей реального світу
- семантичне перевантаження
- слабка підтримка обмежень цілісності і корпоративних обмежень
- однорідна структура даних
- обмежений набір операцій
- труднощі організації рекурсивних запитів
- проблема неузгодженості
- проблеми, пов'язані з паралельністю, змінами схеми і слабкими засобами доступу

@ М.В.Добролюбова 2

Основні концепції об'єктно-орієнтованого підходу

Абстракція – це процес ідентифікації найбільш важливих аспектів сутності та ігнорування всіх інших її малозначних властивостей. В контексті створення програмного забезпечення це означає концентрацію уваги на те, що представляє собою об'єкт та що він може робити ще до вибору методу його реалізації.

Основні аспекти абстракції:

- інкапсуляція
- приховування інформації

Інкапсуляція означає, що об'єкт містить як структуру даних, так і набір операцій, за допомогою яких цією структурою можна маніпулювати.

Приховування інформації означає, що всі зовнішні аспекти об'єкта відокремлюються від подробиць його внутрішнього устрою, прихованих від зовнішнього світу.

Представлення про інкапсуляцію:

- представлення об'єктно-орієнтованої мови програмування (ООМП)
- адаптація представлення об'єктно-орієнтованої мови програмування для потреб баз даних

Об'єкт – це сутність, яка унікально ідентифікується та містить атрибути, що описують стан об'єктів реального світу і пов'язані з ними дії.

Атрибут – це елемент для опису поточного стану об'єкту.

Простий атрибут – це атрибут, який може мати примітивний тип та приймати літеральне значення.

Складовий атрибут – це атрибут, який може містити колекції і/або посилання.

Складовий об'єкт – це об'єкт, який містить один або кілька складових атрибутів.

Властивості ідентифікатора об'єкта:

- генерується системою
- унікально позначає об'єкт
- інваріантний
- не залежить від значень атрибутів об'єкта
- прихований від користувача

Переваги використання ідентифікаторів об'єктів:

- ефективність
- швидкість
- неможливість зміни користувачем
- незалежність від змісту даних

@ М.В.Добролюбова 3

Основні концепції об'єктно-орієнтованого підходу

Метод – це функція або процедура, яка визначає поведінку об'єкта та складається з імені і тіла. В об'єктно-орієнтованих мовах програмування тіло складається з блоку програмного коду, який виконує необхідні функції.

Повідомлення – це запит, спрямований одним об'єктом (відправником) на адресу іншого об'єкта (одержувача) та який вимагає, щоб об'єкт-одержувач виконав один зі своїх методів.

Клас – це шаблон для визначення набору подібних об'єктів.

Екземпляр – це об'єкт класу. В об'єктно-орієнтованих мовах новий екземпляр зазвичай створюється за допомогою команди new.

Конструктор – це метод створення екземплярів.

Деструктор – це метод видалення екземплярів з вивільненням зайнятого ними простору пам'яті.

Метаклас – це екземпляром класу більш високого рівня.

Наслідування – це парадигма об'єктно-орієнтованого програмування, яка дозволяє визначати один клас на основі більш загального класу.

Підклас – це менш загальний клас.

Суперклас – це більш загальний клас.

Узагальнення – це процес утворення суперкласу.

Спеціалізація – це процес утворення підкласу.

Зв'язок типу AKO (A Kind Of) – це зв'язок між підкласом і суперкласом.

@ М.В.Добролюбова 4

Основні концепції об'єктно-орієнтованого підходу

Види наслідування :

- одиничне
- множинне
- повторне
- вибіркове (селективне)

Рисунок 2 – Приклад множинного наслідування

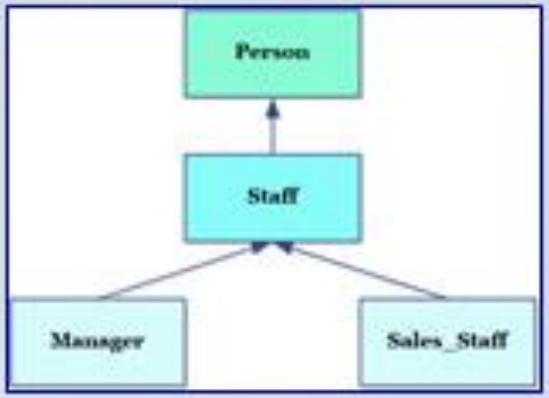


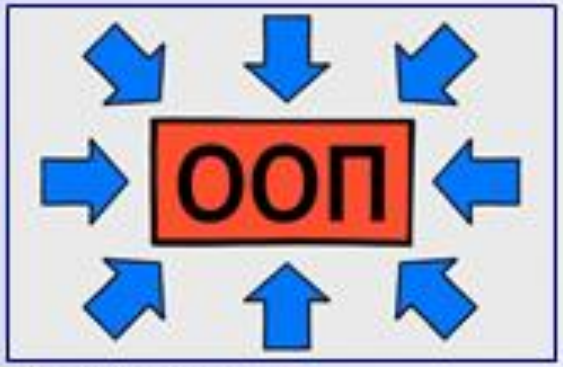
Рисунок 1 – Приклад одиничного наслідування

Рисунок 3 – Приклад повторного наслідування

@ М.В.Добролюбова

5

Основні концепції об'єктно-орієнтованого підходу



Перевизначення – це визначення властивості суперкласу у підкласі.

Перезавантаження – це окремий випадок поліморфізму – дозволяє повторно використовувати ім'я методу всередині визначення класу і визначень декількох класів.

Типи поліморфізму:

- робочий поліморфізм (перезавантаження)
- поліморфізм включення (метод, визначений в суперкласі та успадкований в його підкласі)
- параметричний поліморфізм (використання типів в якості параметрів в оголошеннях універсального типу або класу)

Зв'язування (binding) – це процес вибору відповідного методу, заснований на типі об'єкта.

Динамічне (dynamic) або *пізнє* (late) *зв'язування* – це відкладання процесу визначення типу об'єкта до настання часу виконання (а не під час компіляції).

@ М.В.Добролюбова

6

Визначення

Об'єктно-орієнтована модель даних – це логічна модель даних, що враховує семантику об'єктів, яка застосовується в об'єктно-орієнтованому програмуванні.

Об'єктно-орієнтована база даних – це перманентний набір (колекція) об'єктів для сумісного використання, визначений властивостями об'єктно-орієнтованої моделі даних.

Об'єктно-орієнтована СУБД – це система управління об'єктно-орієнтованими базами даних.

Мінімальна модель для об'єктно-орієнтованої СУБД:

1. Забезпечення функціональності бази даних
2. Підтримка ідентичності об'єкта
3. Забезпечення інкапсуляції
4. Підтримка об'єктів із складним станом

Об'єктно-орієнтована СУБД

1. "Об'єктно-орієнтований підхід" = "абстрактні типи даних" + "наслідування" + "ідентичність об'єктів".
2. "Об'єктно-орієнтована СУБД" = "об'єктно-орієнтований підхід" + "можливості бази даних".

Об'єктно-орієнтована СУБД

1. Високорівнева мова запитів із засобами оптимізації, реалізованими в базовій системі.
2. Підтримка перманентності і збереження атомарності транзакцій, що забезпечуються механізмами управління паралельністю і відновленням.
3. Підтримка складних об'єктних сховищ, індексів і методів доступу, призначених для швидкого і ефективного отримання даних.
4. "Об'єктно-орієнтована СУБД" = "об'єктно-орієнтована система" + «умови пунктів 1, 2 і 3".

@ М.В.Добролюбова 7

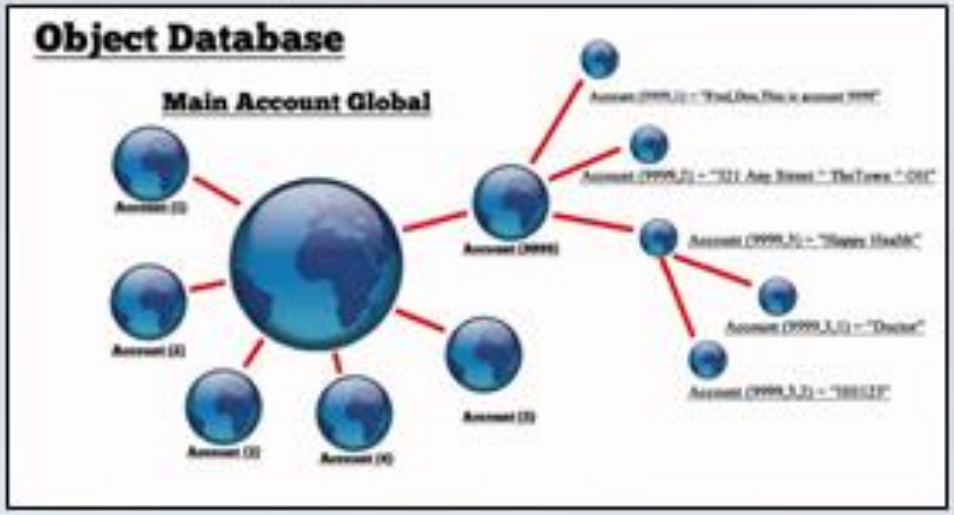
Перманентні мови програмування

Перманентна мова програмування – це мова, яка дозволяє її користувачам (прозоро) зберігати дані безпосередньо при послідовному виконанні програми, після чого ці дані зможуть використовуватися і багатьма іншими програмами.

Мова програмування баз даних – це мова, в якій інтегруються деякі ідеї, взяті як з моделі програмування баз даних, так і з концепцій традиційних мов програмування.

Підходи для розробки ООСУБД:

1. Розширення існуючої об'єктно-орієнтованої мови програмування можливостями роботи з базою даних
2. Надання розширюваних об'єктно-орієнтованих бібліотек СУБД
3. Вставка конструкцій об'єктно-орієнтованої мови бази даних в звичайну базову мову
4. Розширення існуючої мови бази даних об'єктно-орієнтованими функціями
5. Розробка нової мови бази даних або моделі даних



© М.В.Добролюбова

8

Аспекти функціонування

Компоненти і функціональні можливості сучасних систем баз даних:

1. Модель даних
2. Перманентність даних
3. Спільне використання даних
4. Надійність
5. Масштабованість
6. Безпека і цілісність
7. Розподіленість

Схеми забезпечення перманентності мов програмування:

1. Створення контрольних точок
2. Серіалізація
3. Явна підкачка сторінок

Аспекти функціонування та побудови СОСУБД:

- транзакції
- версії
- еволюція схеми
- архітектура

Транзакція – це логічна одиниця роботи, яка завжди повинна переводити базу даних з одного несуперечливого стану в інший.

Підходи для вирішення проблем з довготривалими транзакціями:

- використання версій
- удосконалення моделей обробки транзакцій

@ М.В.Добролюбова 9

Аспекти функціонування

Управління версіями – це процес супроводу еволюції об'єктів.

Способи управління версіями:

- тимчасові версії
- робочі версії
- остаточні версії

Перелік типових змін, які можуть вноситись в схему бази даних:

1. Зміна визначення класу
   - зміна атрибутів
   - зміна методів
2. Зміна ієрархії наслідування
   - визначення класу *S* як суперкласу для класу *C*
   - видалення класу *S* зі списку суперкласів для класу *C*
   - зміна порядку суперкласів для класу *C*
3. Зміни набору класів, наприклад створення і видалення класів, зміна імен класів

@ М.В.Добролюбова 10

Аспекти функціонування

Аспекти архітектурних рішень ООСУБД:

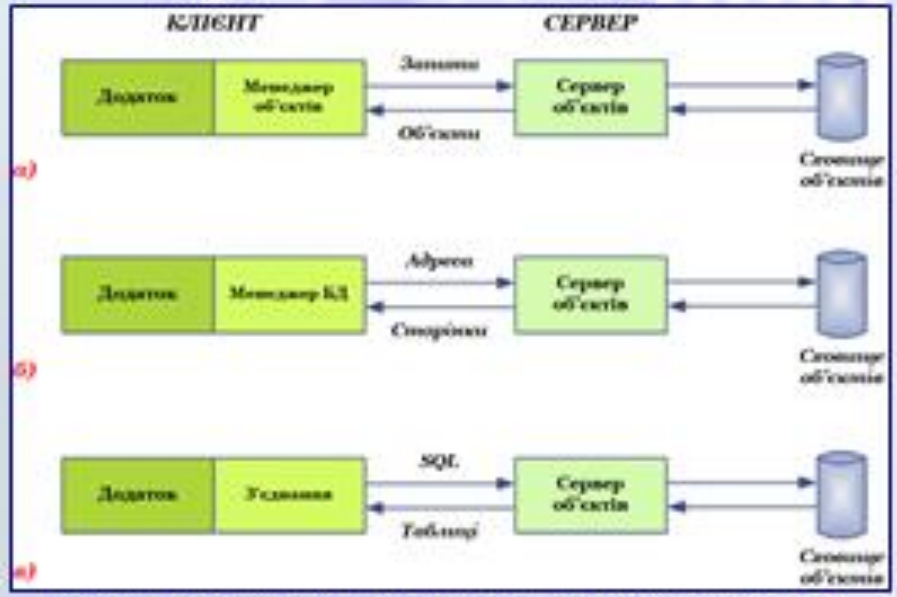
- оптимальне застосування архітектури клієнт/сервер
- способи збереження методів

Основні типи моделі "клієнт/сервер":

- сервер об'єктів
- сервер сторінок
- сервер бази даних

Підходи до управління методами:

- збереження методів у зовнішніх файлах
- збереження методів в базі даних



Варіанти архітектури «клієнт-сервер», які використовуються в ООСУБД: а) сервер об'єктів; б) сервер сторінок; в) сервер бази даних

Переваги підходу із збереженням методів в базі даних:

- виключається надлишковий код
- спрощуються модифікації
- методи захищені більшою мірою
- збереження методів в базі даних дозволяє застосувати до них усі заходи безпеки, що надаються ООСУБД
- методи можуть сумісно використовуватися у паралельному режимі
- підвищена цілісність

@ М.В.Добролюбова 11

Маніфест, переваги та недоліки ООСУБД

Маніфест об'єктно-орієнтованих СУБД

Правило 1. Підтримка складених об'єктів
Правило 2. Підтримка ідентичності об'єктів
Правило 3. Підтримка інкапсуляції
Правило 4. Підтримка типів або класів
Правило 5. Підтримка наслідування типів або класів від їх предків
Правило 6. Підтримка динамічного зв'язування
Правило 7. Мова DML повинна володіти обчислювальною повнотою
Правило 8. Набір типів даних має бути розширюваним
Правило 9. Підтримка перманентності даних
Правило 10. Підтримка дуже великих баз даних
Правило 11. Підтримка паралельної роботи багатьох користувачів
Правило 12. Забезпечення відновлення після збоїв апаратного та програмного забезпечення
Правило 13. Надання простих способів створення запитів до даних

Переваги ООСУБД

- збільшення можливості моделювання
- можливість розширення
- усунення проблеми невідповідності
- більш виразна мова запитів
- підтримка еволюції схеми
- підтримка довготривалих транзакцій
- застосування для складних спеціалізованих додатків баз даних
- підвищена продуктивність

Недоліки ООСУБД

- відсутність універсальної моделі даних
- недостатність досвіду експлуатації
- вплив оптимізації запитів на інкапсуляцію
- вплив блокування на рівні об'єкта на продуктивність
- складність
- відсутність підтримки представлень
- недостатність коштів забезпечення безпеки

@ М.В.Добролюбова 12


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 2. НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ
Лекція 14
«Об'єктно-реляційні СУБД»
@ М.В.Добролюбова

Вступ до об'єктно-реляційних СУБД

Об'єктно-реляційна СУБД (Object-Relational DBMS) – це СУБД з розширеною реляційною моделлю даних такими об'єктно-орієнтованими компонентами, як система типів, що розширюється користувачем, інкапсуляція, наслідування, поліморфізм, динамічне зв'язування методів, використання складових об'єктів підтримка ідентичності об'єктів.

Переваги ОРСУБД

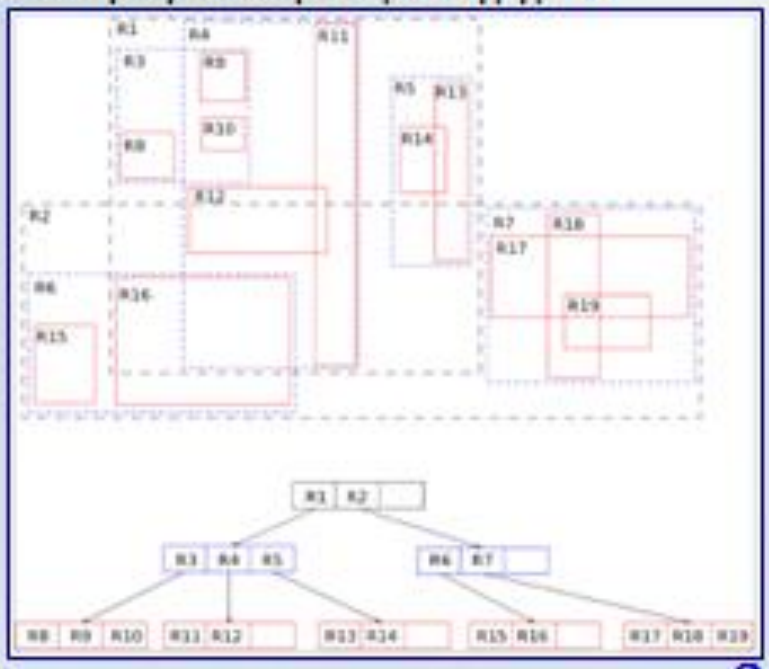
- повторне і спільне використання компонентів
- підвищення продуктивність праці розробників і кінцевих користувачів
- великий об'єм накопичених знань і досвіду, пов'язаних з розробкою реляційних додатків

Недоліки ОРСУБД

- складність
- підвищені витрати

Основне завдання розробників ОРСУБД

- повне розкриття особливостей оптиміватора запитів – важливого та складного в налаштуванні компонента СУБД
- навчання сторонніх розробників технологіям оптимізації



@ М.В.Добролюбова

2

Маніфести новітніх БД

Маніфест баз даних, опублікований комітетом CADF

Правило 1. Велика система типів

Правило 2. Підтримка механізму наслідування

Правило 3. Підтримка функцій, процедур, методів та механізму інкапсуляції

Правило 4. Присвоєння засобам СУБД унікальних ідентифікаторів для записів тільки в тому випадку, коли не можна використовувати первинні ключі, визначені користувачем

Правило 5. Правила (тригери, обмеження) не повинні бути пов'язані з якоюсь конкретною функцією або колекцією

Правило 6. Здійснення всіх програмних способів доступу до бази даних за допомогою непроцедурної мови високого рівня

Правило 7. Існування принаймні двох способів визначення колекцій: один – на основі перерахування членів колекції, інший – з використанням мови запитів для визначення членства в колекції

Правило 8. Наявність представлень, які оновлюються

Правило 9. Індикатори продуктивності не повинні мати ніякого відношення до моделей даних та не повинні бути присутніми в них

Правило 10. СУБД повинні бути доступні з багатьох мов високого рівня

Правило 11. Наявність у мов програмування перманентних форм

Правило 12. Мова SQL є та залишається "міжгалактичною мовою роботи з даними"

Правило 13. Запити та відповіді на них повинні складати найнижчий рівень взаємодії клієнта і сервера

© М.В.Добролюбова 3

Маніфести новітніх БД

Маніфест Дарвена та Дейта на захист реляційної моделі даних

Вимоги до мови D

Реляційна модель	
Приписи	**Заборони**
1. Домени 2. Типізовані скаляри 3. Скалярні оператори 4. Фактичне подання 5. Значення істинності 6. Тип-конструктор TUPLE 7. Тип-конструктор RELATION 8. Оператор еквівалентності 9. Кортежі 10. Відношення 11. Скалярні змінні 12. Кортежні змінні 13. Змінні відношень (rel-var) 14. Базові або похідні змінні відношень 15. Змінні баз даних (db-var) 16. Транзакції та змінні баз даних 17. Оператори створення/видалення 18. Реляційна алгебра 19. Імена змінних відношень та значення явних відношень 20. Функції відношень 21. Присвоєння відношення кортежу 22. Порівняння 23. Обмеження цілісності 24. Предикати відношень баз даних 25. Каталог 26. Макет мова	1. Відсутність упорядкування атрибутів 2. Відсутність упорядкування кортежів 3. Відсутність дублікатів кортежів 4. Відсутність невизначених значень (NULL) 5. Відсутність NULL-логічних поняття 6. Відсутність конструкцій на внутрішньому рівні 7. Відсутність операторів на рівні кортежів 8. Відсутність складових стовпців 9. Відсутність підміни перевірки відповідності домену 10. Відсутність мови SQL
Інші вимоги, що не мають відношення до реляційної моделі	
Приписи	**Заборони**
1. Контроль типів під час компіляції 2. Одиничне наслідування (частково) 3. Множинне наслідування (умовно) 4. Обчислювальна повнота 5. Явне розмежування транзакцій 6. Вкладені транзакції 7. Агреговані значення та порожні множини	1. Змінні відношень (rel-var) не є доменами 2. Відсутність ідентифікаторів об'єктів 3. Відсутність "відкритих змінних екземплярів" 4. Відсутність "закритих змінних екземплярів" або дружніх їм

Дуже суворі вимоги мови D

Реляційна модель	Інші вимоги
1. Потенційні ключі для похідних змінних відношення	1. Наслідування типів
2. Генеровані системою ключі	2. Типи-конструктори колекцій
3. Цілісність посилань	3. Перетворення відношення назад
4. Виведення потенційних ключів	4. Одноріднене сховище
5. Часткові запити (наприклад, "знайти трьох найбільш молодих співробітників")	
6. Транзитивне замикання (для відношень)	
7. Параметри кортежу відношення	
8. Значення, що приймаються за замовчуванням	
9. SQL-чистий	

@ М.В.Добролюбова

4

Функція	ОРСУБД	ООСУБД
Ідентичність об'єкта	Підтримується на основі типу REF	Підтримується
Інкапсуляція	Підтримується на основі UDF-типів	Підтримується, але порушується для запитів
Наслідування	Підтримується (окремі ієрархії для UDT-типів та таблиць)	Підтримується
Поліморфізм	Підтримується (виклик UDF-функцій на основі універсальної функціональної моделі)	Підтримується як в об'єктно-орієнтованій мові програмування
Складові об'єкти	Підтримуються на основі UDT-типів	Підтримуються
Зв'язки	Строга підтримка на основі обмежень цілісності користувача	Підтримуються (наприклад, за допомогою бібліотек класів)

Функція	ОРСУБД	ООСУБД
Створення перманентних даних та доступ до них	Підтримується, але не прозоро	Підтримується, але ступінь прозорості варіюється для різних продуктів
Декларативні запити	Сувора підтримка	Підтримується на основі стандарту ODMG
Навігація	Підтримується на основі типу REF	Сувора підтримка
Обмеження цілісності	Сувора підтримка	Не передбачені
Сервер об'єктів / сервер сторінок	Сервер об'єктів	Обидва
Еволюція схеми	Обмежена підтримка	Підтримується, але ступінь підтримки варіюється для різних продуктів

Функція	ОРСУБД	ООСУБД
ACID-транзакції	Сувора підтримка	Підтримується
Відновлення	Сувора підтримка	Підтримується, але ступінь підтримки варіюється для різних продуктів
Вдосконалені моделі обробки транзакцій	Не передбачені	Підтримується, але ступінь підтримки варіюється для різних продуктів
Безпека, цілісність та представлення	Сувора підтримка	Обмежена підтримка


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 2. НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ
Лекція 15
«Web-технології та СУБД»
© М.В.Добролюбова

Основні поняття Internet та Web

Мережа APRANET

Мережа NSFNET

Протокол TCP – це протокол, який регламентує коректну доставку повідомлень з одного комп'ютера на інший.

Протокол IP – це протокол, який регламентує відправку і отримання пакетів даних при передачі їх між різними комп'ютерами на основі чотирибайтової адреси призначення (IP-адреси).

Intranet – це єдиний Web-сайт або група Web-сайтів, що належать одній організації і є доступними тільки членам цієї організації.

Брандмауер – це захисний екран між глобальним Internet і локальною комп'ютерною мережею організації, який виконує функцію перевірки і фільтрації даних, що надходять з Internet, тобто в залежності від налаштувань може пропустити їх або заблокувати.

Extranet – це різновид мережі Intranet, яка частково доступна санкціонованим зовнішнім користувачам.

EDI (Electronic Data Exchange) – це специфікація електронного обміну даними, яка дозволяє організаціям зв'язати воєдино системи обліку товарів і обліку покупок / замовлень.

Аплет – це несамостійний компонент програмного забезпечення, який працює в контексті повноцінного додатка та не має цінності у відриві від базової програми.

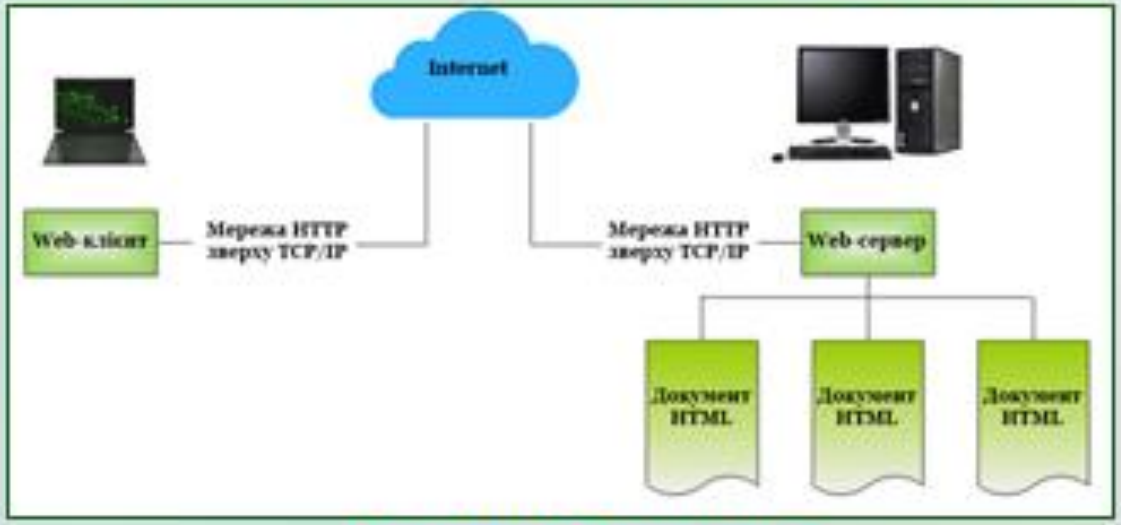
@ М.В.Добролюбова 2

Основні поняття Internet та Web

World Wide Web – це гіпермедіа система, яка надає прості засоби для перегляду інформації в мережі Internet за допомогою механізму гіперпосилань.

Протокол HTTP (Hyper Text Transfer Protocol) – це протокол, згідно з яким відбувається обмін інформацією між Web-сервером і браузером.

Схема взаємодії основних компонентів середовища WWW



Етапи HTTP-транзакції:

- підключення
- запит
- відгук
- закриття

© М.В.Добролюбова 3

Основні поняття Internet та Web

HTML – це мова форматування документів, яка використовується для створення більшості Web-сторінок.

URL-адреса (Uniform Resource Locator) – це спеціальна адреса, за допомогою якої визначаються документи і окремі місця розташування даних всередині них.

Статична Web-сторінка – це HTML-документ, що зберігається в окремому файлі, вміст якого не змінюється до тих пір, поки не зміниться сам файл.

Динамічна Web-сторінка – це сторінка, вміст якої генерується всякий раз при доступі до неї.

Текст деякої Web-сторінки мовою HTML та її відображення в браузері

Синтаксис URL-адреси

<протокол>: // <вузол> [: <порт>] / абсолютний_шлях [? аргументи]

WEB-середовище, як платформа для додатків баз даних

Вимоги до інтеграції СУБД в середовище Web

- можливість захищеного доступу до цінних корпоративних даних
- спосіб підключення, що не залежить від даних і розробника програмного забезпечення
- можливість взаємодії з базою даних, незалежно від конкретного типу використовуваного Web-браузера або Web-сервера
- наявність такого підключення, яке дозволяє отримати всі переваги всіх компонентів СУБД, що використовується в даній організації
- відкритість архітектури, що дозволяє взаємодіяти з різноманітними системами та технологіями
- економічно ефективне рішення, що допускає масштабованість, зростання, зміну стратегічних напрямків та сприяє скороченню витрат на розробку і супровід додатків
- підтримка транзакцій, які покривають кілька HTTP-запитів
- підтримка сеансів на основі ідентифікації користувачів засобами СУБД і додатків
- прийнятна продуктивність
- мінімальний рівень адміністрування
- набір високорівневих інструментів розробки

@ М.В.Добролюбова 5

WEB-середовище, як платформа для додатків баз даних

Переваги інтеграції СУБД в середовище Web

- переваги використання функцій СУБД
- простота реалізації
- незалежність від платформи
- графічний інтерфейс користувача
- стандартизація
- міжплатформена підтримка
- прозорий мережевий доступ
- масштабованість розгортання
- іноваційність

Недоліки інтеграції СУБД в середовище Web

- недостатня надійність
- слабка захищеність
- висока вартість
- труднощі з визначенням масштабу
- обмежена функціональність мови HTML
- відсутність запам'ятовування стану
- високі вимоги до пропускної здатності мережі
- недостатня продуктивність
- недосконалість інструментів розробки

© М.В.Добролюбова

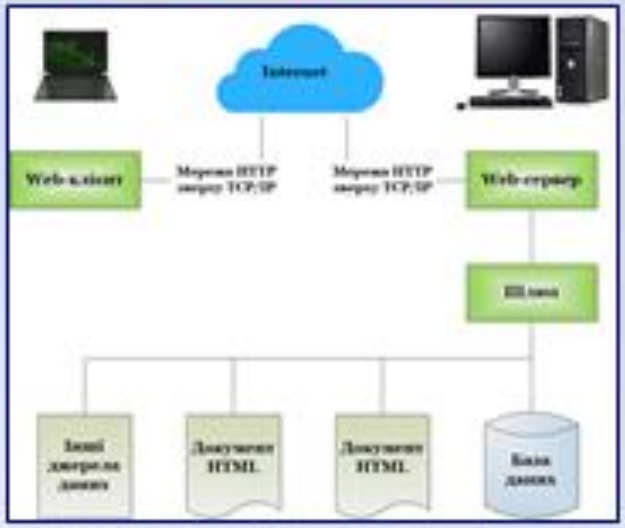
6

Методи інтеграції СУБД у середовище WEB

- CGI-сценарії
- Серверні вставки Server-Side Includes
- Файли cookies мови HTTP
- Розширення Web-сервера
- Мови C# та Java і їх розширення
- Мови сценаріїв JavaScript і VBScript
- Платформа Active Platform фірми Microsoft і її трансформація в .Net

`Content-type: text/html`	HTML-документ
`Content-type: text/plain`	Звичайний текст
`Content-type: image/gif`	Графічний файл формату GIF (Graphics Interchange Format)

Загальна схема CGI-механізма



Порівняльні характеристики сценаріїв мови JavaScript та аплетів мови Java

Сценарії мови JavaScript	Аплети мови Java
Інтерпретуються (не компілюються) на стороні клієнта	Компілюються на боці сервера до виконання на боці клієнта
Об'єктного типа. В коді використовуються вбудовані, розширювані об'єкти, але без визначення класів або наслідування	Об'єктно-орієнтований. Аплети складаються з об'єктних класів з можливістю наслідування
Код може інтегруватися та впроваджуватися в HTML-документи	Аплети розміщуються поза HTML-тексту (викликаються з HTML-сторінок)
Тип даних змінних не оголошується (несуворий контроль типів)	Тип даних змінних має бути оголошеним (суворий контроль типів)
Динамічне зв'язування. Посилання на об'єкти перевіряються під час виконання	Статичне зв'язування. Посилання на об'єкти повинні перевірятися під час компіляції
Не може автоматично записувати дані на жорсткий диск	Не може автоматично записувати дані на жорсткий диск

© М.В.Добролюбова 7


Методи забезпечення безпеки
Безпека в C# та Java середовищах
Проксі-сервери
Брандмауери
Протоколи Secure Sockets Layer (SSL) та Secure HTTP (S-HTTP)
БЕЗПЕКА
Алгоритми стиснення повідомлень та цифрові підписи
Цербери
Цифрові сертифікати
@ М.В.Добролюбова
8

Протокол HTTP та мова XML

Версії протоколу HTTP

- HTTP / 0.9
- HTTP / 1.0
- HTTP / 1.1
- HTTP / 2.0
- HTTP / 3.0

Корисні елементи мови XML

- визначення схеми бази даних
- зв'язування споріднених об'єктів або елемнтів
- підтримка двонапрямлених посилань

@ М.В.Добролюбова 9


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 2. НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ
Лекція 16
«Сховища даних»
@ М.В.Добролюбова

Поняття сховищ даних

Сховище даних – предметно-орієнтований, інтегрований, прив'язаний до часу незмінний набір даних, призначений для підтримки та прийняття рішень.

Характеристики сховища даних:

- предметна орієнтованість
- інтегрованість
- прив'язка до часу
- незмінюваність

Технологія сховищ даних – це технологія управління даними і їх аналізу.

Переваги технології сховищ даних:

- потенційно висока віддача від інвестицій
- підвищення конкурентоспроможності
- підвищення ефективності праці осіб, відповідальних за прийняття рішень

@ М.В.Добролюбова

2

Поняття сховищ даних

Порівняння основних характеристик типової OLTP-системи та сховища даних

OLTP-система	*Сховище даних*
Містить поточні дані	Містить історичні дані
Зберігає докладні відомості	Зберігає докладні відомості, також частково та значно узагальнені дані
Дані є динамічними	Дані в основному є статичним
Повторюваний спосіб обробки даних	Нерегламентований, неструктурований і евристичний спосіб обробки даних
Висока інтенсивність обробки транзакцій	Середня та низька інтенсивність обробки транзакцій
Передбачуваний спосіб використання даних	Непередбачуваний спосіб використання даних
Призначена для обробки транзакцій	Призначена для проведення аналізу
Орієнтована на прикладні області	Орієнтована на предметні області
Підтримка прийняття повсякденних рішень	Підтримка прийняття стратегічних рішень
Обслуговує велику кількість працівників виконавчої ланки	Обслуговує відносно малу кількість працівників керівної ланки

Проблеми сховищ даних:

- недооцінювання ресурсів, необхідних для завантаження даних
- приховані проблеми джерел даних
- відсутність необхідних даних у наявних архівах
- підвищення вимог кінцевих користувачів
- гомогенізація даних
- високі вимоги ресурсів
- володіння даними
- складний супровід
- довготривалий характер проєктів
- складнощі інтеграції

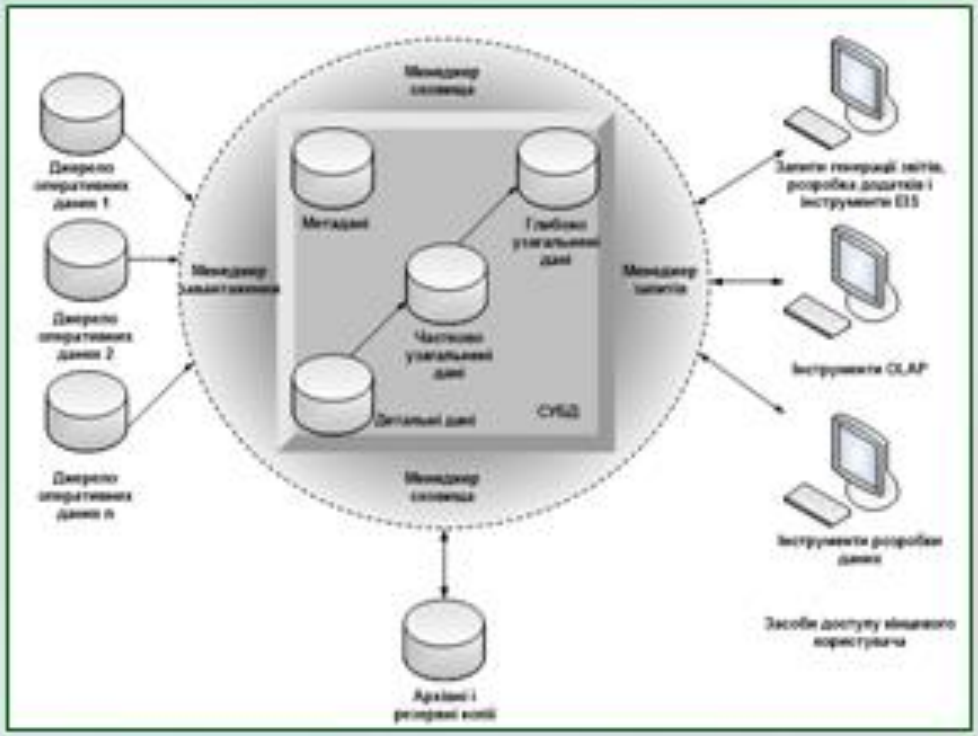
© М.В.Добролюбова

3

Архітектура сховищ даних

Основні компоненти сховищ даних:

- оперативні дані
- менеджер завантаження
- менеджер сховища
- менеджер запитів
- детальні дані
- частково та глибоко узагальнені дані
- архівні та резервні копії
- метадані
- засоби доступу до даних кінцевого користувача



Типова архітектура сховища даних

@ М.В.Добролюбова

4

Архітектура сховищ даних

Менеджер завантаження – це зовнішній компонент, який виконує всі операції, пов'язані з отриманням і завантаженням даних в сховище.

Менеджер сховища – це компонент, який виконує всі операції, пов'язані з управлінням інформацією, розміщеною в сховищі даних.

Операції менеджера сховища:

- аналіз несуперечності даних
- перетворення і переміщення вихідних даних з тимчасового сховища в основні таблиці сховища даних
- створення індексів і представлень для базових таблиць
- денормалізація даних (в разі необхідності)
- узагальнення даних (в разі необхідності)
- резервне копіювання та архівування даних

Менеджер запитів – це внутрішній компонент, який виконує всі операції, пов'язані з управлінням призначеними для користувача запитами.

Основні групи інструментів користувачів для доступу до даних:

- інструменти створення звітів і запитів
- інструменти розробки додатків
- інструменти інформаційної системи керівника
- інструменти оперативної аналітичної обробки (OLAP-інструменти)
- інструменти розробки даних

OLAP-інструменти – це інструменти оперативної аналітичної обробки даних, які створюються на основі концепції багатовимірної бази даних та дозволяють кваліфікованим користувачам аналізувати дані за допомогою складних багатовимірних представлень.

Розробка даних – це процес відкриття нових усвідомлених кореляцій, розподілів і тенденцій шляхом переробки величезної кількості інформації, витягнутої із сховища або магазину даних, з використанням статистичних і математичних методів, а також методів штучного інтелекту.

© М.В.Добролюбова 5

Інформаційні потоки в сховищі

Вхідний потік – це процеси, пов'язані з отриманням, очищенням та завантаженням інформації з джерел даних у сховище даних.

Висхідний потік – це процеси, пов'язані з підвищенням цінності даних, що зберігаю у сховищі, за допомогою узагальнення, упаковки та розподілення вихідних даних.

Низхідний потік – це процеси, пов'язані з архівуванням та резервним копіюванням інформації у сховищі даних.

Вихідний потік – це процеси, пов'язані з наданням даних користувачам.

Метапотік – це процеси, пов'язані з управлінням метаданими.

@ М.В.Добролюбова 6

Інструменти та технології сховищ

Інструменти і технології, які використовуються при створенні і супроводі сховищ даних:

- інструменти отримання, очищення та перетворення даних
- СУБД для сховища даних
- метадані сховища даних
- інструменти управління та адміністрування

Інтегровані рішення для отримання, очищення та перетворення даних з наступним завантаженням у конкретну систему:

- генератори коду
- інструменти реплікації інформації бази даних
- механізми динамічного перетворення

Вимоги до СУБД для сховища даних:

- висока продуктивність завантаження даних
- можливість обробки даних під час завантаження
- наявність засобів управління якістю даних
- висока продуктивність запитів
- широка масштабованість за розміром
- масштабованість за кількістю користувачів
- можливість організації мережі сховищ даних
- наявність засобів адміністрування сховища
- підтримка інтегрованого багатовимірного аналізу
- розширений набір функціональних засобів запитів

@ М.В.Добролюбова 7

Інструменти та технології сховищ

Основне призначення метаданих – це збереження інформації про виникнення даних з метою надання можливості адміністраторам сховища знати історію будь-якого елементу даних, який розміщений у сховищі.

Метадані, пов'язані з перетворенням та завантаженням даних, повинні описувати джерело даних та будь-які зміни, внесені в ці дані.

Метадані, пов'язані з управлінням даними, повинні описувати спосіб збереження даних у сховищі.

Операції, які повинні дозволяти виконувати інструменти управління і адміністрування сховищ даних:

- моніторинг завантаження даних з декількох джерел
- перевірка якості і цілісності даних
- управління та оновлення метаданих
- моніторинг поточної продуктивності бази даних з метою забезпечення швидкої реакції на запит та ефективного використання ресурсів
- аудит процесів використання сховища даних з метою збору інформації про вартість виконаної кожним користувачем роботи
- реплікація, розбиття та розподіл даних
- підтримка ефективного управління сховищем даних
- очищення даних
- архівування і резервне копіювання даних
- засоби відновлення після збою
- управління засобами захисту

@ М.В.Добролюбова 8

Проєктування сховищ

Схема "зірка" – це логічна структура, в центрі якої знаходиться таблиця фактів (з детальними даними), оточена таблицями розмірності (з даними-посиланнями).

Чинники, необхідні для оптимального проєктування бази даних:

- визначення необхідного часу відгуку для кожного додатку підтримки прийняття рішення
- пошук компромісу між необхідністю використання статистичних вибірок підмножин даних та необхідністю обробки детальних відомостей
- визначення стовпців, що видаляються
- скорочення розміру стовпців таблиці фактів
- визначення найкращого способу застосування зовнішніх ключів, які налаштовуються та не налаштовуються
- визначення оптимального підходу для введення в таблицю фактів розмірності "час"
- секціонування таблиць фактів для поліпшення їх керованості

Схема "сніжинка" – це варіант схеми "зірка", в якому кожна розмірність може мати свої власні розмірності.

Схема "зірка-сніжинка" – це гібридна структура, яка включає комбінацію денормалізованої схеми "зірка" і нормалізованої схеми "сніжинка".

@ М.В.Добролюбова 9


ПРОГРАМУВАННЯ БАЗ ДАНИХ
РОЗДІЛ 2
Деякі аспекти експлуатації баз даних
Тема 2. НОВІ ТА ПЕРСПЕКТИВНІ НАПРЯМИ
Лекція 17
«OLAP та розробка даних»
© М.В.Добролюбова

Інтерактивна аналітична обробка даних

Оперативна аналітична обробка (OLAP) – це динамічний синтез, аналіз та консолідація великих об'ємів багатовимірних даних.

Основні аналітичні операції серверів багатовимірних баз даних на основі OLAP:

- консолідація
- низхідний аналіз
- розбиття з поворотом

Правила для OLAP-систем:

1. Багатовимірне концептуальне представлення даних
2. Прозорість
3. Доступність
4. Незмінна продуктивність підготовки звітів
5. Архітектура "клієнт / сервер"
6. Універсальність вимірювань
7. Динамічне управління розрідженістю матриць
8. Розрахована на багато користувачів підтримка
9. Необмежені перехресні операції між розмірностями
10. Підтримка інтуїтивно зрозумілого маніпулювання даними
11. Гнучкість засобів формування звітів
12. Необмежена кількість вимірювань та рівнів узагальнення

Основні категорії OLAP-інструментів:

- багатовимірні OLAP-інструменти
- реляційні OLAP-інструменти
- кероване середовище запитів

© М.В.Добролюбова 2

Технологія розробки даних

Розробка даних – процес вилучення з великих баз даних достовірної, попередньо невідомої, комплексної та значимої інформації і використання її для прийняття відповідальних бізнес-рішень.

Приклади додатків технології розробки даних

Роздрібна торгівля та маркетинг
Виявлення закономірностей у покупках, зроблених клієнтами
Пошук асоціативних зв'язків серед демографічних характеристик клієнтів
Прогнозування реакції на кампанію розсилки поштових матеріалів
Аналіз споживчого кошика
Банківська сфера
Виявлення закономірностей у шахрайському використанні кредитних карток
Пошук постійних клієнтів
Прогнозування клієнтів, які, мабуть, змінять свою приналежність до кредитної карткової системи
Визначення середніх витрат за кредитними картками для різних груп клієнтів
Страхування
Визначення тих клієнтів, які куплять нові страхові поліси
Медицина
Визначення характеристик поведінки пацієнтів з метою планування їх прийомів
Пошук успішних терапевтичних процедур при різних захворюваннях

Технологія розробки даних

Операції розробки даних та пов'язані з ними методи

Операції	*Методи*
Прогнозуюче моделювання	Класифікація Прогнозування значення
Сегментування баз даних	Демографічна кластеризація Нейронна кластеризація
Аналіз зв'язків	Виявлення асоціативних зв'язків Пошук послідовних закономірностей Пошук схожих тимчасових послідовностей
Виявлення відхилень	Статистика Візуалізація

@ М.В.Добролюбова 4

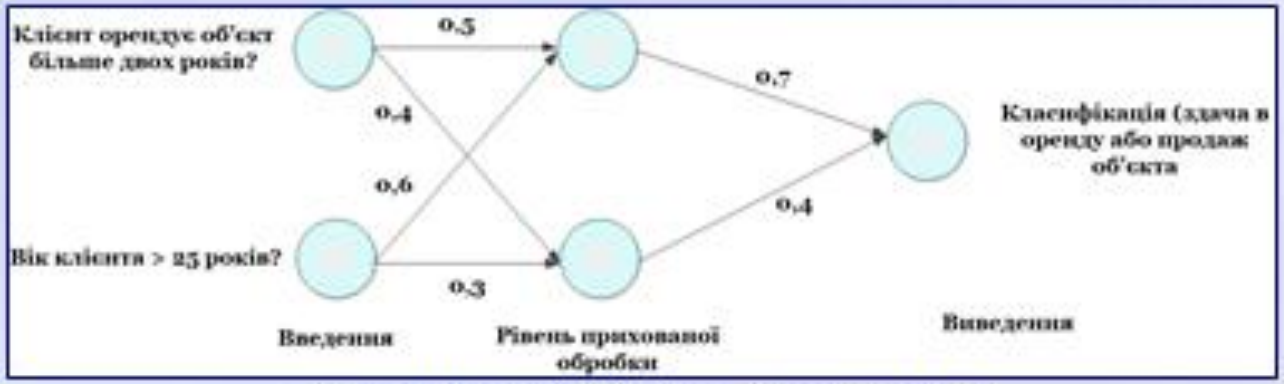
Технологія розробки даних

ПРОГНОЗУЮЧЕ МОДЕЛЮВАННЯ

Методи прогнозуючого моделювання:

- класифікація
- прогнозування значення

Приклад класифікації на основі деревовидної індукції



Приклад класифікації на основі нейронної індукції

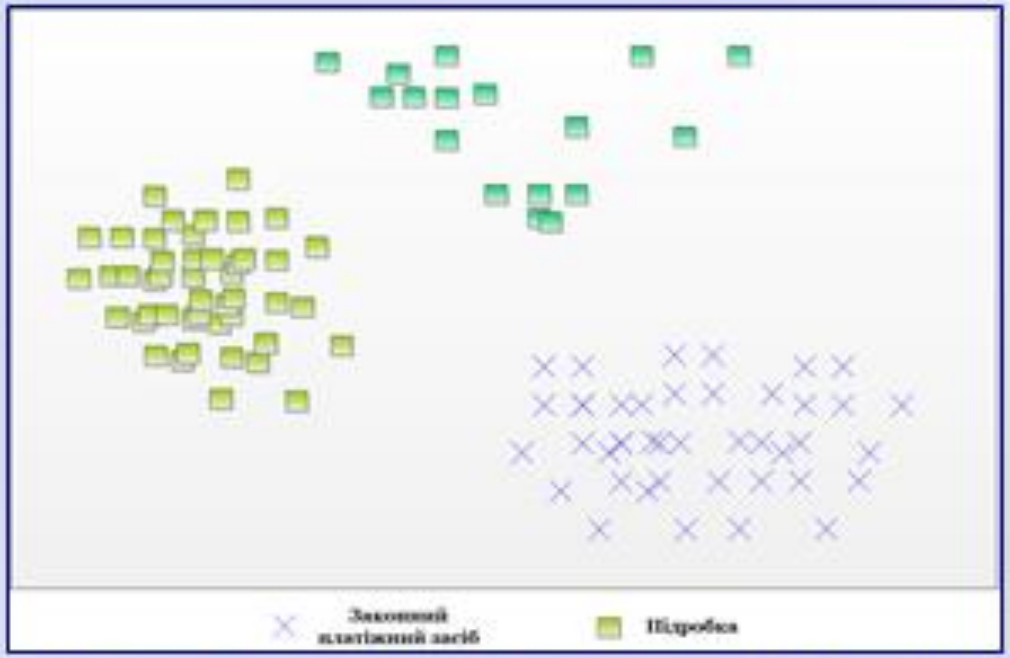
@ М.В.Добролюбова

5

Технологія розробки даних

СЕГМЕНТУВАННЯ БАЗИ ДАНИХ

Мета сегментування бази даних – розбиття бази даних на деяку, заздалегідь невідому кількість сегментів, або кластерів, що складаються з подібних записів, тобто записів, які мають деякі загальні властивості, що дозволяє вважати їх однорідними.



Приклад використання методу сегментування бази даних на основі графіка

@ М.В.Добролюбова 6

Технологія розробки даних

АНАЛІЗ ЗВ'ЯЗКІВ

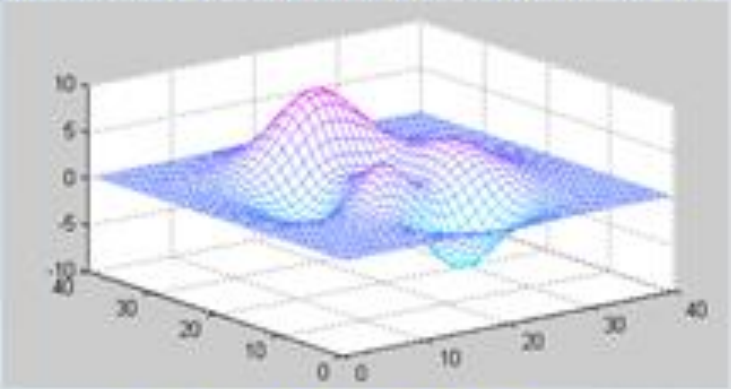
Мета аналізу зв'язків – встановлення зв'язків, або асоціацій, між окремими записами або наборами записів в базі даних.

Спеціалізації методу аналізу зв'язків:

- пошук асоціацій
- пошук послідовних закономірностей
- пошук аналогічних часових послідовностей

ВИЯВЛЕННЯ ВІДХИЛЕНЬ

Метод виявлення відхилень передбачає аналіз викидів, які висловлюють ступінь відхилення від деяких попередньо відомих значень математичного очікування і нормального середнього.



Приклад візуалізації даних

© М.В.Добролюбова

7

ДЯКУЮ ЗА УВАГУ!

Лектор

к.т.н., доц., доцент кафедри ІВТ
КПІ ім. Ігоря Сікорського
М. В. Добролюбова