Дата: 19.01.2023
Група: 23
Предмет: Мова SQL та бази даних

УРОК 3
Тема: «Основні функції систем керування базами даних»

Мета:

- Опанувати поняття «система управління базою даних»
- Сформувати в учнів уяву про класифікацію СУБД, про основні інструменти MS ACCESS
- Вивчити характеристики СУБД, її функції, об’єкти та види
- Виховати інформаційно освічену і компетентну особистість, зацікавленість до обраної професії

1. Поняття про СКБД (Системи керування базами даних)

Уже при роботі з першими БД виявилось, що на створення програмного забезпечення для кожної з них витрачається багато часу і зусиль. Тому виникла необхідність у розробці програм універсального характеру відносно змісту тієї інформації, яка зберігається в БД. Це привело до появи систем керування БД як універсального інструменту для створення та експлуатації баз даних.

Під системою керування базами даних (СКБД) розуміють таку програмну систему, яка забезпечує виконання всіх операцій, які пов'язані із створенням БД, збереженням її на магнітних носіях, обробкою даних, що міститься в БД, розв'язанням прикладних задач, формуванням звітів та підсумкових документів.

Функціями СКБД є:

● можливість здійснювати опис даних та змінювати їх структуру;

● маніпулювання даними: виконання над даними операцій, які дозволять розв'язувати різні задачі;

● можливість формувати звіти;

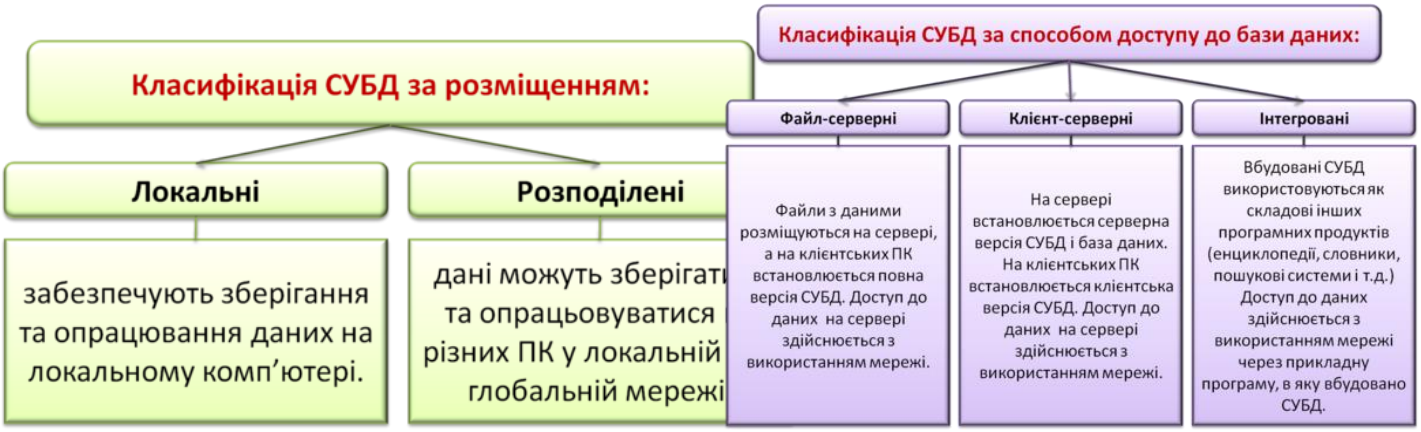
● система команд для реалізації запитів;

● діалогові засоби спілкування з користувачем.

Добре відомими є СКБД типу dbase (dbase III Plus, dbase IV, Foxbase, FoxPro). На даний час популярними є СКБД Oracle, Access, Paradox, MySQL.

2. Класифікація СУБД

3.



Технологія файл-сервер і клієнт-сервер

Розпізнають централізовані та розподілені системи БД. **Централізована БД** зберігається в пам'яті однієї обчислювальної системи, тобто розміщена на одному комп'ютері. **Розподілена БД** складається з декількох частин, розміщених на різних машинах обчислювальної мережі.

Системи централізованих БД поділяються в залежності від архітектури. Системи централізованих БД, в яких СКБД і прикладна програма розміщені на одному комп'ютері і мережна підтримка не потрібна, є застарілою. Розглянемо архітектури файл-сервер і клієнт-сервер. Цими термінами визначають перш за все архітектуру або логічне розподілення функцій. Клієнт – це прикладна програма, яку також називають прикладною програмою переднього плану (frontend), а сервер – це прикладна програма заднього плану (backend).

В архітектурі файл-сервер БД розміщується на файл-сервері (або на декількох серверах). В якості останнього може використовуватись більш потужна машина із усіх, об'єднаних в мережу. Функції файл-сервера полягають в основному в збереженні БД та забезпеченні доступу до них користувачів, які працюють на різних комп'ютерах. Файли БД у відповідності з запитами користувачів передаються на робочі станції, де в основному і виконується обробка. Передані дані оброблюються СКБД, яка знаходиться на комп'ютерах користувачів. Після того як користувачі виконають необхідні зміни, вони копіюють файли знову на файл-сервер, де інші користувачі, в

свою чергу, можуть знову їх використовувати. Явним недоліком подібного підходу є висока ймовірність втрати змін, виконаних одними користувачами, при записуванні змінених файлів на центральний сервер.

За технологією клієнт-сервер (рис. 1) припускається, що, крім зберігання БД, центральний комп’ютер (сервер БД) повинен забезпечувати виконання основного об’єму обробки даних. При технології клієнт-сервер запит на виконання операції з даними (наприклад, звичайний вибір), який видає клієнт (робоча станція), примушує сервер виконувати пошук і вилучення даних. Отримані дані, але не файли, транспортуються по мережі від серверу до клієнта.

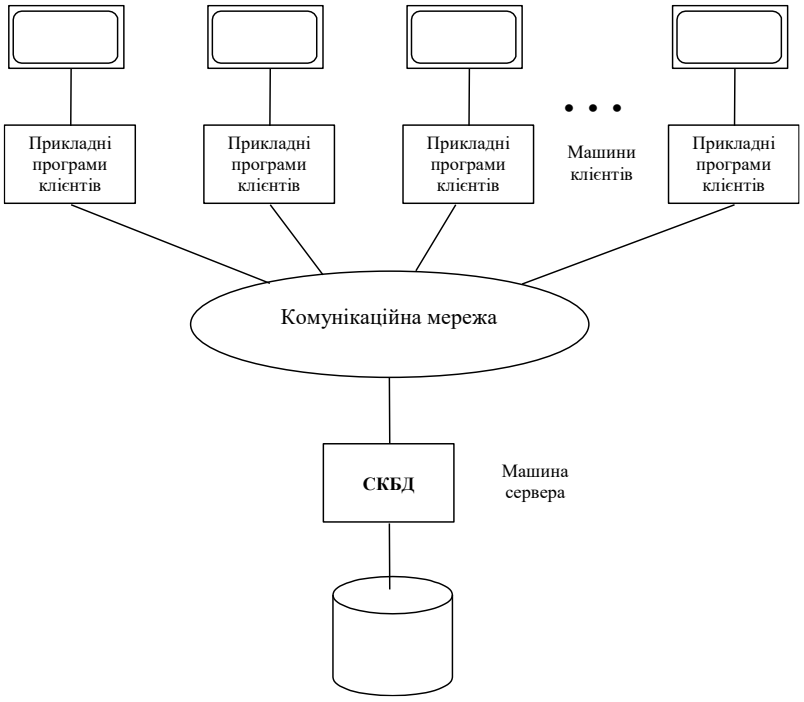


Рис. 1 Архітектура клієнт-сервер.

Архітектура клієнт-сервер, для якої призначені серверні СКБД, є деякою мірою поверненням до колишньої "мейнфреймової" моделі, заснованої на централізації збереження й обробки даних на одному виділеному комп'ютері, де функціонує спеціальна прикладна програма або сервіс, що називається сервером БД. Сервер БД відповідає за роботу з файлами БД, підтримку цілісності посилань, резервне копіювання, забезпечення авторизованого доступу до даних, протоколювання операцій і, звичайно, за виконання запитів користувача на вибір і модифікацію даних і метаданих (визначення інших об’єктів системи). Клієнтські прикладні програми, що є джерелами цих запитів, функціонують на персональних комп'ютерах у мережі. Можна відзначити, що при використанні серверних СКБД виконання запитів виробляється самим сервером, тому клієнтські прикладні програми одержують від сервера тільки результати самого запиту і не вимагають передачі всього індексу чи всієї таблиці, що істотно знижує мережний трафік при обробці запитів.

Однією з важливих переваг архітектури клієнт-сервер є зниження інтенсивності мережного трафіку при виконанні запитів. Наступна перевага полягає в можливості

зберігання бізнес-правил (наприклад, правил цілісності посилань або обмежень на значення даних) на сервері, що дозволяє уникнути дублювання коду в різних клієнтських прикладних програмах, які використовують загальну базу даних.

Сучасні серверні СКБД мають такі сервіси: реалізація для декількох платформ; зручні утиліти адміністрування; резервне копіювання даних; обслуговування реплікацій (копіювання даних в декілька інших БД); паралельна обробка даних в багатопроцесорних системах; підтримка OLAP і створення сховищ даних; виконання розподілених запитів і транзакцій; автоматизовані засоби проектування даних; підтримка Web-технологій.

У процесі роботи СКБД виникає необхідність захисту БД від випадкових та навмисних ситуацій, коли існує імовірність втрати даних. Одним із засобів вирішення цієї проблеми є механізм транзакцій. Транзакція *(Transaction)* – це логічна послідовність операцій над даними (читання, віддалення, вставки, модифікації), неподільна з точки зору впливу на БД, яка або виконується вся разом, або вся разом скасовується. Завершення *(Commit)* транзакції означає, що всі операції, що входять до складу транзакції, успішно завершені, і результат їхньої роботи збережений у базі даних. Відкат *(Rollback)* транзакції означає, що усі уже виконані операції, що входять до складу транзакції, скасовуються і всі об'єкти БД, порушені цими операціями, повернуті у вихідний стан. Для реалізації можливості відкату транзакцій деякі СКБД підтримують запис у *log*-файли, що дозволяють відновити вихідні дані при відкоті. Транзакція може складатися з декількох вкладених транзакцій. Деякі СКБД підтримують двофазне завершення транзакцій – процес, що дозволяє здійснювати транзакції над декількома БД, що відносяться до одній і тієї ж СКБД. Для підтримки розподілених транзакцій (тобто транзакцій над БД, керованих різними СКБД), існують спеціальні засоби, що звуться моніторами транзакцій.

Крім зазначених переваг сучасні серверні СКБД володіють можливістю оптимізації запитів, яка виконується спеціальним компонентом СКБД – оптимізатором запитів. В оброблювачі запитів СКБД MS SQL Server [5] реалізовано нові методи пошуку, які підвищують швидкість обробки комплексних запитів. SQL Server використовує техніку перетину та з'єднання індексів для таблиць з декількома індексами, нові методи пошуку, які підвищують швидкість обробки комплексних запитів. SQL Server підтримує паралельне виконання запитів, підтримку розподілених транзакцій, що дозволяє сполучати в один запиті віддалені сервери. Не дивлячись на це, не можна вважати, що знайдено адекватне вирішення задачі глобальної оптимізації в розподіленій СКБД.

Розподілені бази даних

Основна задача систем управління розподіленими базами даних полягає в забезпеченні засобів інтеграції локальних БД, розташованих у вузлах обчислювальної мережі, для того, щоб користувач, працюючий в будь-якому вузлі мережі, мав доступ до всіх її частин, як до єдиної БД.

Розподілена система БД (рис. 2) означає, що окрема прикладна програма може “прозоро” обробляти дані, розподілені між множиною різних БД, керування якими

здійснюють різні СКБД, які працюють на з'єднаних комунікаційними мережами машинах різних типів з різними операційними системами. Поняття "прозоро" означає, що прикладна програма виконує обробку даних з логічної точки зору так, якби управління даними повністю здійснювалося однією СКБД, яка працює на одній машині. Тобто клієнт може отримувати доступ до будь-якої кількості серверів одночасно (за один запит можливо отримати дані з двох і більше серверів). В цьому випадку сервери розглядаються клієнтом як єдиний сервер (з логічної точки зору) і користувач може не знати, на який машині знаходиться та або інша частина даних. Фундаментальний принцип розподіленої системи БД можна сформулювати так: для користувача розподілена система повинна виглядати так, як нерозподілена .

Реалізація розподілених БД переслідує наступні цілі: локальна незалежність, відсутність опори на центральний вузол, неперервне функціонування, незалежність від розташування, незалежність від фрагментації, незалежність від реплікації (фрагментарне копіювання), обробка розподілених запитів, управління розподіленими транзакціями, апаратна незалежність, незалежність від операційної системи, незалежність від мережі, незалежність від типу СКБД.

Незалежність від типу СКБД є дуже важливою вимогою. В цьому випадку звертаються за допомогою до спеціальних програм, так званих шлюзів. Робота шлюзів полягає в тому, що за рахунок їх використання одна СКБД бачить роботу іншої в зрозумілому для неї вигляді, для чого використовуються загальні протоколи обміну інформацією, типи даних (або здійснюється перетворення одних типів даних в інші) та забезпечується сумісна реалізація блокувань, виконання транзакцій і т.ін.

Для забезпечення інтеграції локальних БД потрібно вирішити проблеми декомпозиції початкового запита, оптимального засобу виконання запита, забезпечення синхронізації, знаходження та рішення розподілених тупиків, відновлення стану БД.

Традиційно виділяють однорідні і неоднорідні розподілені БД. В однорідних розподілених БД кожна локальна БД керується однією й тією ж СКБД. На відміну від цього, в неоднорідній системі локальні БД можуть відноситись до різних моделей даних, тому мережна інтеграція неоднорідних БД – це актуальна, але складна проблема. Багато рішень відомі на теоретичному рівні, але поки що не вдається подолати головну проблему – недостатню ефективність інтегрованих систем. Вирішенні цієї проблеми в значній мірі сприяє стандартизація мови SQL та загальне слідування виробників СКБД принципам відкритих систем.

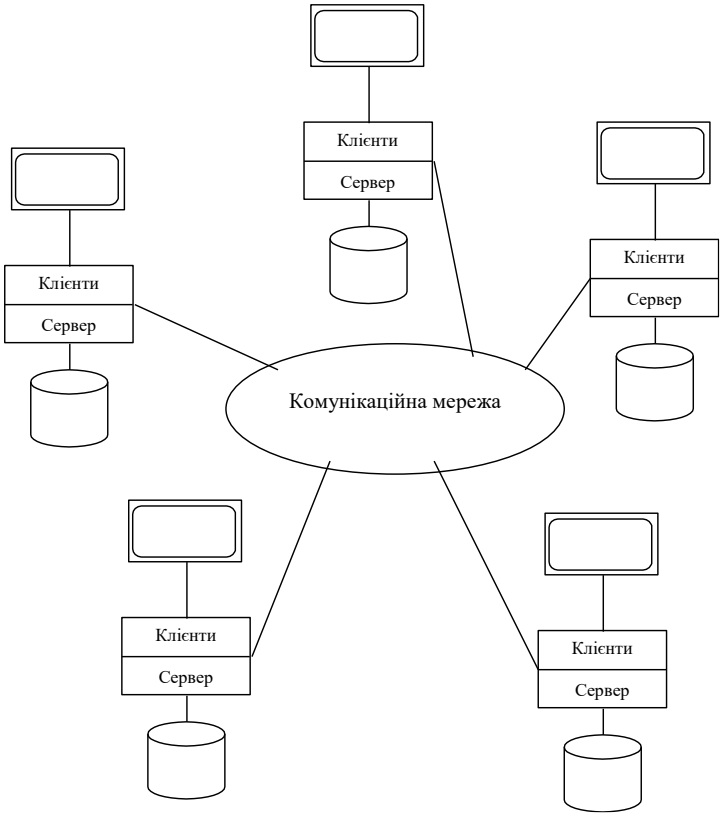


Рис. 2 Розподілена система БД.

Основною ідеєю відкритих систем є спрощення сполучення обчислювальних систем за рахунок стандартизації апаратної та програмної частини. Головною причиною розвитку концепції відкритих систем є перехід до використання локальних обчислювальних мереж та проблеми спряження апаратно-програмних засобів, які викликали цей перехід. В зв'язку з розвитком технології глобальних комунікацій відкриті системи набувають ще більше значення та масштабність.

Стосовно розподіленої СКБД архітектура клієнт-сервер цікава та актуальна головним чином тому, що вона забезпечує просте і відносно дешеве рішення проблеми колективного доступу до БД в локальній мережі. В деякому роді системи, засновані на архітектурі клієнт-сервер, є спрощеним наближенням до розподілених систем, які не вимагають рішення основного набору проблем дійсно розподілених БД. Архітектура клієнт-сервер – це окремий (поодинокий) випадок розподілених систем в цілому. Реальне розповсюдження архітектури клієнт-сервер стало можливим завдяки розвитку і широкому впровадженню в практику концепції відкритих систем.

4. Прикладні програми для проектування БД

Розглянемо класифікацію прикладних програм, які використовують БД. Почнемо їхній розгляд з прикладних програм, що працюють в архітектурі клієнт-сервер. ІС, створені в такій архітектурі, являють собою сервер БД, який маніпулює даними, і клієнтську прикладну програму, що звертається до нього і використовує для цього або клієнтські *API*, або один з універсальних механізмів доступу до даних. Звичайно при використанні такої архітектури прикладних програм на сервер БД покладається також контроль дотримання бізнесів-правил, реалізованих у виді збережених процедур,

тригерів, серверних обмежень і інших об'єктів БД. Для створення клієнтських прикладних програм у цьому випадку найчастіше застосовуються засоби розробки, що володіють розвинутими візуальними інструментами, такі як *Microsoft Visual Basic, Borland Delphi, Sybase PowerBuilder, Borland C++Builder*. В теперешній час найбільш популярним середовищем розробки є Visual Studio. Visual Studio надає засоби проектування, розробки, налагодження та розгортання веб-додатків, XML веб-служб і традиційних клієнтських додатків, а також інших типів проектів на мовах програмування Visual Basic, Visual C#, Visual C++, Visual F#. F # – це мова програмування, що забезпечує підтримку функціонального програмування, а також об'єктно-орієнтованого та імперативного (процедурного) програмування.

Слід зазначити, однак, що вибір архітектур сучасних прикладних програм не вичерпується "класичною" архітектурою клієнт-сервер, яка припускає, що прикладна програма складається із сервера БД і клієнтських прикладних програм, взаємодіючих з цим сервером. Існують так звані розподілені прикладні програми. Розподілені (або багатоланкові) прикладні програми звичайно складаються з презентаційних сервісів, сервісів бізнес-логіки, реалізованих у вигляді бізнес-об'єктів і сервісів даних (звичайно складаються із сервера БД і механізмів доступу до даних). Сервіси бізнес-логіки призначені для одержання введених користувачем даних від презентаційних сервісів, взаємодії із сервісами даних для виконання бізнес-операцій (наприклад, обробки або замовлень розрахунку бухгалтерського балансу) і повернення результатів цих операцій презентаційним сервісам. Деякі з бізнес-об'єктів можуть звертатися до сервісів даних, використовуючи ті або інші механізми доступу до даних. Оскільки кінцевий користувач не взаємодіє безпосередньо з бізнес-об'єктами, останні звичайно не мають інтерфейсу користувача у звичному розумінні. Фізично бізнес-об'єкти можуть бути реалізовані у виді сервісів операційної системи, консольних прикладних програм або *Windows*-прикладних програм, а також у вигляді бібліотек, які завантажуються в адресний простір спеціально призначеної для цієї мети серверної прикладної програми (Web-сервера, сервера прикладних програм і т. ін.). Для створення бізнес-об'єктів застосовуються як засоби розробки з розвинутими візуальними інструментами, так і засоби розробки, орієнтовані на “ручне” створення коду. Відзначимо, що новітні версії майже усіх найбільш популярних засобів розробки *Windows*-прикладних програм (*Microsoft Visual Basic, Visual FoxPro і Visual C++, Borland Delphi і C++Builder, Sybase PowerBuilder*) підтримують створення різних типів бізнес-об'єктів (*Web*-прикладних програм, *ASP*-об'єктів, *COM*-серверів і ін.), за винятком, мабуть, *Microsoft Access* – цей продукт розрахований скоріше на кваліфікованих користувачів, ніж на розроблювачів розподілених систем. Нерідко для цієї мети використовуються і засоби створення *Java*-прикладних програм (такі як *Borland JBuilder*).

Дані, які використовуються для опису предметної області, можна представити у вигляді трьохрівневої схеми (так звана модель *ANSI/SPARC*) (рис. 4).

Рис. 4 Архітектура ANSI/SPARC.

Розрізняють концептуальний, внутрішній і зовнішній рівні даних БД, яким відповідають моделі аналогічного призначення. Концептуальний рівень відповідає логічному уявленню даних предметної області в узагальненому виді. Це – ядро проектування БД. Концептуальна модель складається з логічно структурованих різних типів даних. Внутрішній рівень відображає організацію даних у середовищі збереження і відповідає фізичному уявленню даних. Внутрішня модель складається з окремих записів, фізично збережених у зовнішніх носіях. Зовнішній рівень підтримує подання даних, які необхідні конкретним користувачем. За допомогою зовнішніх моделей підтримується доступ до різних прикладних програм.

Зовнішня схема даних є сукупністю вимог до даних з боку деякої конкретної функції, виконуваної користувачем. Концептуальна схема є повною сукупністю усіх вимог до даних, отриманої з уявлень користувача про реальний світ. Внутрішня схема – це сама БД. Звідси випливають основні етапи, на які розбивається процес проектування БД для ІС.

Питання для самоперевірки:

1. Дайте визначення поняттю СКБД
2. Назвіть функції СКБД
3. Які Вам відомі СКБД?
4. Наведіть класифікацію СКБД
5. Дайте роз’яснення даній класифікації
6. Дайте роз’яснення файл-серверній організації даних, наведіть приклад схеми
7. Дайте роз’яснення клієнт-серверній організації даних, наведіть приклад схеми
8. Назвіть переваги та недоліки вище зазначених моделей

Домашнє завдання:

1) Замалюйте у зошит приклад 4 видів зв’язку між даними БД (один-до- одного, багато-до – одного, один-до-багатьох, багато-до-багаьох

2) читать підручник:

Руденко В., Потієнко В. «Інформатика 10 (11)» (рік видання 2019) §3.2
Руденко В., Потієнко В. «Інформатика 11» (рік видання 2019) стор.10