Київський національний університет імені Тараса Шевченка

Г.В. Верьовкіна

Система управління базами даних Access

Київський національний університет імені Тараса Шевченка

Г.В. Верьовкіна

Навчальний посібник з дисципліни "СУБД"

Система управління базами даних Access

для студентів механіко-математичного факультету,
які навчаються за освітнім рівнем "Бакалавр"
спеціальність "Математика"

Київ 2022

Рецензенти:

Є.С. Вакал кандидат фіз.-мат. наук, доцент

О.М. Харитонов кандидат фіз.-мат. наук, доцент

Навчальний посібник рекомендовано до друку вченою радою механіко-математичного факультету Київського національного університету імені Тараса Шевченка

протокол № 2 від 15.09.2022

Вступ

За навчальним планом студенти механіко-математичного факультету, які навчаються за освітнім рівнем "Бакалавр" спеціальність "Математика" вивчають курс "СУБД". Даний навчальний посібник складений для підготовки і проведення практичних та лабораторних занять з курсу та розрахований для студентів і викладачів. Під час підготовки використано багаторічний досвід викладання курсів "СУБД" (Системи управління базами даних) та "Бази даних" на механіко-математичному факультеті Київського національного університету імені Тараса Шевченка.

Навчальний посібник "Система управління базами даних Access" містить матеріал, який викладається протягом першого модуля вивчення курсу "СУБД". Весь матеріал розбито на окремі теми. До кожної теми запропоновано огляд теоретичного матеріалу та наведено приклади застосування з включенням графічної інтерпретації інтерфейсу прикладної програми Access.

Для підготовки навчального посібника авторка використовувала літературу [1–4], список якої дано в кінці видання.

Тема 1. Бази даних. СУБД (СКБД).

Що таке база даних (БД).

База даних (database) – сукупність даних, організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами; ця сукупність підтримує щонайменше одну з областей застосування. В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти. Дані у базі організовують відповідно до моделі організації даних. Таким чином, сучасна база даних, крім саме даних, містить їх опис та може містити засоби для їх обробки.

1960-ті рр. розроблення перших БД. CODASYL — мережева модель даних та одночасно незалежне розроблення ієрархічної БД фірмою North American Rockwell, яка пізніше взята за основу IMS — власної розробки IBM.

1970-ті рр. наукове обґрунтування Едгаром Ф. Коддом основ реляційної моделі, котра на початку зацікавила лише наукові кола. Уперше цю модель було використано у БД Ingres (Берклі) та System R (IBM), що були лише дослідними прототипами, анонсованими протягом 1976 року.

1980-ті рр. поява перших комерційних версій реляційних БД Oracle та DB2. Реляційні БД починають успішно витісняти мережеві та ієрархічні. Дослідження децентралізованих (розподілених) систем БД, проте вони не відіграють особливої ролі на ринку БД.

1990-ті рр. увага науковців спрямовується на об'єктно-орієнтовані БД, які знайшли застосування в першу чергу в тих галузях, де використовуються комплексні дані: інженерні, мультимедійні БД.

2000-ні рр. головним нововведенням є підтримка та застосування XML у БД. Розробники комерційних БД, які панували на ринку у

1990-их рр., отримують все більшу конкуренцію з боку руху відкритого програмного забезпечення. Реакцією на це стає поява безкоштовних версій комерційних БД.

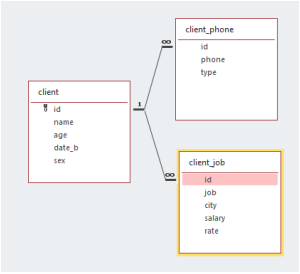
Види баз даних.

Бази даних класифікують за різними критеріями.

За моделлю організації даних розрізняють такі бази даних:

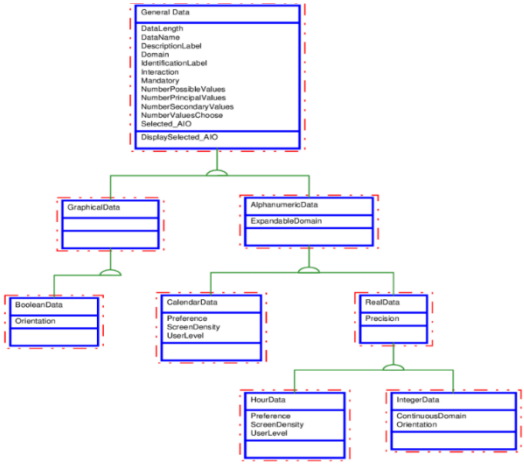
Ієрархічна. Ієрархічна база даних може бути представлена як дерево, що складається з об'єктів різних рівнів. Між об'єктами існують зв'язки «предок-нащадок». При цьому можлива ситуація, коли об'єкт не має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок.

Мережева. Така база даних подібна до ієрархічної, за винятком того, що кожен об'єкт може мати більше одного предку.



Реляційна. Реляційна база даних зберігає дані у вигляді таблиць-відношень, встановлюючи зв’язки між ними. Найуживаніші СУБД використовують реляційну модель даних.

Об'єктно-орієнтована. У базі даних цього виду дані оформляють у вигляді моделей об'єктів.



Об'єктно-реляційна. Має спільні риси з двома попередніми моделями.

Функціональна. Використовуються для вирішення аналітичних задач фінансового моделювання та управління продуктивністю. Підтримує інтерактивні обчислення: значення залежних клітинок автоматично оновлюються, коли змінюється значення клітинки.

Модель «сутність-зв'язок».

Модель «сутність-зв'язок» або ER-модель (англ. *Entity-relationship model або entity-relationship diagram*) — модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків. ER-модель — це мета-модель даних, тобто засіб опису моделей даних. Існує ряд моделей для представлення знань, але одним з найзручніших інструментів уніфікованого представлення даних, незалежного від програмного забезпечення, що його реалізує, є модель «сутність-зв'язок». Важливим є той факт, що з моделі «сутність-зв'язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об'єктна), тому вона є найзагальнішою.

За розміщенням даних виділяють такі види баз:

Локальна, або централізована. Така база даних підтримується на одному комп'ютері.

Розподілена. Частини такої бази даних розміщують на різних комп'ютерах мережі.

За технологією фізичного зберігання виділяють:

БД у вторинній пам'яті (традиційні) – використовують жорсткий диск.

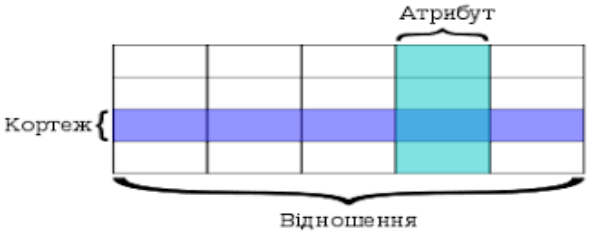
БД в оперативній пам'яті (in-memory database).

БД у третинній пам'яті (tertiary database) – середовищем постійного зберігання є від'єднаний від сервера пристрій масового зберігання.

Реляційна модель.

Реляційна модель даних — логічна модель даних. Вперше була запропонована британським ученим співробітником компанії IBM Едгаром Франком Коддом (E. F. Codd) в 1970 році в статті «A Relational Model of Data for Large Shared Data Banks». В даний час ця модель є фактичним стандартом, на який орієнтуються практично всі сучасні комерційні системи керування базами даних (СУБД).

Структурою даних є нормалізоване n-арне відношення. **Відношення** зручно представляти у формі таблиць, де кожен рядок є **кортеж**, а кожен стовпець — **атрибут**, визначений на деякому домені. Даний неформальний підхід до поняття відношення дає більш звичну для розробників і користувачів форму представлення, де реляційна база даних подається як кінцевий набір таблиць.



Будь-який кортеж будь-якого відношення відмінний від будь-якого іншого кортежу цього відношення, тобто іншими словами, будь-яке відношення має володіти первинним ключем. Вимога цілісності щодо посилань, або вимога зовнішнього ключа полягає в тому, що для кожного значення зовнішнього ключа, що з'являється у відношенні, на яке веде посилання, повинен знайтися кортеж з таким же значенням первинного ключа, або значення зовнішнього ключа повинно бути невизначеним (тобто ні на що не вказувати).

Можна провести аналогію між елементами реляційної моделі даних і елементами моделі «сутність-зв'язок». Реляційні відносини відповідають наборам зв’язків, а кортежі — сутностям. Тому, як і в моделі «сутність-зв'язок», стовпці в таблиці, що представляє реляційне відношення, називають атрибутами.

Кожен атрибут визначений на домені, тому домен можна розглядати як множина допустимих значень даного атрибуту. Кілька атрибутів одних відношень і навіть атрибути різних відношень можуть бути визначені на одному і тому ж домені.

Іменована множина пар «ім'я атрибута — ім'я домену» називається схемою відношення. Потужність цієї множини — називають ступенем чи «арністю» відносини. Набір іменованих схем відносин називають схемою бази даних.

Атрибут, значення якого однозначно ідентифікує кортежі, називається ключовим (або просто ключем). Якщо кортежі ідентифікуються тільки зчепленням значень декількох атрибутів, то говорять, що відношення має складений ключ. Відношення може містити кілька ключів. Завжди один із ключів оголошується первинним, його значення не можуть оновлюватися. Всі інші ключі відношення називають можливими ключами.

Переваги реляційної моделі:

простота і доступність для розуміння користувачем. Єдиною використовуваною інформаційною конструкцією є «таблиця»;

суворі правила проектування, які базуються на математичному апараті;

повна незалежність даних. Зміни в прикладній програмі при зміні реляційної БД мінімальні;

для організації запитів і написання прикладного ПЗ немає необхідності знати конкретну організацію БД у зовнішній пам'яті.

Недоліки реляційної моделі:

далеко не завжди предметна область може бути представлена у вигляді «таблиць»;

в результаті логічного проектування з'являється множина «таблиць». Це призводить до труднощів розуміння структури даних;

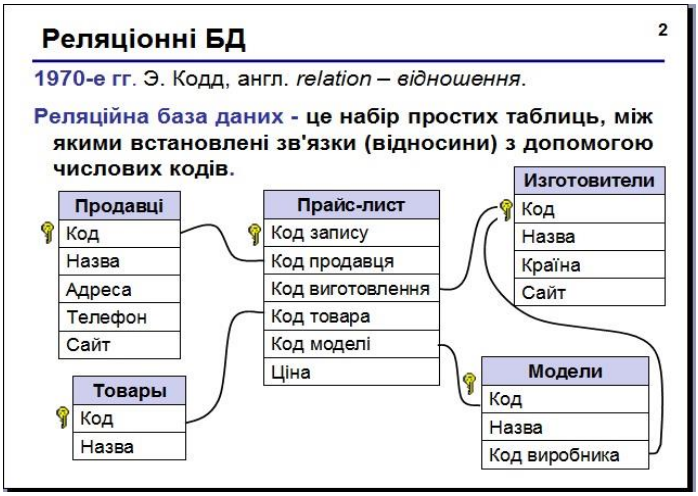
БД займає відносно багато зовнішньої пам'яті;

відносно низька швидкість доступу до даних.

Реляційна БД.

Реляційна база даних — база даних, заснована на реляційній моделі даних.

Слово «реляційний» походить від англ. relation. Для роботи з реляційними БД застосовують реляційні СУБД. Інакше кажучи, реляційна база даних — це база даних, яка сприймається користувачем як набір нормалізованих відношень різного ступеня. Реляційна база даних є сукупністю елементів даних, організованих у вигляді набору формально описаних таблиць, з яких дані можуть бути доступними або повторно зібрані багатьма різними способами без необхідності реорганізації таблиць бази даних.



Принципи нормалізації:

- в кожній таблиці БД не повинно бути повторюваних полів;
- в кожній таблиці повинен бути унікальний ідентифікатор (первинний ключ);
- кожному значенню первинного ключа повинна відповідати достатня інформація про тип суті або про об’єкт таблиці (наприклад, інформація про успішність, про групу або студентах);
- зміна значень в полях таблиці не повинна впливати на інформацію в інших полях (крім змін у полях ключа).

Види логічного зв’язку.

Зв’язок встановлюється між двома загальними полями (стовпцями) двох таблиць. Існують зв’язки з відношенням «один-до-одного», «один-до-багатьох» і «багато-до-багатьох».

Відносини, які можуть існувати між записами двох таблиць:

один — до — одного — кожному запису з однієї таблиці відповідає один запис у іншій таблиці;

один — до — багатьох — кожному запису з однієї таблиці відповідає кілька записів у іншій таблиці;

багато — до — одного — безлічі записів з однієї таблиці відповідає один запис у іншій таблиці;

багато — до — багатьох — безлічі записів з однієї таблиці відповідає кілька записів в іншій таблиці.

Відношення «один-до-багатьох» створюється в тому випадку, коли тільки одне з полів є полем первинного ключа або унікального індексу.

Відношення «один-до-одного» створюється в тому випадку, коли обидва поля є ключовими або мають унікальні індекси.

Відношення «багато-до-багатьох» фактично є двома відносинами «один-до-багатьох» з третьої таблицею, первинний ключ якої складається з полів зовнішнього ключа двох інших таблиць.

Ключі.

Ключ — це стовпець (може бути декілька стовпців), що додається до таблиці і дозволяє встановити зв’язок із записами в іншій таблиці.

Існують ключі двох типів: первинні і вторинні (зовнішні).

Первинний ключ — це одне або кілька полів (стовпців), комбінація значень яких однозначно визначає кожний запис у таблиці. Первинний ключ не допускає значень Null і завжди повинен мати унікальний індекс. Первинний ключ використовується для зв’язування таблиці з зовнішніми ключами в інших таблицях.

Зовнішній (вторинний) ключ — це одне або кілька полів (стовпців) у таблиці, що містять посилання на поле або поля первинного ключа в іншій таблиці. Зовнішній ключ визначає спосіб об’єднання таблиць.

З двох логічно пов’язаних таблиць одну називають таблицею первинного ключа або головною таблицею, а іншу таблицею вторинного (зовнішнього) ключа або підпорядкованою таблицею. СУБД дозволяють зіставити споріднені записи з обох таблиць і спільно вивести їх у формі, звіті або запиті.

Існує три типи первинних ключів: ключові поля лічильника (лічильник), простий ключ і складовий ключ.

Поле лічильника (Тип даних «Лічильник»). Для кожного запису цього поля таблиці автоматично заноситься унікальне числове значення.

Простий ключ. Якщо поле містить унікальні значення, такі як коди чи інвентарні номери, то це поле можна визначити як первинний ключ. В якості ключа можна визначити всі поля, що містить дані, якщо це поле не містить повторювані значення або значення Null.

Складені ключі. У випадках, коли неможливо гарантувати унікальність значень кожного поля, існує можливість створити ключ, що складається з декількох полів. Найчастіше така ситуація виникає для таблиці, використовуваної для зв’язування двох таблиць відношенням «багато — до — багатьох».

СУБД (СКБД).

У сучасних інформаційних системах для забезпечення роботи з базами даних використовують системи управління (керування) базами даних СУБД (СКБД). **Система управління або керування базами даних** — це система, заснована на програмних та технічних засобах, яка забезпечує визначення, створення, маніпулювання, контроль, керування та використання баз даних (за стандартом ISO/IEC 2382:2015[2]). Застосунки для роботи з базою даних можуть бути частиною СУБД або автономними. Найпопулярнішими СУБД є MySQL, PostgreSQL, Microsoft SQL Server, Oracle, Sybase, Interbase, Firebird та IBM DB2. СУБД дозволяють ефективно працювати з базами даних, обсяг яких робить неможливим їх ручне опрацювання.

Через тісний зв'язок баз даних з СУБД під терміном «база даних» інколи необґрунтовано та неточно мають на увазі систему керування базами даних. Але варто розрізняти базу даних — сховище даних, та СУБД — засоби для роботи з базою даних. СУБД з інформаційної системи може бути видалена, але база даних продовжить існувати. І навпаки: СУБД може функціонувати без жодної бази даних.

В загальному базу даних неможливо просто перемістити з однієї СУБД до іншої. Але **СУБД** використовують стандарти (**SQL, ODBC, JDBC**), які уніфікують ряд операцій по роботі з даними і дозволяють різним застосункам працювати з базами даних різних СУБД. СУБД часто класифікують за моделлю організації даних. Найуживаніші СУБД використовують реляційну модель, у якій дані подають у виді таблиць. Для кінцевого користувача (та прикладних програм) робота з базою даних напряму неможлива. Всі маніпуляції над даними здійснюють через спеціальні запити, які надсилають до СУБД. СУБД опрацьовує їх і повертає результат. Безпосередньо з базою даних працює виключно СУБД.

Сучасні **СУБД** забезпечують функції щодо керування даними, які можна поділити на такі групи:

Оголошення даних — створення, зміна та видалення визначень, які описують організацію даних.

Модифікація даних — додавання даних, їх редагування та видалення.

Отримання даних — надання даних за запитом застосунку у формі, яка дозволяє їх безпосереднє використання. Дані можуть надаватись або у формі, в якій вони зберігаються у базі даних, або в іншій формі (наприклад, через поєднання різних даних).

Адміністрування даних — реєстрування та відслідковування дій користувачів, дотримання безпеки роботи з даними, забезпечення надійності та цілісності даних, моніторинг

продуктивності, резервне копіювання та відновлення даних тощо.

Реалізації СУБД.

Комерційні СУБД

Microsoft SQL Server, Oracle, DB2, Interbase, Informix, Sybase, Дінай

СУБД з відкритим кодом

MySQL, PostgreSQL, Firebird, SQLite

Тема 2. База даних Access. Таблиці.

Microsoft Office Access — система управління базами даних, програма, що входить до складу пакету офісних програм Microsoft Office. Має широкий спектр функцій, включаючи зв'язані запити, сортування по різних полях, зв'язок із зовнішніми таблицями і базами даних. Завдяки вбудованій мові VBA, в самому Access можна писати підпрограми.

Основні компоненти MS Access:

- конструктор таблиць;
- конструктор екранних форм;
- конструктор SQL-запитів (мова SQL в MS Access не відповідає стандарту ANSI);
- конструктор звітів, що виводяться на друк;
- макроси і код

Таблиця — це основний об'єкт бази даних, призначений для збереження даних, документів та інших облікових записів.

Запит — вибирає дані з таблиць згідно з умовами, що задаються.

Форма — відображає дані з таблиць або запитів відповідно до форматів, описаних користувачем. Форма дозволяє переглядати, редагувати та друкувати дані.

Звіт — відображає і друкує дані з таблиць або запитів згідно з описаним користувачем форматом. У звіті дані редагувати не можна.

Макроси і Код –інструменти, за допомогою яких можна автоматизувати завдання та розширити можливості форм, звітів і елементів керування без та з написанням коду в модулі Visual Basic for Applications (VBA).

Створення бази даних.

Коли ви відкриєте Access, у поданні Backstage з'явиться вкладка Створити. На вкладці Створити доступні перелічені нижче команди для створення бази даних.

Пуста база даних. Створити базу даних можна з нуля. Це зручно, якщо ви маєте специфічні вимоги до структури бази даних або коли наявні дані потрібно розмістити в базі даних чи вбудувати їх у неї.

Шаблон, який інсталюється разом з Access. Скористайтеся шаблоном, щоб дати гарний початок новому проекту. Access постачається з кількома стандартними шаблонами.

Шаблон із Office.com. Крім стандартних шаблонів Access, на порталі Office.com можна знайти чимало інших шаблонів. Щоб скористатися ними, не потрібно відкривати браузер, оскільки вони вже доступні на вкладці Створити

Збереження в Access.

Access, при роботі з базою даних, інакше взаємодіє з жорстким (або гнучким) диском, ніж інші програми. В інших програмах, файл-документ, при відкритті, повністю завантажується в оперативну пам'ять, і нова редакція цього файлу (змінений файл) цілком записується на диск тільки при натисканні кнопки «зберегти». В Access нова редакція вмісту зміненої комірки таблиці записується на диск (зберігається) відразу, як тільки курсор клавіатури буде поміщений в іншу комірку. Цілісність

даних в Access забезпечується також за рахунок механізму транзакцій.

Кнопка «Зберегти» в Access для збереження таких змін, як:

- зміна ширини стовпців і висоти рядків,
- перестановка стовпців в режимі перегляду даних, «закріплення» стовпців і звільнення закріплених стовпців,
- зміна сортування, застосування нового фільтра,
- зміна шрифту; кольору тексту, сітки і тла
- в режимі «Конструктор» для збереження змін структури об'єкта бази даних, зроблених в цьому режимі.

Основні компоненти інтерфейсу користувача Access:

Стрічка – це смуги вкладок у верхній частині вікна програми, що містить груп команд.

Подання Backstage – Це набір команд, які відображаються на вкладці «файл» на стрічці.

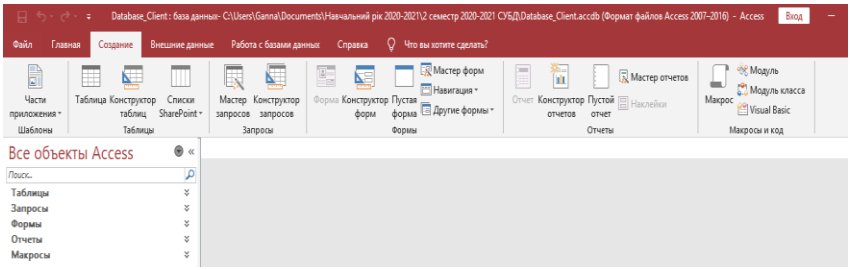
Область переходів – Це області в лівій частині вікна програми Access, що дасть змогу працювати з об'єктами бази даних.

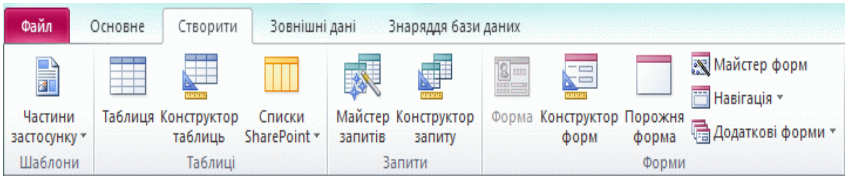
Стрічка.

Стрічка основною заміною меню та панелей інструментів і забезпечує інтерфейс основної команди у програмі Access. Одна з основних переваг програми на стрічці є що об'єднує, в одному місці, цих завдань або запис точок, що використовуються для меню, панелі інструментів, області завдань, а також інші компоненти інтерфейсу користувача для відображення. Таким чином, у вас є одне місце, в якому для пошуку команд, замість кількість розрядів.

Під час відкриття бази даних, на стрічці відображається у верхній частині головного вікна Access, де відображається команди на вкладці активні команди.

Стрічка містить ряд вкладки команд, які містять команди. У програмі Access основної команди вкладок – це **Файл, Головна, Створити, Зовнішні дані, Знаряддя бази даних.** Кожної вкладки містить групи спорідненими командами, але ці групи поверхні деякі додаткові новий інтерфейс користувача елементи, як новий тип елемента керування, який представляє варіанти візуально колекції.

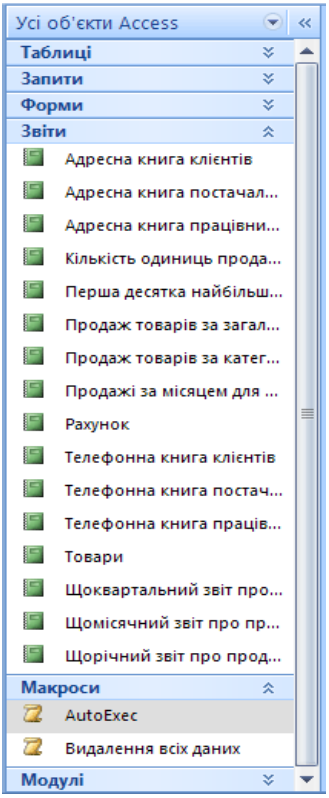




Подання Backstage.

Подання Backstage містить команди та відомості, які застосовуються до всієї бази даних, наприклад стискання й відновлення, а також команди, які були в меню "файл" у попередніх версіях, наприклад, Друк.

Область переходів.



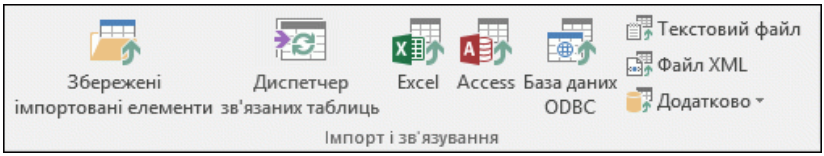
Область переходів дає змогу впорядкування об'єктів бази даних а є основним способом відкриття або змінення оформлення об'єкта бази даних.

Область переходів упорядковані за категоріями й групами. Можна вибрати різні параметри організації та можна також створити власний настроюваний організації схеми в області переходів. За промовчанням нової бази даних використовується тип об'єкта категорію, яка містить групи, які відповідають різні види об'єкта бази даних. Тип об'єкта категорії організовують бази даних, об'єкти аналогічним чином вікна бази даних за промовчанням відображення з попередніх версій.

Можна згорнути область переходів, а також можна приховати її, але не можна приховувати область переходів, відкривши об'єктів у базі даних перед його.

Створення Таблиць.

- На вкладці **Створити** у групі **Таблиці** натисніть кнопку **Таблиця**.
- На вкладці **Створити** у групі **Таблиці** натисніть кнопку **Конструктор Таблиць**.
- На вкладці **Створення** в групі **Таблиці** натисніть кнопку **Списки SharePoint**.
- На вкладці **Зовнішні дані** в групі **Імпорт і зв'язування** виберіть одне з доступних джерел даних.



Типи даних Access.

Тип даних	Використання	Розмір
Короткий текст (колишня назва "Текст")	Буквено-цифрові дані (імена, заголовки тощо).	До 255 символів.
Довгий текст (колишня назва Memo)	Великі обсяги буквено-цифрових даних (речення та абзаци).	До 1 ГБ, але елементи керування, у яких може відображатися довгий текст, вміщуватимуть лише перші 64 000 символів.
Число	Числові дані.	1, 2, 4, 8 або 16 байт.
Велике число	Числові дані.	8 байтів.
Дата й час	Дати й години.	8 байтів.
Грошова одиниця	Грошові дані з 4 знаками після коми.	8 байтів.

Тип даних	Використання	Розмір
Автонумерація	Унікальне значення, яке програма Access створює для кожного нового запису.	4 байти (16 байт для ідентифікатора реплікації).
Так/Ні	Логічні значення ("Істина" або "Хибність"). В Access числове значення –1 відповідає значенню "Істина", а 0 – "Хибність".	1 байт.
Об’єкт OLE;	Зображення, графічні об’єкти або інші об’єкти ActiveX з інших програм на платформі Windows.	До 2 ГБ.
Гіперпосилання	Адреса посилання на документ або файл в Інтернеті, інтрамережі, локальній мережі або на локальному комп’ютері.	До 8192 символів (кожна частина типу даних "Гіперпосилання" може містити до 2048 символів).

Тип даних	Використання	Розмір
Вкладення	Можна вкладати такі файли, як зображення, документи, електронні таблиці або діаграми; може містити необмежену кількість вкладень на один запис до граничного обсягу сховища файлу бази даних.	До 2 ГБ.
Обчислюваний	Можна створити вираз, який використовує дані з одного або декількох полів. На основі виразу можна призначити різні типи даних результату. Зверніть увагу, що файли у форматі MDB не підтримують тип даних "Обчислюваний".	Залежить від типу даних властивості "Тип результату". Результат "Короткий текст" до 243 символів. Результат "Довгий текст", "Число", "Так/Ні" та "Дата й час" має відповідати вимогам для свого типу даних.
Майстер підстановок	Елемент "Майстер підстановок", доступний у	Залежить від типу даних поля підстановки.

Тип даних	Використання	Розмір
	стовпці "Тип даних" у режимі конструктора – це не тип даних. Допоможе визначити просте або складне поле підстановки. Просте поле використовує вміст іншої таблиці або список значень для перевірки існування одного значення в кожному рядку. У складному полі підстановки можна зберігати кілька значень одного типу даних у кожному рядку.	

Збереження таблиці.

Створивши або змінивши таблицю, потрібно зберегти її макет. Якщо ви зберігаєте таблицю вперше, дайте їй ім’я, що описує дані, які вона містить. Можна використовувати до 64 букв і цифр, включно з пробілами. Access пропонує широкі можливості іменування таблиць.

Обмеження, які потрібно врахувати. Ім’я таблиці може містити не більше 64 символів. Воно може складатися з будь-якого поєднання букв, чисел, пробілів і спеціальних символів, за винятком крапки (.), знаку оклику (!), квадратних дужок ([]), початкового пробілу, знаку рівності (=) і недрукованих символів,

наприклад символу повернення каретки. Ім’я також не може містити такі символи: ` / \ : ; * ? " ' < > | # <TAB> { } % ~ &.

Ключі.

У таблиці бажано налаштувати **первинний ключ**. Проте якщо ви маєте певну причину не робити цього, його можна не встановлювати. Програма Access автоматично створює індекс для первинного ключа, який може підвищити продуктивність бази даних, а також стежить за тим, щоб кожен запис мав унікальне значення в полі первинного ключа.

Коли ви створюєте нову таблицю у вікні табличного подання даних, програма Access автоматично створює первинний ключ і призначає йому ім’я поля ідентифікатора та тип даних "Автонумерація".

У режимі конструктора можна змінити або видалити первинний ключ, а також установити первинний ключ для таблиці, яка ще не має його.

Для первинного ключа такі вимоги:

- Кожен запис має унікальне значення для поля або комбінації полів.
- Поле або комбінація полів не можуть бути пусті або нульові – вони завжди мають значення.
- Ці значення незмінні.

На вкладці **Конструктор** у групі **Знаряддя** елемент **Первинний ключ**.

Будь-яке поле, якому бракує однієї або кількох характеристик правильного первинного ключа, – поганий вибір для первинного ключа. Ось кілька прикладів, які небажано використовувати як первинні ключі для таблиці з поясненням причин.

Неправильний первинний ключ	**Причина**
Особисте ім’я	Може бути недостатньо унікальним і може змінюватися.
Телефон	Може змінюватися.
Адреса електронної пошти	Може змінюватися.
Поштовий індекс	Кілька осіб можуть мати один і той самий індекс.
Поєднання фактів і чисел	Факти можуть змінюватися, що робить обслуговування обтяжливим. Це може призвести до непорозуміння, якщо факти дублюються в іншому полі. Наприклад, комбінація з назви міста та збільшуваного числа (наприклад, NEWYORK0579) може бути невдалим вибором, якщо назва міста також зберігається як поле.
Номери соціального страхування	Це приватні відомості, які заборонено використовувати в урядових установах і деяких організаціях. У деяких людей немає номера соціального страхування В однієї особи їх може бути кілька протягом життя

Установлення або змінення первинного ключа:

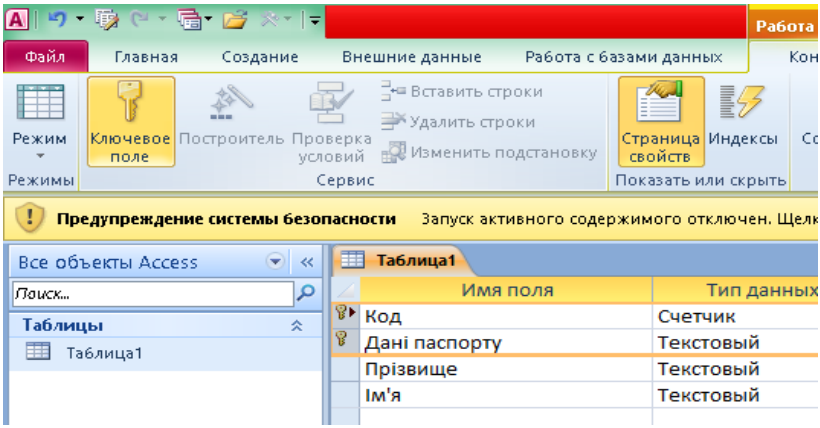
1. Виберіть таблицю, первинний ключ якої потрібно задати або змінити.

2. На вкладці **Головне** у групі **Подання** натисніть кнопку **Вигляд** і виберіть пункт **Конструктор**.

3. На бланку таблиці виберіть поле або поля, які потрібно використовувати як первинний ключ.

4. На вкладці **Конструктор** у групі **Знаряддя** виберіть елемент **Первинний ключ**.

Щоб вибрати одне поле, клацніть маркер виділення для потрібного поля. Щоб вибрати декілька полів, натиснути Ctrl та виділити потрібні поля. Потім позначити **Первинний ключ**.



Для встановлення логічного зв’язку між таблицями визначають **зовнішній ключ**.

Цілісність БД. Зв’язки. Схема БД.

Правила цілісності бази даних полягають в визначенні ключів, обмежень та зв’язків.

Первинний ключ (PRIMARY KEY) використовується для ідентифікації рядків таблиці, в нього є певні властивості:

- В одній таблиці БД може бути тільки один PK.
- Рядки, які мають це обмеження, не можуть мати повторювані або не визначені дані.

- Призначений для декількох стовпців (складений ключ) ставить унікальність комбінацій відповідних значень, хоча окреме значення в кожному стовпці складеного ключа не обов'язково має бути унікальним.

Зовнішній ключ (FOREIGN KEY) використовується для логічного зв'язку двох таблиць, в нього є певні властивості:

- Поле у головній таблиці, на яке буде адресуватися FG має бути оголошене первинним ключем.
- Поле у головній таблиці і поле з якої відбувається адресація повинні мати один й той самий тип даних. Виключення можуть стосуватися типу Лічильник.
- За допомогою FG можна реалізувати каскадне видалення і оновлення даних.

В Access існує три типи зв’язків між таблицями.

Зв’язок "один-до-багатьох". Щоб представити зв’язок "один-до-багатьох" у структурі власної бази даних, візьміть первинний ключ на стороні зв’язку "один" і вставте його як додаткове поле або поля в таблицю на стороні зв’язку "багато" – зовнішній ключ.

Зв’язок "багато-до-багатьох". Для представлення зв’язку "багато-до-багатьох" потрібно створити третю таблицю, яку часто називають розподільною, щоб розділити зв’язок "багато-до-багатьох" на два зв’язки "один-до-багатьох". Первинний ключ із кожної із двох таблиць потрібно вставити у третю таблицю. Унаслідок цього у третій таблиці буде записано всі випадки або екземпляри зв’язків.

Зв’язок "один-до-одного". У зв’язку "один-до-одного" кожному запису в першій таблиці може відповідати лише один запис у другій таблиці, а кожному запису у другій таблиці може відповідати лише один запис у першій таблиці. Цей зв’язок не дуже поширений, оскільки зазвичай відомості, пов’язані між собою в такий спосіб, зберігаються в одній таблиці. Зв’язок "один-до-одного" можна використовувати для розділення таблиці з великою кількістю полів, для відокремлення частини таблиці з

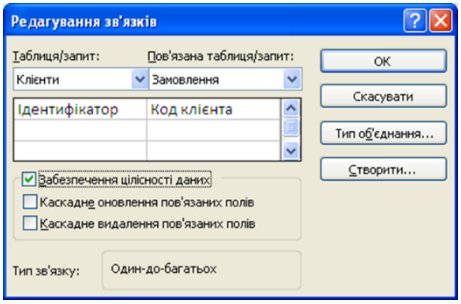
міркувань безпеки або для зберігання даних, які застосовуються лише до підмножини головної таблиці. У разі визначення такого зв'язку в обох таблицях мають бути спільні поля.

Зв'язки між таблицями – це основа для забезпечення **цілісності даних**, що дає змогу запобігти виникненню в базі даних відірваних записів. Відірваний запис – це запис із посиланням на інший запис, якого не існує, наприклад запис про замовлення, який посилається на відсутній запис про клієнта.

Під час створення бази даних усі відомості поділяються на таблиці, кожна з яких має первинний ключ. Потім до пов'язаних таблиць можна додати зовнішні ключі, які посилатимуться на первинні ключі. Такі пари "зовнішній ключ – первинний ключ" створюють основу для зв'язків між таблицями та багато табличних запитів. Тому важливо, щоб посилання "зовнішній ключ – первинний ключ" постійно синхронізувалися. Цілісність даних залежить від зв'язків між таблицями та допомагає забезпечити синхронізацію посилань.

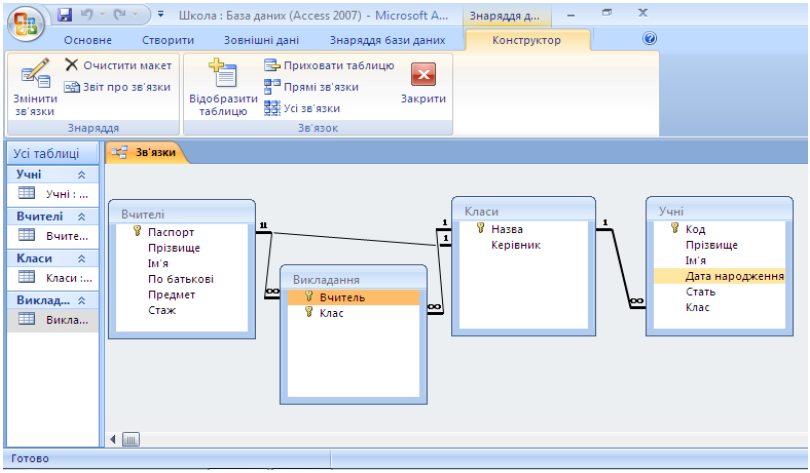
Визначення зв'язків відбувається у Схемі БД.

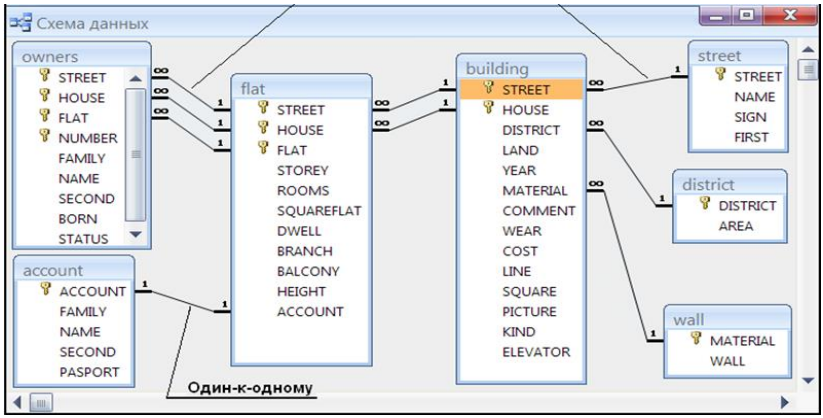
На вкладці **Робота з базами даних** у групі **Схема бази даних** створення зв'язків між таблицями та визначення Схеми БД.



На вкладці **Робота з базами даних** у групі **Схема бази даних** елемент **Редагувати відношення.** Визначення властивостей зв'язків з або **без Забезпечення цілісності даних**. Відповідні характеристики визначаються встановленням прапорців в полях **Редагування зв'язків.**

Приклади схем БД:





Тема 3. Запити. Вибірковий запит.

Запит.

Запит — спеціальний об'єкт, призначений для вибірки даних з таблиць бази, а також для виконання обчислень та інших операцій з базовими таблицями, включаючи їхнє перетворення.

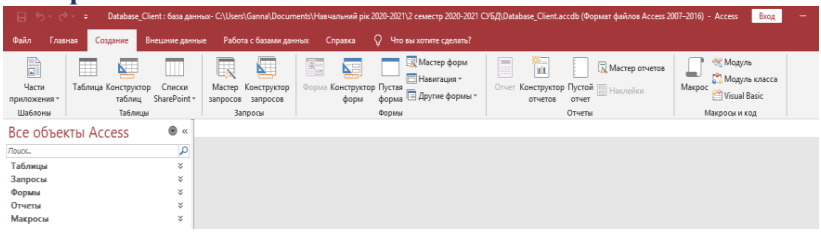
На відміну від реальної таблиці, цей набір записів реально не існує в базі даних. У результаті запиту утвориться таблиця. Інформація є актуальною, відбиває останні зміни даних. Запит полегшує роботу, якщо необхідно переглянути, додати, змінити або видалити дані в базі даних Access.

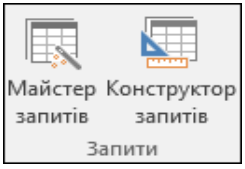
Використання запитів. Види запитів.

Типи	Використання
Вибір	Отримання даних із таблиці або виконання розрахунків.
Дія	Додавання, змінення або видалення даних. Кожне завдання має певний тип запиту на змінення.

- **Вибір – запити на вибірку (вибіркові запити)**; запити на вибірку з полем, що обчислюється; підсумкові запити; запити з параметром; перехресні запити;
- **Дія – запити на зміну** (активні запити) – на створення таблиці, на додавання, на оновлення, на видалення.

Створення>Запити





Основа процесу створення запиту – використання режиму **Майстер запитів** або **Конструктор запитів.**

Майстер запитів

Запит можна створити в режимі **Майстер запитів**, що автоматизую певні процеси.

- **Новий запи**т вибирати **Майстер простих запитів.**

Запит на вибірку по вибраних полях таблиці або зв'язних таблиць з можливістю визначення параметрів зведення інформації.

- **Новий запит** вибрати **Майстер перехресних запитів.**

Перехресний запит – це тип вибіркового запиту. Якщо виконати перехресний запит, результати відображаються в таблиці даних з особливою структурою. Цей вибірковий запит групує зведені дані по стовпцях, а також може відображати ті самі дані, але групує їх одночасно по строках та по стовпцях.

- **Новий запит** вибрати **Майстер пошуку повторюваних записів.**

Дані можуть повторюватися, коли кілька користувачів одночасно додають їх до бази даних Access або коли база даних не підтримує перевірку повторень. Повторювані дані можуть з'являтись у вигляді кількох таблиць із тими самими даними або двох записів, що містять поля (стовпці) з однаковими даними. Знайшовши повторювані записи, їх можна змінити або видалити за допомогою запиту.

- **Новий запит** вибрати **Пошук записів, які не мають підлеглих**.

Потрібно порівняти дві таблиці і знайти в одній з них записи, які не мають відповідних (підлеглих) в іншій таблиці.

Конструктор запитів.

Запит можна створити в особистому режимі в поданні **Конструктора запитів**. Використовуючи подання конструктора, ви маєте більше можливостей вибирати деталі структури запиту,

але при цьому легше припуститися помилок у структурі, а процес займає більше часу, ніж у майстрі.

Створення запиту:

1. Додавання джерел даних

2. Об’єднання пов’язаних джерел даних

3. Додавання полів виводу

4. Установлення критеріїв

Можна обмежити записи, які повертає запит, за допомогою критеріїв. При цьому перевіряється, чи відповідають значення полів указаним критеріям (умовам відбору). Установлення критеріїв для поля виводу відбувається у рядку бланка Критерії для поля, що містить значення, які потрібно обмежити – введіть вираз, якому мають відповідати значення, що включаються до результатів відбору.

5. Підсумування даних

6. Перегляд результатів

Access дозволяє створення в режимі **Конструктор запитів:**

вибіркового запиту

параметризованого запиту

запиту підсумків

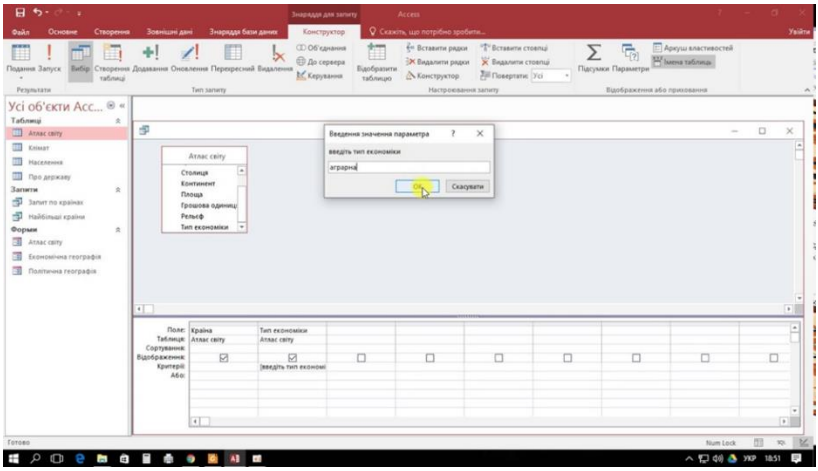
перехресного запиту

запиту на створення таблиці

запиту на додавання

запиту на оновлення

запиту на видалення



Зауваження. В режимі конструктор можливий перехід в **Режим SQL** на панелі інструментів зліва на вкладинці **Результати**. Цей режим дозволяє будувати запити мовою **SQL**. Access реалізує більшість запитів мовою **SQL**.

Вибірковий запит. Параметризований запит.

1. **Створення** натисніть кнопку **Конструктор запитів**.
2. У діалоговому вікні **Відображення таблиці** на вкладці **Таблиці** виберіть потрібну таблицю.
3. На бланку запиту в рядку **Критерії** в певному стовпці визначить критерії (умови) відбору або для параметризованого запиту введіть **[ТЕКСТ?]**.

Рядок **[ТЕКСТ?]** – це параметр, який пропонуватиметься вказати. Квадратні дужки вказують на необхідність введення параметра під час виконання запиту, а ТЕКСТ – це питання, яке відображатиметься, коли буде запропоновано ввести значення параметра.

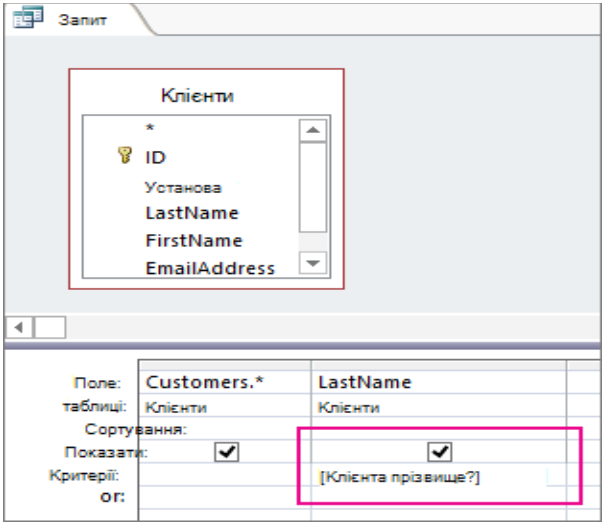
Також у рядку **Або**, розташованому під рядком **Критерії,** за потреби укажіть альтернативні критерії.

4. На вкладці **Конструктор**у групі меню-інструменти **Результати** натисніть кнопку **Виконати (!)**. Якщо запит

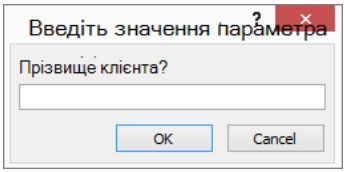
параметризований, то вкажіть в діалоговому вікні критерій відбору.

5. Створення запиту закінчіть введенням імені об’єкта **Запит** і натисніть клавішу **Enter.**

Вибірковий запит з параметром:



В діалоговому вікні задати значення відбору:



Групові операції:

- **Кількість**, щоб підраховувати кількість строк (включаючи NULL значення).
- **Сума**, щоб підсумовувати стовпець чисел.
- **Середнє**, щоб знаходити середнє значення для стовпця чисел.

- **Максимум**, щоб визначати найбільше значення в полі.
- **Мінімум**, щоб визначати найменше значення в полі.
- **Стандартне відхилення**, щоб визначити, наскільки широко розташовані точки даних відносно їхнього середнього значення.
- **Відхилення**, щоб вимірювати статистичне відхилення всіх значень у стовпці.

Встановлення доступу до групових операцій через Стрічку **Підсумки (∑).**

Критерії відбору. Оператори.

Арифметичні оператори.

Використовуючи арифметичні оператори, можна виконувати арифметичні дії з кількома числами, а також змінювати знак числа з додатного на від'ємний або навпаки.

Оператор	Призначення	Приклад
+	Сума двох чисел.	[Підсумок] + [Податок]
-	Визначення різниці між двома числами або позначення від'ємного значення числа.	[Ціна]-[Знижка]
*	Множення двох чисел.	[Кількість]*[Ціна]

/	Ділення першого числа на друге.	[Підсумок]/[Кіл-ть_позицій]
\	Округлення обох чисел до цілих чисел, ділення першого на друге та скорочення результату до цілого числа.	[Зареєстровано]\[Номери]
Mod	Ділення першого числа на друге й повернення лише залишку.	[Зареєстровано] Mod [Номери]
^	Піднесення числа до степеня.	[Число] ^ [Степінь]

Оператори порівняння.

Оператори порівняння використовуються, щоб порівнювати значення, і повертають результат "Істина", "Хибність" або Null-значення.

Оператор	Призначення	Приклад
<	Повертає значення "Істина", якщо перше значення менше за друге.	значення1 < значення2
<=	Повертає значення "Істина", якщо перше значення менше за друге або рівне йому.	значення1 <= значення2
>	Повертає значення "Істина", якщо перше значення більше за друге.	значення1 > значення2
>=	Повертає значення "Істина", якщо перше значення більше за друге або рівне йому.	значення1 >= значення2
=	Повертає значення "Істина", якщо перше значення дорівнює другому.	значення1 = значення2
<>	Повертає значення "Істина", якщо перше значення не дорівнює другому.	значення1 <> значення2

Примітка: Якщо перший або другий операнд має Null-значення, результат – також Null-значення. Оскільки Null-значення позначає невідоме значення, результат порівняння з Null-значенням також буде невідомий.

Логічні оператори.

За допомогою логічних операторів можна об’єднувати два логічні значення. Вони повертають результат True, False або Null-значення. Логічні оператори також називаються булевими.

Оператор	Призначення	Приклад
And	Повертає значення "Істина", якщо вираз 1 і вираз 2 істинні.	вираз1 And вираз2
Or	Повертає значення "Істина", якщо вираз 1 або вираз 2 істинний.	вираз1 Or вираз2
Eqv	Повертає значення "Істина", якщо вираз 1 і вираз 2 обидва істині або обидва хибні.	вираз1 Eqv вираз2
Not	Повертає значення "Істина", якщо вираз хибний, повертає "Хибність", якщо вираз істинний.	Not вираз
Xor	Повертає значення "Істина", якщо істинний лише вираз 1 або лише вираз 2, але не обидва одночасно.	вираз1 Xor вираз2

Примітка: Якщо перший або другий операнд має Null-значення, результат – також Null-значення. Оскільки Null-значення позначає невідоме значення, результат порівняння з Null-значенням також буде невідомий.

Оператори об’єднання.

Оператори об’єднання використовуються, щоб об’єднувати два текстові значення в одне.

Оператор	Призначення	Приклад
&	Поєднання двох рядків в один.	рядок1 & рядок2

+	Об’єднує два рядки в один і розповсюджує Null-значення (якщо один операнд має Null-значення, увесь вираз матиме Null-значення).	рядок1 + рядок2

Спеціальні оператори.

Спеціальні оператори повертають результат "Істина" або "Хибність".

Оператор	Призначення	Приклад
Is Null або **Is Not Null**	Визначає, чи дорівнює величина Null-значенню.	Поле1 Is Not Null
Like "шаблон"	Зіставляє рядкові значення за допомогою операторів узагальнення **?** , *****.	Поле1 Like "інструк*"
Between значення1 **And** значення2	Визначає, чи належить число або дата до діапазону.	Поле1 Between 1 And 10 Поле1 Between #01.07.2007# And #31.12.2007#

In(значення1,значення 2...)	Визначає, чи належить число до набору значень.	Поле1 In ("червоний ", "зелений", "синій") Поле1 In (1,5,7,9)

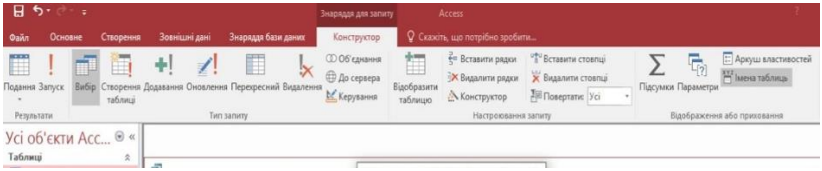
Тема 4. Запити на зміну даних. Активні запити.

Запит — спеціальний об’єкт, призначений для вибірки даних з таблиць бази, а також для виконання обчислень та інших операцій з базовими таблицями, включаючи їхнє перетворення.

Види запитів:

- **Вибір – запити на вибірку (вибіркові запити)**; запити на вибірку з полем, що обчислюється; підсумкові запити; запити з параметром; перехресні запити;
- **Дія – запити на зміну** (активні запити) – на створення таблиці, на додавання, на оновлення, на видалення.

Активні запити будуються в режимі **Конструктор запитів**. Кожен з цих запитів має свою позначку.



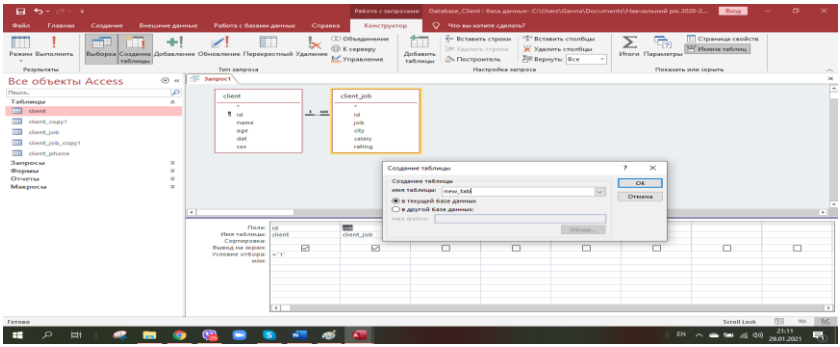
Запит на створення таблиці.

Можна скористатися запитом на створення таблиці, щоб створити нову таблицю на основі даних, які зберігаються в інших таблицях.

1. На вкладці **Створення** у групі **Запити** натисніть кнопку **Конструктор запитів**.
2. У діалоговому вікні **Відображення таблиці** виберіть таблиці-витоки, а потім закрийте діалогове вікно **Відображення таблиці**.
3. Зазначити **Поля Таблиці Критерії** відбору
4. На вкладці **Конструктор** у групі **Результати** натисніть **Виконати (!)** переконайтесь у необхідних результатах
5. На вкладці **Конструктор** у групі **Тип запиту** натисніть **Створення таблиці**.
6. У діалоговому вікні **Створити таблицю** в полі І**м'я таблиці** введіть ім'я нової таблиці (не має співпадати з існуючими) та натисніть кнопку **ОК**. Також можливо створити таблицю в іншій БД, поставивши відповідну позначку в діалоговому меню.
7. На вкладці **Конструктор** у групі **Результати** натисніть кнопку **Виконати**. Нову таблицю створено з даними відповідно встановленим критеріям відбору, її можна бачити в **Області переходів**.
8. Якщо потрібно зберегти запит – закрийте Запит або натисніть сполучення клавіш Ctrl+S та задайте ім'я нового об'єкта **Запит**.

Зауваження. 1. Запит на створення таблиці є запит на зміну і супроводжується повідомленням системи.

2. Запит на створення таблиці спрямований на створення **копій** об'єктів БД.



Запит на додавання.

Можна скористатися запитом на додавання, щоб отримати дані з однієї або кількох таблиць і додати їх до іншої таблиці.

1. На вкладці **Створення** у групі **Запити** натисніть кнопку **Конструктор запитів**.

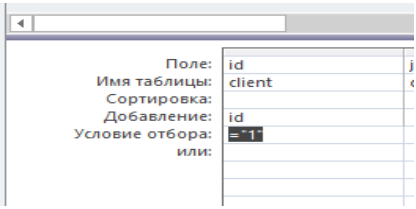
2. У діалоговому вікні **Відображення таблиці** виберіть таблиці-витоки, а потім закрийте діалогове вікно **Відображення таблиці**.

3. Зазначити **Поля Таблиці Критерії** відбору

4. На вкладці **Конструктор** у групі **Результати** натисніть **Виконати (!)** переконайтесь у необхідних результатах

5. На вкладці **Конструктор** у групі **Тип запиту** натисніть **Додавання.**

6. У діалоговому вікні **Додавання** в полі **Ім'я таблиці** введіть ім'я існуючої таблиці-прийомник, КУДИ будуть розміщені дані та натисніть **OK**. Також можливо додати дані в існуючу таблицю в іншій БД, поставивши відповідну позначку в діалоговому меню.

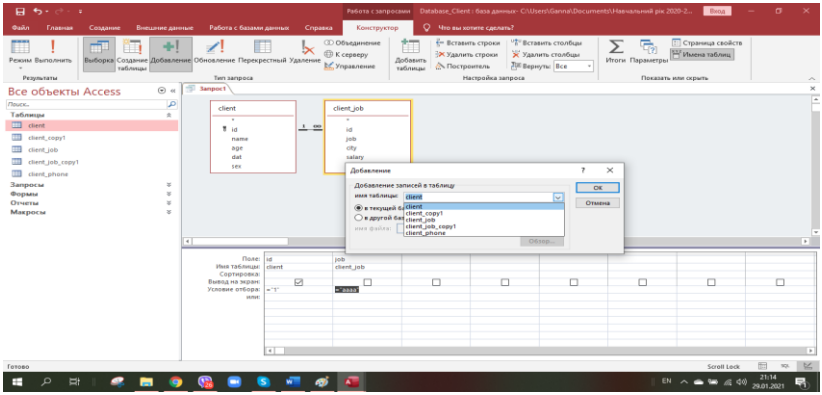


7. На бланку у рядку **Додавання** вибрати поле вказаної в пункті 6 таблиці-прийомник, КУДИ будуть розміщені дані.

8. На вкладці **Конструктор** у групі **Результати** натисніть кнопку **Запуск**. У вказану таблицю занесено відповідні дані згідно встановленого критерію відбору.

9. Якщо потрібно зберегти запит – закрийте Запит або натисніть сполучення клавіш Ctrl+S та задайте ім'я нового об'єкта **Запит**.

Зауваження. Запит на додавання даних є запит на зміну і супроводжується повідомленням системи.



Запит на оновлення.

Можна скористатися запитом на оновлення для змінення даних у таблицях. За допомогою цього запиту також можна ввести критерії для визначення рядків, які потрібно оновити. Запит на оновлення надає можливість переглянути оновлені дані до виконання оновлення.

1. На вкладці **Створення** у групі **Запити** натисніть кнопку **Конструктор запитів**.

2. У діалоговому вікні **Відображення таблиці** виберіть таблицю для оновлення даних, а потім закрийте діалогове вікно **Відображення таблиці**.

3. На вкладці **Конструктор** у групі **Тип запиту** натисніть **Оновлення.**

4. На бланку зникнуть рядки **Сортування** й **Відображення**, і з'явиться рядок **Оновлення до**.

5. На бланку в рядках **Поле, Таблиці** вибрати відповідні, що потребують оновлення, а у рядку **Оновлення до** введіть значення або у квадратних дужках вираз, що буде запускати діалогове вікно для введення оновлення.

Можна скористатися запитом на оновлення, щоб видалити значення полів за допомогою пустого рядка або значення NULL у рядку **Оновлення до**.

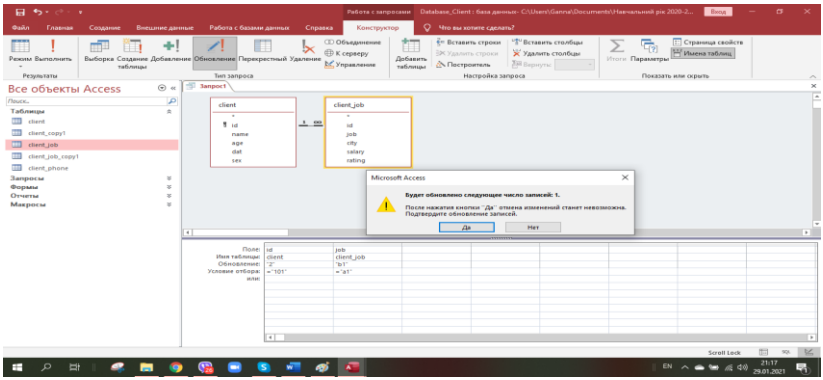
6. У рядку **Критерії** введіть критерій відбору для визначення поновлюваних записів.

7. У вікні табличного подання даних можна переглянути, які значення будуть змінені за допомогою запиту на оновлення.

8. На вкладці **Конструктор** у групі **Результати** натисніть **Виконати (!).** У вказаній таблиці оновлено відповідні дані згідно встановленого критерію відбору.

9. Якщо потрібно зберегти запит – закрийте Запит або натисніть сполучення клавіш Ctrl+S та задайте ім'я нового об'єкта **Запит**.

Зауваження. Запит на оновлення даних є запит на зміну і супроводжується повідомленням системи.



Запит на видалення.

Щоб видалити дані з таблиць, можна скористатися запитом на видалення. За допомогою запиту на видалення можна також ввести критерії для визначення, які рядки потрібно видалити. Запит на видалення надає можливість переглянути рядки, що мають бути видалені, до виконання видалення.

1. На вкладці **Створення** у групі **Запити** натисніть кнопку **Конструктор запитів**.

2. У діалоговому вікні **Відображення таблиці** виберіть одну таблицю для видалення даних, а потім закрийте діалогове вікно **Відображення таблиці**.

3. На вкладці **Конструктор** у групі **Тип запиту** натисніть кнопку **Видалення**. На бланку зникнуть рядки **Сортування** й **Відображення**, а рядок **Видалення** з’явиться.

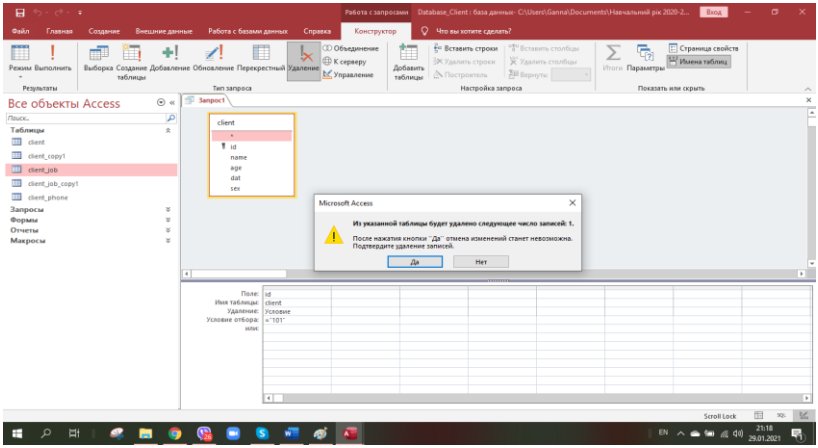
4. Зазначити **Поле, Таблиці, Критерії** відбору.

5. На вкладці **Конструктор** у групі **Результати** натисніть **Виконати (!).**

6. Якщо потрібно зберегти запит – закрийте Запит або натисніть сполучення клавіш Ctrl+S та задайте ім’я нового об’єкта **Запит**.

Зауваження. 1. Запит на видалення даних є запит на зміну і супроводжується повідомленням системи.

2. Корисно спочатку зробити запит на вибірку, потім на видалення даних.



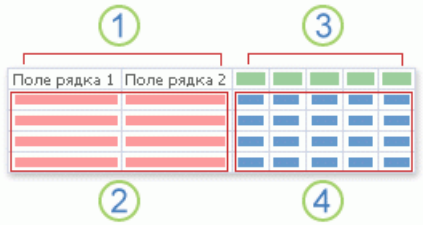
Перехресний запит.

Перехресний запит – це тип вибіркового запиту. Якщо виконати перехресний запит, результати відображаються в таблиці даних з особливою структурою.

Перехресний запит може відображати ті самі дані, але групує їх одночасно по горизонталі та по вертикалі, щоб зробити дані в табличному поданні компактнішими та зрозумілішими.

Створюючи перехресний запит, слід вказати поля, що міститимуть **заголовки рядків**, поле, що міститиме **заголовки стовпців**, і поле, що міститиме значення, які потрібно **підсумувати**. Для заголовків стовпців і значень, які потрібно підсумувати, можна використовувати лише по одному полю. Для заголовків рядків можна використовувати до трьох полів.

Також для створення заголовків рядків, заголовків стовпців і значень, які потрібно підсумувати, можна використовувати вирази.



1. Один, два або три стовпці з цієї сторони містять заголовки рядків. Імена полів, що використовуються як заголовки рядків, відображаються у верхньому рядку цих стовпців.

2. Тут відображаються заголовки рядків. Якщо використовується кілька полів заголовків рядків, кількість рядків у перехресній таблиці даних може швидко збільшуватись, оскільки відображається кожна комбінація заголовків рядків.

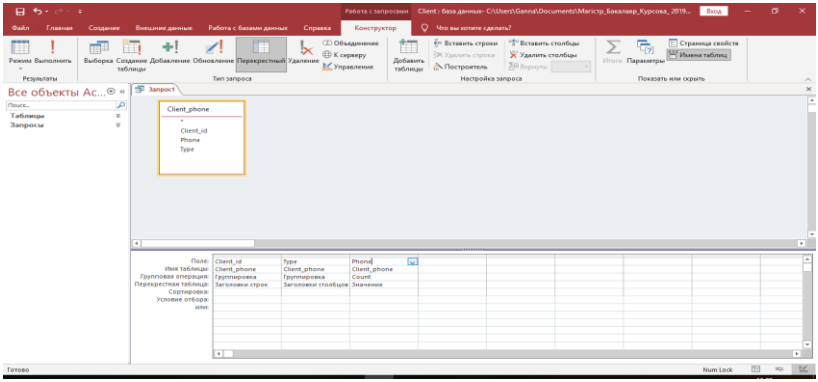
3. Стовпці з цієї сторони містять заголовки стовпців і зведені значення. Зауважте, що ім’я поля заголовка стовпця не відображається в таблиці даних.

4. Тут відображаються зведені значення.

1. На вкладці **Створення** у групі **Запити** натисніть кнопку **Конструктор запитів**.
2. У діалоговому вікні **Відображення таблиці** виберіть таблицю (таблиці), а потім закрийте діалогове вікно **Відображення таблиці**.
3. На вкладці **Конструктор** у групі **Тип запиту** натисніть **Перехресний**.
4. У рядку **Перехресний** виберіть із розкривного списку такі значення: **Заголовок рядка** для першого стовпця (до трьох значень), **Значення** для другого стовпця (одне значення), **Заголовок стовпця** для третього стовпця (одне значення).
5. На вкладці **Конструктор** у групі **Результати** натисніть **Виконати (!)**.

6. Натисніть сполучення клавіш CTRL+S, щоб зберегти запит.

Зауваження. Перехресний запит є запит на вибірку.



Зауваження. Всі запити на зміну призводять до відповідних змін, що не можна відмінити!

Тема 5. Форми Access.

Форма.

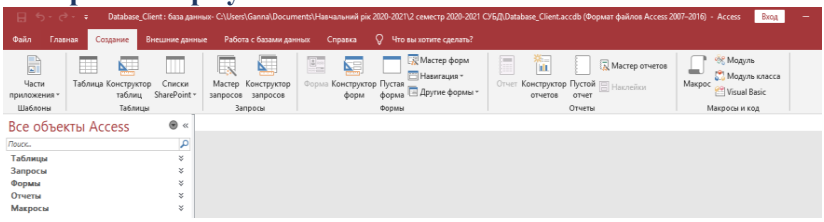
Форма Access – це об’єкт бази даних, за допомогою якого можна створити інтерфейс користувача для програми бази даних.

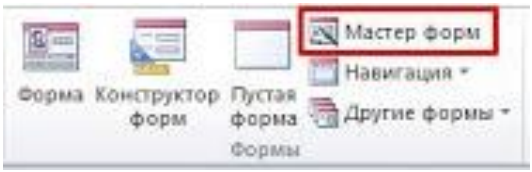
Розрізняють два основні види форм:

- **"зв’язана" форма** – безпосередньо підключена до джерела даних таблиці чи запиту, за допомогою якої можна вводити, редагувати або відображати дані з цього джерела даних,
- **"вільна" форма –** не веде прямо до джерела даних, але містить кнопки, підписи або інші елементи керування, необхідні для роботи з програмою.

Ефективна форма прискорює використання бази даних, оскільки в ній не потрібно нічого шукати. Візуально приваблива форма підвищує зручність і ефективність роботи з базою даних, а також допомагає запобігати введенню неприпустимих даних.

Створити>Форму





Режим Форма.

1. В області переходів клацніть таблицю або запит із даними, які мають відображатись у формі.

2. На вкладці **Створення** в групі **Форми** натисніть **Форма**.

У програмі Access створиться форма, яка відобразиться в режимі розмічування. У режимі розмічування можна змінювати макет форми під час відображення в ній даних. Наприклад, можна настроїти розмір текстових полів відповідно до обсягу даних.

Якщо Access знаходить одну таблицю, яка має зв’язок "один-до-багатьох" із таблицею чи запитом, що використовувався для створення форми, то програма додає таблицю даних до форми на основі пов’язаної таблиці чи запиту. У разі наявності кількох таблиць, які мають зв’язок "один-до-багатьох" із таблицею, що використовувалася для створення форми, програма Access не додаватиме таблиці даних до форми.

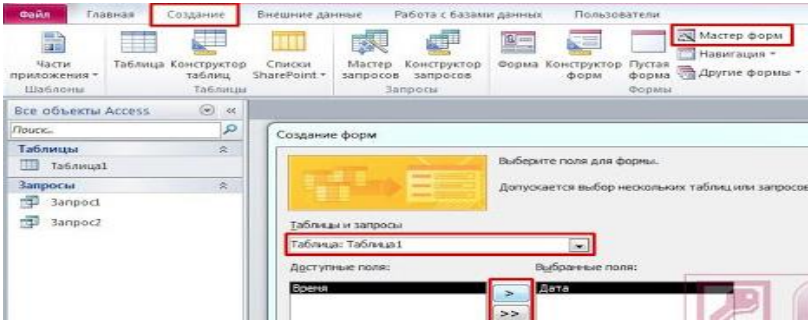
Режим Майстер форм.

Щоб вибрати поля, які мають відображатись у формі, використовуйте майстер форм. Можна визначити спосіб групування й сортування даних і використати поля з кількох таблиць або запитів за умови, що зв'язки між таблицями й запитами вказано заздалегідь.

1. На вкладці **Створення** в групі **Форми** натисніть **Майстер форм**. Дотримуйтеся вказівок на сторінках майстра форм.

Якщо потрібно – додайте до форми поля з кількох таблиць і запитів. Повторіть кроки для вибору таблиці чи запиту та виберіть додаткові поля, які потрібно включити до форми. Після цього натисніть **Далі** або **Готово**, щоб продовжити.

2. На останній сторінці майстра натисніть кнопку **Готово**.



Режим Пуста форма.

Якщо майстер або інструменти для побудови форми не відповідають вашим потребам, ви можете створити форму за допомогою засобу "Пуста форма". Це швидкий спосіб створення форми, особливо якщо у формі потрібно відображати лише кілька полів.

1. На вкладці **Створення** в групі **Форми** натисніть **Пуста форма**.

В Access відкриється пуста форма в режимі розмітки та відобразиться область **Список полів**.

2. В області **Список полів** натисніть знак + поруч з однією або кількома таблицями, що містять поля, які мають відображатись у формі. Щоб додати до форми поле, двічі клацніть або перетягніть на форму.

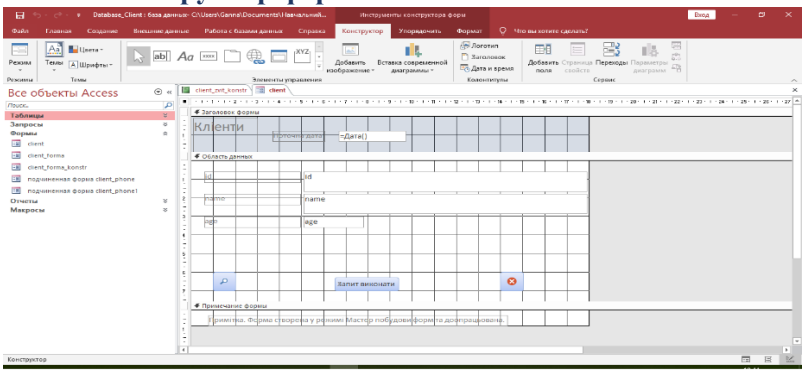
Порядок таблиць в області **Список полів** може змінюватися залежно від того, яку частину форми вибрано. Якщо потрібне поле не відображається, спробуйте вибрати іншу частину форми, а потім додати поле.

3. За допомогою інструментів на вкладці **Конструктор** у групі **Колонтитули** можна додати до форми логотип, заголовок, дату й час.

4. За допомогою інструментів на вкладці **Конструктор** у груп **Елементи керування** можна додати до форми різноманітні елементи керування.

Якщо вам потрібен більший вибір елементів керування, перейдіть у режим **Конструктор**.

Режим Конструктор форм.



Режим макета (розмітки) – це подання, у якому найзручніше змінювати форму. У цьому режимі можна внести практично будь-які можливі зміни до форми в Access. У режимі розмітки форма фактично працює. Таким чином, ви можете побачити, як відображатимуться дані під час використання форми. Проте в цьому режимі також можна змінити макет форми. Якщо завдання не можна виконати в Режимі розмітки, перейдіть у Режим конструктора на панелі інструментів зліва.

Режим Конструктор надає деталізоване подання структури форми: **Заголовок форми** (верхній колонтитул), **Область даних**, **Примітки форми** (нижній колонтитул). У режимі конструктора форма не працює.

Можна виконувати дії:

- додавати до форми елементи керування;
- редагувати джерела елемента керування "текстове поле" в самих текстових полях;
- змінювати розділи форми та змінювати певні властивості форми.

Можна редагувати вже існуючу форму в режимі **Конструктор** додаючи або змінюючи певні елементи. Можна створити форму в режимі **Конструктор**.

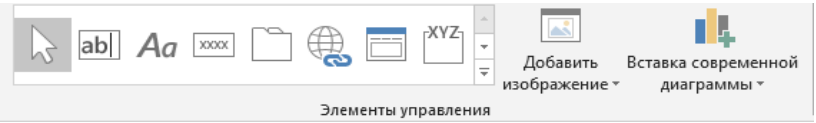
1. На вкладці **Створення** в групі **Форми** натисніть **Конструктор форм**.

В Access відкриється пуста форма в режимі **Конструктор**, що складається з **Області даних**. Можна додати елементи структури форми **Верхній колонтитул, Нижній колонтитул**, **Заголовок форми, Примітки форми**, натиснувши на **Заголовок** на панелі інструментів на вкладці **Конструктор** у групі **Колонтитули,** також можна додати до форми емблему, назву або дату й час.

2. На вкладці **Конструктор** у групі **Знаряддя** (Сервіс) натисніть **Додавання полів**.

Поля можна перетягнути безпосередньо з області **Список полів** усі разом до форми. Щоб одночасно додати кілька полів, виберіть та перетягніть поля в необхідне місце форми, утримуючи натиснутою клавішу Ctrl.

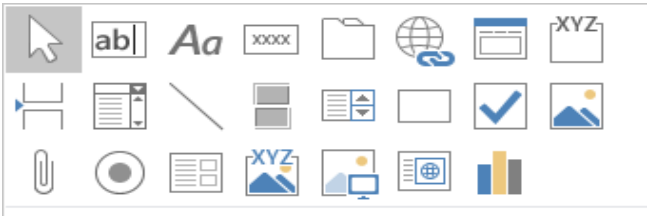
3. За допомогою інструментів на вкладці **Конструктор** у групі **Елементи керування** можна додати до форми різноманітні елементи керування.



4. Закрити та зберегти форму, задавши ім’я форми.

Типи Елементів керування.

Нижче наведено перелік елементів керування, які можна використовувати в базах даних Access.



Елемент керування	Опис
Вказівник	Використовується для позначення місця або об’єкта.
Текстове поле	Використання поля для відображення тексту, чисел, дат, часу та записки поля. Текстове поле можна прив'язати до одного

	поля базової таблиці або запиту, Тоді нове значення в текстовому полі оновлює значення в зв’язній таблиці.
Підпис	За допомогою цієї команди можна створювати окремі заголовки.
Кнопка	Використовується для активації макросу або процедури Visual Basic. Також можна вказати адресу гіперпосилання, що відкривається після натискання кнопки користувачем.
Вкладка	Використовується для створення ряду сторінок у формі. Кожна сторінка може містити кілька інших елементів керування для відображення інформації.
Посилання	Додавання гіперпосилання, може містити уніфікований покажчик ресурсу (URL), що вказує на розташування в Інтернеті, локальній мережі або локальному диску. Посилання створюють наведенням вказівника на файл, веб-сторінку та вказівкою в діалоговому вікні Додавання гіперпосилання.
Навігація	Використовується для надання простого способу переходу до різних форм та звітів в базі даних. Забезпечує інтерфейс схожий на веб-сайт. Доступний в Access 2010 і пізніших версій.

Група параметрів	Використовується для позначення декількох перемикачів однією позначкою групи.
Розрив сторінки	Використовується для розділення між сторінками у формі кількох сторінок.
Поле зі списком	Містить список потенційних значень і повідомлення, що можна редагувати, як текстове поле. Щоб створити список – треба вибрати Джерело рядків (можна вказати таблиці або запити, як джерела значень у списку). Якщо зв'язати поле зі списком в поле базової таблиці або запиті, можна змінити значення поля, вибравши нові значення у списку..
Лінія	Використання елемента для покращення вигляду.
Кнопка-перемикач	Вмикнення логічного значення true, або false, або немає значення. Коли натискаєте кнопку перемикач – значення стає 1 або true, а кнопка натиснутою. Натисніть кнопку ще раз – її значення стає 0 або false, кнопку повертається до звичайного.
Список	Для створення списку, треба ввести значення в полі властивості Джерело рядків. Також можна вказати таблиці або запити, як джерела значень у списку.

Прямокутник	Додавання прямокутників до форми, щоб покращити вигляд або для візуального групування різних елементів керування.
Прапорець	Отримання повідомлення для вмикання логічного значення true, або false, або немає значення. Коли ви вибираєте прапорець – значення стає 1 або true, а позначка відображається в полі. Інакше – значення стає 0 або false, а позначка зникає з поля.
Вільна рамка об'єкта	Додавання об'єкту з іншої програми, яка підтримує зв'язування та вбудовування об'єктів (OLE). Об'єкт стає частиною форми, не є частиною даних з базової таблиці або запиту. Можна додати зображення, звуки, діаграми або слайди, щоб покращити форми.
Вкладення	Цей елемент керування, можна використовувати, наприклад, щоб відобразити зображення або вкласти інші файли.
Кнопка «Параметри»	Використовується як «перемикач» (іноді її називають перемикач), треба ввести повідомлення та вибрати логічний вираз, що приймає значення true, або false, або немає значення..
Підформа Підзвіт	Використовується для вбудовування іншої форми або звіту в поточній формі. Підформу або підзвіт можна використовувати для відображення даних із

	таблиці або запиту, пов'язаний із даними головної форми.
Зв'язаний об'єкт рамки	Використання кадру зв'язаний об'єкт для відображення та редагування поле об'єкт OLE з основних даних. Access можна відобразити більшості зображень і графіки безпосередньо на формі. Для інших об'єктів у програмі Access відобразиться піктограма застосунку, у якому створено об'єкт.
Зображення	За допомогою елемента керування зображенням розмістити статичні зображення у формі. Не можна редагувати зображення у формі.
Веб-браузер	Використання браузера для відображення вмісту з веб-сторінок безпосередньо у формі. Доступний в Access 2010 і пізніших версіях.
Діаграма	Додавання діаграми до форми за допомогою керування діаграми. Натисніть цю кнопку і виберіть розміщення до елемента керування на формі запуск майстра діаграм.

Елемент керування може бути:

- **Зв’язаний елемент керування** – джерелом даних є поле в таблиці або запиті, що називається Приєднаний елемент керування. Зв'язані елементи керування використовується для відображення значень, які постачаються з полів БД. Значення може бути текстом, числом, логічним, зображенням або схемою.

- **Вільний елемент керування** – не має джерела даних (наприклад, поля або вираз). Використовується для відображення інформації, зображень, ліній або прямокутників. Наприклад, заголовок, який відображає назву форми – це вільний елемент керування.
- **Обчислюваний елемент керування** – джерелом даних є вираз, а не поле. Вираз може бути комбінацією операторів, елементів керування, імен полів, функцій, що повертають одне значення, констант. Наприклад, вираз = [Вартість одиниці товару] * 0,75 обчислює вартість одиниці товару зі знижкою 25%

Інші форми.

Access має можливості створювати спеціальні форми:

- Декілька елементів
- Таблиця
- Розділена форма
- Модальне діалогове вікно

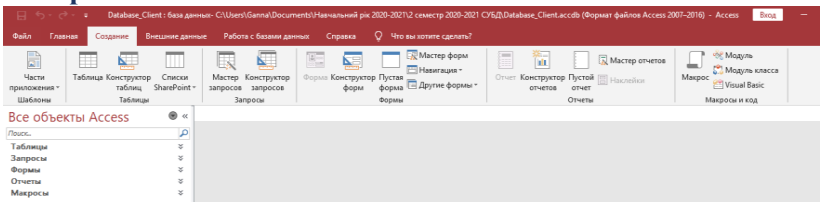
Тема 6. Звіти Access.

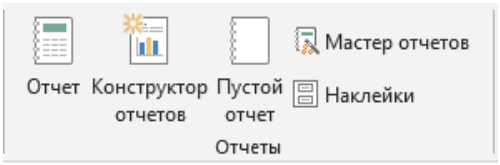
Звіти.

Звіт – це об’єкт БД, яким зручно користуватися для представлення відомостей, зведення даних, надання докладних відомостей про окремі записи, створення підписів та підготовки до друку.

За допомогою звітів можна переглядати, форматувати та підсумовувати дані.

Створити>Звіт.





Режим Звіт.

Створює простий табличний звіт, який містить усі поля вибраного в області переходів джерела записів.

1. В області переходів вибрати таблицю або запит, на основі яких потрібно побудувати звіт.

2. На вкладці **Створення** в групі **Звіти** натисніть **Звіт**.

Access сформує звіт і відобразить його в **Режимі розмітки (макета)**.

3. Зберегти з ім’ям та закрити звіт, вихідну таблицю або запит, що використовувались, як джерело інформації.

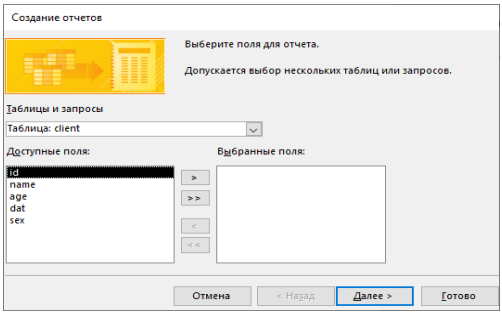
Відкривши звіт за ім’ям, Access відобразить оновлені дані з джерела записів.

Режим Майстер звітів.

Запускає покроковий майстер, у якому можна вказати поля, рівні групування/сортування та параметри макета для створення звіту.

1. На вкладці **Створення** у групі **Звіти** натисніть **Майстер звітів**.

2. Дотримуйтеся вказівок на сторінках Майстра звітів. На останній сторінці натисніть кнопку **Готово**.



Режим Пустий звіт.

Відкриває звіт у поданні макета та відкриває **Список полів**, за допомогою якого можна додавати поля до звіту.

5. На вкладці **Створення** в групі **Звіт** натисніть **Пустий звіт**.

6. Пустий звіт відобразиться в **Режимі розмітки**, а в правій частині вікна Access з'явиться область **Список полів**.

7. В області **Список полів** клацніть знак + біля таблиці або таблиць з полями, які потрібно відображати у звіті. Перетягніть поля до звіту по одному або кілька, натиснувши й утримуючи клавішу Ctrl.

8. За допомогою інструментів на вкладці **Конструктор** у групі **Колонтитули** можна додати до форми логотип, заголовок, дату й час та нумерацію сторінок.

9. За допомогою інструментів на вкладці **Конструктор** у груп **Елементи керування** можна додати до форми нові елементи керування.

Режим Конструктор звітів.

Відкриває пустий звіт у режимі **Конструктор.** До цього звіту можна додавати потрібні поля й елементи керування.

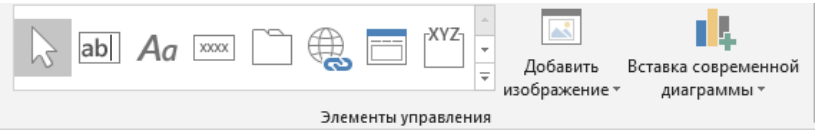
5. На вкладці **Створення** в групі **Звіти** натисніть **Конструктор звітів**.

В Access відкриється пустий звіт в режимі **Конструктор**, що складається з **Області даних, Верхній колонтитул, Нижній колонтитул**. Можна додати елементи структури звіту **Заголовок звіту (Верхній колонтитул звіту), Примітки звіту (Нижній колонтитул звіту)**, натиснувши на **Заголовок** на панелі інструментів на вкладці **Конструктор** у групі **Колонтитули,** також можна додати до форми емблему, назву або дату й час.

6. На вкладці **Конструктор** у групі **Знаряддя** (Сервіс) натисніть **Додавання полів**.

Поля можна перетягнути безпосередньо з області **Список полів** усі разом до звіту. Щоб одночасно додати кілька полів, виберіть та перетягніть поля в необхідне місце звіту, утримуючи натиснутою клавішу Ctrl.

7. За допомогою інструментів на вкладці **Конструктор** у групі **Елементи керування** можна додати до форми різноманітні елементи керування, що були розглянуті в темі Форми.



8. Якщо необхідно **Додати групування** та **Додати сортування** даних групи.

На вкладці **Конструктор** у групі **Групування та підсумки** натисніть кнопку **Групування та сортування**. Натисніть **Додати групу** або **Додати сортування**, а потім укажіть поле, за яким потрібно виконати групування або сортування. Щоб указати додаткові параметри натисніть **Розгорнути**.

9. Закрити та зберегти звіт, задавши ім’я звіту.

Структура звіту.

Розділ	Відображення в друці	Використання
Верхній колонтитул звіту.	На початку звіту. **Примітка.** Спочатку друкується верхній колонтитул звіту, а потім – верхній колонтитул сторінки.	Верхній колонтитул використовується для даних, які зазвичай розміщуються на титульній сторінці – логотипи, заголовки, дати. Якщо у верхньому колонтитулі розмістити обчислюваний елемент керування, з агрегатною функцією Sum, то сума обчислюватиметься для всього звіту.
Верхній колонтитул сторінки.	У верхній частині кожної сторінки.	Верхній колонтитул сторінки використовується для повторення заголовка звіту на кожній сторінці.
Верхній колонтитул групи.	На початку кожної нової групи записів.	Верхній колонтитул групи використовується для друку назви групи. Якщо у верхньому

Розділ	Відображення в друці	Використання
		колонтитулі групи розмістити обчислюваний елемент керування, з агрегатною функцією Sum, то сума обчислюватиметься для поточної групи.
Область даних	Відображається один раз.	Саме тут розміщуються елементи керування, які становлять тіло звіту.
Нижній колонтитул групи.	Наприкінці кожної групи записів.	Нижній колонтитул групи використовується для відображення зведених даних для групи.
Нижній колонтитул сторінки.	Наприкінці кожної сторінки.	Нижній колонтитул сторінки використовується для друку номерів сторінок або даних, які стосуються кожної сторінки.

Розділ	Відображення в друці	Використання
Нижній колонтитул звіту.	Наприкінці звіту. **Примітка.** У режимі конструктора нижній колонтитул звіту розташовується під нижнім колонтитулом сторінки. Але під час друку або попереднього перегляду нижній колонтитул звіту розташовуватиметься *над* нижнім колонтитулом сторінки,	Нижній колонтитул звіту використовується для друку підсумків або інших зведених даних всього звіту.

Зауваження. Створювати змістовні звіти набагато простіше, коли база даних має добре спроектовану структуру таблиці та правильно визначені зв’язки.

Тема 7. Макроси і Код.

Макроси та Код.

На початку роботи з новою базою даних зазвичай створюють кілька її об’єктів, як то таблиць, форм і звітів. Потім виникає потреба додати до бази даних програмовані можливості, щоб автоматизувати певні процеси та зв’язати між собою об’єкти бази даних.

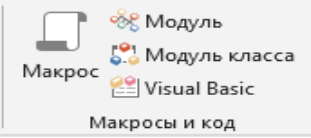
В Access програмування – це розширення функціональних можливостей бази даних за допомогою **Макросів Access** або **Коду** мовою **Visual Basic for Applications (VBA)**.

Наприклад, для створених форми та звіту потрібно додати кнопку до форми, натисканням якої відкриватиметься звіт. У цьому випадку програмування означає створення макросу або процедури VBA і налаштування властивості події **OnClick**-кнопки так, що після її натискання починається виконання макросу або процедури.

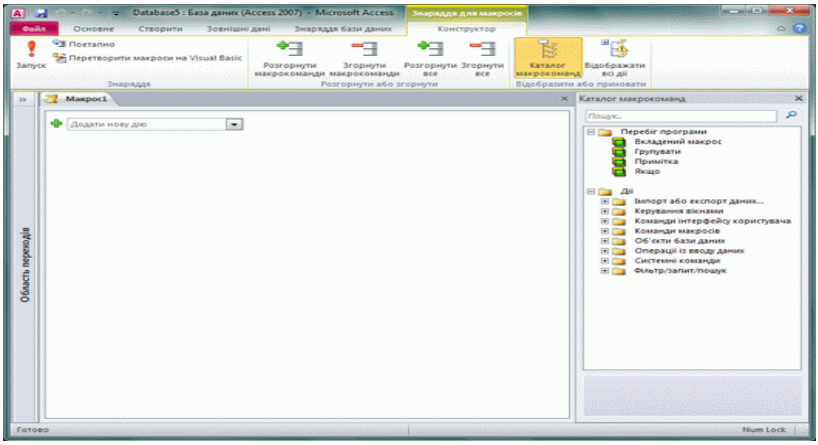
Макрос – це інструмент програми Access, за допомогою якого можна автоматизувати завдання та розширити можливості форм, звітів і елементів керування.

Макроси Access – це спрощена мова програмування, за допомогою якої складається список дій, які потрібно виконати. Під час створення макросу вибирають дії з розкривного списку (без написання коду в модулі Visual Basic for Applications VBA), а потім вводите потрібні відомості для кожної з дій. Макроси – підмножина команд, доступних у VBA.

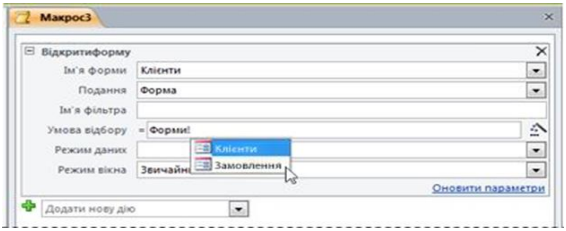
Створити>Макрос.



Макроси створюються за допомогою конструктора макросів, який наведено на знімку екрана нижче.



1. На вкладці **Створення** в групі **Макроси та код** натисніть кнопку **Макрос**.
2. Побудуйте макрос на основі макрокоманд з **Каталогу макрокоманд** справа.
3. Якщо макрос з декількох макрокоманд – визначте порядок, використовуючи стрілку переміщення макрокоманд.
4. Збережіть макрос з ім'ям, закривши вікно конструктора макросів.



Зауваження. Якщо макрос має запускатися при відкритті вказаної БД, цей макрос має бути іменований **AutoExec**.

Елемент керування Кнопка з вбудованим Макросом.

1. В області переходів виберіть форму, до якої потрібно додати кнопку, і виберіть режим **Конструктор** відкриття форми.

2. На вкладці **Конструктор** у групі **Елементи керування** виберіть **Кнопка**.

3. На бланку форми, клацніть місце, де потрібно розташувати кнопку. Та запуститься майстер **Створення кнопок**.

4. На першій сторінці майстра в списку **Категорії** клацніть кожну категорію, щоб переглянути дії для кнопки, які можна запрограмувати за допомогою майстра. В категорії **Різне** вибрати **Виконати макрос.** Вказати ім'я макроса з створених або AutoExec для виконання. Натиснути **Далі**.

5. Установіть перемикач **Текст** або **Зображення**, щоб відобразити на кнопці відповідно текст чи зображення. Натисніть кнопку **Далі**.

6. Введіть зрозуміле ім'я для кнопки. Цей крок необов'язковий, адже ім'я на кнопці не відображається але ідентифікує її.

7. Натисніть кнопку **Готово**.

Access розташує кнопку у формі, що натисканням запускає макрос.

Створити>Модуль.

Модуль – це об'єкт, використовуючи який, можна розширити функціональні можливості БД.

Щоб створити модуль, потрібно скористатися мовою програмування Visual Basic for Applications (VBA). Модуль – це колекція оголошень, інструкцій і процедур, які зберігаються разом.

Два види модулів.

Модулі класів підключаються до форм або звітів і зазвичай містять процедури для відповідної форми чи звіту.

Стандартні модулі містять загальні процедури, не пов’язані з жодним іншим об’єктом. Стандартні модулі відображаються в області переходів у розділі Модулі, а модулі класів – ні.

1. На вкладці **Створення** в групі **Макроси та код** натисніть кнопку **Модуль** або **Модуль класу**.
2. Використовувати Visual Basic for Applications (VBA) для визначення структури модуля.

Список використаних джерел

1. https://support.microsoft.com/en-us/office/build-an-access-database-to-share-on-the-web-cca08e35-8e51-45ce-9269-8942b0deab26
2. https://www.microsoft.com/en-us/microsoft-365/access
3. https://docs.microsoft.com/en-us/office/troubleshoot/access/define-table-relationships
4. https://accesshelp.ru/

Зміст

Навчальне видання

Верьовкіна Ганна Володимирівна

Навчальний посібник
з дисципліни "СУБД"

Система управління базами даних Access

для студентів механіко-математичного факультету,
які навчаються за освітнім рівнем "Бакалавр"
спеціальність "Математика"

Оригінал-макет виготовлено авторкою

Комп’ютерне верстання:
Верьовкіна Г.В.

Дизайн обкладинки:
Верьовкіна Г.В.