Інформатика

Профільний рівень

підручник для 11 класу
закладів загальної середньої освіти

Харків
Видавництво «Ранок»
2019

УДК 004:37.016 (075.3)
Р83

Руденко В. Д.
Р83 Інформатика (профільний рівень) : підруч. для 11 кл. закл. загал. серед. освіти / В. Д. Руденко, Н. В. Речич, В. О. Потієнко. — Харків : Вид-во «Ранок», 2019.
ISBN

УДК 004:37.016 (075.3)

Інтернет-підтримка
Електронні матеріали
до підручника розміщено на сайті
interactive.ranok.com.ua

ISBN

Зміст

Передмова

РОЗДІЛ 1. БАЗИ ДАНИХ

1. Загальні відомості про бази даних

1.1. Поняття бази даних і системи управління базами даних 6

1.2. Поняття моделі даних 9

1.3. Основні відомості про системе управління базами даних Access 13

2. Таблиці

2.1. Створення й уведення структури таблиць 16

2.2. Ключові поля, індекси, зв'язування таблиць 21

2.3. Введення, пошук і редагування даних у таблиці 25

2.4. Сортування і фільтрування записів. Операції над таблицями 28

3. Запити

3.1. Загальні відомості про запити 31

3.2. Запити на вибірку даних 34

3.3. Запити з функціями і з полями, що обчислюються 37

3.4. Запити з параметрами. Перехресні запити 41

3.5. Запити на змінення даних 44

4. Інтерфейс користувача. Основи мови SQL. Імпорт та експорт даних

4.1. Створення інтерфейсу користувача для введения даних у базу даних 47

4.2. Основи мови запитів SQL 51

4.3. Імпорт і експорт об'єктів баз даних 53

РОЗДІЛ 2. АЛГОРИТМИ

5. Алгоритми і числа

5.1. Методи проектування і подання алгоритмів 57

5.2. Поняття про кодування і складність алгоритмів 60

5.3. Основні поняття теорії чисел

5.3.1. Системи числення 68

5.3.2. Робота з великими числами 72

5.3.3. Факторізація чисел 75

6. Алгоритми сортування і пошуку даних

6.1. Алгоритми сортування даних

6.1.1. Квадратичні алгоритми сортування 80

6.1.2. Сортування вставками, злиттям і підрахунком 81

6.2. Алгоритми пошуку даних

6.2.1. Послідовний пошук 91

7. Обробка рядків

7.1. Основні відомості про рядки і операції над ними 98

7.2. Функції і методи опрацювання рядків 101

7.3. Приклади програм опрацювання рядків 103

8. Графи

8.1. Основні поняття і терміни теорії графів 106

8.2. Способи представлення графів у комп'ютері 110

8.3. Пошук у глибину і ширину 112

8.4. Визначення найкоротшого шляху у графі 116

9. Динамічне програмування і жадібні алгоритми
9.1. Динамічне програмування ... 127
9.2. Жадібні алгоритми ... 134
10. Основи обчислювальної геометрії
10.1. Базові поняття ... 139
10.2. Операції над векторами ... 142
10.3. Обчислення площі багатокутника ... 146
10.4. Побудова опуклої оболонки ... 150

РОЗДІЛ 3. ВЕБ-ТЕХНОЛОГІЇ
11.1. Основні тренди у веб- дизайні ... 155
11.2. Види сайтів та цільова аудиторія ... 157
11.3. Інформаційна структура сайта ... 162
11.4. Системи керування вмістом ... 165
11.5. Адміністрування сайта ... 167
11.6. Інструменти веб-розробника ... 170
11.7. Мова гіпертекстової розмітки ... 173
11.8. Каскадні таблиці стилів ... 175
11.9. Проектування та верстка веб-сторінок ... 177
11.10. Адаптивна верстка ... 182
11.11. Кросбраузерність ... 185
11.12. Графіка для веб-середовища ... 187
11.13. Анімаційні ефекти ... 189
11.14. Мультимедіа на веб-сторінках ... 193
11.15. Об'єктна модель документа ... 195
11.16. Веб-програмування та інтерактивні сторінки ... 198
11.17. Хостинг сайта ... 201
11.18. Веб-сервер та база даних ... 204
11.19. Взаємодія «клієнт-сервер» ... 206
11.20. Валідація сайта та збереження даних форм ... 209
11.21. Прикладний програмний інтерфейс ... 212
11.22. Правила ергономічного розміщення відомостей на веб-сторінці ... 216
11.23. Пошукова оптимізація та просування веб-сайтів ... 218

РОЗДІЛ 4. ПАРАДИГМИ ТА ТЕХНОЛОГІЇ ПРОГРАМУВАННЯ
12.1. Підходи до системного аналізу, етапи та методології розробки. Уніфікований процес розробки програмного забезпечення ... 223
12.2. Інструменти для проектної роботи, системи комунікації та контролю версій ... 226
12.3. Мова візуального моделювання архітектури програмного забезпечення ... 229
12.4. Аналіз та документація вимог проекту. Діаграми прецедентів ... 232
12.5. Моделювання даних і архітектури програмного забезпечення. Діаграми класів ... 235
12.6. Моделювання даних і архітектура ПЗ. Діаграми класів ... 239
12.7. Проектування інтерфейсу користувача. Продуктовий дизайн ... 242
12.8. Розроблення прототипу та тестування. Оцінювання системи ... 244
12.9. Системна архітектура, апаратні та програмні рішення, стандарти та тренди ... 246
Комп'ютерний словник ... 249
Алфавітний покажчик ... 250

Шановні учні та учениці!

В 11 класі ви завершуєте вивчення основ шкільного курсу інформатики. Це відповідальний період, адже в цей час формуються практичні й дослідницькі навички та ключові компетентності. Ви досягли певного рівня інформаційної культури і здатні самостійно оволодівати сучасними інформаційними технологіями. Та інформатика — дуже динамічна наука. Її подальші напрямки й темпи розвитку значною мірою визначатимуться рівнем підготовки людей, які мають ґрунтовні знання в цій галузі.

Цього року ви будете працювати з новими програмними засобами, освоїте розробку найпростішої бази даних навчального призначення в середовищі Access, навчитеся створювати веб-сайти з використанням систем керування вмістом, реалізовувати базові алгоритми засобами мови програмування Python і середовища програмування IDLE та створювати й налаштовувати програми за розробленими алгоритмами, опануєте основні етапи та методологію розробки програмного забезпечення тощо.

Бажаємо вам успіхів, *авторський колектив*

Підручник, який ви тримаєте в руках, — ваш надійний помічник. У ньому ви знайдете завдання для самостійного виконання — виконуйте їх на комп'ютері з натхненням, повторюйте теоретичний матеріал і викладайте основні положення на папері.

Матеріали на підтримку практичних робіт ви можете знайти на сайті interactive.ranok.com.ua.

Скориставшись цим посиланням, ви також маєте змогу пройти комп'ютерне тестування з автоматичною перевіркою результату.

Різнорівневі питання для перевірки знань і завдання для самостійного виконання відповідають рівням навчальних досягнень:

— початковий і середній рівні

— достатній рівень

— високий рівень

У тексті використано також позначення:


— питання на повторення

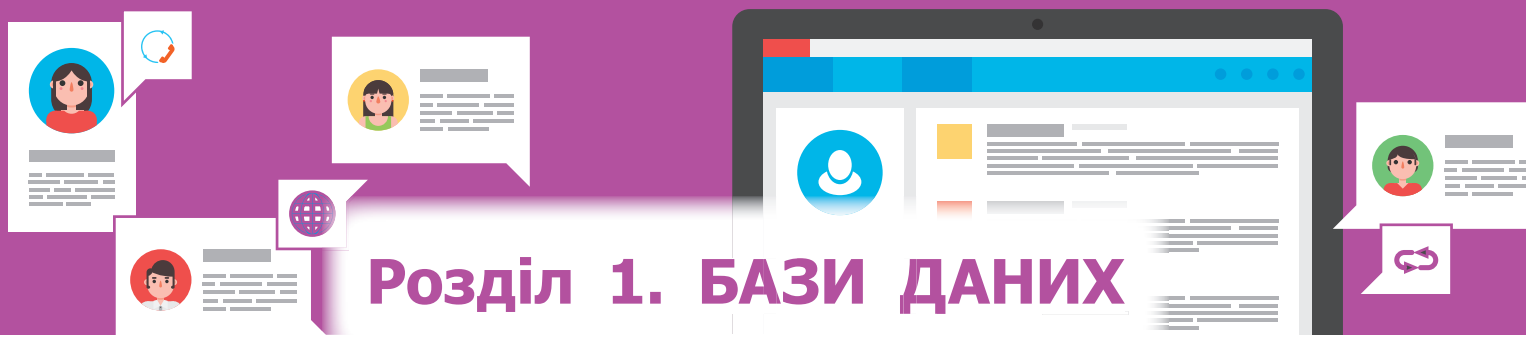

— означення, висновок


— зверніть увагу


— цікаво знати


— завдання для виконання й обговорення в парах або групах


— вправи для домашнього виконання



Розділ 1. БАЗИ ДАНИХ

1. Загальні відомості про бази даних

1.1. Поняття бази даних і системи управління базами даних

Згадайте, з якими базами даних вам доводилося працювати раніше. Наведіть приклади баз даних.

Нині важко уявити роботу великого супермаркету, системи резервування й продажу авіаквитків, банку, морського порту, готельного комплексу тощо без використання інформаційних систем, основою яких є бази даних (БД).

База даних — це сховище організованої сукупності даних різного типу, які відображують стан об'єктів певної предметної галузі та зв'язки між ними.

Предметною галуззю називають сферу застосування конкретної БД, наприклад школа, будівельна організація, аеропорт, банк, поліклініка, супермаркет тощо.

Об'єктом предметної галузі є те, про кого або про що зберігаються дані в БД. Якщо предметною галуззю є, наприклад, школа, то її об'єктами можуть бути учні, вчителі, директор школи, кабінети.

Кожен об'єкт характеризується сукупністю атрибутів, або властивостей (приклад 1). Далі об'єкти БД будемо позначати так: великими літерами — назва об'єкта, у круглих дужках — перелік його атрибутів, які відокремлюються один від одного комою. Наприклад, об'єкт ПОТЯГ можна позначити так: ПОТЯГ (номер потягу, станція відправлення, час відправлення, кінцева станція, час прибуття на кінцеву станцію).

За структурою даних БД поділяють на дві основні групи: документальні й фактографічні (рис. 1.1).

У фактографічних БД кожен атрибут об'єкта має певну сукупність значень, тобто елементів даних, які є найменшими неподільними одиницями даних. Наприклад, атрибут Центр (див. рис. 1.1) має значення Полтава і Хмельницький, атрибут Площа — значення 28 748 і 20 600, а атрибут Районів — значення 25 і 13.

База даних є однією з найважливіших складових сучасної інформаційної системи, побудованої на основі комп'ютерних систем і мереж. Робота з БД у таких системах здійснюється за допомогою спеціальної мови БД або програмного забезпечення — системи управління базами даних (СУБД).

Вперше термін *database* (база даних) з'явився на початку 60-х років XX ст. і був уведений у вжиток на симпозіумах, організованих фірмою System Development Corporation (США) у 1964 і 1965 роках. Широкого розповсюдження в сучасному розумінні цей термін набув у 1970-ті роки з розвитком ЕОМ.

Приклад 1. Об'єкт УЧЕНЬ може мати такі атрибути: прізвище, ім'я, рік народження, домашня адреса, школа, клас, зріст, а об'єкт АВТОМОБІЛЬ — такі: модель, потужність двигуна, максимальна швидкість, вантажопідйомність.

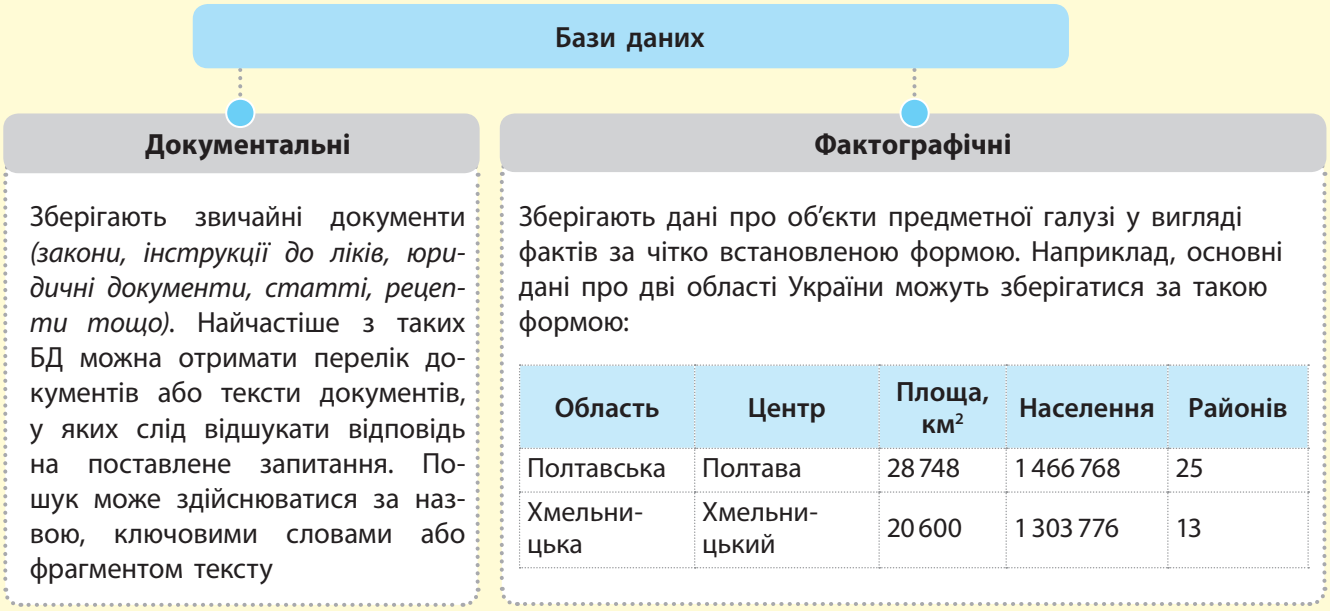


Рис. 1.1. Класифікація баз даних за структурою даних

Система управління базами даних — це інструмент, призначений насамперед для створення структури БД, уведення й оновлення даних, пошуку необхідних даних та їх опрацювання за певним алгоритмом.

Оскільки до БД може звертатися велика кількість користувачів, то важливою функцією СУБД є забезпечення цілісності й безпечності даних. Окрім функцій, безпосередньо пов'язаних зі створенням і підтримкою БД, окремі СУБД виконують також функцію підтримки спеціалізованих мов програмування, що мають загальну назву «мови баз даних». Наприклад, СУБД Access 2016 підтримує мову запитів SQL. Отже, для створення якісних БД і кваліфікованої роботи з ними необхідно добре опанувати СУБД.

СУБД класифікують за багатьма ознаками. До найголовніших можна віднести *призначення*, *модель даних*, *спосіб доступу*. Спрощену схему класифікації СУБД подано на рис. 1.2.

Нині фактичним стандартом мови БД є мова SQL. Однак у деяких випадках доводиться користуватися й іншими мовами програмування, наприклад мовою VBA. Розробники БД засобами СУБД та іншими мовами програмування можуть розробляти прикладні програми, за допомогою яких користувач натисканням однієї кнопки може отримати з БД необхідні дані або опрацювати їх за певним алгоритмом. Наприклад, обчислити суму реалізованих у супермаркеті певних товарів за добу, нарахувати заробітну платню працівникам фірми або отримати інформацію про наявність вільних місць у готелях міста Відня, що не дорожчі ніж 200 євро за добу.

Рис. 1.2. Класифікація систем управління базами даних

Отже, взаємодія користувача з БД може здійснюватися як засобами СУБД, так і за допомогою прикладних програм, що пояснюється схемою (рис. 1.3).

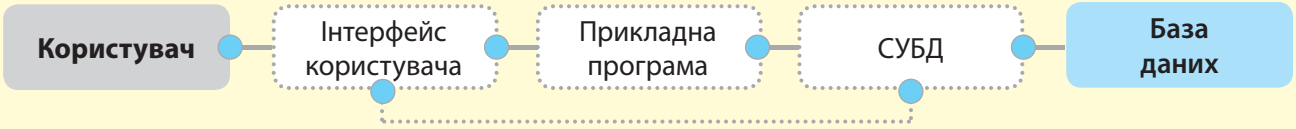


Рис. 1.3. Варіанти взаємодії користувача з базою даних

Приклад 2. Якщо в системі продажу квитків на потяг замовлення на квиток із будь-яких причин не виконано, у БД жодних змін щодо наявності квитків внесено не буде, тобто відбудеться відкат.

Приклад 3. Сутність ХОЛОДИЛЬНИК характеризується такими атрибутами, як назва, маса, ціна, потужність. Конкретний холодильник є екземпляром сутності ХОЛОДИЛЬНИК. Атрибут, що є унікальним, тобто однозначно визначає екземпляр сутності, називають ключем.

Сучасні БД мають величезні обсяги даних і зберігаються в комп'ютерних системах на жорстких магнітних дисках. Користувач позбавлений необхідності знати тонкощі фізичного розміщення даних на них. Ця функція повністю реалізується СУБД разом з операційною системою.

Важливою функцією СУБД є також керування транзакціями. Транзакція — це послідовність операцій над даними, яка сприймається СУБД як єдине ціле.

Якщо всі операції з послідовності виконано успішно, то вважається, що й транзакцію завершено успішно. Усі зміни даних, виконані за цією транзакцією, вносяться в зовнішню пам'ять. Та якщо хоча б одну операцію послідовності завершено невдало, транзакція вважається невиконаною і здійснюється відкат — скасування змін у всіх даних, виконаних у процесі транзакції, та повернення БД до початкового стану виконання транзакції (приклад 2).

Ще однією важливою функцією СУБД є так звана журналізація, під якою розуміють облік уведених у БД змін. Перед виконанням потрібних змін їх вносять до спеціального журналу. У разі апаратного або програмного збою БД можна повністю відновити за допомогою архівної копії БД і журналу.

Одним із засобів моделювання предметної галузі на етапі проектування БД є модель «сутність — зв'язок». Основними поняттями такої моделі є *сутність*, *атрибут* і *зв'язок*.

Сутність — це деякий об'єкт реального світу. Вона має екземпляри, які відрізняються один від одного значеннями атрибутів. Атрибут — це властивість сутності. Зв'язок фактично встановлює взаємодію між сутностями (приклад 3).

У реляційних БД сутності відповідає таблиця, а екземпляру — запис.

Запитання для перевірки знань

1. Що називають предметною областю БД?
2. Наведіть приклади властивостей об'єкта **смартфон**.
3. Як позначають об'єкти БД?
4. Як поділяються БД за структурою?
5. Які БД називають фактографічними?
6. Наведіть означення БД.
7. Назвіть основні функції СУБД.
8. Поясніть сутність транзакції.

1.2. Поняття моделі даних

Пригадайте означення моделі та моделювання. Що, на вашу думку, означає термін «модель даних»?

Як вже зазначалося, об'єкти предметної галузі характеризуються сукупністю атрибутів (властивостей) та їх значеннями. Одне значення атрибуту називають елементарною одиницею даних. Наприклад, для об'єкта АВТОМОБІЛЬ його елементарними одиницями можуть бути марка — Volkswagen і двигун — дизельний.

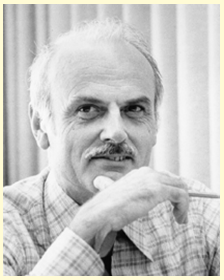
Таким чином, об'єкти БД характеризуються сукупністю елементарних одиниць даних, між якими повинні бути встановлені однозначні зв'язки. Це означає, що основою будь-якої структури даних є відображення елементарної одиниці даних у вигляді трійки: <об'єкт, атрибут об'єкта, значення атрибута>, наприклад: <учень, прізвище, Костирко>; <учень, клас, 11>.

Дані, що зберігаються в БД, мають певну логічну структуру, тобто **описуються деякою моделлю даних**, яка підтримується відповідною СУБД.

Існують різні способи відображення зв'язків між даними, тобто різні моделі даних. Нині є три класичні моделі даних: *ієрархічна*, *мережева* і *реляційна*. Розвиваються й інші моделі даних, засновані на класичних, наприклад *об'єктно-реляційна*.

Таким чином, модель даних визначає, як відбувається об'єднання даних у структури. Вона також визначає можливі операції над даними й обмеження на їх значення.

Ієрархічна і мережева моделі засновані на таких поняттях, як *рівень*, *вузол*, *зв'язок*. Приклад структури і стислий опис сутності цих моделей подано на рис. 1.4.



У 1970-х роках американський математик Е. Кодд розробив теоретичні основи реляційної моделі даних. У 1981 році за вагомий внесок у теорію і практику створення реляційних БД учений отримав премію Тюрінґа.

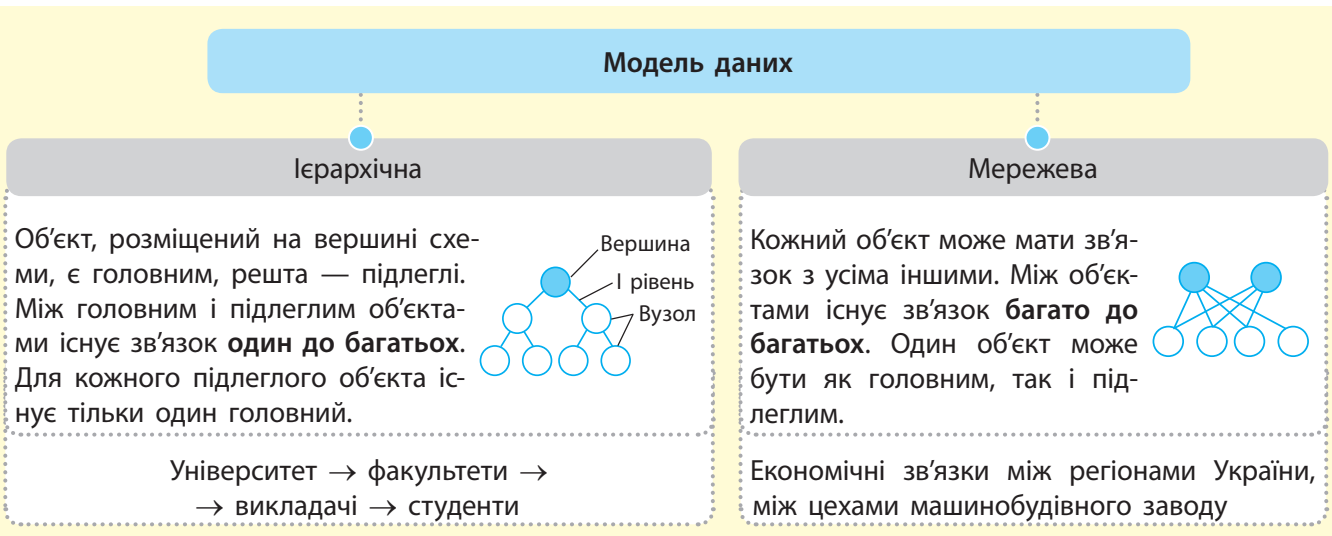


Рис. 1.4. Структури ієрархічної і мережевої моделей даних

Із рис. 1.4 видно, що в цьому випадку *ієрархічна* модель містить три рівні об'єктів. На верхньому рівні міститься головний об'єкт, на другому рівні розташовано два вузли, на третьому — три вузли. Зв'язки між вузлами зображено стрілками. Як бачимо, вузли верхнього рівня мають зв'язки з вузлами найближчого нижнього рівня. *Мережева* модель містить два рівні (їх може бути скільки завгодно), на кожному з яких є два вузли. Звернемо увагу на те, що в цій моделі кожний вузол може мати зв'язок із будь-яким іншим вузлом.

Основним недоліком ієрархічних і мережевих БД є складність їх розроблення, тому нині поширення набула реляційна модель даних — фактографічна база даних, що є набором взаємопов'язаних таблиць. Основна перевага цієї моделі полягає у простоті розроблення БД і систем управління ними.

Найпростіша БД містить одну таблицю, а складні — десятки й навіть сотні таблиць. Розглянемо приклад найпростішої реляційної БД, яка містить тільки одну таблицю УЧНІ (табл. 1.1).

У реляційних моделях об'єкти і взаємозв'язки між даними подаються за допомогою відношень. Порядок розміщення рядків і стовпців у таблиці є довільним. Таблиці в теорії БД називають **відношеннями**, рядки — **записами**, а стовпці — **полями**.

Таблиця 1.1. УЧНІ

Номер	Прізвище	Дата народження	Адреса	Клас	Зріст, см
1	Колот А. І.	07.02.2002	Зоряна, 2, кв. 7	10	172
2	Таранов С. О.	02.06.2003	Поштова, 3, кв. 9	9	174
3	Федорчук Ю. А.	30.05.2003	Лісова, 5	9	165

Не кожна таблиця може бути об'єктом БД. Для того щоб таблиця стала об'єктом БД, потрібно виконати їх нормалізацію. Сутність нормалізації полягає в тому, що таблиця повинна бути перетворена відповідно до основних вимог.

Основні вимоги до таблиці як об'єкта БД такі:

- кожне поле повинно мати унікальне ім'я;
- усі поля мають бути однорідними, тобто значення елементів одного поля можуть бути лише одного типу (наприклад, тільки числовими, тільки рядковими);
- у таблиці не може бути однакових записів, вони мають відрізнятися значеннями хоча б одного поля;
- таблиця повинна мати ключове поле, або ключ.

Зазвичай таблиця має унікальне поле або кілька полів, які ідентифікують записи. Таке поле називають ключовим (ключем). Воно використовується для швидкого пошуку даних, а також для зв'язування даних із різних таблиць.

Ключ, який містить тільки одне поле, називають простим, а який містить кілька полів — складним. У таблиці УЧНІ складним ключем можна вважати поля Прізвище і Дата народження, оскільки вони однозначно ідентифікують записи.

У таблиці може бути кілька ключів, але тільки один із них можна визнати як первинний. Найкраще первинним ключем вибрати простий ключ і бажано, щоб він мав цілочисловий тип. У цьому випадку операції опрацювання даних виконуватимуться швидше. У таблиці УЧНІ простим є поле з іменем Номер.

У 1973 році американський вчений Чарльз Бахман отримав премію Тюрінґа за керування роботою Data Base Task Group (робоча група по базах даних, США), яка розробила стандартну мову опису даних і маніпулювання даними.

У таблиці часто використовують поле — воно називається лічильником, яке використовується для того, щоб зробити кожний запис унікальним. Крім того, лічильник забезпечує нумерацію записів. У таблиці УЧНІ лічильником є поле з іменем Номер.

Важливо усвідомити, що на основі однієї таблиці можна створити БД будь-якої складності. Таблиця може містити сотні полів і тисячі записів, і працювати з нею досить складно. Щоб не сталося значного дублювання даних, для кожного об'єкта розробляється власна таблиця. А щоб можна було одночасно отримувати дані з кількох таблиць, потрібно встановлювати зв'язки між ними (приклад 1).

В основній таблиці вибирають *первинний* ключ, а в допоміжній — *зовнішній* ключ. Зовнішній ключ повинен однозначно визначати поле основної таблиці. У ньому не може бути даних, відсутніх у первинному ключі, інакше зв'язок буде некоректним. Часто для забезпечення зв'язку між таблицями в допоміжну таблицю спеціально вводять поле з таким самим іменем, що й ім'я первинного ключа основної таблиці. У такому випадку деякі СУБД автоматично встановлюють зв'язок між таблицями. Якщо імена зазначених полів різні, то користувач повинен сам встановити зв'язок між ними. Пояснимо сутність зв'язків між двома таблицями на прикладі 2.

Приклад 1. Для БД фірми в одній таблиці можуть зберігатися дані про співробітників, у другій — дані про їхню заробітну платню, у третій — відомості про постачальників продукції. Такий підхід спрощує подальшу модифікацію БД.

Зв'язки можуть встановлюватися між двома, трьома й більшою кількістю таблиць. Для встановлення зв'язків між двома таблицями одну з них вибирають **основною** (**батьківською**), а другу — **допоміжною** (дочірньою).

Приклад 2. Нехай у БД будівельної компанії є дві таблиці: табл. 1.2 ПОСТАЧАЛЬНИКИ і табл. 1.3 ТОВАРИ.

Таблиця 1.2. ПОСТАЧАЛЬНИКИ

Фірма	Директор	Телефон
РПЗ-1	Сопко А. І.	345-23-51
БУТ-5	Маслов В. М.	295-44-87
ДМК-2	Бондаренко К. О.	454-98-56

Таблиця 1.3. ТОВАРИ

Матеріал	Маса, кг	Фірма
Бетон	100	РПЗ-1
Бетон	120	БУТ-5
Бетон	200	ДМК-2
Цемент	50	БУТ-5

У табл. 1.2 ПОСТАЧАЛЬНИКИ первинним ключем є поле з іменем Фірма. У табл. 1.3 ТОВАРИ поле з цим іменем не може бути первинним ключем, оскільки в ньому повторюються назви фірм. Воно може бути зовнішнім ключем, тому що його значення збігаються зі значеннями однойменного поля табл. 1.2 ПОСТАЧАЛЬНИКИ. Більше того, вони мають однакове ім'я. За даними цього поля можна встановити зв'язок між двома таблицями.

Щоб дізнатися телефон і прізвище директора фірми, яка постачає 120 тонн бетону і 50 тонн цементу, із табл. 1.3 ТОВАРИ слід вибрати назву фірми БУТ-5 і за її назвою у табл. 1.2 ПОСТАЧАЛЬНИКИ знайти прізвище Маслов В. М., телефон 295-44-87.

Таким чином, зв'язки між таблицями дозволяють отримати дані з кількох таблиць. Окрім того, вони забезпечують цілісність даних у пов'язаних таблицях, якщо з деяких причин сталися зміни в одній таблиці.

Пояснимо сутність цілісності даних на прикладі 3 вже розглянутих таблиць.

Приклад 3. Припустимо, що зв'язок між табл. 1.2 ПОСТАЧАЛЬНИКИ і табл. 1.3 ТОВАРИ не встановлено. Із табл. 1.2 випадковим чином вилучено запис про те, що директором фірми є Маслов В. М., а в табл. 1.3 всі дані збереглися, тобто є цілісними. Ця ситуація відображена у табл. 1.4 ПОСТАЧАЛЬНИКИ.

Тепер прізвище директора фірми БУТ-5 і його телефон невідомі. Вважається, що у цьому випадку трапилося порушення цілісності даних, і ситуація має бути автоматично виявлена.

Таблиця 1.4. ПОСТАЧАЛЬНИКИ

Фірма	Директор	Телефон
РПЗ-1	Сопко А. І.	345-23-51
ДМК-2	Бондаренко К. О.	454-98-56

Між таблицями можуть існувати 4 види зв'язку (рис. 1.5).

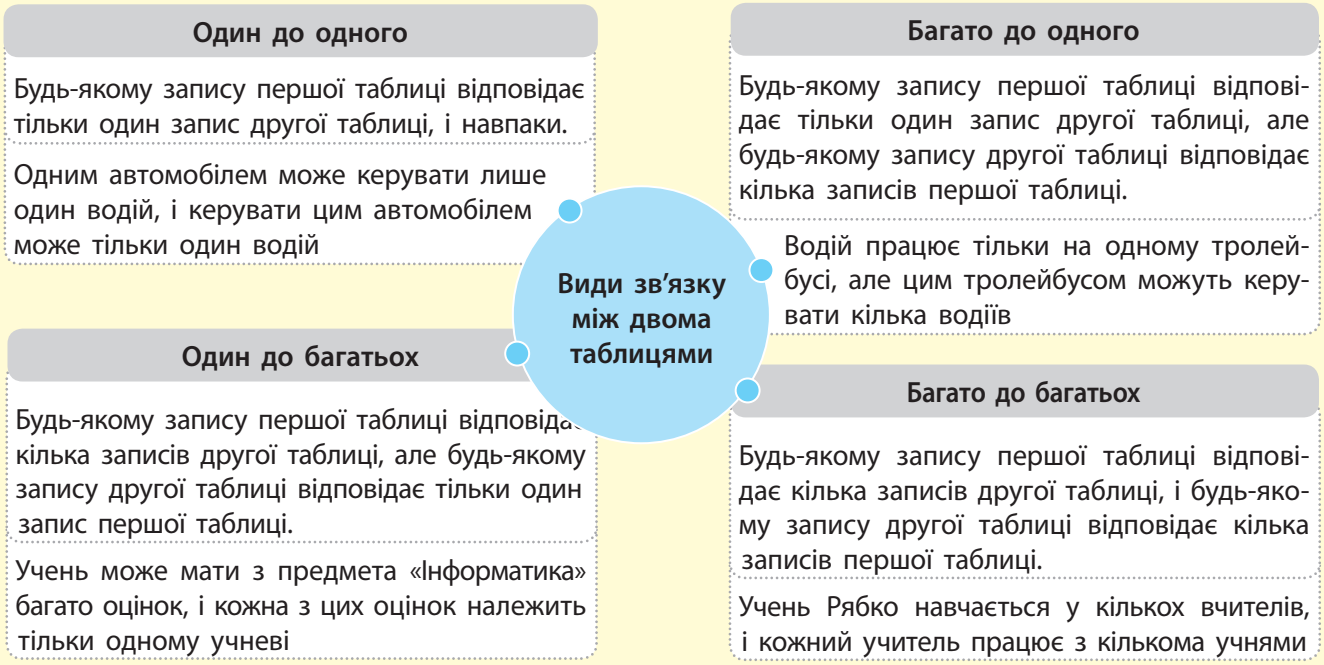


Рис.1.5. Види зв'язку між двома таблицями

Найчастіше між таблицями реляційної БД існує зв'язок один до багатьох.

Запитання для перевірки знань

1. Що називають елементарною одиницею даних у БД?
2. Назвіть основні моделі даних у БД.
3. Поясніть сутність ієрархічної моделі даних.
4. Які існують види зв'язку між таблицями?
5. Поясніть сутність реляційної моделі даних.
6. Які поля таблиць називають ключем?
7. Які існують ключі в таблицях БД?
8. Наведіть означення моделі даних.
9. Назвіть основні вимоги до таблиць БД.
10. У чому полягає сутність забезпечення цілісності даних БД?

1.3. Основні відомості про систему управління базами даних Access

СУБД призначені для створення й супроводу БД. Спробуйте конкретизувати їхні основні функції.

Історія розвитку БД і систем управління ними налічує кілька етапів. За цей час розроблено багато СУБД, наприклад dBase, FoxPro, Oracle 8.4, MS SQL Server 7.0, SQL Base, MS Access 7 та ін. Усі вони по-різному працюють із об'єктами і мають різні функціональні можливості. Та попри це більшість із них спирається на єдиний комплекс основних понять, що дає нам можливість розглянути одну систему та узагальнити її поняття, прийоми й методи на весь клас СУБД. Далі розглядатимемо одну з найпоширеніших сьогодні СУБД — Access 2016.

СУБД Access 2016 входить до складу пакета Microsoft Office і призначена для створення й підтримки роботи з реляційними БД. Розглянемо її основні об'єкти та їх призначення (рис. 1.6).

СУБД Access 2016 функціонує під керуванням ОС Windows. Системні вимоги: бажано, щоб процесор мав частоту не менше за 800 МГц, обсяг оперативної пам'яті — не менше за 512 Мб, вільний обсяг пам'яті на жорсткому диску — не менше за 2 Гб.

Рис.1.6. Об'єкти СУБД та їх призначення

Запуск системи Access 2016 можна здійснити стандартними способами, що передбачені в ОС Windows.

Після запуску системи на екрані монітора з'явиться її стартове вікно (рис. 1.7).

На ділянці вікна зліва відображено імена БД, з якими користувач працював останнім часом, на ділянці справа — піктограми шаблонів і піктограма порожньої БД.

У середовищі Access 2016 БД можна створити «з нуля», тобто повністю самостійно, або скористатися шаблонами, які має система. Якщо наявних шаблонів не вистачає, їх можна знайти в Інтернеті, скориставшись полем пошуку. Для якісного оволодіння способами створення й супроводу БД

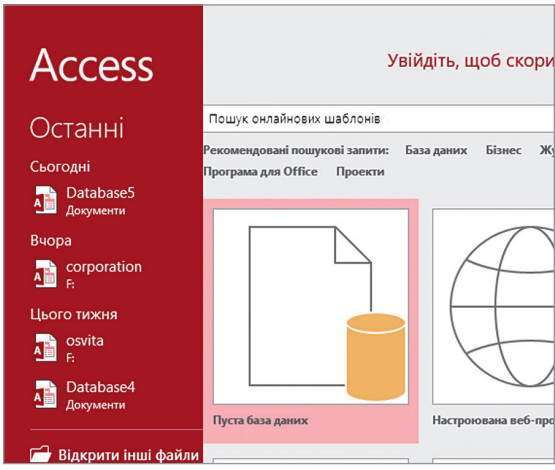


Рис. 1.7. Стартове вікно Access 2016

користувачу-початківцю доцільно спочатку навчитися створювати нову БД.

Серед шаблонів у Access є Пуста база даних, яка слугує для створення нової БД. У подальшому ми будемо використовувати саме цей спосіб.

Розглянемо порядок дій для створення нової БД.

Крок 1

Клацнути піктограму **Пуста база даних**.
Відкриється вікно, зображене на рис. 1.8.

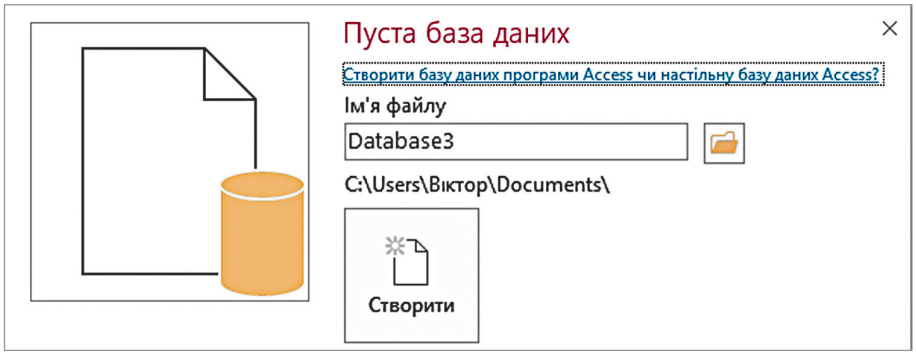


Рис. 1.8. Вікно для створення нової бази даних

Крок 2

У рядок **Ім'я файлу** ввести ім'я файла майбутньої БД, наприклад abc, натиснути кнопку **Знайти розташування для бази даних**, що розташована праворуч від цього рядка.

Крок 3

У вікні **Створення бази даних**, що відкриється, вибрати місце збереження файла БД, наприклад диск F:, і натиснути кнопку **ОК**, а потім — кнопку **Створити**.

У результаті цих дій файл БД буде зареєстровано в кореневому каталозі диска F:, а на екрані з'явиться вікно для створення таблиці 1 (рис. 1.9).

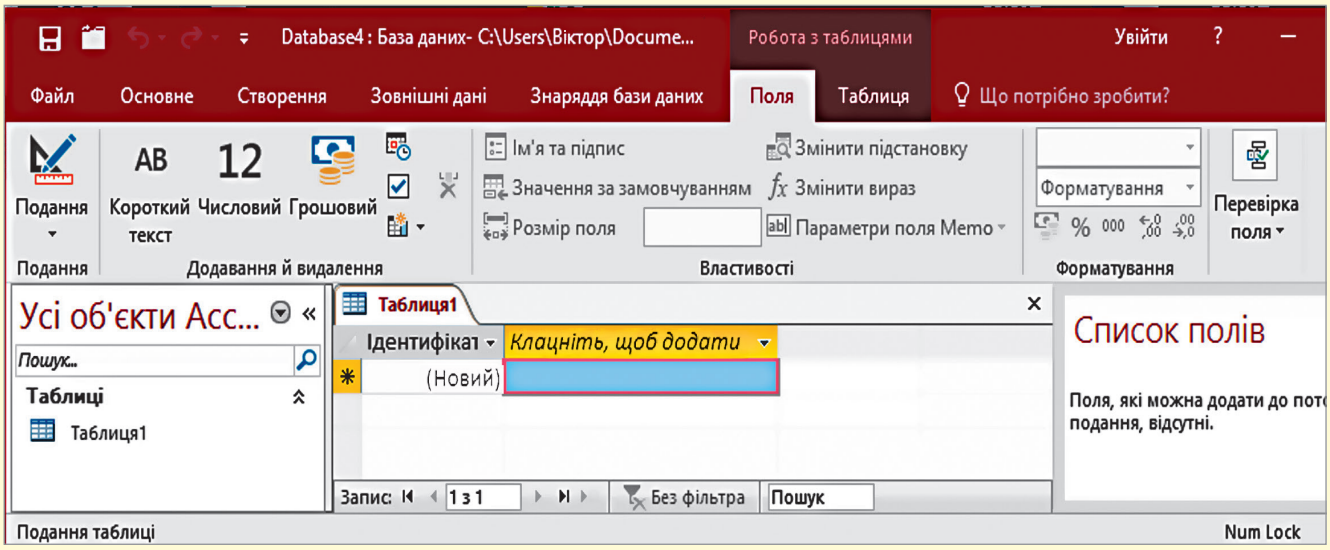


Рис. 1.9. Вікно для створення таблиці

Основним об'єктом вікна є горизонтальна стрічка, на якій розташовано команди й інструменти Access 2016. Їх призначення будемо розглядати поступово під час безпосереднього використання. Розглянемо ті, які потрібні на цьому етапі.

У верхній частині стрічки розміщено вкладки Файл, Основне, Створення та ін. Зміст команд та елементів керування, які відображені на стрічці, залежить від того, яку на цей момент вкладку відкрито.

На рис. 1.9 відкрито вкладку Поля, і в цьому випадку всі команди й елементи керування згруповані у три розділи: Подання, Додавання й видалення, Форматування. Якщо відкрити іншу вкладку, наприклад Основне, з'являться нові елементи й команди, які буде згруповано в нові розділи.

Стисло ознайомимося з призначенням вкладок вікна.

Вкладка Основне містить команди й елементи керування, які найчастіше використовуються в процесі роботи з БД. Зокрема тут містяться команди для роботи з буфером обміну, форматування тексту, сортування й фільтрування даних та ін.

Вкладка Створення містить команди для створення таблиць, запитів, форм та інших об'єктів БД, вкладка Зовнішні дані — команди для експортування й імпортування даних, вкладка Знаряддя бази даних — команди встановлення зв'язків між таблицями, аналізування й переміщення даних між програмами та ін.

У лівій частині екрана вміщено Усі об'єкти Access — панель переходів, на якій можуть відображатися назви всіх створених об'єктів, між якими можна здійснювати перехід простим натисненням відповідних назв. Праворуч від панелі переходів міститься область редагування, у якій можуть одночасно відображатися таблиці, запити та інші об'єкти БД.

Відкрити вже створену БД можна за допомогою кнопки Відкрити на панелі швидкого доступу, а щоб закрити, потрібно скористатися командою Закрити, що розташована на вкладці Файл.

У кожний момент часу Access 2016 підтримує роботу тільки з однією БД. Число користувачів, які одночасно працюють із БД, може досягати 255. Імена об'єктів можуть включати до 64 символів, максимальний обсяг файла БД становить 2 Гб.

Щоб згорнути й розгорнути стрічку, слід скористатися відповідними командами кнопки **Настроїти панель швидкого доступу**, яка міститься на панелі швидкого доступу, або натиснути праву кнопку миші в будь-якому місці та виконати команду **Згорнути стрічку** в контекстному меню, що з'явиться.

Запитання для перевірки знань

1. Назвіть основні об'єкти Access 2016 та їх призначення.
2. Що розміщено в області **Усі об'єкти Access**?
3. Як можна відкрити вже створену БД?
4. Поясніть сутність створення БД «з нуля».
5. Які основні дії можна виконувати на вкладці **Створення**?
6. Які основні системні вимоги для Access 2016?

Завдання для самостійного виконання

1. Запустіть систему Access 2016. Створіть на жорсткому диску файл БД з іменем **Mybasa**. Переконайтеся, що файл зареєстровано.
2. Проаналізуйте призначення об'єктів початкового вікна БД, відкриваючи різні вкладки.
3. Відкрийте та проаналізуйте призначення команд кнопки **Настроїти панель швидкого доступу.**
4. Відкрийте і проаналізуйте вміст вкладки **Основне**.

2. Таблиці

Створюючи БД Access, користувачі зберігають дані в таблицях. Об'єкти БД залежать від структури таблиць, тому розробку БД потрібно починати зі створення власне таблиць і лише після цього можна переходити до будь-яких інших об'єктів.

2.1. Створення й уведення структури таблиць

Поміркуйте, з якою метою застосовуються таблиці в БД. Якою є основна функція таблиць?

У середовищі Access 2016 існують такі інструменти створення таблиць, як **Конструктор таблиць**, **Майстер таблиць**, **Режим таблиць**.

Таблиці в середовищі Access 2016 можна створювати різними способами. Та найчастіше їх створюють розробники БД — починаючи від проектування на папері й закінчуючи створенням електронних працездатних таблиць. Саме такий варіант ми розглянемо далі.

Розробці таблиць передує детальний аналіз предметної області, визначення вимог, змісту та структури документів, які необхідно отримати. У процесі розроблення слід визначити кількість таблиць, а також назву, структуру і вміст кожної таблиці. Далі порядок розроблення й опрацювання таблиць розглянемо на прикладі предметної галузі Магазини.

Приклад 1. Припустимо, що БД деякої мережі магазинів містить дві таблиці: табл. 2.1 МАГАЗИНИ і табл. 2.2 КАДРИ. Визначимо їх структуру і вміст.

Таблиця 2.1. МАГАЗИНИ

Номер магазину	Адреса	Директор	Телефон	Працівників
21	вул. Паркова, 33	Коцюба П. М.	234-54-63	20
31	вул. Печерська, 21	Борзов А. С.	234-22-98	13
6	вул. Річкова, 24	Середа К. М.	234-67-92	15

Таблиця 2.2. КАДРИ

Номер справи	Прізвище	Посада	Рік народження	Освіта	Стаж	Оклад	Номер магазину
105	Сокіл Т. Л.	Касир	1960	Середня	27	3500	6
132	Таран В. Д.	Диспетчер	1973	Вища	15	4000	31
120	Рябко Р. П.	Експерт	1981	Вища	8	4200	21
111	Семко М. М.	Диспетчер	1970	Середня	16	4000	21
115	Горошко Ф. Р.	Диспетчер	1975	Середня спеціальна	17	4000	31
116	Раков Г. П.	Аналітик	1965	Вища	19	4500	21
109	Шрамко Т. Л.	Диспетчер	1961	Середня	24	4000	6

Головною таблицею вважатимемо табл. 2.1 МАГАЗИНИ, а допоміжною — табл. 2.2 КАДРИ.

Найпотужнішим і універсальним інструментом створення таблиць є Конструктор таблиць. Далі розглянемо цей спосіб, інші способи простіші, ними можна оволодіти самостійно.

Після визначення структури таблиць слід визначити типи полів з урахуванням тих типів, із якими може працювати Access. У середовищі Access 2016 використовуються типи даних, перелік яких наведено на рис. 2.1.

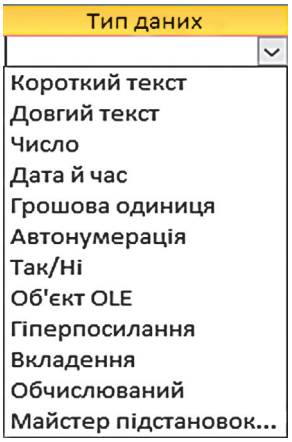


Рис. 2.1. Типи даних Access 2016

Кожен із наведених типів даних має власний набір властивостей. Деякі властивості є унікальними, тобто містяться тільки в одному конкретному типі даних, а деякі є загальними, тобто містяться в різних типах даних. Далі будемо використовувати в основному такі типи даних: Короткий текст, Довгий текст, Число, Дата й час.

Тип Короткий текст — це послідовність символів завдовжки від 0 до 255, а тип Довгий текст — послідовність символів до 65 536. У полі Число зберігаються числа, їх розмір і конкретний тип визначаються значенням властивості Розмір поля. У полях типу Дата й час зберігаються дата й час різних форматів.

Приклад 2. Для більш зручної роботи під час уведення структури табл. 2.1 і табл. 2.2 доцільно створити на папері додаткові таблиці (табл. 2.3 і табл. 2.4) з іменами полів, зазначенням типу даних та описом.

Таблиця 2.3. Структура таблиці МАГАЗИНИ

Ім'я поля	Тип даних	Опис	Властивості
Номер магазину	Число	Первинний ключ	
Адреса	Короткий текст		50
Директор	Короткий текст		40
Телефон	Короткий текст		20
Працівників	Число	Станом на 1 липня	

Таблиця 2.4. Структура таблиці КАДРИ

Ім'я поля	Тип даних	Опис	Властивості
Справа	Число	Первинний ключ	
Прізвище	Короткий текст		40
Посада	Короткий текст		30
Рік народження	Число		
Освіта	Короткий текст		30
Стаж	Число	Станом на 1 січня	
Оклад	Число		
Номер магазину	Число		

На цьому підготовку таблиць для введення в комп'ютер завершено. Перейдемо безпосередньо до створення структур таблиць у режимі конструктора і розглянемо приклад 3.

Приклад 3

1. Запустимо систему Access 2016 і відкриємо вже створену БД з іменем abc. Для цього в стартовому вікні системи в області Останні натиснемо кнопку миші на імені abc.
2. У вікні, що відкриється, активуємо вкладку Створення. Відкриється вікно зі стрічкою (рис. 2.2).
3. На стрічці натиснемо кнопку Конструктор таблиць. У результаті до БД додасться порожня таблиця (рис. 2.3).

Зверніть увагу на те, що вміст стрічки змінився, і тепер на екрані відкрито вкладку Конструктор (рис. 2.4).

4. Введемо ім'я поля в порожню таблицю (див. рис. 2.3) стандартним способом.

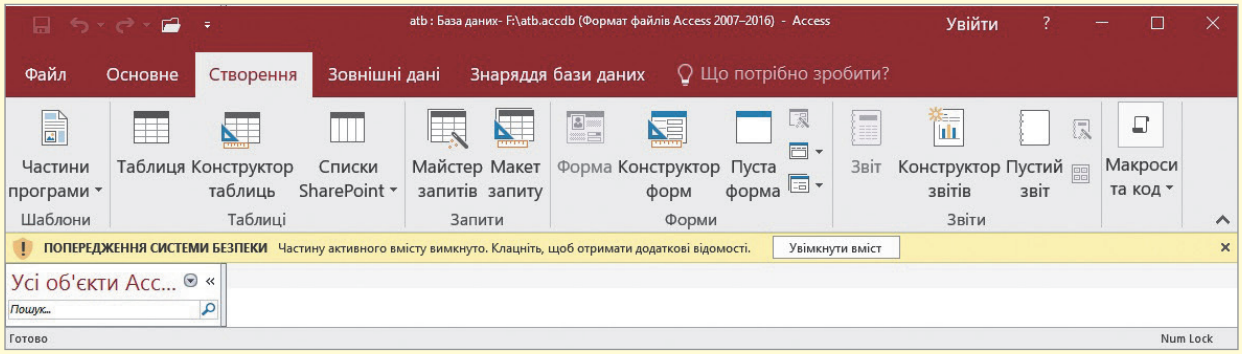


Рис. 2.2. Вікно Access з активованою вкладкою **Створення**

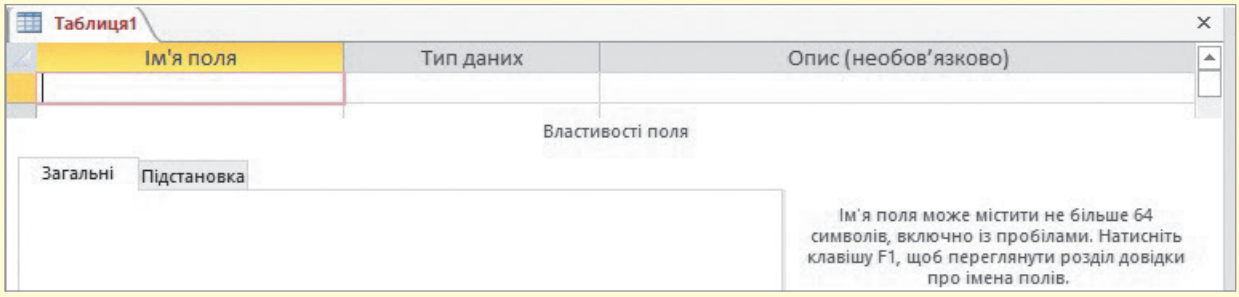


Рис. 2.3. Порожня таблиця в режимі конструктора

Рис. 2.4. Стрічка з активованою вкладкою **Конструктор**

5. Введемо типи полів. Їх краще не вводити з клавіатури, а вибрати зі списку типів (див. рис. 2.1). Щоб відкрити цей список, необхідно у певній клітинці поля Тип даних клацнути кнопку Прапорець, вибрати потрібний тип даних і встановити його властивість. Перелік властивостей наведено нижче від імен полів (рис. 2.5).

Описувати поля не обов'язково. Опис потрібен для уточнення призначення поля та його допустимих значень.

6. Уведемо у відкриту на екрані порожню таблицю (див. рис. 2.3) дані з табл. 2.3 Структура таблиці МАГАЗИНИ. Вміст таблиці набуде вигляду, як наведено на рис. 2.6.

Властивості поля

Загальні | Підстановка

Розмір поля	40
Формат	
Маска вводу	
Підпис	
Значення за промовчан	
Правило перевірки	
Текст перевірки	
Обов'язково	Ні
Дозволити нульову дов	Так
Індексовано	Ні
Стискання Юнікод	Так
Режим редактора IME	Без елемента керування
Режим речення редактс	Немає
Вирівнювання тексту	Загальне

Рис. 2.5. Властивості полів

МАГАЗИНИ

Ім'я поля	Тип даних	Опис (необов'язково)
Номер магазину	Число	Первинний ключ
Адреса	Короткий текст	
Директор	Короткий текст	
Телефон	Короткий текст	
Працівників	Число	станом на 1 липня

Рис. 2.6. Структура таблиці в режимі **Конструктора**

7. Збережемо таблицю з іменем МАГАЗИНИ. Для цього на панелі швидкого доступу натиснемо кнопку Зберегти (або скористаємося сполученням клавіш Ctrl + S) — відкриється віконце (рис. 2.7). Уведемо в нього ім'я МАГАЗИНИ і клацнемо кнопку OK.

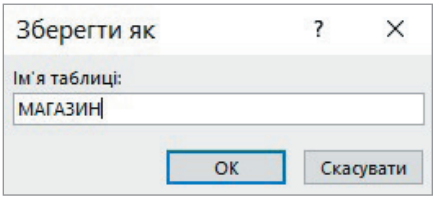


Рис. 2.7. Віконце для збереження таблиці

На екран буде виведено попередження (рис. 2.8).

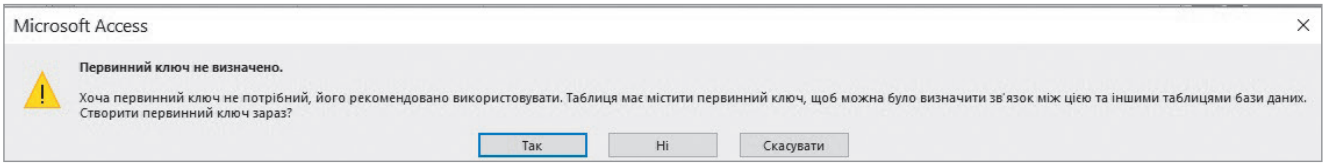


Рис. 2.8. Вікно попередження про невизначений первинний ключ

8. Клацнувши кнопку Так, ми зумовимо створення первинного ключа типу Лічильник. Але на цьому кроці встановлювати первинний ключ не обов'язково, тому клацнемо кнопку Ні. Таблицю буде збережено, а її ім'я з'явиться в області переходів.

9. Закриємо таблицю МАГАЗИНИ, для чого відкриємо її контекстне меню і виконаємо команду Закрити.

10. Аналогічно введемо і збережемо структуру таблиці КАДРИ. Після цього закриємо таблицю.

Зазначимо, що властивості таблиць установлюються у вікні Аркуш властивостей, яке для відкритої таблиці доступне на вкладці Конструктор. У цьому вікні містяться десятки назв властивостей таблиць. Аналогічне вікно властивостей мають і інші об'єкти БД (запити, форми, звіти). Та на цьому етапі вивчення БД ми будемо використовувати властивості, пропоновані за замовчуванням.

Таблицю можна модифікувати на будь-якому кроці створення БД. Але краще це робити до встановлення зв'язків між таблицями і введення в них даних.

Запитання для перевірки знань

1. Опишіть способи створення таблиць.
2. Чому таблиці бажано спочатку створювати на папері?
3. Які дії слід виконати для збереження структури таблиці?
4. Що розуміють під терміном «структура таблиці»?
5. Чому для створення таблиці найчастіше користуються конструктором таблиць?
6. Назвіть основні типи даних таблиць Access.
7. Поясніть порядок уведення структури таблиць Access.
8. Поясніть порядок уведення типу даних поля та його властивостей.

Завдання для самостійного виконання

1. Запустіть СУБД Access 2016 і створіть БД skola з таблицями КЛАСИ (табл. 2.5) й УЧНІ (табл. 2.6).
2. Розробіть на папері структуру табл. 2.5 КЛАСИ і табл. 2.6 УЧНІ. Типи даних полів і їхні властивості виберіть самостійно.
3. Уведіть і збережіть структуру табл. 2.5 КЛАСИ і табл. 2.6 УЧНІ.

Зі створеними БД ви працюватимете протягом періоду їх вивчення, тому працюйте акуратно, не забувайте зберігати, не порушуйте структуру таблиць.

Таблиця 2.5. КЛАСИ

Клас	Учнів	Класний керівник	Займаються спортом
9	27	Зотов А. М.	5
10	24	Дерев'янко Н. С.	11
11	25	Терещенко Б. В.	14

Таблиця 2.6. УЧНІ

Прізвище	Адреса	Дата народження	Зріст	Улюблений предмет	Інформатика	Історія	Клас
Ларін Л. К.	Лугова, 5	5.06.2002	163	Математика	11	8	9
Лобов С. П.	Сонячна, 7	9.08.2002	168	Географія	9	9	9
Костенко В. С.	Річкова, 4	13.09.2001	170	Фізика	10	10	10
Рамко Б. В.	Лугова, 9	15.11.2001	169	Інформатика	11	8	10
Пека П. О.	Сонячна, 7	20.12.2001	167	Інформатика	12	9	10
Сергіна В. В.	Лісна, 5	3.03.2002	171	Історія	8	11	10
Хижа Р. А.	Лугова, 2	4.04.2002	172	Історія	9	12	11
Собко О. К.	Вишнева, 5	9.10.2001	175	Інформатика	11	8	11
Настін К. Б.	Лісна, 10	6.11.2001	171	Географія	10	10	11

2.2. Ключові поля, індекси, зв'язування таблиць

Одне з полів таблиці може мати унікальні значення. Для чого, на вашу думку, може використовуватися таке поле?

Згадаємо, що кожна таблиця повинна мати ключове поле (ключ) — поле, значення якого не повторюється в жодному іншому записі. Таблиця може мати кілька ключових полів, але використовується тільки одне з них, яке називають первинним ключем.

Найчастіше первинний ключ складається з одного поля, а як первинний ключ використовується поле типу Лічильник. Якщо в ролі первинного ключа використовуються два і більше полів, його називають складним. Наприклад, у таблиці КАДРИ поле Прізвище не може бути первинним ключем, тому що в мережі магазинів може бути працівник із таким самим прізвищем. А поля Прізвище і Рік народження разом можна вважати таким ключем, оскільки вони, ймовірно, не дублюються.

Ключі у БД відіграють важливу роль — за їх допомогою СУБД ідентифікує об'єкти.

Для створення первинного ключа потрібно відкрити таблицю в режимі конструктора, виділити поле, що використовується як первинний ключ, і натиснути кнопку Ключове поле, що знаходиться в розділі Знаряддя вкладки Конструктор.

Складні ключі бажано не використовувати як первинний ключ, оскільки в цьому випадку ускладнюється процес роботи з БД.

Створити первинний ключ можна також за допомогою контекстного меню певного поля, у якому необхідно виконати команду Ключове поле. Для цього слід відкрити таблицю в режимі конструктора. Скористаємося цим способом, і в таблиці МАГАЗИНИ визначимо як первинний ключ поле Номер магазину. Поряд із назвою цього поля з'явиться зображення ключа (рис. 2.9). Далі збережемо таблицю.

МАГАЗИНИ

Ім'я поля	Тип даних	Опис (необов'язково)
Номер магазину	Число	Первинний ключ
Адреса	Короткий текст	
Директор	Короткий текст	
Телефон	Короткий текст	
Працівників	Число	станом на 1 липня

Рис. 2.9. Структура таблиці МАГАЗИНИ з ключовим полем

Якщо деяке поле в процесі створення структури таблиці оголошено типу Автонумерація (див. рис. 2.1), тобто типу Лічильник, то воно стає ключовим за замовчуванням. Його можна також додати в таблицю навіть у тому випадку, якщо явної потреби в цьому немає.

Поле типу **Лічильник** обов'язково встановлюється в тому разі, якщо ключ у таблиці взагалі визначити неможливо.

Розглянемо тепер сутність і порядок індексування таблиць.

Індексування — це процес створення додаткових таблиць для певного поля. Ці таблиці (їх ще називають простими індексними таблицями) зазвичай містять лише одне поле, у якому зберігаються вказівники на певні записи таблиці. За допомогою вказівників визначають порядок розміщення записів, упорядкованих за значенням цього поля (приклад 1).

Приклад 1. Індексна таблиця для поля Директор таблиці МАГАЗИНИ матиме такі значення:

2
1
3

Цифра 2 в першому рядку означає, що першим за алфавітом у таблиці є друге прізвище (Борзов А. С.), цифра 1 — що другим за алфавітом є перше прізвище (Коцюба П. М.), а цифра 3 — що третім за алфавітом є третє прізвище (Середа К. М.).

Індексних таблиць для певної таблиці БД може бути кілька, наприклад за кількістю працівників, за номерами магазинів тощо.

Головним призначенням індексних таблиць є підвищення швидкості пошуку необхідних даних (інколи вона зростає до 5 разів). Щоб знайти деякий запис у таблиці, в Access спочатку потрібно знайти його положення в індексі, потім вибрати з нього місце запису в таблиці, що використовується для пошуку даних.

Поля, значення яких змінюються часто, індексувати недоцільно. Для однієї таблиці бажано мати не більш ніж 5 або 6 індексних таблиць.

Зазначимо, що первинний ключ завжди індексований. За замовчуванням записи таблиці виводяться відсортованими за його значеннями. У процесі введення даних у таблицю обов'язково перевіряється значення первинного ключа на дублювання. Якщо значення дублюється, введення запису блокується. Значення первинного ключа типу Лічильник у процесі введення даних формується автоматично (приклад 2).

Приклад 2. Створимо просту індексну таблицю для полів Директор і Працівників таблиці МАГАЗИНИ. Для цього відкриємо таблицю в режимі конструктора, виберемо поле Директор і в розділі Властивості поля в рядку Індексовано ввімкнемо перемикач Так (Без повторень), оскільки малоймовірно, що в цій мережі магазинів буде два директори з однаковими прізвищами. Для поля Працівників увімкнемо перемикач Так (Повторення дозволені), тому що в магазинах може бути однакова кількість працівників.

Основна вимога до ключів така: значення зовнішнього ключа мають збігатися зі значеннями первинного ключа головної таблиці. Імена цих ключів можуть бути різними, але якщо імена однакові, то процес зв'язування таблиць буде простішим.

Під час зв'язування двох таблиць одна з них вважається головною, а інша — допоміжною. Первинний ключ головної таблиці зв'язується із зовнішнім ключем допоміжної.

Часто первинний ключ таблиці штучно вводять у другу таблицю саме з метою їх зв'язування. Цей ключ не є первинним ключем другої таблиці, тому що його значення можуть повторюватися. Наприклад, поле Магазин є первинним ключем таблиці МАГАЗИНИ, а в таблиці КАДРИ це поле є зовнішнім ключем, тому що в ній значення цього поля дублюються.

Приклад 3. Розглянемо порядок створення зв'язку на прикладі таблиць МАГАЗИНИ і КАДРИ (табл. 2.1 і 2.2 відповідно).

1. Завантажимо БД abc й у вікні, що відкриється, активуємо вкладку Знаряддя бази даних. Далі натиснемо кнопку Зв'язки. Відкриється вікно Відображення таблиці (рис. 2.10).
2. У вікні Відображення таблиці виберемо таблиці, які потрібно зв'язати (у нашому випадку обидві таблиці), і натиснемо кнопку Додати. На екрані з'являться ці таблиці з іменами їх полів.
3. Установимо курсор на первинному ключі таблиці МАГАЗИНИ, натиснемо кнопку миші і, не відпускаючи її, перемістимо курсор у поле зовнішнього ключа й відпустимо кнопку. У результаті відкриється вікно Редагування зв'язків (рис. 2.11).
4. Увімкнемо прапорець Забезпечення цілісності даних. Після цього стануть доступними прапорці Каскадне оновлення пов'язаних полів і Каскадне видалення пов'язаних полів. Увімкнемо останній прапорець.

Увімкнення прапорця Забезпечення цілісності даних дає змогу зберегти цілісність даних. Якщо цей прапорець вимкнутий, то в таблиці можна додавати нові записи, змінювати ключові поля й вилучати пов'язані записи без попередження про порушення цілісності.

Сутність каскадного оновлення пов'язаних полів полягає в тому, що за будь-якої зміни первинного ключа в головній таблиці автоматично оновиться значення відповідного поля у всіх зв'язаних таблицях.

Сутність каскадного вилучення пов'язаних полів полягає в тому, що під час вилучення будь-якого запису з головної таблиці автоматично вилучаються зв'язані записи у зв'язаній таблиці. Отже, каскадне оновлення й каскадне вилучення прискорюють роботу з БД і сприяють підвищенню надійності її функціонування.

5. У вікні Редагування зв'язків натиснемо кнопку Створити, у результаті чого у вікні Звяз'ки з'явиться лінія зв'язку між певними полями таблиць (рис. 2.12). Збережемо БД.

У вікні Редагування зв'язків можна вилучити встановлений зв'язок за допомогою кнопки Скасувати. За допомогою кнопки Створити... можна відкрити нове вікно й установити зв'язок заново.

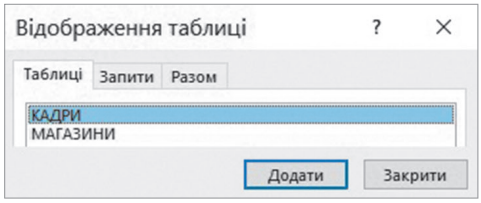


Рис. 2.10. Вікно з переліком таблиць бази даних **abc**

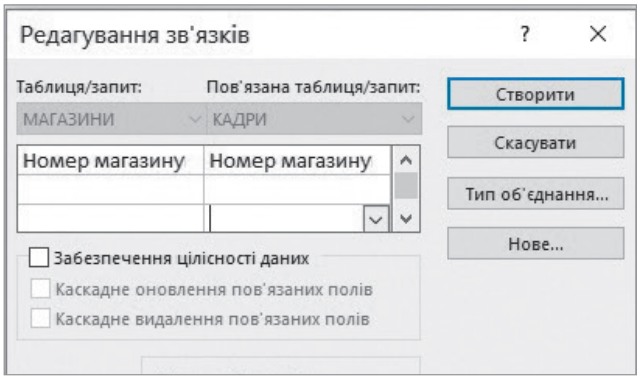


Рис. 2.11. Вікно **Редагування зв'язків**

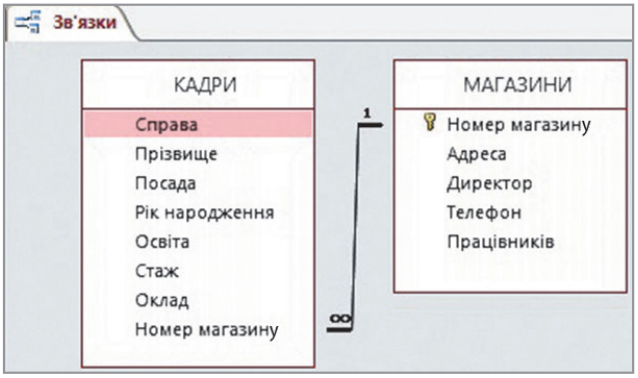


Рис. 2.12. Зв'язок типу **один до багатьох** між таблицями

6. Повернемося до вікна Редагування зв'язків (якщо в цей момент його немає на екрані, слід виконати команду Змінити зв'язки в області Знаряддя — і вікно відкриється). Далі натиснемо кнопку Тип об'єднання... З'явиться вікно Параметри об'єднання (рис. 2.13).
7. За замовчуванням установлюється перший тип об'єднання, який називають об'єднанням за еквівалентністю. Зазвичай розробники БД установлюють відношення за еквівалентністю. Натиснемо в цьому вікні кнопку OK і закриємо вікно Редагування зв'язків.

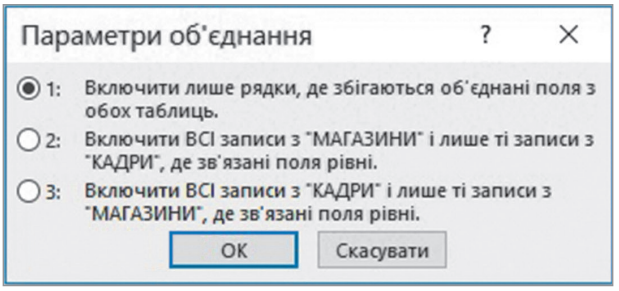


Рис. 2.13. Вікно для встановлення параметрів об'єднання таблиць

Щоб переглянути зв'язки між таблицями БД, необхідно на відкритій вкладці Знаряддя бази даних натиснути кнопку Зв'язки, активувати вкладку Конструктор і в розділі Зв'язок натиснути кнопку Усі зв'язки.

Якщо схема складна, можна приховати частину таблиць і зв'язків, вилучивши їх із вікна Зв'язки. Для цього потрібно виділити певну таблицю й натиснути кнопку Delete. При цьому зв'язки й таблиці вилучаються лише з вікна, фізично ж вони залишаються, тому в будь-який час їх можна відновити.

Щоб скасувати зв'язки між таблицями БД, необхідно встановити курсор на лінії зв'язку, натиснути кнопку миші, а потім — клавішу Delete. Можна також відкрити контекстне меню лінії зв'язку й виконати команду Видалити.

Запитання для перевірки знань

1. Що називають первинним ключем таблиці?
2. Які первинні ключі називають простими; складними?
3. Для чого використовується ключове поле типу **Лічильник**?
4. Який порядок створення ключового поля?
5. Що називають індексуванням таблиць?
6. Як можна скасувати зв'язок між таблицями?
7. З якою метою індексуються таблиці?
8. Для чого потрібно зв'язувати таблиці?
9. Опишіть порядок зв'язування таблиць.

Завдання для самостійного виконання

1. Створіть просту індексну таблицю для поля **Учнів** таблиці **КЛАСИ** і індексну таблицю для поля **Адреса** таблиці **УЧНІ**.
2. Створіть у таблиці **КЛАСИ** БД **skola** первинний ключ і збережіть таблицю. Переконайтеся, що ключ встановлено правильно.
3. Виконайте зв'язування таблиці **КЛАСИ** і таблиці **УЧНІ**. Доведіть, що зв'язок дійсно встановлений. Вилучіть зв'язок між таблицями й установіть його ще раз.

2.3. Введення, пошук і редагування даних у таблиці

Поясніть, у який спосіб можна здійснити пошук необхідних даних у таблиці.

Дані в таблиці можна вводити після створення їх структур. Існує два способи введення даних у таблиці: за допомогою форм і безпосереднє введення даних у таблиці.

Розглянемо алгоритм уведення даних на прикладі 1.

Приклад 1

1. Відкриємо таблицю МАГАЗИНИ в режимі таблиці. Для цього двічі клацнемо кнопкою миші ім'я в області Усі об'єкти. Можна також скористатися контекстним меню цієї таблиці.
2. Уведемо дані першого запису таблиці МАГАЗИНИ, який позначено зірочкою. Щойно дані будуть уведені, курсор автоматично переміститься на наступний запис, що означає готовність до уведення даних у другий рядок. Слід пам'ятати, що вводити в поля можна лише ті типи даних, які збігаються з оголошеним типом поля.
3. У такому самому порядку введемо дані всіх інших записів таблиці й збережемо таблицю. Після введення останнього запису вміст таблиці набуде вигляду, як наведено на рис. 2.14.

Зверніть увагу на те, що записи виведені в іншому порядку, ніж уводилися (записи вводили в порядку, який подано у табл. 2.1).

На рис. 2.14 записи впорядковано в порядку значень ключового поля.

МАГАЗИНИ

Номер магазину	Адреса	Директор	Телефон	Працівникі	Клацніть, щоб додати
6	вул. Річкова, 24	Середа К. М.	234-67-92	15	
21	вул. Паркова, 33	Коцюба П. М.	234-54-63	20	
31	вул. Печерська, 21	Борзов А. С.	234-22-98	13	
0				0	

Запис: 4 з 4 Без фільтра Пошук

Рис. 2.14. Вміст таблиці МАГАЗИНИ

Під час уведення даних автоматично перевіряються такі типи даних: числові, грошові, дата і час, логічні. На вкладці Основне в групі Форматування тексту містяться елементи, за допомогою яких можна змінити розмір і накреслення символів та інші параметри.

Якщо на екрані не поміщаються всі записи, слід скористатися вертикальною смугою прокручування, а якщо не поміщаються всі поля — горизонтальною. Окремі поля можна розширити або звузити звичайним порядком.

У нижній частині вікна таблиці розміщено кнопки навігації для переміщення курсору в перший, сусідній або останній запис.

Для додавання нового запису в таблицю необхідно натиснути кнопку Створити запис на панелі навігації та ввести дані.

Навігацію в таблиці можна здійснювати за допомогою миші, смуг прокручування, сполучення деяких клавіш.

У такому самому порядку введемо дані в таблицю КАДРИ. Її вміст подано на рис. 2.15.

КАДРИ

Справа	Прізвище	Посада	Рік народження	Освіта	Стаж	Оклад	Номер магазину
105	Сокіл Т. Л.	касир	1960	середня	27	3500	6
109	Шрамко Т. Л.	диспетчер	1961	середня	24	4000	6
120	Рябко Р. П.	експерт	1981	вища	8	4200	21
111	Семко М. М.	диспетчер	1970	середня	16	4000	21
116	Раков Г. П.	аналітик	1965	вища	19	4500	21
132	Таран В. Д.	диспетчер	1973	вища	15	4000	31
115	Горошко Ф. Р.	диспетчер	1975	середня спеціальна	17	4000	31
0			0		0	0	0

Запис: 8 з 8 Без фільтра Пошук

Рис. 2.15. Вміст таблиці КАДРИ

Потрібний запис у таблиці можна знайти за значенням будь-якого її поля або за фрагментом його значення. Розглянемо пошук запису в таблиці на прикладі 2.

Приклад 2

1. У таблиці КАДРИ в режимі таблиці встановимо курсор на поле, за значенням якого потрібно шукати запис (наприклад, на поле Прізвище), і натиснемо кнопку Знайти у розділі Пошук — відкриється вікно Пошук і заміна (рис. 2.16).
2. У вікно Пошук і заміна в поле Знайти введемо потрібне значення, наприклад Сокіл Т. Л., і натиснемо кнопку Знайти далі. Курсор встановиться на записі з прізвищем Сокіл Т. Л.

У полі Зіставити можна вибрати одне зі значень, що зменшує необхідність використання метасимволів:

- будь-яку частину поля (зразок може міститися всередині значення поля);
- усе поле (зразок без метасимволів має збігатися з усім значенням поля);
- початок поля (будуть знайдені тільки ті поля, які починаються зі зразка).

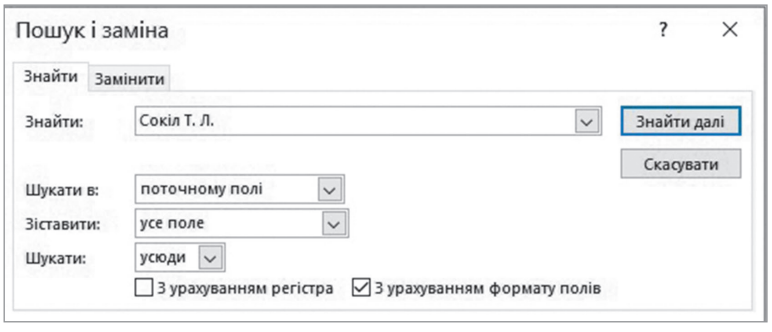


Рис. 2.16. Вікно для пошуку й заміни даних

У процесі введення даних у поле **Знайти** можна використовувати такі метасимволи:

* — довільна кількість будь-яких символів;

? — один довільний символ;

≠ — одна довільна фраза.

Якщо прапорець З урахуванням регістра встановлено, то зразок повинен точно відповідати шуканому фрагменту; якщо знято, то зразок можна вводити великими і малими буквами.

Якщо прапорець З урахуванням формату полів встановлено, то зразок повинен бути таким, як і шуканий фрагмент, наприклад $345; якщо знято, то форматування не враховується, наприклад значення $345 буде знайдено і для зразка 345.

Після завершення пошуку на знайдений запис встановлюється курсор, і поле цього рядка висвітлюється іншим кольором. Оскільки в таблиці може бути кілька значень, що відшукуються, то для продовження пошуку необхідно ще раз натиснути кнопку Знайти далі. Для пошуку першого входження зразка можна використовувати поле Пошук. Пошук завершується після першого визначення рядка.

Знайдене значення можна змінювати, вводити нове. Поле типу Лічильник, заблоковані поля та поля, що обчислюються, змінювати не можна (приклад 3).

Записи з таблиці можна вирізати й копіювати до буфера обміну, за допомогою кнопки Вставити вставляти в іншу таблицю, а також у документи Word і Excel.

Розглянемо алгоритм вилучення запису з таблиці.

1. Виділити потрібний запис і натиснути кнопку Видалити на вкладці Основне. Відкриється меню цієї кнопки (рис. 2.17), у якому слід виконати команду Видалити запис.
2. У вікні, що відкриється (рис. 2.18), необхідно підтвердити або відмінити вилучення. Якщо потрібно вилучити весь запис, його слід виділити й натиснути кнопку Видалити, у меню (див. рис. 2.17) виконати команду Видалити запис.

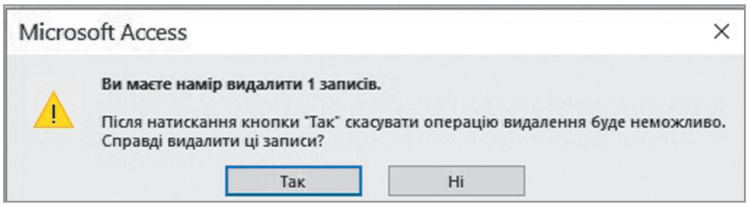


Рис. 2.18. Вікно підтвердження/відміни видалення запису

Слід пам'ятати, що у зв'язаних таблицях зі встановленим прапорцем Забезпечення цілісності даних вилучити запис не завжди вдається.

Приклад 3. Знайдемо в таблиці КАДРИ запис, у полі **Рік народження** якого є значення 1981. Буде виділено запис із прізвищем Рябко Р. П.

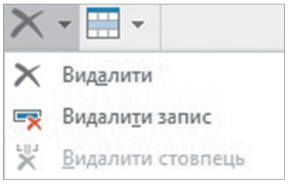


Рис. 2.17. Меню кнопки **Видалити**

Коли потрібно приховати деякі поля, їх слід виділити й у групі **Записи** виконати команду **Додатково** → **Приховати поля**. Щоб відновити приховані поля, слід виконати команду **Додатково** → **Відобразити поля** та у вікні, що відкриється, увімкнути прапорець відповідного поля.

Запитання для перевірки знань

1. Поясніть порядок уведення записів у таблицю.
2. За значенням якого поля впорядковуються записи за замовчуванням?
3. Як можна здійснити навігацію в таблиці?
4. Як здійснюється пошук запису в таблиці?
5. Як вилучити запис із таблиці?
6. Які дії слід виконати для додавання нового запису в таблицю?
7. З якою метою і як приховують поля?

Завдання для самостійного виконання

1. Уведіть у таблиці **КЛАСИ** й **УЧНІ** БД **skola** наведені в них дані.
2. Виконайте навігацію по таблиці **УЧНІ** за допомогою кнопок навігації.
3. Додайте в таблицю **УЧНІ** новий запис, а потім вилучте його. Збережіть таблицю.
4. Знайдіть у таблиці **УЧНІ** запис за прізвищем Пека П. О.
5. Використайте метасимволи для пошуку записів у таблиці **УЧНІ** за значенням поля **Адреса**.
6. Вилучте з таблиці **УЧНІ** будь-який запис, а потім відновіть його. Збережіть таблицю.

2.4. Сортування і фільтрування записів. Операції над таблицями

Сортування записів у таблиці можна реалізувати за значенням будь-якого поля. А що ви розумієте під терміном «фільтрація записів»?

Згадаємо, що за замовчуванням записи таблиці виводяться впорядкованими за значенням первинного ключа. Проте часто виникає необхідність отримати записи, упорядковані за значенням інших полів.

Сортування записів — це впорядкування записів за значеннями одного поля або кількох полів.

Для сортування **за значенням одного поля** треба його виділити й натиснути кнопку **За зростанням (А → Я)** або **За спаданням (Я → А)**. Можна також скористатися контекстним меню поля.

Для **сортування за значеннями кількох полів** необхідно ці поля виділити й скористатися одним із наведених далі способів.

Сортування записів виконується спочатку за значенням лівого виділеного поля. Якщо в ньому є поля, значення яких збігаються, то певні записи впорядковуються за значенням наступного поля. Наприклад, у результаті сортування записів За зростанням за значеннями полів Освіта і Стаж таблиці КАДРИ отримаємо розміщення записів, як наведено на рис. 2.19.

КАДРИ

Справа	Прізвище	Посада	Рік народження	Освіта	Стаж	Оклад	Номер магазину
120	Рябко Р. П.	експерт	1981	вища	8	4200	21
132	Таран В. Д.	диспетчер	1973	вища	15	4000	31
116	Раков Г. П.	аналітик	1965	вища	19	4500	21
111	Семко М. М.	диспетчер	1970	середня	16	4000	21
109	Шрамко Т. Л.	диспетчер	1961	середня	24	4000	6
105	Сокіл Т. Л.	касир	1960	середня	27	3500	6
115	Горошко Ф. Р.	диспетчер	1975	середня спеціальна	17	4000	31
0			0		0	0	0

Рис. 2.19. Таблиця КАДРИ, упорядкована за значенням полів **Освіта** і **Стаж**

Як бачимо з рис. 2.19, основне сортування виконано за зростанням значення поля Освіта. Записи з однаковим значенням упорядковано за зростанням значення поля Стаж (приклад 1).

Приклад 1. Якщо здійснити фільтрацію записів таблиці КАДРИ за значенням **більше або дорівнює 24** в полі **Стаж**, то отримаємо записи з прізвищами Сокіл Т. Л. і Шрамко Т. Л.

Фільтрування записів — це відбір із таблиці записів, які містять задане значення у вибраних полях.

Фільтрування можна виконати за виділенням і формою.

Фільтрування за виділенням — це відбір записів на основі значень поточного поля. Для його реалізації спочатку треба впорядкувати записи за значенням поля, яке використовується у фільтрації; встановити курсор на тому значенні поля, за яким буде виконуватися фільтрування; натиснути на кнопку Виділення в групі Сортування й фільтр та вибрати необхідну умову в меню, що відкриється (приклад 2).

Приклад 2. Відфільтруємо записи таблиці КАДРИ за значенням середня поля Освіта.

1. Упорядкуємо записи таблиці КАДРИ за значенням поля Освіта.
2. Встановимо курсор на назві середня цього поля і натиснемо кнопку Виділення в групі Сортування й фільтр. Відкриється меню з умовами (рис. 2.20).
3. Виберемо першу умову Дорівнює «середня». Відкриється таблиця, вміст якої наведено на рис. 2.21.

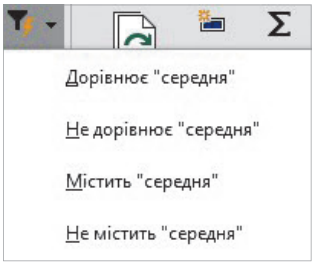


Рис. 2.20. Меню умов для фільтрації записів

У нижній частині таблиці висвітлиться кнопка з написом Відфільтровано (або Не відфільтровано). Це означає, що записи відфільтровано (або не відфільтровано). Натискаючи цю кнопку, можна вмикати й вимикати фільтрування записів. До відфільтрованих записів можна застосовувати ще кілька фільтрів, наприклад за значеннями полів Посада, Оклад та ін.

КАДРИ

Справа	Прізвище	Посада	Рік народження	Освіта	Стаж	Оклад	Номер магазину
109	Шрамко Т. Л.	диспетчер	1961	середня	24	4000	6
111	Семко М. М.	диспетчер	1970	середня	16	4000	21
105	Сокіл Т. Л.	касир	1960	середня	27	3500	6
*			0		0	0	0

Запис: 4 з 4 Відфільтровано Пошук

Рис. 2.21. Таблиця з відфільтрованими записами

Використовуючи фільтрування за формою, можна вводити критерії в поля таблиці умов. Розглянемо алгоритм відкриття таблиці умов на прикладі 3.

Приклад 3

1. У розділі Сортування й фільтр відкриємо меню кнопки Параметри розширеного фільтра й виконаємо команду Розширений фільтр/сортування. Відкриється перелік полів таблиці, а в нижній частині вікна — таблиця конструктора.
2. У таблицю конструктора, наприклад, із відфільтрованої таблиці КАДРИ (див. рис. 2.21) перенесемо ім'я поля, у яке вводитиметься критерій. Для цього достатньо двічі клацнути його в таблиці. Перенесемо, наприклад, ім'я поля Стаж.
3. У таблиці конструктора в запис Критерії цього поля введемо умову, наприклад, >16. Далі відкриємо меню кнопки Параметри розширеного фільтра, у якому виконаємо команду Застосувати фільтр/сортування. Отримаємо записи (рис. 2.22).

КАДРИ КАДРИФільтр1

Справа	Прізвище	Посада	Рік народження	Освіта	Стаж	Оклад	Номер магазину
109	Шрамко Т. Л.	диспетчер	1961	середня	24	4000	6
105	Сокіл Т. Л.	касир	1960	середня	27	3500	6
*			0		0	0	0

Запис: 3 з 3 Відфільтровано Пошук

Рис. 2.22. Відфільтровані записи відібрано за критерієм поля **Стаж >16**

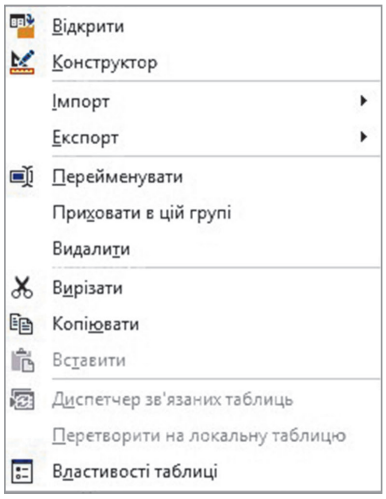


Рис. 2.23. Контекстне меню таблиці, відкрите в області **Усі об'єкти**

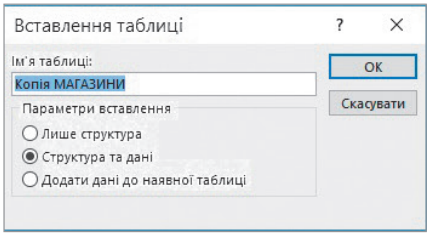


Рис. 2.24. Вікно для копіювання таблиці

У таблиці умов можна задати довільну кількість критеріїв фільтрування, які об'єднуються операторами І/АБО.

Над таблицями в середовищі Access можна виконувати операції перейменування, вилучення, копіювання та ін.

Перейменувати таблицю можна за допомогою її контекстного меню, яке відкривається шляхом клацання імені таблиці в області Усі об'єкти. Вміст меню наведено на рис. 2.23.

Для перейменування таблиці слід виконати такі дії:

Крок 1	Виконати команду **Перейменувати**
Крок 2	В області **Усі об'єкти** в текстове поле цієї таблиці ввести нове ім'я
Крок 3	Натиснути клавішу **Enter**. Слід пам'ятати, що після цього потрібно змінити це ім'я в усіх об'єктах БД (запитах, звітах та ін.)

Для вилучення таблиці можна також скористатися її контекстним меню й виконати в ньому команду Видалити.

Для копіювання таблиці в контекстному меню слід виконати команду Копіювати, ще раз викликати контекстне меню й виконати команду Вставити. Відкриється вікно Вставлення таблиці (рис. 2.24). Далі в поле Ім'я таблиці потрібно ввести нове ім'я та ввімкнути один із розташованих нижче перемикачів.

Якщо таблиця копіюється повністю, то слід увімкнути перемикач Структура та дані й натиснути кнопку ОК.

Запитання для перевірки знань

1. Які існують способи сортування записів?
2. Як упорядкувати записи за значенням одного поля?
3. Що називають фільтрацією записів?
4. Як упорядкувати записи за значеннями двох полів?
5. Які операції виконують над таблицями?
6. Як перейменувати таблицю?
7. Як здійснити фільтрування записів за виділенням?
8. Поясніть сутність фільтрації записів за формою.
9. Як скопіювати таблиці?
10. Як вилучити таблиці?

Завдання для самостійного виконання

1. Упорядкуйте записи таблиці **УЧНІ** БД **skola** за значенням поля **Зріст**.
2. Упорядкуйте записи таблиці **УЧНІ** за значеннями полів **Прізвище** й **Історія**.
3. Виконайте фільтрування записів таблиці **УЧНІ** за одним зі значень поля **Улюблений предмет**.
4. Використайте фільтр **за формою** для фільтрування записів таблиці **УЧНІ** за одним зі значень поля **Дата народження**.
5. Надайте таблиці **КЛАСИ** нове ім'я. Відновіть ім'я **КЛАСИ**.
6. Виконайте копіювання таблиці **УЧНІ** й створіть на її основі нову таблицю з іменем **ПЕРША**.

3. Запити

3.1. Загальні відомості про запити

Пригадайте основне призначення запитів.

Запит — один із основних об'єктів БД Access. Головне його призначення полягає у відборі потрібних даних із таблиць, їх опрацюванні та поданні користувачеві у зручній формі.

Запит застосовується також для змінення даних у БД.

Створений запит можна зберігати з певним іменем і потім неодноразово виконувати. Якщо між першим і другим запусками запиту дані в таблицях змінилися, то в процесі другого його виконання будуть використовуватися оновлені дані.

Запити класифікують за багатьма ознаками. Розподіл запитів за основними ознаками наведено на рис. 3.1.

Запити не містять даних. Під час кожного нового виконання запиту формуються необхідні дані з тих таблиць, на основі яких його створено.

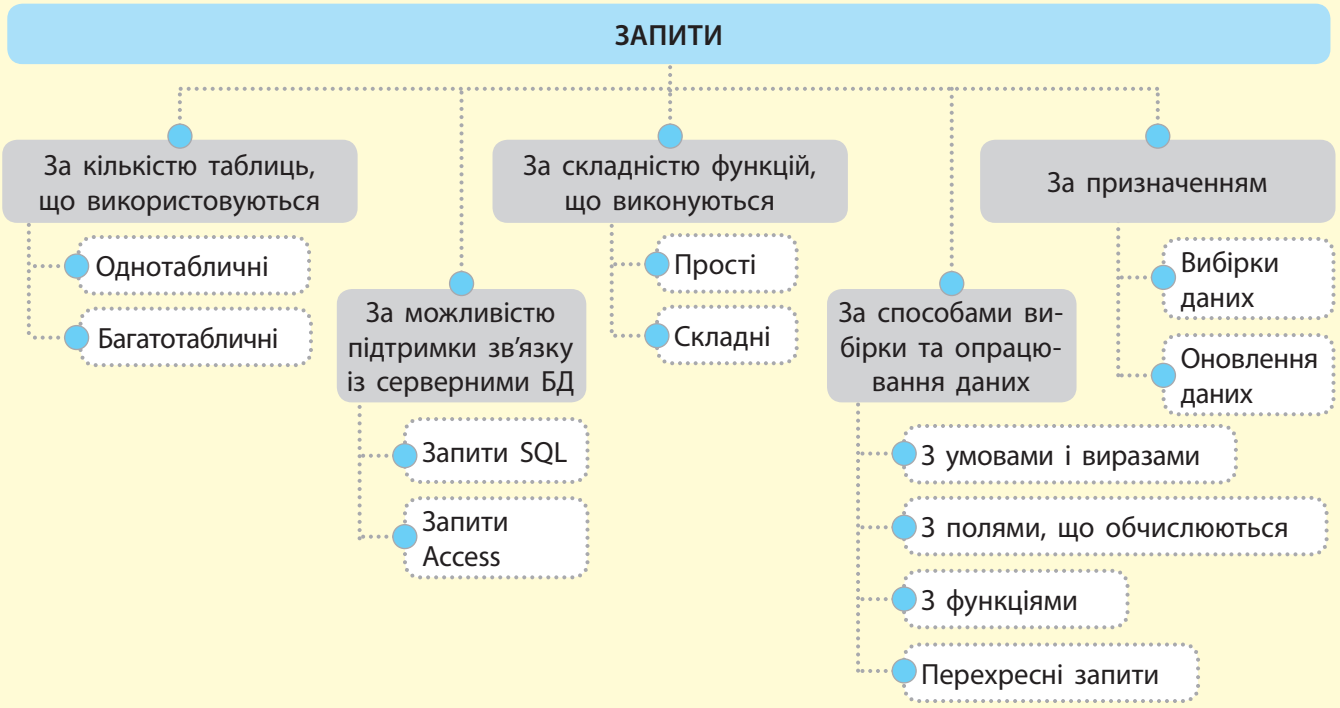


Рис. 3.1. Класифікація запитів

Запити, за допомогою яких вибирають дані з однієї таблиці, називають однотабличними.

Запити, за допомогою яких вибирають дані з кількох таблиць, називають багатотабличними.

Простими називають запити, за якими вибирають дані за критерієм одного поля однієї таблиці (приклад 1).

Складними називають запити, за якими вибирають дані за критеріями кількох полів із кількох таблиць (приклад 2).

Приклад 1. Приклади *простих* запитів: вибрати з таблиці МАГАЗИНИ ті номери магазинів, де кількість працівників більша за 13; вибрати з таблиці КАДРИ прізвища тих працівників, які народилися після 1979 року.

Приклад 2. Складний запит для таблиць МАГАЗИНИ і КАДРИ, за яким формуються дані, наведено в табл. 3.1. Тут із таблиць МАГАЗИНИ і КАДРИ відібрано прізвища осіб, які працюють диспетчерами в магазинах 21 і 31.

Таблиця 3.1. Результат виконання складного запиту

Магазин	Адреса	Прізвище	Посада
21	вул. Паркова, 33	Семко М. М.	диспетчер
31	вул. Печерська, 21	Таран В. Д.	диспетчер
31	вул. Печерська, 21	Горошко Ф. Р.	диспетчер

До однієї БД Access може бути розроблено кілька запитів. Кожен із них можна виконати в будь-який час, і кожен із них виконує чітко визначені функції. Запити можуть виконуватися самостійно, але найчастіше їх використовують як складові форм і звітів.

Як вам відомо, запити не містять даних, лише формують потрібні дані з таблиць. Наприклад, дані, наведені в табл. 3.1, не зберігаються, а формуються під час виконання запиту. Щоб зберегти дані, необхідно створити таблицю та скопіювати до неї ці дані. Описаний тип називають запитом на вибірку.

Компанія Oracle є абсолютним лідером на ринку систем управління БД. Їй належить 45 % ринку.

Запити на вибірку даних — запити, які забезпечують добір потрібних даних із таблиць. Такий тип запитів є одним із найбільш розповсюджених.

Разом із тим у Access використовуються й запити на змінення (оновлення даних) — запити, за допомогою яких здійснюється модифікування структури таблиць і змінення в них даних.

Найчастіше дані за допомогою запитів вибираються на основі критеріїв. Окрім того, система Access має набір убудованих функції, за допомогою яких дані можна вибрати з таблиць, а також опрацювати й узагальнити.

З опрацьованих даних можна створювати нові поля. Такий тип запитів називають запитами з полями, що обчислюються.

У запитах різних типів найчастіше реалізуються такі операції:

- вибір даних із вказаних полів на основі заданих критеріїв;
- упорядкування даних із таблиць за значеннями вказаних полів;
- побудова нової таблиці або діаграми з отриманих даних;
- опрацювання вибраних із таблиць даних за допомогою вбудованих функцій;
- використання отриманих за допомогою запиту даних як джерела для інших запитів;
- додавання даних, отриманих за допомогою запитів, до інших таблиць;
- обмін даними з іншими БД, а також текстовим редактором Word і таблицями Excel.

Вирази в критеріях, що застосовуються в запитах, будуються на основі звичайних арифметичних операцій, операцій порівняння й логічних операцій (**And**, **Or**, **Xor**, **Not**).

У запитах можуть використовуватися специфічні оператори.

- Рядкові оператори:
  Like (які збігаються/відповідність);
  Not Like (які не збігаються/невідповідність);
  об'єднання рядків (&).

Оператори Like і Not Like використовуються для порівняння двох рядкових виразів. При цьому перевіряється, чи збігаються ці вирази, і залежно від результату повертаються значення Так, Ні або Null.

Оператор Like має таку структуру: <ім'я поля> Like <зразок> (приклад 3).

- Оператори списку й діапазону:
  In (входження в список);
  Is (наявність значення);
  Between...And (входження в діапазон).

За допомогою оператора In перевіряється, чи збігається значення поля з одним зі значень списку. Якщо збігається, повертається значення Так, інакше — Ні (приклад 4).

Оператор Is використовується тільки з ключовим словом Null для з'ясування, чи містить об'єкт будь-яке значення. Повертається значення Так, якщо вираз порожній (не містить жодного значення).

Оператор Between...And має таку структуру: <ім'я поля> Between <нижня межа> And <верхня межа>. Повертається значення Так, якщо значення поля знаходяться між значеннями <нижня межа> і <верхня межа> (приклад 5).

Критерії поділяються на *прості* й *складні*.

Прості критерії зазвичай містяться лише в одному полі, складні критерії — у кількох полях. Наприклад, вибрати з таблиць МАГАЗИНИ і КАДРИ прізвища працівників магазинів, які народилися у період з 1961 до 1975 років і працюють диспетчерами або аналітиками.

Далі буде стисло описано методику створення усіх типів запитів (див. рис. 3.1), крім запитів SQL, для розроблення яких потрібно володіти мовою SQL.

Приклад 3. Для таблиці КАДРИ вираз **<Прізвище Like "Раков Г. П.">** повертає значення **Так**, оскільки поле **Прізвище** містить значення **Раков Г. П.**

Приклад 4. Для таблиці КАДРИ оператор **<Прізвище In ('Семко Н. Н.', 'Горошко Ф. Р')>** повертає значення **Так**, тому що в цьому полі є зазначені прізвища.

Приклад 5. Для таблиці КАДРИ за допомогою оператора **[Рік народження] Between 1965 And 1973** повертається значення **Так**, оскільки в цьому полі є зазначені діапазони.

Запитання для перевірки знань

1. Назвіть основне призначення запитів.
2. Які запити називають простими?
3. Опишіть, як класифікують запити за призначенням.
4. Назвіть основні класифікаційні ознаки запитів.
5. Наведіть приклад простого запиту.
6. Які функції виконують запити на вибірку?
7. Назвіть операції, які найчастіше реалізуються в запитах.
8. Які існують оператори списку й діапазону?
9. Наведіть приклад використання операторів **Like** і **Not Like**.

Завдання для самостійного виконання

1. Накресліть класифікацію запитів за призначенням.
2. Наведіть приклад простого запиту для таблиці **КАДРИ**.
3. Наведіть приклад простого запиту для таблиці **МАГАЗИНИ**.
4. Наведіть приклад використання оператора **In** для таблиці **МАГАЗИНИ**.
5. Наведіть приклад складного запиту для таблиць **МАГАЗИНИ** і **КАДРИ**.
6. Накресліть класифікацію запитів за основними ознаками.

3.2. Запити на вибірку даних

Поміркуйте, що необхідно зробити для одночасного отримання потрібних даних із кількох таблиць БД.

Запити на вибірку даних — це запити, які забезпечують вибір необхідних даних із однієї або кількох таблиць.

Розглянемо загальний порядок створення простого запиту на вибірку (запиту для однієї таблиці) та приклад 1.

Крок 1	Відкрити БД, активувати вкладку **Створення** й у розділі **Запити** клацнути кнопку **Макет запиту**, який фактично є конструктором запиту. У результаті відкриються вікно конструктора запиту (вікно **Запит1**) і вікно **Відображення таблиці,** у якому містяться імена всіх таблиць цієї БД. На панелі інструментів вкладки **Конструктор** з'явилася група кнопок **Тип запиту**, у якій виділено кнопку **Вибір**. Це означає, що запит на вибірку створюється за замовчуванням. Якщо створюватимуться інші типи запитів, то потрібно вмикати відповідну кнопку в цій групі.
Крок 2	Вибрати у вікні **Відображення таблиці** необхідну таблицю — відкриється перелік її полів.
Крок 3	Створити запит на основі вмісту таблиці.

Приклад 1. Створити простий запит із іменем Запит_1, за допомогою якого з таблиці КАДРИ виводяться дані про співробітників зі стажем понад 16 років. Результуючий набір записів повинен містити такі поля: Справа, Прізвище, Рік народження, Стаж, Номер магазину.

1. Для створення запиту відкриємо БД abc і виконаємо команду Створення → Макет запиту. Відкриються вікно конструктора запитів і вікно Відображення таблиці, у якому виберемо таблицю КАДРИ. Для цього встановимо курсор на імені цієї таблиці й клацнемо кнопку Додати. Після цього вікно Відображення таблиці можна буде закрити.
2. У записі Поле таблиці конструктора запитів послідовно розмістимо зазначені імена полів (Справа, Прізвище, Рік народження, Стаж, Номер магазину) таблиці КАДРИ (рис. 3.2). Для цього достатньо двічі клацнути кнопкою миші на імені певного поля цієї таблиці.

Запис Сортування використовується для сортування даних у таблиці, яку буде отримано після виконання запиту. Сортувати дані можна за значенням кількох полів. Прапорець, установлений на перетині запису Відображення й певного поля, означає, що це поле буде виведено на екран, інакше воно виводитися не буде. Запис Критерії призначено для запису виразу, на основі якого відбираються записи. Запис Або слугує для визначення додаткової умови відбору записів.

3. У запис Критерії поля Стаж уведемо вираз >16. Збережемо запит, для чого на панелі швидкого доступу натиснемо кнопку Зберегти й у вікні, що відкриється, уведемо ім'я Запит_1 і натиснемо кнопку ОК. У результаті ім'я цього запиту з'явиться в області переходів.
4. Виконаємо запит. Для цього на стрічці в групі Результати натиснемо кнопку Запуск!. Отримаємо результат, як подано на рис. 3.3.

5. Для закриття запиту відкриємо його контекстне меню і виконаємо команду Закрити.

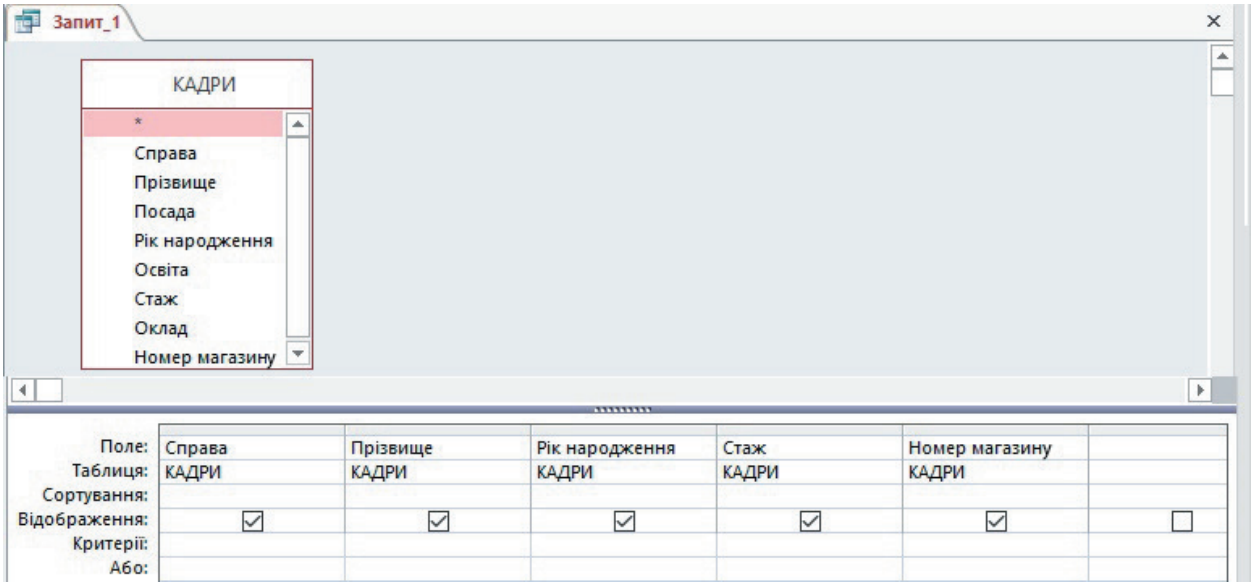


Рис. 3.2. Вікно конструктора запитів

Запит_1

Справа	Прізвище	Рік народження	Стаж	Номер магазину
105	Сокіл Т. Л.	1960	27	6
115	Горошко Ф. Р.	1975	17	31
116	Раков Г. П.	1965	19	21
109	Шрамко Т. Л.	1961	24	6
(0)		0	0	0

Рис. 3.3. Дані про працівників магазинів зі стажем роботи понад 16 років

Створений запит можна перейменовувати й редагувати.

Щоб перейменувати запит, слід відкрити його контекстне меню й виконати команду Перейменувати. Ім'я цього запиту в області переходів буде виділено прямокутником іншого кольору. У поле слід увести нове ім'я й натиснути клавішу Enter.

У процесі редагування запиту можна виконувати такі дії:

- додавати поля в запит із таблиці;
- вилучати поля;
- додавати нові поля;
- змінювати розміри полів;
- змінювати порядок розміщення полів;
- змінювати критерії відбору записів, порядок їх сортування й порядок виведення (невиведення) полів;
- перейменовувати поля запиту;
- вилучати таблиці із запиту (для багатотабличних запитів).

Для виконання цих операцій запит потрібно відкрити в режимі конструктора. Їх виконують так само, як і аналогічні операції в таблицях (приклад 2).

Методика **створення запиту для кількох таблиць** багато в чому схожа на методику створення запиту для однієї таблиці. Лише слід ураховувати, що таблиці обов'язково повинні мати між собою зв'язок.

Приклад 2. Створити Запит_2, за допомогою якого в результуючу таблицю виводяться прізвища диспетчерів із полями Номер магазину і Телефон із таблиці МАГАЗИНИ, а з таблиці КАДРИ — поля Прізвище, Посада й Освіта. Упорядкувати записи за прізвищами працівників в алфавітному порядку.

1. У відкритій БД abc активуємо вкладку Створення й виконаємо команду Макет запиту. Додамо у вікно конструктора запиту обидві таблиці й закриємо вікно Відображення таблиць.
2. У запис Поле таблиці конструктора запиту перенесемо поля Номер магазину й Телефон таблиці МАГАЗИНИ і поля Прізвище, Посада й Освіта таблиці КАДРИ.
3. У запис Критерії поля Посада введемо назву професії диспетчер, а в записі Сортування поля Прізвище встановимо значення За зростанням.
4. Збережемо запит з іменем Запит_2 і виконаємо його. На екрані повинні з'явитися записи, як наведено на рис. 3.4.

Запит_2

Номер магазину	Телефон	Прізвище	Посада	Освіта
31	234-22-98	Горошко Ф. Р.	диспетчер	середня спеціальна
21	234=54-63	Семко М. М.	диспетчер	середня
31	234-22-98	Таран В. Д.	диспетчер	вища
6	234-67-92	Шрамко Т. Л.	диспетчер	середня

Рис. 3.4. Дані про диспетчерів, вибрані з двох таблиць

Запитання для перевірки знань

1. Які запити називають запитами на вибірку даних?
2. Як зберігається запит?
3. Опишіть режими, у яких можна відкрити запит.
4. Для чого призначено запис **Критерії** таблиці конструктора запиту?
5. Як можна змінити порядок розміщення полів у запиті?
6. Поясніть різницю між створенням запиту для однієї таблиці та кількох таблиць.
7. Як створити запит на вибірку даних?
8. Які дії можна виконувати під час редагування запиту?

Завдання для самостійного виконання

1. Створіть **Запит41** на основі таблиці УЧНІ, за допомогою якого вибираються прізвища учнів, улюбленим предметом яких є історія. Записи повинні мати поля **Прізвище**, **Клас**, **Улюблений предмет**.
2. Створіть **Запит42** на основі таблиці УЧНІ, за допомогою якого вибираються прізвища учнів, зріст яких більший від 170.
3. Створіть **Запит43** на основі таблиці УЧНІ, за допомогою якого вибираються прізвища учнів, улюбленими предметами яких є математика та інформатика.
4. Створіть **Запит44** на основі таблиць КЛАСИ і УЧНІ, за допомогою якого вибираються прізвища класних керівників учнів, які з інформатики та історії мають бали, більші від 10.
5. Створіть **Запит45** на основі таблиці УЧНІ, за допомогою якого вибираються прізвища учнів, які народилися у 2002 році.
6. Створіть **Запит46** на основі таблиць КЛАСИ й УЧНІ, за допомогою якого для класів, у яких учнів більше ніж 24, вибираються прізвища учнів, які з історії мають більше від 9 балів.

3.3. Запити з функціями і з полями, що обчислюються

Назвіть переваги використання стандартних функцій у запитах.

Ми вже розглянули запити, за допомогою яких із таблиць можна вибрати необхідні дані за певними критеріями. Отримані дані можна також опрацьовувати (наприклад, обчислити середнє значення поля, знайти серед знайдених запис із мінімальним значенням певного поля тощо). Так, для таблиці КАДРИ можна обчислити середній стаж роботи працівників із вищою освітою.

У системі Access є вбудовані функції, що дають змогу узагальнити дані деяких полів і полегшити опрацювання даних.

Запити, у яких використовуються такі функції, називають по-різному, наприклад підсумковими запитами. Але найчастіше їх називають запитами з функціями.

У системі Access 2.16 існує два способи використання перелічених функцій:

- до запиту, відкритого в режимі таблиці, додається запис підсумків, у якому для кожного поля може використовуватись одна з функцій;
- у режимі конструктора створюється підсумковий запит, у якому обчислюються проміжні підсумки за групами записів.

Розглянемо на прикладі 1 перший спосіб.

Деякі функції системи Access:

- **Sum** (Сума) — обчислює суму значень елементів поля;
- **Avg** (Середнє) — обчислює середнє значення поля;
- **Max/Min** (Максимум/Мінімум) — повертає елемент із максимальним/мінімальним значенням поля;
- **Count** (Кількість) — підраховує кількість записів за значенням поля.

Приклад 1. Створити Запит_4, за допомогою якого з таблиці КАДРИ вибираються записи про співробітників, які народилися після 1961 року і мають стаж понад 15 років. Результуючі записи повинні містити поля: Прізвище, Посада, Рік народження, Стаж і Оклад. Підрахувати кількість результуючих записів за значенням поля Прізвище й обчислити загальну суму окладів цих осіб.

1. Створимо в режимі конструктора звичайний запит на вибірку (рис. 3.5).
2. Збережемо запит з іменем Запит_4 і виконаємо його. Результат виконання запиту наведено на рис. 3.6.
3. На вкладці Основне в групі Записи натиснемо кнопку Підсумки (Σ). Під останнім записом таблиці (рис. 3.6) з'явиться новий запис Підсумок. У цьому записі клацнемо поле Прізвище та в списку, що відкриється, виберемо функцію Кількість.
4. Аналогічно в цьому самому записі поля Оклад виберемо функцію Сума.

   У результаті отримаємо результат, як наведено на рис. 3.7.
5. Для збереження внесених змін ще раз клацнемо кнопку Зберегти.

Поле:	Прізвище	Посада	Рік народження	Стаж	Оклад
Таблиця:	КАДРИ	КАДРИ	КАДРИ	КАДРИ	КАДРИ
Підсумок:	Групування за	Групування за	Групування за	Групування за	Групування за
Сортування:					
Відображення:	☑	☑	☑	☑	☑
Критерії:			>1961	>15	
Або:					

Рис. 3.5. Приклад запиту на вибірку даних у режимі конструктора

Запит_4

Прізвище	Посада	Рік народження	Стаж	Оклад
Семко М. М.	диспетчер	1970	16	4000
Горошко Ф. Р.	диспетчер	1975	17	4000
Раков Г. П.	аналітик	1965	19	4500
*		0	0	0

Рис. 3.6. Результат виконання запиту на вибірку даних

Запит_4

Прізвище	Посада	Рік народження	Стаж	Оклад
Семко М. М.	диспетчер	1970	16	4000
Горошко Ф. Р.	диспетчер	1975	17	4000
Раков Г. П.	аналітик	1965	19	4500
*		0	0	0
	3			12500

Рис. 3.7. Запит у режимі таблиці з підсумковим записом

Розглянемо на прикладі 2 другий спосіб.

Приклад 2

1. Створимо звичайний запит на вибірку в режимі конструктора, наприклад запит, за допомогою якого з таблиці КАДРИ вибираються прізвища працівників з окладом понад 4000 грн і підраховується їх кількість. Результуючий перелік записів має містити поля Справа, Прізвище, Стаж, Оклад.
2. Уведемо в рядок Критерії поля Оклад вираз >4000.
3. На вкладці Конструктор відкриємо меню кнопки Відображення або приховання й виконаємо команду Підсумки (Σ). У конструкторі запиту з'явиться рядок Підсумок, а в кожному полі цього запису буде зазначено Групування за.
4. У записі Підсумок клацнемо те поле, за яким потрібно виконати підрахунок кількості записів (наприклад, поле Справа).

   У списку, що відкриється, виберемо функцію Кількість (рис. 3.8).
5. Збережемо запит та виконаємо.

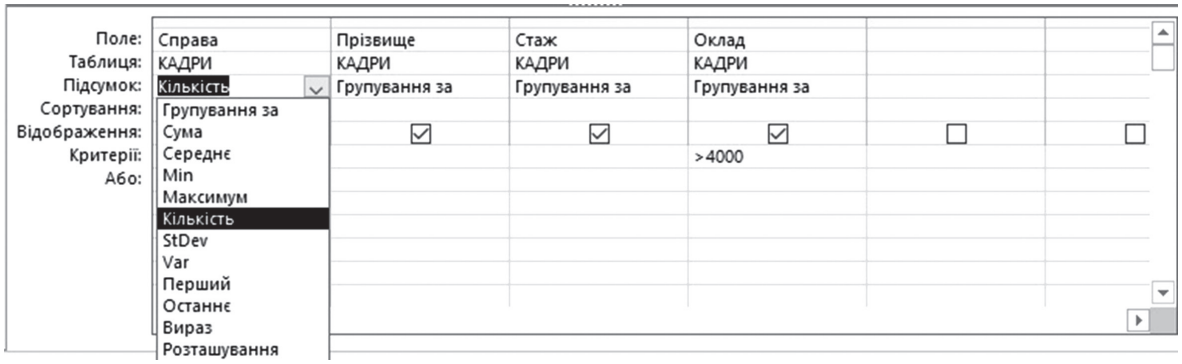


Рис. 3.8. Приклад підсумкового запиту в режимі конструктора

Система Access 2016 дозволяє створювати запити з полями, що обчислюються. Таких полів у запиті може бути кілька.

Запити з полями, що обчислюються, — це запити, які дозволяють виводити в результуючий набір записів не лише поля таблиць, а й нові поля, які створює сам користувач.

У запитах із полями, що обчислюються, містяться дані, отримані під час обчислення даних полів таблиць. Наприклад, на основі даних таблиці КАДРИ в результуючий набір записів можна ввести поле Доплата, у якому обчислюється доплата до окладу залежно від стажу працівника (приклад 3).

На основі таблиці КАДРИ створимо запит з іменем Запит_5, за допомогою якого виводяться всі записи таблиці з полями Прізвище, Стаж, Оклад і Доплата, значення якого обчислюється за наведеною формулою. Порядок створення запиту такого типу несуттєво відрізняється від порядку звичайних запитів (приклад 4).

Приклад 3. Припустимо, що за кожен рік стажу понад 5 років працівники отримують надбавку у розмірі 1% від посадового окладу. Тоді надбавку можна обчислити за формулою:

Доплата = Оклад*(Стаж–5)/100.

Приклад 4

1. Відкриємо БД abc, активуємо вкладку Створення й клацнемо кнопку Макет запиту. Із таблиці КАДРИ перенесемо в конструктор запиту поля Прізвище, Стаж, Оклад, а в наступне поле введемо вираз: Доплата:[Оклад]*([Стаж]–5)/100. Зверніть увагу на те, що імена полів, які входять у вираз, беруться у квадратні дужки.
2. Установимо в записі Сортування поля Прізвище значення За зростанням для того, щоб прізвища виводилися в алфавітному порядку. Створений запит зображено на рис. 3.9.
3. Збережемо запит з іменем Запит_5 (згадаємо, що для цього потрібно натиснути кнопку Зберегти, у вікні, що відкриється, ввести ім'я запиту й клацнути кнопку ОК). У результаті виконання запиту має з'явитися результат, як наведено на рис. 3.10.
4. Закриємо Запит_5.

Поле:	Прізвище	Стаж	Оклад	Доплата: [Оклад]*([Стаж]-5)/100
Таблиця:	КАДРИ	КАДРИ	КАДРИ	
Сортування:	За зростанням			
Відображення:	☑	☑	☑	☑
Критерії:				
Або:				

Рис. 3.9. Запит із полем, що обчислюється

Запит_5

Прізвище	Стаж	Оклад	Доплата
Горошко Ф. Р.	17	4000	480
Раков Г. П.	19	4500	630
Рябко Р. П.	8	4200	126
Семко М. М.	16	4000	440
Сокіл Т. Л.	27	3500	770
Таран В. Д.	15	4000	400
Шрамко Т. Л.	24	4000	760
*		0	0

Рис. 3.10. Результат виконання запиту з полем, що обчислюється

Запитання для перевірки знань

1. Назвіть основні вбудовані функції для опрацювання даних у запиті.
2. Які запити називають запитами з полями, що обчислюються?
3. Яке призначення має функція **Кількість**?
4. Назвіть способи використання вбудованих функцій у запитах.
5. Опишіть порядок створення запиту з функціями.
6. Як додати до запиту підсумковий запис у режимі таблиці?
7. Поясніть порядок створення підсумкового запиту.
8. Поясніть порядок створення запиту з полями, що обчислюються.

Завдання для самостійного виконання

1. Створіть **Запит51** на основі таблиці **УЧНІ**, за допомогою якого підраховується кількість учнів, улюбленими предметами яких є географія та фізика. Записи повинні містити поля **Прізвище**, **Клас**, **Улюблений предмет**.
2. Створіть **Запит52** на основі таблиці **УЧНІ**, за допомогою якого обчислюється середній бал успішності учнів 11 класів окремо з інформатики й окремо з історії.
3. Створіть **Запит53** на основі таблиці **УЧНІ**, за допомогою якого обчислюється кількість учнів, які мешкають на вул. Лугова.
4. Створіть **Запит54** на основі таблиць **КЛАСИ** й **УЧНІ**, за допомогою якого обчислюється спільний середній бал успішності з інформатики та історії учнів класу, яким керує Дерев'янко Н. С.
5. Створіть **Запит55** на основі таблиці **УЧНІ**, за допомогою якого обчислюється середній зріст учнів кожного класу.
6. Створіть **Запит56** на основі таблиць **КЛАСИ** й **УЧНІ**, за допомогою якого для учнів 10 класу, улюбленим предметом яких є інформатика, обчислюється різниця середнього бала успішності з інформатики та історії.

3.4. Запити з параметрами. Перехресні запити

Ви вже знайомі з деякими типами запитів. Які ще функції доцільно використовувати для виконання запитів?

Ми вже розглядали запити з постійними критеріями, тобто запити, під час повторного виконання яких критерій відбору записів не змінювався. На практиці ж часто виникає потреба у зміненні цих критеріїв. Наприклад, під час першого виконання запиту слід вибрати із таблиці КАДРИ прізвища диспетчерів зі стажем роботи понад 10 років, а під час другого — прізвища диспетчерів зі стажем роботи понад 15 років. Такі дії можна виконувати за допомогою запитів із параметрами.

Запит із параметрами — це запит, у процесі виконання якого пропонується ввести деякі дані, наприклад умову, яку потрібно вставити в поле. Його ще називають *запитом зі змінними критеріями*.

За запитами з параметрами на початку їх виконання на екран виводяться повідомлення про необхідність уведення нового критерію (виразу). Методика створення запиту такого типу несуттєво відрізняється від методики створення звичайного запиту.

Розглянемо порядок створення такого запиту на прикладі 1.

Пошукова система Google була створена у 1998 році. За перші два роки її існування у БД цієї пошукової системи було зібрано дані про мільярд веб-сторінок, а у 2008 році — близько трильйона.

Приклад 1

Розробити запит з іменем Запит_6, за допомогою якого з БД abc вибиратимуться прізвища працівників за посадами диспетчер і експерт із магазинів, номери яких уводяться при виконанні запиту. Результуючі записи мають містити поля Номер магазину й Адреса таблиці МАГАЗИНИ і поля Прізвище й Посада таблиці КАДРИ.

Порядок створення запиту може бути таким.

1. Відкриємо БД abc і виконаємо команду Створення → Макет запиту. Виділимо обидві таблиці, клацнемо кнопку Додати й закриємо вікно Відображення таблиці.
2. Із таблиці МАГАЗИНИ перенесемо в таблицю конструктора поля Номер магазину й Адреса, а з таблиці КАДРИ — поля Прізвище й Посада.
3. В умові завдання визначено, що за допомогою запиту мають відбиратися записи тільки за посадами диспетчер і експерт, тобто ця умова є незмінною. Тому в клітинку на перетині запису Критерії і поля Посада уводимо вираз "диспетчер" Or "експерт".

Щоразу після запуску на виконання запиту користувач може вводити будь-який номер магазину. Тому на перетині запису Критерії та поля Номер магазину можна увести, наприклад, текст [У якому магазині?]. Головне, щоб текст містився у квадратних дужках.

У результаті отримаємо запит у режимі конструктора, як наведено на рис. 3.11.

Поле:	Номер магазину	Адреса	Прізвище	Посада
Таблиця:	МАГАЗИНИ	МАГАЗИНИ	КАДРИ	КАДРИ
Сортування:				
Відображення:	☑	☑	☑	☑
Критерії:	[У якому магазині?]			"диспетчер" Or "експерт"
Або:				

Рис. 3.11. Запит із параметром

4. Збережемо й виконаємо Запит_6. На екрані висвітлиться вікно Введення значення параметра із запитанням У якому магазині? (рис. 3.12).

Уведемо, наприклад, номер магазину 21, клацнемо кнопку ОК. Отримаємо результат, як наведено на рис. 3.13.

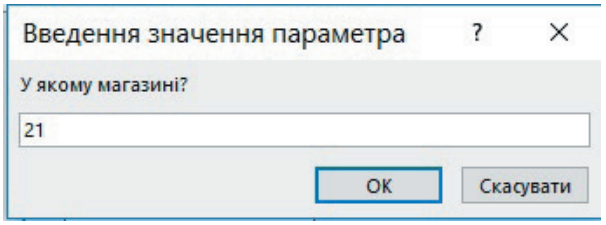


Рис. 3.12. Вікно введення значення параметра

Запит_6

Номер магазину	Адреса	Прізвище	Посада
21	вул. Паркова, 33	Рябко Р. П.	експерт
21	вул. Паркова, 33	Семко М. М.	диспетчер
*			

Рис. 3.13. Результат виконання запиту з параметром

Проаналізуємо вміст таблиці КАДРИ й переконаємося, що дійсно у магазині з номером 21 на посадах диспетчера й експерта працюють відповідно Семко М. М. і Рябко Р. П.

Якщо потрібно здійснити пошук прізвищ диспетчерів і експертів у кількох магазинах, то в запис Критерії поля Номер магазину необхідно ввести іншу умову, а саме — діапазон номерів магазинів, наприклад:

>[Більше якого?] And <[Менше якого?].

У процесі виконання запиту спочатку з'явиться повідомлення Більше якого?, а потім — Менше якого?, на які потрібно дати певну відповідь.

Один запит може містити декілька параметрів у різних полях.

Перехресний запит — це запит на вибірку даних із можливостями групування записів.

Групування можна виконувати як за значеннями полів, так і за значеннями записів. Наприклад, із таблиці КАДРИ можна отримати кількість працівників на всіх посадах у кожному магазині. Як заголовки полів можуть бути використані й деякі вирази.

У режимі конструктора перехресний запит спочатку створюється як звичайний запит на вибірку даних, а потім установлюється режим перехресного запиту.

Розглянемо порядок створення перехресного запиту на прикладі 2.

Для створення перехресного запиту потрібно використати щонайменше три поля:
- поле для визначення заголовка записів;
- поле для визначення заголовка полів;
- поле для вибору значень, над якими будуть виконуватися обчислення.

Приклад 2. Створити перехресний запит, за допомогою якого підраховуються працівники на кожній посаді в кожному магазині.

1. Створимо звичайний запит на вибірку даних у режимі конструктора. Для цього виконаємо команди Створити → Макет запиту → Додати → Закрити. Перенесемо в таблицю конструктора запиту поля Посада й Номер магазину. Виконаємо запит і переконаємося, що він функціонує правильно.
2. Перемкнемо запит у режим конструктора і перетворимо в тип перехресний. У групі Тип запиту клацнемо кнопку Перехресний. У таблиці конструктора запиту з'являться записи Підсумок і Перехресний.

У записі Підсумок обох полів не змінюємо значення Групування за. Клацнемо клітинку на перетині поля Посада й запису Перехресний і виберемо зі списку, що розкриється, Заголовок рядка, а в полі Номер магазину цього самого запису — Заголовок стовпця.

3. У третє поле таблиці конструктора запиту перенесемо поле Номер магазину, у записі Підсумок якого встановимо функцію Кількість, а в записі Перехресний — Значення.

У записі Сортування першого поля можна встановити потрібне сортування записів. Таблиця конструктора запиту матиме вміст, як наведено на рис. 3.14.

Поле:	Посада	Номер магазину	Номер магазину
Таблиця:	КАДРИ	КАДРИ	КАДРИ
Підсумок:	Групування за	Групування за	Кількість
Перехресний:	Заголовок рядка	Заголовок стовпця	Значення
Сортування:			
Критерії:			
Або:			

Заголовок рядка
Заголовок стовпця
Значення
(не відображається)

Рис. 3.14. Таблиця конструктора перехресного запиту

Запитання для перевірки знань

1. Які запити називають запитами з параметрами?
2. Які поля обов'язково використовують у перехресних запитах?
3. Наведіть приклад запиту з параметрами.
4. Поясніть сутність перехресного запиту.
5. Поясніть порядок створення запиту з параметрами.
6. Опишіть порядок створення перехресного запиту.

Завдання для самостійного виконання

1. Створіть **Запит61** із параметрами на основі таблиці **УЧНІ**, за допомогою якого виводяться прізвища учнів із різними улюбленими шкільними предметами, назви яких уводяться в процесі виконання запиту.
2. Створіть **Запит62** із параметрами на основі таблиці **УЧНІ**, за допомогою якого виводяться прізвища учнів різного зросту, значення яких уводяться в процесі виконання запиту.
3. Створіть перехресний **Запит63**, за допомогою якого підраховується кількість учнів, які мають певний улюблений предмет, у всіх класах.
4. Створіть перехресний **Запит64**, за допомогою якого обчислюється кількість учнів окремо в кожному класі, зріст яких більше за 163 см.
5. Створіть **Запит65** із параметрами на основі таблиці **УЧНІ**, за допомогою якого виводяться прізвища учнів, їхній улюблений предмет і номер класу, що вводяться під час виконання запиту.
6. Створіть перехресний **Запит66**, за допомогою якого обчислюється кількість учнів, які мають певний улюблений предмет, у кожному класі.

3.5. Запити на змінення даних

Дані можна змінювати безпосередньо в таблицях. Чи доцільно, на вашу думку, застосовувати запити для виконання аналогічних операцій? Чому?

Рис. 1. Типи запитів даних

Запит на змінення даних — це запит, за допомогою якого в таблицю вносяться зміни. Можна не лише вибирати необхідні дані з таблиць, а й створювати з вибраних даних нову таблицю, змінювати дані в уже створених таблицях, додавати нові записи в створені таблиці, вилучати з таблиць записи.

У Access існують різні типи запитів на змінення (рис. 3.15).

Порядок створення запитів на змінення в режимі конструктора такий самий, як і порядок створення звичайних запитів на вибірку даних. Потім створений запит перетворюється на запит одного з перелічених типів.

Далі розглянемо особливості розроблення запитів для створення нової таблиці та додавання записів у таблицю.

За допомогою запитів для створення нової таблиці вибираються дані з однієї або кількох таблиць і з них формується нова таблиця. Вона може бути розміщена як у поточній БД, так і в іншій, ім'я якої вказується під час створення запиту цього типу. Нова таблиця не має зв'язку з тими таблицями, з яких вона створена. Отже, якщо в таблицях-джерелах відбулися зміни, то дані в ній автоматично не оновлюються.

Розглянемо створення запиту на прикладі 1.

Приклад 1. Розробити запит, за допомогою якого на основі даних таблиць МАГАЗИНИ й КАДРИ створюється нова таблиця з іменем ДОДАТКОВА, у якій містяться поля Справа, Прізвище, Рік народження, Працівників тих магазинів, де кількість працюючих перевищує 14.

1. У відкритій БД abc виконаємо команду Створити → Макет запиту, виділимо обидві таблиці й перенесемо з таблиці КАДРИ поля Справа, Прізвище й Рік народження, а з таблиці МАГАЗИНИ — поле Працівників. Закриємо вікно Відображення таблиці. У запис Критерії поля Працівників уведемо вираз >14. Виконаємо запит і переконаємося, що отримано правильний результат.
2. Перемкнемо запит у режим конструктора й перетворимо запит на вибірку в запит на створення таблиці. Для цього на вкладці Конструктор у групі Тип запиту клацнемо кнопку Створення таблиці. Відкриється вікно Створити таблицю (рис. 3.16).

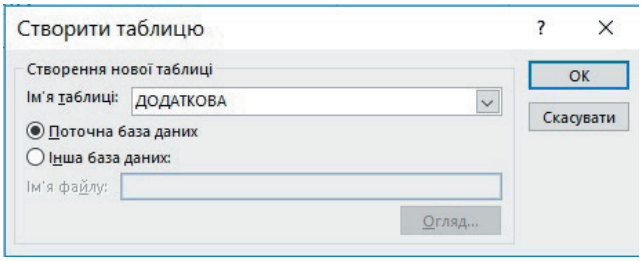


Рис. 3.16. Вікно для створення таблиці

3. У поле Ім'я таблиці введемо ім'я нової таблиці, наприклад ДОДАТКОВА, і ввімкнемо перемикач Поточна база даних, оскільки цю таблицю зберігатимемо у відкритій (поточній) БД abc. Після цього клацнемо кнопку OK.
4. На панелі швидкого доступу клацнемо кнопку Зберегти й збережемо запит із іменем Запит_7.
5. Виконаємо Запит_7 — відкриється вікно (рис. 3.17).

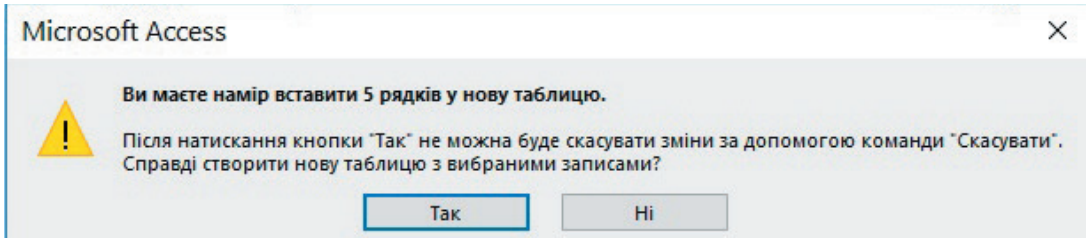


Рис. 3.17. Вікно для підтвердження збереження таблиці

6. Для збереження таблиці в поточній БД клацнемо кнопку Так. В області переходів з'явиться ім'я таблиці ДОДАТКОВА. Після цього закриємо запит і відкриємо створену таблицю, вміст якої наведено на рис. 3.18.
7. Проаналізуємо вміст таблиці й переконаємося, що вона сформована правильно.

ДОДАТКОВА

Справа	Прізвище	Рік народження	Працівників
120	Рябко Р. П.	1981	20
111	Семко М. М.	1970	20
116	Раков Г. П.	1965	20
105	Сокіл Т. Л.	1960	15
109	Шрамко Т. Л.	1961	15

Рис. 3.18. Вміст таблиці ДОДАТКОВА

Запити на додавання даних призначено для додавання нових записів у таблицю на основі опрацювання за певними критеріями даних, які вже є в раніше створених таблицях.

Інколи для додавання всіх записів усіх полів із наявної таблиці в нову доцільніше скористатися командами Копіювати і Вставити. Записи можна додавати як у відкриту, так і в закриту таблицю. Якщо записи додаються в таблицю іншої БД, то слід вказати ім'я та маршрут файла цієї БД.

Загальний порядок розроблення запитів такий:

1) створюється запит на вибірку, за допомогою якого формуються необхідні записи для додавання;
2) перетворюється запит на вибірку в запит на додавання;
3) вибирається таблиця, у яку додаватимуться записи;
4) зберігається й виконується запит.

Розглянемо порядок дій на прикладі 2.

Якщо у записі **Поле** таблиці конструктора запиту є символ «зірочка» (*), то це означає, що використовувати в запиті окремі поля цієї самої таблиці неможливо.

Приклад 2. Створити Запит_8, за допомогою якого до таблиці ДОДАТКОВА додаються з таблиць МАГАЗИНИ й КАДРИ прізвища працівників магазинів, у яких працює 13 робітників, народжених у 1975 році.

1. Створимо запит на вибірку. Із таблиці МАГАЗИНИ перенесемо в таблицю конструктора запитів поля Справа, Прізвище й Рік народження, а з таблиці КАДРИ — поле Працівників. Після виконання цього запиту має з'явитися результат, як наведено на рис. 3.19.

Запит1

Справа	Прізвище	Рік народження	Працівникі
115	Горошко Ф. Р.	1975	13

Рис. 3.19. Запит на вибірку даних із таблиць МАГАЗИНИ й КАДРИ

2. Перетворимо створений запит на вибірку в запит на додавання. Для цього перейдемо в режим конструктора і в групі Тип запиту виконаємо команду Додавання.

3. У вікні Додавання, що відкрилося, введемо ім'я таблиці — ДОДАТКОВА, виберемо варіант Поточна база даних і клацнемо кнопку ОК.

4. Збережемо запит з ім'ям Запит_8 і виконаємо його.
У результаті відкриється вікно, як наведено на рис. 3.20.

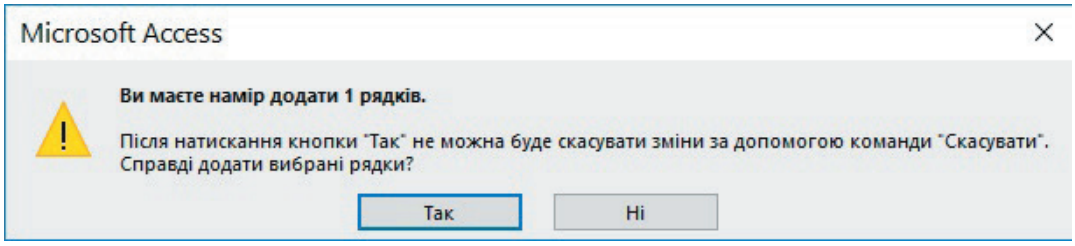


Рис. 3.20. Вікно для підтвердження додавання запису

ДОДАТКОВА

Справа	Прізвище	Рік народж	Працівникі
120	Рябко Р. П.	1981	20
111	Семко М. М.	1970	20
116	Раков Г. П.	1965	20
105	Сокіл Т. Л.	1960	15
109	Шрамко Т. Л.	1961	15
115	Горошко Ф. Р.	1975	13
115	Горошко Ф. Р.	1975	13

Рис. 3.21. Таблиця після додавання запису

5. Підтвердимо додавання запису, для чого клацнемо кнопку Так. Закриємо Запит_8 і відкриємо таблицю ДОДАТКОВА.
У результаті повинна з'явитися таблиця, як наведено на рис. 3.22.

Запитання для перевірки знань

1. Назвіть типи запитів на змінення.
2. Як перетворити запит на вибірку в запит на створення таблиці?
3. Як перетворити запит на вибірку в запит на додавання?
4. Як створити запит для створення таблиці?
5. Яка сутність запитів на додавання записів?
6. Опишіть загальний порядок створення запиту на додавання.
7. Наведіть приклад запиту для створення нової таблиці.
8. Наведіть приклад запиту на додавання.

Завдання для самостійного виконання

1. Розробіть **Запит71**, за допомогою якого на основі таблиці **УЧНІ** створюється таблиця **ПЕРША1**. Таблиця повинна містити дані про учнів 10 класу з полями **Прізвище**, **Дата народження**, **Зріст**, **Улюблений предмет**, **Клас**.
2. Створіть **Запит72**, за допомогою якого до таблиці **ПЕРША1** додаються записи про учнів 9 класу з тими самими полями, що й у завданні 1. Збережіть таблицю з іменем **ПЕРША2**.
3. Створіть **Запит73**, за допомогою якого на основі даних таблиць **КЛАСИ** й **УЧНІ** створюється таблиця з іменем **ПЕРША3** з полями **Прізвище**, **Дата народження**, **Улюблений предмет**, **Клас** для класів, у яких кількість учнів менша за 27.
4. Розробіть **Запит74**, за допомогою якого до таблиці **ПЕРША3** додається поле **Інформатика**. Дані збережіть у таблиці **ПЕРША4**.
5. Розробіть **Запит75**, за допомогою якого до таблиці **ПЕРША4** додаються записи про класи, у яких кількість учнів дорівнює 27 і які мають з історії успішність 9 балів.

4. Інтерфейс користувача. Основи мови SQL. Імпорт та експорт даних

4.1. Створення інтерфейсу користувача для введення даних у базу даних

Спробуйте самостійно визначити основні недоліки таблиць і запитів для роботи з даними з точки зору кінцевого користувача.

Як ви знаєте, дані БД зберігаються у таблицях, і в них можна безпосередньо вводити дані, редагувати їх і вилучати. Не кожен користувач має право це робити. Задля забезпечення цілісності даних і безпеки БД для більшості користувачів доступ до структури і власне таблиць БД має бути обмежений. Наприклад, продавець супермаркету має право внести дані про реалізацію конкретного товару, але не може змінити ціну товару, це право належить адміністрації.

У середовищі Access для введення даних використовується форма.

З одного боку, форми забезпечують безпеку даних від несанкціонованого доступу, а з іншого — спрощують сам процес уведення даних, — користувачу не потрібно переглядати таблицю з десятками полів і відшукувати потрібні дані. Користувачу пропонується бланк, наприклад, із двома-трьома полями, у які він має право внести необхідні дані. Важливо й те, що за допомогою форми дані можуть оновлюватися одночасно у кількох пов'язаних таблицях.

Система Access 2016 має різні засоби створення форм, які містяться на вкладці Створення у групі Форми (рис. 4.1). Найпотужнішим інструментарієм створення форм є Конструктор форм.

Із таблицею як об'єктом працює в основному адміністратор БД. Для роботи з даними користувачам потрібен більш простий і зручний інтерфейс. Для кожної категорії працівників бажано розробити свої правила й засоби роботи з БД.

Форма — це фактично електронний бланк, який для кожної групи користувачів має свою структуру. У цій структурі містяться імена полів, доступні саме для цієї групи. Після заповнення полів дані потрапляють у відповідні поля таблиць для подальшого збереження.

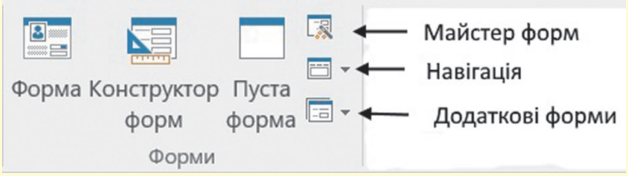


Рис. 4.1. Засоби створення форм

Розглянемо порядок використання Конструктора форм для введення даних у БД на прикладі 1.

Приклад 1. Створити для БД abc на основі таблиці МАГАЗИНИ форму з іменем Форма_6 і з полями Номер магазину, Адреса, Директор, Телефон. Перед першими двома полями ввести текст Координати, а ще перед двома — Контакти.

1. Виділимо в області переходів таблицю МАГАЗИНИ, активуємо вкладку Створити й у групі Форми виконаємо команду Конструктор форм. На екрані з'явиться порожня форма з розділом Подробиці й буде активовано вкладку Конструктор.

2. У групі Колонтитули клацнемо кнопку Назва. У результаті відкриється вікно (рис. 4.2), у якому містяться підрозділи Верхній колонтитул форми, Нижній колонтитул форми і текстове поле з іменем Форма1.
3. Установимо курсор у поле Форма1, введемо текст заголовка, наприклад Магазини нашого району, і клацнемо кнопку Enter.
4. У верхню частину області Подробиці уведемо текст Координати, який відповідно до умови має міститися перед першими двома полями. Для цього клацнемо кнопку Напис, установимо курсор у потрібну позицію області Подробиці, уведемо текст Координати й клацнемо кнопку Enter.
5. У правій частині вікна має розташовуватися поле Список полів (якщо воно відсутнє, клацнемо кнопку Додавання наявних полів, і воно з'явиться). У цьому полі клацнемо кнопку Відображати всі таблиці. Ліворуч від назви таблиці МАГАЗИНИ встановимо перемикач у положення (–) — під назвою таблиці відкриється список її полів. Перемістимо поле Номер магазину в те місце розділу Подробиці, куди слід помістити це поле. У результаті на формі буде розташовано два пов'язані елементи: власне поле введення і підпис для нього (підпис співпадає з іменем цього поля).
6. Аналогічно перемістимо до бланку форми поле Адреса.
7. Після двох перших полів розмістимо назву ще двох полів. Для цього за аналогією у групі Елементи керування клацнемо кнопку Напис, установимо курсор у потрібну позицію області Подробиці, уведемо назву Контакти й натиснемо клавішу Enter. Перемістимо поля Директор і Телефон у розділ Подробиці. Розмістимо елементи форми більш зручно (рис. 4.3).

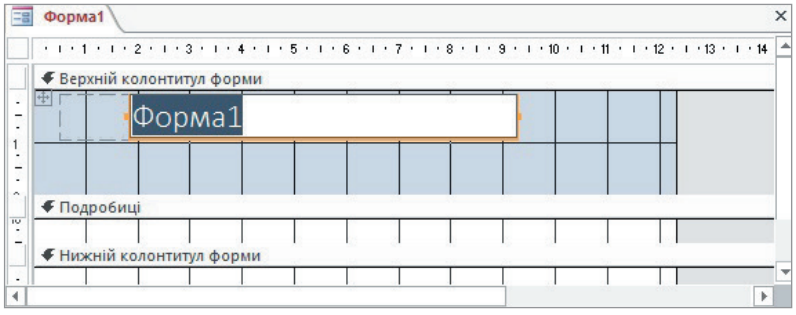


Рис. 4.2. Порожня форма в режимі конструктора

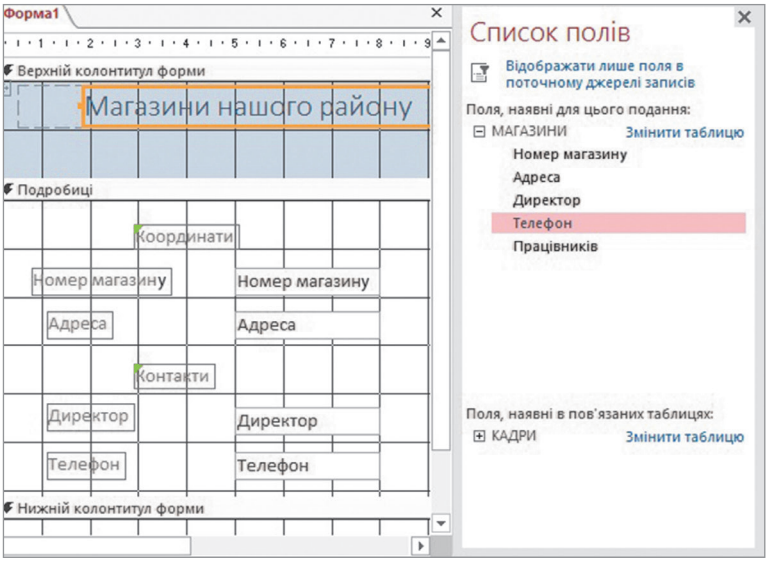


Рис. 4.3. Форма, створена в режимі конструктора

Після цього можна закрити вікно Список полів. Якщо у форму необхідно додати поля з кількох таблиць, то їх потрібно розкрити так само, як і поля таблиці МАГАЗИНИ.

8. Збережемо форму з іменем Форма_6. Тепер форму можна переглянути в таких режимах: Режим форм, Подання таблиці, Режим розмітки.

   На рис. 4.4 створену форму відкрито в режимі форми.

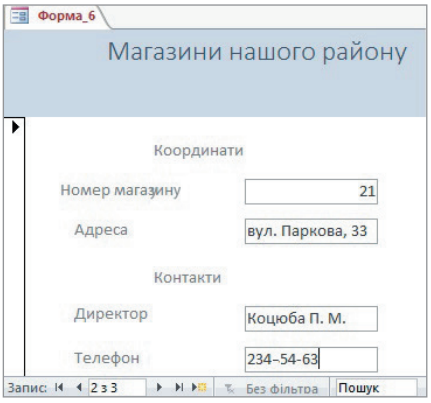


Рис. 4.4. Форму відкрито в режимі форми

Зовнішній вигляд форми можна налаштувати з урахуванням потреб користувача. Для цього слід відкрити контекстне меню форми: установити курсор на вільному місці форми, відкритої у режимі конструктора, і клацнути праву кнопку миші.

Вміст контекстного меню форми зображено на рис. 4.5.

Щоб змінити колір фону форми, вказівник миші слід установити на команді Колір заливки/фону та в наборі кольорів, що відкриється, вибрати потрібний колір.

Або вибрати інший спосіб: відкрити контекстне меню заголовка форми чи інших її компонентів і встановити потрібні кольори. На формі можна також установити або зняти лінійку, сітку чи виконати інші налаштування.

Один із варіантів налаштування зовнішнього вигляду форми наведено на рис. 4.6.

За допомогою кнопок навігації можна перейти до будь-якого запису.

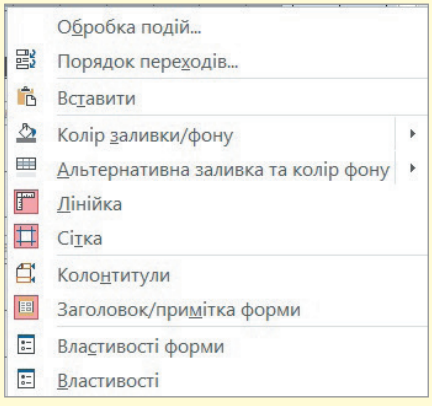


Рис. 4.5. Контекстне меню форми

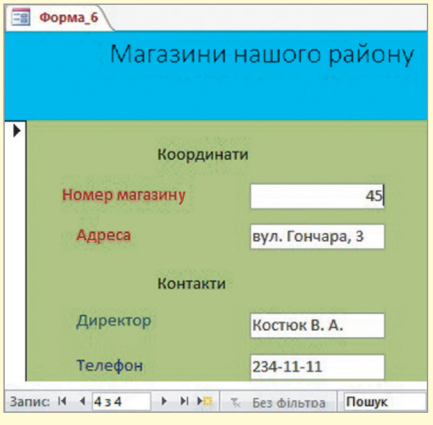


Рис. 4.6. Варіант оформлення зовнішнього вигляду форми

Порядок додавання записів на основі створеної форми розглянемо на прикладі 2.

Приклад 2

1. Відкриємо Форму_6 у режимі форми. Перейдемо на порожній запис, для чого клацнемо, наприклад, кнопку Створити запис, розташовану на панелі навігації.
2. Уведемо в порожній рядок форми дані, наприклад такі: Номер магазину — 6; Адреса — вул. Річкова, 24; Директор — Середа К. М.; Телефон — 234-67-92 (рис. 4.7).
3. Збережемо й закриємо форму. Таблиця МАГАЗИНИ, відкрита в режимі подання таблиці, набуде вмісту, як наведено на рис. 4.8.

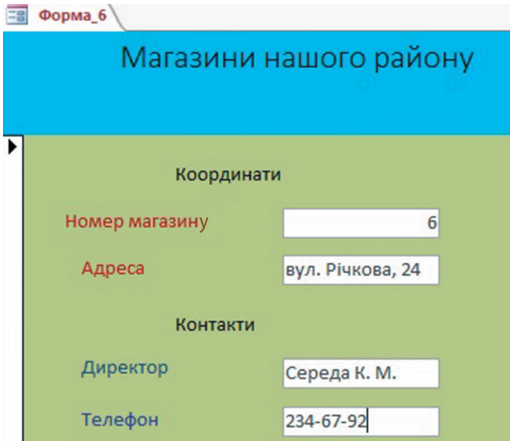


Рис. 4.7. Запис, доданий у таблицю

МАГАЗИНИ

Номер магазину	Адреса	Директор	Телефон	Працівникі	Клацніть, щоб додати
6	вул. Річкова, 24	Середа К. М.	234-67-92	15	
21	вул. Паркова, 33	Коцюба П. М.	234=54-63	20	
31	вул. Печерська, 21	Борзов А. С.	234-50-50	13	
45	вул. Гончара, 3	Костюк В. А.	234-11-11	0	
0				0	

Рис. 4.8. Вміст таблиці МАГАЗИНИ після додавання запису

Зверніть увагу на те, що поле Працівників для введеного запису містить нуль, оскільки у Формі_6 такого поля створено не було і дані в нього не вводилися.

Запитання для перевірки знань

1. Яке призначення **Конструктора форм**?
2. Поясніть порядок створення форм за допомогою **Конструктора форм**.
3. Як перейти до будь-якого запису?
4. Поясніть порядок оформлення загального вигляду форми.

Завдання для самостійного виконання

1. Створіть за допомогою **Конструктора форм** для таблиці **КЛАСИ** форму з полями **Клас**, **Учнів**, **Класний керівник**. Здійсніть навігацію по записах форми.
2. Створіть за допомогою **Конструктора форм** для таблиці **УЧНІ** форму з полям **Прізвище**, **Клас**, **Інформатика**.
3. Створіть за допомогою **Конструктора форм** для таблиці **КЛАСИ** та таблиці **УЧНІ** форму з полями **Клас**, **Класний керівник**, **Прізвище**, **Адреса**, **Дата народження**. Перегляньте форму в різних режимах. Самостійно виберіть колір кожного компонента форми.

4.2. Основи мови запитів SQL

Поміркуйте, як мова запитів SQL розширює функціональні можливості СУБД для роботи з даними.

Сучасні системи управління реляційними БД, у тому числі Access 2016, містять потужні візуальні засоби, зрозумілий і зручний графічний інтерфейс, що забезпечує ефективну роботу з БД. Водночас існують спеціальні мови для створення й супроводу БД. Однією з таких мов є SQL (Structured Query Language — структурована мова запитів).

У системі Access 2016 реалізовано частину SQL, що забезпечує формування запитів і роботу з ними. Далі стисло розглянемо основні відомості саме про мову запитів SQL.

SQL фактично стала стандартом мови реляційних БД. Вона постійно розвивається й удосконалюється, змінюються її міжнародні стандарти.

Як і інші мови програмування, SQL має власний синтаксис. У синтаксичних конструкціях використовуються такі поняття, як ключові слова, оператори, інструкції, речення та ін.

У мові запитів SQL найчастіше використовуються такі оператори:

- SELECT — визначає поля, із яких необхідно вибрати дані;
- FROM — визначає таблицю, поля якої вказано в реченні SELECT; ключові слова SELECT і FROM завжди використовуються разом;
- WHERE — визначає умову відбору полів, за якою вибираються дані;
- ORDER BY — визначає порядок сортування отриманих результатів;
- GROUP BY — визначає порядок групування записів.

Інструкція — це логічно завершена конструкція, яка може інтерпретуватися самостійно. Вона складається із речень і закінчується крапкою з комою.

Речення — це частина інструкції, що обов'язково містить ключове слово, яке визначає його назву.

Наприклад, SELECT Прізвище називають реченням SELECT, а WHERE Посада = 'вчитель' — реченням WHERE.

У мові SQL структура найуживаніших речень така:

```
SELECT <список полів>              -- речення SELECT
FROM <ім'я таблиці>                -- речення FROM
WHERE <ім'я поля> = <умова>;       -- речення WHERE
```

Інструкції можуть містити коментарі, які не впливають на їх виконання. Найчастіше застосовуються однорядкові коментарі, які починаються двома символами «--».

Наприклад, за допомогою інструкції:

```
SELECT Прізвище, Адреса, Телефон
FROM Школа
WHERE Посада = 'вчитель';
```

із таблиці Школа вибираються всі записи, у полі Посада яких є значення вчитель. Результуючий набір записів містить поля Прізвище, Адреса, Телефон.

Конструкції у фігурних дужках є обов'язковими, у квадратних дужках — необов'язковими.

Багато термінів, понять, синтаксичних правил і властивостей мови SQL збігаються з відповідними назвами класичних мов програмування. Усі конструкції мови (ключові слова, оператори тощо) однаково сприймаються і великими, і малими літерами.

За стандартом мови SQL для імен об'єктів (таблиць, полів та ін.) використовується латинський алфавіт, але в багатьох випадках допускається використання й національного алфавіту.

Розглянемо порядок створення найпростішого запиту за допомогою мови SQL на прикладі.

Приклад. Створити найпростіший запит на основі таблиці КАДРИ за допомогою мови запитів SQL.

1. У системі Access 2016 виконаємо команду Створити → Макет запиту.
2. Закриємо вікно Відображення таблиці.
3. На вкладці Конструктор у групі Результати клацнемо кнопку SQL. Відкриється вікно Запит1, у робочому полі якого висвітлиться оператор SELECT. Він обов'язково використовується з оператором FROM і має таку мінімальну загальну структуру:

SELECT <список імен полів>
FROM <ім'я таблиці>;
Наприклад, інструкція:
SELECT*
FROM <ім'я таблиці>;
забезпечує виведення всіх полів таблиці.

Після виконання інструкції:
SELECT Справа, Прізвище, Освіта, Стаж
FROM КАДРИ;
отримаємо результат, як наведено на рис. 4.9.

Запит1

Справа	Прізвище	Освіта	Стаж
105	Сокіл Т. Л.	середня	27
109	Шрамко Т. Л.	середня	24
120	Рябко Р. П.	вища	8
111	Семко М. М.	середня	16
116	Раков Г. П.	вища	19
132	Таран В. Д.	вища	15
115	Горошко Ф.Р.	середня спеціальна	17
0			0

Рис. 4.9. Записи таблиці КАДРИ з окремими полями

Як бачимо, виводяться записи з полями, зазначені в операторі SELECT.

Зауважимо, що порядок розміщення полів у реченні SELECT може бути довільним і не збігатися з порядком їх розміщення в початковій таблиці.

Поля виводяться в порядку їх розміщення у реченні SELECT.

Запитання для перевірки знань

1. Які оператори містить найпростіша інструкція мовою SQL?
2. Поясніть загальний формат найпростішої інструкції мовою SQL.
3. Наведіть приклад найпростішої інструкції мовою SQL.
4. Які дужки використовують для обов'язкових конструкцій?

4.3. Імпорт і експорт об'єктів баз даних

Чому, на вашу думку, виникає потреба імпортувати й експортувати дані із БД в інші об'єкти?

На практиці досить часто доводиться одночасно працювати не з одним файлом БД, а з кількома, у тому числі розміщеними в Інтернеті. У таких випадках виникає потреба в обміні даними (об'єктами) між БД, створеними за допомогою однієї СУБД (наприклад, Access 2016), різних СУБД (наприклад, Access і Paradox).

Окрім того, інколи буває потрібно обмінятися даними між Access та іншими програмами пакета Windows, наприклад Word-, Excel-, Outlook-, HTML- і XML-документами.

Найпростішим способом обміну об'єктами між Windows-програмами є копіювання об'єктів і вставлення за допомогою буфера обміну. Наприклад, у документ, створений у текстовому процесорі Word, можна додати рисунок, створений у графічному редакторі Paint.

У системі Access 2016 для обміну об'єктами здійснюють спеціальні операції, що отримали назву імпорту й експорту. Крім того, для отримання даних із інших зовнішніх джерел системою Access підтримується технологія зв'язування, яка тут не розглядається.

У процесі імпортування об'єкти перетворюються у формат Access 2016 і розміщуються в новому об'єкті, а в об'єкті-джерелі залишаються без змін. Наприклад, нова і початкова таблиці існують незалежно одна від одної. Технології імпорту з різних систем (Word, Excel та ін.) відрізняються одна від одної. Але загальний порядок імпортування однаковий для всіх систем.

Етап 1	Відкриття БД-приймача і вибір системи, з якої здійснюватиметься імпортування (наприклад, Access, Excel).
Етап 2	Вибір об'єктів, які необхідно імпортувати, і налаштування параметрів їх імпортування (у разі необхідності).
Етап 3	Виконання імпортування.

За своїм змістом експорт є операцією, зворотною до імпорту. У процесі експортування здійснюється перенесення даних із об'єктів системи Access в іншу БД цієї системи, а також у зовнішні файли різних форматів. Під час експортування дані перетворюються у новий формат, а об'єкти-джерела залишаються незмінними.

Оскільки загальні принципи імпортування й експортування всіх типів об'єктів БД ідентичні, далі розглядатимемо лише технологію та правила імпортування об'єктів із однієї БД Access 2016 в іншу БД Access 2016.

Для отримання об'єктів у БД із зовнішніх джерел в Access 2016 використовується **технологія імпорту**, а для передавання об'єктів в інші застосунки — **технологія експорту**.

Імпортувати можна будь-які об'єкти БД. Одночасно можна імпортувати як кілька об'єктів одного типу (наприклад, кілька запитів), так і кілька об'єктів різного типу (наприклад, кілька таблиць і запитів).

Розглянемо технологію імпортування на прикладі.

Приклад. Здійснити імпорт таблиці КАДРИ і запиту Запит_1 із БД abc у БД skola.

Отже, джерелом є база abc, а приймачем — база skola.

1. Відкриємо БД-приймач skola, активуємо вкладку Зовнішні дані. Панель інструментів набуде вигляду, як зображено на рис. 4.10.

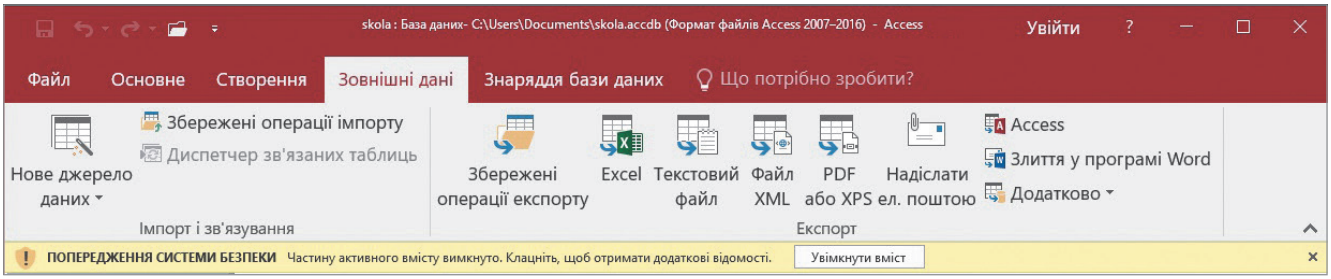


Рис. 4.10. Панель інструментів вкладки **Зовнішні дані**

2. У групі Імпорт і зв'язування відкриємо меню кнопки Нове джерело даних, установимо в ньому курсор на назві Із бази даних. Відкриється перелік джерел, як наведено на рис. 4.11.
3. Оскільки виконується імпорт об'єктів із однієї БД Access в іншу БД Access, клацнемо кнопку Access. Відкриється перше вікно Майстра імпорту, як наведено на рис. 4.12.

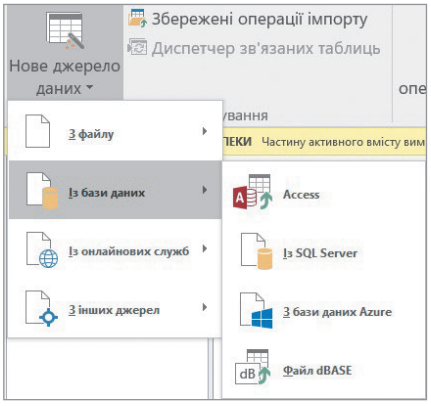


Рис. 4.11. Перелік джерел даних для імпортування

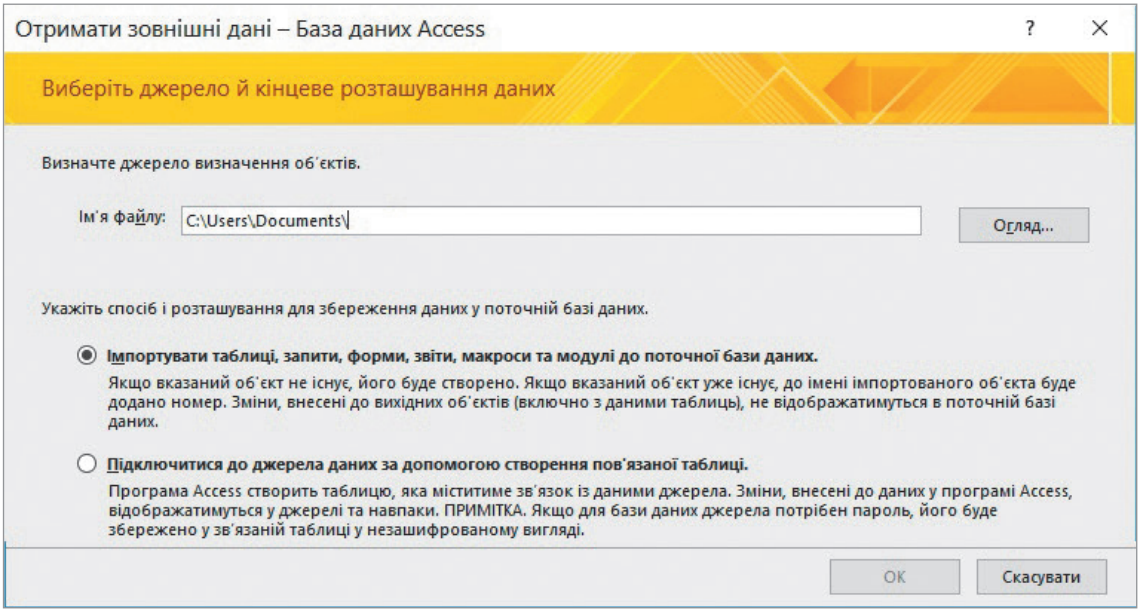


Рис. 4.12. Перше вікно **Майстра імпорту**

У поле Ім'я файлу цього вікна можна ввести повне ім'я файла БД-джерела. Можна також скористатися кнопкою Огляд... і вибрати файл abc. Клацнемо кнопку Огляд та знайдемо ім'я abc, увімкнемо перемикач Імпортувати таблиці, запити... і клацнемо кнопку ОК. Відкриється вікно Імпортувати об'єкти (рис. 4.13).

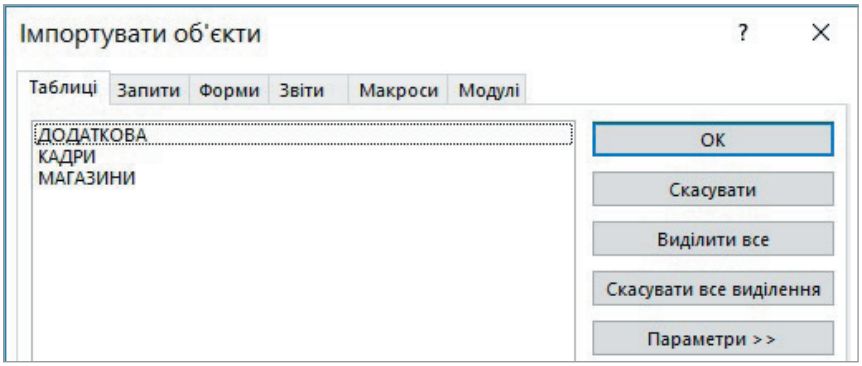


Рис. 4.13. Вікно для імпортування об'єктів

У цьому вікні містяться вкладки об'єктів БД-джерела abc. На вкладці Таблиці виберемо таблицю КАДРИ, для чого встановимо курсор на її імені й клацнемо праву кнопку миші. Потім на вкладці Запити виберемо Запит_1.

4. Можна налаштувати деякі параметри імпортування вибраних об'єктів. Для цього клацнемо кнопку Параметри>>. У результаті у вікні відобразиться область налаштування параметрів імпорту (рис. 4.14).

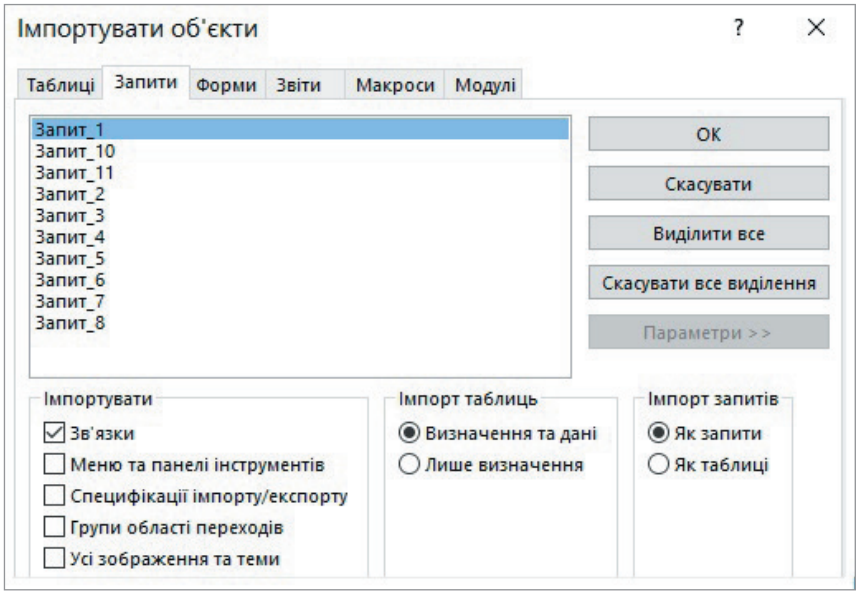


Рис. 4.14. Вікно з областю налаштування параметрів імпорту

Розглянемо призначення параметрів групи Імпорт таблиць.

Перемикач Визначення та дані встановлюється в тому випадку, коли з БД-джерела необхідно імпортувати і структури, і дані всіх таблиць, вибраних користувачем у допоміжному вікні.

Якщо вибрати перемикач Лише визначення, то імпортуватимуться тільки структури таблиць, дані не імпортуватимуться. Увімкнемо перший перемикач.

Зазначимо, що Запит_1 створено лише на основі таблиці КАДРИ, яку вже імпортовано. Таким чином, для його запуску в БД skola

жодних операцій виконувати не потрібно, увімкнемо перемикач Як запити.

5. Для збереження всіх налаштувань клацнемо кнопку ОК. Одразу почнеться імпортування вибраних об'єктів у БД skola. Після завершення імпортування на екран буде виведено повідомлення про результат виконання операції, як наведено на рис. 4.15.
6. Усі кроки, що виконувалися під час імпортування, можна зберегти для того, щоб за потреби можна було повторити їх без використання Майстра імпорту (за замовчуванням ці кроки не зберігаються). Для збереження виконаних кроків у вікні, що відкрито, необхідно ввімкнути прапорець Зберегти етапи імпортування. Але тут виконувати цю операцію немає потреби, тому просто клацнемо кнопку Закрити. Якщо тепер відрити БД skola, то в ній побачимо таблицю КАДРИ і Запит_1. Їх вміст такий самий, як у БД abc.

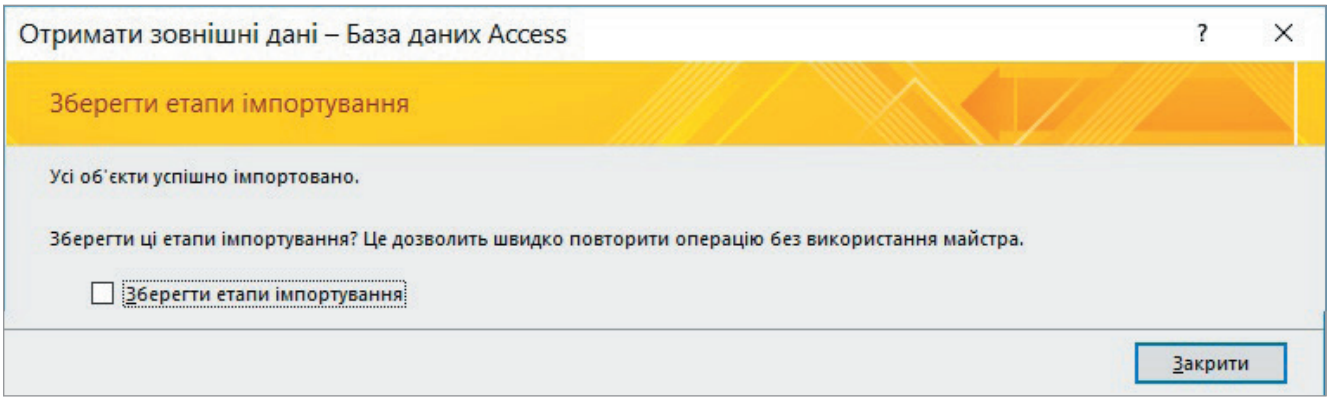


Рис. 4.15. Вікно для збереження етапів імпортування

Для імпортування таблиці із книги Excel потрібно відкрити БД-приймач, на вкладці **Зовнішні дані** клацнути кнопку **Excel** і вибрати команду **Майстра імпорту**.

Щоб імпортувати текстовий файл, його необхідно попередньо структурувати так, щоб кожний рядок файла був записом, а кожний запис поділявся на окремі поля.

Для розмежування полів часто використовується крапка з комою. Приклад структурованого текстового файла:

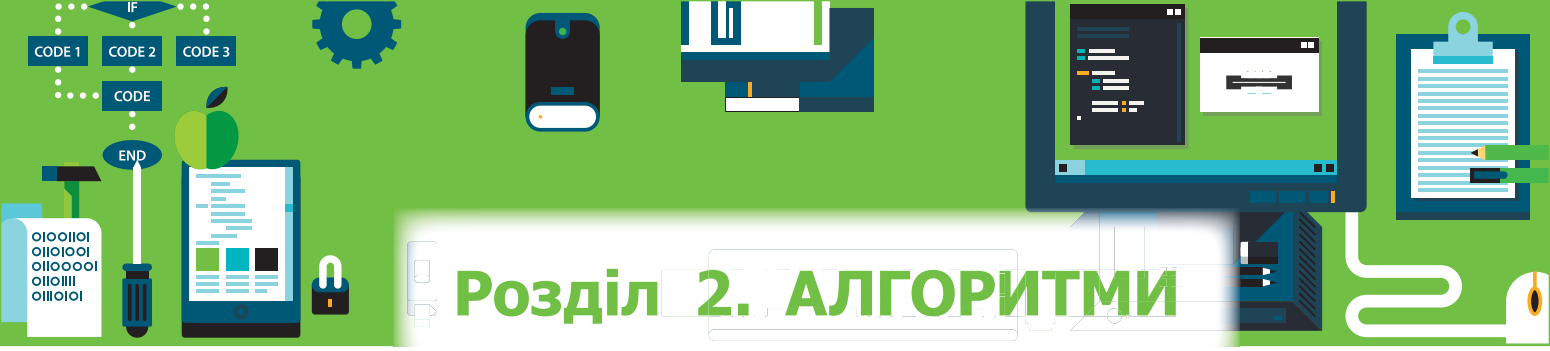
Микола; 050-400-22-33; 21 січня
Олена; 063-333-11-55; 5 травня

Запитання для перевірки знань

1. Опишіть найпростіший спосіб обміну даними між програмами Windows.
2. Між якими програмами можна здійснювати обмін даними з БД Access 2016?
3. Яка сутність імпорту даних у БД Access?
4. З якою метою здійснюється налаштування параметрів імпортування?
5. Поясніть загальний порядок імпортування даних у БД Access.
6. Що використовують для розмежування полів?

Завдання для самостійного виконання

1. Для виконання завдання створіть на жорсткому диску БД **persha**. Імпортуйте в БД **persha** форму з іменем **Форма_1** із БД atb. Перевірте правильність імпортування.



Розділ 2. АЛГОРИТМИ

5. Алгоритми і числа

5.1. Методи проектування і подання алгоритмів

Пригадайте, яких правил ви дотримувались у процесі розроблення програм у попередніх класах. У якій формі ви подавали алгоритми?

Проектування алгоритмів і програм є виключно творчим процесом. Не існує універсального методу розроблення алгоритму розв'язування для будь-якого завдання. Для кожного завдання необхідно знайти свій, найбільш раціональний.

У процесі проектування алгоритму намагаються:

- забезпечити мінімальний час розв'язування завдання;
- використати мінімальний обсяг необхідної пам'яті;
- досягти потрібної точності й надійності обчислення;
- забезпечити ефективне використання можливостей наявних бібліотек, зокрема мінімізувати вартість розроблення алгоритму.

Методи проектування алгоритмів класифікуються за багатьма ознаками. Основними з них є ступінь автоматизації проектування алгоритмів і програм та методологія проектування програмних продуктів (рис. 5.1.).

Неавтоматизовані методи використовуються у процесі розроблення невеликих і нескладних програмних продуктів за участю невеликої кількості розробників. Такі методи застосовуються, зокрема, у процесі розроблення програмних продуктів навчального призначення. *Автоматизовані* методи застосовуються у великих компаніях і потребують додаткового апаратно-програмного забезпечення і високої кваліфікації працівників.

В основі *структурного проектування програмних продуктів* лежать послідовна декомпозиція і структурування програмного продукту на окремі складові. *Структурне проектування програмних продуктів* засноване на створенні алгоритмів із базових структурних алгоритмічних одиниць. Доведено, що такими одиницями є: слідування, розгалуження і повторення (цикли). Ці алгоритмічні конструкції послідовно з'єднуються або укладаються одна в одну з дотриманням певних правил. Алгоритм виконується послідовно зверху вниз.

Правильність виконання алгоритму можна відслідковувати на кожному етапі його побудови і виконання. Будь-який алгоритм може бути еквівалентно поданий структурованим алгоритмом, що складається з базових алгоритмічних структур.

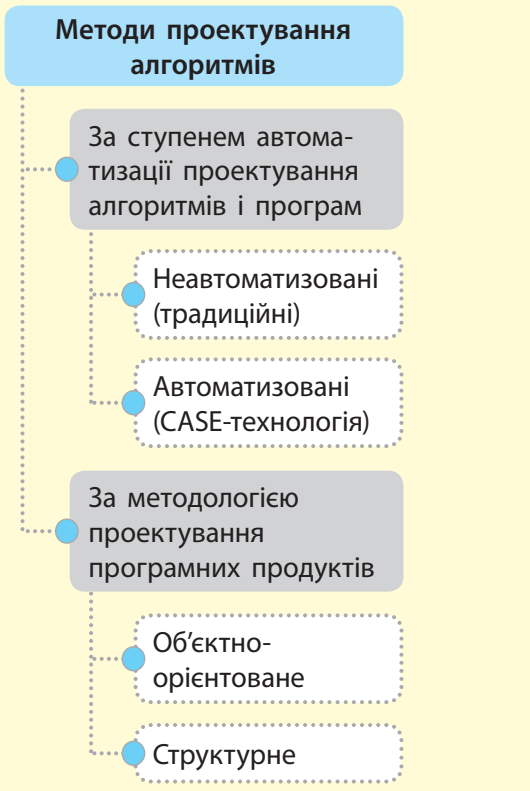


Рис. 5.1. Класифікація методів проектування алгоритмів

Розглянемо типові методи структурного проектування програмних продуктів:

Метод	Опис
Спадний (згори вниз)	Його сутність полягає в тому, що задача поступово (за кроками) ділиться на ряд допоміжних підзадач (підалгоритмів), кожна з яких може бути реалізована сукупністю простих і елементарних операцій (процедур).
Висхідний (знизу догори)	Вже наявні й заздалегідь розроблені підалгоритми розв'язування окремих підзадач поступово об'єднуються в загальну структуру доти, доки не буде досягнуто розв'язання поставленого завдання.
Модульний	Модуль — це окрема самостійна частина алгоритму (деякий блок), що має свою назву, функціональну цілісність і завершеність. Посилання на модуль здійснюється за допомогою його імені. Виклик і актуалізація модуля можливі лише через його заголовок. Перевага модульного методу полягає в тому, що різні модулі одночасно можуть розробляти різні фахівці. Кожний модуль може тестуватися і налагоджуватися окремо від інших.

Об'єктно-орієнтоване проектування програмних продуктів засноване на тому, що кожний об'єкт об'єднує дані й програми (методи) їх опрацювання в єдину конструкцію. Кожний об'єкт належить до відповідного класу.

У процесі реалізації такого підходу створюється ієрархія класів, визначаються властивості об'єктів і розробляються методи їх опрацювання, а також дотримуються специфічні принципи об'єктно-орієнтованого програмування.

Розглянемо способи подання алгоритмів:

Спосіб	Опис
Словесний	Передбачає опис алгоритму природною мовою, широко застосовується у повсякденному житті (наприклад. у вигляді інструкцій з експлуатації приладів, рецептів виготовлення ліків тощо). Інструкція складається з указівок, форма запису яких довільна. Головне, щоб указівки були точними й зрозумілими всім користувачам. Словесний спосіб є досить простим і доступним, проте опис алгоритмів часто є громіздким, а їхні вказівки можуть сприйматися різними виконавцями неоднозначно.
Словесно-формульний	Використовує природну мову, математичні вирази, а також спеціальні символи. Наприклад, широко застосовується оператор присвоювання (:=). У мові Python цей оператор позначається символом «=» (дорівнює). **Наприклад,** розв'яжемо квадратне рівняння ax^2+bx+c. 1. Уведемо значення змінних a, b, c. 2. Обчислимо дискримінант $d := b^2 - 4ac$. 3. Якщо $d<0$, то виконаємо пункт 6, інакше — пункт 4. 4. Знайдемо корені рівняння: $x_1 = \frac{-b-\sqrt{d}}{2a}$; $x_2 = \frac{-b+\sqrt{d}}{2a}$. 5. Перейдемо до пункту 7. 6. Дискримінант від'ємний. Рівняння не має розв'язків. 7. Кінець.

Графічний	Передбачає подання алгоритму у вигляді геометричних фігур (блоків), з'єднаних стрілками (лініями зв'язку). Подання алгоритмів за допомогою блоків називають блок-схемами, вони мають високу наочність. На рис. 5.2 зображено основні геометричні фігури (блоки) певного вигляду, за допомогою яких створюються блок-схеми алгоритмів.
Комбінований	Поєднує в собі подання алгоритмів, засноване на словесному, словесно-формульному і графічному методах.

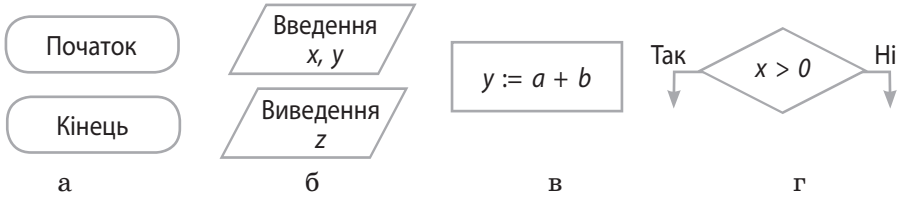


Рис. 5.2. Основні графічні позначення на блок-схемах: термінатор (а); дані (б); процес (в); розгалуження (г)

Запитання для перевірки знань

1. Які існують методи подання алгоритмів?
2. За якими ознаками класифікують методи проектування програмних продуктів?
3. Назвіть методи структурного проектування.
4. Поясніть сутність структурного проектування програмних продуктів.
5. У чому полягає сутність модульного проектування програмних продуктів?
6. Які критерії слід ураховувати під час проектування алгоритму?
7. Назвіть переваги й недоліки кожного методу проектування алгоритмів.

Завдання для самостійного виконання

1. Користуючись словесним способом подання, опишіть алгоритм виправлення помилок у слові **алгарітм** за допомогою текстового процесора Word, щоб отримати слово **алгоритм**.
2. Відомо, які оцінки з усіх навчальних предметів отримали за перше півріччя учениця 11 класу Анастасія та учень цього класу Володимир. Розробіть графічну схему алгоритму визначення, у кого з них середній бал успішності вищий.
3. Знайдіть в Інтернеті відстані від Києва до Житомира, Чернігова і Білої Церкви. Користуючись словесно-формульним способом, розробіть алгоритм визначення, яке з цих міст розташоване ближче до Києва.
4. Знайдіть в Інтернеті відомості про п'ять найбільших за кількістю населення міст Хмельницької та Черкаської областей. Користуючись словесно-формульним способом, розробіть алгоритм визначення, у якому місті мешкає людей найбільше і до якої області воно належить.
5. Розробіть графічну схему алгоритму обчислення виразу:

$$y=\begin{cases}\left(a^3+b^2\right)\cdot c-1, & \text{якщо } a=0,\\ a^3+b^2, & \text{якщо } a\neq 0.\end{cases}$$

6. Розробіть графічну схему алгоритму обчислення виразу:

$$y=\begin{cases}a^2-bx, & \text{якщо } x>0,\\ a^2-\left(c+\dfrac{x}{b}\right), & \text{якщо } x\leqslant 0.\end{cases}$$

7. У банк покладено S1 грн під k відсотків річних. Розробіть графічну схему алгоритму визначення, через скільки років сума вкладу перевищуватиме S2 грн.

5.2. Поняття про кодування і складність алгоритмів

З якими мовами програмування ви вже знайомі? Чи можна один і той самий алгоритм реалізувати мовою програмування різними способами?

Кодування алгоритму — це запис алгоритму мовою програмування.

Для того самого алгоритму можна розробити різні варіанти програм, які відрізняються наочністю, обсягом потрібної пам'яті, швидкістю виконання, формою подання отриманих результатів та ін.

Наочність програмного коду досягається за рахунок чіткої структуризації тексту програми і виділення блоків команд. Структурування в мові програмування Python виконується автоматично. Це означає, що відступи інструкцій кожного блоку встановлюються автоматично, але після останньої інструкції блоку програміст має самостійно змінити відступ.

Приклад структурованого програмного коду мовою Python зображено на рис. 5.3.

File Edit Format Run Options Window Help

```
kom = int(input("Увести номер команди: "))
if (kom==1 or kom==2 or kom==5):          #команда Іспанії
    kr = "Іспанія"
else:
    if (kom==3 or kom==7):                #команда Німеччини
        kr = "Німеччина"
    else:
        if (kom==4 or kom==9 or kom==10):  #команда Англії
            kr = "Англія"
        else:
            if (kom==6 or kom==8):        #команда Португалії
                kr = "Португалія"
            else:
                kr="Рейтинг невідомий"
print ("", kr)
input ()
```

Рис. 5.3. Приклад структурованої програми

У програмі слід використовувати раціональну довжину рядків, щоб її можна було легко модифікувати. Не бажано застосовувати довгі рядки.

Така програма є досить наочною, її легко читати, розуміти та здійснювати пошук помилок. Простим і водночас потужним засобом забезпечення наочності програми є використання коментарів.

У програмному коді доцільно використовувати не однобуквені ідентифікатори, а смислові назви, так, масив краще позначити не буквою m, а ідентифікатором massif або іншим.