ЛЕКЦІЯ №9

Тема: ЗАПИТИ

ПЛАН

9.1 Види запитів та способи їх створення
9.2 Види з'єднань
9.3 Запити з параметрами
9.4 Створення запиту з фільтра
9.5 Виконання запиту і його зміна
9.6 Створення обчислюємих полів у запитах
9.7 Форми і звіти

Час: 2 год.

Література: [1-3, 5].

9.1 Види запитів та способи їх створення

Після введення даних у таблиці Access можна постановки перейти до постановки запитів до бази даних. Для побудови запиту в Access використовується метод ***QBE*** (*Query By Example* – запит за зразком). Запити забезпечують простий доступ до певної підмножини полів і записів однієї чи кількох таблиць. Запити використовуються для перегляду, аналізу й зміни даних в одній або декількох таблицях.

SQL (англ. *Structured Query Language* - «мова структурованих запитів») – універсальна комп'ютерна мова, що застосовується для створення модифікації та управління даними в реляційних базах даних.

SQL є перш за все, інформаційно-логічною мовою, призначеною для опису, зміни і вилучення даних, які зберігаються в базі даних. Кожне речення SQL – це або запит даних з бази, або звернення до бази даних, яке призводить до зміни даних в базі.

Існує кілька видів запитів:

- запити, що автоматично заповнюють поля для нового запису;
- запити на вибірку, що виконують вибірку даних з таблиць;
- запити на зміну, які дають можливість модифікувати дані в таблицях (у тому числі видаляти, обновляти й додавати записи);
- запити на створення таблиці.

Для створення запиту вручну виконуємо такі дії:

1. У вікні бази даних відкриваємо вкладку **Запросы**. Даємо команду **Создать**. З'явиться діалогове вікно **Новый запрос**.
2. У діалоговому вікні **Новый запрос** виберемо пункт **Конструктор** і натиснемо кнопку **ОК**. У результаті відкриється вікно конструктора запиту з діалоговим вікном **Добавление таблицы**.

3. На вкладці **Таблицы** або **Таблицы и запросы** вибираємо потрібні таблиці та натискаємо кнопку **ОК**.
4. Після закриття діалогового вікна **Добавление таблицы** на весь екран розкриється вікно конструктора запиту, яке містить вікно таблиць і запитів, що використовуються у створюваному запиті.

Вікно конструктора запиту поділено на дві частини. У верхній частині міститься вікно таблиць і запитів, які використовуються у створюваному запиті, а нижня частина містить бланк запиту QBE, в якому задаються поля, ім’я таблиці, порядок сортування, умови відбору.

Для додавання полів у запит переміщуємо їх з вікна таблиць у рядок **Поле**. Потім у рядку **Условия отбора** задаємо умови відбору у вигляді виразів. У полях **Сортировка** задаємо порядок сортування **По возрастанию** або **По убыванию**. В рядку Вывод можна управляти виводом на екран полів.

5. Після створення запиту й установки всіх його параметрів запит можна виконати. Для цього натискаємо на кнопку **Запуск** панелі інструментів або використовуємо команду **Запрос/Запуск.** Запит відбирає дані і відображає результати в динамічному наборі.

9.2 Види з'єднань

Залежно від розв'язуваних завдань з'єднання між таблицями в запитах може бути декількох видів. Найбільш поширеним є ***внутрішнє з'єднання***.

Якщо таблиці пов'язані відношенням "один-до-багатьох", з'єднання ґрунтуються на унікальному значенні поля первинного ключа в одній таблиці та значеннях поля зовнішнього ключа в іншій таблиці. У результуючу безліч запиту потрапляють всі записи з головної таблиці (таблиці на стороні "один"), для яких існують відповідні записи в підлеглій таблиці (таблиці на стороні "багато хто"). Якщо в підлеглій таблиці записи із заданою величиною відсутні, то відповідні записи в головній таблиці в результуючу безліч не включаються.

У ході розробки баз даних, в яких передбачається використання запитів на основі внутрішніх з'єднань, дотримуйтеся наступних правил.

- Кожна таблиця "один" повинна мати первинний ключ із унікальними значеннями. Відсутність повторення значень поля або полів первинного ключа в таблиці Access установлює автоматично.
- Відношення "багато-до-багатьох" реалізуйте на основі проміжної таблиці, що пов'язана з кожною із двох таблиць відношенням "багато -до-одного". Для обох зв'язків проміжна таблиця буде перебувати на боці "багато ".
- Необхідно витягти дані, що повторюються, в нову таблицю й пов'язати її з таблицею, з якої ці дані були отримані, відношенням "багато хто-до-одного". Основна мета - однозначно визначити витягнуті дані. Частіше для цього доводиться використовувати первинний ключ, що складається з декількох полів.

Для створення запиту, що поєднує всі записи з однієї таблиці, та тільки ті записи із другої, у яких пов'язані поля співпадають, використовують ***зовнішнє з'єднання***. У цьому випадку незалежно від того, чи є відповідні записи в другій таблиці, всі записи першої потрапляють у результуючу безліч запиту. Зовнішні з'єднання бувають лівими або правими. Запит, у якому беруть участь таблиці з лівим зовнішнім з'єднанням, виводить усі записи таблиці “один”, в незалежності від того, чи є відповідні їм записи в таблиці "багато хто".І навпаки, запит, у якому беруть участь таблиці із правим зовнішнім з'єднанням, виводить всі записи таблиці "багато хто", у незалежності від того, чи є відповідні їм записи в таблиці "один".

Якщо необхідно зв'язати дані будь-яким відношенням, крім відношення рівності, використовують ***з'єднання по відношенню***.

Для зв'язування даних в одній таблиці застосовують ***рекурсивне з'єднання***. Воно створюється шляхом додавання в запит копії таблиці (у результаті чого Access призначає псевдонім для копії) і зв'язування полів ідентичних таблиць.

9.3 Запити з параметрами

Запит в Access є **об'єктом**, що зберігається у файлі бази даних і може багаторазово повторюватися. Якщо потрібно повторити який або запит з іншими значеннями в умовах відбору, його потрібно відкрити в режимі Конструктора, змінити умову й виконати. Щоб не робити багаторазово цих операцій, можна створити **запит з параметрами**. При виконанні такого запиту видається діалогове вікно **Введіть значення параметра**, у якому користувач може ввести конкретне значення й потім отримати потрібний результат.

9.4 Створення запиту з фільтра

Ще одним способом створення запиту є збереження фільтра у вигляді запиту. Ми говорили, що фільтр використовується при перегляді таблиці для відбору потрібних записів. Якщо ми створили досить складний фільтр і знаємо, що його доведеться використовувати й надалі, можна зберегти його в базі даних у вигляді запиту. Для цього досить нажати кнопку **Зберегти як запит** на панелі інструментів при відкритому вікні **Фільтр**.

9.5 Виконання запиту і його зміна

Щоб подивитися результати запиту, досить відкрити його в режимі **Таблиці**. Для цього необхідно:

1. У вікні бази даних на панелі об'єктів вибрати ярлик **Запити**.

2. Виділити потрібний запит у списку запитів і нажати на кнопку **Відкрити** або двічі клацнути лівою кнопкою миші на потрібному запиті.

Після цього на екрані з'являється таблиця, що містить тільки ті записи, які задовольняють критеріям відбору, зазначеним у запиті, і ті поля, які зазначені в бланку запиту. Якщо в запиті зазначений порядок сортування записів, вони виводяться на екран у відповідному порядку.

Зміни зберігаються в записах базової таблиці, на основі якої побудований запит. Якщо запит створений на основі двох і більше пов'язаних таблиць, то не завжди стовпці в запиті можна редагувати. Оскільки ця таблиця є віртуальною, насправді редагування полів запиту означає редагування полів у таблицях, на основі яких будувався запит. Однак таке редагування не завжди можливо, і при спробі змінити значення якогось поля ви можете одержати повідомлення, що дані в запиті не є обновлюваними, або просто звуковий сигнал.

Будь-який запит має дві властивості: **Унікальні значення** й **Унікальні записи**. Властивість **Унікальні записи** вилучає з результуючого набору дубльовані записи, тобто в результат запиту будуть включені записи, які мають унікальні значення хоча б в одному з полів. Властивість **Унікальні значення** вимагає включення в результуючий набір тільки тих записів, які мають унікальні значення в усіх полях. Значення цих властивостей не можуть одночасно бути рівними Так (Yes), хоча обоє можуть мати значення **Ні** (No). Значення цих властивостей можуть установлюватися у вікні властивостей запиту. Більшість запитів, властивість **Унікальні записи** яких має значення Так (Yes), можуть використовуватися для відновлення даних. Запити, властивість **Унікальні значення** яких має значення Так (Yes), не допускають ні відновлення наявних у них записів, ні додавання нових.

Неможливо додавати й змінювати записи в запитах, якщо:

- дві таблиці запиту пов'язані відношенням "один-до-багатьох" і при цьому в таблиці "один" не задано полів первинного ключа;
- у запиті використовуються рекурсивні з'єднання; П у запиті застосовуються статистичні функції ***SQL***.

Можливо додати або обновити записи в запитах, якщо:

- таблиця є єдиною в запиті;
- таблиці в запиті зв'язані відношенням "один-до-одного";
- якщо таблиці в запиті зв'язані відношенням "один-до-багатьох", можна змінювати поля тільки в таблиці "багато ".

9.6 Створення обчислюємих полів у запитах

Ви можете створювати стовпці в запиті, які є результатом обчислень над значеннями інших стовпців. Такі стовпці називаються обчислюємі. Це суттєво розширює можливості запитів. Найпростішим прикладом

обчислюємого поля у запиті може бути поле, що поєднує ім'я та прізвище людини.

Щоб створити обчислюєме поле, потрібно ввести вираз, який обчислює необхідне значення, у рядок **Поле** вільного стовпця бланка запиту або використати **Будівник виразів**, викликавши його щигликом по кнопці **Побудувати** на панелі інструментів.

Використання виразів у запитах

Вирази активно використовуються в запитах для опису критеріїв вибірки записів, як уже згадувалося раніше. Кожний вираз може містити одного або декілька операторів та одну або декілька констант, ідентифікаторів або функцій.

Константи — характеризують незмінні значення. Їх часто використовують для створення значень за замовчуванням і для порівняння значень в полях таблиць.

Ідентифікатори — це імена об'єктів в Access (наприклад, полів таблиць або запитів), які при обчисленні виразів замінюються їхніми поточними значеннями.

Функції повертають у вираз значення замість імені функції. На відміну від ідентифікаторів, більшість функцій вимагають брати в дужки свої аргументи (ідентифікатори або значення під вираз).

Для створення виразів в Access існує шість категорій операторів:

- арифметичні;
- оператори присвоювання;
- логічні оператори;
- оператори конкатенації;
- ідентифікації
- порівняння із зразком.

Арифметичні оператори, як виходить з назви, виконують додавання, віднімання, множення й ділення. Арифметичні оператори оперують тільки із числовими значеннями й повинні, за винятком ***унарного мінусу***, мати два числових операнда.

Звичайно в якості ***оператора присвоювання*** значення об'єкту, змінної або константі використовується знак рівності (=). Наприклад, вираз =Now() може присвоювати полю таблиці значення за замовчуванням, і тоді знак рівності діє як ***оператор присвоювання***. З іншого боку, знак = являє собою оператор порівняння, що визначає, чи рівні два операнда. Оператор порівняння співвідносить значення двох операндів і повертає логічні значення (*True або False*), що відповідають результату порівняння. Основне призначення операторів порівняння - створення умов на значення,

установлення критеріїв вибірки записів у запитах, визначення дій макросів і контроль виконання програм в VBA.

Логічні оператори використовуються для об'єднання результатів двох або більше виразів порівняння в єдине ціле:

And – кон'юнкції (логічного І);

Or - диз'юнкції (логічного АБО);

Not - логічного заперечення;

Хог - що виключає АБО;

Eqv - логічної еквівалентності;

Imp - логічної імплікації.

Вони можуть складатися тільки з виразів, що повертають логічні значення *True, False або Null*. У противному випадку виконується побітове порівняння. Логічні оператори завжди вимагають двох операндів, за винятком *Not* - логічного еквівалента унарного мінуса.

Оператори злиття строкових значень (конкатенації) . Стандартний значок оператора конкатенації ***SQL***, амперсант (&), є більш кращим, ніж значок плюса (+), хоча обоє вони приводять до однакового результату: об'єднанню двох текстових значень у єдиний рядок символів. Застосування значка плюс (+) двозначно, його основне призначення - додавання двох числових операндів.

Приклад: злиття "*Visual" & "Basic" дає "Visual Basic*". Зверніть увагу на додатковий пробіл у першому слові, без нього результат виглядав би трохи інакше: "*Visual Basic*".

Оператори ідентифікації. Будь-який об'єкт Access має ім'я, по якому його можна однозначно ідентифікувати в деякій системі об'єктів. Крім позначення коротким ім'ям об'єкта, ідентифікатор можна позначити кваліфікованим (або повним) ім'ям, коли об'єкт ідентифікується як один з об'єктів у сімействі об'єктів. У цьому випадку ім'я ідентифікатора складається з імені сімейства (клас об'єкта), відокремленого від привласненого імені (імені об'єкта) знаком оклику або крапкою (символами операції ідентифікації "!" й "."). Тому імена об'єктів не повинні містити символів "!" й "." В Access для розподілу імен таблиць й імен полів використовується "!", а крапка розділяє об'єкти і їхні властивості. Використовуючи ідентифікатори, можна повертати значення полів в об'єкти форм і звітів, а також будувати нові вирази.

Оператори ідентифікації застосовуються в якості роздільників в посиланнях на об'єкти (оператор "!"), їхні методи або властивості (оператор "."):

КласОб’єкта!Ім’яОб’єкта

КласОб’єкта!Ім’яОб’єкта.Властивість

КласОб’єкта!Ім’яОб’єкта.Метод()

Ім’яОб’єкта.Властивість

Ім’яОб’єкта.Метод().

Ці оператори дозволяють поєднувати імена об'єктів і класів об'єктів для відбору специфічних об'єктів або їхніх властивостей, розрізняти імена об'єктів та їхніх властивостей, ідентифікувати визначені поля в таблицях.

Наприклад:

Forms!Категорії, Tables ! Категорії — ідентифікація форми й таблиці з однаковими іменами;

MyTextbox.Caption = "Будьте уважні!" — тут MyTextbox — об'єкт керування, а Caption — властивість;

Замовлення! [Код клієнта] - визначає поле "Код клієнта" (Customerl) в таблиці "Замовлення" (Customers).

Оператори порівняння зі зразком. Інші оператори Access спрощують створення виразів для вибірки записів у запитах і відносяться до операторів порівняння зі зразком. Ці оператори повертають *True* або *False*, залежно від відповідності значення в полі обраної специфікації оператора. Наявність цих операторів в умовах на значення дозволяє або включати запис у запит, якщо логічне значення, повертаєме виразом, дорівнює True, або відхиляє, якщо це значення - False.

8.7 Структурована мова запитів (SQL)

Якщо після створення запиту виконати команду **Вид/Режим SQL**, то на екран буде виведена невелика програма мовою SQL.

Початкова версія мови SQL (Structured English Query Language) базувалася на раніше розробленій мові square. Прототип цієї версії SQL був реалізований у науково-дослідній лабораторії фірми IBM у 1874 році.

SQL забезпечує повний набір операцій запам’ятовування – Insert (Включити), Delete(Видалити), Update (Обновити), а також інші можливості.

Основною операцією в SQL є відображення, яке синтаксично є блоком SELECT-FROM-WHERE (ВИБРАТИ-З-ДЕ). Наприклад, запит «Одержати номери та статути постачальників, які перебувають у Парижі», можна виразити таким чином:

```
SELECT S#, STATUS
FROM S
WHERE CITY=”PARIS”
```

8.8 Форми і звіти

Формами називаються діалогові вікна, що набудовують, що зберігають у базі даних у вигляді об'єктів спеціального типу. Форми використаються в додатку для уведення й відображення даних. Форми містять так називані елементи керування, за допомогою яких здійснюється доступ до даних у таблицях.

Найпростішим способом створення форм є використання засобів автоматичного створення форм на основі таблиці або запиту. Автоматично створювані форми (автоформи) бувають декількох видів:

- Автоформа, організована "у стовпець". У такій формі поля кожного запису відображаються у вигляді набору елементів керування, розташованих в один або кілька стовпців.
- Таблична. Форма буде виглядати так само, як звичайна таблиця Access.
- Стрічкова. У такій формі поля кожного запису розташовуються в окремому рядку.
- Автоформа у вигляді зведеної таблиці або зведеної діаграми.

Автоматично створена форма включає всі поля обраного джерела даних. Щоб створити форму за допомогою засобу автоматичного створення форм:

1. Клацніть по ярлику **Форми** у вікні бази даних і натисніть кнопку **Створити**. З'явиться діалогове вікно **Нова форма**.
2. У списку діалогового вікна **Нова форма** виділите один з варіантів автоформы.

За допомогою майстра можна створювати форми на основі однієї таблиці й більше складні форми на основі декількох таблиць і запитів, що мають підлеглі форми:

1. Клацніть по ярлику **Форми** у вікні бази даних.
2. Натисніть кнопку **Створити** на панелі інструментів вікна бази даних.

У списку варіантів у діалоговому вікні, що **з'явилося, Нова** форма виділите **елемент Майстер** форм і натисніть **кнопку** ОК.

Будь-яка форма може включати наступні розділи:

- розділ **Заголовок** форми визначає верхню частину форми. Цей розділ додається у форму разом з розділом примітки форми. В область заголовка форми можна помістити текст, графіку й інші елементи керування. При печатці багатосторінкової форми розділ заголовка відображається тільки на першій сторінці;
- розділ **Область** даних визначає основну частину форми, що містить дані, отримані із джерела. Даний розділ може містити елементи керування, що відображають дані з таблиць і запитів, а також незмінні дані, наприклад напису. При печатці многостраничной форми цей розділ відображається на кожній сторінці;
- розділ **Примітка** форми визначає нижню частину форми. Цей розділ додається у форму разом з розділом заголовка форми. При печатці багатосторінкової форми примітка форми буде відображено тільки внизу останньої сторінки.

В Access існує три різновиди *елементів керування*, залежно від типу вмісту в них, тобто від способу заповнення їхніми даними:

- приєднані
- вільні

- обчислювальні

Приєднані елементи керування пов'язані з полями базової таблиці, тобто тієї таблиці, що є джерелом даних для форми. Якщо джерелом даних є запит, то приєднані елементи керування можуть зв'язуватися з полями в різних таблицях. У приєднаному елементі відображаються дані, які втримуються в пов'язаному з ним поле таблиці, і при зміні цих даних відповідним чином обновляється й значення в поле таблиці. У приєднаних елементах можна відображати всі доступні в Access типи даних, у тому числі об'єкти OLE і гіперпосилання.

Вільні елементи керування не пов'язані з таблицями. Вони призначені або для уведення інформації, що використається не для безпосереднього редагування даних у джерелі, а в інших цілях (звичайно макросами або програмами VBA), або для відображення об'єктів OLE, які зберігаються в самих формах. Вільними елементами є також всі елементи, не пов'язані з якими-небудь даними, а призначені лише для поліпшення візуального сприйняття форм, такі як лінії, прямокутники, малюнки.

Елементи керування, що обчислюють - це такі елементи, значення яких обчислюються на основі значень інших елементів. Як джерело даних для цих елементів використаються вираження й функції.

Основні елементи керування

Елемент **Напис** використається для розміщення у формі тексту: заголовків полів, заголовка форми, різних написів, що пояснюють

Элемент керування **Поле** служить для уведення й відображення даних. Звичайне значення в полі вводиться користувачем, однак можна задати його програмно, привласнюючи значення властивості Text цього елемента.

Для надання користувачеві можливості вибирати значення із запропонованого набору варіантів використають елементи керування: **прапорець**, **перемикач** і **групи елементів**.

Елемент керування **Список** використається для подання на екрані поля, можливі значення якого обмежуються списком.

Елемент керування **Поле зі списком** багато в чому аналогічний елементу керування **Список**, але з наступними відмінностями:

- **Поле зі списком** дозволяє не тільки вибирати значення зі списку, але й уводити його прямо в поле уведення;
- **Поле зі списком** не накладає таких обмежень на довжину списку, як **Список,** тому що в ньому відображається тільки поточне значення, а інші значення виводяться, коли користувач клацає мишею по стрілці вниз із правої сторони поля. Відповідно, елемент **Поле зі списком** займає на формі менше місця, чим елемент **Список;**
- **Поле зі списком** дозволяє вибрати тільки один елемент зі списку.

Елемент керування **Кнопка** є елементом, з яким зв'язані різні дії, виконувані користувачем у додатку (збереження уведених даних, виклик іншої форми, висновок на печатку документа й т.д.).

Елементи керування **Вільна рамка об'єкта** й **Приєднана рамка об'єкта** використаються для того, щоб вставити у форму об'єкти, створені в інших додатках, наприклад документ Word або таблицю Excel або малюнок.

Елемент керування **Малюнок** призначений для вставки у форму графічних файлів, але вимагає менше системних ресурсів і працює швидше, ніж **Вільна рамка об'єкта.** Формати графічних файлів, які підтримуються Access, різноманітні - це растрові малюнки, метафайлы, значки й ін.

Елемент керування **Набір вкладок використається** для створення многостраничных форм. У цьому елементі може бути від однієї до декількох вкладок. Перемикання між вкладками форми виконується щигликом кнопкою миші по ярлику вкладки. Вкладки можуть мати два варіанти оформлення: ярлики й кнопки.

Елемент керування **Підлегла форма/звіт** використається для відображення у формі іншої форми.

Елементи керування **Лінія** й **Прямокутник** використаються для оформлення зовнішнього вигляду форм, найчастіше для виділення групи логічно зв'язаних елементів керування.

Створення багатотабличних форм

Багатотабличні форми – це форми, побудовані на основі запиту, що поєднує кілька таблиць, а також форми із впровадженими в них іншими формами.

Підлеглою формою називається форма, що вбудовується в іншу форму. При цьому форма, що включає підлеглу форму, називається основною формою. Звичайно такі форми застосовуються для відображення даних зі зв'язаних таблиць, наприклад категорії товарів і товари, організації й контактні особи в цих організаціях і т.д. Застосування убудованих підлеглих форм забезпечує більше компактне подання на екрані даних з декількох таблиць, чим використання різних форм для кожної таблиці.

Звіти

Однієї з основних завдань створення й використання баз даних є надання користувачам необхідної інформації на основі існуючих даних. В Access для цих цілей призначені **форми** й **звіти**. Звіти дозволяють вибрати з бази даних необхідну користувачем інформацію й оформити її у вигляді документів, які можна переглянути й надрукувати. Джерелом даних для звіту може бути таблиця або запит. Крім даних, отриманих з таблиць, у звіті можуть відображатися обчислені за вихідним даними значення, наприклад підсумкові суми.

Звіти й форми Access мають багато загального. Однак, на відміну від форм, звіти не призначені для уведення й виправлення даних у таблицях. Вони дозволяють лише переглядати й друкувати дані. У звіті неможливо змінити вихідні дані за допомогою елементів керування, як це можна зробити за допомогою форм. Хоча у звітах можна використати такі ж елементи керування для вказівки стану перемикачів, прапорців і списків.

Звіт, як і форма, може бути створений за допомогою майстра. Розділи звіту подібні до розділів форми й включають заголовок і примітка звіту, область даних, а також верхній і нижній колонтитули. У примітку звіту часто поміщають поля з підсумковими значеннями. Форми можуть містити підлеглі форми, а звіти можуть містити підлеглі звіти.

Access надає цілий ряд можливостей перетворення звітів в інші формати: RTF (Word),XLS (Excel), HTML, XML й ін.

Контрольні запитання

1 Що називається запитом?
2 Які існують типи запитів?
3 Як створити запис вручну?
4 Опишіть види з’єднань.
5 Як створити запит з параметрами?
6 Коли неможна додавати і змінювати запити?
7 Коли можливо додавати і змінювати запити?
8 Що називається константами?
9 Що називається ідентифікаторами?
10 Що називається функцією?
11 Які категорії операторів використовуються в Access?
12 Опишіть арифметичні оператори.
13 Опишіть логічні оператори.
14 Опишіть оператори присвоювання.
15 Опишіть оператори злиття строкових значень.
16 Опишіть оператори ідентифікації.
17 Оператори порівняння зі зразком.
18 Опишіть сутність мови SQL.
19 Що називається формою?
20 На які види розділяються авто форми?
21 Як автоматично створити форму?
22 Які розділи включає в себе форма?
23 Опишіть основні елементи керування.
24 Для чого призначені звіти в Access?