МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ

до вивчення курсу “Програмні засоби проектування ЕМС» (частина 1.2), виконання лабораторних робіт для студентів спеціальності 141 "Електроенергетика, електротехніка та електромеханіка" (освітні програми «Електричні і електронні апарати» та «Електромеханічне обладнання енергоємних виробництв») всіх форм навчання

2018

Методичні вказівки до вивчення курсу «Програмні засоби проектування ЕМС»(частина 1.2), виконання лабораторних робіт для студентів спеціальності 141 "Електроенергетика, електротехніка та електромеханіка" (освітні програми «Електричні і електронні апарати» та «Електромеханічне обладнання енергоємних виробництв») всіх форм навчання. /Укл.: Л.О.Бондаренко, Л.С.Скрупська. – Запоріжжя: ЗНТУ, 2018. с. 50.

Укладачі: Л.О.Бондаренко
Л.С.Скрупська

Рецензент: М.О.Поляков

Відповідальний за випуск: П.Д.Андрієнко.

Затверджено
на засіданні кафедри
“Електричні та електронні апарати”
Протокол № 11 від 1 червня 2018 р.

Рекомендовано до видання НМК
Електротехнічного факультету
протокол № 11 від 21 червня 2018р.

ЗМІСТ

1 **Лабораторна робота №1.** Створення бази даних з використанням імпорту об'єктів та даних в СУБД MSAccess 5
1.1 Мета роботи 5
1.2 Завдання на лабораторну 5
1.3 Основні відомості 6
1.3.1 Імпортування об'єктів 6
1.3.2 Імпортування даних з використанням запитів 8
1.4 Порядок виконання лабораторної роботи 8
1.5 Зміст звіту 12
2 **Лабораторна робота №2.** Створення запитів на вибірку мовою SQL 13
2.1 Мета роботи 13
2.2 Основні відомості та приклади створення запитів 13
2.3 Порядок виконання лабораторної роботи 22
2.4 Зміст звіту 23
2.5 Контрольні питання
3 Лабораторна робота №3. Розробка об'єктів інтерфейсу форм для введення, перегляду, коригуванню та пошуку інформації в БД «Трансформатори струму» 25
3.1 Мета роботі 25
3.2 Основні відомості 25
3.2..1 Етапи розробки інтерфейсу 25
3.3.2 Визначення послідовності завантаження таблиць з документів 26
3.3 Приклад побудови основної форми з двома підпорядкованими формами 28
3.4 Завдання на самостійну роботу 32
3.5 Зміст звіту 32
3.6 Контрольні питання 33
4 **Лабораторна робота №4.** Створення додатку БД «Трансформатори струму» 34
4.1 Мета роботи 34
4.2 Основні відомості 34
4.2.1 Створення та використання макросів 35

4.2.2 Кнопкові форми 42
4.2.3 Налаштування параметрів автозапуску 47
4.3 Порядок виконання лабораторної роботи 48
4.4 Зміст звіту 49
4.5 Контрольні питання 49
Література 50

1 ЛАБОРАТОРНА РОБОТА №1. СТВОРЕННЯ БД З ВИКОРИСТАННЯМ ІМПОРТУ ОБ'ЄКТІВ ТА ДАНИХ В СУБД ACCESS

1.1 Мета роботи

Вивчення можливостей створення БД з використанням інформації та об'єктів іншої бази даних.

1.2 Завдання на лабораторну роботу

Потрібно створити нову базу даних **ПараметрыТрансформаторовТока**, яка **буде** містити параметри сухих трансформаторів, а також граничні значення похибок вимірювальних трансформаторів залежно від класу точності та складатися з наступних 6-и таблиць: **ТипиТрансформаторов, ТипоисполнениеТрансформаторов, ПараметрыОтветвлений-Трансформаторов, МагнитныеХарактеристики, КлассТочности, ПараметрыКлас-совТочности.** Ці таблиці будемо створювати шляхом імпортування їх з іншої (корпоративної) БД**.**

Як корпоративну БД використовуйте БД **Трансформатор** (файл **Трансформатор.accdb,** розташування файлу за вказівкою викладача). Запропонована база має нступну схему даних (рис.1.1):

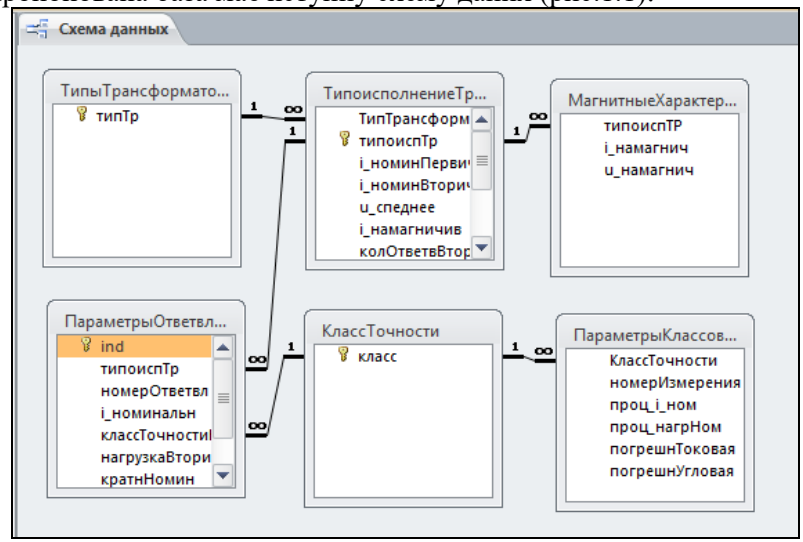


Рисунок 1.1 – Схема даних БД Трансформатор

Увага!

1) Таблиці **ТипиТрансформаторов, КлассыТочности, ПараметрыКлассовТочности** імпортуємо разом з усіма даними, збереженими в цих таблицях.

2) Дані про типовиконання трансформаторів та їх параметри, що збережені в таблицях **ТипоисполнениеТрансформаторов, ПараметрыОтветвлений-Трансформаторов, МагнитныеХарактеристики,** треба відібрати за варіантом завдання. Варіанти завдань вказані в полі **вариантСтудЗадания** таблиці **ТипоисполнениеТрансформаторов.**

1.3 Основні відомості

Хоча можна використовувати Microsoft Access як замкнуту систему, однією з основних переваг цього програмного продукту є можливість роботи з найрізноманітнішими даними інших баз, електронних таблиць чи текстових файлів. Поряд з використанням власної бази даних можна імпортувати дані чи зв'язувати дані, що зберігаються в інших базах даних Access, у файлах DBASE, Microsoft FoxPro та в будь-яких базах даних SQL, які підтримують стандарт ODBC (Open Database Connectivity – відкритий доступ до даних). Microsoft Access дає можливість також імпортувати дані у свою БД із використанням механізму запитів.

1.3.1 Імпортування об'єктів

У Microsoft Access розрізняють імпортування даних і зв'язування даних. При імпорті даних створюється їх копія в новій таблиці поточної БД. Вихідна таблиця чи файл при цьому не змінюються. Зв'язування даних дозволяє читати та у більшості випадків обновляти дані в зовнішньому джерелі даних без їхнього імпорту. Формат зовнішніх даних не змінюється , тому файл можна використовувати в додатку, в якому він був створений, але при цьому з'являється можливість додавати, видаляти чи змінювати дані в поточній БД.

Слід звернути увагу на те, що в Microsoft Access для позначення зв'язаних таблиць і таблиць, що зберігаються в поточній базі даних, використовуються різні значки. Якщо видалити значок зв'язаної таблиці, видаляється зв'язок з нею, але не сама зовнішня таблиця.

Можна імпортувати кожний із шести основних типів об'єктів

(таблиці, запити, форми, звіти, макроси і модулі) з однієї бази даних в іншу двома способами, а саме:

1. Імпортування об'єктів з використанням операцій копіювання і вставки через буфер обміну. Реалізується з використанням кнопок (чи команд) *Копировать* (зовнішньої БД) і *Вставить*(поточної БД).

Примітка. При копіюванні таблиці в діалоговому вікні *Вставка таблицы* пропонується вибрати параметри, що забезпечують або вставку тільки структури таблиці, або вставку структури і даних, або додавання даних в існуючу таблицю. При копіюванні об'єкта копіюються усі його властивості.

2. Імпортування чи пов'язування таблиць із зовнішньої БД MS Access.

Послідовність дій при цьому така:

а) відкрийте створювану вами БД та натисніть на кнопку *Access* в групі *Импорт и связи* на вкладці *Внешние данные* стрічки меню;

б) в діалоговому вікні *Внешние данные* виберіть джерело даних (ім'я файлу зовнішньої БД) та вкажіть дію, яка вам потрібна – імпортування чи пов'язування;

в) у діалоговому вікні *Импорт объектов* виберіть кожну потрібну таблицю;

г) якщо імпортуються тільки структури обраних таблиць (а не дані, що в них містяться), натисніть на кнопку *Параметры* і виберіть *Только структура*.

Примітки:

1. Якщо також потрібно імпортувати зв'язки між таблицями – виставте відповідні прапорці в групі *Импорт*.

При імпорті таблиці, що містить поля підстановки, необхідно імпортувати також таблиці чи запити, на які є посилання в полях підстановок. Якщо цього не зробити, то при відкритті імпортованої таблиці в режимі конструктора буде виведено повідомлення про помилку для кожної відсутньої таблиці чи запиту.

Якщо немає можливості чи бажання імпортувати ці таблиці чи запити:

– відкрийте імпортовану таблицю в режимі конструктора;

– виберіть поле підстановки, що містить посилання на відсутню таблицю чи запит, виберіть вкладинку *Подстановка* і задайте для властивості *Тип елемента управления* значення *Поле*.

2. Коли ви імпортуєте об'єкт доступу, весь об'єкт імпортується як об'єкт з тим самим ім'ям у активну базу даних. Ви не можете імпортувати лише вибрані поля або записи. Якщо в активній базі даних вже є об'єкт з тим самим ім'ям, Access імпортує новий об'єкт з номером, доданим до кінця імені.

1.3.2 Імпортування даних з використанням запитів

Microsoft Access дозволяє імпортувати дані з таблиц однієї БД у нову таблицю іншої БД із використанням *запитів на створення таблиць* (дивись лабораторну роботу по вивченню створення запитів на модифікацію).

Процес створення таблиці за допомогою запиту складається із трьох кроків:

1. Створення запиту на вибірку.

2. Перетворення запиту на вибірку в запит на створення нової таблиці.

3. Виконання запиту на створення.

В результаті отримуємо нову таблицю із записами відібраними запитом на вибірку.

1.4 Порядок виконання лабораторної роботи

Завдання 1. Створіть у власній папці нову БД з ім'ям **ПараметрыТрансформаторовТока.** З розділу 1.3.1 вам відомо, що в MS Access імпортувати можливо як тільки структури таблиць, так й всю таблицю разом з даними. Використаємо цю можливість для імпортування з зовнішньої БД **Трансформатор** (файл **Трансформатор.accdb**) в створену БД 3-и таблиці: **ТипиТрансформаторов, КлассыТочности, ПараметрыКлассов-Точности** разом із збереженою в них інформацією.

Завдання 2. З таблиць **ТипоисполнениеТрансформаторов, Парамет-рыОтветвленийТранс-форматоров, МагнитныеХарактеристики** зовнішньої БД потрібно відібрати тільки записи з параметрами трансформаторів вашого варінту. Для цього використаємо можливість MS Access імпортування даних за допомогою запитів на створення таблиць

Створення таблиці *ТиповиконанняТрансформаторів*

Розглянемо специфіку створення запиту на створення таблиці на прикладі формування в вашій БД таблиці **ТипоисполнениеТранс-**

форматоров, що має містити список тільки тих типовиконань з їх параметрами, що належать , наприклад, до аріанту 5.

1. Перейдіть в корпоративну БД **Трансформаторы** (файл **Трансформаторы.accbd**) Створіть запит **СозданиеТаблТипоисполнениеТрансформаторов.** Для цього:
   а) В режимі конструктора створіть запит на вибірку, вказавши джерелом даних таблицю **ТипоисполнениеТрансформаторов** з умовами відбору типовиконань трансформаторів, що належать до варианту 5 (рис. 1.2).

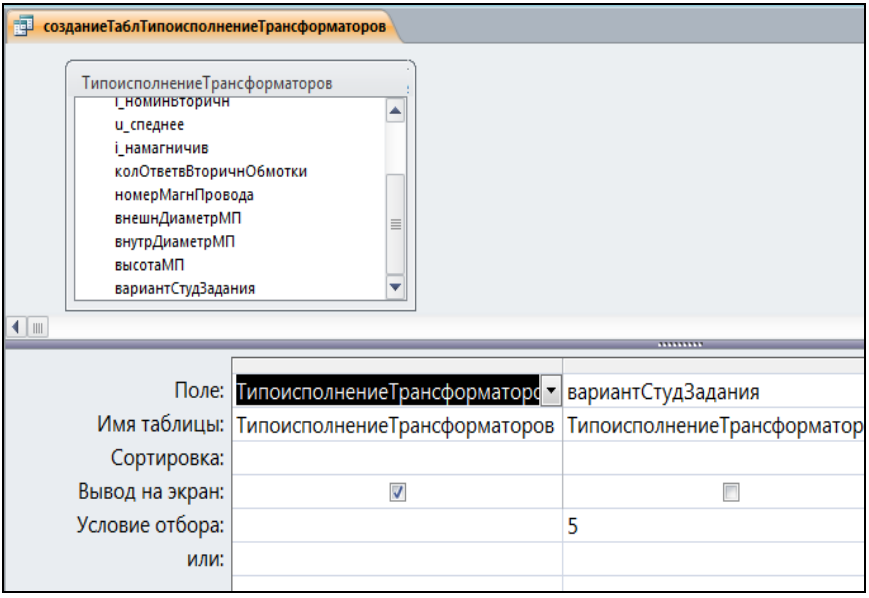


Рисунок 1.2 – Конструктор запиту на створення таблиці ТипоисполнениеТрансформаторов

   б) До того як використовувати відібрані дані для створення таблиці, доцільно впевнитись у правильності їх відбору, переключившись до режиму *Таблиця* або натиснувши кнопку *Запуск* на вкладці *Конструктор* у групі *Результати*.
   в) Перетворіть створений запит на вибірку в запит на створювання таблиці, натискаючи на вкладці *Конструктор* стрічки меню у групі *Тип запроса* кнопку *Создание таблицы* або обираючи в контекстному меню вільної області джерела даних у підменю *Тип запроса* пункт *Создание таблицы*. У вікні *Создание*

тадлицы (рис. 1.3), що з'явиться на екрані, вкажіть назву створюваної таблиці – **ТипоисполнениеТрансформаторов**, встановіть перемикач на *Другая база даних* та в полі *Имя файла* – введіть назву вашої БД **ПараметрыТрансформаторовТока** (файл **ПараметрыТрансформаторовТока.accbd**).

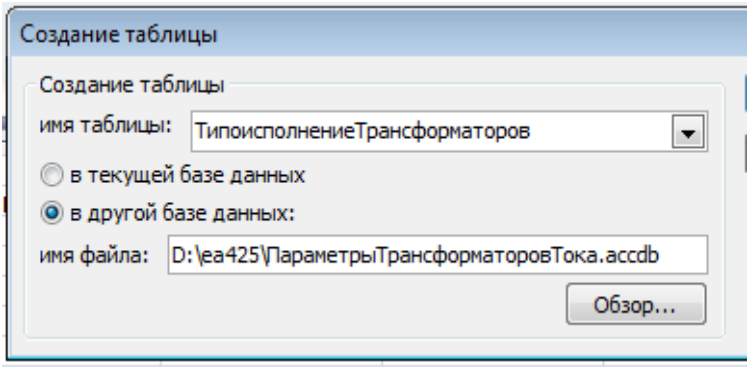


Рисунок 1.3 – Вікно Создание таблицы

2. Запустіть створений запит на виконання – на вкладці *Работа с запросами/Конструктор* у групі *Результаты* натисніть на кнопку *Выполнить*. З'явиться вікно з повідомленням (рис.1.4) про кількість відібраних записів, що будуть розміщені в новій таблиці **ТипоисполнениеТрансформаторов** в БД **ПараметрыТрансформаторовТока**.

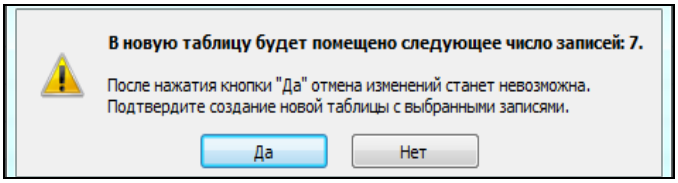


Рисунок 1.4 – Вікно повідомлення про кількість запитів, що будуть розміщені в новій таблиці.

Створення таблиці МагнитныеХарактеристики

1. Перейдіть в корпоративну БД **Трансформаторы** (файл **Трансформаторы.accbd**) Створіть запит ***СозданиеТаблМагнитныеХарактеристики*.** Для цього:
   В режимі конструктора створіть запит на вибірку, вказавши джерелом даних таблиці **ТипоисполнениеТрансформаторов і МагнитныеХарактеристики** з умовами відбору типовиконань

трансформаторів, що належать до вашого варианту (рис. 1.5).

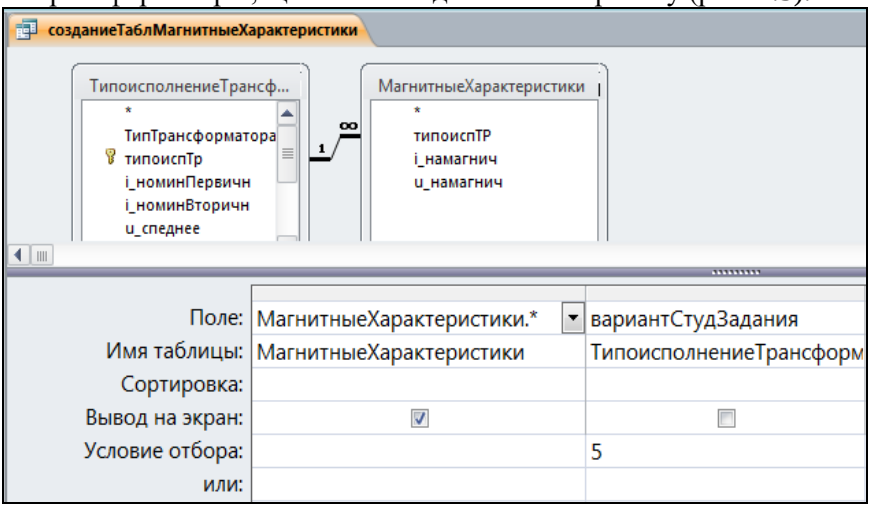


Рисунок 1.5 – Конструктор запиту на створення таблиці МагнитныеХарактеристики з записами, що відібрані по варіанту 5 (поле вариантСтудЗадания)

2. Перетворіть створений запит на вибірку в запит на створювання таблиці. У вікні *Создание тадлицы*, вкажіть назву створюваної таблиці – **МагнитныеХарактеристики**, встановіть перемикач на *Другая база даних* та в полі *Имя файла* – введіть назву вашої БД **ПараметрыТрансформаторовТока.**
3. Запустіть запит на виконання.

Створення таблиці ПараметрыОтветвленийТрансформаторов

1. Перейдіть в корпоративну БД **Трансформаторы** (файл **Трансформаторы.accbd**) Створіть запит на вибірку **СозданиеТаблПарамОтветвТрансф**, вказавши джерелом даних таблиці **ТипоисполнениеТрансформаторов і ПараметрыОтветвленийТрансформаторов** з умовами відбору типовиконань трансформаторів, що належать до вашого варианту.
2. Перетворіть створений запит на вибірку в запит на створювання таблиці. У вікні *Создание тадлицы*, вкажіть назву створюваної таблиці – **ПараметрыОтветвленийТрансформаторов**, встановіть перемикач на *Другая база даних* та в полі *Имя файла* – введіть назву вашої БД **ПараметрыТрансформаторовТока.**
3. Запустіть запит на виконання.

Завдання 3. Треба пом’ятати, що властивості полів і ключові поля не переходять до даних у новій таблиці яка створена з використанням запиту на створення таблиці. Тому потрібно відкоригувати ключові поля і властивості таблиць **ТипоисполнениеТрансформаторов, ПараметрыОтветвленийТрансформаторов, МагнитныеХаракте-ристики** вашої БД.

Завдання 4. Створіть *Схему даних* БД **ПараметрыТрансформаторовТока**, яка повинна повністю збігатися зі схемою даних БД **Трансформатор.**

1.5 Зміст звіту

Звіт повинен містити:

1. Опис структури створеної БД, розміщення даних за таблицями та встановлені зв’язки між таблицями.
2. Опис способу створення кожної таблиці нової БД.
3. Стислі відповіді на наступні запитання:
   а) Механізми імпортування та експортування даних в MS Access?
   б) Різниця між імпортуванням даних і зв’язуванням даних?
4. Описати алгоритм створення запитів на створення таблиць.

1.6 Контрольні питання

1. Структура таблиці;
2. Типи даних;
3. Введення даних;
3. Ключове поле;
4. Встановлення зв’язків між таблицями;
5. Способи створення таблиць;
6. Робота в «Режимі Таблиці»;
7. Сортування та фільтрація даних в таблицях;
8. Робота в «Режимі Конструктора»;
9. Створення зв’язків між таблицями та робота з ними.

2 ЛАБОРАТОРНА РОБОТА №2. СТВОРЕННЯ ЗАПИТІВ НА ВИБІРКУ МОВОЮ SQL

2.1 Мета роботи

Освоєння принципів створення запитів на вибірку на мові SQL – Structured Query Language.

2.2 Основні відомості та приклади створення запитів

Будь-який запит у Microsoft Access реалізується за допомогою мови SQL. Більшість запитів ви можете побудувати, користуючись можливостями режиму конструктора, але й у цьому випадку вони будуть зберігатися у вигляді інструкцій SQL. Для створення так званих підлеглих запитів, результати яких використовуються як умови порівняння в інших запитах, необхідно знати мову SQL. До того ж не всі типи запитів можна побудувати в режимі конструктора. У таких випадках доведеться використовувати SQL.

Надалі, описуючи синтаксис запитів і елементів виразів, в < > будемо позначати місця для ідентифікаторів та виразів користувача, в [ ] – необов'язкові елементи, а через | будемо перераховувати елементи, серед яких користувач має обрати лише один.

Інструкція **SELECT** є ядром мови SQL. Вона використовується для відбору рядків і стовпчиків з таблиць бази даних та містить п'ять основних операторів. У загальному випадку її синтаксис можна подати в такому вигляді:

SELECT [ALL | DISTINCT | DISTINCTROW | TOP n] список полів
FROM список таблиць
[WHERE умови відборів записів**]**
[**GROUP BY** ім'я поля, ...]
[**HAVING** умови відборів груп]
[**ORDER BY** ім'я поля [**ASC | DESC**],...];

Бачимо, що обов'язковими реченнями (операторами) в наведеній інструкції відбору є лише SELECT та FROM. Окремі речення, як правило, набираються з нового рядка, хоча й можуть вводитися через пробіл після попереднього речення.

Оператор **SELECT**

Після слова **SELECT** перераховуються лише ті поля, що виводяться при виконанні запиту (для них в режимі конструктора встановлюється прапорець *Вывод на экран*). Якщо формується вираз обчислювального поля, то його ім'я вказується після формули виразу і відділяється від нього сполученням **AS.**

В режимі SQL ім'я таблиці вказується перед іменем її поля і відділяється від нього крапкою. Першу частину імені (включаючи точку) можна опустити, якщо поле є тільки в одній з таблиць пропозиції **FROM**.

Імена, що містять пробіли, обов'язково повинні полягати в квадратні дужки.

При визначенні списку полів використання символу "*" замість імені поля вказує, що потрібно відобразити всі стовпці даної таблиці. Якщо в якості списку полів використаний символ "*", то відбираються всі стовпці всіх таблиць, зазначених у пропозиції **FROM**.

Оператор **FROM**

Таблиці, запити, зв'язки між ними, які формують джерело даних і в режимі конструктора відображаються в його верхній частині, в режимі SQL описуються після **FROM**. Якщо джерело містить тільки одну таблицю чи запит, то після **FROM** вказується лише його назва, якщо ж більше – то крім назв базових об'єктів ще й описуються зв'язки між ними:

– INNER JOIN (внутрішнє об**'**єднання),

– LEFT JOIN (ліве зовнішнє об**'**єднання),

– RIGHT JOIN (праве зовнішнє об**'**єднання).

Приклад 1. Наступний запит відображає всі поля таблиці *ТипоисполнениеТрансформаторов* а також обчислювальне поле для площі поперечного перетину магнітопровода по формулі

$\pi/4 \cdot$(внешнийДиаметрМП-внутреннийДиаметрМП)

на мові SQL:

```
SELECT ТипоисполнениеТрансформаторов.*,
            3.14/4*([внешнДиаметрМП-внутрДиаметрМП]) AS S
FROM ТипоисполнениеТрансформаторов;
```

Приклад 2. Наступний запит відображає список тільки тих

типовиконань трансформаторів (поле **типоиспТР** таблиці **ТипоисполнениеТрансформаторов**), для яких у таблиці **МагнитныеХарактеристики** відсутня інформація. Запит наведений на мові SQL та в режимі конструктора (рис.2.1):

```
SELECT ТипоисполнениеТрансформаторов.типоиспТр
FROM ТипоисполнениеТрансформаторов LEFT JOIN
        МагнитныеХарактеристики ON
         ТипоисполнениеТрансформаторов.типоиспТр =
        МагнитныеХарактеристики.типоиспТР
WHERE (((МагнитныеХарактеристики.типоиспТР) Is Null));
```

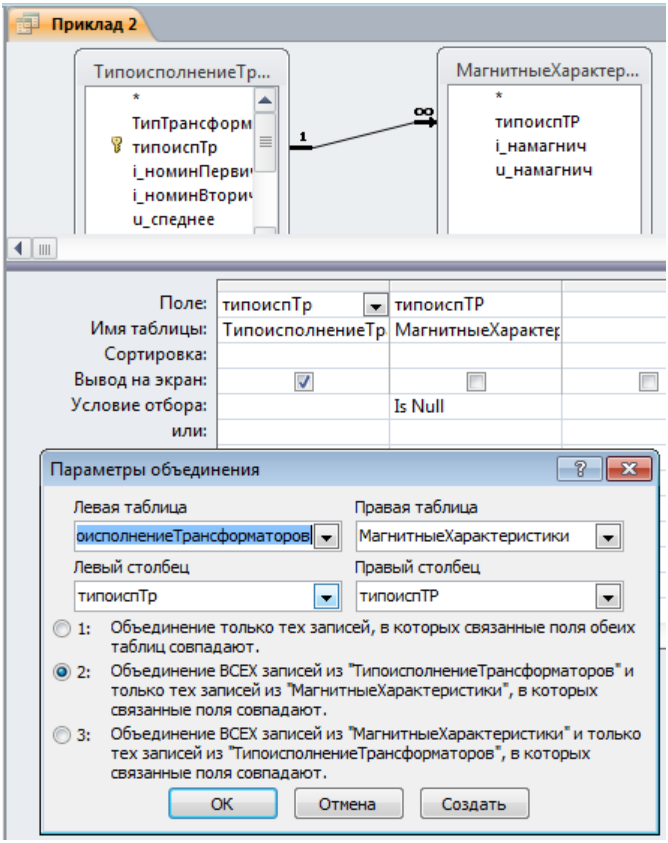


Рисунок 2.1 – Запит прикладу 2 в режимі конструктора

Оператор **WHERE**

Окремі умови відбору записів, які в режимі конструктора записуються в рядку *УСЛОВИЕ ОТБОРА* та рядках *ИЛИ* і можуть міститися у різних стовпцях, в режимі SQL послідовно вказуються після WHERE та поєднуються між собою логічними операндами AND чи OR.

Оператор **ORDER BY**

Як і в режимі конструктора (рядок *Сортировка*), найважливішим полем сортування в режимі SQL є перше поле, яке зазначається відразу після ORDER BY, адже саме за ним впорядковуються записи. Лише ті записи, в яких значення першого поля сортування однакові, між собою впорядковуються за другим полем сортування і т.д.

Для сортування за зростанням після назви поля вказується **ASC,** за спаданням – DESC. За замовчуванням сортування виконується за зростанням.

Приклад 3. Наступний запит відображає всі поля таблиці *ТипоисполнениеТрансформаторов*. Записи потрібно відсортувати за зростанням по полю **типТр**, а для кожного типу трансформатора відсортувати відсортувати за спаданням по полю **типоиспТр**.

Запит наведений на мові SQL:

```
SELECT ТипоисполнениеТрансформаторов.*
FROM ТипоисполнениеТрансформаторов
ORDER BY типТр asc , типоиспТр desc;
```

Приклад 4.

Для зазначеного типу трансформатора відібрати типовиконання трансформаторів у яких значення середньої напруги лежить в інтервалі від 100 до 500 і номінальний струм більше 1 або значення середньої напруги більше 500 і номінальний струм дорівнює 1,26. Відібрану інформації відсортуємо по полю [u_среднее] за зростанням. Запит наведений на мові SQL та в режимі конструктора (рис.2.2:

```
SELECT типоиспТр, u_среднее, i_намагничив
FROM ТипыТрансформаторов INNER JOIN
```

```
        ТипоисполнениеТрансформаторов ON
        ТипыТрансформаторов.типТр =
        ТипоисполнениеТрансформаторов.ТипТрансформатора
WHERE ((типТр="ТТВ 110-І") AND (u_среднее Between 100 And 500)
        AND (i_намагничив<1))
        OR ((типТр="ТТВ 110-І") AND (u_среднее>500) AND
        (i_намагничив=0.126))
ORDER BY u_среднее;
```

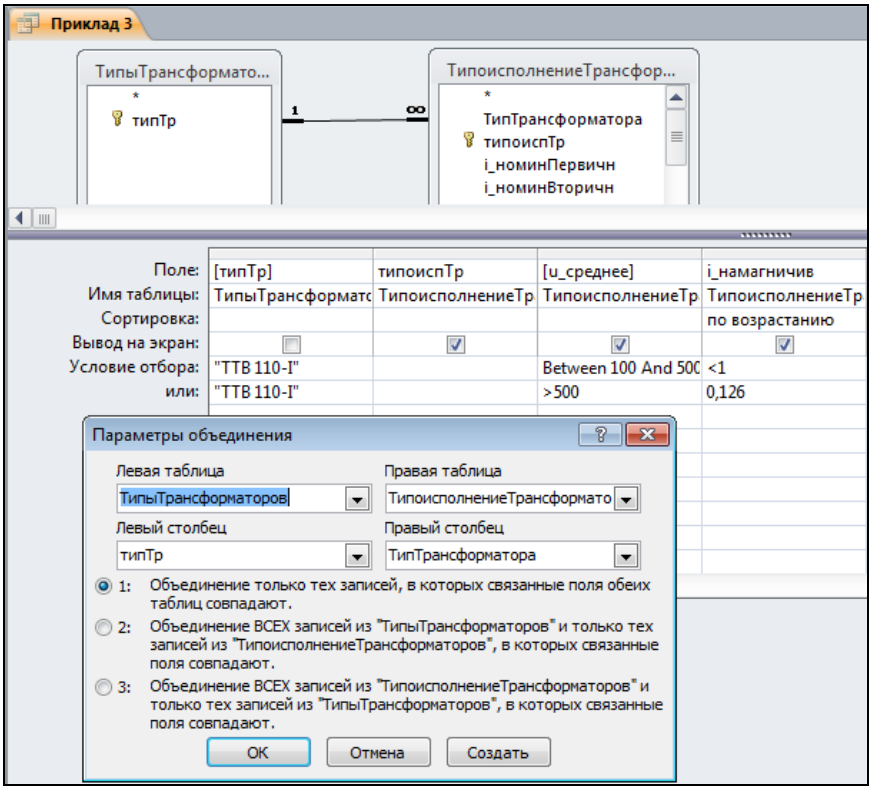


Рисунок 2.2 - Запит прикладу 4 в режимі конструктора

Оператор **GROUP BY** задає групування записів у підсумковому запиті, а оператор **HAVING** використовується для відбору тих груп, що мають бути включені в підсумковий набір записів.

Як вже зазначалося в лабораторній роботі №2, для забезпечення групування відібраних записіві виконання обчислень у групах в

режимі конструктора необхідно натиснути на кнопку *Итоги.* Після цього, в бланку запиту з'явиться додатковий рядок *Групповая операция*, в якому для кожного поля можна обрати одне з значень:

Группировка - використовується для групування відібраних записів. Якщо це значення встановлюється для декількох полів, то групи створюються згідно унікальних комбінацій їх значень. Коли ж це значення не встановлено для жодного поля, то всі записи джерела буде віднесено до однієї групи. Поля, для яких в режимі конструктора обрана *Группировка*, в режимі SQL послідовно перераховуються в реченні GROUP BY. Решта з перерахованих нижче значень списку, крім 2-х останніх, це статичні функції SQL, що застосовуються до значення поля.

SUM – обчислює суму значень поля в кожній групі.
COUNT – обчислює кількість непорожніх значень поля в кожній групі.
MIN – обчислює мінімальне значення в кожній групі.
MAX – обчислює максимальне значення в кожній групі.
AVG – обчислює середнє значення в кожній групі.
VAR – обчислює дисперсію значення в кожній групі.
FIRST – повертає перше значення в кожній групі.
LAST – повертає останнє значення в кожній групі
Условие – містить умову відбору окремих записів в групу і тому поле, на яке назначена умова, не виводиться на екран.
Выражение – назначається на обчислювальне поле.

Приклад 5. Наступний запит підраховує для кожного типовиконання трансформаторів (поле **типоиспТР** таблиці **ТипоисполнениеТрансформаторов**) кількість відповідних записів в таблиці **МагнитныеХарактеристики**, середнє значення струму намагнічування, 10 відсотків від середнього значення струму намагнічування. Прі цьому нас цікавлять тільки ті типовиконання, у яких номінальний вторинний струм дорівнює 1. Запит наведений в режимі конструктора (рис.2.3) та на мові SQL:

```
SELECT ТипоисполнениеТрансформаторов.типоиспТр,
Count(МагнитныеХарактеристики.типоиспТР) AS [Count-типоиспТР],
Avg(i_намагнич) AS [Avg-i_намагнич], Avg([i_намагнич])*0.1 AS
ТокаНамагн10Проц
FROM ТипоисполнениеТрансформаторов LEFT JOIN
МагнитныеХарактеристики
```

```
ON ТипоисполнениеТрансформаторов.типоиспТр =
МагнитныеХарактеристики.типоиспТР
WHERE (((ТипоисполнениеТрансформаторов.i_номинВторичн)=1))
GROUP BY ТипоисполнениеТрансформаторов.типоиспТр;
```

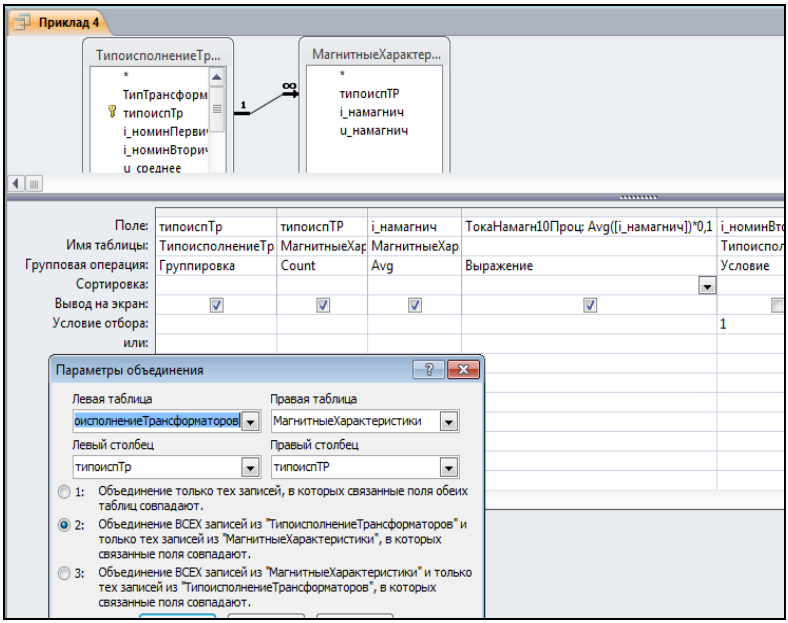


Рисунок 2.3 - Запит прикладу 5 в режимі конструктора

Приклад 6. Створимо, так званий, параметричний запит. В одному запиті можливо вказувати стільки параметрів, скільки потрібно. Для забезпечення коректності введених параметрів потрібно задати для них тип даних. Для цього відкрийте вікно *Параметры запроса* за допомогою команди *Параметры…* контекстного меню вільного поля області джерела даних запиту, заповніть виведену таблицю. В наслідок цього, інструкція SELECT доповнюється оператором PARAMETERS, який записується в першому рядку, тобто до оператора SELECT.

В наведеному прикладі, запит підраховує тільки для одного типовиконання трансформаторів (поле **типоиспТР** таблиці **ТипоисполнениеТрансформаторов**) кількість відповідних записів в таблиці **МагнитныеХарактеристики**. У режимі конструктора назву типовиконання будемо призначати у вигляді параметра в рядку

Условие отбора на поле з установкою *Группировка* (поле **типоиспТР)**. У режимі SQL умови для відбору груп записуються після оператора HAVING. Запит наведений в режимі конструктора (рис.2.4) та на мові SQL:

```
PARAMETERS [Ведите название типоисполнения] Text ( 255 );
SELECT ТипоисполнениеТрансформаторов.типоиспТр,
Count(МагнитныеХарактеристики.типоиспТР) AS [Count-типоиспТР]
FROM ТипоисполнениеТрансформаторов INNER JOIN
        МагнитныеХарактеристики ON
        ТипоисполнениеТрансформаторов.типоиспТр =
        МагнитныеХарактеристики.типоиспТР
GROUP BY ТипоисполнениеТрансформаторов.типоиспТр
HAVING (((ТипоисполнениеТрансформаторов.типоиспТр)=
        [Ведите название типоисполнения]));
```

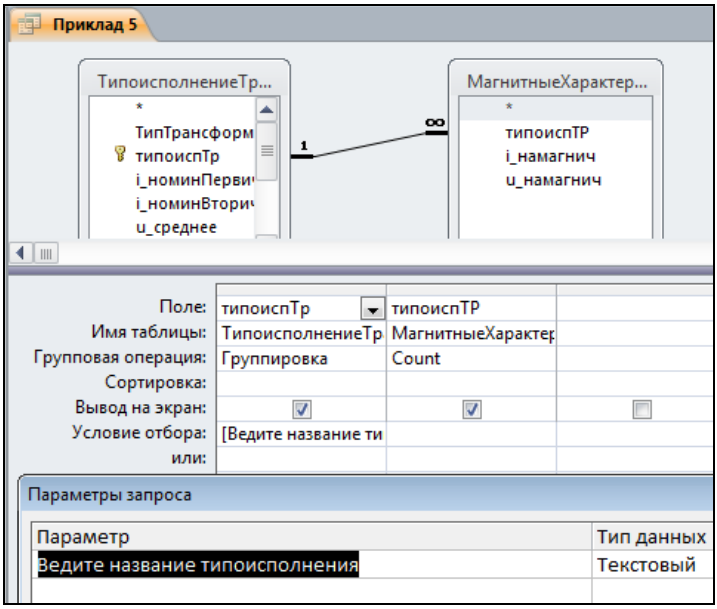
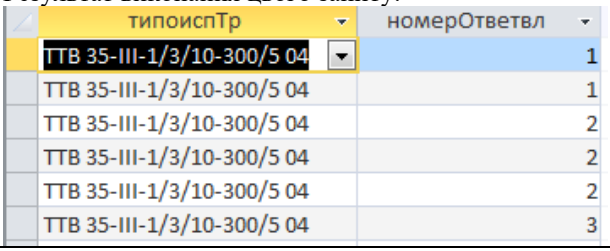


Рисунок 2.4 - Запит прикладу 6 в режимі конструктора

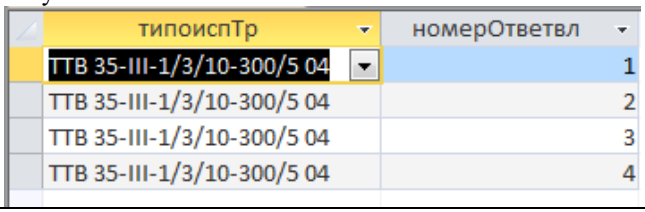
Приклад 7. В таблиці 2.1 наведено опис призначення та приклади використання можливих значення предикату: **[ALL | DISTINCT | DISTINCTROW | TOP n].**

Таблиця 2.1 - Можливі значення предикату

Значення	**Опис**
ALL	Передбачено, якщо ні один із предикатів не включено. Обробник баз даних Microsoft Access обирає всі записи, які відповідають умовам. Наведені два приклади є еквівалентними та повертають всі записи з таблиці. SELECT ALL * FROM ПараметрыТипоисполнений ORDER BY колОтвВторичнОбмотки; SELECT * FROM ПараметрыТипоисполнений ORDER BY колОтвВторичнОбмотки;
DISTINCT	Пропускає записи, які містять повторювані дані у полях, перекислених в операторі SELECT. Щоб включити їх у результати запитів, значення кожного поля має бути унікальним. Наприклад, поле номерОтветвл в таблиці **ПараметрыОтветвлений-Трансформаторов** може мати однакові значення. Створюємо запит з предікатом ALL: SELECT типоиспТр, номерОтветвл FROM ПараметрыОтветвленийТрансформаторов WHERE (((типоиспТр)="ТТВ 35-III-1/3/10-300/5 04")); Результат виконання цього запиту: 

типоиспТр	номерОтветвл
ТТВ 35-III-1/3/10-300/5 04	1
ТТВ 35-III-1/3/10-300/5 04	1
ТТВ 35-III-1/3/10-300/5 04	2
ТТВ 35-III-1/3/10-300/5 04	2
ТТВ 35-III-1/3/10-300/5 04	2
ТТВ 35-III-1/3/10-300/5 04	3

Продовження таблиці 2.1

	Цей же запит з предікатом DISTINCT: SELECT DISTINCT типоиспТр, номерОтветвл FROM ПараметрыОтветвленийТрансформаторов WHERE (((типоиспТр)="ТТВ 35-III-1/3/10-300/5 04")); Результат його виконання: 
DISTINCTROW	Пропускає дані повністю повторюваних записів, а не лише повторюваних полів.
TOP n	Повертає n записів, які з'являються серед перших або останніх записів діапазону, зазначеного в реченні ORDER BY.

2.3 Порядок виконання лабораторної роботи

Для бази даних, яка містить параметри трансформаторів струму та була створена в лабораторній роботі №1 виконайте наступні завдання.

Завдання 1, 2, 3. Створіть кілька запитів, що вибирають дані з однієї таблиці за кількома різноманітними умовами. В одному з запитів передбачте створення полів, що обчислюються. Створіть параметричний запит. Для кожного зі створених запитів перепишіть програму мовою SQL і поясніть її.

Наприклад, такі запити:

- відібрати типовиконання з відповідними параметрами магнітопроводу;
- параметричний запит того ж призначення (за типовиконанням);
- додати поле, що обчислюється, (наприклад, обчислити площу поперечного перерізу магнітопроводу).

Завдання 4. Створіть підсумковий запит для однієї з таблиць. Наприклад, запит для таблиці, у якій наведені граничні значення

похибок вимірювальних трансформаторів залежно від класу точності (таблиця **ПараметрыКлассовТочности**). При створені цього запиту використайте всі три установки (*Группировка, Условие, Выражение*) та декілька агрегатних функцій. Виконайте. Перевірте правильність отриманої інформації. Перепишіть запит на мови SQL.

Завдання 5, 6. Створіть два запити в режимі SQL, які протестують підлеглі таблиці **МагнитныеХарактеристики** і **ПараметрыОтветвленийТрансформаторов.** Ці запит потрібні для виводу списку тільки тих типовиконань трансформаторів (поле **типоиспТР** головної таблиці **ТипоисполнениеТрансформаторов**), для яких у підлеглих таблицях відсутня інформація.

Завдання 7. У таблиці **КлассТочности** наведений повний перелік існуючих класів точності. В таблиці **ПараметрыОтветвленийТрансформаторов** в полі **классТочности** для кожного відгалуження типовиконання трансформатора вказані ті класи, які використовуються при проведенні випробувань Створіть запит в режимах конструктора та SQL, що виводить тільки ті класи точності, які не використовуються для конкретного типовиконання.

Завдання 8. Створіть запит, що виводить граничні значення похибок вимірювальних трансформаторів (поля **погрешнУгловая** і **погрешнТоковая** таблиці **ПараметрыКлассовТочности**) залежно від використаного класу точності кожного відгалуження для зазначеного у вигляді параметра типовиконання.

2.4 Зміст звіту

Для кожного завдання вивести: зміст запиту, запит в режимі конструктора, запит мовою SQL, результат виконання запитів.

2.5 Контрольні питання

1. Що таке запит? Робота з даними, відібраними запитом в режимі перегляду.

2. Мова SQL. Інструкція **SELECT**, оператори інструкції, формат запису кожного оператора, призначення кожного оператора.

3. Перерахуйте всі засоби задавання умов відбору в запитах. Функції Between, In(), Like(), Iif(). Завдання умов відбору для дат і часу, функції DateDiff, Date, Day, Month, Year.

4. Створення полів, що обчислюються.

5. “*Построитель выражений*” і його застосування.

6. Параметричний запит: створення і використання.

7. Поняття підсумкового запиту і його створення. Які групові операції використовуються при створенні підсумкових запитів. Умови в групових операціях.

8. Багатотабличні запити. У яких випадках у багатотабличних запитах можна змінювати дані? Типи об’єднання таблиць, що беруть участь у запиті та їх вплив на зміст запиту.

3 ЛАБОРАТОРНА РОБОТА №3 РОЗРОБКА ОБ'ЄКТІВ ІНТЕРФЕЙСУ ФОРМ ДЛЯ ВВЕДЕННЯ, ПЕРЕГЛЯДУ, КОРИГУВАННЮ ТА ПОШУКУ ІНФОРМАЦІЇ В БД «ТРАНСФОРМАТОРИ СТРУМУ»

3.1 Мета роботи

Розширити набути вміння та навички створення форм - головного об'єкту інтерфейсу баз даних.

3.2 Основні відомості

Діалогові програми користувача баз даних, перш за все, повинні забезпечувати зручний графічний інтерфейс для роботи з документами предметної області. Основним засобом створення такого інтерфейсу є форми. Форми, адекватні формам первинних документів, дозволяють виконувати завантаження даних, в будь-який момент переглядати і редагувати вміст раніше введених в базу даних документів. Крім того, може бути передбачена роздруківка підготовленого документа засобами звітів.

Інструментарій розробки форм надає широкі можливості по створенню графічного діалогового інтерфейсу користувача для роботи з документами, що зберігаються в базі даних. Такий інтерфейс є основою роботи з базою даних в практичному застосуванні користувача. Після остаточного створення додатка користувач, як правило, не працює безпосередньо з таблицями бази даних. Розробник програми часто обмежує повністю або частково безпосередній доступ користувача до таблиць.

3.2.1 Етапи розробки інтерфейсу

Перш ніж вводити, відображати або коригувати дані таблиць через екранну форму, треба її спроектувати і сконструювати. Далі розглядаються основи проектування форм для побудови зручного інтерфейсу користувача для роботи з документами. Детально описана технологія розробки форми, забезпечує початкове введення, перегляд та оновлення документів в базі даних.

У процесі розробки технології завантаження бази даних і проектування форм доцільно визначити:

- перелік документів-джерел, що зберігаються в базі і містять

необхідні дані для завантаження таблиць бази даних;

- таблиці - об'єкти завантаження для кожного документа-джерела;
- зміст і послідовність завантаження. При цьому необхідно враховувати, що для забезпечення зв'язковий цілісності головні таблиці повинні бути завантажуватися раніше підлеглих;
- підсхему даних кожної форми (фрагмент схеми даних), що складається з таблиць, необхідних для створення електронного документа. При цьому для багатотабличній форми вибирається:
  - таблиця, яка буде базовим джерелом записів головної форми, і таблиці для відображення довідкових даних в цій частині форми;
  - таблиця, яка буде джерелом записів підпорядкованої форми, включеній в головну форму, і таблиці для відображення довідкових даних в підлеглій формі;
- макет форми, т. е. її загальну структуру, відповідну структурі документа-джерела і отриманої підсхеми даних. При цьому розподіляється простіл форми для розміщення підлеглих форм;
- склад і розміщення елементів, пов'язаних з полями таблиць, і написів для кожної з частин складовою форми. При цьому:
  - в головну форму обов'язково треба вводити ключові поля таблиці-джерела даних (наприклад, поле **типоиспТР** таблиці **ТипоисполнениеТрансформаторов**);
  - в підлеглій формі передбачити тільки ті ключові поля таблиці - базового джерела підпорядкованої форми, яких немає в таблиці-джерелі головної форми .

Після виконання перерахованих пунктів і отримання макета форми можна приступити до розробки форм засобами Access.

3.2.2 Визначення послідовності завантаження таблиць з документів

При розробці форм, що забезпечують завантаження взаємопов'язаних таблиць бази даних, слід мати на увазі вимоги до послідовності завантаження записів в таблиці відповідно до схеми даних і встановленими параметрами підтримання цілісності. Ці вимоги можна сформулювати наступним образом:

- незалежно можуть створюватися записи таблиць, які не підпорядковані будь-яким іншим таблицям в схемі даних;

- запис таблиці, підпорядкованої будь-яким іншим таблицям, може створюватися при наявності пов'язаних з нею записів в головних таблицях. Записи головною таблиці повинні бути завантажені раніше (таблиці довідкових даних) або повинні створюватися разом з підлеглою записом в одній формі.

Розглянемо технологію завантаження на прикладі бази даних **ПараметрыТрансформаторовТока**. Таблиці бази даних і зв'язки між ними відображені в схемі даних, наведеної на рис. 3.1.

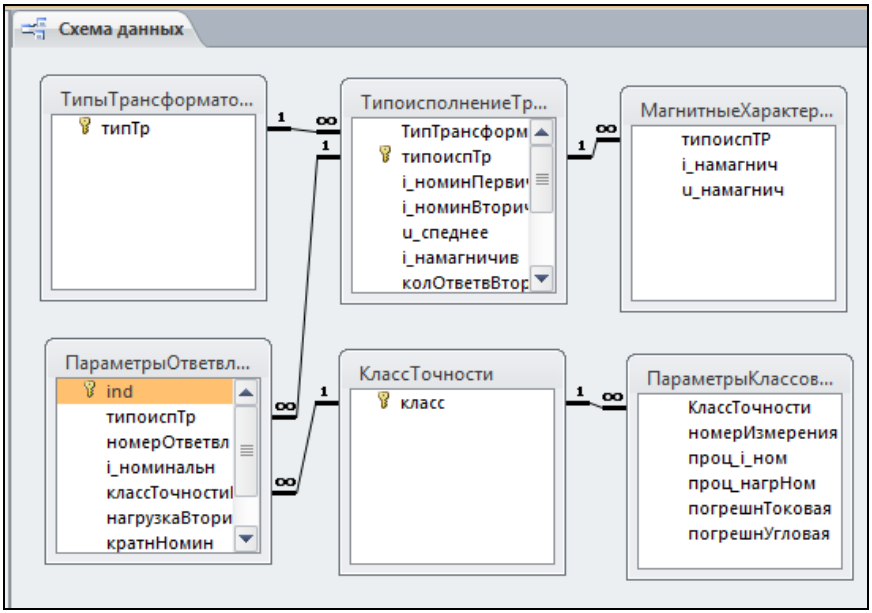


Рисунок 3.1 - Схема даних БД ПараметрыТрансформаторовТока

Таблиці довідкової інформації **ТипиТрансформаторов, Класс-Точности** на схемі даних знаходяться на верхньому рівні і не підпорядковані іншим таблицям, тому їх завантаження відбувається в будь-якій послідовності.

Завантаження таблиці **ТипоисполнениеТрансформаторов** може проводитися після завантаження таблиці **ТипиТрансформаторов**, так як таблиця **ТипоисполнениеТрансформаторов** в схемі даних підпорядкована таблиці **ТипиТрансформаторов**.

Завантаження таблиць **ПараметрыОтветвленийТрансформаторов, Магнитные-Характеристики** може проводитися тільки після

завантаження таблиці **ТипоисполнениеТрансформаторов**, тому що ці. таблиці підпорядковані таблиці **ТипоисполнениеТрансформаторов.** Або загрузка записів цих таблиць проводиться одночасно, що забезпечить формування взаємозв'язків записів цих таблиць. При цьому забезпечується одноразове введення значень типовиконання трансформатору (поле **типоиспТР**) для всіх пов’язаних записів.

Завантаження таблиці **ПараметрыКлассовТочности** може проводитися після завантаження таблиці **КлассТочности**, так як таблиця **ПараметрыКлассовТочности** в схемі даних підпорядкована таблиці **КлассТочности**, або загрузка записів цих таблиць проводиться одночасно, що забезпечить формування взаємозв'язків записів цих таблиць.

3.3 Приклад побудови основної форми з двома підпорядкованими формами

Розглянемо процес проектування форми **ПарамТипоисполнений** для, введення, перегляду і коригування даних про параметри типовиконання трансформаторів току, тобто даних, що зберігаються в таблицях **ТипоисполнениеТрансформаторов, ПараметрыОтветвленийТрансформаторов, МагнитныеХарактеристики**:.

При проектуванні форми визначається підсхема даних, що включає об'єкти завантаження форми, загальна структура форми - проект макета і розміщення реквізитів відповідно до підсхеми даних, враховуються особливості призначення та роботи з формою.

Вибір підсхеми даних для побудови форми визначається такими міркуваннями:

1. Так як форма забезпечує завантаження даних трьох таблиць (таблиці -**ТипоисполнениеТрансформаторов** та, пов'язаних ставленням 1:М з нею, таблиць **ПараметрыОтветвлений-Трансформаторов і МагнитныеХарактеристики)**, то головна таблиця **ТипоисполнениеТрансформаторов** повинна бути джерелом записів основної форми, підпорядкована **Параметры-ОтветвленийТрансформаторов** - джерелом записів першої підпорядкованої форми, підпорядкована **МагнитныеХарактеристики** - джерелом записів другої підпорядкованої форми.
2. Для відображення довідкових даних в основній формі повинна використовуватися таблиця **ТипыТрансформаторов**.

Таким чином, підсхема даних для форми введення / виведення

параметрів типовиконань трансформаторів повинна мати вигляд, показаний на рис. 3.2.

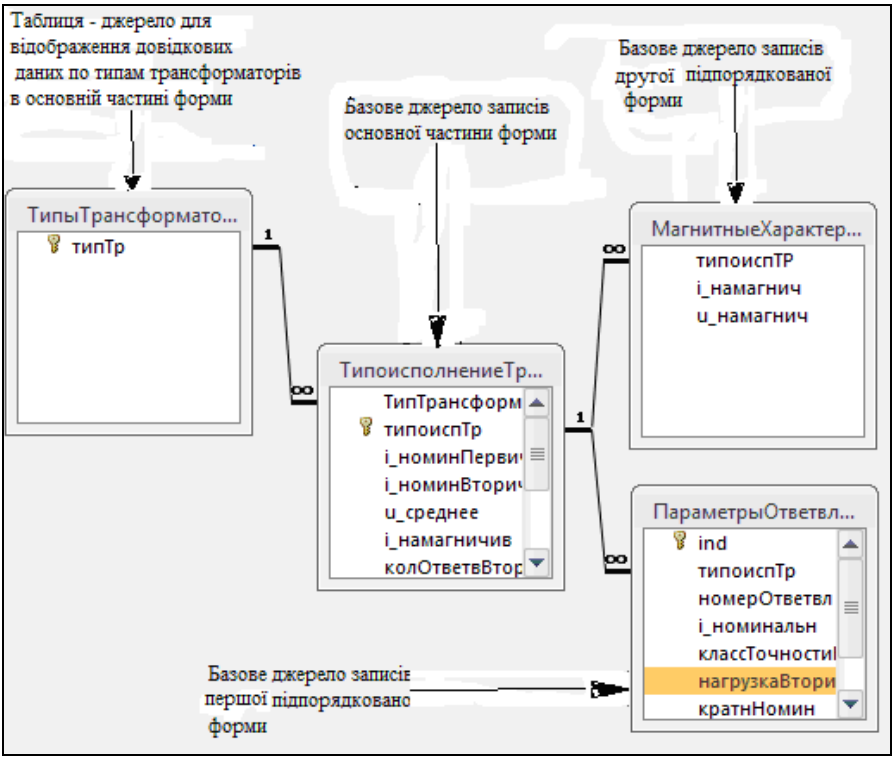


Рисунок 3.2 - Підсхема даних для форми введення/виведення параметрів типовиконань трансформаторів

Відповідно до визначених об'єктів завантаження в підсхемі даних (рис.3.2) багатотаблична форма **ПарамТипоисполнений** (рис.3.3 та рис.3.4). повинна складатися з трьох форм: основній і включених в неї двох підпорядкованих форм

Джерелом записів головної форми стане **ТипоисполнениеТрансформаторов** а таблиця **ТипыТрансформаторов** буде використана для відображення довідкової інформації. Через цю частину багатотабличної форми виконується введення, перегляд і коригування основних параметрів одного типовиконання трансформатору. Число доступних записів визначається кількістю записів в таблиці **ТипоисполнениеТрансформаторов**.

Джерелом записів першої підпорядкованої форми стане таблиця **ПараметрыОтветвленийТрансформаторов**. Через цю частину агато табличного форми Ви набираєте, перегляд і корегування даних для кожного відгалуження типовиконання трансформатора. Число доступних записів визначається кількістю записів в таблиці **ПараметрыОтветвленийТрансформаторов**. Одночасно в підлеглій формі повинні відображатися тільки записи, що пов'язані з відкритим в головній формі типовиконанням трансформатору.

Джерелом записів другої підпорядкованої форми стане таблиця **МагнитныеХарактеристики.** Одночасно в цій формі повинно відображатися по 4-и записи, що пов'язані з відкритим в головній формі типовиконанням трансформатору.

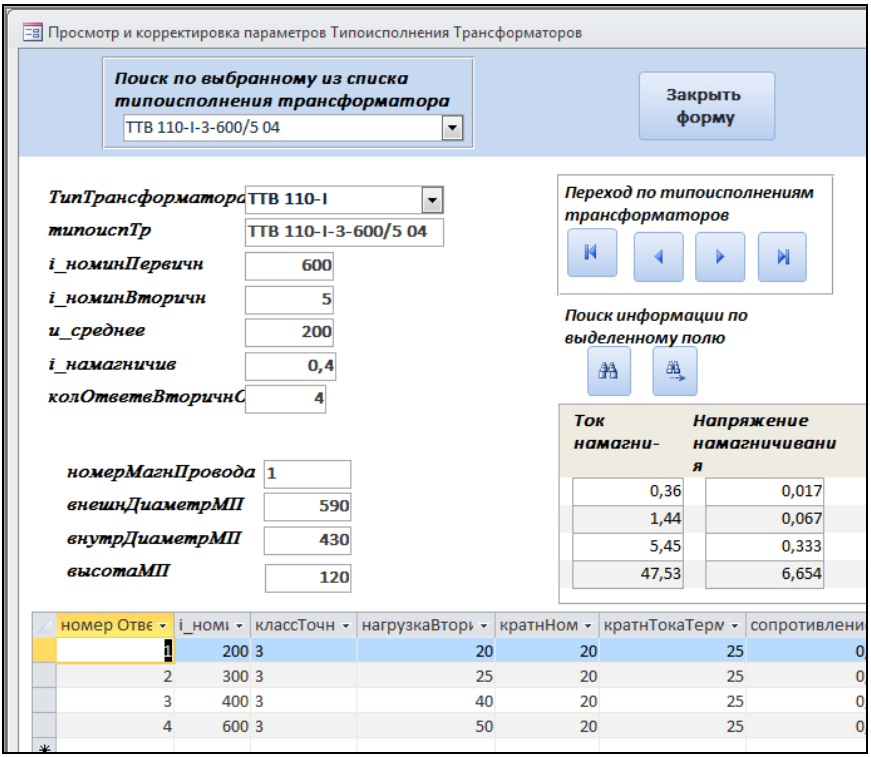


Рисунок 3.3 – форма ПарамТипоисполнений

Порядок створення форми **ПарамТипоисполнений**, наведеній на рис.3.3:

1. За допомогою майстра форм створіть відповідно до результатів проектування форму **ПарамТипоисполнений**, що складається з основної форми і включеної першої підпорядкованої форми.
2. Відкрийте створену форму в режимі *Конструктор*. За допомогою елемента керування *Подчиненная форма/отчет* додайте до основної форми другу підпорядковану форму.
3. Щоб підготувати більш зручний інтерфейс для роботи відредагуйте створену основну та підпорядковані форми засобами *Конструктора форм*.

Підлеглі форми не обов'язково вбудовуються в основну форму (не завжди для них достатньо місця на основній формі). Їх можна викликати з основної форми за допомогою кнопок. Для цього будем використовувати комплекс Форма (основна форма) – Запит (параметричний запит, що відбирає записи з таблиці **МагнитныеХарактеристики** по значенню в полі **типоиспТр** основної форми) – Форма (з даними другої підпорядкованої форми).

Порядок створення форми **ПарамТипоисполнений**, наведеній на рис.3.4:

1. За допомогою майстра форм створіть відповідно до результатів проектування форму **ПарамТипоисполнений**, що складається з основної форми і включеної першої підпорядкованої форми.
2. Створюємо параметричній запит **запросПарамТипоисп** для вибору записів з таблиці **МагнитныеХарактеристики,** що відповідають вибраному типовиконанню трансформатора в основній формі (поле **типоиспТр** в формі **ПарамТипоисполнений)**:

```
SELECT МагнитныеХарактеристики.i_намагнич,
    МагнитныеХарактеристики.u_намагнич
FROM МагнитныеХарактеристики
WHERE (((МагнитныеХарактеристики.типоиспТР)=
    Forms.ПарамТипоисполнений.типоиспТр));
```

3. За допомогою майстра створюємо другу підпорядковану форму (під назвою **Вторая подчиненная форма**) джерелом даних якої є параметричний запит **запросПарамТипоисп.**
4. Основну форму **ПарамТипоисполнений** відкриваєм в режимі *Конструктора форм* и доповнюємо її кнопкою, що дозволяє вивести на екран другу підпорядковану форму.

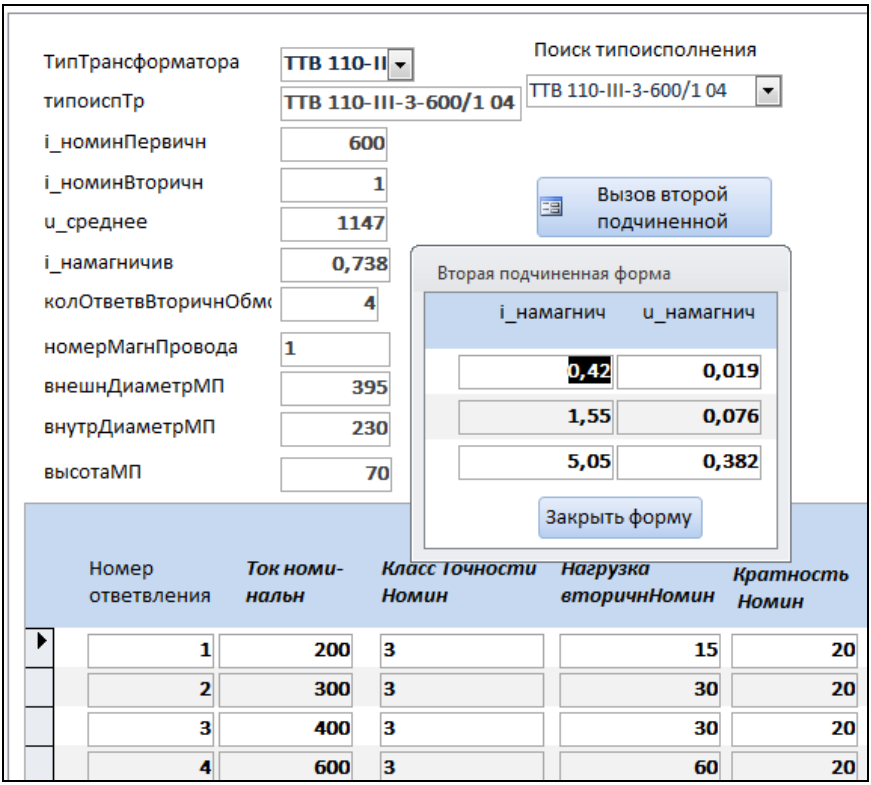


Рисунок3.4 - Приклад багатотабличної форми ПарамТипоисполнений

3.4 Завдання на самостійну роботу

Використовуючи знання про створення форм, придбанні під час виконання лабораторних робіт №3 та №4 попереднього семестру а також під час створення основної форми з двома підлеглими формами поточної лабораторної роботи, створіть для БД "**Трансформаторів струму**" форми, що реалізують основні функції будь-якого додатка Access, а саме:

– форми для уведення даних у таблиці;
– форми для перегляду та редагуванню даних в таблицях;
– форми для пошуку інформації по заданим параметрам.

3.5 Зміст звіту

Звіт повинен містити:

1. Перелік та зображення створених форм.
2. Стислі відповіді на контрольні питання.

3.6 Контрольні питання

1. Для чого призначені форми у базі даних?
2. Як створюються додаткові елементи форми?
3. Як створити на формі обчислюване поле?
4. Які види форм ви знаєте? Охарактеризуйте кожен з них.
5. У якій області форми розташовують підсумкові значення?
6. Призначення підсхеми даних для форми введення/виведення інформації.
7. Створення підлеглих форм.

4 ЛАБОРАТОРНА РОБОТА №4. СТВОРЕННЯ ДОДАТКУ БД «ТРАНСФОРМАТОРИ СТРУМУ»

4.1 Мета роботи

Набути вмінь та навичок створення додатків бази даних для автоматизації роботи користувача, а саме:

• створювати макроси, які автоматизують виконання різних операцій у БД;

• під’єднувати макроси до елементів керування, використовуючи події;

• створювати кнопкові форми за допомогою конструктора;

• настроювати параметри автозапуску додатку.

4.2 Основні відомості

У попередніх лабораторних роботах була розглянута технологія розробки об'єктів бази даних Access: таблиць, форм, запитів, звітів як засобів вирішення завдань і розробки інтерфейсу додатку користувача. При цьому велика кількість об'єктів, що не згрупованих за функціями додатка, ускладнює виконання користувачем завдань обробки даних предметній області, що автоматизується.

Для організації ефективної роботи користувача потрібно створити цілісний додаток даної предметної області, всі компоненти якого повинні бути згруповані за функціональним призначенням. При цьому необхідно забезпечити зручний графічний інтерфейс користувача.

Особливу роль при створенні програми грають форми, так як вони є основним діалоговим засобом роботи користувача. Форми побудовані таким чином, що будь-яка дія користувача викликає реакцію системи, тобто сприймається як подія, в залежності від котрої можуть виконуватися необхідні дії. Для виконання цих дій використовуються макроси або процедури обробки події, створені користувачем на мові VBA, які пов'язані з подією. Таким чином, ходом управління програми можна керувати, обробляючи події, що виникають в формах.

4.2.1 Створення та використання макросів

Макрос – іменований набір макрокоманд для виконання певних операцій, що дозволяє автоматизувати роботу додатка. Для побудови того чи іншого макросу необхідно визначитись з послідовністю його макрокоманд. Ці макрокоманди можна вибрати під час створення макросу за допомогою кнопки «**Создать**» у закладці «**Макросы**».

Нижче наведені макрокоманди та їх короткий зміст:

1. **Восстановить.** Поновлення попередніх розмірів вікна.
2. **ВывестиВФормате**. Виведення даних, що містяться у вказаному об'єкті бази даних MS Access (таблиці, формі, звіті або модулі) у файл в форматі MS Excel 97 (*.xls), в текстовому форматі MS-DOS (*.txt) або в форматі RTF (*.rtf). Також можливе виведення у файли формату HTML (*.html), Microsoft Internet Information Server (*.htx, *.idc) або у файл в форматі сторінок Microsoft ActiveX Server (*.asp).
3. **ВыводНаЭкран.** Визначає режим виведення на екран результатів виконання поточних операцій. Наприклад, ця макрокоманда дозволяє вивести на екран або сховати проміжні результати виконання макросу.
4. **ВыделитьОбъект**. Виділення вказаного об'єкта бази даних.
5. **ВыполнитьКоманду**. Виконання команди MS Access.
6. **Выход**. Вихід з MS Access. При цьому передбачений вибір параметрів зберігання об'єктів бази даних.
7. **Добавить меню**. Виконання наступних дій: — побудова спеціального рядка меню для форми або звіту. Спеціальний рядок меню замінює вбудований рядок меню форми або звіту. — побудова спеціального контекстного меню для форми, елемента управління або звіту. Спеціальне контекстне меню замінює вбудоване контекстне меню форми, елемента управління або звіту. — побудова загального рядка меню. Загальний рядок меню замінює вбудоване головне меню у всіх вікнах MS Access, за винятком тих, що містять спеціальні рядки меню форми або звіту. — побудова загального контекстного меню. Загальне контекстне меню замінює вбудоване контекстне меню для всіх полів таблиць або запитів в режимі «Таблицы», форм в режимі «Формы», режимі «Таблицы» і режимі попереднього перегляду, а також звітів в режимі попереднього перегляду, за виключенням тих, де користувач додав спеціальне контекстне меню форми, звіту або

елемента управління. *Примітка.* Для побудови нових меню рекомендується все ж таки використовувати діалогове вікно «**Настройка панелей инструментов**», з меню «**Вид**» команди — «**Панели инструментов**» і кнопки «**Настройка**».

8. **ЗадатьЗначение**. Завдання значень поля, елемента управління або властивості в формі
9. **ЗадатьКомандуМеню**. Завдання стану команд спеціального рядка меню або загального рядка меню для активного вікна.
10. **Закрыть**. Зачинення вказаного вікна, вікна MS Access або поточного вікна, за угодою.
11. **ЗапускЗапросаSQL**. Запуск запиту на зміни MS Access за допомогою відповідної інструкції SQL. Крім того, ця макрокоманда дозволяє запустити керуючий запит.
12. **ЗапускМакроса**. Запуск макросу. Можна вказувати макрос з групи макросів.
13. **ЗапускПриложения**. Запуск із MS Access програмного забезпечення Windows або MS-DOS, наприклад, Microsoft Excel, Microsoft Word для Windows або Microsoft PowerPoint. Наприклад, ця макрокоманда дозволяє виконати вставку електронної таблиці MS Excel в базу даних MS Access.
14. **ЗапускПрограммы.** Виклик функції Visual Basic.
15. **КомандыКлавиатуры**. Передача натискань клавіш безпосередньо у MS Access або в активне програмне забезпечення MS Windows.
16. **КопироватьОбъект**. Копіювання вказаного об’єкта бази даних в іншу базу даних MS Access або в ту ж саму базу даних під новим ім’ям. Наприклад, цю макрокоманду використовують для копіювання або зберігання існуючого об’єкта в іншій базі даних або при використанні існуючого об’єкта в якості прототипу нового об’єкта.
17. **КЭлементуУправления**. Переведення фокусу на вказане поле або елемент управління у поточному записі форми, таблиці, запиту. Крім цього, ця макрокоманда використовується для автоматичного переміщення по формі у відповідності з визначеними умовами. Наприклад, якщо оператор введе «Так» в поле «Холост», то поле «Супруг» буде пропущено автоматично, а фокус переданий наступному елементу управління.

18. **Назапась**. Зробити вказаний запис поточним для таблиці, форми або записів запиту.
19. **НайтиЗапись**. Пошук даних, що задовольняють умовам пошуку цієї макрокоманди.
20. **НаСтраницу**. Передача фокусу в активній формі першому елементу управління. Ця макрокоманда використовується при побудові багатосторінкових форм з групуванням даних на різних сторінках.
21. **ОбновитьОбъект**. Завершення всіх відкладених операцій оновлення вказаного об'єкта бази даних або активного об'єкта бази даних, за угодою. При необхідності, виконується перерозрахунок значень елементів управління в цьому об'єкті.
22. **Обновление**. Оновлення даних у вказаному елементі управління в активному об'єкті шляхом повторного перегляду даних.
23. **ОстановитьВсеМакросы**. Зупинка виконання всіх макросів, що виконуються
24. **ОстановитьМакрос**. Зупинка виконання поточного макросу, що виконується
25. **ОткрытьЗапрос**. Відкриття запиту на вибірку або перехресного запиту в режимі «Таблицы», в режимі «Конструктор» або в режимі попереднього перегляду. Крім цього, ця макрокоманда може вказати для запиту режим введення даних.
26. **ОткрытьМодуль**. Відкриття вказаної процедури в модулі Visual Basic. Ця процедура може бути процедурою Sub, процедурою Function або процедурою обробки подій.
27. **ОткрытьОтчет**. Відкриття звіту в режимі Конструктор, в режимі попереднього перегляду або виведення звіту на друк. При цьому допускається вибірка записів, що включаються в звіт.
28. **ОткрытьТаблицу**. Відкриття таблиці в режимі «Таблица», в режимі «Конструктор» або в режимі попереднього перегляду. При цьому допускається вибір режиму введення даних в таблицю.
29. **ОткрытьФорму**. Відкриття форми в режимі «Форма», в режимі «Конструктор форми», в режимі попереднього перегляду або в режимі «Таблица». При цьому допускається вибір режиму введення даних та режиму вікна, а також відбір записів, які необхідно показати у формі.
30. **ОтменитьСобытие**. Відміна події, яка викликала запуск макросу.

31. **ОтправитьОбъект**. Включення вказаного об’єкта в режимі «Таблица», «Форма», «Отчет» або «Модуль MS Access» у повідомлення електронної пошти, перегляд та відправлення. Допускається включення об’єктів у форматі MS Excel (*.xls), текстовому форматі MS-DOS (*.txt), форматі RTF (*.rtf) або HTML у повідомлення електронної пошти Microsoft Exchange, Microsoft Mail, Microsoft Windows для робочих груп або іншого програмного забезпечення електронної пошти, яке використовує інтерфейс MAPI (Microsoft Mail Applications Programming Interface). Є можливість пересилки об’єктів MS Access у повідомлення електронної пошти VIM- інтерфейсу.
32. **ПанельИнструментов**. Виведення на екран або видалення з екрану вбудованої або спеціальної панелі інструментів. Користувач має можливість вказати виведення вбудованої панелі інструментів у всіх вікнах MS Access або тільки в одному вікні, для якого ця панель інструментів призначена, наприклад, вивести панель інструментів «Режим» форми тільки у вікні форми в режимі «Форма».
33. **Переименовать**. Перейменування вказаного об’єкта бази даних.
34. **ПесочныеЧасы**. Надати вказівнику форму піскового годинника, або іншого вказаного значка, на час виконання макросу. Ця макрокоманда дозволяє побудувати вказівник, що свідчить про виконання дій поточним макросом. Це особливо зручно, коли виконання макрокоманди або самого макросу займає достатньо великий час.
35. **Печать**. Друк активного об’єкта відкритої таблиці бази даних.
36. **ПоказатьВсеЗаписи**. Відміна будь-якого поточного фільтру.
37. **Преобразовать БазуДанных** . Перетворення бази даних при імпорті або експорті даних між поточною базою даних MS Access та іншою базою даних
38. **ПреобразоватьТекст**. Перетворення тексту при імпорті або експорті даних між поточною базою даних MS Access та текстовим файлом (а також файлом HTML (*.html)).
39. **Преобразовать ЭлектроннуюТаблицу**. Перетворення електронної таблиці при імпорті або експорті даних між поточною базою даних MS Access та файлом електронної таблиці.

40. **ПрименитьФильтр**. Застосування фільтру, запиту або інструкції WHERE (SQL) до таблиці, форми або звіту для вибірки і сортування записів.
41. **Развернуть**. Збільшення розмірів активного вікна до розмірів вікна MS Access.
42. **Свернуть**. Згортання активного вікна.
43. **СдвигРазмер**. Зміна місця та розмірів активного вікна.
44. **Сигнал**. Подання звукового сигналу через динамік комп'ютера.

Використовуючи макроси для виконання рутинних операцій, можна істотно заощадити час і сили. Крім того, оскільки всякий раз при запуску макросу буде здійснюватися одна і та сама послідовність дій, макрос зробить роботу з базою даних більш ефективною і акуратною.

Макроси можуть бути як простими, що складаються з однієї дії, так і складними, що включають логічні операції і кілька взаємопов'язаних дій.

Об'єктам (наприклад, формам, звітам) і елементам керування на них (наприклад, кнопкам, текстовим полям) можна на виконання різних подій призначати різні макроси. Прикладами подій можуть бути клацання кнопки миші, відкривання або закривання форми, запуск звіту або змінення даних у текстовому полі тощо. Наприклад, якщо до форми додати кнопку, подія *OnClick* кнопки пов'язуватиметься з макросом, що містить команди, які мають виконуватися при кожному натисканні кнопки.

Створення об'єкту Макрос

Для створення макросів використовується *Конструктор макросів* (рис.4.1), який можна запустити командою *Макрос* на вкладці *Створення* у групі *Макроси та код*:

Виконання макросів ініціюється простою операцією і може зводитися до його відкриттю, як це робиться і для інших об'єктів бази даних. Крім цього, Access надає можливість автоматично ініціювати виконання макросу при настанні деякої події. Для зв'язку макросу з подією досить у вікні властивостей об'єкта або його елемента управління внести в рядок цього події ім'я макросу або створити впроваджений макрос. Події, з якими можна зв'язати макрос, представлені у властивостях форм і звітів і їх елементів управління.

Рисунок 4.1 - Кнопка для запуску Конструктора Макросів

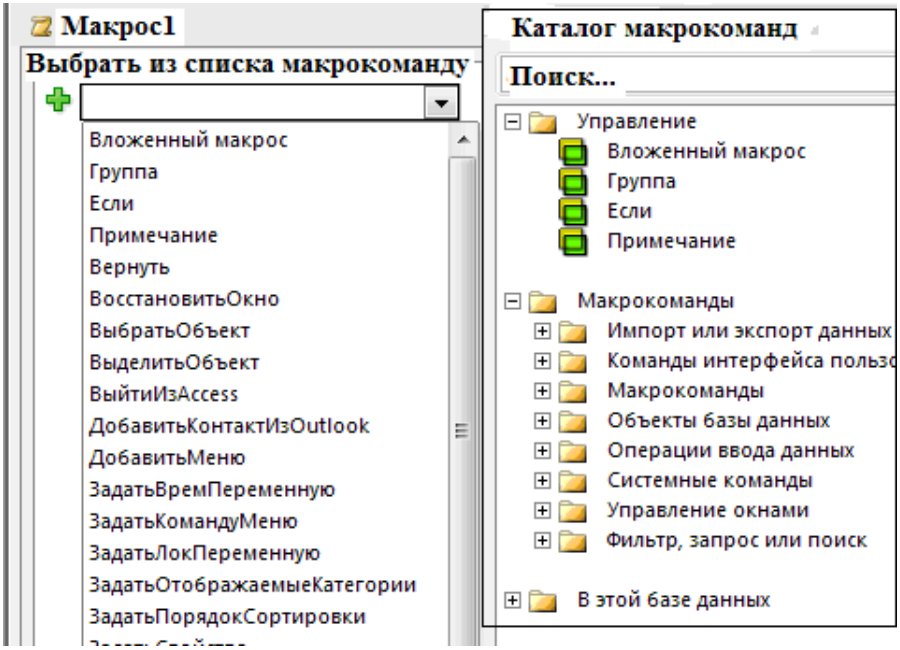


Рисунок 4.2 - Вікно Конструктора Макросів

Формування макрокоманд в вікні макросу

При створенні нового макросу в його вікні відображається поле *Добавить новую макрокоманду* (Add New Action) із списком (рис. 4.2). В списку представлений весь доступний набір макрокоманд. Цей набір можна змінювати, натискаючи кнопку *Показать все действия* (Show All Action) на стрічці конструктора макросів в групі *Показать или скрыть* (Show / Hide). Якщо кнопка не було натиснуто, то в набір не включаються так звані небезпечні дії.

Після введення макрокоманди в макросі відображається блок, що містить ім'я макрокоманди і рядки її аргументів . Значення

аргументів задаються шляхом вибору їх зі списку, що відкривається в рядку аргументу, за допомогою будівника або вручну.

Для обраного аргументу або макрокоманди виводиться підказка. Знак мінус зліва від імені макрокоманди дозволяє приховати її аргументи. Зелені стрілки в правій частині блоку дозволяють переміщати макрокоманду вище або нижче інших макрокоманд. Тут же є значок видалення макрокоманди.

Для введення в макрос коментаря використовується блок *Примечание* (Comment), розміщений у вікні каталогу макрокоманд в розділі *Управление* (Program Flow). При перетягуванні цього блоку в макрос створюється порожній блок, в який і вводиться потрібний коментар. Після завершення введення коментар відображається рядком зеленого кольору, укладеної в знаки / * і */.

Після введення всіх макрокоманд в макрос його треба зберегти, скориставшись командою *Сохранить* (Save).

Управління послідовністю виконання макрокоманд

У макросі макроси виконуються в порядку їх розташування. Однак для реалізації алгоритму в програмах часто необхідно порушити послідовність їх виконання в залежності від деяких умов. Умови дозволяють визначити порядок передачі управління між макрокомандами в макросі і забезпечують виконання певних гілок алгоритму. Наприклад, в макросі перевіряється значення поля в формі на відповідність заданим умовам, і для одних значень може знадобитися вивести повідомлення, а для інших значень провести висновок звіту.

Для зміни порядку виконання макрокоманд може бути використаний керуючий блок:

Если условное выражение ***то*** **[**макрокоманды_то**]** ***иначе*** [макрокоманды_иначе]

Конец блока "Если" **(If ... Then ... Else ... End If)**

Блок забезпечує виконання однієї або іншої групи макрокоманд в залежності від значення умовного виразу.

Умовний вираз є логічним виразом, яке повертає значення **Истина** (True) або **Ложь** (False). Якщо умовний вираз має значення **Истина**, то виконуються [макрокоманди_то], в іншому випадку – [макрокоманди_ інакше].

Для включення в макрос блоку ***Если*** (If) потрібно вибрати його

зі списку в поле *Додать новую макрокоманду* (Add New Action) або перетягнути в потрібне місце з каталогу макрокоманд.

4.2.2 Кнопкові форми

Для об'єднання об'єктів в єдиному діалоговому додатку можуть бути створені так звані кнопкові форми. Кнопкова форма є панеллю керування додатком. Кнопки такої форми забезпечують виклик інших кнопкових форм, а також окремих об'єктів: звітів, форм, макросів, модулів, з яких починається рішення задачі. Сюди ж поміщаються і кнопки для повернення до кнопкових форм попередніх рівнів та для виходу з Access. Зазвичай також передбачається кнопка для змін самої кнопкової форми.

Користувач, натискаючи кнопку на панелі управління, ініціює подію натискання кнопки. До цієї події можуть прив'язуватися дії по відкриванню інших кнопкових форм або конкретні дії по обробці даних, що реалізують функції програми.

Виклик головної кнопкової форми при відкриттю бази даних дозволяє користувачеві відразу почати роботу в середовищі додатку та приступити до виконання завдань.

Проектування схеми керування БД «Трансформатори струму»

Проектування схеми керування створеної бази даних (додатку) виконується за технологією «зверху-донизу», тобто від головної кнопкової форми до функціональних форм і звітів, що викликаються. Оскільки всі функціональні форми і звіти вже існують (створені в лабораторній роботі №8), то залишилося побудувати тільки кнопкові форми.

Фрагмент структури додатка БД «Трансформатори струму» ілюструє рис. 4.3, а приклад відповідної головної кнопкової форми наведений на рис. 4.4.

Зрозуміло, що структура додатку та головна кнопкова форма додатку можуть мати будь який вигляд. Так, на рис.4.5 наведений ще один приклад головної форми додатку БД «Трансформатори струму».

Створення кнопкової форми

Головна і підлеглі їй кнопкові форми можуть бути створені користувачем самостійно в режимі конструктора.

Для створення кнопкової форми необхідно у вікні бази даних на вкладці *Создание* натиснути кнопку *Конструктор форм*. При цьому

не має зазначатися джерело даних. Відкривається форма в режимі конструктора. У цій формі можна створити кнопки для виклику кнопкових форм або виклику будь-яких інших об'єктів програми. Кнопкова форма може бути збережена під будь-яким ім'ям і в будь-який момент відредагована в режимі конструктора. Таким образам можуть бути створені всі необхідні кнопкові форми додатка.

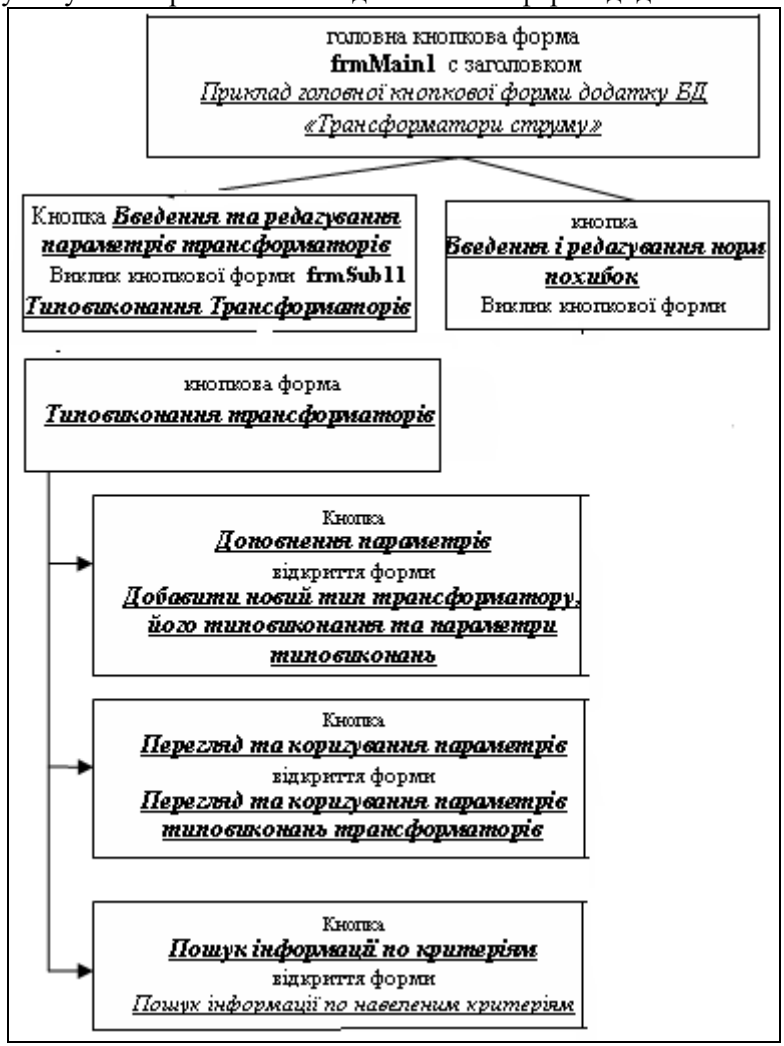


Рисунок 4.3 - Фрагмент структури додатка БД «Трансформатори струму»

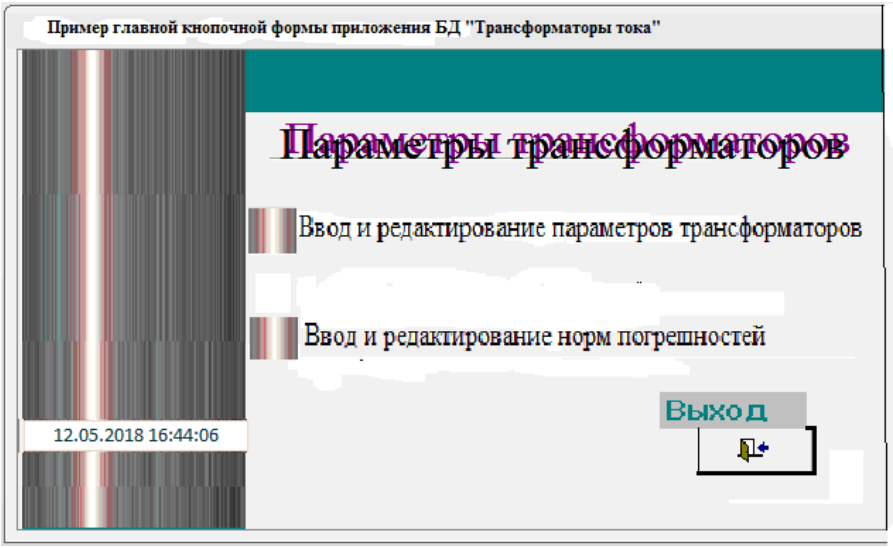


Рисунок4.4 – Приклад головної кнопкової frmMain1 додатку БД «Трансформатори струму»

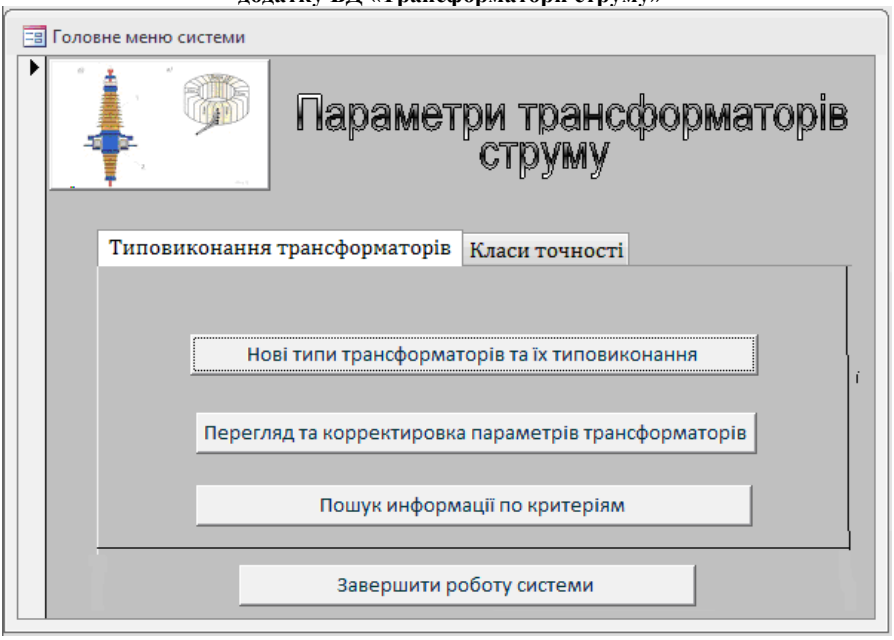


Рисунок 4.5 – Приклад головної кнопкової форми frmMain2 додатку БД «Трансформатори струму»

Створення кнопок за допомогою Майстра

Користувач має можливість створити кнопку самостійно або за допомогою майстра. Майстри значно прискорює процес побудови кнопки і зв'язування її з потрібними діями, автоматично виконуючи всю необхідну роботу. Майстер веде діалог з користувачем і на основі його відповідей створює кнопку. Технологія створення кнопок *Майстром* використовувалася в лабораторній роботі №4.

Майстер дозволяє створювати кнопки більше 30 типів. Наприклад, кнопки для від-криття форми, сторінки, виконання запиту, макросу, виходу з програми або виконання інших дій. Відкриття кнопкової форми нічим не відрізняється від відкриття звичайної форми.

Створення кнопки на формі за допомогою Макросу

На вкладці **Типовиконання трансформаторів** головної кнопкової форми **frmMain2** (рис.4.5) створимо перехід до форми **frmSub1** (рис.4.6) за допомогою макросу **macForBtn1**.

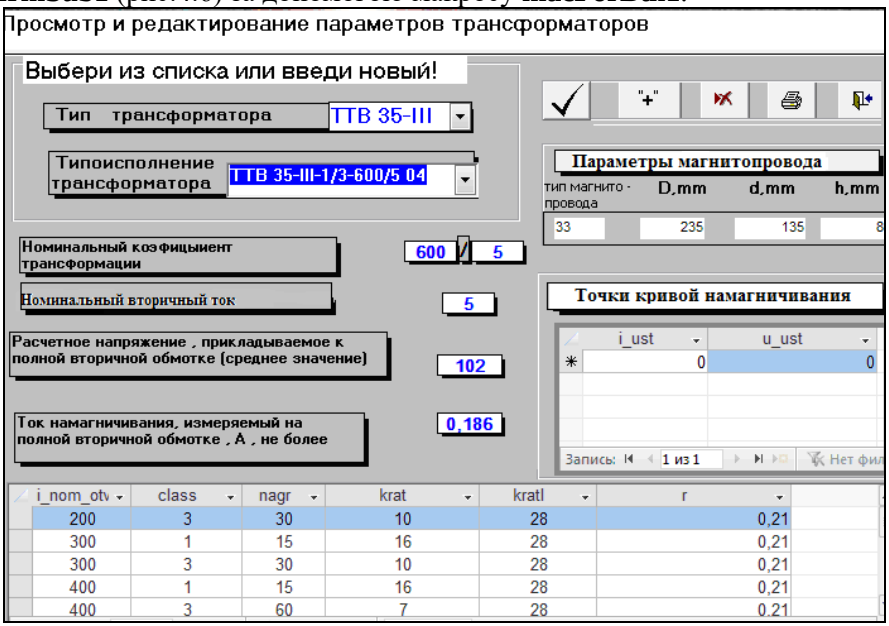


Рисунок 4.6 – Форма frmSub1

Створення макросу виконується в такому порядку:

1. На вкладці Создание панелі Макросы и код натисніть кнопку

Макрос.

3. Натисніть кнопку *Сохранить*, введіть ім'я макросу **macForBtn1** у вікно, що з'явилося, і натисніть кнопку ОК.

3. У вікні, що з¢явилося, виберіть елемент *Открыть Форму* у списку, що розкривається.

4. У полі *Имя формы* виберіть ім`я форми **frmSub1**.

5. Уполі *Режим данных* виберіть у списку, що розкривається, *Изменение*.

6. У полі *Режим окна* залиште *Обычное*. Закрийте вікно конструктора макросів із збереженням зроблених змін (рис. 4.7).

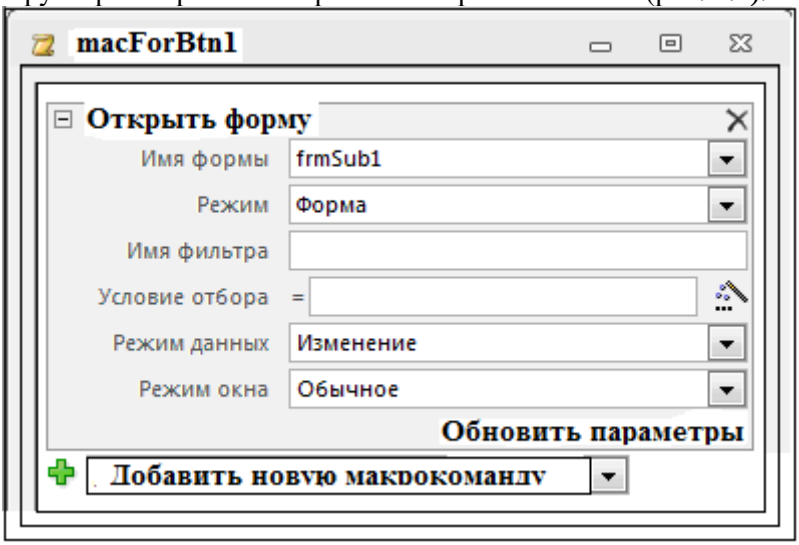


Рисунок. 4.7 – Формування макросу macForBtn1

Створений таким чином макрос не виконує ніяких дій, поки не станеться подія, з якою він пов'язаний, тобто необхідно створити зв'язок макросу з кнопкою **btn1** з надписом **«Перегляд та корегування параметрів»** на сторінці **Типовиконання трансформаторів** головної кнопкової форми **frmMain2** (рис.4.5).

Для цього необхідно виконати таку послідовність дій:

1. Відкрийте форму ***frmMain2***.

2. За допомогою вкладки *Конструктор* панелі *Элементы управления* створіть кнопку. У вікні, що з`явилося, майстра *Создание кнопок* натисніть *Отмена*.

3. Виділіть на формі кнопку і викличте вікно її властивостей.

4. У вкладці *События* клацніть у рядку *Нажатие кнопки*.

5. У списку, що розкривається, виберіть щойно створений макрос **macForBtn1** (рис. 4.7).

6. Збережіть форму й перейдіть у режим форми.

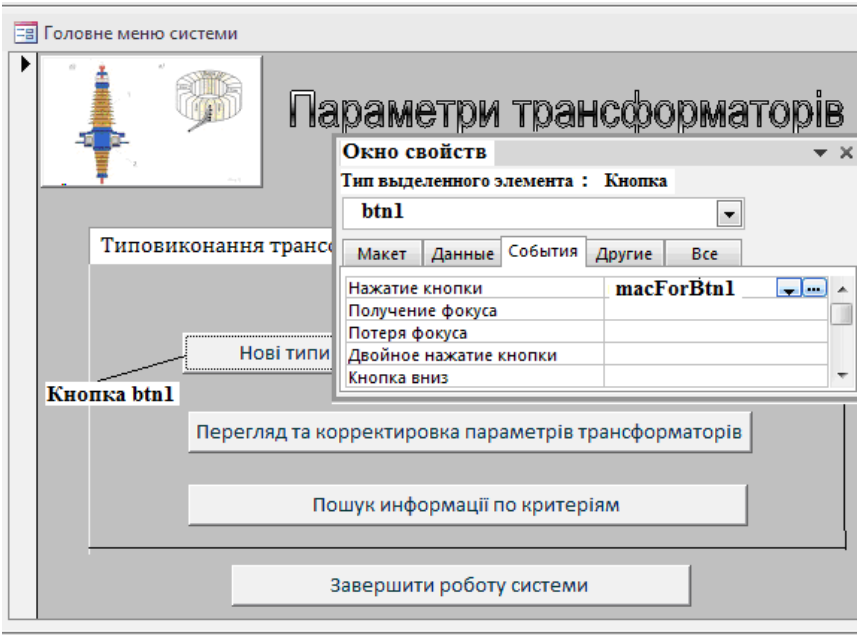


Рисунок 4.7 – Зв'язування макросу macForBtn1 з подією *Нажатие кнопки* btn1

4.2.3 Налаштування параметрів автозапуску

Налаштуйте застосування додатку БД «**Трансформатори струму**» таким чином, щоб користувач працював із даними бази тільки через кнопкові форми. Для цього потрібно налаштувати параметри автозапуску MS Access. Для того щоб після запуску додатку відразу відображалася головна кнопкова форма, а вікно бази даних було відсутнє і у заголовках всіх вікон додатку було видно відповідний значок, потрібно виконати такі дії:

1. На вкладці *Файл* вибрати команду *Параметры*. У вікні *Параметры Access*, що з'явилося, вибрати розділ *Текущая база данных*.

2. У розділі *Параметры приложений* вибрати головну кнопкову форму як *форму Просмотра***,** а також за допомогою кнопки *Обзор* вказати розташування малюнка додатку.

3. У розділі *Навигация* зняти прапорець із команди *Область навигации*.

4. У розділі *Навигация* зняти прапорець із команди *Параметры ленты и панелей инструментов*.

Після цього слід закрити базу даних. При подальшій роботі з базою даних **«Трансформатори струму»** користувач зможе виконувати навігацію тільки за допомогою кнопкової форми, тобто буде тільки переглядати та додавати в разі необхідності нові дані, виконувати пошук необхідної інформвції, вносити зміни у структуру бази даних він не зможе.

У разі необхідності повернутися до режиму редагування створеного додатка треба в момент відкриття бази даних, тримаючи натиснутою клавішу **Shift**.

4.3 Порядок виконання лабораторної роботи

1. За аналогією з вищерозглянутим інтерфейсом БД **“Бібліотека”** створіть інтерфейс БД **“Трансформатори струму”.** Інтерфейс повинен мати:
   – головну (кнопкову) форму, яка виконує дві функції:
      а) являє собою точку запуску користувальницького додатку;
      б) дає можливість вибору подальших дій;
   – форми, що використовуються для доповнення таблиць даними;
   – форми для перегляду даних, наприклад:
      а) форма в якій для вибраного типовиконання трансформатора маємо переглянути всі його параметри;
      б) форма, яка виводить список всіх типовиконань трансформаторів по заданих значеннях номінальних токів(параметри i_1, i_2);
   – звіти по тим трансформаторам , які на данний час не виробляються, і дані по цім трансформаторам перенесені до архівної таблиці;
   – декілька форм для пошуку інформації по заданим критеріям.
2. За допомогою *Майстрів* та *Макросів* автоматизуйте запуск і роботу

вашого додатка.

3. Настройте вікно MSAccess таким чином, щоб елементи інтерфейсу були відключені і замінені елементами інтерфейсу вашої бази даних.

4.4 Зміст звіту

Звіт повинен містити опис створеного інтерфейсу БД **“Трансформатори струму”**

4.5 Контрольні питання

1. Етапи розробки інтерфейсу БД.
2. Головні об’єкти інтерфейсу БД.
3. Призначення *Кнопкової* форми.
4. Створення макросу в Конструкторі макросів.
5. Способи запуску макросів на виконання.
6. Способи налаштування параметрів авто запуску БЛ.

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна

1. Гурвиц Г. А. Microsoft Access 2010. Разработка приложений на реальном примере. – Издательство: BHV, 2010. – 496с.
2. Мирошниченко Г. А. Реляционные базы данных: практические приемы оптимальных решений. –СПб: БХВ-Петербург, 2005. –400с.
3. Єрьоміна Н. В. Проектування баз даних: Навчальний посібник. –К. : КНЕУ, 1998. – 208 с.
4. Джеффри Д. Ульман, Дженнифер Уидом. Введение в системы баз даннях. – Издательство «Лори», 2000. – 376 с.
5. Сайт додатків Office корпорації Microsoft [Електронний ресурс]. http://office.microsoft.com/ru-ru/access-help/.

Додаткова

1. Методичні вказівки до самостійної роботи з курсу “Програмні засоби проектування електромеханічних систем” для студентів денної та заочної форм навчання напряму підготовки спеціальності 141 "Електроенергетика, електротехніка та електромеханіка" – / Укл.: Л.О.Бондаренко, О.О.Каплієнко. – Запоріжжя: ЗНТУ, 2015.– с.70. №5706е.
1. Ivankov V. F. Calculation of CFD-thermal models of oil-cooled transformer equipment / V. F. Ivankov, A. V. Basova // Електротехніка та електроенергетика. - 2016. - № 2. - С. 19-32.
2. Ніценко В. В. Дослідження похибок трансформаторів струму у системах релейного захисту в усталених та перехідних режимах енергосистеми / В. В. Ніценко, Д. О. Кулагін, П. В. Махлін // Електротехніка та електроенергетика. - 2016. - № 2. - С. 59-71.
3. Ніценко В. В. Дослідження похибок трансформаторів струму у системах релейного захисту в усталених та перехідних режимах енергосистеми / В. В. Ніценко, Д. О. Кулагін, П. В. Махлін // Електротехніка та електроенергетика. - 2016. - № 2. - С. 59-71.
4. Дивчук Т. Е. Особенности определения параметров силовых трансформаторов методами схемно-полевого моделирования / Т. Е. Дивчук, Д. К. Мимоход, С. А. Кутилин, А. Э. Кузнецов, Ю. В. Гуразда, И. С. Сырых // Електротехніка та електроенергетика. - 2017. - № 1. - С. 61-70.