# ЛАБОРАТОРНА РОБОТА №11

## Тема

## Реалізація виконання запитів до бази даних

## Мета роботи

Розробка програмного продукту, що реалізує інтерфейс з базою даних

## Теоретичні відомості

Програмна система, що використовує інформацію, збережену у базі даних, має надавати можливість виконувати певні операції з нею. Усі такі операції – це запити до бази даних. Кожна з них – це окрема транзакція, що містить команди SQL.

## Реалізація за допомогою C++

Розіб'ємо ці операції на три групи і розглянемо, як використати ADO-компоненти та стандартні компоненти задля виконання їх.

1. Операції відбору та представлення результатів на екран

Для відбору даних використовується команда *SQL SELECT*, для формування параметрів запиту та представлення результатів можуть бути використані стандартні компоненти C++Builder: *Edit*, *StringGrid*, *ComboBox*, *CheckBox* та інші.

Найбільш зручний компонент для виконання команди *SQL* – це *ADOQuery*. За допомогою властивості «SQL» цього компонента задається текст команди, переведення властивості «Active» у значення «True» дозволяє виконати цю команду, властивість «RecordCount» визначає кількість отриманих записів, метод «FindNext» реалізує перехід по записах, метод «FieldByName» - отримання значення вказаного стовпчика поточного запису.

На рис. 1 представлений код методу, який дозволяє виконати будь-який запит *SELECT*.

```
int __fastcall ActiveQuery(AnsiString TextWhere,TADOQuery *ADOQ) {
if(ADOQ->Active)
ADOQ->Close(); ADOQ->SQL->Clear();
ADOQ->SQL->Add(TextWhere); ADOQ->Open();
return ADOQ->RecordCount;
}
```

Рис. 1 Виконання запиту SQL

На рис.2 наведений приклад використання методу *ActiveQuery*.

```
AnsiString TextQuerySpec="select specCod, spMainCod from SpSpec where facultetCod=";
TextQuerySpec+=arrayCodFacultet[edFacultet->ItemIndex];
int FormSpecCount=ActiveQuery(TextQuerySpec,DMAb->ADOQAll);
```

Рис.2 Використання методу ActiveQuery

У прикладі формується команда *SQL* для відбору записів з таблиці *SpSpec* (спеціальності ВНЗ), які викладаються на факультеті з кодом, що вибирається за допомогою компонента *ComboBox (edFacultet)* із заздалегідь сформованого масиву *arrayCodFacultet*.

На рис.3 представлений код фрагменту програми, що представляє отримані запити в осередку компонента *ComboBox*.

```
int SpMainCount=ActiveQuery(TextQuerySpMain,DMAb->ADOQAll);
shSpMain->Clear();
if (SpMainCount!=0) {
DMAb->ADOQAll->FindFirst();
int i=0;
for(int j=1;j<=SpMainCount;j++) {
shSpMain->Items->Add(DMAb->ADOQAll->FieldByName("SpMainName")->AsString);
arraySpMainCod[i++] =DMAb->ADOQAll->FieldByName("SpMainCod")->AsString.SubString(1,8);
if (j!=SpMainCount)
DMAb->ADOQAll->FindNext();
}
}
else {
shSpMain->Items->Add("Для факультету не визначені напрями підготовки.");
return;
}
```

Рис.3 Фрагмент програми

У прикладі викликається вище описаний метод *ActiveQuery*. Разом із заповненням назв напрямів навчання в осередок компонента *ComboBox (shSpMain)* заповнюється і масив *arraySpMainCod*, що містить коди напрямів.

2. Операції вставки, корегування та видалення записів

Ці операції реалізуються командами *INSERT, UPDATE, DELETE*. Якщо Ви їх не виконуєте за допомогою збережених процедур, то можете використати також компонент *ADOQuery*. Але необхідно зауважити, що ці команди, на відміну від команди *Select*, не повертають кількість отриманих записів. Тому, для їхнього виконання за допомогою компонента *ADOQuery*, необхідно використати метод «ExecSQL». Приклад фрагменту програми виконання команди *Insert* наведений на рис. 4.

```
DMAb->ADOQuery1->SQL->Clear();
DMAb->ADOQuery1->SQL->Add(TextInsert); try {
DMAb->ADOQuery1->ExecSQL();
}
catch (...) {
MessageBoxA(0,"Запис до тимчасової таблиці не відбувся. ",
"Помилка!", MB_OK); return ;
}
```

Рис.4 Вставка запису

3. Операції, пов’язані із викликом збережених процедур

Для виклику збережених процедур Ви можете скористатися спеціальним компонентом *ADOStoredProc*. За допомогою властивостей компонента вказуєте назву процедури та її параметри. Виконуєте її або за допомогою метода, або за допомогою властивості *Active*.

Ви також можете скористатися знову компонентом *ADOQuery*. Тоді в осередку властивості *SQL* Ви записуєте команду *Exec*, назву процедури, її параметри. Параметри можете задавати за допомогою властивості *Parameters*. Виконання цього запиту реалізується за допомогою команди *ExecSQL*.

## Реалізація за допомогою C#

1. Операції відбору

При створені проекту в VisualStudio для представлення результатів можуть бути використані такі компоненти: *TextBox*, *Label*, *DataGridView*, *ComboBox* та інші.

На рис. 5 наведено використання методів класу *SqlCommand*, для виконання запиту *SELECT*. У прикладі формується запит для визначення середнього значення продуктивності господарства з таблиці *Productivity*. Отриманий результат записується у зміну типу Double для подальшого використання (відображення у відповідному *TextBox)*.

```
SqlConnection connection;
…
try
{
Double avg;
string sql = "SELECT AVG(ProductivityFact.egg_count) FROM Productivity";
SqlCommand cmd = new SqlCommand(sql, connection);
avg = Convert.ToDouble(cmd.ExecuteScalar());
text_avg.Text = aver.ToString();
}
catch (Exception ex)
{
MessageBox.Show(ex.Message, "Error",  MessageBoxButtons.OK, MessageBoxIcon.Error);
}
```

Рис. 5 Виконання запиту відбору SQL

2. Операції вставки, корегування та видалення записів

На рис. 6 наведено приклад коду виконання параметризованого запиту Delete. За допомогою запиту відбувається видалення з бази даних відповідної породи. Породу, яку необхідно видалити користувач обирає за допомогою компоненту *ComboBox – cb_id_breed*. Для виконання INSERT, UPDATE, DELETE підходить метод класу *SqlCommand – ExecuteNonQuery*. Цей метод виконує *SQL*-запит і повертає кількість доданих/змінених/видалених записів.

```
var result = MessageBox.Show("Ви впевнені що бажаєте видалити дані ", "Yes or No",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
if (result == System.Windows.Forms.DialogResult.Yes)
{
try
{
string sql = "DELETE FROM breed_group WHERE id_breed=@id";
SqlParameter idParam = new SqlParameter("@id", cb_id_breed.SelectedValue.ToString());
SqlCommand command = new SqlCommand(sql, connection);
command.Parameters.Add(idParam);
int number = command.ExecuteNonQuery();
MessageBox.Show("Видалено " + number + " рядків", "", MessageBoxButtons.OK,
MessageBoxIcon.Information);
}
catch (Exception exc)
{
MessageBox.Show(exc.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error); }
}
```

Рис.6 Видалення запису

3. Операції, пов’язані із викликом збережених процедур

Для виклику збережених процедур Ви також можете скористатися методами класу *SqlCommand*. Тоді у властивості *CommandType* необхідно вказати *CommandType.StoredProcedure*. Параметри можна задавати за допомогою властивості *Parameters*. Виконання такого запиту можна реалізувати за допомогою команд *ExecuteScalar*, *ExecuteReader*, *ExecuteNonQuery* (якщо процедура нічого не повертає).

## Хід виконання роботи

1. Розробіть програмний код, який дозволяє реалізувати усі операції по обробці інформації відповідно постановці завдання. Ви можете запропонувати свій підхід до реалізації необхідних алгоритмів.

Результати подати у вигляді звіту з представленням коду усіх функцій.