Laboratorul 5

Lucrul cu baze de date în Visual Basic .NET

Ce ne propunem astăzi?

În laboratorul de astăzi ne propunem să realizăm o aplicație de gestiune a datelor studenților, date care sunt stocate într-o bază de date Microsoft Access (Figura 1).

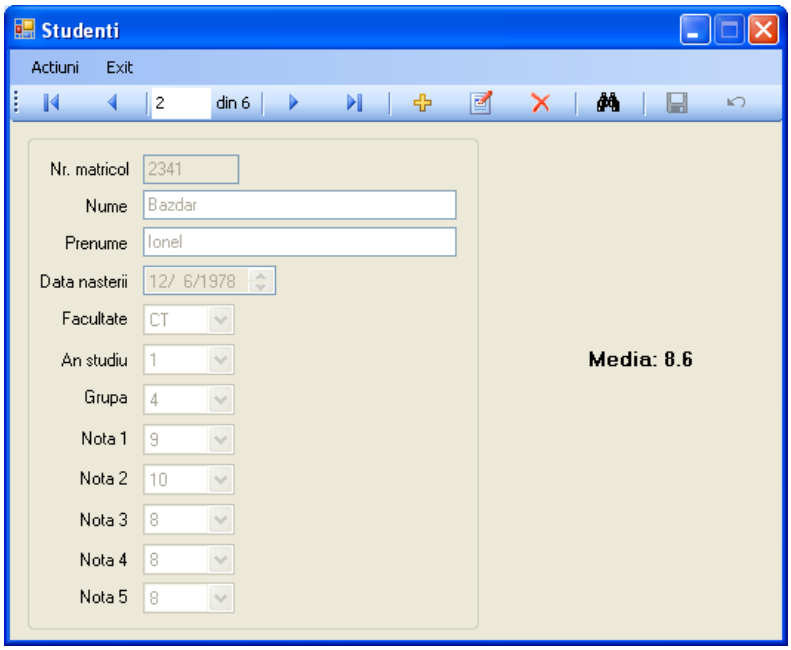

Figura 1. Interfaţa principală a aplicaţiei.

Mai pe larg, vom proceda astfel...

Primul pas în realizarea acestei aplicaţii este crearea bazei de date propriu-zise. Pentru aceasta se va deschide aplicaţia Microsoft Access şi se va crea o bază de date goală (care va fi salvată în C:\Work\MTP). Se va crea apoi un tabel nou în această bază de date (comanda „Create table in Design view" – Figura. 2).

În continuare se va construi structura tabelului adăugând câmpurile ce se pot vedea în Figura 3. Câmpul Nr_matricol va fi ales cheie primară (click dreapta pe numele câmpului → Primary key). Un câmp sau un set de câmpuri de tip cheie primară identifică în mod unic fiecare înregistrare din tabel. Odată declarat cheie primară, un câmp nu poate conţine duplicate sau valori NULL. Se aleg chei primare câmpuri de tip Autonumber (câmpuri numerice cu auto-incrementare),

câmpuri de care suntem siguri că nu vor conţine valori duplicate (de exemplu CNP sau un număr unic de identificare), sau mai multe câmpuri care prin combinare dau o valoare unică.

După ce au fost introduse şi stabilite proprietăţile câmpurilor, se va salva tabelul cu numele „Studenti” apoi se vor adăuga câteva înregistrări.

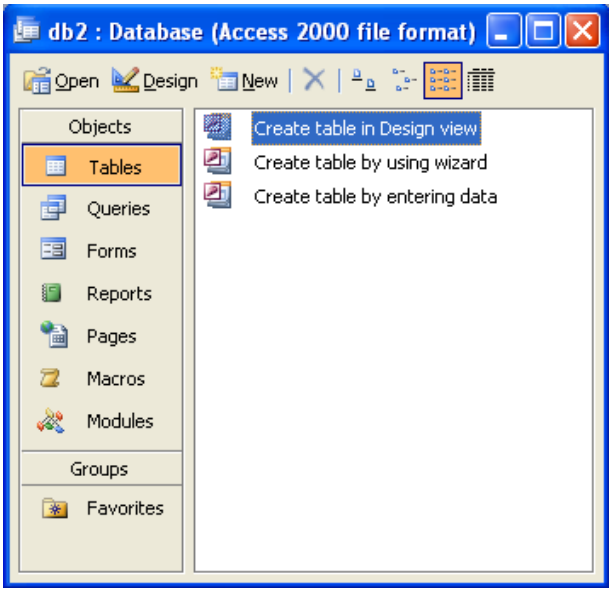

Figura 2. Crearea unui tabel nou

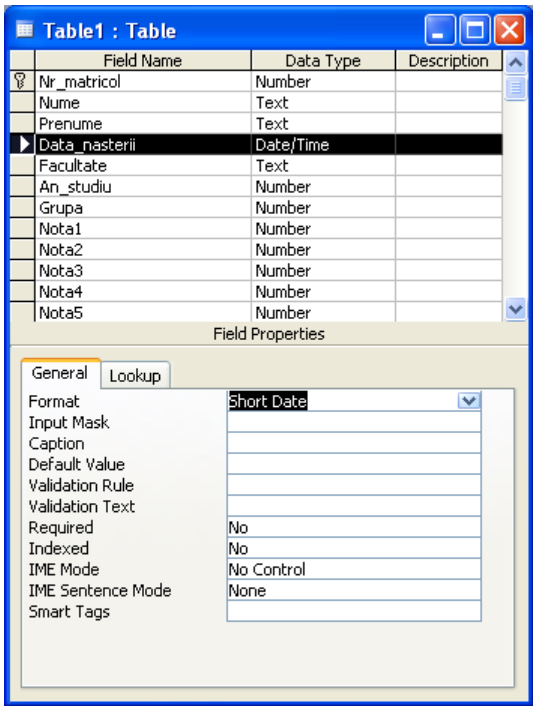

Figura 3. Structura tabelului Studenţi

În continuare se crea o aplicaţie Visual Basic .NET de tip Windows Forms Application care va conţine o fereastră de dimensiuni fixe precum cea din Figura 1.

Fereastra va conţine un meniu care va implementa acţiunile din Figura 4 şi o bară de unelte care va permite navigarea printre înregistrările din baza de date şi apelarea rapidă a acţiunilor din meniu.

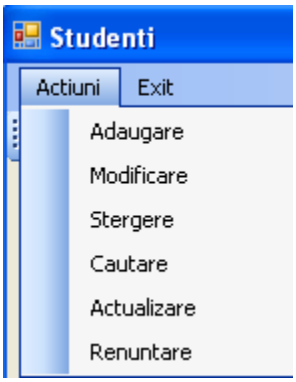

Figura 4. Meniul ferestrei principale.

Pe fereastră se vor mai adăuga: un control BindingNavigator, o componentă BindingSource, 12 etichete (Label) cu numele celor 12 câmpuri din tabelul Studenti, iar alăturat lor, 12 controale după cum urmează:

- Nr_matricol: TextBox.
- Nume: TextBox.
- Prenume: TextBox
- Data_nasterii va fi de tipul DateTimePicker (se vor seta proprietăţile Format = Short, ShowUpDown = True).
- Facultate: ComboBox, având valorile posibile AC, ETC, MEC, CT şi CI.
- An_studiu şi Grupa vor fi de tipul ComboBox, având ca valori posibile: 1, 2, 3, 4.
- Nota 1, Nota 2, Nota 3, Nota 4, Nota 5: ComboBox având valori posibile: 5, 6, 7, 8, 9, 10.

Controlul BindingNavigator vă pune la dispoziţie comenzi pentru parcurgerea înregistrărilor din baza de date. Obiectele de tip BindingSource încapsulează datele din baza de date la care vă legaţi prin intermediul aplicaţiei şi oferă funcţii pentru manipularea acestora. BindingNavigator va naviga printre datele din BindingSource, iar controale de pe formular (cum ar fi casetele de text şi ComboBox-urile) vor fi legate la acest BindingSource.

În continuare se doreşte editarea codului sursă al programului. În Visual Studio.NET lucrul cu baze de date se poate realiza utilizând tehnologiile OLEDB sau ADO.

OLEDB (Object Linking and Embedding, Database) este un set de interfeţe proiectat de Microsoft pentru accesarea a diferite tipuri de date stocate într-o manieră uniformă. A fost proiectată ca un înlocuitor de un nivel mai înalt, fiind de fapt succesorul lui ODBC, extinzându-i caracteristicile. OLEDB separă printr-un set de abstractizări datele de aplicaţia care are nevoie să le acceseze. OLEDB este divizată la nivel conceptual între consumatori şi furnizori. Consumatorii sunt aplicaţiile care au nevoie să acceseze datele iar furnizorul este componenta software care implementează interfaţa şi drept urmare care redă datele consumatorului.

ADO (ActiveX Data Objects) reprezintă un set de obiecte utilizate pentru accesarea surselor de date. ADO furnizează un nivel de abstractizare între client şi interfeţele OLEDB. ADO permite dezvoltatorului să scrie programe care accesează datele fără să ştie cum este implementată baza de date. Sunt necesare cunoştinţe privind baza de date doar pentru conectare. Nu sunt necesare cunoştinţe de SQL pentru a accesa baza de date, deşi se poate folosi ADO pentru a executa comenzi SQL.

ADO este bazat pe OLEDB, cele două nu sunt separate, nu sunt tehnologii distincte. OLEDB, fiind la un nivel mai jos decât ADO, este mai rapid. ADO prezintă un subset din capabilităţile OLEDB şi abstractizează mult din funcţionarea OLEDB.

Dezvoltarea aplicației utilizând OLEDB

Pentru a avea acces uşor la obiectele OLEDB este necesară importarea spaţiului de nume OLEDB:

```
Imports System.Data.OleDb
```

Pentru dezvoltarea aplicaţiei sunt necesare:

```
Dim  cale_BD As String = "d:\Work"
Dim string_conectare As String = "Provider=Microsoft.Jet.OLEDB.4.0; Data source=" + _
```

```
 cale_BD + "\BD.mdb;User ID=admin;Password=;"

Dim conexiune As OleDbConnection
Dim sql As String = "SELECT * FROM Studenti"
Dim comanda As New OleDbCommand
Dim adaptor As OleDbDataAdapter
Dim construire_comenzi As OleDbCommandBuilder
Dim ds As DataSet
```

Obiectul de tip OleDbConnection, permite conectarea la baza de date prin intermediul string-ului de conectare:

```
conexiune = New OleDbConnection(string_conectare)
conexiune.Open()
```

Obiectul de tip OleDbCommand, permite efectuarea unei interogări asupra bazei de date, dar şi executarea unor comenzi SQL NonQuery cum ar fi INSERT, UPDATE, DELETE etc.:

```
comanda.Connection = conexiune
comanda.CommandText = sql
```

Un DataSet reprezintă o copie în memorie a datelor extrase dintr-o sursă de date. Un DataSet constă dintr-o colecţie de tabele (DataTable), un tabel dintr-o colecţie de înregistrări (DataRow), iar o înregistrare dintr-o colecţie de câmpuri sau coloane (DataColumn).

```
ds = New DataSet("tabel")
ds.Tables.Add("tabel")
```

Clasa OleDbAdapter reprezintă adaptorul de servicii utilizate pentru a efectua interogări asupra surselor de date OLE DB. OleDbAdapter reprezintă o punte între un DataSet şi o sursă de date (cum ar fi o tabelă a unei baze de date) pentru extragerea datelor şi salvarea lor. Adaptorul furnizează această punte prin utilizarea metodei *Fill* pentru a încărca datele din sursa de date (tabela) într-un DataSet, respectiv *Update* pentru a trimite schimbările făcute în DataSet înapoi în sursa de date.

```
adaptor = New OleDbDataAdapter(comanda)
adaptor.Fill(ds.Tables("tabel"))
```

Un obiect OleDbCommandBuilder generează automat comenzile necesare pentru ca schimbările făcute într-un tabel dintr-un DataSet să se reflecte şi în baza de date din care a fost încărcat DataSet-ul. Pentru ca un adaptor sa poată aplica schimbările făcute într-un DataSet asupra bazei de date (metoda *Update*), este necesară asocierea lui cu un OleDbCommandBuilder. Asocierea dintre obiectul OleDbCommandBuilder şi adaptor se poate face prin intermediul constructorului:

```
construire_comenzi = New OleDbCommandBuilder(adaptor)
```

În continuare urmează specificarea sursei de date a obiectului BindingSource, realizarea legăturii dintre controale din formular şi BindingSource, precum şi dintre BindingNavigator şi BindingSource.

```
BindingSource1.DataSource = ds.Tables("tabel")
BindingSource1.Position = 0

'legarea proprietăţii Text a casutei txtMatricol la campul Nr_matricol
txtMatricol.DataBindings.Add(New Binding("Text", BindingSource1,_
 "Nr_matricol", True))
'in mod similar se va proceda si cu celelalte controale...
BindingNavigator1.BindingSource = BindingSource1
```

Până în acest punct aplicaţia va permite vizualizarea înregistrărilor din baza de date şi navigarea printre aceste înregistrări.

Aplicaţia va avea 2 stări de funcţionare, care vor fi implementate în continuare:

- prima stare în care se va putea naviga printre înregistrările din baza de date, acestea putând fi doar vizualizate nu şi modificate. Butonul de Adăugare, va fi activ, iar cele de Salvare şi Renunţare vor fi inactive. Butoanele Editare, Ştergere şi Căutare vor fi active doar dacă există înregistrări în baza de date. (Figura 5)
- a doua stare se instaurează atunci când se apasă Adăugare sau Editare. Se vor dezactiva controalele de navigare şi comenzile Adăugare, Editare, Ştergere, Căutare, singurele acţiuni posibile fiind Salvare sau Renunţare (Figura 6).

Media notelor studentului curent va fi afişată în partea dreaptă a formularului.

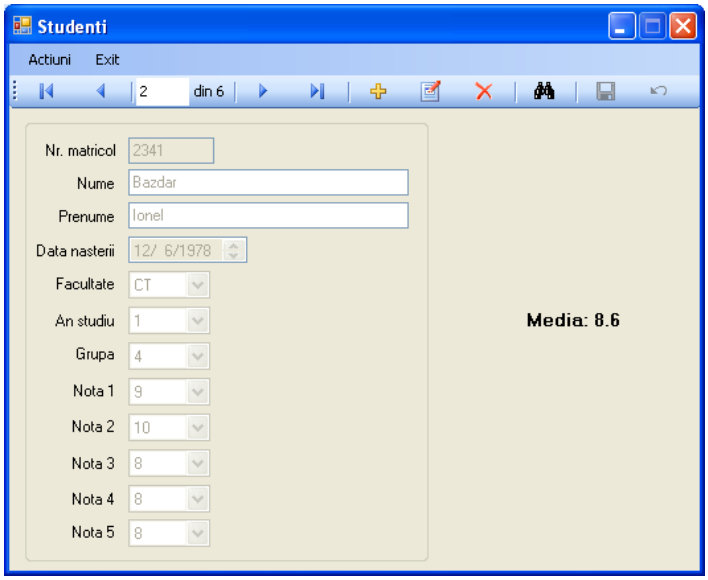

Figura 5. Starea 1

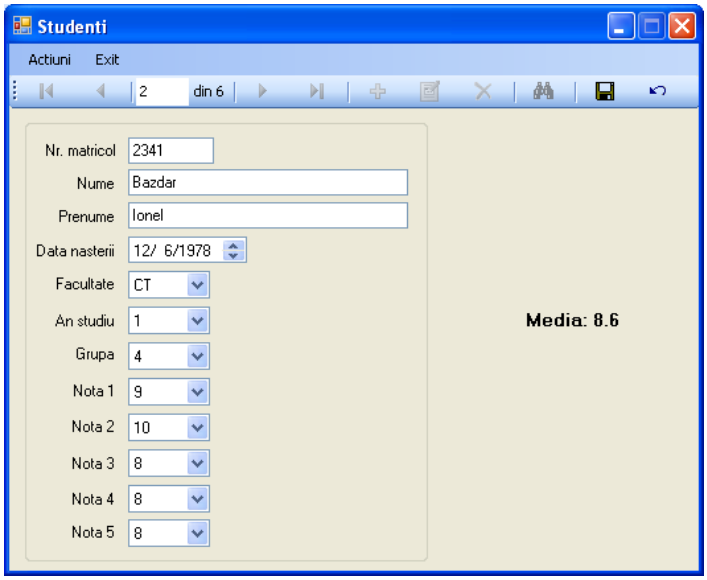

Figura 6. Starea 2

În continuare se vor implementa funcţionalităţile controalelor Adăugare, Editare, Ştergere, Căutare, Salvare şi Renunţare:

- Adăugare – butonul de adăugare poate să rămână cel implicit al obiectului BindingNavigator, sau se poate crea un buton nou. La apăsarea butonului adăugare se va trece din starea 1 în starea 2.
- Editare – singura acţiune care se va petrece la editare va fi cea de trecere din starea 1 în starea 2.
- Ştergerea nu va fi realizată fără o confirmare suplimentară, precum cea din Figura 7. Pentru a putea implementa aceasta funcţionalitate se va selecta BindingNavigator1 şi se va pune proprietatea *DeleteItem* pe None. Ştergerea se poate realiza prin comanda de mai jos şi va fi urmată de salvare:

```
BindingSource1.RemoveCurrent()
```

Figura 7. Confirmare ştergere

- Salvarea va presupune trecerea din Starea 2 în Starea 1 şi actualizarea modificărilor făcute în DataSet, în tabela considerată:

```
BindingSource1.EndEdit()
adaptor.Update(ds.Tables("tabel"))
```

- Renunţarea presupune trecerea din Starea 2 în Starea 1 şi refacerea DataSet prin reîncărcarea datelor din baza de date:

```
ds.Clear()
adaptor.Fill(ds.Tables("tabel"))
```

În continuare se va implementa fereastra pentru căutare:

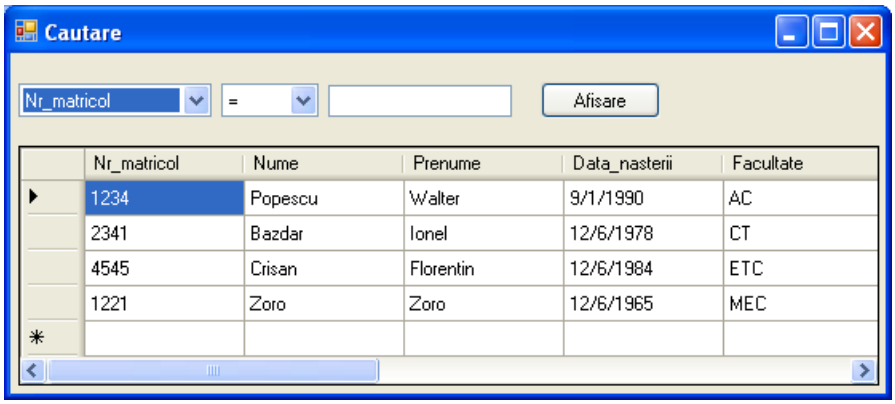

Figura 8. Căutarea după un câmp

Figura 9. Valori luate de cele două controale ComboBox în vederea efectuării unei căutări

Fereastra va conţine un control DataGridView, care iniţial va afişa toate înregistrările din baza de date, iar în urma apăsării butonului Afişare se vor afişa în DataGridView doar înregistrările care respectă condiţiile precizate ("câmp = valoare" sau „câmp like valoare").

Pentru încărcarea DataGridView cu toate datele din tabela bazei de date se va proceda în mod similar exemplului de mai sus cu menţiunea că obiectele de tip BindingNavigator, BindingSource şi OleDbCommandBuilder nu mai sunt necesare.

Pentru încărcarea datelor din DataTable în DataGridView se va proceda în felul următor:

```
DataGridView1.DataSource = ds.Tables("tabel")
DataGridView1.ReadOnly = True
```

Iniţial DataGridView va conţine înregistrările din întreaga tabelă. Interogarea SQL va fi construită dinamic, astfel încât dacă se apasă butonul afişare şi nu este introdus nimic în câmpul pentru valoare, în DataGridView va fi afişat conţinutul întregii tabele, iar altfel se vor afişa în DataGridView doar înregistrările filtrate prin interogare. În cazul în care interogarea SQL nu este corectă se va afişa un mesaj de eroare, urmat de afişarea comenzii în clar. Pentru aceasta se va folosi blocul Try…Catch.

Sfaturi utile

1. *Pentru a asigura comutarea între cele două stări complementare se recomandă crearea unei funcţii de dezactivare/activare a controalelor, care are ca şi parametru o variabilă boolean, corespunzătoare stării.*
2. *Pentru includerea unei confirmări suplimentare la operaţia de ştergere, se recomandă renunţarea la acţiunea implicită pe care o are butonul de ştergere în cadrul BindingNavigator-ului şi editarea manuală a codului necesar ştergerii.*
3. *În fereastra de căutare, pentru completarea numelor tuturor câmpurilor din tabelul Studenti în controlul ComboBox se recomandă parcurgerea prin intermediul unei bucle For…Each a tuturor coloanelor din tabelul creat şi adăugarea prin cod a denumirilor acestora.*

Salvarea în tabelă a modificărilor făcute se poate realiza şi fără utilizarea unui OleDbCommandBuilder şi a metodei Update a adaptorului, prin construirea dinamică a comenzii SQL (INSERT, DELETE, UPDATE) şi executării ei, la fel ca în exemplul de mai jos:

```
'se construieste interogarea insert, update, sau delete
command.CommandText = sql
command.ExecuteNonQuery()
```

Cu ce ne-am ales?

Prin aplicaţia dezvoltată am învăţat să realizăm aplicaţii desktop pentru lucrul cu baze de date şi să implementăm principalele operaţii care apar în aplicaţiile cu baze de date: adăugare, modificare, ştergere sau căutare. Aplicaţiile cu baze de date sunt utile şi necesare în toate domeniile. Aplicaţia dezvoltată operează asupra unei baze de date Access dar tehnologia OLEDB poate fi aplicată şi altor baze de date precum Oracle, Microsoft SQL Server, MySQL, etc.

Bibliografie

[1] http://msdn.microsoft.com/en-us/vbasic/default.aspx