

TEHNICI DE PROGRAMARE AVANSATA – Curs 1

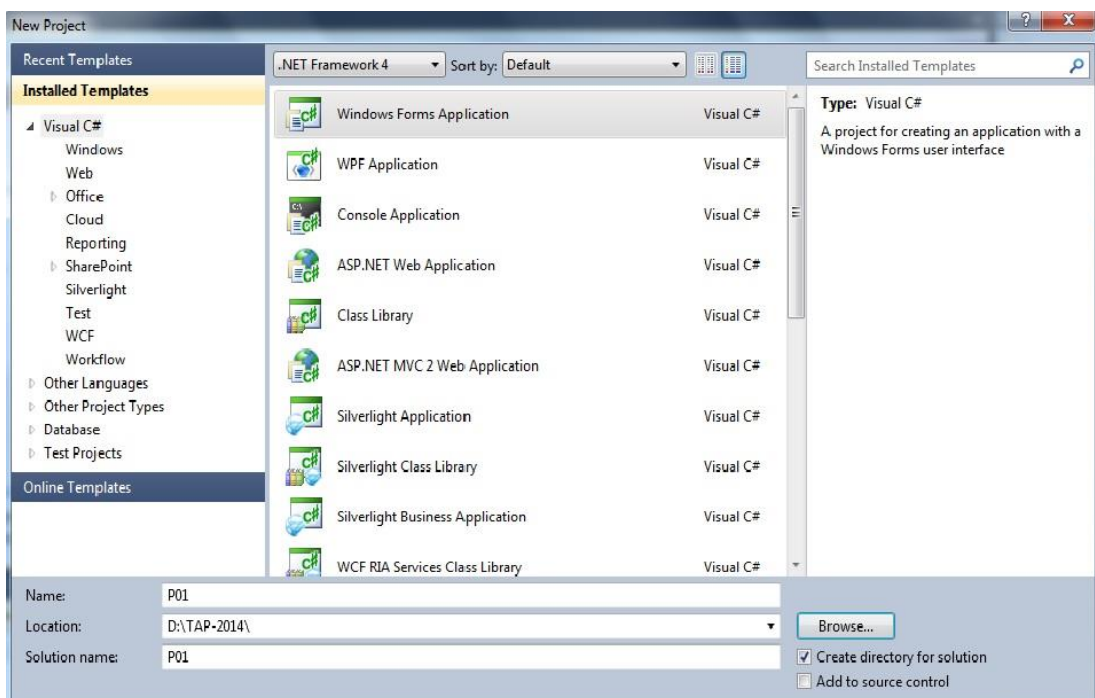
- Platforma Microsoft Visual Studio
- Tehnologia ADO.NET
- Aplicatie 1: vizualizarea datelor dintr-o tabela a unei baze de date MS ACCES
- Aplicatie 2: vizualizarea unor date ce implica mai multe tabele ale unei baze de date MS ACCESS

Platforma Microsoft Visual Studio



⇒ **New Project**

Limbaaj de programare Visual C#



Tehnologia Windows Forms

In aceasta tehnologie o aplicatie consta dintr-o multime de ferestre grafice (clase) ce mostenesc clasa predefinita Form.

Dintre ferestre se alege una, numita fereastra de start, ce va fi afisata in momentul executiei aplicatiei.

Fereastra de start este specificata in clasa Program.

Tehnologia ADO.NET

ADO.NET ne permite sa utilizam urmatoarea arhitectura pentru aplicatiile de tip Windows Forms:



ADO.NET este tehnologia Microsoft pt. conectarea aplicatiilor la o sursa de date externa (baze de date, fisiere XML, ect).

In imaginea de mai sus, sunt prezentate o serie de clase predefinite, rolul central fiind detinut de clasa DataSet.

Aplicatie 1: vizualizarea datelor dintr-o tabela a unei baze de date MS ACCESS

Analiza aplicatiei

Vrem sa realizam o aplicatie care sa contina o ferestra grafica cu o imagine tabelara continand date despre persoane: Nume, CNP, Salariu. Datele sunt stocate intr-o baza de date Access, in tabela Persoane.

Aplicatia nu permite actualizarea datelor.

Aplicatia sa permita sortarea persoanelor dupa nume sau cnp sau salariu.

Fereastra mai contine un buton cu textul *Refresh*, care prin apasare realizeaza refresh-ul imaginii tabelare.

Proiectarea aplicatiei


Arhitectura aplicatiei consta din:

- o clasa ce defineste ferestra grafica
- o clasa ce defineste datele necesare aplicatiei (independent de baza de date)
- o baza de date Access cu o tabela Persoane avand schema: Persoane (IdPersoana, Nume, CNP, Salariu)

Implementarea aplicatiei

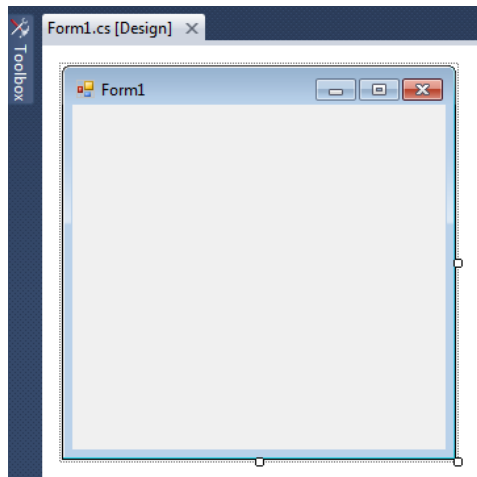
Pas 1. Se creaza o baza de date Access noua (DB-01.accdb) in directorul de lucru.

Se creaza tabela Persoane (IdPersoana, Nume, CNP, Salariu) si se completeaza cu date.

Persoane	
Field Name	Data Type
 IdPersoana	AutoNumber
Nume	Short Text
CNP	Short Text
Salariu	Currency

Persoane			
IdPersoana	Nume	CNP	Salariu
2	Ionescu Vasile	1801123131210	1,500.00 lei
3	Vasilescu Doina	2840516131211	1,700.00 lei
4	Popa Lucian Vasile	1981230131214	1,700.00 lei
6	Popvici Violeta	2950621131215	1,000.00 lei
10	Popa Dana	2940521131216	1,800.00 lei
29	Ivascu Dan	1930415131217	1,500.00 lei
32	Popescu Eugen	1920313131218	2,300.00 lei

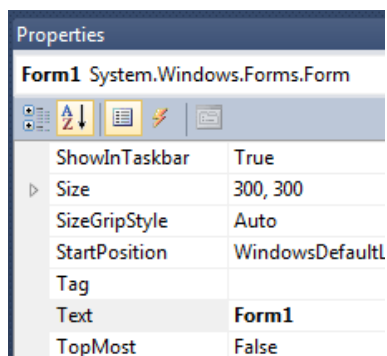
Pas 2. Se creaza o noua aplicatie cu numele Curs-01 in directorul de lucru si o clasa Form1.



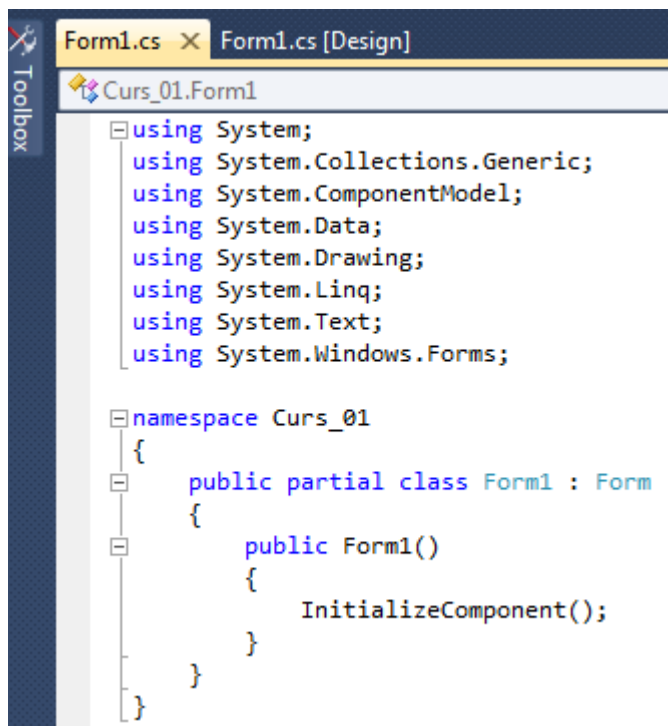
Imaginea reprezinta un instrument (designer) pt. proiectarea ferestrei grafice.

Componentele ferestrei (obiecte) pot fi preluate din fereastra Toolbox (stanga sus) vizibila doar in modul designer.

Obiectele ferestrei au o serie de attribute ale caror valori pot fi setate in fereastra Properties (dreapta jos).



Codul clasei poate fi vizualizat prin click dreapta pe spatial liber al Form-ulu si selectia optiunii View Code sau F7.



Visual Studio ne-a creat automat clasa Form1 ce mosteneste clasa predefinita Form.

In C# exista conceptul de clasa partiala ce ne permite sa despartim codul unei clase in mai multe fisiere sursa (cu extensia cs), ce vor fi unite la momentul compilarii.

Visual Studio genereaza automat cod in timp ce lucram cu *designer-ul*, intr-un fisier distinct numit in cazul lui Form1, Form1.Designer.cs. Aceste fisiere pot fi vizualizate si editate (nu este recomandat).

Codul scris de noi il plasam in Form1.cs.

Codul clasei se compune astfel dintr-o parte generata in totalitate de sistem (Form1.Designer.cs) si o parte generata initial tot de sistem dar editata ulterior de noi (Form1.cs).

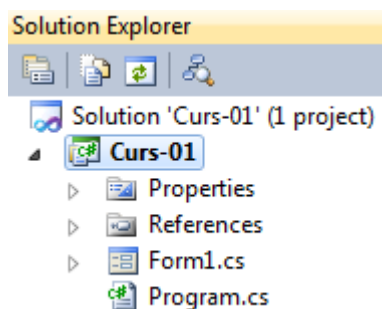
Observam ca in Form1.cs ne-a generat automat si constructorul.

Apar o serie de directive using ce specifica asa numitele spatii de nume (similar package-urilor din Java) ce contin clasele necesare programului.

Ni s-a creat si un spatiu de nume avand acelasi nume cu cel al form-ului.

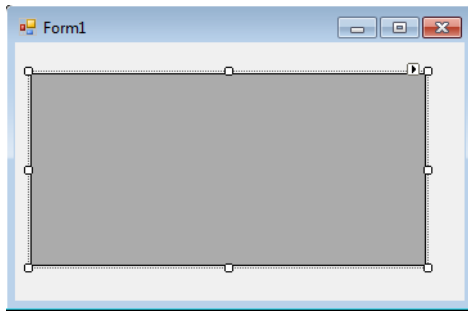
Nu este recomandat sa modificam acest cod initial.

Arhitectura aplicatiei poate fi vazuta in fereastra Solution explorer (dreapta sus):

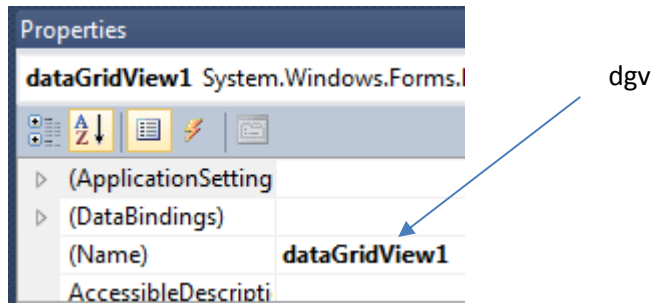


Pas 3. Plasam pe form un control de tip DataGridView:

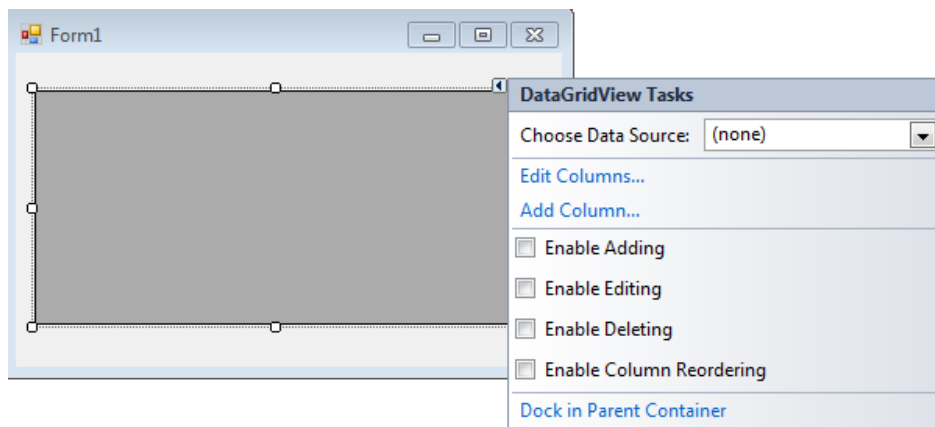
Toolbox => Data => DataGridView => Dublu click (sau drag and drop)



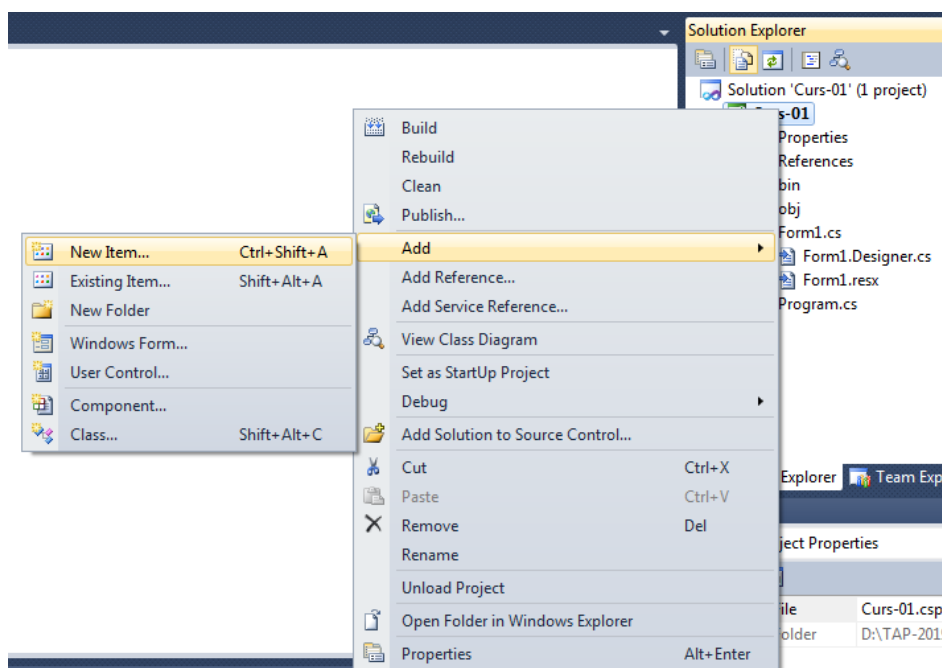
Se modifica proprietatea Name a controlului DataGridView1 -> DGV.



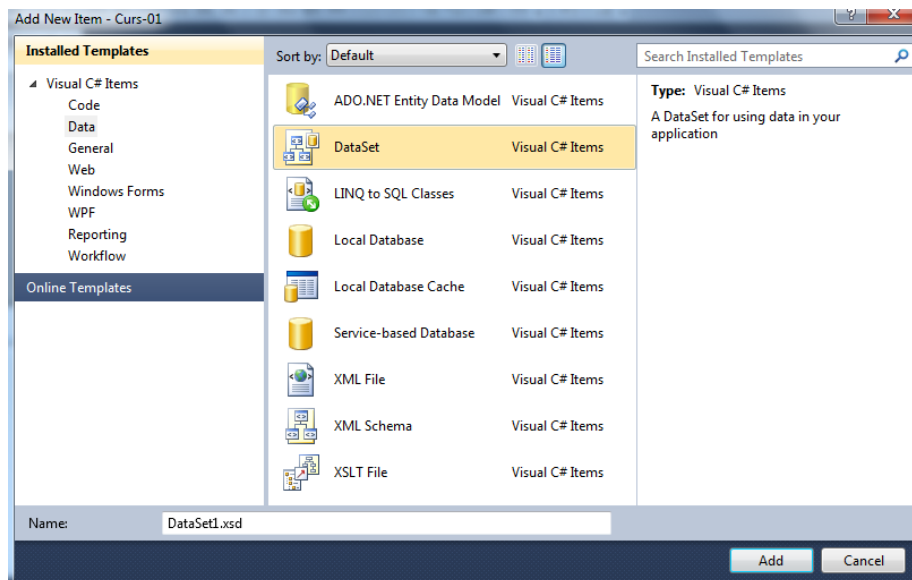
Se modifica proprietatile Enable Adding, Enable Editing, Enable Deleting pe False (se debifeaza) in fereastra DataGridView Tasks. Astfel grid-ul nu va permite actualizari.



Pas 4. Se adauga aplicatiei, o component DataSet:

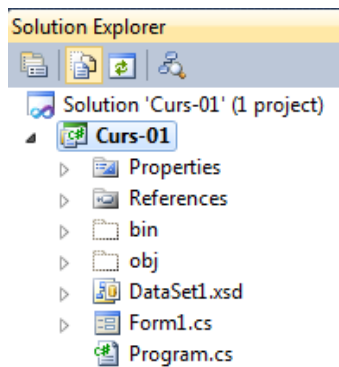


Click-dr pe numele aplicatiei (Curs-01) in fereastra Solution Explorer) => Add => New Item



Putem modifica numele componentei DataSet, in caseta Name. Il vom lasa asa, nu este esential pt. acest exemplu.

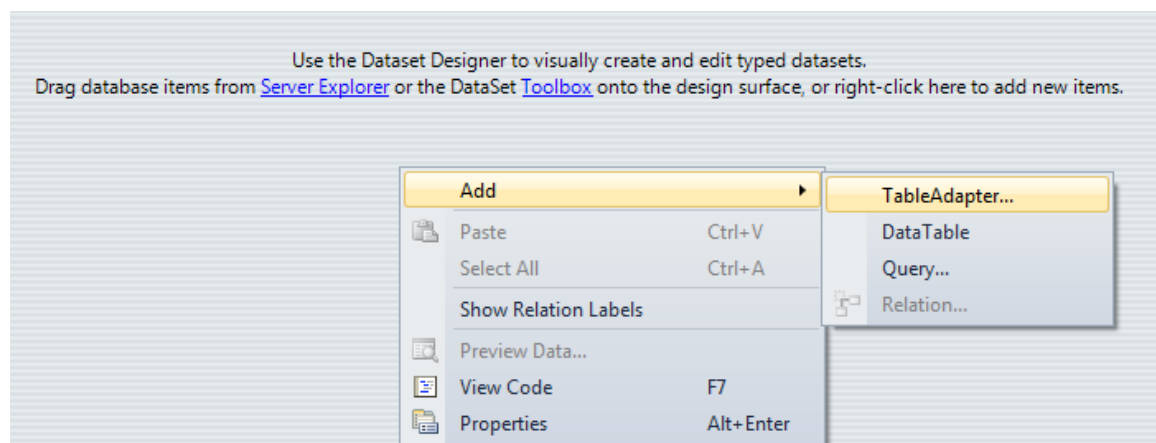
In fereastra Solution explorer apare aceasta noua componenta (DataSet1).



Se deschide totodata un nou *Designer* (DataSet designer) ce ne permite sa configuram DataSet-ul.

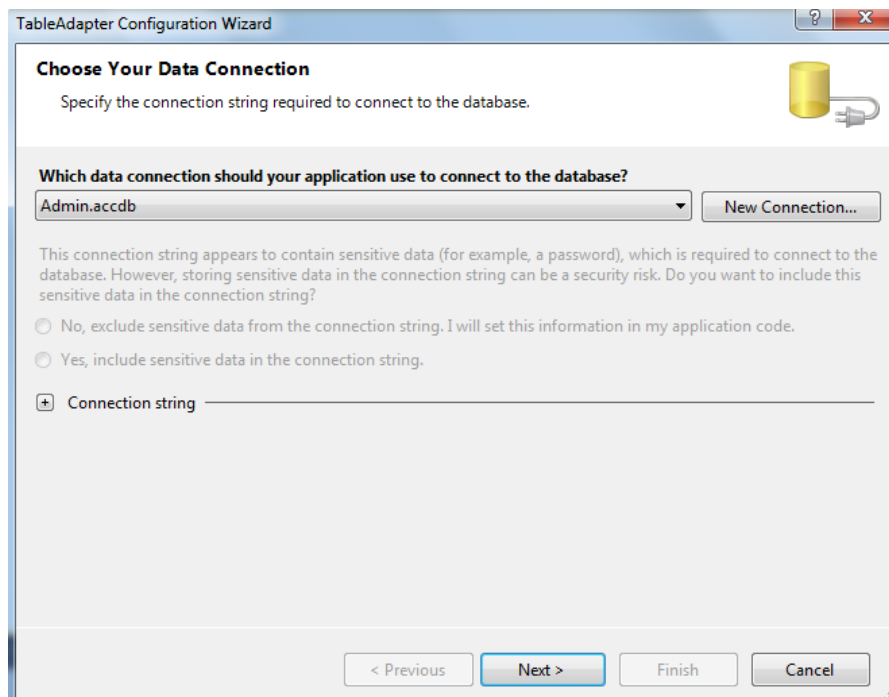
Pas 5. Configuram DataSet-ul

Click-dr pe spatial liber al DataSet-ului.



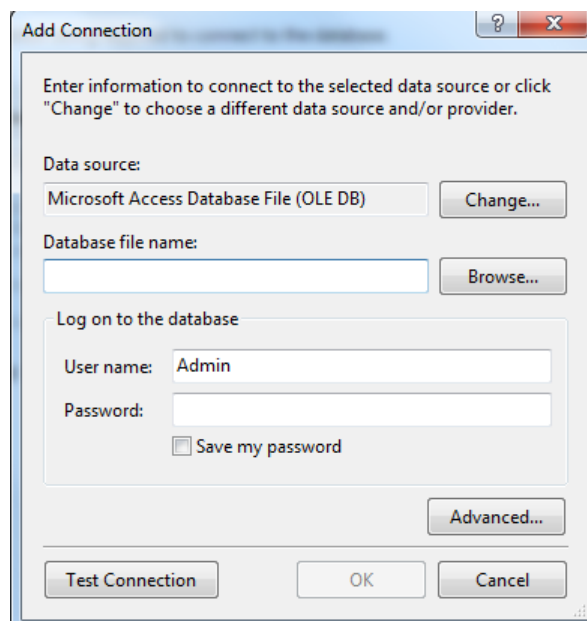
Add => TableAdapter

Se activeaza un asistent (wizard) pt. configurarea unui TableAdapter



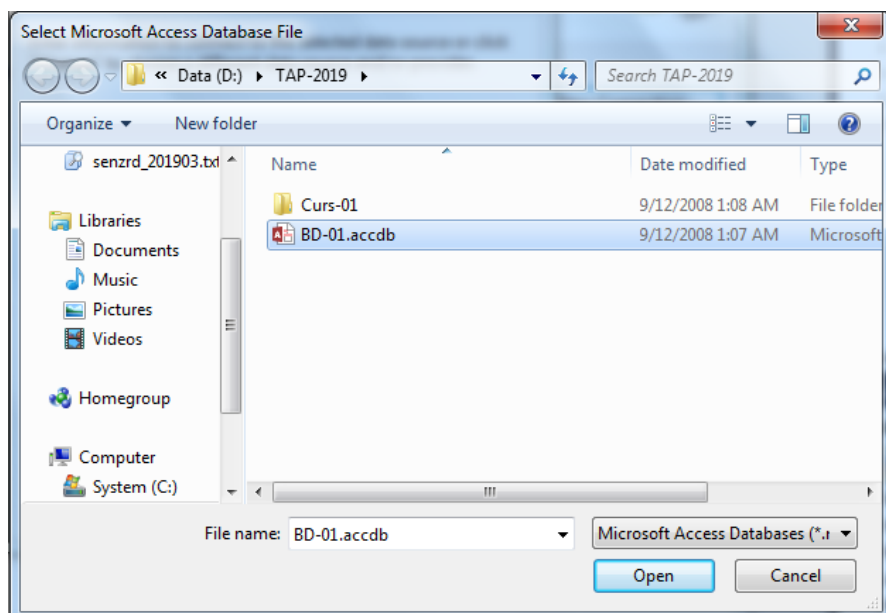
TableAdapter-ul este o componenta care face legatura intre o baza de date si un DataTable din DataSet.

Declaram baza de date. Click pe butonul New Connection.

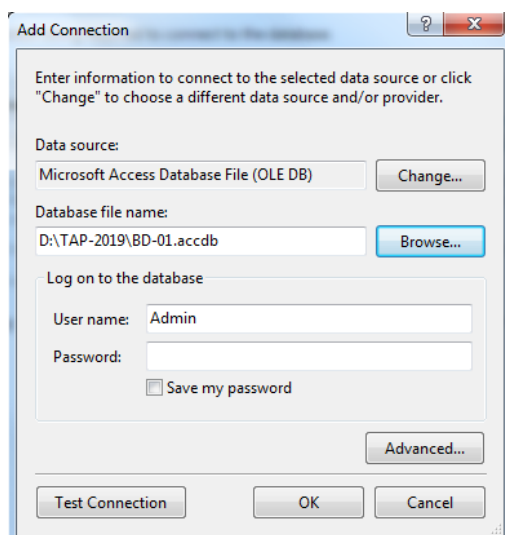


Tipul bazei de date este specificat in caseta Data source. Implicit este Microsoft Access. Daca avem un alt tip de baza de date, facem click pe butonul *Change*. Lasam tipul implicit.

Click pe butonul Browse, pt. a selecta baza de date.

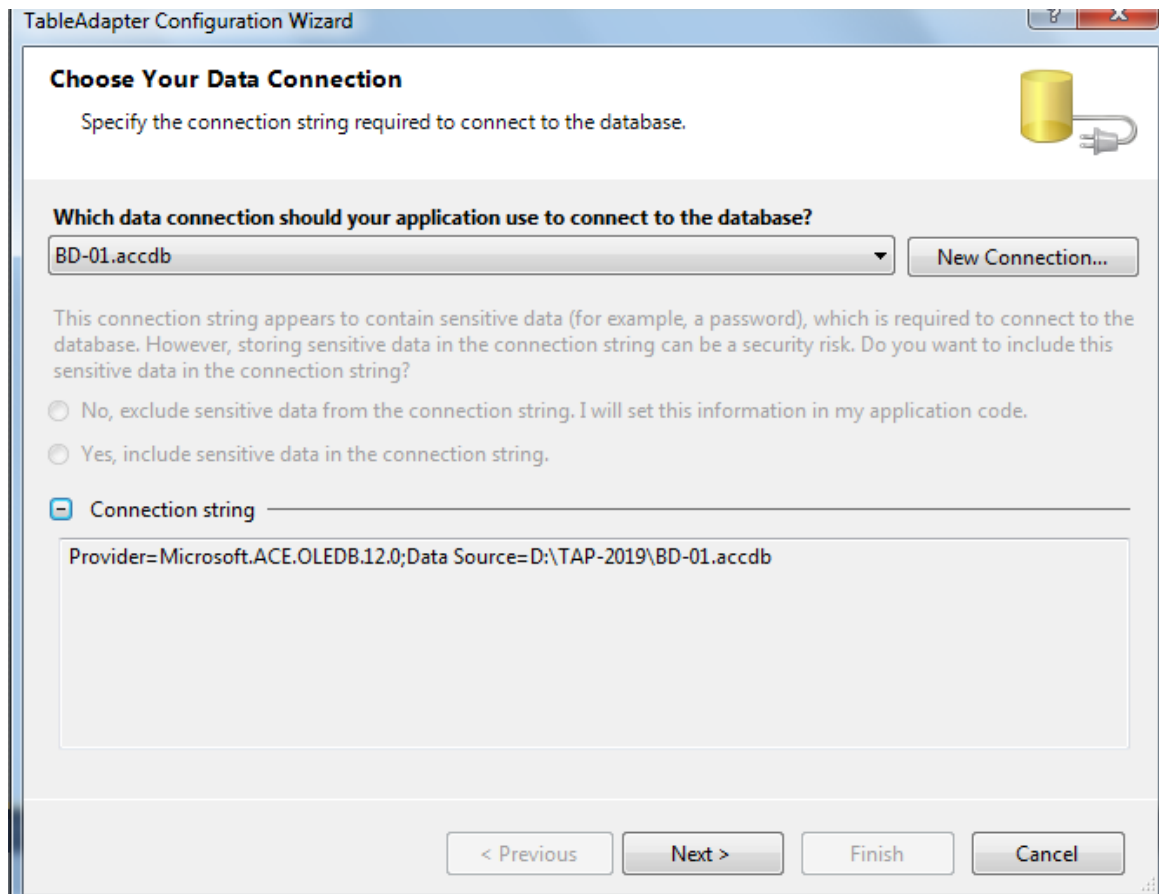


Selectam baza de date si facem click pe butonul *Open*.



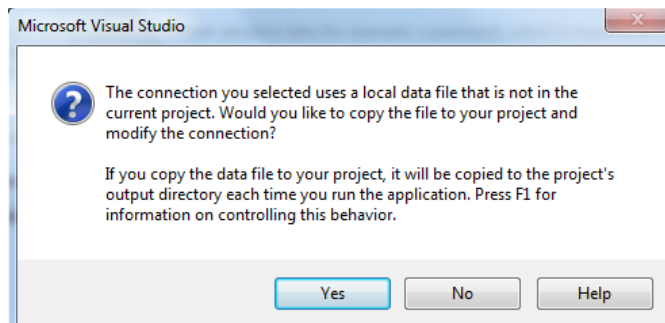
Putem testa conexiunea prin butonul Test Connection.

Apasam butonul OK.

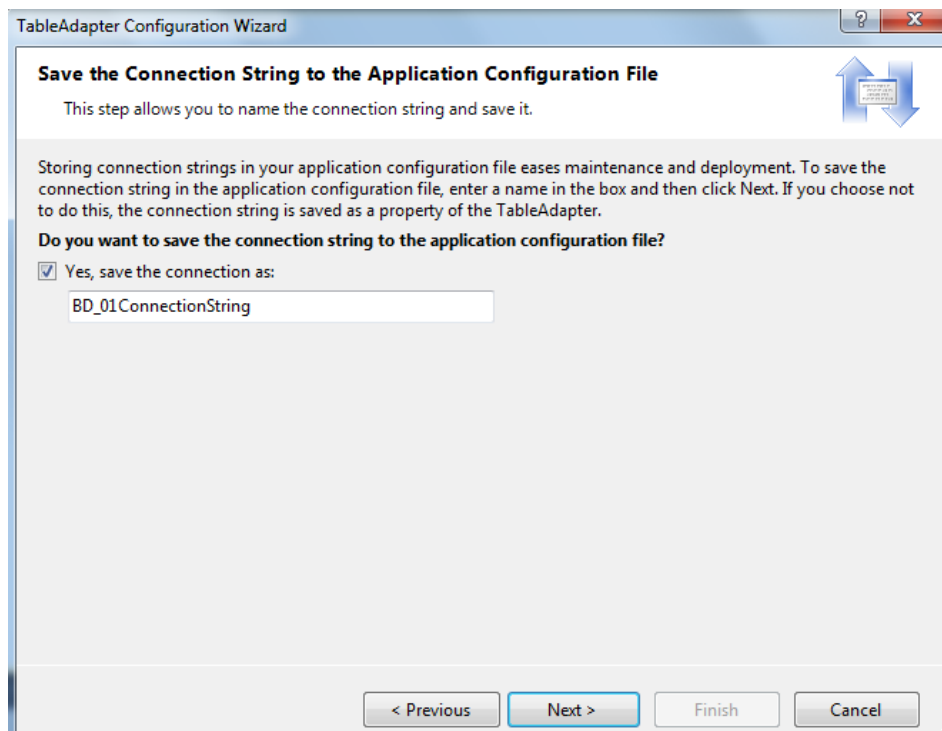


Verificam sirul de conectare (Connection string) Click pe butonul + din dreapta casetei Connection string.

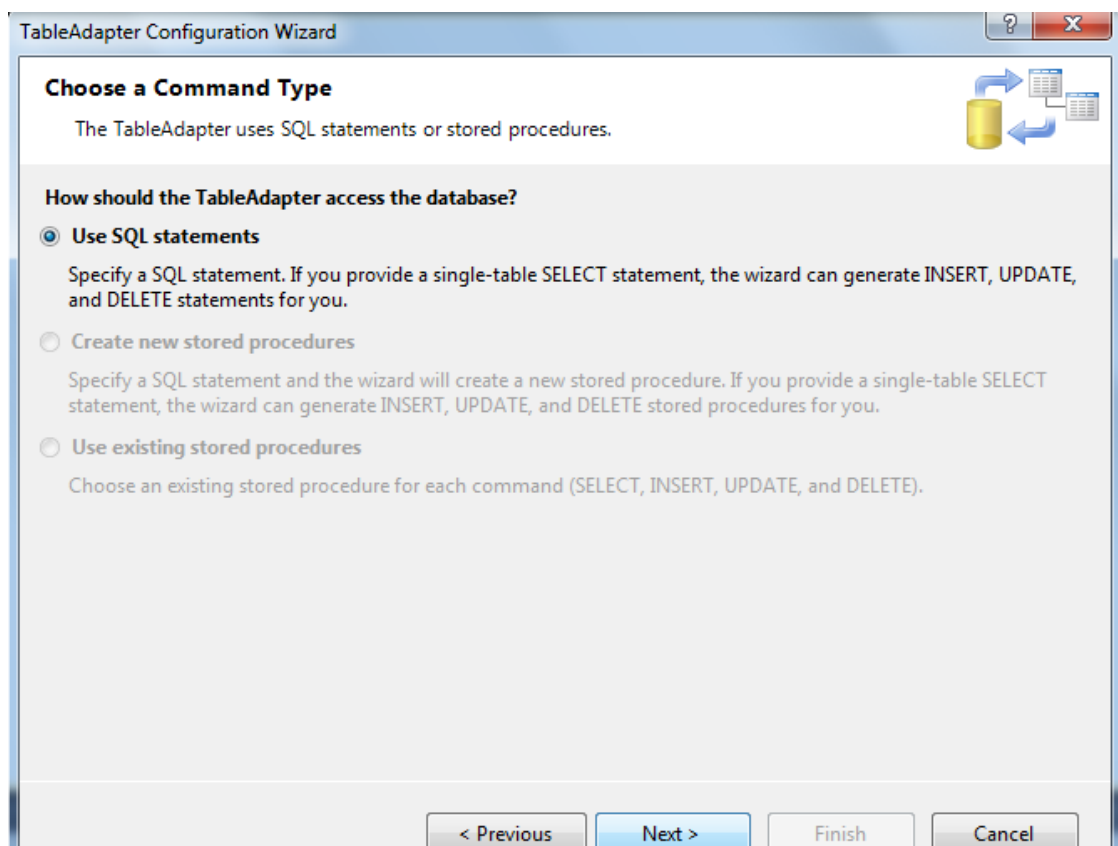
Click pe butonul Next.



Aceasta fereastră ne propune sa ne creeze o copie a bazei de date in cadrul aplicatiei si sa lucreze cu aceasta copie. Solicitam optiunea No.



Urmatoarea fereastră ne întreabă dacă vrem să ne salvăm șirul de conectare (Connection string) în fișierul de Configurare al aplicației. Acceptăm și apăsăm butonul Next.

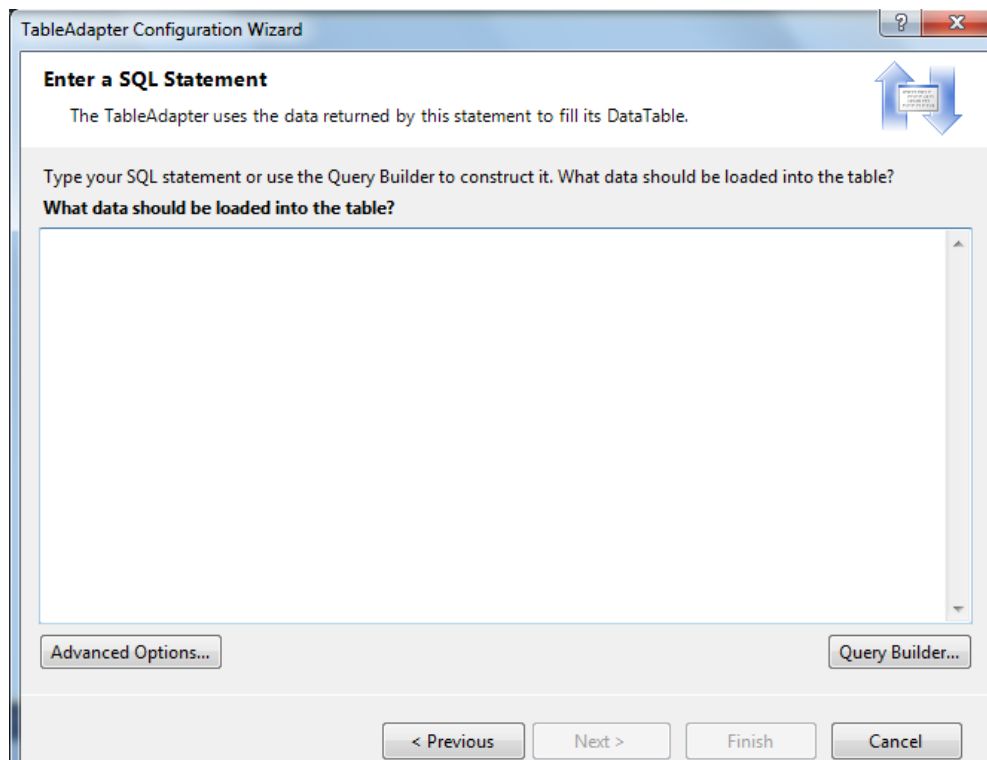


Urmatoarea fereastră ne propune să alegem modul de accesare a bazei de date. Lasăm modul implicit *Use SQL statements*.

Urmatoarea fereastră ne solicită cererea SQL care să ne furnizeze datele necesare.

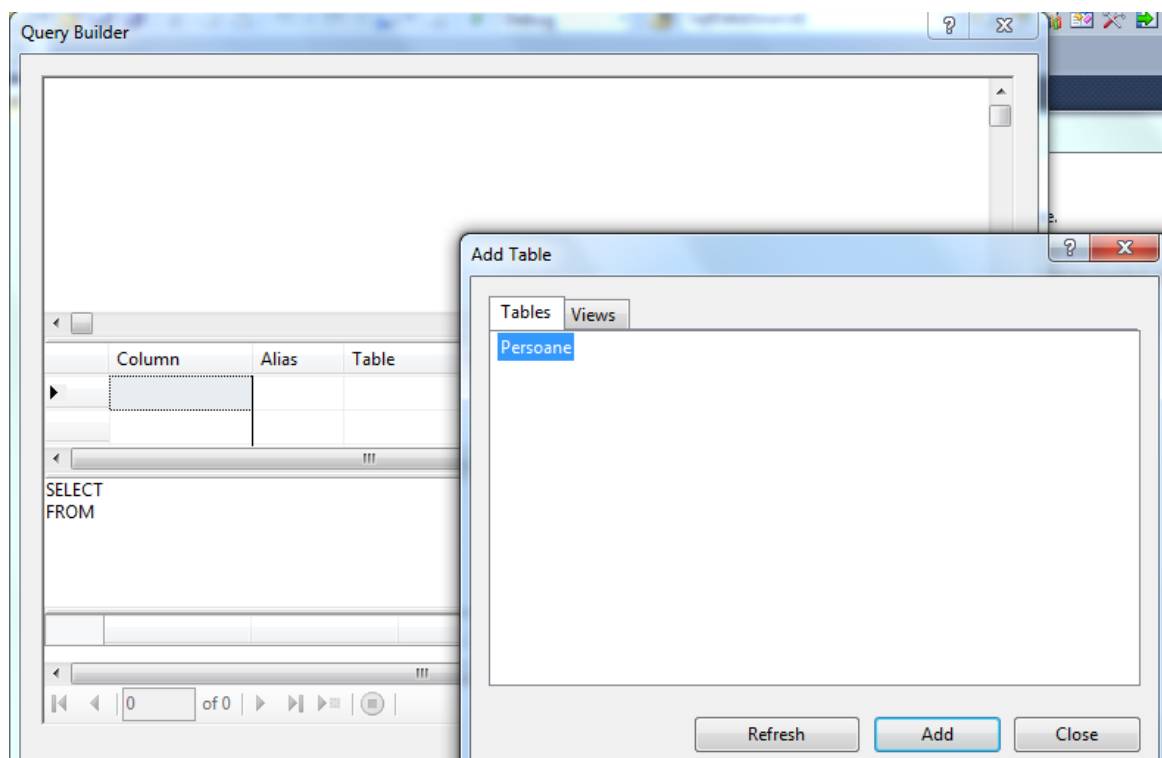
Putem specifica cererea SQL în fereastră de editare, sau putem solicita opțiunea Query Builder.

Vom solicita Query Builder.

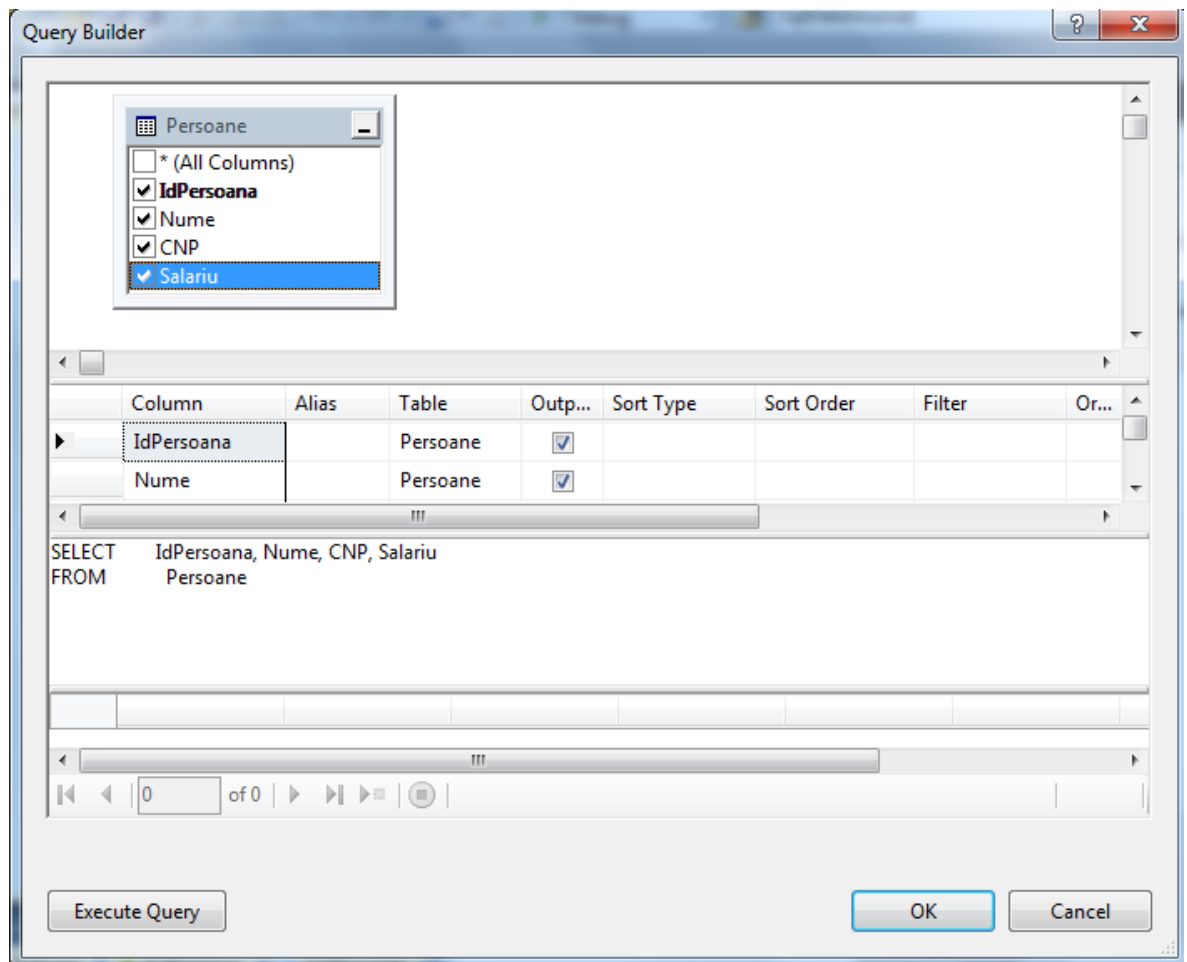


Se deschide o fereastră *Add Table* prin care solicităm tabelele implicate în cerere.

În cazul nostru solicităm *Add* după care închidem fereastra *Add Table* (butonul *Close*).

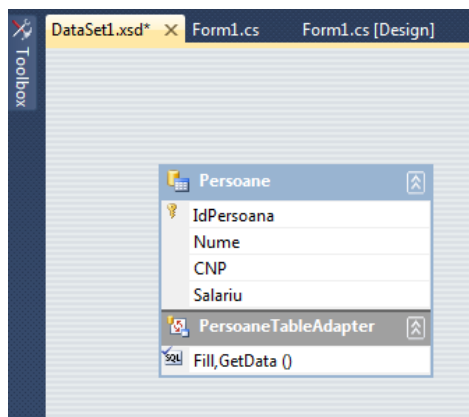


În fereastra *Query Builder* bifăm toate atributele tabelii *Persoane*. Observăm cum ne generează cererea SQL. Putem interveni și aici în fereastra de editare a cererii SQL sau să completăm tabela din fereastra. Putem de asemenea să testăm execuția cererii.



Apasam ok apoi Next, Next si Finish.

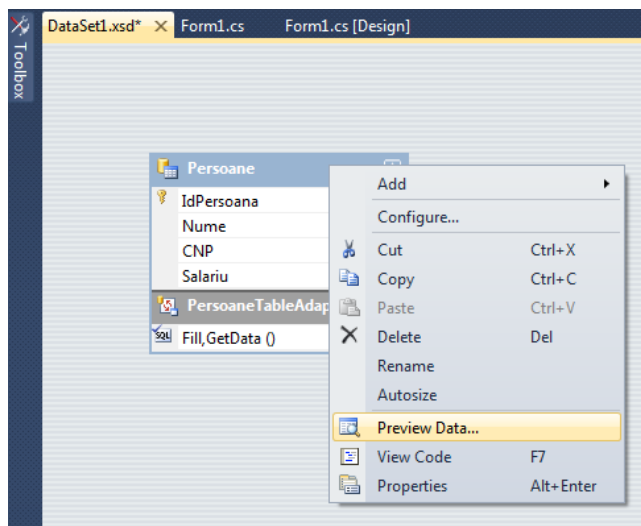
Pas 6. Observam ca sistemul ne genereaza un DataTable in cadrul DataSet-ului. Numele DataTable-ului este Persoane.



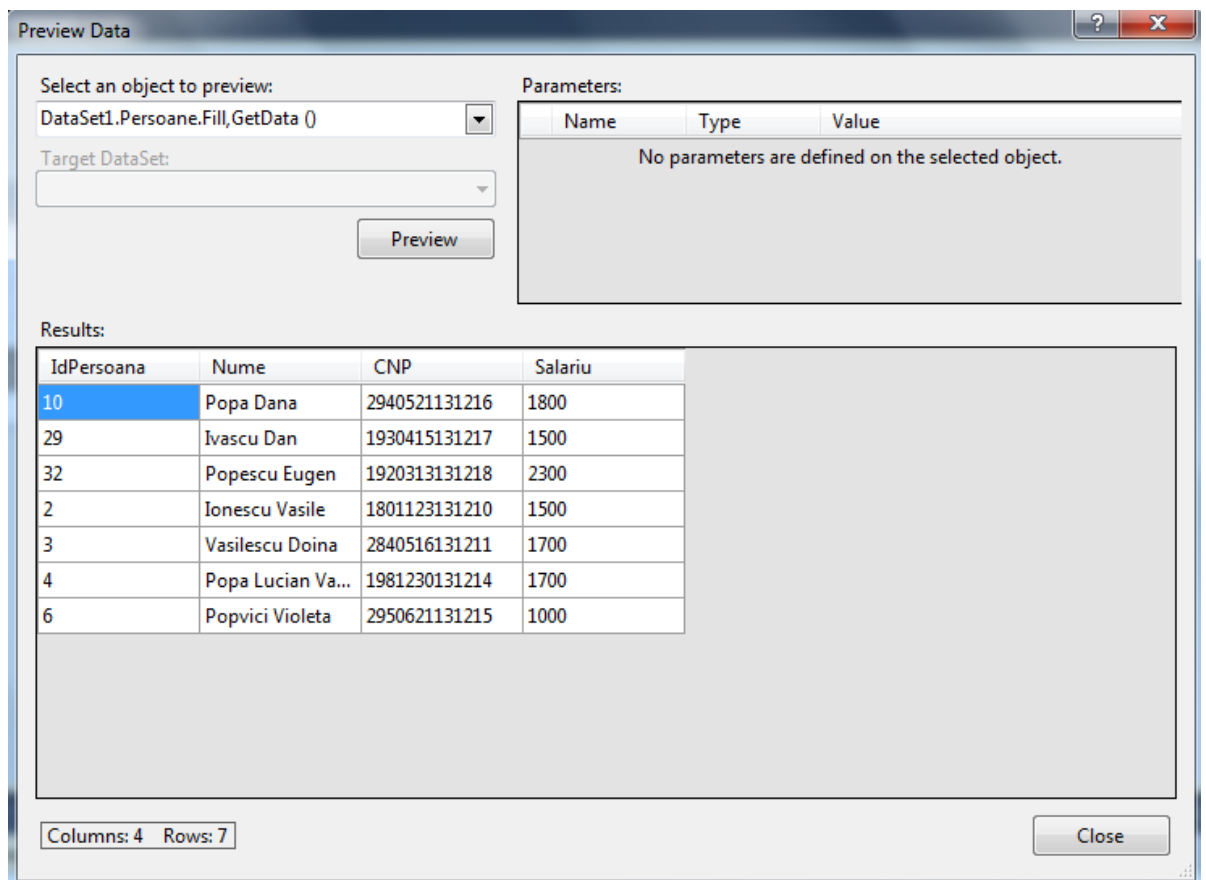
DataTable-ul are schema corespunzatoare cererii SQL.

Putem testa la nivelul DataSet-ului functionarea TableAdapterului (PersoaneTableAdapter)

Click dr pe antetul DataTable-ului.



Alegem Preview Data



Si apoi click pe butonul Preview.

Este important sa observam ca testarea accesului la baza de date se poate face la nivelul DataSet-ului.

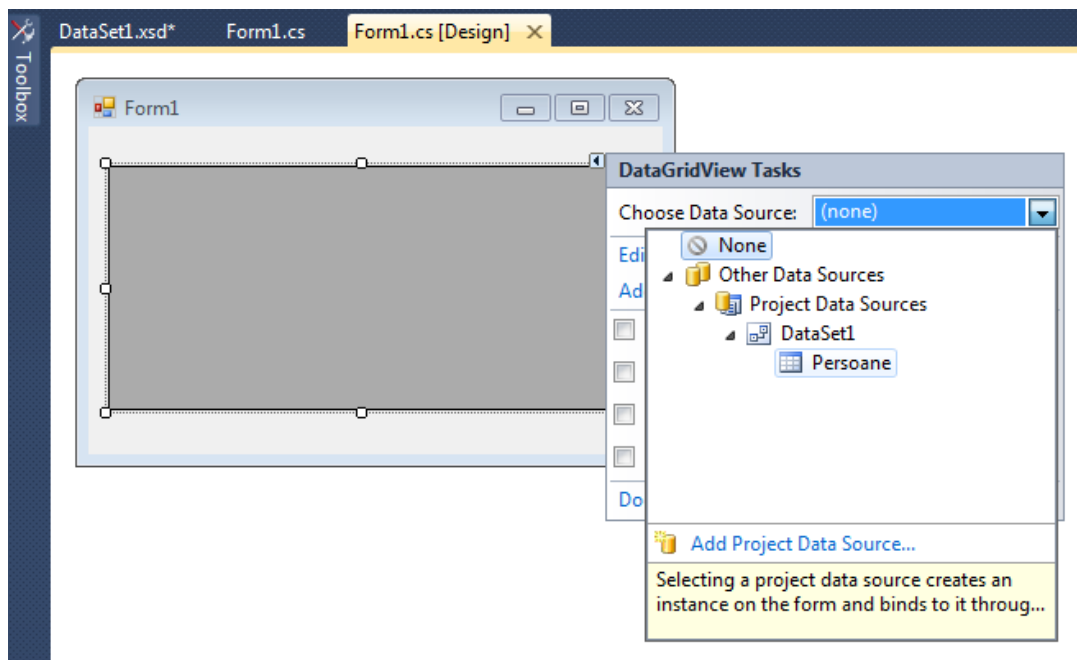
Pas 7. Revenim la nivelul form-ului si stabilim sursa de date pt. grid.

Selectam tab-ul Form1.cs [Design]

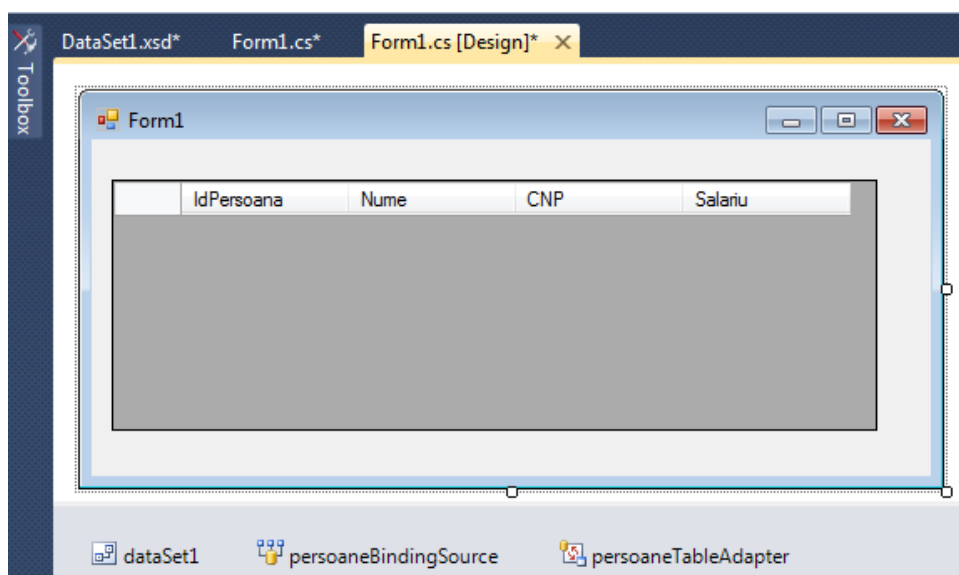
Deschidem fereastra DataGridView Tasks a grid-ului.

Deschidem comboBox-ul Choose Data Source.

Expandam Other Data Sources => Project Data Sources => DataSet1 => Click Persoane



Obtinem:



Observam ca grid-ul a fost structurat conform DataTable-ului Persoane din DataSet.

Observam ca s-au inclus 3 obiecte noi: dataSet1, persoaneBindingSource, persoaneTableAdapter, pe care le putem utiliza in program.

Daca vizualizam din nou codul atasat form-ului, observam aparitia unei metode eveniment Form1_Load, care se activeaza la deschiderea form-ului. Aceasta metoda contine instructiunea:

```
this.persoaneTableAdapter.Fill(this.dataSet1.Persoane);
```

care apeleaza metoda fill a obiectului persoaneTableAdapter ce are ca parametru DataTable-ul ce trebuie incarcat.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
namespace Curs_01
```

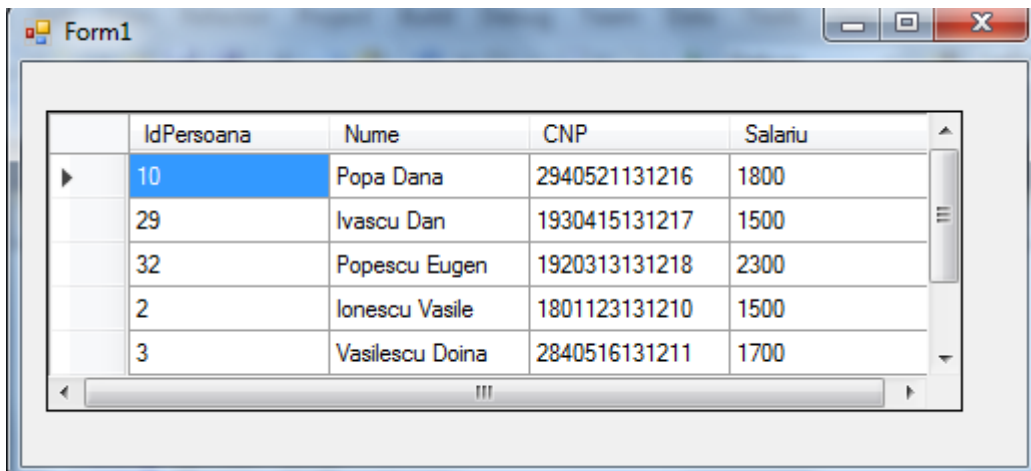
```

{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'dataSet1.Persoane' table. You
            // can move, or remove it, as needed.
            this.persoaneTableAdapter.Fill(this.dataSet1.Persoane);
        }
    }
}

```

Pas 8. Executam aplicatia: CTRL-F5.



Pas 9. Adaugarea butonului Refresh

Intram pe Form1.cs [Design].

Aducem din ToolBox o component de tip Button si o plasam pe form.

Modificam proprietatea Text a butonului cu textul *Refresh*.

Dublu click pe buton.

Intram pe codul form-ului.

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Curs_01
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'dataSet1.Persoane' table. You can
            // move, or remove it, as needed.

```

```

        this.persoaneTableAdapter.Fill(this.dataSet1.Persoane);
    }

    private void button1_Click(object sender, EventArgs e)
    {
    }
}

```

Observam ca s-a generat metoda eveniment button1_click ce se va activa la apasare.

Completam codul de mai jos:

```

private void button1_Click(object sender, EventArgs e)
{
    dataSet1.Persoane.Clear();
    this.persoaneTableAdapter.Fill(this.dataSet1.Persoane);
}

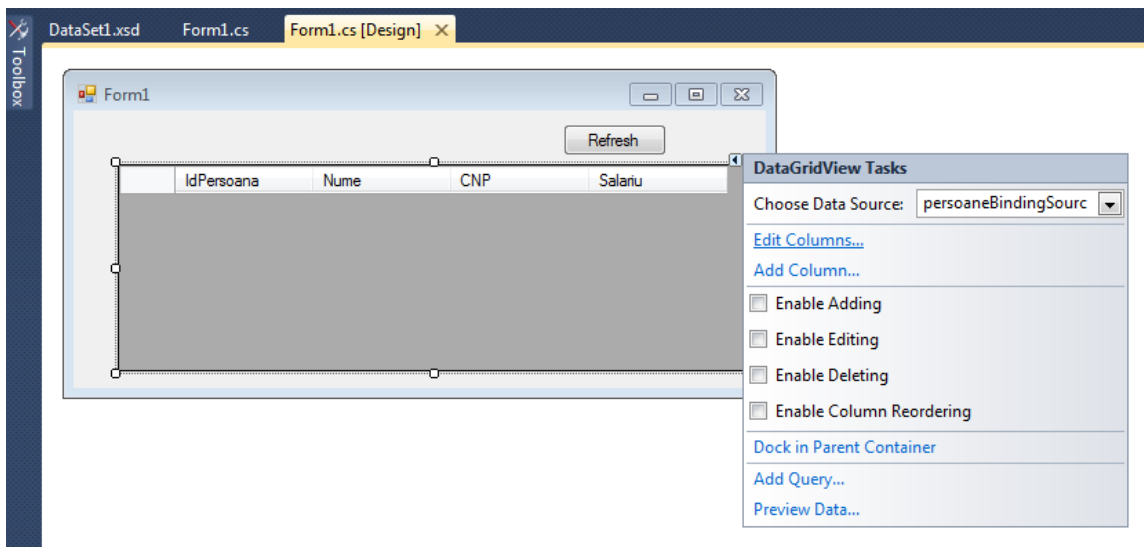
```

El contine golirea dataTable-ului si incarcarea lui.

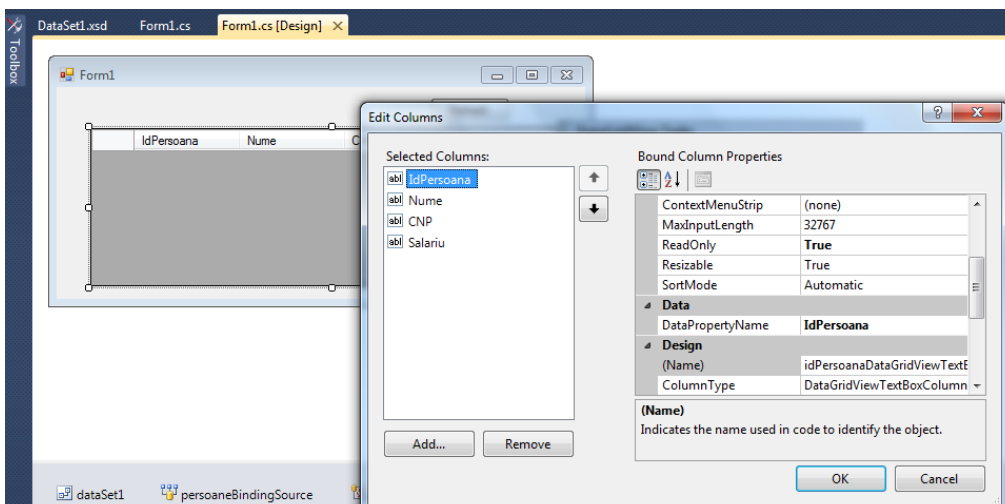
Putem testa refresh-ul executand aplicatia, apoi sa facem o modificare in tabela (cu Access-ul) apoi apasam butonul refresh. Aplicatia trebuie sa afiseze datele modificate.

Pas 10 Eliminam din grid coloana IdPersoana

Intram pe DataGridView Tasks



Selectam Edit Columns



Selectam coloana pe care dorim s-o eliminam (IdPersoana este deja selectata) si apasam butonul Remove.

Atentie! Coloana este eliminata doar din grid. Din dataTable-ul Persoane (din DataSet) a ramas.

O coloana eliminata din grid poate fi reintrodusa (tot cu EditColumns).

Cu EditColumns se pot modifica antet-ul coloanelor, latimea coloanelor, ordinea coloanelor, etc.

Pas 11 Sortarea liniilor din grid

Este implementata implicit prin obiectul DataGridView.

Se face click pe antetul coloanei dupa care se face sortarea.

Se poate obtine sortare crescatoare sau descrescatoare.

Aplicatie 2: vizualizarea unor date ce implica mai multe tabele ale unei baze de date MS ACCESS

Analiza aplicatiei

Vrem sa realizam o aplicatie care sa contina o ferestra grafica cu o imagine tabelara continand date despre persoane si localitatile de domiciliu: Nume, CNP, Salariu, Localitate. Datele sunt stocate intr-o baza de date Access, in tabelele Persoane1 si Localitati..

Aplicatia nu permite actualizarea datelor.

Aplicatia sa permita sortarea persoanelor dupa nume sau cnp, salariu sau Localitate.

Fereastra mai contine un buton cu textul *Refresh*, care prin apasare realizeaza refresh-ul imaginii tabelare.

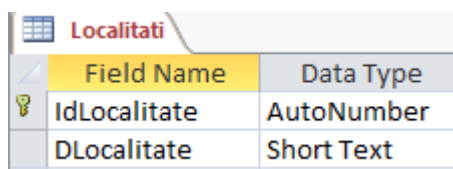
Proiectarea aplicatiei

Arhitectura aplicatiei consta din:

- o clasa ce defineste ferestra grafica
- o clasa ce defineste datele necesare aplicatiei (independent de baza de date)
- o baza de date Access cu 2 tabele
 - Persoane1 (IdPersoana, Nume, CNP, Salariu, IdLocalitate)
 - Localitati(IdLocalitate, DLocalitate)

Implementarea aplicatiei


Pas 1. Se modifica baza de date folosita in prima aplicatie.



Field Name	Data Type
IdLocalitate	AutoNumber
DLocalitate	Short Text



IdLocalitate	DLocalitate
1	Constanta
2	Bucuresti
3	Craiova
4	Braila
5	Tulcea
6	Mangalia
7	Medgidia
8	Cluj

Persoane1	
Field Name	Data Type
 IdPersoana	AutoNumber
Nume	Short Text
CNP	Short Text
Salariu	Currency
IdLocalitate	Number

Persoane1				
IdPersoana	Nume	CNP	Salariu	IdLocalitate
2	Ionescu Vasile	1801123131210	1,500.00 lei	1
3	Vasilescu Doina	2840516131211	1,700.00 lei	2
4	Popa Lucian Vasile	1981230131214	1,700.00 lei	3
6	Popvici Violeta	2950621131215	1,000.00 lei	4
10	Popa Dana	2940521131216	2,800.00 lei	5
29	Ivascu Dan	1930415131217	1,500.00 lei	6
32	Popescu Eugen	1920313131218	2,300.00 lei	3

Pas 2. Cream un form nou, Form2:

Click dr pe L01 (Fereastra Solution Explorer) => Add => New Item => Windows Form;

Apasam apoi butonul Add;

Setam proprietatea Text a noului form -> Persoane cu Localitati.

Pas 3. Aduugam un nou obiect DataTable la DataSet1:

Click dr DataSet1.xsd (din Solution explorer (Curs-01) => View Designer

Click dr pe spatiul liber => Add => Table Adapter => Next => Next => Query Builder

Aducem in spatiul Query Builder cele doua tabele: Persoane1 si Localitati.

Bifam campurile IdPersoana, Nume, CNP, Salariu de la Persoane si DLocalitate de la Localitati.

Schimbam tipul de join: click dr pe legatura dintre tabele => Select all rows from Persoane

Se obtine query-ul sql:

```
SELECT    Persoane1.IdPersoana, Persoane1.Nume, Persoane1.CNP, Persoane1.Salariu, Localitati.DLocalitate
FROM      (Localitati RIGHT OUTER JOIN
           Persoane1 ON Localitati.IdLocalitate = Persoane1.IdLocalitate)
```

Putem deasemenea verifica executia query-ului apasand butonul Execute query.

Apasam OK si apoi Finish.

Observam aparitia unui nou obiect DataTable1 cu campurile solicitate si cu adapterul corespunzator.

Schimbam din proprietatile acestui obiect numele in PersoaneLoc.

Pas 4. Revenim pe form-ul Form2.

Plasam un control DataGridView pe care-l numim dgv.

Deschidem fereastra DataGridView Tasks => Choose Data Source => Deschidem Other Data Sources =>

Deschidem Project Data Sources => Deschidem DataSet1 => Selectam PersoaneLoc.

Observam ca s-a structurat grid-ul cu coloanele dorite.

Observam ca s-au adaugat si de data aceasta cele 3 obiecte: dataSet1, persoaneLocBindingSource, persoaneLocTableAdapter.

Pas 5. Declaram form-ul Form2 ca form de start. Modificam codul in Program.cs.

```
Application.Run(new Form2());
```

Executam aplicatia cu CTRL-F5.

Pas 6. Schimbam antetul de coloana: DLocalitate -> Localitate.

In fereastra DataGridView Tasks => Edit Columns => Header Text => OK

Eliminam din grid coloana IdPersoana